Titre: Title:	Estimating Distances Between Ellipsoids with Machine Learning
Auteur: Author:	Gaston César Banna
Date:	2023
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Banna, G. C. (2023). Estimating Distances Between Ellipsoids with Machine Learning [Mémoire de maîtrise, Polytechnique Montréal]. PolyPublie. https://publications.polymtl.ca/53456/

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Estimating Distances between Ellipsoids with Machine Learning

GASTON CÉSAR BANNA
Département de mathématiques et de génie industriel

Mémoire présenté en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
Mathématiques appliquées

Mai 2023

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Ce mémoire intitulé :

Estimating Distances between Ellipsoids with Machine Learning

présenté par **Gaston César BANNA**
en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
a été dûment accepté par le jury d'examen constitué de :

Luc ADJENGUE, président
Marc LAFOREST, membre et directeur de recherche
Serge PRUDHOMME, membre et codirecteur de recherche
Charles BÉRUBÉ, membre

DEDICATION

This work is dedicated to my beloved grandparents, whose unconditional love and support have been my guiding light throughout my academic journey.

Their unwavering faith in my abilities and their constant encouragement have been the driving force behind my pursuit of knowledge. I am forever grateful to them.

ACKNOWLEDGEMENTS

I want to express my heartfelt appreciation to my advisors, Prof. Marc Laforest and Prof. Serge Prudhomme, for their invaluable guidance and unwavering support throughout my academic journey. Their expertise and encouragement have been instrumental in shaping the direction and quality of my work. Their constant feedback and constructive criticism have helped me develop rigorous thinking and reach my full potential. I am truly inspired by their patience and dedication to my academic growth.

I would like to thank all the study participants and individuals, who contributed their time to this project, Guillaume Choquet in particular, as well as the faculty and staff of the Mathematics Department MAGI for creating a stimulating academic environment and providing me with the resources necessary to conduct this research. I would like to express my gratitude to the members of the jury, Dr. Luc Adjengue and Dr. Charles Bérubé.

Finally, my family and friends have been a constant source of encouragement and motivation throughout my academic journey, and I am deeply grateful for their love and support.

RÉSUMÉ

Ce mémoire présente une nouvelle approche pour générer un ensemble de données et entraîner un réseau neuronal capable d'estimer la distance entre des paires d'ellipsoïdes. Cette approche s'inspire du cadre théorique développé dans des travaux antérieurs, qui a fourni une base mathématique pour générer des données relatives à des particules ellipsoïdales, analyser et améliorer les algorithmes de détection de contact pour les paires d'ellipsoïdes.

La procédure de génération de l'ensemble des données a révélé des lacunes dans les méthodes existantes. En particulier, les travaux antérieurs laissaient les biais observés dans l'ensemble de données sans interprétation et n'ont pas cherché à les éliminer. Afin de répondre à ces limitations, la méthodologie a été étendue en exploitant les propriétés bijectives des transformations utilisées dans le modèle pour produire des données bien plus configurables. En particulier, il est maintenant possible de choisir des paramètres représentatif du monde réel lorsque les statistiques sont mesurables.

Avec ces améliorations, un ensemble de données conséquent de paires d'ellipsoïdes a été généré en considérant un système de paramètres uniformément distribués et en documentant les points de contacts entre les ellipsoïdes. L'ensemble de données a ensuite été utilisé pour entraîner un réseau neuronal à estimer la distance entre lesdites paires d'ellipsoïdes.

Le réseau neuronal a été entraîné sur un sous-ensemble des données générées et testé sur un sous-ensemble distinct pour évaluer sa performance. Les résultats montrent que le réseau neuronal peut estimer de manière fiable l'ordre de grandeur de la distance entre les paires d'ellipsoïdes.

La méthode proposée offre une nouvelle approche pour créer des ensembles de données et entraîner des réseaux neuronaux pour les problèmes de détection de contact. Le cadre théorique révisé offre une compréhension plus approfondie du fonctionnement du système et permet la génération de données plus fiables et représentatives. Cela rend ainsi possible la création de modèles de réseau neuronal pour les problèmes de détection de contact qui sont plus fiables et précis.

ABSTRACT

This master's thesis presents a novel approach to generate a dataset and train a neural network that estimates the distance between pairs of ellipsoids. The approach builds on the theoretical framework developed in previous work, which provided a mathematical foundation for sampling data representing ellipsoidal particles, analyzing and improving contact detection algorithms for pairs of ellipsoids.

An earlier dataset generation procedure possessed significant shortcomings. In particular, the previous framework failed to interpret observed biases in the dataset and did not attempt to eliminate them. In order to address these limitations, the methodology was expanded by exploiting the bijective properties of the maps used in the model to produce data that is much more customizable (and indeed, indicative of real-world settings when appropriate).

With the help of this improved framework, a sizable dataset of ellipsoid pairs was created by modelling a uniformly distributed system and documenting the contacts between the ellipsoids. The dataset was then used to train a neural network to estimate the distance between pairs of said ellipsoids.

The neural network was trained on a subset of the generated data and tested on a separate subset to evaluate its performance. The results demonstrate that the neural network can reliably estimate the order of magnitude of the distance between pairs of ellipsoids.

The proposed method offers a novel approach to create datasets and train neural networks for contact detection problems. The revised theoretical framework offers a more thorough understanding of the workings of the system and enables the generation of more reliable and representative data. This work demonstrates that it is possible to create neural network models for contact detection issues that are more reliable and accurate, although the optimal form of this network has yet to be identified.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1 INTRODUCTION

1.1 Scientific Context

Mathematical models are used to describe the behavior of physical systems in both a quantitative and a qualitative manner. In the field of mechanics, such models can be used to understand the motion and deformation of particles under different loading conditions [1]. A particle can be defined as a discrete entity that has mass, volume, and shape, and can transfer forces through contact and compression with other particles and its environment. Mathematical models of particles can be broadly divided into two categories: continuum models and discrete models.

Continuum models describe the behavior of a continuous medium, such as a fluid or a solid, by treating it as a mathematical abstraction [2]. These models are based on the assumption that the material is homogeneous and isotropic, and that the averaged properties of the material do not vary significantly above a certain length scale. Continuum models are widely used in many engineering applications, including fluid dynamics, elasticity, and plasticity.

On the other hand, discrete models represent the behavior of a system by explicitly modeling individual particles or components of the system. These models are based on the assumption that the material is composed of individual particles that interact with each other and their environment and for which discrete physics dominates large-scale behavior. Discrete models are used in various applications such as granular materials, molecular dynamics, and discrete element simulations.

Discrete models can be further classified based on the shape of the particles. For instance, in the field of granular materials, particles are often modeled as spheres or ellipsoids. These models are based on the assumption that the particles interact with each other through contact forces, and that the shape of the particles affects their packing and deformation behavior [3, 4]. In contrast to continuum models, discrete models can capture the non-linear behavior of granular materials.

Most commonly, the Discrete Element Method (DEM) is a numerical method used to solve problems where each individual particle has its position, orientation, speed and momenta and surface features accounted for. It models each particle and their contacts individually, using Newtonian mechanics and Hookian or Hertzian contact models to compute the force between particles. DEM was first proposed in 1971 to

analyze rock mechanics problems using only spheres, and it accounts for the small displacement and rotation of particles while allowing them to overlap in order to account for their deformation [5]. DEM is commonly used to predict transitions in granular material and study mechanical properties of composite materials, cement matrices in concrete technology, and powder-based structures.

Spheres are the most common particle model for soil simulation due to the simplicity of their collision detection. However, spheres are different from real soil particles, in terms of shape, centers of mass, and hence, the manner by which one rotates around the other. Some studies demonstrate that the particle shape plays a key role in the simulated macroscopic properties of static and dynamic assemblies of particles. More complex geometries have been proposed, such as ellipsoids and superquadrics, however, using such complex geometrical representations make it prohibitively expensive to find the contact points and thereby to apply the DEM.

1.2 Discrete Element Method

Large-scale simulations of granular materials will require the development of continuum models that can bridge the particle scale features to the larger scales. The most appealing approach remains to study particle scale effects with the DEM to either build equations of states with hysteresis or to develop hybrid continuum-probabilistic models. In either case, it is essential to study particle-scale physics for complex particle geometries using the DEM. it is therefore essential to continue the development of algorithms that can accelerate DEM while also improving the treatment of the physics.

The standard DEM algorithm is summarized in Algorithm 1. The two key steps are the use of staggered discretizations of position and velocity in order to reach 2nd order accuracy during smooth displacements, although this so-called Verlet discretization has other advantages like being *symplectic*. The calculation of the forces, especially for Hookian models, can be tedious but it is explicit and relatively quick. The most computationally expensive step, for particles other than spheres, is the **Broad Phase Search** and the **Narrow Phase Search**. In the Broad Phase Search, for each particle, we identify all particles that have the potential to be in contact with the i-th particle, thereby leading to a list of neighbors called $\mathcal{N}_i$. The Narrow Phase Search computes the precise distance between pairs of particles, often at a very high cost.

This Broad Phase Search is usually performed by superimposing a grid the size of the

Algorithm 1: The DEM of Cundall and Strack.

Data: N particles in a domain Ω
Result: the positions $\mathbf{x}_i$ and velocities $\mathbf{v}_i$ of particles at time T

```
n = 0 ;
t_n = 0 ;
while t_n < T do
    t_{n+1} ← t_n + Δt;
    for i = 1 : N do
        x_i(t_{n+1}) ← x_i(t_n) + Δt v_i(t_{n+1/2})        ▷ update positions ;
    end
    for i = 1 : N do
        Broad Phase Search: build list of neighbors of particle i ;
        Narrow Phase Search: compute distance between neighbors ;
        F_i(t_{n+1}) ← force of neighbors on i ;
    end
    for i = 1 : N do
        v_i(t_{n+3/2}) ← v_i(t_{n+1/2}) + (Δt/m_i) F_i(t_{n+1})        ▷ update velocities ;
    end
    n ← n + 1 ;
end
```

largest particle and, roughly speaking, stipulating that particles whose centers belong to neighboring cells are in potential contact. This algorithm can be parallelized [6] but when the cardinality of $\mathcal{N}_i$ is roughly constant then the total cost of the Narrow Phase Search is linear with respect to N, the total number of particles, rather than $N!$. Unfortunately, when the particles range in diameter over more than one order of magnitude, as they do in practice, then this algorithm overestimates the sizes of $\mathcal{N}_i$ and therefore affects the total cost of the DEM.

The Narrow Phase Search requires very accurate estimates of the separation/penetration distance ϵ between pairs of particles, with the convention that $\epsilon \leq 0$ indicates contact and penetration. The force is very sensitive to the value of the penetration and the estimates must be reliable from one timestep to the next, otherwise the velocities and the positions will move erratically. In the next section, we will review Contact Detection Algorithms (CDA) which are available in the literature.

This work was largely motivated by the construction of a new method of computing the neighbors in the Broad Phase Search. Our initial objective was to develop an algorithm that

- could estimate separation distance quickly, that is at a fraction of the cost of a

typical CDA;

- could estimate separation distance reliably, i.e without fail;
- could estimate separation distance between pairs of particles that were either very far apart, or with vastly different aspect ratios.

In this research, we focus exclusively on assemblies of ellipsoids because they have been shown to model phenomena absent from assemblies of spheres, and yet the resulting algorithms still present computational challenges on modern computational platforms. It is also hoped that what can be learned will translate to other convex shapes, such as hyperquadrics.

1.3 Contact Detection Algorithms

There exist a wide-range of CDAs but we will focus on those for pairs of ellipses/ellipsoids because they represent the first step in complexity beyond spheres. One approach involves approximating an ellipse with segments of circles [7–10], grid or polar representations [11], polyhedral surfaces [12], or Non-Uniform Rational Basis Splines (NURBS) [13]. A recent survey by Kheradmand, Laforest, and Prudhomme [14] focusing only CDA between pairs of analytically defined pairs of ellipsoids has proposed a classification of algorithms into a a few broad categories which depend on the notion of **distance** between pairs of ellipsoids.

- **Intersection Algorithms (IA) :** In this category, we have algorithms that compute the pair of intersection points between the two quadratic ellipses. When ellipsoids are treated, the intersection is a curve and the approach is much less straigthforward [15].
- **Common Normal Algorithms (CNA) :** This complex algorithm estimates the smallest distance between the two ellipses by observing that at the pair of points (one on the boundary of each ellipse) where the distance is shortest, then the normals are opposite. The problem corresponds to the usual definition of distance but the problem is large, nonlinear, but needs to solved only once.
- **Geometric Potential Algorithms (GPA) :** The distance is defined to be the distance between two points (one on each ellipse) each of which is the solution to an independent problem, called the *minimum potential pair*. Most algorithms

fall in this category and were further classified into 5 subcategories. Typically, each problem on an ellipse leads to a quartic polynomial and determining the closest pair of points requires verifying all pairs of potential potential points (16 pairs).

When the two ellipses are in contact at a single point, then all these definitions coincide. Otherwise, there are slight differences between CNAs and GPAs. The survey [14] highlighted the critical role played by the co-gradient locus in the understanding of GPAs. Unfortunately, none of the published algorithms truly took advantage of the concept. More on this topic is discussed in Chapter 3.

One of the contributions of this research was the discovery of a new GPA for the estimation of the distance that constrains the pair of points to belong to the co-gradient locus (rather than being constrained to belong to the ellipses). This allows the GPA to forego expensive linear transformations required to position the pair of ellipses into some *normalized* configuration.

1.4 Scientific Machine Learning

Machine learning has emerged as a critical area of research and development in recent years. It involves the development of algorithms and techniques that enable computers to learn from data without being explicitly programmed. This has led to a wide range of applications, from image and speech recognition to autonomous vehicles and natural language processing. Another popular area of research in machine learning is reinforcement learning, which involves training agents to make decisions based on a reward signal. This has led to significant progress in areas such as robotics and game playing.

Scientific Machine Learning aims at using the flexible approximation properties found in many machine learning algorithms in order to develop new approximations of complex phenomena, like those that are history dependent, or approximate at lower cost solutions to complex problems. These topics are similar to those appearing in reduced-order modeling, or data-dependent physics modeling.

In the context of particle models, machine learning has been applied to contact detection [16], although with limited success. One of the key challenges in particle-based simulations is the accurate detection of contacts between particles. This problem has been addressed using machine learning algorithms, such as support vector machines

(SVMs) and neural networks, to predict the contact state of particle pairs based on their geometrical and mechanical properties.

Despite their flexibility, the learning in a neural networks can be biased by the data fed into the network, as we have all seen when facial recognition software is improperly trained on collections of images that are biased for sex, skin color, ethnicity, and even clothing. Nevertheless, once the training is done, the evaluation of a neural network is roughly tantamount to a large matrix-vector operation ; that is to say that once trained and validated, neural network inference is extremely fast in comparison to traditional scientific computing methods.

1.5 Objectives

The main objective of this research is to train a neural network to estimate distances between particles in the hope of detecting contacts. This approach is expected to improve the speed of contact detection, leading to a more efficient DEM. Training the neural network requires a proper sampling of the space of all pairs of ellipses/ellipsoids, where the main parameters are their separation, their relative orientation, and their relative dimensions.

The first objective leads to the second objective, that is to improve an existing algorithm [17] that samples pairs of particles by addressing observed biases, making it more accurate and customized. It was observed that the algorithm suffered from the existence of biases related to the non-linearity of transformations in parameter space that affect the usability of the generated pairs to train a neural network, leading to erroneous results. By addressing these issues, the algorithm can be fine-tuned to generate more accurate pairs that can be customized to meet specific requirements of accuracy over a specific family of parameters.

Lastly, the research seeks to develop a better analytical understanding of contact points by deriving parametric representations from the definitions, that is a representation parametrized along the co-gradient locus rather than along the ellipses. The current method of contact points detection between particles is delicate to implement and can lead to inaccurate results. By deriving parametric representations, the definition of contact points can be made more robust, leading to more accurate results.

CHAPTER 2 GENERATION OF PAIRS OF ELLIPSES AND ELLIPSOIDS FROM DISTRIBUTIONS IN PARAMETER SPACE

2.1 Introduction

Rather than proposing an algorithm to compute the separation/penetration distance between ellipses or ellipsoids, this paper describes an algorithm to produce pairs of ellipses or ellipsoids of known principal axes, axis lengths, and separation/penetration distance. Given probability distributions over the space of configurations of pairs of ellipses, this algorithm can then be used to sample pairs from these distributions.

This novel algorithm has many potential applications but the authors were initially motivated by the possibility of comparing contact detection algorithms for accuracy, stability, and efficiency over a large sample space of pairs of ellipses and ellipsoids [14]. Without an unbiased sampler of the space of pairs of ellipses, the study would have been restricted to using the expensive DEM to generate pairs of ellipses in close proximity. Past research on collision detection algorithms contained few comparisons because the contact detection algorithms were so different that comparisons on a small and select set of pairs of ellipses were bound to omit configurations of pairs of ellipses that were problematic for one algorithm but not the other [18]. The algorithm proposed here is an improvement over an earlier version that introduced bias in the relative orientation of the ellipses [17]. At the moment, the authors are using this new algorithm to explore broad-phase collision detection using machine learning algorithms [6,19]. There could be potential applications in statistics, in machine learning, or computer graphics that the authors have yet to anticipate.

In Section 2.2, we find a detailed description of the algorithm in 2D. The critical step in the algorithm is to start with an arbitrary ellipse and construct a circle around it with a known separation/penetration distance. The ellipse and the circle are then mapped into two ellipses with given relative volume ratio and orientation. The algorithm in 3D is briefly summarized in Section 2.3 as a straightforward extension of the 2D Algorithm. Section 2.4 presents a series of numerical experiments that characterize the separation/penetration distance, which is shown to be well controlled after introducing some normalization.

2.2 Generation of pairs of ellipses

2.2.1 Representation of an ellipse

In this article, we define an ellipse as the set of points

$$\mathcal{E} = \left\{ \boldsymbol{x} \in \mathbb{R}^2 \ : \ f(\boldsymbol{x}) = 0 \right\} \tag{2.1}$$

where f is the so-called potential function of the ellipse:

$$f(\boldsymbol{x}) = (\boldsymbol{x} - \boldsymbol{c})^T \mathcal{Q} (\boldsymbol{x} - \boldsymbol{c}) - 1 \,, \tag{2.2}$$

$\mathcal{Q}$ is a 2×2 symmetric positive-definite matrix, and $\boldsymbol{c} = (x_0, y_0) \in \mathbb{R}^2$ is the center of the ellipse. This definition is reviewed in detail in what follows, our main objective being to introduce the notation that will be used later to define the transformations of an ellipse and a circle into a pair of ellipses.

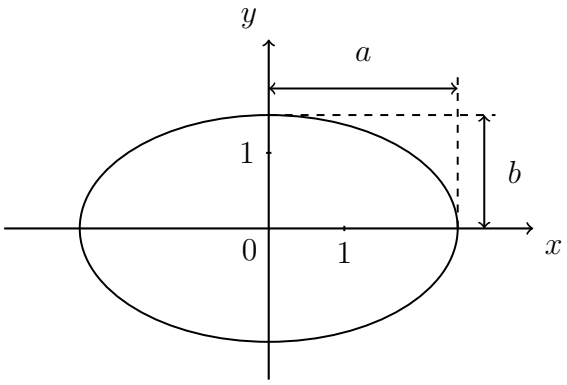


Figure 2.1 Representation of a standard ellipse with semi-axes of lengths $a \geq b$

The standard analytical form of an ellipse in Cartesian coordinates, assuming that the center of the ellipse is at the origin and that the x-axis is the major axis, is given by the points (x, y) that satisfy

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1 \,, \tag{2.3}$$

where $a \geq b > 0$ are the lengths of the semi-axes of this given ellipse, see Figure 2.1. Representation (2.3) allows for two degrees of freedom, namely the values of a and b. Of course, if we wish to represent any ellipse, such as described by the set $\mathcal{E}$ (2.1), there are a couple more parameters to consider:

1. the position of the center $\boldsymbol{c} \in \mathbb{R}^2$ on the Cartesian plane;

2. the orientation of the ellipse, which is measured by the angle between the x-axis and the direction of the major axis.

By applying a translation and a rotation in the plane, the ellipse given by (2.3) can be mapped to an ellipse $\mathcal{E}$ with these features, as described below.

Translation of the axes

In order to displace the center of the standard ellipse, we can use a translation sending the points of a coordinate system $(O; x, y)$ to those in $(O'; x', y')$. Such a transformation is illustrated in Figure 2.2.

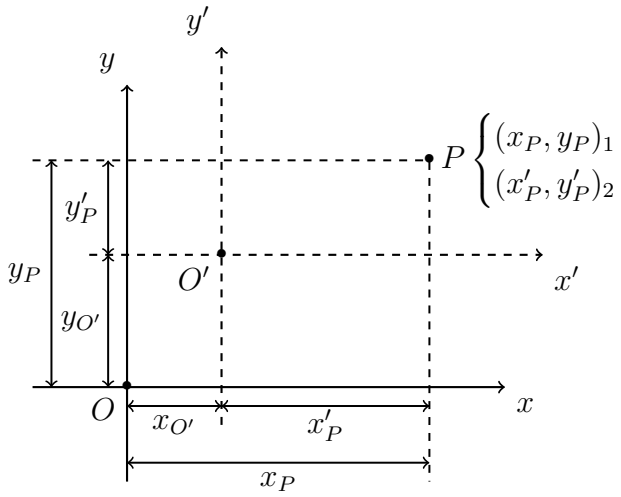


Figure 2.2 Translation of the coordinate system $(O; x, y)$ into $(O'; x', y')$

Suppose $O' = (x_{O'}, y_{O'})_1 = (0, 0)_2$, where the subscripts 1 and 2 refer to the coordinate systems $(O; x, y)$ and $(O'; x', y')$, respectively. So, for a given point $P = (x_P, y_P)_1 = (x'_P, y'_P)_2$, we have, as suggested by Figure 2.2:

$$(x_P, y_P) = (x'_P + x_{O'}, y'_P + y_{O'}) \,,$$

or, equivalently,

$$(x'_P, y'_P) = (x_P - x_{O'}, y_P - y_{O'}) \,.$$

In the case of an ellipse centered at a point $O' = (x_{O'}, y_{O'})_1$, the representation (2.3)

in $(O'; x', y')$ is therefore given in $(O; x, y)$ as

$$\frac{x'^2}{a^2} + \frac{y'^2}{b^2} = \frac{(x - x_{O'})^2}{a^2} + \frac{(y - y_{O'})^2}{b^2} = 1\,. \tag{2.4}$$

Rotation of the axes

To rotate a standard ellipse (2.3) around the origin, we can introduce a rotation of the coordinate system (as shown in Figure 2.3). In order to do so, we must specify the desired angles in the expression of our coordinates, using the polar form. Let

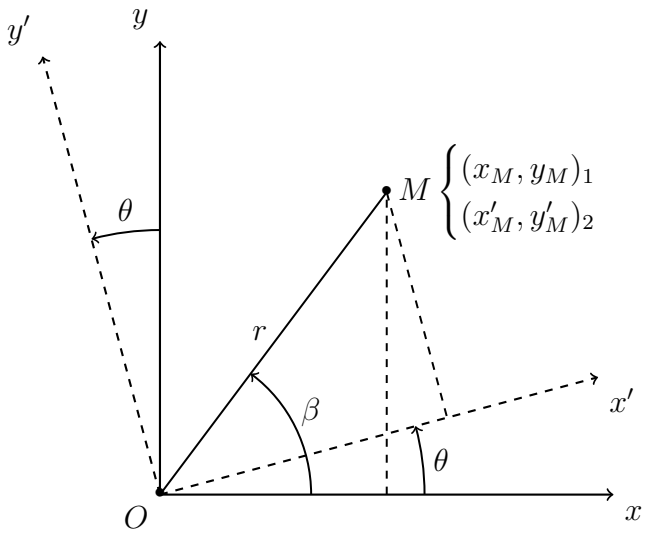


Figure 2.3 Rotation of a coordinate system $(O; x, y)$ into $(O; x', y')$ by an angle θ

$(O; x, y)$ define the usual Cartesian coordinate system. Moreover, let $M = (x_M, y_M)$ be a given point with Euclidean distance r from the origin O and let β be the relative angle between the x-axis and the line segment $\overline{OM}$. The point M in polar coordinates is given by $M = (r\cos\beta, r\sin\beta)$. Consider now the coordinate system $(O; x', y')$ obtained by rotation of the original coordinate system $(O; x, y)$ by an angle θ around the origin, where point M in $(O; x, y)$ is written as $M = (x_M, y_M)$ while it is written as $M = (x'_M, y'_M)$ in $(O; x', y')$. The relations between the coordinates in each system is given by

$$(x_M, y_M) = (x'_M\cos\theta - y'_M\sin\theta,\ x'_M\sin\theta + y'_M\cos\theta)\,,$$

or, equivalently,

$$(x'_M, y'_M) = (x_M \cos\theta + y_M \sin\theta, -x_M \sin\theta + y_M \cos\theta)\,.$$

So, for the case of an ellipse oriented at an angle θ, the representation (2.3) in $(O; x', y')$ is transformed into the rotated representation in $(O; x, y)$ as:

$$\frac{x'^2}{a^2} + \frac{y'^2}{b^2} = \frac{(x\cos\theta + y\sin\theta)^2}{a^2} + \frac{(-x\sin\theta + y\cos\theta)^2}{b^2} = 1\,. \tag{2.5}$$

Matrix notation

To achieve the most general analytical representation of an ellipse in Cartesian coordinates, we need to combine the translation and rotation. Before doing that, we rewrite the modified equations in matrix-vector multiplication form in order to lighten the notation. The standard equation of an ellipse can be rewritten as

$$\begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} 1/a^2 & 0 \\ 0 & 1/b^2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = 1\,,$$

where we have identified a diagonal matrix that will henceforth be known as the scaling matrix and will be denoted

$$\mathcal{D}(a, b) := \begin{bmatrix} 1/a^2 & 0 \\ 0 & 1/b^2 \end{bmatrix}\,.$$

By denoting $\boldsymbol{x} = (x, y) \in \mathbb{R}^2$ and omitting for now the dependence on a and b, we can write

$$\mathcal{E} : \boldsymbol{x}^T \mathcal{D} \boldsymbol{x} = 1\,.$$

Now, to translate the center of an ellipse $\mathcal{E}$ at a given point $\boldsymbol{c} \in \mathbb{R}^2$, one replaces $\boldsymbol{x}$ by $\boldsymbol{x} - \boldsymbol{c}$ as per Eq. (2.4), thus obtaining

$$(\boldsymbol{x} - \boldsymbol{c})^T \mathcal{D} (\boldsymbol{x} - \boldsymbol{c}) = 1\,.$$

For the rotation by an angle θ, as per Eq. (2.5), one replaces $\boldsymbol{x}$ by

$$\begin{bmatrix} x\cos\theta + y\sin\theta \\ -x\sin\theta + y\cos\theta \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}^T \boldsymbol{x}\,,$$

where we can identify the orthogonal matrix, referred to as the rotation matrix,

$$\mathcal{R}(\theta) := \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} .$$

Using that, omitting the dependence on θ, Eq. (2.5) becomes

$$(\mathcal{R}^T \boldsymbol{x})^T \mathcal{D} (\mathcal{R}^T \boldsymbol{x}) = \boldsymbol{x}^T \mathcal{R} \mathcal{D} \mathcal{R}^T \boldsymbol{x} = 1 \, .$$

Finally, by combining the translation and rotation, we replace $\boldsymbol{x}$ by $\boldsymbol{x}' = \mathcal{R}^T(\boldsymbol{x} - \boldsymbol{c})$, thereby obtaining the general form

$$\boldsymbol{x}'^T \mathcal{D} \boldsymbol{x}' = (\boldsymbol{x} - \boldsymbol{c})^T \mathcal{R} \mathcal{D} \mathcal{R}^T (\boldsymbol{x} - \boldsymbol{c}) = 1 \, . \tag{2.6}$$

These two equivalent equations allow one to see the ellipse from two points of view: either as the set of all points verifying Eq. (2.3) in $(\boldsymbol{c}; x', y')$, or the set of all points verifying the general equation (2.6) in $(O; x, y)$. Note that $\mathcal{R}\mathcal{D}\mathcal{R}^T$ is the spectral decomposition of a symmetric positive-definite matrix $\mathcal{Q} \in \mathbb{R}^{2\times 2}$; indeed, for all non-zero vectors $\boldsymbol{x} - \boldsymbol{c} \in \mathbb{R}^2$, we have

$$(\boldsymbol{x} - \boldsymbol{c})^T \mathcal{Q} (\boldsymbol{x} - \boldsymbol{c}) = \boldsymbol{x}'^T \mathcal{D} \boldsymbol{x}' > 0 \, .$$

The eigenvalues $\lambda_1 \in \mathbb{R}_+^*$ and $\lambda_2 \in \mathbb{R}_+^*$ of $\mathcal{Q}$ are related to the lengths of the semi-axes according to

$$\lambda_1 = 1/a^2 \quad \text{and} \quad \lambda_2 = 1/b^2 \quad \text{with} \quad 0 < \lambda_1 \leq \lambda_2 \, .$$

This explains why a general ellipse $\mathcal{E} \subset \mathbb{R}^2$ centered at $\boldsymbol{c} \in \mathbb{R}^2$ is the set of points

$$\mathcal{E} = \left\{ \boldsymbol{x} \in \mathbb{R}^2 \, : \, (\boldsymbol{x} - \boldsymbol{c})^T \mathcal{Q} (\boldsymbol{x} - \boldsymbol{c}) = 1 \right\}$$

with $\mathcal{Q} \in \mathbb{R}^{2\times 2}$ a symmetric positive-definite matrix. A single ellipse is thus completely determined by exactly five parameters. One way to see it is through the matrix equation (2.6) (the three distinct entries of the symmetric matrix $\mathcal{Q}$ and the two coordinates of the center $\boldsymbol{c}$). Another way to see it is through the five geometric parameters used to describe the ellipse (its major axis length a, its minor axis length b, the angle θ it makes with the x-axis, and the polar coordinates r and β of the center $\boldsymbol{c}$). Alternatively, the shape of the ellipse, usually described by a and b, can

be parameterized by

$$\gamma = \frac{a}{b} \quad \text{and} \quad \omega = \frac{\pi ab}{\pi} = ab\,. \tag{2.7}$$

These quantities will be respectively called the *aspect ratio* and the *area ratio* (the latter is indeed the ratio between the area πab of the ellipse and the area π of the unit circle).

2.2.2 An ellipse and a circle of fixed separation

In this section, we present a configuration of an ellipse and a circle for which it is possible to measure the separation/penetration distance. This simple configuration plays a key role in our construction of arbitrary pairs of ellipses with known separation/penetration distance.

We will show that given an ellipse $\bar{\mathcal{E}}_i$ in the coordinate system $(\bar{\boldsymbol{c}}_i, \bar{x}, \bar{y})$, associated with the standard ellipse of center $\bar{\boldsymbol{c}}_i$ placed at the origin, a point $\bar{\boldsymbol{x}}_i \in \bar{\mathcal{E}}_i$, and a signed distance $\bar{\varepsilon}$, we can place a circle of radius unity, say $\bar{\mathcal{C}}_j$, at a distance of $\bar{\varepsilon}$ in the direction normal to $\bar{\boldsymbol{x}}_i$ on the ellipse. When $\bar{\varepsilon}$ is positive, $\bar{\mathcal{E}}_i$ and $\bar{\mathcal{C}}_j$ are separated, but when $\bar{\varepsilon}$ is negative, they are overlapping.

We assume that the ellipse $\bar{\mathcal{E}}_i$ is centered at the origin $\bar{\boldsymbol{c}}_i = (0, 0)$ with principal axes along the coordinate axes, hence the geometry of $\bar{\mathcal{E}}_i$ is entirely determined by its aspect ratio $\bar{\gamma}_i$ and area ratio $\bar{\omega}_i$. The lengths of the semi-axes are given by $\bar{a}_i = \sqrt{\bar{\omega}_i \bar{\gamma}_i}$ and $\bar{b}_i = \sqrt{\bar{\omega}_i / \bar{\gamma}_i}$. The ellipse is thus described as the set of points $\bar{\boldsymbol{x}} \in \mathbb{R}^2$ satisfying

$$\bar{\mathcal{E}}_i\colon\ \bar{\boldsymbol{x}}^T \bar{\mathcal{D}}_i \bar{\boldsymbol{x}} = 1\,,$$

where $\bar{\mathcal{D}}_i = \mathcal{D}(\bar{a}_i, \bar{b}_i)$. The potential function associated with $\bar{\mathcal{E}}_i$ is given here by $\bar{f}_i(\bar{\boldsymbol{x}}) = \bar{\boldsymbol{x}}^T \bar{\mathcal{D}}_i \bar{\boldsymbol{x}} - 1$.

The goal is now to place the circle $\bar{\mathcal{C}}_j$ close to the chosen point $\bar{\boldsymbol{x}}_i \in \bar{\mathcal{E}}_i$ defined in polar coordinates as

$$\bar{\boldsymbol{x}}_i = \begin{bmatrix} \bar{a}_i \cos\phi \\ \bar{b}_i \sin\phi \end{bmatrix}\,, \tag{2.8}$$

where $\phi \in\,]-\pi, \pi]$. A simple calculation shows that the unit outward normal to $\bar{\mathcal{E}}_i$ at $\bar{\boldsymbol{x}}_i$ is given by

$$\bar{\boldsymbol{n}}_i = \frac{\nabla \bar{f}_i(\bar{\boldsymbol{x}}_i)}{\|\nabla \bar{f}_i(\bar{\boldsymbol{x}}_i)\|} = \frac{1}{\sqrt{\bar{b}_i^2 \cos^2\phi + \bar{a}_i^2 \sin^2\phi}} \begin{bmatrix} \bar{b}_i \cos\phi \\ \bar{a}_i \sin\phi \end{bmatrix}\,.$$

It should be obvious that if $\bar{\mathcal{C}}_j$ does not overlap with $\bar{\mathcal{E}}_i$, then $\bar{\boldsymbol{x}}_i$ is the closest point to $\bar{\mathcal{C}}_j$ on $\bar{\mathcal{E}}_i$ if and only if the centre $\bar{\boldsymbol{c}}_j$ of $\bar{\mathcal{C}}_j$ belongs to the line in the direction $\bar{\boldsymbol{n}}_i$ passing through $\bar{\boldsymbol{x}}_i$. In fact, if we write

$$\bar{\boldsymbol{c}}_j = \bar{\boldsymbol{x}}_i + (1 + \bar{\varepsilon})\, \bar{\boldsymbol{n}}_i \,, \tag{2.9}$$

then when $\bar{\varepsilon} = 0$, the ellipse $\bar{\mathcal{E}}_i$ and the circle $\bar{\mathcal{C}}_j$ are tangent at $\bar{\boldsymbol{x}}_i$. On the other hand, if $\bar{\varepsilon} > 0$, it defines the separation distance between the sets $\bar{\mathcal{E}}_i$ and $\bar{\mathcal{C}}_j$ and if $\bar{\varepsilon} < 0$ with $\bar{\varepsilon}$ small, the construction produces an ellipse and a circle with a small overlap. This is illustrated in Figure 2.4 for $\bar{\varepsilon} > 0$.

The construction can therefore be summarized as follows: given an ellipse $\bar{\mathcal{E}}_i$ of known $\bar{\gamma}_i$, $\bar{\omega}_i$, a point $\bar{\boldsymbol{x}}_i$ along its surface, and a penetration/separation distance $\bar{\varepsilon}$, we can compute $\bar{\boldsymbol{c}}_j$ the center of the neighboring circle of radius unity. We describe below the two affine transformations that are used to map Ellipse $\bar{\mathcal{E}}_i$ and Circle $\bar{\mathcal{C}}_j$ onto two general ellipses $\mathcal{E}_i$ and $\mathcal{E}_j$.

First transformation

The first transformation moves the circle $\bar{\mathcal{C}}_j$ so that it is centered at the origin and applies a rotation of angle $-\hat{\theta}_i \in \,]-\pi, \pi]$. In other words, the transformation $\hat{\boldsymbol{x}} = T_1(\bar{\boldsymbol{x}})$ corresponds to a change in reference from $(\bar{\boldsymbol{c}}_i; \bar{x}, \bar{y})$ to the coordinate system $(\bar{\boldsymbol{c}}_j; \hat{x}, \hat{y})$ making an angle $\angle(\bar{x}, \hat{x}) = -\hat{\theta}_i \in \,]-\pi, \pi]$ with the original reference system. For now, the orientation of this new coordinate system is arbitrary. Note that there is a reason why a simple translation is not used: it would have the effect of producing two ellipses that have the same orientation in the end result, thereby introducing a bias in our data set. The transformation can be represented as

$$\hat{\boldsymbol{x}} = \hat{\mathcal{R}}_i(\bar{\boldsymbol{x}} - \bar{\boldsymbol{c}}_j) \,, \tag{2.10}$$

where $\hat{\mathcal{R}}_i = \mathcal{R}^T(-\hat{\theta}_i) = \mathcal{R}(\hat{\theta}_i)$. We can thus redefine the ellipse and the circle in the new frame of reference as

$$\begin{aligned} \hat{\mathcal{E}}_i &: \;\; (\hat{\boldsymbol{x}} - \hat{\boldsymbol{c}}_i)^T \hat{\mathcal{Q}}_i (\hat{\boldsymbol{x}} - \hat{\boldsymbol{c}}_i) = 1 \;, \\ \hat{\mathcal{C}}_j &: \;\; \hat{\boldsymbol{x}}^T \hat{\boldsymbol{x}} = 1 \,, \end{aligned} \tag{2.11}$$

where

$$\hat{\boldsymbol{c}}_i = -\hat{\mathcal{R}}_i \bar{\boldsymbol{c}}_j \tag{2.12}$$

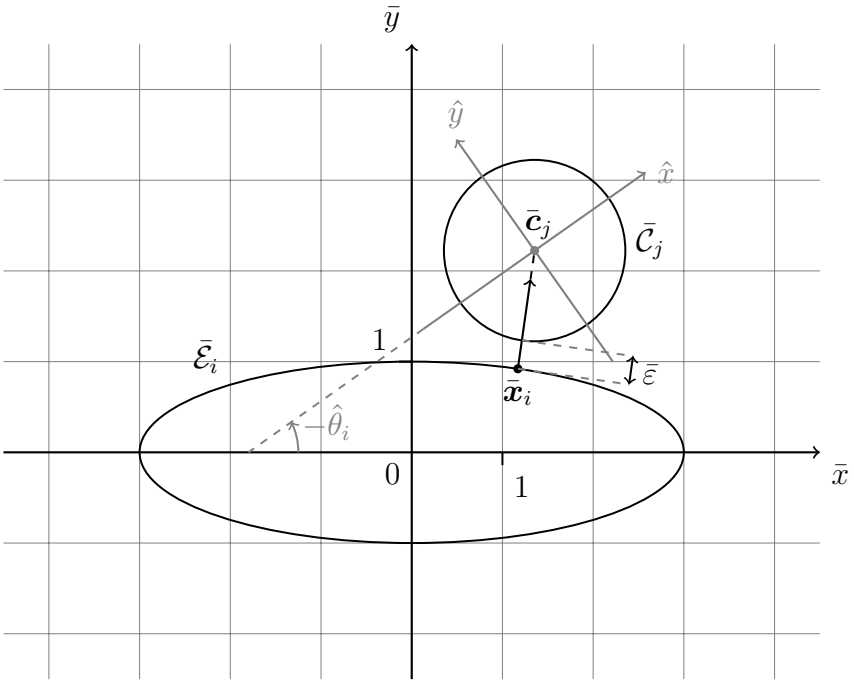


Figure 2.4 Initial configuration of an ellipse and a circle of known separation

and $\hat{\mathcal{Q}}_i = \hat{\mathcal{R}}_i \hat{\mathcal{D}}_i \hat{\mathcal{R}}_i^T$ with $\hat{\mathcal{D}}_i = \bar{\mathcal{D}}_i$, see Figure 2.5.

Second transformation

The second transformation $\boldsymbol{x} = T_2(\hat{\boldsymbol{x}})$ maps the unit circle $\hat{\mathcal{C}}_j$ to an ellipse $\mathcal{E}_j$ of area ratio $\omega_j = 1$, thus enabling by prior composition with T_1 to send the circle $\bar{\mathcal{C}}_j$ to an ellipse that has a unit area ratio. So, for a given aspect ratio γ_j, we define the semi-axes lengths $a_j = \sqrt{\gamma_j}$ and $b_j = 1/a_j$. By also choosing $\theta_j \in \,]-\pi,\, \pi]$ and $\boldsymbol{c}_j \in \mathbb{R}^2$, the circle $\hat{\mathcal{C}}_j$ becomes

$$\mathcal{E}_j \colon \; (\boldsymbol{x} - \boldsymbol{c}_j)^T \mathcal{Q}_j (\boldsymbol{x} - \boldsymbol{c}_j) = 1 \,,$$

where $\mathcal{Q}_j = \mathcal{R}_j \mathcal{D}_j \mathcal{R}_j^T$ with $\mathcal{R}_j = \mathcal{R}(\theta_j)$ and $\mathcal{D}_j = \mathcal{D}(a_j,\, b_j)$. This transformation, see Figure 2.6, corresponds to the affine map

$$\boldsymbol{x} = \mathcal{R}_j \mathcal{D}_j^{-1/2} \hat{\boldsymbol{x}} + \boldsymbol{c}_j \,. \tag{2.13}$$

Applying it to $\hat{\mathcal{E}}_i$, we get the new equation of the ellipse $\mathcal{E}_i$

$$\mathcal{E}_i \colon \; (\boldsymbol{x} - \boldsymbol{c}_i)^T \mathcal{Q}_i (\boldsymbol{x} - \boldsymbol{c}_i) = 1 \,,$$

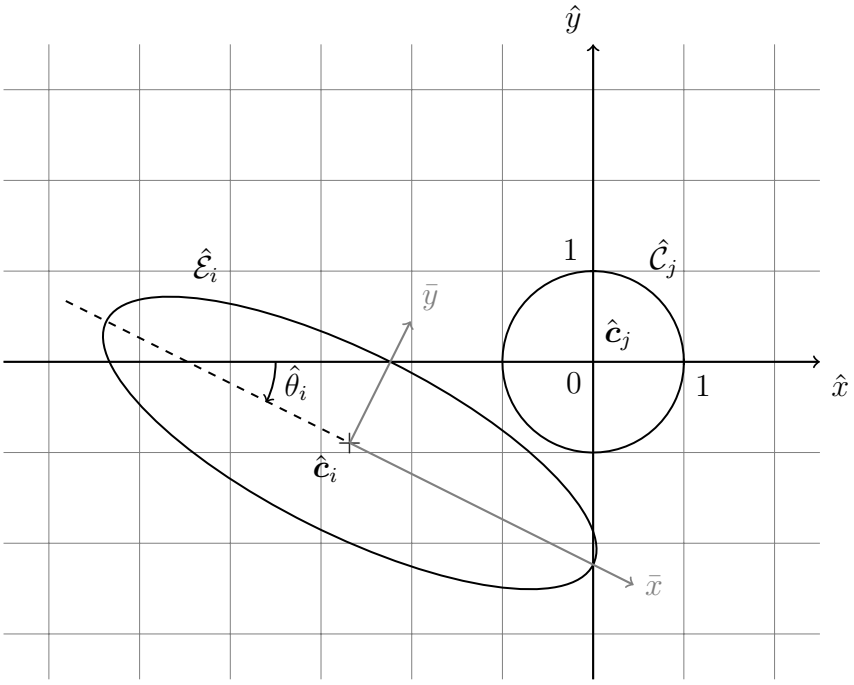


Figure 2.5 Intermediate configuration of an ellipse and a circle of known separation

where

$$\boldsymbol{c}_i = \mathcal{R}_j \mathcal{D}_j^{-1/2} \hat{\boldsymbol{c}}_i + \boldsymbol{c}_j \,, \tag{2.14}$$

and

$$\mathcal{Q}_i = \mathcal{R}_j \mathcal{D}_j^{1/2} \hat{\mathcal{R}}_i \hat{\mathcal{D}}_i \hat{\mathcal{R}}_i^T \mathcal{D}_j^{1/2} \mathcal{R}_j^T \,. \tag{2.15}$$

By spectral decomposition of $\mathcal{Q}_i = \mathcal{R}_i \mathcal{D}_i \mathcal{R}_i^T$, we can determine the parameters γ_i, ω_i, and θ_i.

Using the construct described above, that is, using the geometric parameters of $\bar{\mathcal{E}}_i$ and $\bar{\mathcal{C}}_j$ and the parameters $\bar{\varepsilon}$, ϕ, and $\hat{\theta}_j$ describing their relative positioning, in order to transform $\bar{\mathcal{E}}_i$ and $\bar{\mathcal{C}}_j$ into $\mathcal{E}_i$ and $\mathcal{E}_j$, respectively, it has been observed in [17] that certain biases are produced as a result of the nonlinear relationship between different parameters. To be specific, mapping $\bar{\mathcal{E}}_i$ and $\bar{\mathcal{C}}_j$ in this way produces ellipses whose relative orientation is often very similar. In other words, the produced value of θ_i is such that $\theta_i \sim \theta_j + k\pi$, $k = -1$, 0, or $+1$, most of the time. In the next section, we propose a modification of this approach that allows us to fix the relative orientation of the ellipses in the final configuration, and thereby eliminate this bias.

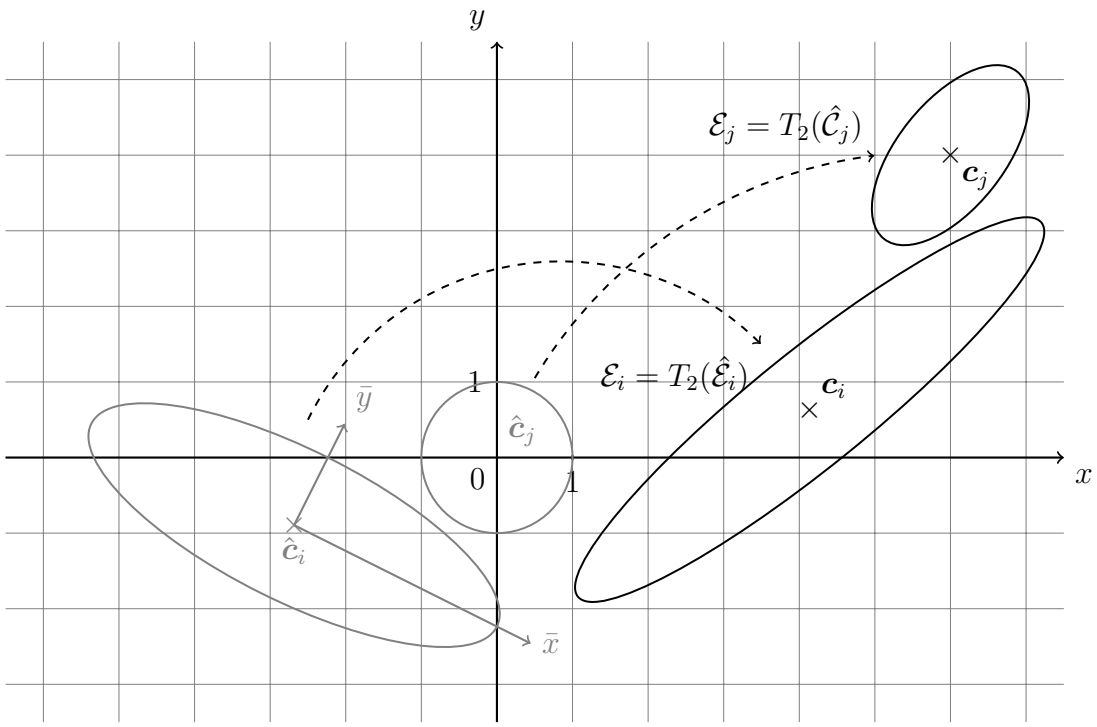


Figure 2.6 Final configuration of a pair of ellipses of "known" separation

2.2.3 Sampling algorithm for arbitrary pairs of ellipses

In this section, we describe the new algorithm to produce pairs of ellipses $\mathcal{E}_i$ and $\mathcal{E}_j$ from the generation of an ellipse and a circle using the two transformations described earlier. In order to avoid the biases in the alignment of the ellipses, we exploit the symmetry of the transformations (they are of full rank) to impose the shape parameters in the final configuration γ_i, ω_i, θ_i, γ_j, and θ_j and, from them, calculate the initial parameters $\bar{a}_i$ and $\bar{b}_i$. Using these two parameters and prescribing $\bar{\varepsilon}$ and ϕ, one can thus construct the ellipse and circle in $(O, \bar{x}, \bar{y})$. Finally, introducing the center $\boldsymbol{c}_j$, one can determine the two transformations T_1 and T_2 and thus compute the center $\boldsymbol{c}_i$ of $\mathcal{E}_i$ as well as the actual distance ε between the two ellipses in the final configuration. We describe the algorithm in more details below.

First, it is useful to distinguish between two types of parameters, the shape parameters $\mathbb{S} = \{\gamma, \omega, \theta\}$, for which notations such as $\hat{\mathbb{S}}_i = \{\hat{\gamma}_i, \hat{\omega}_i, \hat{\theta}_i\}$ or $\mathbb{S}_j = \{\gamma_j, \omega_j, \theta_j\}$ can be used in the appropriate coordinate system, and the parameters describing the relative positioning of the ellipse and the circle in the initial configuration, $\mathbb{P} = \{\phi, \bar{\varepsilon}\}$. The new approach takes as inputs the shape parameters of the resulting ellipses (that is $\mathbb{S}_i$

and $\mathbb{S}_j$ instead of $\bar{\mathbb{S}}_i$ and $\bar{\mathbb{S}}_j$), the relative positioning parameters $\mathbb{P}$, and the absolute positioning parameter $\boldsymbol{c}_j$. In summary, the input parameters are

$$X_{\text{in}} = \mathbb{S}_i \cup \mathbb{S}_j \cup \mathbb{P} \cup \{\boldsymbol{c}_j\} = \{\gamma_i, \omega_i, \theta_i, \gamma_j, \omega_j, \theta_j, \phi, \bar{\varepsilon}, \boldsymbol{c}_j\}\,. \tag{2.16}$$

The corresponding output parameters are the following parameters

$$Y_{\text{out}} = \{\boldsymbol{c}_i, \boldsymbol{x}_i, \boldsymbol{x}_j, \varepsilon\}\,,$$

where the points $\boldsymbol{x}_i \in \mathcal{E}_i$ and $\boldsymbol{x}_j \in \mathcal{E}_j$ are the "closest" points to $\mathcal{E}_j$ and $\mathcal{E}_i$, respectively, and are used to evaluate the distance $\varepsilon = \|\boldsymbol{x}_j - \boldsymbol{x}_i\|$, see e.g. [14]. The points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are also referred to as contact points in the case where $\mathcal{E}_i$ and $\mathcal{E}_j$ overlap.

We begin by observing that the set of all pairs of ellipses can be reduced to those for which $\boldsymbol{c}_j = \boldsymbol{0}$, after a translation, and $\omega_j = 1$, after a scaling of all components. The constraint $\omega_j = 1$ may require a rescaling of the nominal separation distance $\bar{\varepsilon}$ and of the two axes of $\mathcal{E}_i$, namely by scaling γ_i, ω_i, and γ_j. In other words, setting $\omega_j = 1$ is tantamount to a choice of units.

The algorithm to produce a pair of ellipses is a function of the parameters X_{in}. From the data (2.16) and $\omega_j = 1$, one can compute the lengths of the semi-axes a_i, b_i, a_j, and b_j using the relations (2.7). From (2.15), we then get

$$\hat{\mathcal{Q}}_i = \mathcal{D}_j^{-1/2} \mathcal{R}_j^T \mathcal{Q}_i \mathcal{R}_j \mathcal{D}_j^{-1/2}\,.$$

Although we know that the ellipse $\hat{\mathcal{E}}_j$ is actually a circle centered at the origin, at this stage of the algorithm, we do not know $\hat{\boldsymbol{c}}_i$. Yet, we can still compute $\hat{\theta}_i$ from $\hat{\mathcal{Q}}_i$, which allows us to determine the transformation T_1 (2.10) with $\hat{\mathcal{R}}_i = \mathcal{R}(\hat{\theta}_i)$, and the values of $\bar{a}_i = \hat{a}_i$ and $\bar{b}_i = \hat{b}_i$. In the coordinate system $(\bar{x}, \bar{y})$, we use the value of $\bar{\boldsymbol{x}}_i$ along $\bar{\mathcal{E}}_i$ (known from ϕ, see Eq. (2.8)) to compute $\bar{\boldsymbol{n}}_i$ and thus $\bar{\boldsymbol{c}}_j$ from (2.9). Afterwards, Eq. (2.12) gives $\hat{\boldsymbol{c}}_i = T_1(\boldsymbol{0}) = -\hat{\mathcal{R}}_i \bar{\boldsymbol{c}}_j$. Finally, the center of $\mathcal{E}_i$ is computed from the second transformation T_2 (2.13) as

$$\boldsymbol{c}_i = T_2(\hat{\boldsymbol{c}}_i) = T_2 \circ T_1(\boldsymbol{0})\,,$$

or, using directly (2.14), as

$$\boldsymbol{c}_i = \mathcal{R}_j \mathcal{D}_j^{-1/2} \hat{\boldsymbol{c}}_i + \boldsymbol{c}_j\,.$$

This provides all the information to construct $\mathcal{E}_i$ and $\mathcal{E}_j$ in the coordinate system (x, y) except for the distance ε between the two ellipses. The distance $\bar{\varepsilon}$ is only known exactly in $(\bar{x}, \bar{y})$ so that the corresponding distance ε in (x, y) needs to be computed through the transformations. It actually represents an approximation to $\bar{\varepsilon}$ and this error will be studied numerically later. To compute the exact distance $\varepsilon = \|\boldsymbol{x}_j - \boldsymbol{x}_i\|$, first the point $\bar{\boldsymbol{x}}_i$ is mapped to $\boldsymbol{x}_i \in \mathcal{E}_i$ using the two transformations T_1 and T_2, i.e. $\boldsymbol{x}_i = T_2 \circ T_1(\bar{\boldsymbol{x}}_i)$, and the algorithm described in [17] is used to estimate $\boldsymbol{x}_j \in \mathcal{E}_j$ that is closest to $\mathcal{E}_i$. A thorough analysis of the notion of distance between ellipses can be found in [14]. The algorithm is summarized in Algorithm 2.

Finally, if probability distributions are provided for the parameters in (2.16), then Algorithm 2 can be used to produce samples from these distributions.

Algorithm 2: Sampling algorithm for a pair of ellipses

Data: γ_i, ω_i, θ_i, γ_j, θ_j, ϕ, $\bar{\varepsilon}$, $\boldsymbol{c}_j$
Result: $\boldsymbol{c}_i$, $\boldsymbol{x}_i$, $\boldsymbol{x}_j$, ε

compute $a_i = \sqrt{\omega_i \gamma_i}$, $b_i = \sqrt{\omega_i / \gamma_i}$;
compute $\mathcal{D}_i = \mathcal{D}(a_i, b_i)$, $\mathcal{R}_i = \mathcal{R}(\theta_i)$;
compute $a_j = \sqrt{\gamma_j}$, $b_j = 1/a_j$;
compute $\mathcal{D}_j = \mathcal{D}(a_j, b_j)$, $\mathcal{R}_j = \mathcal{R}(\theta_j)$;
compute $\hat{\mathcal{Q}}_i = \mathcal{D}_j^{-1/2} \mathcal{R}_j^T \mathcal{R}_i \mathcal{D}_i \mathcal{R}_i^T \mathcal{R}_j \mathcal{D}_j^{-1/2}$;
by spectral decomposition of $\hat{\mathcal{Q}}_i$, identify $\bar{a}_i = \hat{a}_i = 1/\sqrt{\lambda_1}$, $\bar{b}_i = \hat{b}_i = 1/\sqrt{\lambda_2}$, the orientation angle $\hat{\theta}_i$, and compute $\hat{\mathcal{R}}_i = \mathcal{R}(\hat{\theta}_i)$;
compute $\bar{\boldsymbol{x}}_i = (\bar{a}_i \cos\phi, \bar{b}_i \sin\phi)$;
compute $\bar{\boldsymbol{n}}_i = (\bar{b}_i \cos\phi, \bar{a}_i \sin\phi) / \sqrt{\bar{b}_i^2 \cos^2\phi + \bar{a}_i^2 \sin^2\phi}$;
compute $\bar{\boldsymbol{c}}_j = \bar{\boldsymbol{x}}_i + (1 + \bar{\varepsilon})\, \bar{\boldsymbol{n}}_i$;
compute $\boldsymbol{c}_i = T_2 \circ T_1(\boldsymbol{0})$;
compute $\boldsymbol{x}_i = T_2 \circ T_1(\bar{\boldsymbol{x}}_i)$;
solve for point $\boldsymbol{x}_j$ [17] ;
compute $\varepsilon = \|\boldsymbol{x}_j - \boldsymbol{x}_i\|$

2.3 Extension to the generation of pairs of ellipsoids

This section details how the 2D method described above is extended to generate pairs of ellipsoids. The sampling method is sensibly the same with some adjustments that

come with the added dimension. We define the 3D scaling matrix to be

$$\mathcal{D}(a,b,c) = \begin{bmatrix} 1/a^2 & 0 & 0 \\ 0 & 1/b^2 & 0 \\ 0 & 0 & 1/c^2 \end{bmatrix}$$

where $a \geq b \geq c > 0$ are the lengths of the semi-axes of a given ellipsoid. With these three lengths, there are now three ratios that describe the shape of an ellipsoid, namely two aspect ratios and one volume ratio:

$$\gamma_1 = \frac{a}{b}\,, \quad \gamma_2 = \frac{b}{c}\,, \quad \text{and} \quad \omega = \frac{\frac{4}{3}\pi abc}{\frac{4}{3}\pi} = abc\,. \tag{2.17}$$

Inversely, given the ratios γ_1, γ_2, and ω, the lengths of the semi-axes are given as:

$$a = \sqrt[3]{\omega\gamma_1^2\gamma_2}\,, \quad b = \sqrt[3]{\omega\frac{\gamma_2}{\gamma_1}}\,, \quad \text{and} \quad c = \sqrt[3]{\frac{\omega}{\gamma_1\gamma_2^2}}\,.$$

We also introduce the general rotation matrix in 3D as

$$\mathcal{R}(\theta_x,\theta_y,\theta_z) = \begin{bmatrix} \cos\theta_z & -\sin\theta_z & 0 \\ \sin\theta_z & \cos\theta_z & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta_y & 0 & \sin\theta_y \\ 0 & 1 & 0 \\ -\sin\theta_y & 0 & \cos\theta_y \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_x & -\sin\theta_x \\ 0 & \sin\theta_x & \cos\theta_x \end{bmatrix},$$

where $(\theta_x,\theta_y,\theta_z) \in \,]-\pi,\pi] \times [0,\pi] \times \,]-\pi,\pi]$ are the Euler angles that describe the orientation of that ellipsoid with respect to a fixed coordinate system. Finally, an ellipsoid with semi-axes a, b, and c, in the classical configuration can be parameterized using the ordered pair of angles $(\phi,\,\psi) \in \,]-\pi,\,\pi] \times [0,\,\pi]$ as

$$\bar{\boldsymbol{x}}_i = \begin{bmatrix} a\cos\phi\cos\psi \\ b\sin\phi\cos\psi \\ c\sin\psi \end{bmatrix}.$$

As previously described in 2D, given the shape parameters $\mathbb{S} = \{\gamma_1,\gamma_2,\omega,\theta_x,\theta_y,\theta_z\}$ for the two ellipsoids $\boldsymbol{\mathcal{E}}_i$ and $\boldsymbol{\mathcal{E}}_j$ in the resulting configuration, the relative positioning parameters $\mathbb{P} = \{\phi,\psi,\bar{\varepsilon}\}$ and the center $\boldsymbol{c}_j$ of $\boldsymbol{\mathcal{E}}_j$, positioning the other center $\boldsymbol{c}_i$ with respect to the separation/penetration ε is achieved with the following algorithm that

takes as inputs:

$$X_{\text{in}} = \mathbb{S}_i \cup \mathbb{S}_j \cup \mathbb{P} \cup \{\boldsymbol{c}_j\} = \{\gamma_{1i}, \gamma_{2i}, \omega_i, \theta_{xi}, \theta_{yi}, \theta_{zi}, \gamma_{1j}, \gamma_{2j}, \omega_j, \theta_{xj}, \theta_{yj}, \theta_{zj}, \phi, \psi, \bar{\varepsilon}, \boldsymbol{c}_j\}\,, \tag{2.18}$$

and as outputs:

$$Y_{\text{out}} = \{\boldsymbol{c}_i, \boldsymbol{x}_i, \boldsymbol{x}_j, \varepsilon\}\,.$$

Using the shape parameters associated with $\mathcal{E}_i$ and $\mathcal{E}_j$, choosing $\omega_j = 1$, one can then compute the matrix $\hat{\mathcal{Q}}_i = \mathcal{D}_j^{-1/2}\mathcal{R}_j^T\mathcal{R}_i\mathcal{D}_i\mathcal{R}_i^T\mathcal{R}_j\mathcal{D}_j^{-1/2}$ and identify the lengths $\bar{a}_i = \hat{a}_i$, $\bar{b}_i = \hat{b}_i$, $\bar{c}_i = \hat{c}_i$ of the semi-axes and the three Euler angles $\hat{\theta}_{xi}$, $\hat{\theta}_{yi}$, $\hat{\theta}_{zi}$ by the spectral decomposition of $\hat{\mathcal{Q}}_i$. In the ellipsoid-sphere configuration, the standard ellipsoid is then defined as

$$\bar{\mathcal{E}}_i = \left\{\bar{\boldsymbol{x}} \in \mathbb{R}^3 :\ \bar{f}_i(\bar{\boldsymbol{x}}) = 0\right\},$$

where $\bar{f}_i(\bar{\boldsymbol{x}}) = \bar{\boldsymbol{x}}^T\bar{\mathcal{D}}_i\ \bar{\boldsymbol{x}} - 1$ and $\bar{\mathcal{D}}_i = \mathcal{D}(\bar{a}_i, \bar{b}_i, \bar{c}_i)$. A point $\bar{\boldsymbol{x}}_i$ is then constructed on $\bar{\mathcal{E}}_i$, using the input angles $(\phi,\, \psi) \in\,]-\pi,\, \pi] \times [0,\, \pi]$, that is

$$\bar{\boldsymbol{x}}_i = \begin{bmatrix} \bar{a}_i \cos\phi \cos\psi \\ \bar{b}_i \sin\phi \cos\psi \\ \bar{c}_i \sin\psi \end{bmatrix},$$

at which the unit outward normal vector $\bar{\boldsymbol{n}}_i$ to $\bar{\mathcal{E}}_i$ is given by

$$\bar{\boldsymbol{n}}_i = \frac{\nabla\bar{f}_i(\bar{\boldsymbol{x}}_i)}{\|\nabla\bar{f}_i(\bar{\boldsymbol{x}}_i)\|}\,.$$

One can then use the normal vector and a separation/penetration between $\bar{\mathcal{E}}_i$ and $\bar{\mathcal{S}}_j$ of size $\bar{\varepsilon}$ to position the center

$$\bar{\boldsymbol{c}}_j = \bar{\boldsymbol{x}}_i + (1 + \bar{\varepsilon})\,\bar{\boldsymbol{n}}_i$$

of the unit sphere

$$\bar{\mathcal{S}}_j :\ \ (\bar{\boldsymbol{x}} - \bar{\boldsymbol{c}}_j)^T\,(\bar{\boldsymbol{x}} - \bar{\boldsymbol{c}}_j) = 1\,.$$

It suffices now to determine the two transformations T_1 and T_2, corresponding to the two transformations introduced in the previous section, to recover the center $\boldsymbol{c}_i$ of $\mathcal{E}_i$, in other words, to transform the ellipsoid $\bar{\mathcal{E}}_i$ and the sphere $\bar{\mathcal{S}}_j$ into a pair of ellipsoids $\mathcal{E}_i$ and $\mathcal{E}_j$:

1. We recall that the first transformation T_1 corresponds to the change of refer-

ence from $(\bar{\boldsymbol{c}}_i; \bar{x}, \bar{y}, \bar{z})$, where $\bar{\boldsymbol{c}}_i = (0,0,0)$ is the center of the ellipsoid $\bar{\mathcal{E}}_i$, to $(\hat{\boldsymbol{c}}_j; \hat{x}, \hat{y}, \hat{z})$, where $\hat{c}_j = (0,0,0)$ is a center of the new sphere $\hat{\mathcal{S}}_j$. The transformation thus consists in the translation of vector $-\bar{\boldsymbol{c}}_j$ followed by the rotation $\hat{\mathcal{R}}_i = \mathcal{R}(\hat{\theta}_{xi}, \hat{\theta}_{yi}, \hat{\theta}_{zi})$ using the three Euler angles $\hat{\theta}_{xi}$, $\hat{\theta}_{yi}$, $\hat{\theta}_{zi}$ obtained from the spectral decomposition of $\hat{Q}_i$:

$$\hat{\boldsymbol{x}} = T_1(\bar{\boldsymbol{x}}) = \hat{\mathcal{R}}_i(\bar{\boldsymbol{x}} - \bar{\boldsymbol{c}}_j) \ ,$$

so that the center of $\hat{\mathcal{E}}_i$ is given by

$$\hat{\boldsymbol{c}}_i = -\hat{\mathcal{R}}_i \bar{\boldsymbol{c}}_j \ .$$

In the new frame of reference, the ellipsoid $\bar{\mathcal{E}}_i$ and the unit sphere $\bar{\mathcal{C}}_j$ become:

$$\begin{aligned} \hat{\mathcal{E}}_i &: \ (\hat{\boldsymbol{x}} - \hat{\boldsymbol{c}}_i)^T \hat{\mathcal{Q}}_i (\hat{\boldsymbol{x}} - \hat{\boldsymbol{c}}_i) = 1 \ , \\ \hat{\mathcal{S}}_j &: \ \hat{\boldsymbol{x}}^T \hat{\boldsymbol{x}} = 1 \ . \end{aligned} \tag{2.19}$$

2. The second transformation T_2 maps the unit sphere $\hat{\mathcal{S}}_j$ to an ellipsoid $\mathcal{E}_j$ of volume ratio $\omega_j = 1$. Because we have a choice of units, any parameter could be fixed to a constant, here we fix the value of ω_j allowing for simplifications further down the line. Using $\boldsymbol{c}_j \in \mathbb{R}^3$, the transformation reads:

$$\boldsymbol{x} = T_2(\hat{\boldsymbol{x}}) = \mathcal{R}_j \mathcal{D}_j^{-1/2} \hat{\boldsymbol{x}} + \boldsymbol{c}_j \ ,$$

and allows one to compute the center $\boldsymbol{c}_i$ of $\mathcal{E}_i$ as:

$$\boldsymbol{c}_i = T_2(\hat{\boldsymbol{c}}_i) = \mathcal{R}_j \mathcal{D}_j^{-1/2} \hat{\boldsymbol{c}}_i + \boldsymbol{c}_j \ .$$

Therefore, the second transformation sends the ellipsoid $\hat{\mathcal{E}}_i$ to $\mathcal{E}_i$ and the sphere $\hat{\mathcal{S}}_j$ to $\mathcal{E}_j$ such as:

$$\begin{aligned} \mathcal{E}_i &: \ (\boldsymbol{x} - \boldsymbol{c}_i)^T \mathcal{Q}_i (\boldsymbol{x} - \boldsymbol{c}_i) = 1 \ , \\ \mathcal{E}_j &: \ (\boldsymbol{x} - \boldsymbol{c}_j)^T \mathcal{Q}_j (\boldsymbol{x} - \boldsymbol{c}_j) = 1 \ , \end{aligned}$$

where $Q_i = \mathcal{R}_i \mathcal{D}_i \mathcal{R}_i^T$ and $Q_j = \mathcal{R}_j \mathcal{D}_j \mathcal{R}_j^T$.

Finally, the exact distance between the ellipsoids $\mathcal{E}_i$ and $\mathcal{E}_j$ is computed as $\varepsilon = \|\boldsymbol{x}_j - \boldsymbol{x}_i\|$, where the point $\boldsymbol{x}_i \in \mathcal{E}_i$ is mapped from $\bar{\boldsymbol{x}}_i$ using the two transformations T_1 and T_2, i.e. $\boldsymbol{x}_i = T_2 \circ T_1(\bar{\boldsymbol{x}}_i)$, and the closest point $\boldsymbol{x}_j \in \mathcal{E}_j$ is obtained by the

algorithm described in [17]. The 3D algorithm is detailed in Algorithm 3.

Algorithm 3: Sampling algorithm for a pair of ellipsoids

Data: γ_{1i}, γ_{2i}, ω_i, θ_{xi}, θ_{yi}, θ_{zi}, γ_{1j}, γ_{2j}, θ_{xj}, θ_{yj}, θ_{zj}, ϕ, ψ, $\bar{\varepsilon}$, $\boldsymbol{c}_j$

Result: $\boldsymbol{c}_i$, $\boldsymbol{x}_i$, $\boldsymbol{x}_j$, ε

compute $a_i = \sqrt[3]{\omega_i \gamma_{1i}^2 \gamma_{2i}}$, $b_i = \sqrt[3]{\omega_i \gamma_{2i}/\gamma_{1i}}$, $c_i = \sqrt[3]{\omega_i/(\gamma_{1i}\gamma_{2i}^2)}$;

compute $\mathcal{D}_i = \mathcal{D}(a_i, b_i, c_i)$, $\mathcal{R}_i = \mathcal{R}(\theta_{xi}, \theta_{yi}, \theta_{z_i})$;

compute $a_j = \sqrt[3]{\gamma_{1j}^2 \gamma_{2j}}$, $b_j = \sqrt[3]{\gamma_{2j}/\gamma_{1j}}$, $c_j = 1/(a_j b_j)$;

compute $\mathcal{D}_j = \mathcal{D}(a_j, b_j, c_j)$, $\mathcal{R}_j = \mathcal{R}(\theta_{xj}, \theta_{yj}, \theta_{zj})$;

compute $\hat{\mathcal{Q}}_i = \mathcal{D}_j^{-1/2} \mathcal{R}_j^T \mathcal{R}_i \mathcal{D}_i \mathcal{R}_i^T \mathcal{R}_j \mathcal{D}_j^{-1/2}$;

by spectral decomposition of $\hat{\mathcal{Q}}_i = \hat{\mathcal{R}}_i \hat{\mathcal{D}}_i \hat{\mathcal{R}}_i^T$, identify $\bar{a}_i = \hat{a}_i = 1/\sqrt{\lambda_1}$, $\bar{b}_i = \hat{b}_i = 1/\sqrt{\lambda_2}$, $\bar{c}_i = \hat{c}_i = 1/\sqrt{\lambda_3}$, the Euler angles $\hat{\theta}_{xi}$, $\hat{\theta}_{yi}$, $\hat{\theta}_{zi}$, and compute $\hat{\mathcal{R}}_i = \mathcal{R}(\hat{\theta}_{xi}, \hat{\theta}_{yi}, \hat{\theta}_{zi})$;

compute $\bar{\boldsymbol{x}}_i = (\bar{a}_i \cos\phi \cos\psi, \bar{b}_i \sin\phi \cos\psi, \bar{c}_i \sin\psi)$;

compute $\bar{\boldsymbol{n}}_i = \dfrac{\nabla \bar{f}_i(\bar{\boldsymbol{x}}_i)}{\|\nabla \bar{f}_i(\bar{\boldsymbol{x}}_i)\|}$;

compute $\bar{\boldsymbol{c}}_j = \bar{\boldsymbol{x}}_i + (1 + \bar{\varepsilon})\, \bar{\boldsymbol{n}}_i$;

compute $\boldsymbol{c}_i = T_2 \circ T_1(\boldsymbol{0})$;

compute $\boldsymbol{x}_i = T_2 \circ T_1(\bar{\boldsymbol{x}}_i)$;

solve for point $\boldsymbol{x}_j$ [17] ;

compute $\varepsilon = \|\boldsymbol{x}_j - \boldsymbol{x}_i\|$

2.4 Numerical Experiments

Given that most parameters in this new algorithm are imposed as inputs, the only parameters that are outputs are the coordinates of the center $\boldsymbol{c}_i$ of $\mathcal{E}_i$, the closest points $\boldsymbol{x}_i = T_2 \circ T_1(\bar{\boldsymbol{x}}_i)$ and $\boldsymbol{x}_j$, computed with the contact detection algorithm [17], and the overlap $\varepsilon = \|\boldsymbol{x}_j - \boldsymbol{x}_i\|$ as the image of $\bar{\varepsilon}$. In any case, this overlap being the raison d'être of this whole algorithm, it follows naturally that we should investigate the distribution of ε, especially as a function of the nominal distance $\bar{\varepsilon}$.

2.4.1 Distribution of penetration distance for pairs of ellipses

In this example, we select values of $\bar{\varepsilon} = -0.01$, -0.02, -0.1, -0.5, and -1.0, and, for each value of $\bar{\varepsilon}$, we produce 10^4 pairs of ellipses. We then study the distribution of the true distance ε. The ellipses were generated using the following distributions of

the shape parameters:

$$\begin{aligned} &\gamma_i \sim \mathcal{U}([1,20])\,, \qquad \omega_i \sim \mathcal{U}([1,20])\,, \qquad \theta_i \sim \mathcal{U}(]0,\pi])\,, \\ &\gamma_j \sim \mathcal{U}([1,20])\,, \qquad \omega_j = 1\,, \qquad\qquad\quad \theta_j \sim \mathcal{U}(]0,\pi])\,, \end{aligned}$$

and of the positioning parameters:

$$\begin{aligned} &\phi \sim \mathcal{U}(]-\pi,\pi])\,, \\ &\boldsymbol{c}_j \sim \mathcal{U}(\{\boldsymbol{x} \in \mathbb{R}^2 \;;\; \|\boldsymbol{x}\|^2 \leq 1\})\,. \end{aligned}$$

The results are reported in Table 2.1. In the first two rows of the table, we observe that the average $\mathbb{E}(|\varepsilon|)$ of the penetration distance is consistently about 86% of the value of $|\bar{\varepsilon}|$ with a standard deviation $\sigma(|\varepsilon|)$ roughly equal to 28% of the nominal value of $|\bar{\varepsilon}|$. These results indicate that the order of magnitude of ε is well-determined by $\bar{\varepsilon}$, even for a wide-range of pairs of ellipses. In order to compare values of ε taking into account that the ellipses may be of different sizes, a normalization factor is introduced: the distance on the *co-gradient locus* between the two centers of the ellipses. The co-gradient locus, whose theory is introduced in [14], is in 2D a hyperbola with a branch connecting the two centers that also contains the contact points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ as is shown in Figure 3.2. The theory indicates that an appropriate distance ε would scale with

$$\int_0^1 \|\boldsymbol{x}'(t)\|\; dt \simeq \|\boldsymbol{x}_i - \boldsymbol{c}_i\| + \|\boldsymbol{x}_j - \boldsymbol{c}_j\|\,,$$

where $\boldsymbol{x}(t)$ is the parameterization from $[0,1]$ to $\mathbb{R}^2$ of the co-gradient locus between the two centers $\boldsymbol{c}_i$ and $\boldsymbol{c}_j$. The normalization based on these ideas thus provides the normalized distances $\bar{\varepsilon}_n$ and ε_n:

$$\bar{\varepsilon}_n := \frac{|\bar{\varepsilon}|}{\|\bar{\boldsymbol{x}}_i\| + 1}\,, \qquad \varepsilon_n := \frac{|\varepsilon|}{\|\boldsymbol{x}_i - \boldsymbol{c}_i\| + \|\boldsymbol{x}_j - \boldsymbol{c}_j\|}\,,$$

in the ellipse-circle configuration and ellipse-ellipse configuration, respectively. We observe now from Table 2.1 that the values of $\mathbb{E}(\varepsilon_n)$ gets sometimes closer to $\mathbb{E}(\bar{\varepsilon}_n)$, in particular for larger values of $\bar{\varepsilon}_n$ (except here when $|\bar{\varepsilon}| = 1.0$). Moreover, the standard deviations $\sigma(\bar{\varepsilon}_n)$ and $\sigma(\varepsilon_n)$ are consistently smaller than those of $|\varepsilon|$. We also observe that the order of magnitude of the distances are well determined by $\bar{\varepsilon}$. The averaged distances $\mathbb{E}(\bar{\varepsilon}_n)$ are indeed all about 28% of $|\bar{\varepsilon}|$.

Table 2.1 Distribution of the penetration distance ε for fixed values of $\bar{\varepsilon}$ using 10^4 data points

$\lvert\bar{\varepsilon}\rvert$	0.01	0.02	0.1	0.5	1.0
$\mathbb{E}(\lvert\varepsilon\rvert)$	0.008592	0.017075	0.086249	0.432221	0.862348
$\sigma(\lvert\varepsilon\rvert)$	0.002780	0.005707	0.029575	0.146265	0.283891
$\mathbb{E}(\bar{\varepsilon}_n)$	0.002753	0.005307	0.028316	0.141572	0.282968
$\sigma(\bar{\varepsilon}_n)$	0.000926	0.000787	0.004621	0.009286	0.018613
$\mathbb{E}(\varepsilon_n)$	0.003112	0.006215	0.028432	0.139525	0.320864
$\sigma(\varepsilon_n)$	0.001947	0.001130	0.002937	0.034669	0.046312

2.4.2 Distribution of penetration distance for pairs of ellipsoids

We repeat here a similar experiment but this time for pairs of ellipsoids in 3D. The ellipsoids were generated using the following distributions of the shape parameters:

$$\begin{aligned}
&\gamma_{1i} \sim \mathcal{U}([1,6])\,, && \gamma_{2i} \sim \mathcal{U}([1,3])\,, && \omega_i \sim \mathcal{U}([1,5])\,,\\
&\theta_{xi} \sim \mathcal{U}(]-\pi,\pi])\,, && \theta_{yi} \sim \mathcal{U}([0,\pi])\,, && \theta_{zi} \sim \mathcal{U}(]-\pi,\pi])\,,\\
&\gamma_{1j} \sim \mathcal{U}([1,5])\,, && \gamma_{2j} \sim \mathcal{U}([1,4])\,, && \omega_j = 1\,,\\
&\theta_{xj} \sim \mathcal{U}(]-\pi,\pi])\,, && \theta_{yj} \sim \mathcal{U}([0,\pi])\,, && \theta_{zj} \sim \mathcal{U}(]-\pi,\pi])\,,
\end{aligned}$$

and of the positioning parameters:

$$\begin{aligned}
&\phi \sim \mathcal{U}(]-\pi,\pi])\,,\\
&\psi \sim \mathcal{U}([0,\pi])\,,\\
&\boldsymbol{c}_j \sim \mathcal{U}(\{\boldsymbol{x} \in \mathbb{R}^3\ ;\ \|\boldsymbol{x}\|^2 \leq 1\})\,.
\end{aligned}$$

Moreover, we use a uniform distribution of $\log|\bar{\varepsilon}|$ over one order of magnitude, i.e. for values between 10^{-1} and 10^{-2}, which is seen in the top left graph of Figure 2.7. The distribution of $|\bar{\varepsilon}|$ (without the logarithm) is seen in the first column, second row of Figure 2.7. The corresponding distributions of the distance $|\epsilon|$ are shown in the third and fourth rows of the first column. We observe that the distributions for $\bar{\varepsilon}$ and ε are slightly different. The differences are much less pronounced when we consider their normalized values. In fact, the first two plots in the second column are $-\log|\bar{\varepsilon}_n|$ and $|\bar{\varepsilon}_n|$ while the next two in the second column are $|\varepsilon_n|$ and $-\log|\varepsilon_n|$. For both normalized variables, we observe little change in the underlying distributions.

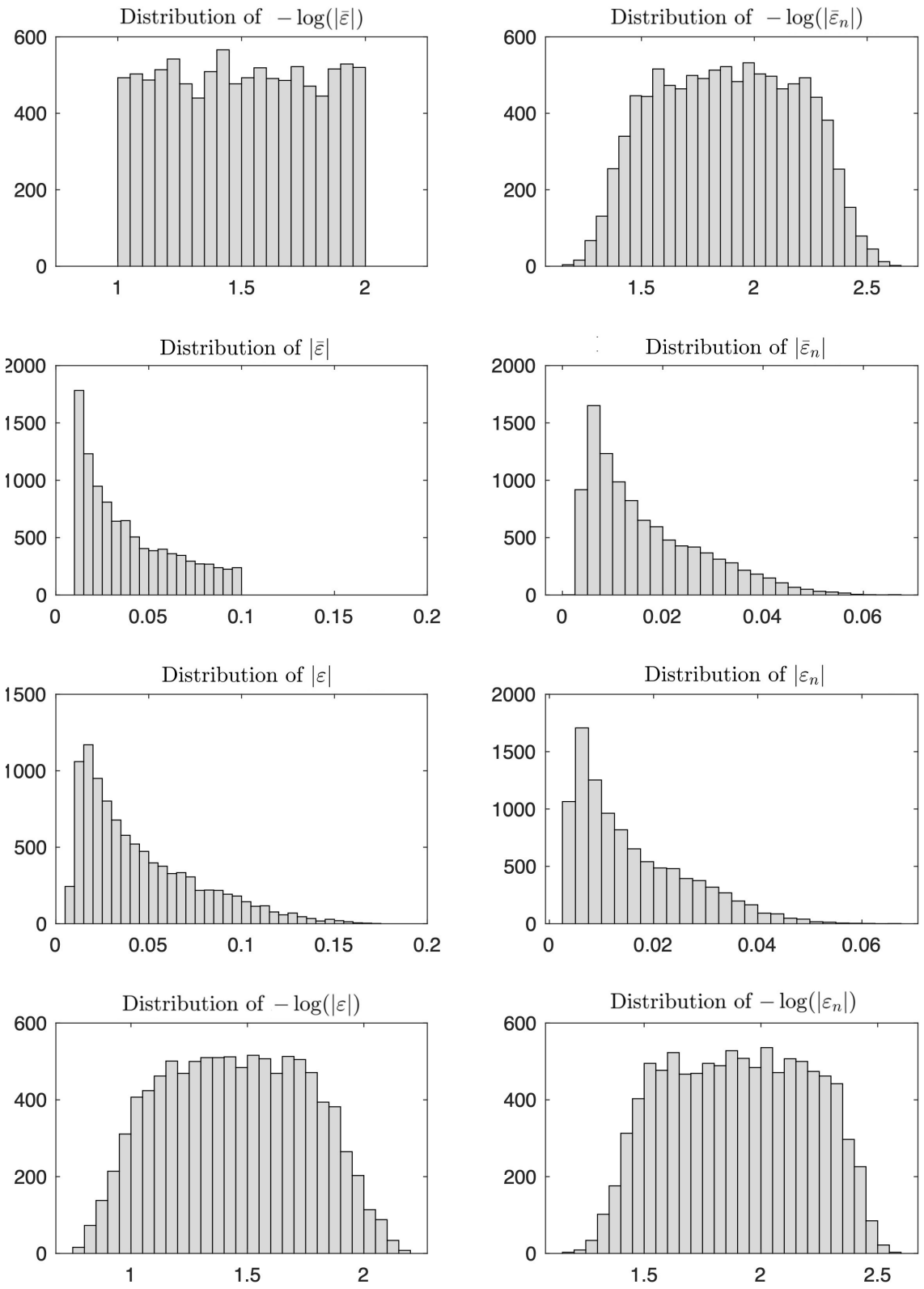


Figure 2.7 Distributions of $\bar{\varepsilon}$ and ε as simulated on 1000 pairs of ellipsoids.

CHAPTER 3 NOVEL ALGORITHM FOR CONTACT DETECTION BETWEEN ELLIPSES AND ELLIPSOIDS

3.1 Introduction

Computer simulations of the dynamical behavior of granular materials, such as sand particles in soil mechanics or powders in chemical engineering, are usually performed using the so-called Discrete Element Method (DEM), or synonymously the Distinct Element Method, that was introduced in the seminal work of Cundall and Strack [5]. In this method, the particles are usually modeled as discs and spheres of variable size, the main reasons being that the definition of contact between spherical particles is unequivocal and that finding the contact points is a straightforward task. However, it has been shown that particle shape may have a significant impact on the prediction of macroscopic properties in static or dynamic behavior of particle assemblies [20,21]. Alternatives to discs and spheres for the simulation of soil mechanics problems are obviously ellipses [22] and ellipsoids [23–25], respectively. Unfortunately, the notion of contact between two overlapping ellipsoids is not uniquely defined and none of the definitions allows one to find analytical formulas to compute the contact points. The problems of finding contact points between ellipsoidal particles are typically nonlinear and numerical methods must be used to find approximations of the contact points. It is therefore of paramount importance to design efficient and robust algorithms to limit the computational bottleneck of estimating the contact points when dealing with a very large number of particles.

Several algorithms for computing contact points between overlapping ellipsoidal particles have been proposed in the last thirty years and a recent review paper [26] has classified the different approaches into three main families of methods depending on the definition of the contact point: 1) the Intersection Method [15, 27, 28], 2) the Geometric Potential Method [18,22,29,30], and 3) the Common Normal Method [30] including the Closest Co-normal Algorithm [16]. The Geometric Potential Method formulates the contact detection problem as two decoupled minimization problems, one for each point of the contact pair. Each of the two points is defined as the closest point on one ellipse to the center of the other ellipse. The Common Normal problem is formulated as a coupled minimization problem for the contact pair, which makes the method more computationally expensive than other methods [30]. On the one hand, when the ellipsoids are separated or in perfect contact, the contact pair in the

Geometric Potential method and Common Normal method coincides. On the other hand, if they overlap, the contact pairs may be different. The most computationally effective algorithms are found to be those based on the Geometric Potential Method.

Following ideas presented in [31] and in the review paper [26], a new algorithm, named the Steered Geometrical Potential Algorithm (S-GPA), for contact detection between pairs of ellipses and ellipsoids was developed in [32] and shown to be more efficient than existing approaches. The algorithm primarily consists in parametrizing each of the ellipsoidal particles and find the contact point as an intersection point between the ellipse or ellipsoid and the so-called co-gradient locus, see [26]. In two dimensions, the co-gradient locus is shown in [26] to be an hyperbola that intersects with one of the two ellipses at at most four points, one of them being the desired contact point. The efficiency of the algorithm relies on the following ingredients: 1) a transformation that maps the pair of ellipses (ellipsoids) into an ellipse (ellipsoid) centered at the origin and a unit circle (sphere); 2) the construction in the transformed configuration of an effective initial guess to the solution of the minimization problem; 3) the use of Newton's method for the root finding problem; 4) and the introduction of an additional constraint to guarantee convergence to the desired point. Although the algorithm was found to be more efficient than existing approaches, most of its computational time was spent for the calculation of the transformations.

We propose in this paper an alternative approach to the above algorithm. It consists this time in parametrizing the co-gradient locus and in finding the pair of points that intersect with the two ellipsoidal particles. The main advantage of the algorithm is that the transformations are not needed any longer. Moreover, the contact points on the co-gradient locus are known to be unique in the portion of the co-gradient locus between the two centers of the ellipsoidal particles. The method presents some similarities with that described in the recent paper by Choi [33], although the objective was to find the closest approach distance between two ellipsoids within a user-specified tolerance and thus search for the point at which two ellipsoids are in perfect contact.

The paper is organized as follows. We briefly present some notations and recall some useful properties about ellipses and ellipsoids in Section 3.2. The geometrical potential method and co-gradient locus for pairs of ellipses are summarized in Section 3.3. The new algorithm is described in detail in Section 3.4 for pairs of ellipses and is extended to the case of ellipsoids in Section 3.5. We present some numerical examples that compare the performance of the algorithm with the Steered Geometrical Potential Algorithm in Section 3.6 before providing some concluding remarks in Section 3.7.

3.2 Preliminaries and notations

We introduce in this section the general notation and some preliminary notions about ellipses and ellipsoids. For convenience, the presentation closely follows that provided in [32], where we first considered the case of ellipses and then briefly extended the definitions and properties to ellipsoids.

An ellipse $\mathcal{E}$ in the Cartesian coordinate system (O, x, y) is defined as the set of points

$$\mathcal{E} = \{\boldsymbol{x} \in \mathbb{R}^2; (\boldsymbol{x} - \boldsymbol{c})^T \mathcal{Q}(\boldsymbol{x} - \boldsymbol{c}) = 1\}, \tag{3.1}$$

where $\mathcal{Q}$ is a symmetric positive-definite matrix in $\mathbb{R}^{2\times 2}$ and $\boldsymbol{c} \in \mathbb{R}^2$ denotes the center of the ellipse. Alternatively, the ellipse can be defined as the set of roots of the so-called geometric potential function

$$f(\boldsymbol{x}) = (\boldsymbol{x} - \boldsymbol{c})^T \mathcal{Q}(\boldsymbol{x} - \boldsymbol{c}) - 1. \tag{3.2}$$

Using the potential function, the unit outward normal vector to the ellipse at $\boldsymbol{x} \in \mathcal{E}$ can be computed as

$$\boldsymbol{n}(\boldsymbol{x}) = \frac{\nabla f(\boldsymbol{x})}{\|\nabla f(\boldsymbol{x})\|} = \frac{Q(\boldsymbol{x} - \boldsymbol{c})}{\|Q(\boldsymbol{x} - \boldsymbol{c})\|}, \tag{3.3}$$

where $\|\cdot\|$ denotes the usual Euclidean norm. We note here that the gradient of the potential function, $\nabla f(\boldsymbol{x})$, provides a vector field in $\mathbb{R}^2$. In particular, $\nabla f(\boldsymbol{c}) = \boldsymbol{0}$. In fact, given an ellipse $\mathcal{E}$ and a point $\boldsymbol{x} \in \mathbb{R}^2 \backslash \{\boldsymbol{c}\}$, the gradient $\nabla f(\boldsymbol{x})$ represents an outward normal vector at $\boldsymbol{x}$ to the ellipse centered at $\boldsymbol{c}$, that is aligned with $\mathcal{E}$, has the same aspect ratio, and passes through $\boldsymbol{x}$. Similarly, we can define the unit vector field $\boldsymbol{n}(\boldsymbol{x})$ (3.3) for all $\boldsymbol{x} \in \mathbb{R}^2 \backslash \{\boldsymbol{c}\}$.

Since $\mathcal{Q}$ is a symmetric positive-definite matrix, we can always associate with an ellipse the so-called $\mathcal{E}$-norm,

$$\|\boldsymbol{x}\|_{\mathcal{E}} = \sqrt{\boldsymbol{x}^T \mathcal{Q} \boldsymbol{x}}, \tag{3.4}$$

which allows one to interpret $\mathcal{E}$ as the "unit circle" satisfying $\|\boldsymbol{x} - \boldsymbol{c}\|_{\mathcal{E}}^2 = (\boldsymbol{x} - \boldsymbol{c})^T \mathcal{Q}(\boldsymbol{x} - \boldsymbol{c}) = 1$.

The singular-value decomposition of $\mathcal{Q}$ provides a diagonal matrix $\mathcal{D}$ and an orthogonal matrix $\mathcal{R}$, i.e. $\mathcal{R}^{-1} = \mathcal{R}^T$, such that

$$\mathcal{Q} = \mathcal{R}\mathcal{D}\mathcal{R}^T. \tag{3.5}$$

The diagonal matrix $\mathcal{D}$ consists of the positive eigenvalues of $\mathcal{Q}$, λ_1 and λ_2, ordered as $0 < \lambda_1 \leq \lambda_2$. Introducing the lengths of the semi-axes $a = 1/\sqrt{\lambda_1}$ and $b = 1/\sqrt{\lambda_2}$, one has

$$\mathcal{D} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} = \begin{bmatrix} 1/a^2 & 0 \\ 0 & 1/b^2 \end{bmatrix}.$$

The orthogonal matrix $\mathcal{R}$ is constructed from the eigenvectors of $\mathcal{Q}$ in the appropriate order of the eigenvalues. It corresponds in 2D to a rotation by the angle $\theta \in [-\pi, \pi[$, which is defined as the angle between the x-axis of the coordinate system and the eigenvector associated with λ_1, i.e. the major semi-axis a:

$$\mathcal{R} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}. \tag{3.6}$$

Introducing the transformation composed of the translation by the vector $-\boldsymbol{c}$ and the rotation by an angle $-\theta$, that is,

$$\boldsymbol{\xi} = \mathcal{R}^T(\boldsymbol{x} - \boldsymbol{c}), \tag{3.7}$$

one can rewrite the geometric potential function (3.2) as

$$\bar{f}(\boldsymbol{\xi}) = \boldsymbol{\xi}^T \mathcal{D} \boldsymbol{\xi} - 1. \tag{3.8}$$

Therefore, the ellipse $\mathcal{E}$ can be alternatively defined as the set of roots of $\bar{f}$, that is, the set of points $\boldsymbol{\xi} = [\xi, \eta]^T \in \mathbb{R}^2$ that satisfy:

$$\frac{\xi^2}{a^2} + \frac{\eta^2}{b^2} = 1, \tag{3.9}$$

which is the classical equation of an ellipse in its local coordinate system (ξ, η) whose origin is the center $\boldsymbol{c}$ of the ellipse, see Figure 3.1.

Similarly, we define an ellipsoid $\mathcal{E}$ in the Cartesian coordinate system (O, x, y, z) as the surface in $\mathbb{R}^3$ such that:

$$\mathcal{E} = \{\boldsymbol{x} \in \mathbb{R}^3; (\boldsymbol{x} - \boldsymbol{c})^T \mathcal{Q} (\boldsymbol{x} - \boldsymbol{c}) = 1\}, \tag{3.10}$$

where $\mathcal{Q}$ is a symmetric positive-definite matrix in $\mathbb{R}^{3\times3}$ and $\boldsymbol{c} \in \mathbb{R}^3$ is the center of the ellipsoid. As for the ellipse, one can define the geometric potential function $f(\boldsymbol{x})$ (3.2), the unit outward normal vector (3.3), and the $\mathcal{E}$-norm (3.4). The matrix $\mathcal{Q}$ can also

be decomposed as $\mathcal{Q} = \mathcal{R}\mathcal{D}\mathcal{R}^T$, where the diagonal matrix $\mathcal{D}$ is formed of the three positive eigenvalues of $\mathcal{Q}$, namely λ_1, λ_2, and λ_3, ordered as $0 < \lambda_1 \leq \lambda_2 \leq \lambda_3$, from which one can calculate the so-called lengths of the semi-axes $a = 1/\sqrt{\lambda_1}$, $b = 1/\sqrt{\lambda_2}$, and $c = 1/\sqrt{\lambda_3}$. Through the change of variables (3.7), one can derive the classical equation of the ellipsoid in the local coordinate system (ξ, η, ζ) centered at $\boldsymbol{c}$ as:

$$\frac{\xi^2}{a^2} + \frac{\eta^2}{b^2} + \frac{\zeta^2}{c^2} = 1. \tag{3.11}$$

Finally, we note that there are several ways to write the rotation matrix $\mathcal{R}$ in 3D. For instance, one may consider the form

$$\mathcal{R} = \mathcal{R}(\theta_x, \theta_y, \theta_z) = \begin{bmatrix} \cos\theta_z & -\sin\theta_z & 0 \\ \sin\theta_z & \cos\theta_z & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta_y & 0 & \sin\theta_y \\ 0 & 1 & 0 \\ -\sin\theta_y & 0 & \cos\theta_y \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_x & -\sin\theta_x \\ 0 & \sin\theta_x & \cos\theta_x \end{bmatrix},$$

where $(\theta_x, \theta_y, \theta_z) \in]-\pi, \pi] \times [0, \pi] \times]-\pi, \pi]$ are the so-called Euler angles that describe the orientation of that ellipsoid with respect to a fixed coordinate system.

3.3 The geometrical potential method and the co-gradient locus for a pair of ellipses

We consider in this section a pair of ellipses $\mathcal{E}_i$ and $\mathcal{E}_j$. We assume that the two ellipses are constructed such that the center of one ellipse cannot lie within the other ellipse, and vice versa, i.e. $\|\boldsymbol{c}_j - \boldsymbol{c}_i\|_{\mathcal{E}_i} > 1$ and $\|\boldsymbol{c}_j - \boldsymbol{c}_i\|_{\mathcal{E}_j} > 1$. We further suppose that they are in near perfect contact in the sense defined in [26], meaning that the ellipses are fully disjoint, or are in perfect contact, or exhibit a small overlap. This ensures that the intersection set between $\mathcal{E}_i$ and $\mathcal{E}_j$ consists of the empty set, or reduces to a single point, the so-called contact point, or is exactly formed of two points, respectively.

The objective in this paper is to present an algorithm that efficiently computes the distance between the two ellipses in the case they are separated or a measure of the overlap in the case they slightly overlap. However, as mentioned in the introduction, the measure of overlap is not unique and depends on the method that one chooses to estimate it. We consider here the definition based on the geometrical potential method, see [26], that consists in finding a pair of points, $\boldsymbol{x}_i \in \mathcal{E}_i$ and $\boldsymbol{x}_j \in \mathcal{E}_j$, which

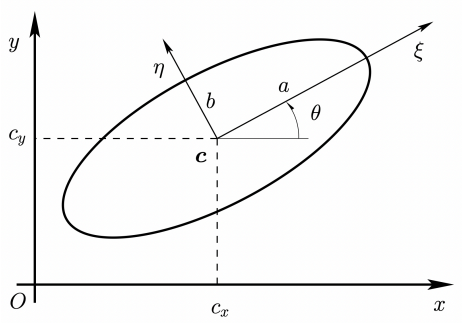


Figure 3.1 Example of an ellipse $\mathcal{E}$. The ellipse is fully described in terms of five parameters: the major semi-axis a and minor semi-axes b, the angle θ, and the coordinates of the center $\boldsymbol{c} = (c_x, c_y)$. Alternatively, it can be described by the three entries of Q and the center $\boldsymbol{c}$.

satisfy the minimization problems:

$$\boldsymbol{x}_i = \underset{\boldsymbol{x}\in\mathcal{E}_i}{\operatorname{argmin}} \|\boldsymbol{x} - \boldsymbol{c}_j\|_{\mathcal{E}_j} = \underset{\boldsymbol{x}\in\mathcal{E}_i}{\operatorname{argmin}} f_j(\boldsymbol{x}), \tag{3.12}$$

$$\boldsymbol{x}_j = \underset{\boldsymbol{x}\in\mathcal{E}_j}{\operatorname{argmin}} \|\boldsymbol{x} - \boldsymbol{c}_i\|_{\mathcal{E}_i} = \underset{\boldsymbol{x}\in\mathcal{E}_j}{\operatorname{argmin}} f_i(\boldsymbol{x}), \tag{3.13}$$

where f_i and f_j are the geometric potential functions associated with $\mathcal{E}_i$ and $\mathcal{E}_j$, respectively. In other words, the point $\boldsymbol{x}_i \in \mathcal{E}_i$ is defined as the closest point to the center $\boldsymbol{c}_j$ of ellipse $\mathcal{E}_j$, with respect to the $\mathcal{E}_j$-norm, while the point $\boldsymbol{x}_j \in \mathcal{E}_j$ is the closest point to the center $\boldsymbol{c}_i$ of $\mathcal{E}_i$, with respect to the $\mathcal{E}_i$-norm. The attractive feature of this approach is that the pair of points, referred here as the contact pair, are solutions to uncoupled minimization problems that can be solved separately. Moreover, for a pair of ellipse in near perfect contact, the solution to each problem exists and is unique [26]. In the case of two ellipses in perfect contact, the two points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ actually coincide and correspond to the contact point $\boldsymbol{x}_c$ between the ellipses. If the two ellipses $\mathcal{E}_i$ and $\mathcal{E}_j$ are separated or overlap, basically in all configurations of the ellipses considered here, the separation or penetration distance can simply be computed as $\varepsilon = \|\boldsymbol{x}_i - \boldsymbol{x}_j\|$.

It is shown in [26] that the solution $\boldsymbol{x}_i$ to (3.12) (and in a similar manner the solution $\boldsymbol{x}_j$ to (3.13)) necessarily satisfies:

$$\boldsymbol{n}_i(\boldsymbol{x}_i) + \boldsymbol{n}_j(\boldsymbol{x}_i) = \boldsymbol{0}. \tag{3.14}$$

The interpretation of this property is that $\mathcal{E}_i$ is co-tangent at $\boldsymbol{x}_i$ with a scaled ellipse

of $\mathcal{E}_j$ (centered at $\boldsymbol{c}_j$, aligned with $\mathcal{E}_j$, and of same aspect ratio a_j/b_j). This condition on the normals satisfied by $\boldsymbol{x}_i$ could be added as a non-binding constraint to the minimization problem (3.12). However, this condition can be relaxed by taking its cross-product with $\boldsymbol{n}_i(\boldsymbol{x}_i)$, so that

$$\boldsymbol{n}_i(\boldsymbol{x}_i) \times \boldsymbol{n}_j(\boldsymbol{x}_i) = 0, \tag{3.15}$$

or, equivalently:

$$\nabla f_i(\boldsymbol{x}_i) \times \nabla f_j(\boldsymbol{x}_i) = 0. \tag{3.16}$$

We now recall the following definition.

Definition 3.3.1 (Co-gradient function and co-gradient locus)**.** *Given two ellipses $\mathcal{E}_i$ and $\mathcal{E}_j$ with distinct centers, the co-gradient function associated with the pair of ellipses is defined as*

$$H(\boldsymbol{x}) = \nabla f_i(\boldsymbol{x}) \times \nabla f_j(\boldsymbol{x}). \tag{3.17}$$

The co-gradient locus is the set of all roots of the co-gradient function, namely

$$\mathcal{H}_{ij} = \Big\{\boldsymbol{x} \in \mathbb{R}^2;\ H(\boldsymbol{x}) = 0\Big\}. \tag{3.18}$$

We show in Figure 3.2 some examples of the co-gradient locus associated with a given pair of ellipses in near-perfect contact. We note that the cross product of two vectors in 2D can be viewed as a 3D vector with only one non-zero component along the z-axis. The co-gradient function is therefore the scalar function:

$$H(x, y) = \partial_x f_i(x, y)\partial_y f_j(x, y) - \partial_y f_i(x, y)\partial_x f_j(x, y),$$

where $\partial_x f_i(x, y)$ denotes the partial derivative of f_i with respect to x. We have shown in [26] that the co-gradient locus $\mathcal{H}_{ij}$ in 2D is in fact a hyperbola whose one of the two branches necessarily passes through both centers $\boldsymbol{c}_i$ and $\boldsymbol{c}_j$ of the pair of ellipses. Furthermore, from Definition (3.3.1), the solutions $\boldsymbol{x}_i$ to (3.12) and $\boldsymbol{x}_j$ to (3.13) both belong to $\mathcal{H}_{ij}$. More specifically, in the case of ellipses in near-perfect contact, the two points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are both located between $\boldsymbol{c}_i$ and $\boldsymbol{c}_j$ along the branch of the co-gradient locus $\mathcal{H}_{ij}$ that passes through $\boldsymbol{c}_i$ and $\boldsymbol{c}_j$.

Finally, taking the scalar product of (3.14) with $\boldsymbol{n}_j(\boldsymbol{x}_i)$ leads to:

$$\boldsymbol{n}_i(\boldsymbol{x}_i) \cdot \boldsymbol{n}_j(\boldsymbol{x}_i) = -1, \tag{3.19}$$

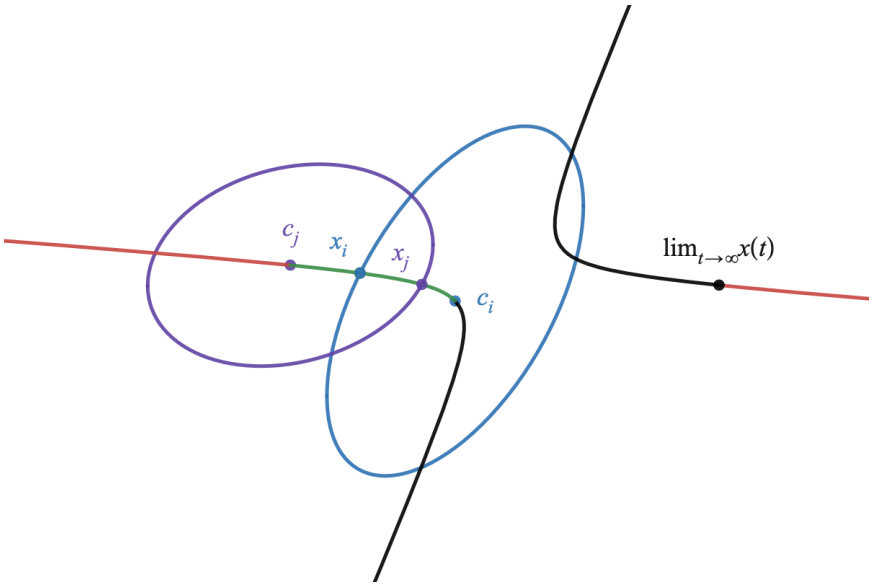


Figure 3.2 Examples of the co-gradient locus associated with a given pair of ellipses in near-perfect contact.

which indicates that the normal vectors $\boldsymbol{n}_i$ and $\boldsymbol{n}_j$ at $\boldsymbol{x}_i$ are of opposite direction. Equivalently, the gradients of the potential functions satisfy:

$$\nabla f_i(\boldsymbol{x}_i) \cdot \nabla f_j(\boldsymbol{x}_i) = -\|\nabla f_i(\boldsymbol{x}_i)\|\|\nabla f_j(\boldsymbol{x}_i)\| \leq 0, \tag{3.20}$$

meaning that the vectors $\nabla f_i(\boldsymbol{x}_i)$ and $\nabla f_j(\boldsymbol{x}_j)$ are linearly dependent and of opposite direction. In fact, for the case of two ellipses in near perfect contact, it was shown in [26] that the points in the section of the co-gradient locus $\mathcal{H}_{ij}$ between the two centers $\boldsymbol{c}_i$ and $\boldsymbol{c}_j$ are the only points $\boldsymbol{x}$ of $\mathcal{H}_{ij}$ satisfying:

$$\boldsymbol{n}_i(\boldsymbol{x}) \cdot \boldsymbol{n}_j(\boldsymbol{x}) = -1.$$

In other words, the point $\boldsymbol{x}_i$ that intersects $\mathcal{E}_i$ and $\mathcal{H}_{ij}$ and satisfies $\boldsymbol{n}_i(\boldsymbol{x}_i) \cdot \boldsymbol{n}_j(\boldsymbol{x}_i) = -1$ is unique and is the solution to the minimization problem (3.12). Similarly, the point $\boldsymbol{x}_j$ that intersects $\mathcal{E}_j$ and $\mathcal{H}_{ij}$ and satisfies $\boldsymbol{n}_i(\boldsymbol{x}_j) \cdot \boldsymbol{n}_j(\boldsymbol{x}_j) = -1$ is also unique and is the solution to the minimization problem (3.13). One can then find $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ by introducing a parametrization of the section between $\boldsymbol{c}_i$ and $\boldsymbol{c}_j$ of the co-gradient locus and searching for the points that intersect $\mathcal{E}_i$ and $\mathcal{E}_j$, respectively. The method is described in the following section.

3.4 Contact detection algorithm

3.4.1 Parametrization of the co-gradient locus

As in Section 3.3, we consider a pair of ellipses $\mathcal{E}_i$ and $\mathcal{E}_j$ in near perfect contact. We construct here a parametrization of the section $\mathcal{S}_{ij}$ between $\boldsymbol{c}_i$ and $\boldsymbol{c}_j$ of the co-gradient locus $\mathcal{H}_{ij}$. Let $\boldsymbol{x} \in \mathcal{S}_{ij}$ such that $\boldsymbol{x} \neq \boldsymbol{c}_i$ and $\boldsymbol{x} \neq \boldsymbol{c}_j$. Since $\boldsymbol{x} \in \mathcal{H}_{ij}$, we know that the vectors $\nabla f_i(\boldsymbol{x}) = 2\mathcal{Q}_i(\boldsymbol{x} - \boldsymbol{c}_i)$ and $\nabla f_j(\boldsymbol{x}) = 2\mathcal{Q}_j(\boldsymbol{x} - \boldsymbol{c}_j)$ are linearly dependent, meaning that there exist two constants $\alpha \neq 0$ and $\beta \neq 0$ such that:

$$\alpha\mathcal{Q}_i(\boldsymbol{x} - \boldsymbol{c}_i) + \beta\mathcal{Q}_j(\boldsymbol{x} - \boldsymbol{c}_j) = \boldsymbol{0}.$$

Moreover, we know that for any point $\boldsymbol{x}$ on $\mathcal{S}_{ij}$, the vectors are of opposite direction, which implies that α and β necessarily satisfy $\alpha\beta > 0$. Rearranging above equation, we get:

$$(\alpha\mathcal{Q}_i + \beta\mathcal{Q}_j)\boldsymbol{x} = \alpha\mathcal{Q}_i(\boldsymbol{c}_i) + \beta\mathcal{Q}_j(\boldsymbol{c}_j).$$

If one chooses $\alpha \geq 0$ and $\beta \geq 0$ such that $\alpha\beta > 0$, the matrix $\alpha\mathcal{Q}_i + \beta\mathcal{Q}_j$ is positive-definite, and hence invertible, since $\mathcal{Q}_i$ and $\mathcal{Q}_j$ are two positive-definite matrices. Therefore,

$$\boldsymbol{x} = (\alpha\mathcal{Q}_i + \beta\mathcal{Q}_j)^{-1}(\alpha\mathcal{Q}_i\boldsymbol{c}_i + \beta\mathcal{Q}_j\boldsymbol{c}_j).$$

In order to parametrize $\mathcal{S}_{ij}$, it suffices to introduce the parameter $t \in (0, 1)$ such that $\alpha = 1 - t$ and $\beta = t$, see [33], so that:

$$\boldsymbol{x}(t) = ((1-t)\mathcal{Q}_i + t\mathcal{Q}_j)^{-1}((1-t)\mathcal{Q}_i\boldsymbol{c}_i + t\mathcal{Q}_j\boldsymbol{c}_j). \tag{3.21}$$

Moreover, we observe that:

$$\lim_{t\to 0} \boldsymbol{x}(t) = \mathcal{Q}_i^{-1}\mathcal{Q}_i\boldsymbol{c}_i = \boldsymbol{c}_i,$$
$$\lim_{t\to 1} \boldsymbol{x}(t) = \mathcal{Q}_j^{-1}\mathcal{Q}_j\boldsymbol{c}_j = \boldsymbol{c}_j.$$

In other words, the parametrization (3.21) defines a bijection from $[0, 1]$ to $\mathcal{S}_{ij}$.

Remark 3.4.1. *If one considers all values $t \in \mathbb{R}$ in (3.21), one actually obtains a parametrization of the whole co-gradient locus. Since it was shown to be a hyperbola in 2D, there should exist two values of t for which the parametrization should be singular, such that one would jump from one branch of the hyperbola to the other. These are given by the values of $t \in \mathbb{R}\backslash[0, 1]$ when the parametrization $\boldsymbol{x}(t)$ is undefined, that*

is, if the matrix $(1-t)\mathcal{Q}_i + t\mathcal{Q}_j$ *is singular, or equivalently:*

$$\det((1-t)\mathcal{Q}_i + t\mathcal{Q}_j) = 0.$$

Above equation provides a polynomial function of degree two in t*, which should admit two roots in the general case.*

3.4.2 Solution of the minimization problems

In order to obtain the solution $\boldsymbol{x}_i$ to the minimization problem (3.12), one needs to find the parameter t_i such that $\boldsymbol{x}(t_i)$ be the intersection point between $\mathcal{S}_{ij}$ and the ellipse $\mathcal{E}_i$. In other words, the problem now reads:

$$\begin{aligned} &\text{Find } t_i \in [0,1] \text{ such that:} \\ &\qquad \hat{f}_i(t_i) \equiv f_i(\boldsymbol{x}(t_i)) = (\boldsymbol{x}(t_i) - \boldsymbol{c}_i)^T \mathcal{Q}_i (\boldsymbol{x}(t_i) - \boldsymbol{c}_i) - 1 = 0, \end{aligned} \tag{3.22}$$

where $\boldsymbol{x}(t_i)$ is given by (3.21) with $t = t_i$. In a similar manner, the solution $\boldsymbol{x}_j$ to (3.13) has to satisfy the problem:

$$\begin{aligned} &\text{Find } t_j \in [0,1] \text{ such that:} \\ &\qquad \hat{f}_j(t_j) \equiv f_j(\boldsymbol{x}(t_j)) = (\boldsymbol{x}(t_j) - \boldsymbol{c}_j)^T \mathcal{Q}_j (\boldsymbol{x}(t_j) - \boldsymbol{c}_j) - 1 = 0. \end{aligned} \tag{3.23}$$

The scalar functions $\hat{f}_i(t)$ and $\hat{f}_j(t)$ are nonlinear functions of $t \in [0,1]$, see Figure 3.3 for a few illustrative examples. Finding the unique roots t_i and t_j of $\hat{f}_i$ and $\hat{f}_j$, respectively, requires using numerical methods such as the bisection method, a fixed point approach, or the Newton method.

Remark 3.4.2. *We note that each evaluation of the functions* $\hat{f}_i(t)$ *and* $\hat{f}_j(t)$ *requires one to compute* $\boldsymbol{x}(t)$ *in* (3.21)*, which involves the inverse of the* 2×2 *matrix* $(1-t)\mathcal{Q}_i + t\mathcal{Q}_j$*. For the sake of efficient calculations, one should simply provide the analytical form of the inverse.*

3.4.3 Root-finding algorithms

We briefly analyze here several root-finding algorithms for the solution of the problems (3.22) and (3.23).

1. **Bisection method.** Considering Problem (3.22), we first observe that $\hat{f}_i(0) = f_i(\boldsymbol{c}_i) = -1$. Since we have assumed that the center $\boldsymbol{c}_j$ of $\mathcal{E}_j$ lies outside $\mathcal{E}_i$, we

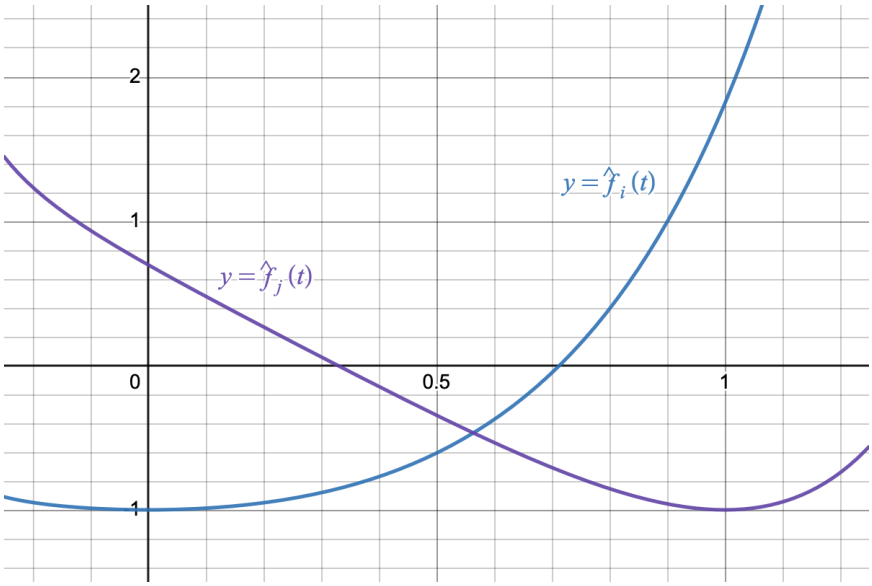


Figure 3.3 Examples of functions $\hat{f}_i(t)$ and $\hat{f}_j(t)$ on $[0, 1]$ associated with the pair of ellipses shown in Fig. 3.2.

also have:

$$\hat{f}_i(1) = f_i(\boldsymbol{c}_j) = \|\boldsymbol{c}_j - \boldsymbol{c}_i\|^2_{\mathcal{E}_i} - 1 > 0.$$

Moreover, the function $\hat{f}_i$ is continuous in $[0, 1]$ and necessarily monotonically increasing when t goes from zero to one. It follows that the bisection method guarantees to find the unique root of $\hat{f}_i$ in $[0, 1]$. However, it is well known that the method is a low-order method as one would need to perform k iterations to obtain a precision of 2^{-k}, that is, 40 iterations if we set the tolerance on the error as $\tau = 10^{-12}$. For better convergence, one can use a hybrid root-finding algorithm such as the Brent's method [34] that combines the bisection method and the secant method, or the Chandrupatla's method [35], which is faster for functions that are flat around the root, or the Interpolate Truncate and Project (ITP) method. The same reasoning applies for finding the root t_j of $\hat{f}_j$ in Problem (3.23).

2. **Fixed-point algorithm.** The root-finding problem (3.22) can be transformed into a fixed-point iteration problem by considering the new scalar function:

$$\hat{g}_i(t) = t - \lambda \hat{f}_i(t).$$

In order for the method to be first-order convergent, it suffices to choose the value of λ such that $g'(t_i) < 1$ so that the fixed point t_i be attractif.

3. **Newton method.** The Newton method is a special case of the fixed-point method where λ is chosen as $1/\hat{f}_i'(t)$, if $\hat{f}_i'(t_i) \neq 0$. Given an initial guess t_i^0, the method consists in finding the next iterates t_i^n, $n = 1, 2, \ldots$, such that:

$$t_i^{n+1} = t_i^n - \frac{\hat{f}_i(t_i^n)}{\hat{f}_i'(t_i^n)}.$$

It is a second-order method if the multiplicity of the root is one, as is the case here. However, there is an extra cost as one needs to evaluate the derivative of $\hat{f}_i$ at each iteration. Unfortunately, the issue is that the derivative $\hat{f}_i$ at a point t may sometimes be very small depending on the configuration of the two ellipses, which would send the next iterate outside the interval $[0, 1]$. The Newton method in that case may not converge to the right point t_i. The key issue here is to provide an initial guess point t_i^0 that falls within the basin of attraction of t_i. One could devise a hybrid approach, in which a few iterations of the bisection method could be used to find an appropriate initial guess.

3.5 Extension to ellipsoids

The contact detection algorithm in the case of two ellipsoids is similar to that of two ellipses. Here we only concentrate on the distinctive features of the algorithm in 3D. We thus consider two arbitrary ellipsoids $\mathcal{E}_i$ and $\mathcal{E}_i$ in near perfect contact with associated geometric potential functions f_i and f_j. In that case, the co-gradient vector-valued function (see Definition 3.3.1) is given by:

$$\boldsymbol{H}(\boldsymbol{x}) = \nabla f_i(\boldsymbol{x}) \times \nabla f_j(\boldsymbol{x}) = 2Q_i(\boldsymbol{x} - \boldsymbol{c}_i) \times 2Q_j(\boldsymbol{x} - \boldsymbol{c}_j),$$

and the co-gradient locus is defined as:

$$\mathcal{H}_{ij} = \{\boldsymbol{x} \in \mathbb{R}^3;\ \boldsymbol{H}(\boldsymbol{x}) = \boldsymbol{0}\}.$$

We show in Figure 3.4 an example of the co-gradient locus associated with a given pair of ellipsoids in near-perfect contact. We observe, numerically at least, that in 3D, the co-gradient locus remains a hyperbola. This suggests that one of the components of $\boldsymbol{H}(\boldsymbol{x})$ is linearly dependent on the others. It also suggests that the polynomial $\det((1-t)\mathcal{Q}_i + t\mathcal{Q}_j) = 0$ must have two real solutions, one of them of multiplicity two. As in the 2D case, the centers $\boldsymbol{c}_i$ and $\boldsymbol{c}_j$ of the two ellipsoids necessarily belong to $\mathcal{H}_{ij}$ and the section $\mathcal{S}_{ij}$ of the co-gradient locus between $\boldsymbol{c}_i$ and $\boldsymbol{c}_j$ can be parametrized

in terms of the parameter $t \in [0,1]$ as:

$$\boldsymbol{x}(t) = ((1-t)\mathcal{Q}_i + t\mathcal{Q}_j)^{-1}((1-t)\mathcal{Q}_i\boldsymbol{c}_i + t\mathcal{Q}_j\boldsymbol{c}_j). \tag{3.24}$$

We note here that $(1-t)\mathcal{Q}_i + t\mathcal{Q}_j$ is a 3×3 matrix whose inverse can be analytically computed. The solutions $t_i \in [0,1]$ and $t_j \in [0,1]$ to the minimization problems satisfy, as in 2D, the root-finding problems (3.22) and (3.23), respectively. One advantage here is that one searches for the root of a scalar function whereas one needs to compute the root of a vector-valued function in S-GPA [32].

3.6 Numerical examples

We present in this section several numerical examples to analyze the performance of the proposed algorithm and compare its efficiency with S-GPA [32]. on large sets of pairs of ellipses and ellipsoids generated randomly by the algorithms described in [36]. These algorithms produce pairs of ellipses or ellipsoids in near perfect contact for which the exact position of the point $\boldsymbol{x}_i$ that satisfies (3.12) is available by construction. However, the exact position of the associated $\boldsymbol{x}_j$ is not known a priori for these pairs of particles.

In this example, we use the algorithm proposed in [36] to produce 10,000 pairs of ellipses. The algorithm takes as inputs the shape parameters $\mathbb{S}_i$ and $\mathbb{S}_j$ associated with ellipses $\mathcal{E}_i$ and $\mathcal{E}_j$, the relative positioning parameters $\mathbb{P}$, and the absolute positioning parameter $\boldsymbol{c}_j$. Here, for a given ellipse $\mathcal{E}$, the set $\mathbb{S}$ is defined as:

$$\mathbb{S} = \{\gamma, \omega, \theta\},$$

where $\gamma = a/b$ and $\omega = ab$ denote the *aspect ratio* and the *area ratio* (the latter is indeed the ratio between the area πab of the ellipse and the area π of the unit circle), and θ determines the orientation of the ellipse with respect to the x-axis. The set $\mathbb{P}$ for a pair of ellipses is defined as:

$$\mathbb{P} = \{\phi, \bar{\varepsilon}\},$$

where ϕ is an angle in $]-\pi, \pi]$ to define a point on the ellipse with respect to which one positions a disc in the corresponding ellipse-disc configuration (after transformation) and $\bar{\varepsilon}$ denotes the penetration/separation distance between the ellipse and the disc,

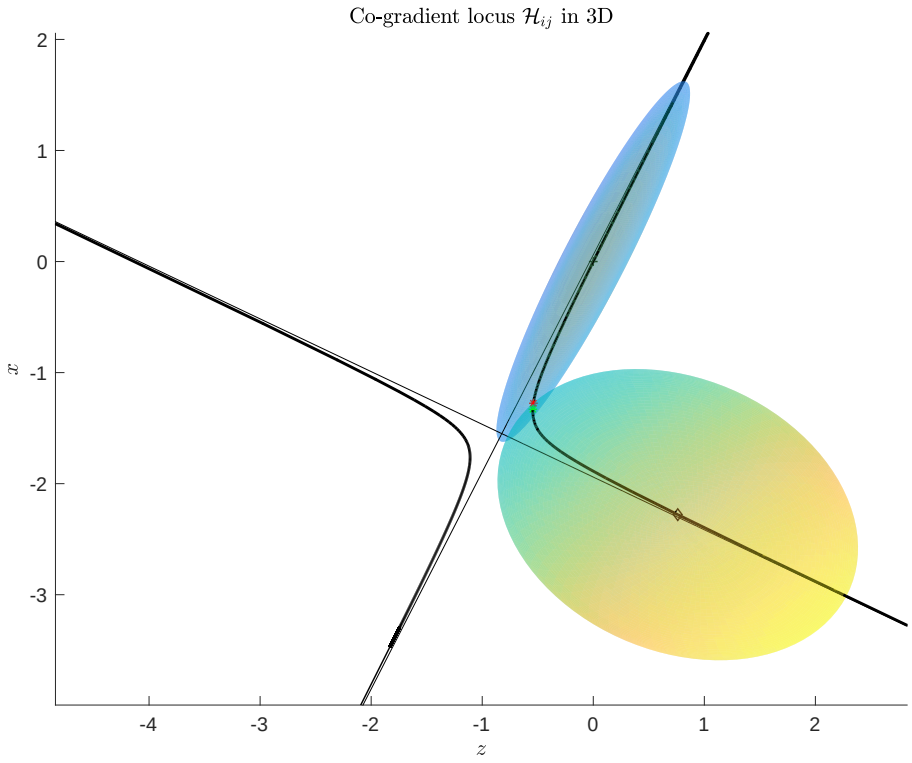


Figure 3.4 Example of the co-gradient locus associated with a given pair of ellipsoids in near-perfect contact.

see [36] for details. In summary, the input parameters are

$$X_{\text{in}} = \mathbb{S}_i \cup \mathbb{S}_j \cup \mathbb{P} \cup \{\boldsymbol{c}_j\} = \{\gamma_i, \omega_i, \theta_i, \gamma_j, \omega_j, \theta_j, \phi, \bar{\varepsilon}, \boldsymbol{c}_j\}, \tag{3.25}$$

while the output parameter for fully describing the pair of ellipses is the center $\boldsymbol{c}_i$ of ellipse $\mathcal{E}_i$. Once the pair of ellipses is provided, one can then compute $\boldsymbol{x}_i \in \mathcal{E}_i$ and $\boldsymbol{x}_j \in \mathcal{E}_j$ as the "closest" points to $\mathcal{E}_j$ and $\mathcal{E}_i$, respectively, i.e. the solutions to (3.12) and (3.13), and can be used to evaluate the distance $\varepsilon = \|\boldsymbol{x}_j - \boldsymbol{x}_i\|$.

We generate general pairs of ellipses using the following distributions of the shape parameters:

$$\begin{aligned} &\gamma_i \sim \mathcal{U}([1, 20]), \quad && \omega_i \sim \mathcal{U}([1, 20]), \quad && \theta_i \sim \mathcal{U}(]0, \pi]), \\ &\gamma_j \sim \mathcal{U}([1, 20]), \quad && \omega_j \sim 1, && \theta_j \sim \mathcal{U}(]0, \pi]), \end{aligned}$$

and of the positioning parameters:

$$\phi \sim \mathcal{U}(]-\pi,\pi]),$$
$$\boldsymbol{c}_j \sim \mathcal{U}(\{\boldsymbol{x} \in \mathbb{R}^2 \ ; \ \|\boldsymbol{x}\|^2 \leq 1\}).$$

Moreover, we consider pairs of overlapping ellipses and choose the penetration distance $\bar{\varepsilon}$ in the range $10^{-10} \leq -\bar{\varepsilon} \leq 10^{-3}$, that is, the distribution of $\bar{\varepsilon}$ in the ellipse-disc configuration is given by:

$$\log(|\bar{\varepsilon}|) \sim \mathcal{U}(-10,-3).$$

The results are reported in Table 3.1.

Table 3.1 Comparison of the contact points obtained with S-GPA and the new algorithm.

Mean distance	Numerical Value
$\mathbb{E}(-\log \|\boldsymbol{x}_i^{\text{exact}} - \boldsymbol{x}_i^{\text{SGPA}}\|)$	9.010311
$\mathbb{E}(-\log \|\boldsymbol{x}_i^{\text{exact}} - \boldsymbol{x}_i^{\text{new}}\|)$	14.103864
$\mathbb{E}(-\log \|\boldsymbol{x}_i^{\text{SGPA}} - \boldsymbol{x}_i^{\text{new}}\|)$	9.010311
$\mathbb{E}(-\log \|\boldsymbol{x}_j^{\text{SGPA}} - \boldsymbol{x}_j^{\text{new}}\|)$	6.647318

3.7 Conclusion

We have presented in this paper a new algorithm that falls within the class of geometric potential methods for contact detection between pairs of ellipses or ellipsoids in near perfect contact. The two minimization problems, whose solutions are combined to determine a contact point between the particles, are transformed into two root-finding problems using a parametric representation of the co-gradient locus.

The efficiency of the new algorithm is compared with that of S-GPA on large sets of pairs of ellipses and ellipsoids, for which we control the distribution of key parameters such as the ratio of areas, the aspect ratios of the ellipses, and the separation/penetration distance between the particles. These tests allow us to conclude that our new algorithm is faster than S-GPA.

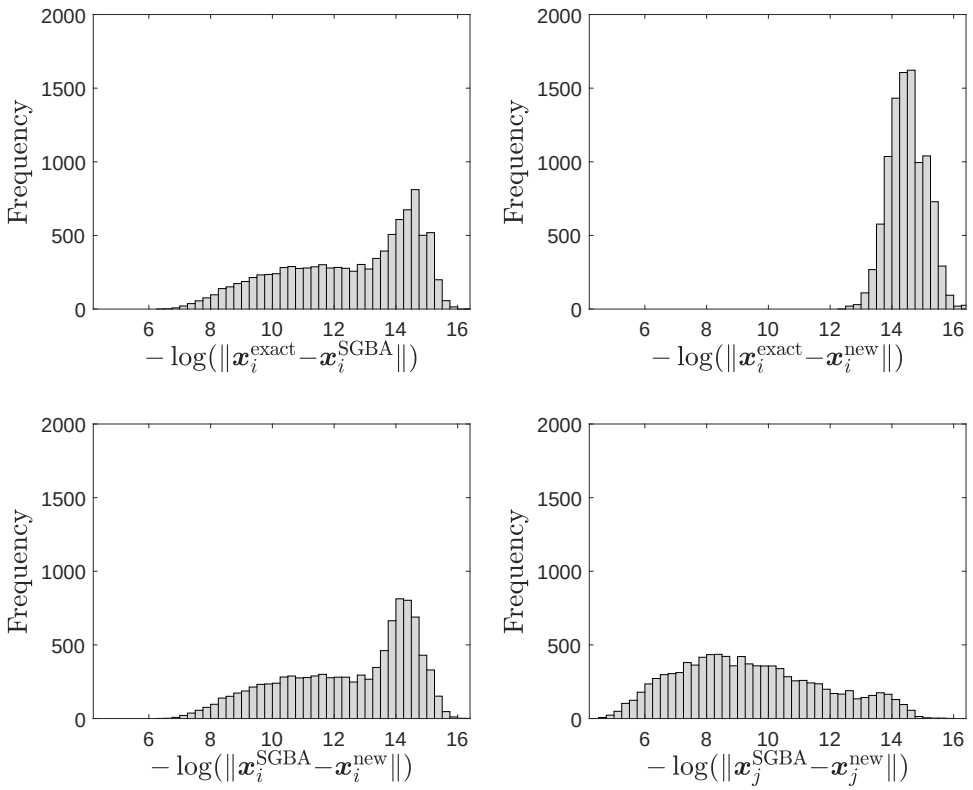


Figure 3.5 Distributions of the difference between differently obtained contact points from a sample of 10,000 pairs of ellipses.

CHAPTER 4 NEURAL NETWORK TO ESTIMATE DISTANCES BETWEEN TWO ELLIPSOIDS

4.1 Introduction

In this chapter, we use a deep learning approach to directly estimate the distance between two ellipsoids. This is done using a graph neural network that takes the ellipsoid parameters as input and produces the distance function as output.

Machine learning is a field of computer science that focuses on developing algorithms and models that can learn from data and make predictions or decisions without being explicitly programmed. There are several types of machine learning methods, including supervised learning, unsupervised learning, and deep learning.

Supervised learning is a type of machine learning where the model learns from labeled data. The goal is to build a model that can predict the output (or label) of new data points based on their input features. There are two main types of supervised learning: classification and regression. Classification is used when the output variable is categorical (i.e., has a finite number of possible values). Some popular classification algorithms include logistic regression, decision trees, and random forests. The book by Hastie, Tibshirani, and Friedman [37] provides a comprehensive overview of these methods. Regression, on the other hand, is used when the output variable is continuous (i.e., can take on any value within a range). Regression methods include linear regression, support vector regression, and neural networks. “Pattern Recognition and Machine Learnin” by Bishop [38] is a great resource for learning about these methods.

Unsupervised learning is a type of machine learning where the model learns from unlabeled data. The goal is to discover patterns or relationships in the data without any prior knowledge of what the output should be. Clustering and dimensionality reduction are two popular types of unsupervised learning. Clustering algorithms group data points based on their similarity to each other. K-means clustering, hierarchical clustering, and DBSCAN are some examples of clustering algorithms. “Machine Learning” [39] is a good resource for learning about these methods. Dimensionality reduction algorithms aim to reduce the number of input features while retaining as much of the original information as possible. Principal component analysis (PCA) and t-SNE are two commonly used dimensionality reduction techniques. The book by Tan, Steinbach, Karpatne, and Kumar “Introduction to Data Mining” [40] covers

these methods in detail.

Deep learning is a type of machine learning that uses artificial neural networks to learn from data. It has gained popularity in recent years due to its ability to achieve state-of-the-art performance on various tasks such as image classification, speech recognition, and natural language processing. “Neural Networks” by Rojas [41] provides a good introduction to the fundamentals of neural networks, while “Deep Learning” by Goodfellow, Bengio, and Courville [42] covers the latest advancements in deep learning.

Machine learning methods are becoming increasingly important in solving real-world problems. By leveraging the knowledge and techniques from these references, one can develop a solid understanding of machine learning and its applications. It has been shown that various types of these algorithms could be considered for this particular application, however neural networks were selected as the preffered approach for the following reasons:

- Non-linear relationships: Neural networks can learn non-linear relationships between inputs and outputs, other schemes such as k-NN and convolutional networks are not able to learn these non-linear relationships as effectively,
- Flexibility: Neural networks can be adapted to a wide range of tasks. This is because they have a large number of parameters that can be adjusted during the training process to optimize performance,
- Scalability: Neural networks can be trained on large datasets using distributed computing, which allows them to learn from vast amounts of data,
- Generalization: Neural networks can generalize well to new data, which means they can perform well on inputs they have not seen before. This is because they learn patterns in the data, rather than simply memorizing specific examples like k-NN does.

Another alternative approach is to use a hybrid method that combines analytical methods with machine learning. For example, a neural network could be trained to predict the parameters of the analytical method, such as the closest points on the ellipsoids, which can then be used to compute the distance. This approach could potentially combine the accuracy of the analytical method with the speed of the neural network. A neural network could also be taught to provide a binary output as

to whether a pair of ellipsoids are close or far. This last option would be the minimum requirement for the use of such a network in the Broad Phase Search of the DEM.

4.2 Neural Networks

Inspired by the workings of the human brain, the theoretical foundations of neural networks can be traced back to the works of John von Neumann and Rudolph Ortvay [43].

A neural network is, for all intents and purposes, a function made up of artificial neurons organized into layers (an input layer, hidden layers and an output layer). Neurons receive inputs, perform a calculation, and output a result. The layers of neurons are connected to each other by weights that are adjusted during training.

Let $\nu\colon \mathbb{R}^{n_0} \to \mathbb{R}$ be that function, taking n_0 inputs that describe two ellipsoids (the two ellipsoids' radii, orientation angles, centers and several combinations thereof) in the form of a vector $\mathbf{x}$ and outputting $\mathbf{y}$, which in this case is a single real number, the signed distance ε. Schematically, to compute the output $\mathbf{y}$, the inputs are propagated through the layers of the network using linear and nonlinear operations: the basic operations of neural networks are linear operations (weighted sums of the values of neurons in any given layer) expressed by matrix multiplication and bias addition whereas the nonlinear operations are called activation functions and introduce nonlinearity into the network. The weights of the network are adjusted during training.

The neural network ν is a composition of functions $f_0, f_1, \ldots, f_N, f_{N+1}$ such that

- $\nu = f_{N+1} \circ f_N \circ \ldots f_1 \circ f_0$
- N is the number of hidden layers,
- n_0 is the number of parameters in the input vector $\mathbf{x} = \mathbf{x}_0$,
- n_{N+2} is the number of parameters in the output vector $\mathbf{y} = \mathbf{x}_{N+2}$,
- each function $f_k\colon \mathbb{R}^{n_k} \to \mathbb{R}^{n_{k+1}}$ is such that $\mathbf{x}_{k+1} = f_k(\mathbf{x}_k) = a_k(\mathbf{W}_k\mathbf{x}_k + \mathbf{b}_k)$,
- a_k are activation functions such as the sigmoid function, tanh and ReLU,
- $\mathbf{W}_k \in \mathbb{R}^{n_{k+1}\times n_k}$ are weight matrices and $\mathbf{b}_k \in \mathbb{R}^{n_{k+1}}$ are bias vectors.

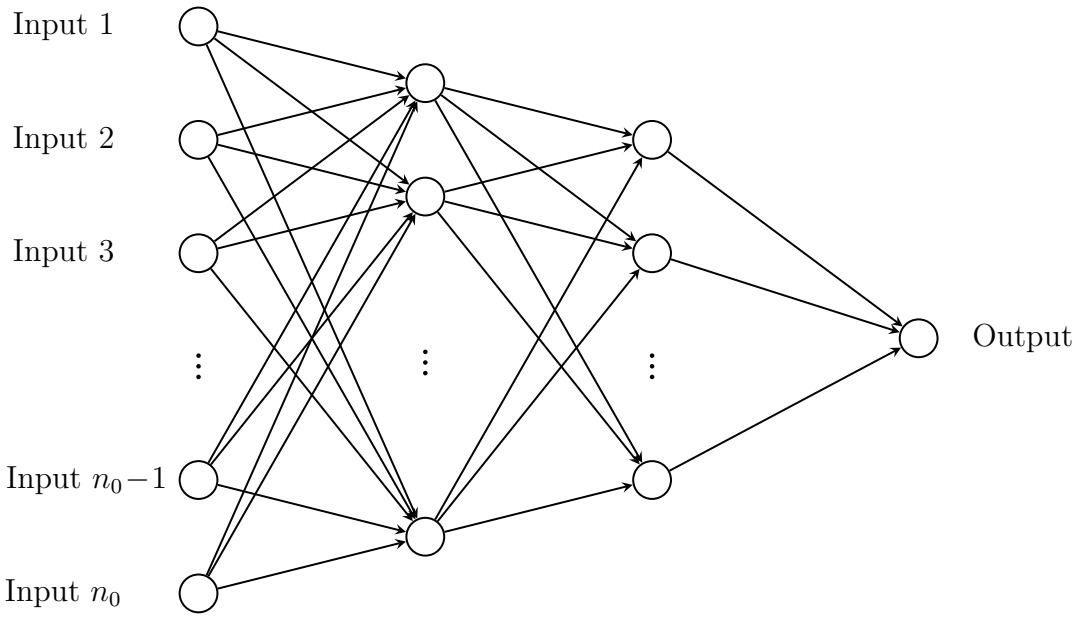


Figure 4.1 Illustration of a neural network architecture.

With given initial weights and biases, a first estimate is produced. A loss function is then introduced to measure the difference between the predicted output and the actual output. This difference is to be minimised through a process called *backpropagation*. The loss function is the key element here and has a big impact on the performance of the model. To achieve optimal performance from a neural network, it is essential to carefully consider the number of inputs n_0, the number of hidden layers N, and the choice of loss function as well as the size m of the training dataset.

4.2.1 Inputs and Outputs

In this particular context $n_{N+2} = 1$, the neural network ν outputs a real number $\mathbf{y}$ that predicts the value of the signed distance ε that separates the contact points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ on the pair of ellipsoids, $\varepsilon > 0$ should the ellipsoids be separated, $\varepsilon = 0$ if they are in perfect contact and $\varepsilon < 0$ otherwise. In order to do that, the neural network takes as input the vector $\mathbf{x} \in \mathbb{R}^{n_0}$, such that $\mathbf{y} = \nu(\mathbf{x})$. Determining which input vector $\mathbf{x}$ to choose is a difficult task. This should be considered a feasibility study. It is important to note that it is not always feasible or necessary to test all possibilities. These types of studies are usually conducted to assess the feasibility of a project, idea, or methodology, rather than to exhaustively test all possible outcomes. As such, the study may only explore a limited set of possibilities, as the main goal is to determine

if the project or idea is worth pursuing further. Therefore, further research may be necessary to explore the full range of options.

In 2D, a pair of ellipses is completely described by ten parameters (five each). So, $\mathbf{x} = (a_i, b_i, \theta_i, \boldsymbol{c}_i, a_j, b_j, \theta_j, \boldsymbol{c}_j)$ for instance should suffice to describe any given pair of ellipse and predict the overlapping or not distance between its contact points. However, seeing as there are known non linear combinations of shape parameters $\mathbb{S} = \{\gamma, \omega, \theta\}$, the neural network was provided with all the combinations that come up in the sampling algorithm of Chapter 2. In 2D, this amounts to $n_0 = 31$ because for each ellipse are given the lengths of the major and minor axis, the orientation angle, the sine and cosine of that angle, the Cartesian coordinates of the center, its polar coordinates (distance to the origin, polar angle, sine and cosine thereof), the components of the matrix $\mathcal{Q}$ and the aspect ratio γ. The area ratio ω is only given for the ellipse i since by construction $\omega_j = 1$ always. Even though information about the contact points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ is available, it is withheld from the inputs because it defeats the purpose of the neural network since $\varepsilon = \|\boldsymbol{x}_j - \boldsymbol{x}_i\|$ by definition and this should be fairly easy to obtain.

Optimally, still, it is expected that ten of those parameters can be identified through weight examination and optimization. Unfortunately, due to time constraints, this was not achieved and the opportunity remains to refine the list of parameters.

In 3D, similarly, the number of inputs is $n_0 = 79$: for each ellipsoid, three radii lengths, three orientation angles, the sines and cosines thereof, the Cartesian coordinates of the center, the spherical coordinates of that same center (distance to the origin, polar angle, azimuthal angle, sines and cosines thereof), the six distinct terms of the associated matrix $\mathcal{Q}$, all nine terms of the rotation matrix $\mathcal{R}$ and the aspect ratios γ_1 and γ_2. The volume ratio ω is only given for the ellipse i since by construction $\omega_j = 1$ always.

When modeling data, it is necessary to make choices about which parameters to include in the model. These choices are based on prior knowledge of the domain as well as exploratory data analysis. The choice of inclusion of the sine and cosine of all of the angles involved, as well as the matrix entries is motivated by the fact that modeling algorithms such as neural networks may have difficulty identifying these non-linear dependencies without an exhaustive search of all possible parameter combinations.

Optimally, in 3D, it should be possible to lower the number of input parameters to $n_0 = 18$, each ellipsoid being completely described by exactly nine parameters but,

as in the 2D case, this possibility remains untested.

4.2.2 Initializing Neural Networks

When training neural networks, the initialization step can be critical to the model's ultimate performance. Indeed, initializing the weights and biases with arbitrary values can lead to several issues. One obvious issue with completely random initialization is that it can lead to slow convergence during training, as the initial weights and biases may not be close to the optimal values. This can make it more difficult and time-consuming to train the network effectively. If the weights and biases are too large or too small, then, from a layer k, the terms of the vector $\mathbf{x}_{k+1} = \mathbf{W}_k\mathbf{x}_k + \mathbf{b}_k$ in the network can quickly become too large or too small as well as the layers go on. This can lead to the infamous problem of exploding or vanishing gradients (the gradients of the cost with the respect to the parameters are too big or too small) which may lead to divergence or a slow-down in the training and makes it difficult for the network to learn.

In order to prevent the gradients from vanishing or exploding, inspired by the Xavier initialization [44], it is assumed that the mean (or bias) of the neurons in any given layer should be zero and that their variance should stay the same across every layer. Under these two assumptions, when optimizing the loss function, the gradient signal should not be multiplied by values too small or too large in any layer.

$$\mathbb{E}[\mathbf{x}_{k+1}] = \mathbb{E}[\mathbf{x}_k]$$
$$\mathrm{Var}[\mathbf{x}_{k+1}] = \mathrm{Var}[\mathbf{x}_k]$$

To address these issues, Xavier Glorot and Yoshua Bengio [44] proposed a more sensible initialization method, known as the Xavier initialization. This method is based on the idea of keeping the variances of the activations and gradients roughly the same throughout the network. The weights and biases of a layer k are initialized taking

$$\mathbf{W}_k \sim \mathcal{N}(0, 1/n_k)$$
$$\mathbf{b}_k = \mathbf{0}$$

It can be easily verified, assuming that all inputs and all layers at initialization are independent and identically distributed, thus, it follows that it is also the case for the

weights and biases. We will assume that all inputs are normalized with zero means, and weights and biases are initialized following a distribution centered in zero, we have $\mathbb{E}[\mathbf{W}_k] = \mathcal{O}$ and $= \mathbb{E}[\mathbf{b}_k] = \mathbf{0}$.

By reasoning on a neuron $x_k^{(i)}$ in the layer k, using $a_k = \tanh$, we get

$$\mathbf{x}_{k+1} = a_k(\mathbf{W}_k\mathbf{x}_k + \mathbf{b}_k) \implies x_{k+1}^{(i)} = \tanh\left(\sum_{j=1}^{n_k} w_{k,ij}x_k^{(j)} + b_k^{(i)}\right)$$

Because around zero $x \simeq \tanh(x)$, $b_k^{(i)}$ is constant and weights and inputs are mutually independent, independent and identically distributed, the variance becomes

$$\begin{aligned}
\mathrm{Var}[x_{k+1}^{(i)}] &\simeq \mathrm{Var}\left[\sum_{j=1}^{n_k} w_{k,ij}x_k^{(j)} + b_k^{(i)}\right] = \sum_{j=1}^{n_k} \mathrm{Var}[w_{k,ij}x_k^{(j)}] \\
&\simeq \sum_{j=1}^{n_k} \Big(\underbrace{\mathbb{E}[w_{k,ij}]^2}_{0} \mathrm{Var}[x_k^{(j)}] + \mathrm{Var}[w_{k,ij}] \underbrace{\mathbb{E}[x_k^{(j)}]^2}_{0} + \mathrm{Var}[w_{k,ij}]\mathrm{Var}[x_k^{(j)}] \Big) \\
&\simeq \sum_{j=1}^{n_k} \mathrm{Var}[w_{k,ij}]\mathrm{Var}[x_k^{(j)}] = n_k \mathrm{Var}[w_{k,ij}]\mathrm{Var}[x_k^{(j)}]
\end{aligned}$$

Since it was assumed $\mathrm{Var}[x_k^{(1)}] = \mathrm{Var}[x_k^{(2)}] = \cdots = \mathrm{Var}[x_k^{(j)}] = \cdots = \mathrm{Var}[x_k]$ in a given layer k, and that for every term $\mathrm{Var}[w_{k,ij}] = \mathrm{Var}[w]$ the variance is constant, if $\mathrm{Var}[x_{k+1}] = \mathrm{Var}[x_k]$, it follows that $\mathrm{Var}[w] \simeq 1/n_k$, which completes the verification.

The Xavier initialization works well with tanh activation functions. For ReLU, a common initialization is He initialization [45] which, for a layer k, is given by

$$\begin{aligned}
\mathbf{W}_k &\sim \mathcal{N}(0, 2/n_k) \\
\mathbf{b}_k &= \mathbf{0}
\end{aligned}$$

Verification for this follows the same process as Xavier but proves to be slightly more technical.

4.3 Loss Functions

A loss function $J\colon \mathbb{R}^{m\times n_{N+2}} \times \mathbb{R}^{m\times n_{N+2}} \to \mathbb{R}$ measures the difference between the predicted outputs $\hat{\mathbf{y}} = \{\mathbf{y}_k\}$ and the actual output $\boldsymbol{\varepsilon} = \{\varepsilon_k\}$ from a training dataset with m datapoints. The goal of learning is to minimize this loss function to get an

accurate output.

$$J(\hat{\mathbf{y}}, \boldsymbol{\varepsilon}) \coloneqq \frac{1}{m} \sum_{k=1}^{m} L(\mathbf{y}_k, \varepsilon_k) \ ,$$

where $L \colon \mathbb{R}^{n_{N+2}} \times \mathbb{R}^{n_{N+2}} \to \mathbb{R}$ is a "local" loss scheme.

The loss function is used to calculate the gradient of the error with respect to the weights of the network. The gradient is then used to adjust the weights during learning. The choice of loss function depends on the type of problem being solved and the characteristics of the data. In this case, the output is an algebraic distance ε with values ranging from small overlaps, say $\mathbf{y}_{\min} = -10^{-2}$, to large gaps, say $\mathbf{y}_{\max} = +10^{1}$. With output values that can be meaningfully small (oders of magnitude smaller than the unit), while others are large in comparison (oders of magnitude larger than the unit), floating arithmetic issues may occur in the evaluation of the loss function when the predicted outcome is orders of magnitude appart from the target, yet still be close to zero. This informs the choice of loss function, by being mindful of the fact that more weight must be attributed to smaller targeted outcomes. This is why, in addition to the popular mean squared error (MSE) between the predicted distance and the ground truth distance

$$L_{\text{MSE}}(\mathbf{y}, \varepsilon) = \frac{1}{2}(\mathbf{y} - \varepsilon)^2 \ ,$$

other loss functions have been considered such as the mean squared logarithmic error (MSLE) between the predicted distance and the ground truth distance,

$$L_{\text{MSLE}}(\mathbf{y}, \varepsilon) = \frac{1}{2}(\log(\mathbf{y} + 1) - \log(\varepsilon + 1))^2 \ ,$$

and a proposed custom one given by

$$L_k(\mathbf{y}, \varepsilon) = (2k + 1)|\mathbf{y} - \varepsilon|^{1/2k+1} \quad \text{with } k \text{ an odd integer.}$$

It has been confirmed through trial and error that L_{MSE} performs worst. As expected, it predicts large distances fairly well, but commits large errors for distances that are close to zero. L_{MSLE} does better in that regard, however it incurs a larger penalty for the underestimation of the actual distance than the overestimation, meaning it leaves us with the opposite problem, that is very good predictions for very small distances and wrong ones for larger distances. Numerically, L_k proved to be better suited to address both problems. Figure 4.2 shows the benefits of this form of loss functions: continuous at zero, yet fairly steep gradient and non-stagnant for larger values. A

sensitivity analysis, testing how variations in input values affect the output of the model or system, is required to best choose the appropriate loss function. Such an analysis has not been done in this study.

4.4 Numerical Results

Using an ellipsoid pair test dataset extracted from the same data set as the training set, the performance of the neural network model is assessed. The model performs similarly to the analytical method in terms of distance estimate, with an average relative error 60%, which is rather high in comparison to the performance of the analytical method. There could be several reasons for this high error rate. One possibility is that the training dataset used to train the neural network was not general enough. The dataset may not have included enough samples of certain types of exotic ellipsoid pairs, leading to poor generalization performance.

In one trial, the neural network was trained with a single hidden layers. This resulted in a faster training time but with a lower accuracy. The average error in distance estimation was around 1 unit, which is not satisfactory for many applications.

One of the main difficulties encountered was the generation of an appropriate dataset of ellipsoid pairs with their corresponding distances ; the dataset has to cover a wide range of ellipsoid shapes and orientations to ensure that the neural network could generalize to new ellipsoid pairs. One solution is to sample separately pairs of ellipsoids (the smaller one having unit volume) that are far apart (with $\varepsilon > 10^{-1}$) and ellipsoids that are close and/or in contact (with $|\varepsilon| < 10^{-1}$). Another potential difficulty that was not investigated is overfitting of the neural network to the training dataset. This might occur considering the neural network has a rather complex architecture.

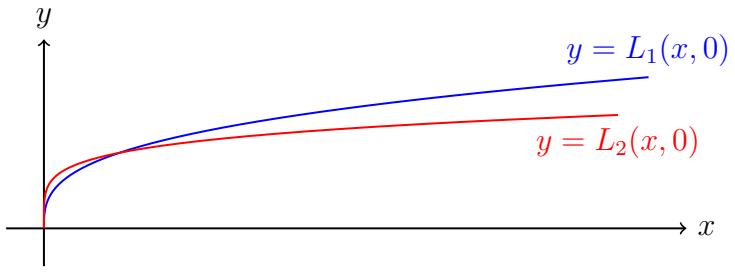
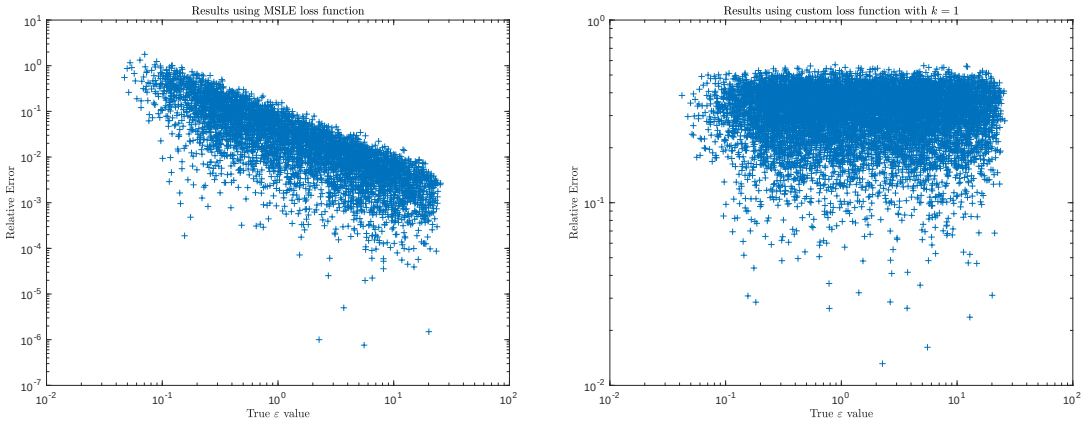


Figure 4.2 Graph of the L_k loss functions for $k = 1$ and $k = 2$.



(a) (b)

Figure 4.3 Comparison of relative errors obtained with L_{MSLE} and L_1 loss functions

To address such difficulties, several potential solutions can be considered. The neural network can be regularized using techniques such as dropout or weight decay. Additionally, the dataset can be augmented by applying random transformations to the ellipsoid pairs, such as rotations and translations, to increase the variability of the dataset. Another possibility is that the neural network's architecture was not appropriate for the given problem. The number of hidden layers, number of neurons per layer, and choice of activation function may not have been optimal for the given dataset, resulting in poor performance. It has been observed that the neural network's accuracy is worse for ellipsoid pairs with distances close to zero, there are a few potential ways to improve the results. One way is to modify the loss function used during training. The standard mean squared error (MSE) loss function may not be appropriate for cases where the true distances are close to zero since the error can become skewed towards larger values. Instead, a loss function that gives more weight to smaller distances, such as the mean absolute error (MAE) or the Huber loss, can be used to improve the accuracy for smaller distances. Another way to improve the results is to augment the training dataset with more samples of ellipsoid pairs with distances close to zero. By providing more examples of such cases, the neural network can learn to better handle them and produce more accurate results.

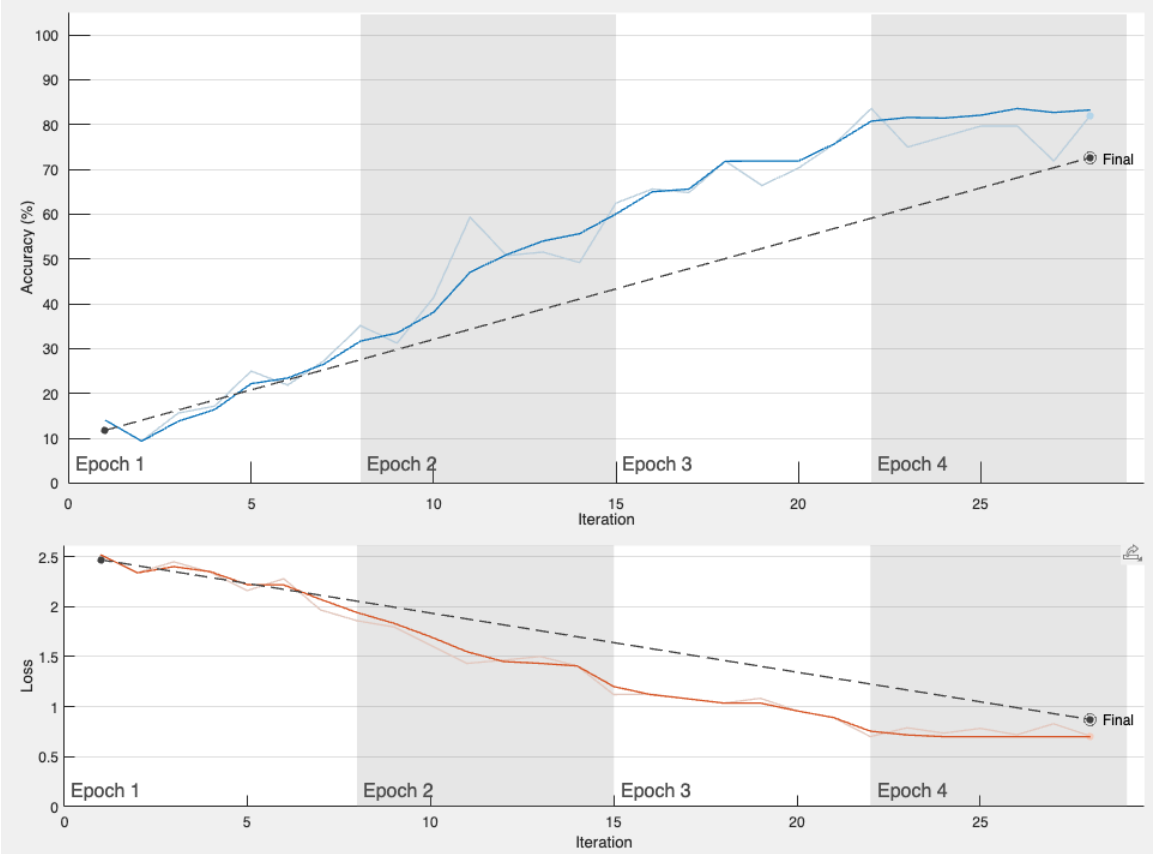


Figure 4.4 Training accuracy and loss function (smoothing using a moving average is shown by semi-transparent lines)

CHAPTER 5 CONCLUSION ET RECOMMENDATIONS

5.1 Summary of Work

This research involved three components, each of which were novel contributions. The initial objective was to develop a neural network for the approximation of the distance between a pair of ellipsoids. Although there exist a large number of highly accurate means of performing this task, they were all too expensive to be useful during the Broad Phase Search, where accuracy can take a backseat to efficiency.

Chapter 4 was dedicated to this aspect of the research. We showed that a neural network could be developed that would approximate the distance with a uniform relative error of at most 60% over the entire range of parameters of pairs of ellipses, but often with much lower error. The key was to use a loss function that would be uniform over the range of parameters, and although a log mean squared error was a natural candidate, we found that a cube root worked best. We note that only one other paper on this topic possessed much less convincing results.

In Chapter 2, we proposed a novel algorithm to generate pairs of ellipses or ellipsoids with known parameters. This was essential to producing a neural network that would be accurate over the desired space of parameters. This work was an improvement over an algorithm previously described by Kheradmand, Prudhomme and Laforest [17]. This particular work, although necessary for the first objective, may be of great interest to other researchers. This algorithm was initially used to compare the accuracy and efficiency of various CDAs over a large sample of the space of pairs of ellipses, and the current version improves on this by removing an orientation bias.

Chapter 3 described a new CDA that constrained the contact points to belong to the co-gradient locus rather than separately on each ellipse. In the process, this removed an important source of indeterminacy from past algorithms, which usually could not identify the contact point from among four roots. The new algorithm was also simple to implement, did not require costly coordinate transformations, and appeared to be more reliable. Unfortunately, we were unable to measure the computational efficiency of the new CDA. At this moment, we are still unable to make a complete comparison between this algorithm and the state of the art.

5.2 Limitations

Due to time constraints, the research in Chapter 4 was exploratory and no attempt was made to identify an optimal network. In particular, the size of the network was not studied and it might have been possible to use less than the list of parameters we selected. Although the accuracy was acceptable, we were unable to explain the range of accuracy observed and in particular the generalization error was not characterized. We also selected only one specific subset of parameter space to train on, but it would be important to test the limits of learning of the neural networks, particularly for pairs of very well-separated ellipses (easier) or ellipses of areas that vary by one or two orders of magnitude (harder).

The algorithm to generate pairs of ellipses was investigated in great detail and forms a novel and self-contained contribution to the scientific literature. Conceptually, one could argue that the algorithm is limited to ellipses/ellipsoids but the approach taken was never intended to be generalized.

In Chapter 3, a new CDA was developed. It was formulated as a nonlinear problem and in our implementation, MATLAB's general purpose `fzero` function was used to solve the resulting problem. Although in some circumstances, this could be viewed as an advantage, further research is required to identify the limitations of this choice. Furthermore, an independent solver would be required if we were to perform additional tests to measure its computational efficiency.

5.3 Future Research

Of all the contributions outlined in this report, the neural network is the one for which most work is required. It would be possible to further explore the space of parameters and hyperparameters. Other alternative methods could also be explored if one only needed binary responses, such as SVM or tree-based methods. With much of ground work established, there is still much that could be done.

The algorithm for generating pairs of ellipses was tested exhaustively over the course of this research and was found to be very reliable. It may not be possible to improve the algorithm, although it may be possible to use it in new situations, such as during the initial deposition phase of DEM, when particles have to be placed inside a domain. There are many algorithm to do this, but the algorithm proposed could be used to place the particles directly in contact at a chosen point and orientation, thereby

avoiding collisions with previously deposited particles.

Finally, the new CDA should be tested against a wide-range of well-known CDAs. So far, only the original algorithm [17] has been used as a benchmark. Since this is the first CDA of its kind, we also hope that it will inspire others to explore this new class of CDAs.

REFERENCES

[1] F. Radjaï, J.-N. Roux, and A. Daouadji, "Modeling granular materials: Century-long research across scales," *Journal of Engineering Mechanics*, vol. 143, p. Article Number: UNSP 04017002, Jan-23-2017 2017.

[2] C. Dafermos, *Hyperbolic Conservation Laws in Continuum Pysics*, ser. Grundlehren der mathematischen Wissenschaften. New York: Springer-Verlag, 2000, vol. 325.

[3] A. K. Ashmawy, V. V. Hoang, and B. Sukumaran, "Evaluating the influence of particle shape on liquefaction behavior using discrete element modeling," all Days of International Ocean and Polar Engineering Conference, ISOPE-I-03-161, 2003.

[4] W. Wang and M. R. Coop, "An investigation of breakage behaviour of single sand particles using a high-speed microscope camera," *Géotechnique*, vol. 66, no. 12, pp. 984–998, 2016.

[5] P. A. Cundall and O. D. L. Strack, "A discrete numerical model for granular assemblies," *Géotechnique*, vol. 29, no. 1, pp. 47–65, 1979. [Online]. Available: https://doi.org/10.1680/geot.1979.29.1.47

[6] H. Nguyen, *GPU Gems 3*, 1st ed. Addison-Wesley Professional, 2007.

[7] D. N. Ilin and M. Bernacki, "A new algorithm for dense ellipse packing and polygonal structures generation in context of FEM or DEM," *MATEC Web Conf.*, vol. 80, p. 02004, 2016.

[8] G. Nolan and P. Kavanagh, "Random packing of nonspherical particles," *Powder technology*, vol. 84, no. 3, pp. 199–205, 1995.

[9] C.-Y. Wang, C.-F. Wang, and J. Sheng, "A packing generation scheme for the granular assemblies with 3D ellipsoidal particles," *International Journal for Numerical and Analytical Methods in Geomechanics*, vol. 23, no. 8, pp. 815–828, 1999.

[10] S. Johnson, J. R. Williams, and B. Cook, "Contact resolution algorithm for an ellipsoid approximation for discrete element modeling," *Engineering Computations*, vol. 21, no. 2/3/4, pp. 215–234, 2004.

[11] C. Hogue, "Shape representation and contact detection for discrete element simulations of arbitrary geometries," *Engineering Computations*, vol. 15, no. 3, pp. 374–390, 1998.

[12] E. G. Gilbert, D. W. Johnson, and S. S. Keerthi, "A fast procedure for computing the distance between complex objects in three-dimensional space," *IEEE Journal on Robotics and Automation*, vol. 4, no. 2, pp. 193–203, April 1988.

[13] K.-W. Lim, K. Krabbenhoft, and J. E. Andrade, "Numerical simulations of granular soil using elliptical particles," *Computational Particle Mechanics*, vol. 1, no. 3, pp. 257–275, 2014.

[14] E. Kheradmand, S. Prudhomme, and M. Laforest, "A mathematical framework for the analysis and comparison of contact detection methods for ellipses and ellipsoids," *Computational Particle Mechanics*, vol. 9, no. 9, pp. 1153–1203, 2022.

[15] L. Rothenburg and R. J. Bathurst, "Numerical simulation of idealized granular assemblies with plane elliptical particles," *Computers and Geotechnics*, vol. 11, no. 4, pp. 315–329, 1991.

[16] C. Wellmann, C. Lillie, and P. Wriggers, "A contact detection algorithm for superellipsoids based on the common-normal concept," *Engineering Computations*, vol. 25, no. 5, pp. 432–442, 2008.

[17] E. Kheradmand, S. Prudhomme, and M. Laforest, "A fast contact detection method for ellipsoidal particles," *International Journal for Numerical and Analytical Methods in Geomechanics*, vol. 45, no. 9, pp. 1169–1194, 2021.

[18] A. Džiugys and B. Peters, "A new approach to detect the contact of two-dimensional elliptical particles," *International Journal for Numerical and Analytical Methods in Geomechanics*, vol. 25, no. 15, pp. 1487–1500, 2001.

[19] S. Hwang *et al.*, "A machine learning-based particle-particle collision model for non-spherical particles with arbitrary shape," *Chemical Engineering Science*, vol. 251, p. 117439, 2022.

[20] R. P. Jensen *et al.*, "Effect of particle shape on interface behavior of DEM-simulated granular materials," *International Journal of Geomechanics*, vol. 1, no. 1, pp. 1–19, 2001.

[21] J. Wang *et al.*, "Particle shape effects in discrete element modelling of cohesive angular particles," *Granular Matter*, vol. 13, no. 1, pp. 1–12, 2011.

[22] J. M. Ting *et al.*, "An ellipse-based discrete element model for granular materials," *International Journal for Numerical and Analytical Methods in Geomechanics*, vol. 17, no. 9, pp. 603–623, 1993.

[23] B. Yan, R. A. Regueiro, and S. Sture, "Three-dimensional ellipsoidal discrete element modeling of granular materials and its coupling with finite element facets," *Engineering Computations*, vol. 27, no. 4, pp. 519–550, 2010.

[24] T.-T. Ng and W. Zhou, "DEM simulations of bi-disperse ellipsoids of different particle sizes," *Comptes Rendus Mécanique*, vol. 342, no. 3, pp. 141–150, 2014.

[25] J. Q. Gan, A. B. Yu, and Z. Y. Zhou, "DEM simulation on the packing of fine ellipsoids," *Chemical Engineering Science*, vol. 156, pp. 64–76, 2016.

[26] E. Kheradmand, M. Laforest, and S. Prudhomme, "A mathematical framework for the analysis and comparison of contact detection methods for ellipses and ellipsoids," *Computational Particle Mechanics*, vol. 9, no. 6, pp. 1153–1203, 2022.

[27] L. Rothenburg and R. J. Bathurst, "Micromechanical features of granular assemblies with planar elliptical particles," *Géotechnique*, vol. 42, no. 1, pp. 79–95, Mar. 1992. [Online]. Available: http://www.icevirtuallibrary.com/doi/10.1680/geot.1992.42.1.79

[28] H. Ouadfel and L. Rothenburg, "An algorithm for detecting inter-ellipsoid contacts," *Computers and Geotechnics*, vol. 24, no. 4, pp. 245–263, 1999.

[29] J. M. Ting, "A robust algorithm for ellipse-based discrete element modelling of granular materials," *Computers and Geotechnics*, vol. 13, no. 3, pp. 175–186, 1992.

[30] X. Lin and T.-T. Ng, "Contact detection algorithms for three-dimensional ellipsoids in discrete element modelling," *International Journal for Numerical and Analytical Methods in Geomechanics*, vol. 19, no. 9, pp. 653–659, 1995.

[31] E. Kheradmand, "Contact detection for pairs of ellipses and ellipsoids: Analysis, comparisons, and improvements," Ph.D. dissertation, Polytechnique Montréal, Canada, 2020.

[32] E. Kheradmand, S. Prudhomme, and M. Laforest, "A fast contact detection method for ellipsoidal particles," *International Journal for Numerical and Analytical Methods in Geomechanics*, vol. 45, no. 9, pp. 1169–1194, 2021.

[33] M. G. Choi, "Computing the closest approach distance of two ellipsoids," *Symmetry*, vol. 12, p. 1302, 2020.

[34] R. P. Brent, *Algorithms for Minimization without Derivatives*. Prentice-Hall, 1973.

[35] T. R. Chandrupatla, "A new hybrid quadratic/bisection algorithm for finding the zero of a nonlinear function without using derivatives," *Advances in Engineering Software*, vol. 28, no. 3, pp. 145–149, 1997.

[36] G. Banna, M. Laforest, and S. Prudhomme, "Generation of pairs of ellipses and ellipsoids from distributions in parameter space," Submitted to Computer Aided Geometric Design, 2023.

[37] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer, 2009. [Online]. Available: https://web.stanford.edu/~hastie/ElemStatLearn/

[38] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York: Springer, 2006.

[39] T. M. Mitchell, *Machine Learning*. New York: McGraw Hill, 1997.

[40] P.-N. Tan *et al.*, *Introduction to Data Mining*. Boston: Pearson, 2018.

[41] R. Rojas, *Neural Networks*. Berlin: Springer, 1996.

[42] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA: MIT Press, 2016. [Online]. Available: https://www.deeplearningbook.org/

[43] S. Levy, *Artificial Life: A Report from the Frontier Where Computers Meet Biology*. New York: Vintage Books, 1992.

[44] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," *Journal of Machine Learning Research*, January 2010.

[45] K. He *et al.*, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *2015 IEEE International Conference on Computer Vision (ICCV)*, February 2015, pp. 1026–1034.