Check for updates

NIST Advanced Manufacturing Series
NIST AMS 100-70

Theory and Computation of Median Straight Lines and Planes in Coordinate Metrology

Craig Shakarji
Kexin Yin
Qunfen Qi
Edward Morse
Vijay Srinivasan

This publication is available free of charge from:
https://doi.org/10.6028/NIST.AMS.100-70

NIST NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY
U.S. DEPARTMENT OF COMMERCE

NIST Advanced Manufacturing Series
NIST AMS 100-70

Theory and Computation of Median Straight Lines and Planes in Coordinate Metrology

Craig Shakarji
Sensor Science Division, Physical Measurement Laboratory

Vijay Srinivasan
Engineering Laboratory

Kexin Yin
University of Huddersfield, UK

Qunfen Qi
University of Briston, UK

Edward Morse
University of North Carolina, Charlotte, USA

This publication is available free of charge from:
https://doi.org/10.6028/NIST.AMS.100-70

November 2025

U.S. Department of Commerce
Howard Lutnick, Secretary

National Institute of Standards and Technology
Craig Burkhardt, Acting Under Secretary of Commerce for Standards and Technology and Acting NIST Director

NIST AMS 100-70
November 2025

Certain equipment, instruments, software, or materials, commercial or non-commercial, are identified in this paper in order to specify the experimental procedure adequately. Such identification does not imply recommendation or endorsement of any product or service by NIST, nor does it imply that the materials or equipment identified are necessarily the best available for the purpose.

NIST Technical Series Policies
Copyright, Use, and Licensing Statements
NIST Technical Series Publication Identifier Syntax

Publication History
Approved by the NIST Editorial Review Board on 2025-05-02

How to Cite this NIST Technical Series Publication
Shakarji C.M., Srinivasan V., Yin K., Qi Q., Morse E. (2025) Theory and Computation of Median Straight Lines and Planes in Coordinate Metrology. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Advanced Manufacturing Series (AMS) NIST AMS 100-70. https://doi.org/10.6028/NIST.AMS.100-70

Author ORCID iDs
Craig Shakarji: 0009-0000-9877-7034
Vijay Srinivasan: 0000-0002-4653-8821
Kexin Yin: 0009-0006-2323-272X
Qunfen Qi: 0000-0001-5936-1714
Edward Morse: 0000-0003-1227-700X

Contact Information
craig.shakarji@nist.gov

Abstract

This paper presents the theoretical foundation and computational techniques for establishing median straight lines and planes for point-clouds in coordinate metrology. The theoretical foundation is based on an optimization formulation that exploits the invariant and equivariant properties of the point-clouds that reside in a Euclidean metric space. The optimization formulation leads directly to versatile computational approaches. The paper shows that the computational challenges can be easily overcome with readily available, free, and high-quality software packages. Such median fits are advantageous in many areas including when one-sided anomalies (which arise in some advanced manufacturing contexts and in some advanced measurement systems) should be ignored. Additional application of median straight lines and planes involves a standardized approach to handling outliers in coordinate metrology. Further applications involve standardized specification of tolerance zones using quartile and percentile zones, thus introducing rank-order statistics in the field of geometrical tolerancing and related metrological practices.

Keywords

computations; coordinate metrology; lines; median statistics; minimum absolute deviation; planes; standards.

Acknowledgement

The authors would like to thank their colleagues in the ASME and ISO standards development organizations and industrial consortia. The work presented in this paper is an official contribution of the National Institute of Standards and Technology (NIST) and not subject to copyright in the United States.

Table of Contents

Introduction

Least-squares fitting has long held enduring appeal in the field of computational coordinate metrology [1-3]. The theoretical and computational foundations for determining *mean* straight lines and planes for point-clouds through least-squares fitting are well-established and widely utilized in various industries. However, the theory and computation of *median* straight lines and planes, particularly in the context of coordinate metrology, are less explored. This is probably due to the perception that median-based methods are computationally more complex, and that there are fewer supporting tools and resources available.

This paper takes the initial steps to bridge this gap by unifying the theoretical concepts of mean and median geometric elements under the umbrella of optimization. It exploits the invariant and equivariant properties of the point-clouds that reside in a Euclidean metric space. Furthermore, it demonstrates that the computational challenges associated with median geometric elements can be readily addressed with the prevalence of free, high-quality software packages.

The concept of median is not new. Median has found common usage in everyday phrases such as ‘median income’ and ‘median house price.’ This is entirely due to the robustness of the median, unaffected by the presence of a few outliers in the data. This notion is carried further in reporting experimental data in scientific and technical literature in the form of box plots [4, 5]. A box plot (also known as a box-and-whisker plot) is a simple, but effective, visual tool to show how a set of measured values are distributed into quartiles, with the median being the central value. The box plot can be used to observe potential outliers in the data and can be refined further into percentiles as a way of visualizing and exploring rank-order statistics. While these concepts are well-established for univariate (one-dimensional) problems in statistical literature, coordinate metrology poses a multivariate (multidimensional) challenge for which the theoretical and computational issues have not yet been fully explored.

This paper extends the use of median and rank-order statistics in coordinate metrology by generalizing the notion of median using optimization. For a two-dimensional (2D) point-cloud in a plane or a three-dimensional (3D) point-cloud in space, it establishes the theoretical foundation for defining a median straight line or a median plane, respectively, employing a multivariate optimization formulation. The paper then provides illustrative examples demonstrating how to find solutions to the optimization problems using direct solvers readily available in open-source software packages.

A median-based fit may be advantageous when the surface irregularities (or measured points) are one-sided. Such cases could be spurious points from reflection when using an optical measuring system or similar artifacts from an X-ray computed tomography system. Certain advanced manufacturing applications can result in a protrusion that creates a one-sided anomaly. In such a case, a feedback mechanism might be used to improve the process. However, in a least-squares fit, the anomaly itself contributes to the fit, resulting in an error map (used for corrections) that does not represent the true nature of the problem. A median based fit would ignore the anomaly, leaving the error map to reveal the isolated problem appropriately. There are also one-sided cases of etchings or fissures on a planar surface that should be ignored, for which a median based fit is ideal.

The results of this paper are also applicable to identifying potential outliers in coordinate metrology and defining quartile and percentile tolerance zones for geometrical specifications. The implications of these applications are significant for the development of future standards in geometrical tolerancing and associated metrological practices. In addition, the success of this approach motivates a powerful variational formulation for future research in computational coordinate metrology, paralleling the well-established success of variational formulations in classical and modern mechanics.

The rest of this paper is organized as follows. Section 2 introduces an optimization formulation for defining and computing mean and median in univariate problems. This concept is generalized to higher dimensions in Sections 3, which exploits the invariant and equivariant properties of the point-clouds that reside in a Euclidean metric space. Computational techniques for median straight lines using direct search methods are discussed in Section 4, which is followed by computation of median planes in Section 5. The paper is summarized, and future research directions are outlined in Section 6.

1. Mean and Median Using Univariate Optimization

The mean (arithmetic average) and median for a one-dimensional (also known as univariate) dataset can be easily expressed through a straightforward closed-form formula and the bisection of a rank-ordered dataset, respectively [6]. For a collection of n real numbers $X = \{x_1, x_2, \dots, x_n\}, x_i \in \mathbb{R}$, elementary statistics textbooks define the mean in closed-form as $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$. The computational time complexity of calculating the mean in this case is evidently $O(n)$.

A comparable definition for median, even in elementary textbooks, demands a bit more effort. Firstly, it is necessary to sort the the input dataset to produce a rank-ordered set, resulting in an associated computational time complexity of $O(n \log n)$. Assuming, without loss of generality, that $X = \{x_1, x_2, \dots, x_n\}$ has been sorted and rank-ordered from the smallest value x_1 to the largest value x_n, the median $\tilde{x}$ is then defined as follows:

- If n is odd, then $\tilde{x} = x_{\lceil \frac{n}{2} \rceil}$. Here, $\left\lceil \frac{n}{2} \right\rceil$ is the ceiling of $\frac{n}{2}$.
- If n is even, then $\tilde{x} \in [x_{\frac{n}{2}}, x_{\frac{n}{2}+1}]$, which is a closed interval. It is customary to pick $\tilde{x}$ as the mid-point of this interval. It can also occur at a repeated number (if $x_{\frac{n}{2}} = x_{\frac{n}{2}+1}$) in the sorted input collection X.

Since half of the data is on one side of $\tilde{x}$ and the other half of the data on the other side of $\tilde{x}$, this type of definition of median can be viewed as the 'bisection definition.' The computational time complexity is clearly dominated by the sorting step, and hence the overall time complexity of computing the median is $O(n \log n)$.

There is an interesting and important connection between such well-established mean and median definitions and the field of optimization. To illustrate this connection, consider the following optimization problem:

> Given a collection of n real numbers $X = \{x_1, x_2, \dots, x_n\}, x_i \in \mathbb{R}$, and an objective function $f_p(x) = \sum_{i=1}^{n} |x - x_i|^p$, solve the unconstrained univariate optimization problem: $\min_{x \in \mathbb{R}} f_p(x)$.

The objective function employs the l_p-norm, with p typically taking values of 1, or 2, or ∞. It can be shown rigorously that for $p = 1$, the minimum is attained when x corresponds to the *median* of the given n numbers. Similarly, for $p = 2$, the minimum is reached when x represents the *mean* of the given n numbers. Thus, in the optimization problem posed above, employing the l_1-norm leads to the median, while the l_2-norm leads to the mean. It is interesting to note that no sorting of the input data is involved for computing the median using this optimization formulation.

To illustrate the optimization formulation further, consider the following two examples (from [7]):

Example A: From the measurements made in 1846 of the vertical semidiameter of the planet Venus, a dataset consisting of fifteen residuals (about a simple model) in arc seconds is given below:

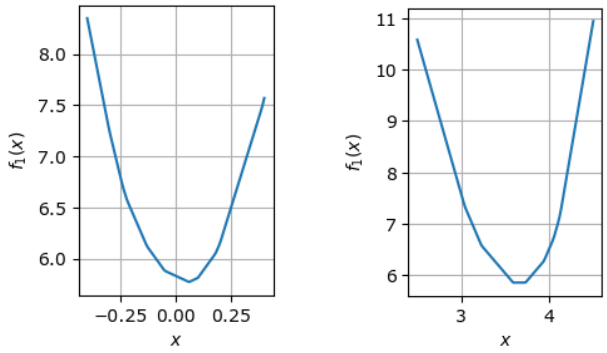
-0.30	-0.44	1.01	0.48	-0.24
0.06	0.63	-0.13	-1.40	-0.22
-0.05	0.20	0.18	0.39	0.10

Example B: Tensile testing of samples made from a plastic material has resulted in the following ten measurements of the percentage elongation at breakage:

3.73	3.59	3.94	4.13	3.04
2.22	3.23	4.05	4.11	2.02

Fig. 1 shows the objective function $f_1(x)$ utilizing the l_1-norm for Examples A and B. A similar plot of the objective function $f_2(x)$ using the l_2-norm would have shown a smooth (C^2-continuous) parabolic curve for each of the examples. For clarity, only a localized section of the graph of $f_1(x)$ near the minimum is presented in Fig.1 for each of the examples. In both examples, the objective function is convex, C^0-continuous, and piece-wise linear. However, a crucial distinction arises:

- When n is odd, as seen in Example A with $n = 15$, the objective function features a distinct minimum. In Fig. 1(a) the minimum occurs at $x = 0.06$, coinciding with the dataset's median.
- When n is even, as seen in Example B with $n = 10$, the objective function displays a flat bottom. As depicted in Fig. 1(b), any x value in the interval $[3.59, 3.73]$ can serve as the median. Although the global minimum lacks uniqueness, it is customary to select the midpoint of this interval, which, in this case, is 3.66, and designate it as the median.



(a) (b)

FIGURE 1. OBJECTIVE FUNCTION $f_1(x)$:
(a) FOR EXAMPLE A; (b) FOR EXAMPLE B.

Obtaining a numerical solution for the optimization problem to compute the median may necessitate the use of direct search methods [8], which do not rely on derivatives. This requirement is due to the non-smooth nature of the objective function. These methods have

been implemented in readily available software libraries [9], with a time complexity of $O(nh)$, where h represents the number of objective function evaluations required for convergence. A noteworthy comparison involves the time complexity of $O(n \log n)$ when using the bisection definition for the median computation, as mentioned earlier.

It is crucial to recognize that when the dataset comprises an even number of data points, a direct search might yield only one solution from the flat bottom region of the objective function. While this may not present a challenge within the realm of rank-order statistics, it is imperative to be mindful of this aspect when drawing conclusions from numerical solutions in optimization problems involving the l_1-norm.

Since determining the mean and median in univariate cases is a straightforward process through closed-form formulas and bisection definitions that involve a simple sorting, one may question the necessity of delving into the optimization formulation and its associated computational cost. However, it will soon become apparent that the optimization formulation plays a pivotal role in extending the concepts of mean and median to higher dimensions, as elaborated in the following Section 3.

2. Higher-Dimensional Generalization Using Multivariate Optimization

Consider the point-clouds shown in Figs. 2 and 3 residing in a plane and space, respectively. The 2D point-cloud of Fig. 2 is derived from an example in a well-known textbook [4] and it contains 30 points. The 3D point-cloud in Fig. 3(b) depicts measured data using an optical probe on an additively manufactured planar surface pictured in Fig. 3(a); it contains 900 points measured on a 30×30 grid.

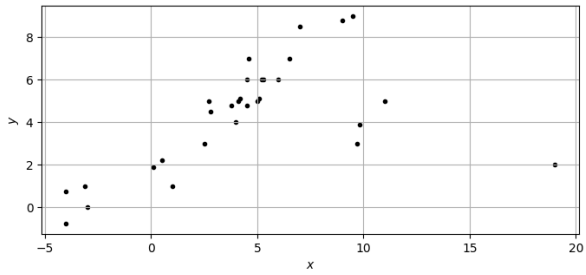
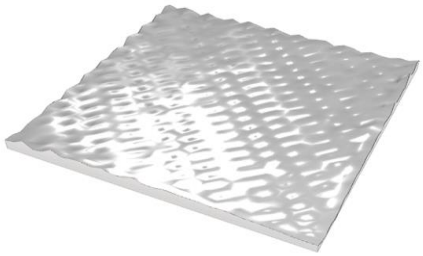
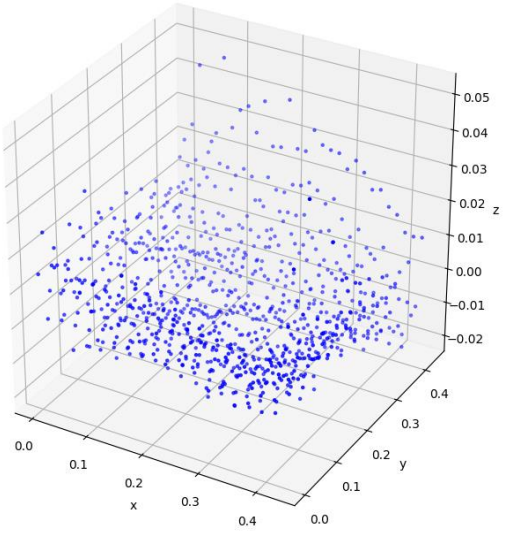


FIGURE 2. EXAMPLE OF A 2D POINT-CLOUD IN A PLANE.



(a)

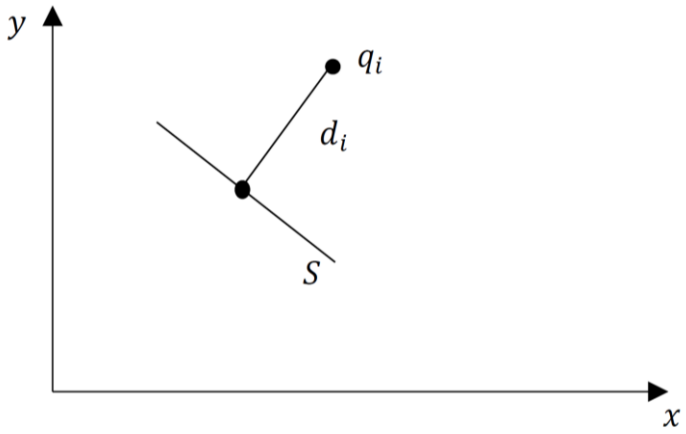


(b)

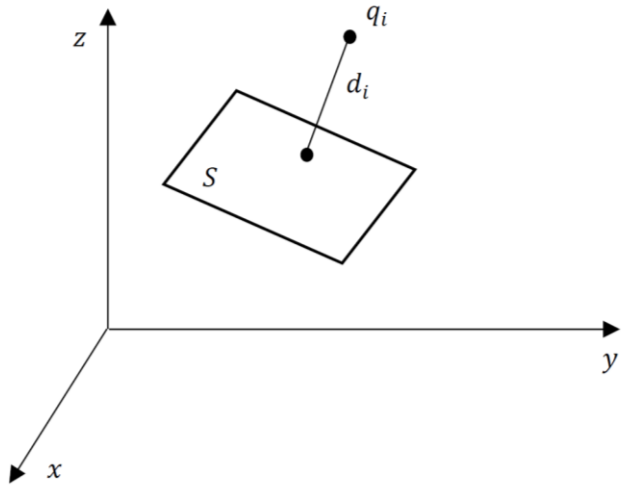
FIGURE 3. EXAMPLE OF A 3D POINT-CLOUD IN SPACE.

Least-squares fitting of a straight line for points in Fig.2 and a plane for points in Fig.3 have been well researched [1-3] and used extensively in industry. These are the generalizations of the mean from univariate cases discussed in Section 2 to multivariate cases – in other words, generalization from one-dimensional cases to multidimensional cases. In the context of coordinate metrology, it is the *total* least-squares fitting (where the deviation of a point from a straight line or plane is measured perpendicularly) that dominates the industrial practice. This topic will be reviewed briefly in Section 3.2.

Finding a median straight line for the point-cloud in Fig.2 and a median plane in Fig 3 will require the generalization of the concept of median to multidimensions. A quick look at these figures reveals the fact that the bisection definition will not yield a good generalization – there is an uncountably infinite number of straight lines that will bisect the point-cloud in Fig. 2, and the same holds true for bisection planes in Fig. 3.



(a)



(b)

FIGURE 4. ILLUSTRATION OF DEVIATION d_i.

A satisfactory generalization of the median as well as the mean can be found by posing a multivariate optimization problem, which parallels the univariate optimization problem presented in Section 2, as follows:

> Given a parameterized family S of straight lines or planes, a collection of n deviations $\{d_1, d_2, \dots, d_n\}, d_i \in \mathbb{R}$, and an objective function $f_p(S) = \sum_{i=1}^{n} |d_i|^p$, solve the unconstrained multivariate optimization problem: $\min_S f_p(S)$.

Here the deviation d_i is the distance measured perpendicularly from a point q_i in the point-cloud to S (which can be a straight line or a plane) as illustrated in Fig. 4. Hence the objective function is defined using an orthogonal l_p-norm; when $p = 1$ it leads to median straight line or plane and setting $p = 2$ yields mean straight line or plane. Using the orthogonal norms is justifiable because of the invariance and equivariance properties of the point-clouds in Euclidean space, as described in the following Section 3.1.

2.1. Invariance and Equivariance in Euclidean Metric Space

The Euclidean distance between any two points q_i and q_j is expressed in terms of the x, y, and z coordinates of the points as $\sqrt{(q_{ix} - q_{jx})^2 + (q_{iy} - q_{jy})^2 + (q_{iz} - q_{jz})^2}$. It is well known that the Euclidean distance between any two distinct points remains (1) invariant under translations and rotations, and (2) equivariant under uniform scaling. Here the term *invariant* means that there is no change in the entity to which the term is applied. For example, a rotational motion will change the cartesian coordinates of all points in a point-cloud, but the distance between any two points in the point-cloud will remain the same. On the other hand, the term *equivariant* means that the entity to which it is applied will undergo the same change as the other entities under transformation. For example, a unform scaling will change the cartesian coordinates of all points in a point-cloud, and the distance between any two points in the point-cloud will also change by the same amount.

A major advantage of using the orthogonal l_p-norm in the optimization formulation is that the optimal straight lines and planes will remain equivariant under translation, rotation, and uniform scaling of the point-clouds. For example, if a rotational motion M is applied to a point-cloud, then the resulting optimal straight line or plane (by a new computation) will be a rotated version of the original optimal straight line or plane by the same rotational motion M. Moreover, the rank-ordering of points in a point-cloud based on their orthogonal deviations from the optimal straight line or plane will remain invariant under translation, rotation, and uniform scaling of the point-cloud – this leads to the important result of transforming a multivariate problem to a univariate problem for rank-order statistics.

A rigorous proof of these assertions relies on the invariant and equivariant properties of the Euclidean distance between points stated earlier. A physical justification for requiring the equivariance of straight lines and planes with point-clouds in coordinate metrology is self-evident: *it is a cardinal principle that the results computed from coordinate measurements should be reproducible under different choices of the coordinate reference frame and with different units of measurements*.

The equivariance properties satisfied under the orthogonal l_p-norm are also important for computations. In any numerical analysis, it is always a good practice to normalize the input data so that no data point has very large numerical value. Therefore, it is advisable to transform the coordinates of an input point-cloud by the following steps in sequence:

1. Translation so that the centroid is at the origin.
2. Uniform scaling so that the absolute value of any coordinate does not exceed unity.
3. Rotation so that the principal axes are aligned with the axes of the reference frame.

Figs. 5 and 6 show the results after applying these steps to the input point-clouds of Figs. 2 and 3, respectively.

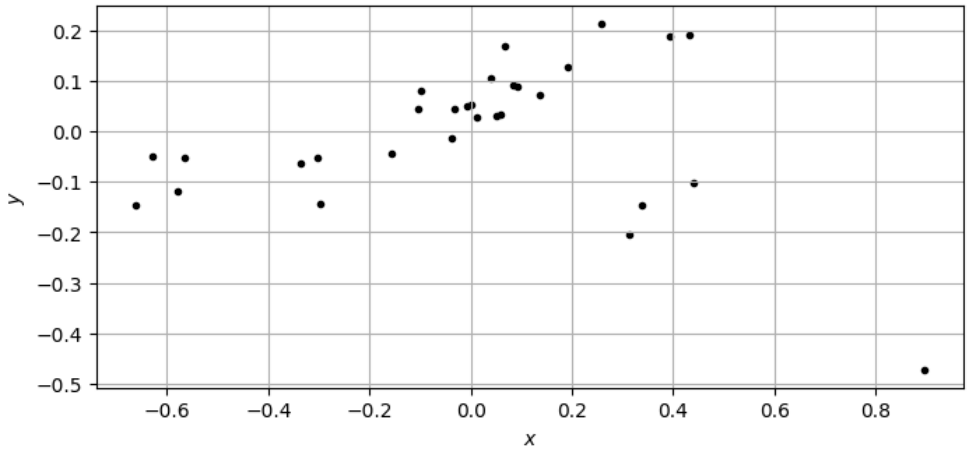


FIGURE 5. TRANSFORMED 2D POINT-CLOUD IN A PLANE.

These transformations are completely reversible. By applying them in the reverse order, the transformed point-cloud and the geometrical elements (e.g., optimal straight lines and planes) computed with it can be brought back to the original data space. All computations discussed in the rest of the paper will be based on first normalizing the input dataset using this sequence of transformations. It is noteworthy that these transformations already provide a powerful solution to the optimization problem when the objective function is defined using the orthogonal l_2-norm, as described in the next Section 3.2.

2.2. Optimization Under Orthogonal l_2-norm

When an orthogonal l_2-norm is used in defining the objective function, the resulting optimal solution is called a total least-squares solution [10]. An elegant algorithm to find the total least-squares fitting of a straight line to a 2D point-cloud in a plane (such as in Fig. 2) can be given as follows:

1. Translate the points so that their centroid is at the origin: $(x_i', y_i') = (x_i - \bar{x}, y_i - \bar{y})$, where $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$, $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$.

2. Compute the singular value decomposition (SVD) of the $n \times 2$ matrix $M = \begin{bmatrix} {x_1}' & {y_1}' \\ \dots & \dots \\ {x_n}' & {y_n}' \end{bmatrix}$. A simpler approach will compute the eigenvalues and eigenvectors of the 2×2 matrix $M^T M$; this approach is considered (slightly) less desirable due to numerical stability issues.
3. The solution is the straight line that (1) passes through the origin, and (2) perpendicular to the singular vector that corresponds to the smallest singular value in the SVD. That singular vector can be shown to be the same as the eigenvector that corresponds to the smallest eigenvalue of $M^T M$.

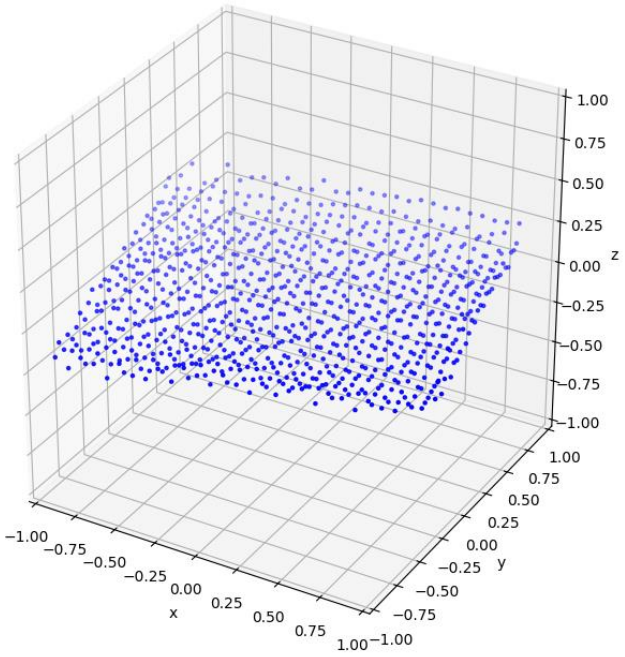


FIGURE 6. TRANSFORMED 3D POINT-CLOUD IN SPACE.

This algorithm can be easily extended to 3D point-clouds by replacing all $n \times 2$ matrices and their operations to $n \times 3$ matrices and their corresponding operations. This means that, under the l_2-norm, the x-axis is the mean straight line for the 2D point-cloud in Fig. 5 and the xy-plane is the mean plane for the 3D point-cloud in Fig. 6. One way to reason about the computational time complexity of these algorithms is to consider using the eigenvalues and eigenvectors, where the computation is dominated by the formation of the matrix $M^T M$. Then the overall algorithms are clearly of $O(n)$ time complexity, because solving the 2×2 and 3×3 matrix eigenproblems take only constant time.

Elegant algorithms, like the one presented here, are currently unavailable for computing median straight lines and planes. However, this gap will be addressed in the following Sections 4 and 5, leveraging powerful direct search methods for optimization that are readily accessible as free software packages.

3. Computing Median Straight Line for a 2d Point-Cloud

A straight line in a plane belongs to a 2-parameter family of curves [11]. Denoting the two parameters as c_1 and c_2, the optimization problem to determine a median straight line for a point-cloud $\{q_1, \dots, q_n\}$, such as the one shown in Fig. 5, can be posed as

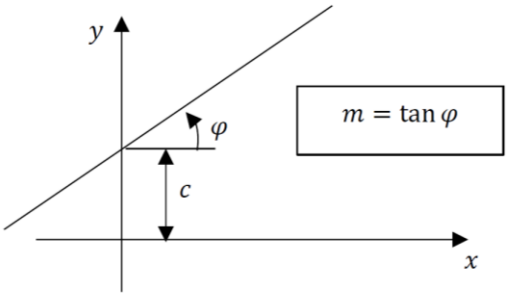
$$\min_{c_1,c_2} \sum_{i=1}^{n} |d_i(c_1, c_2)| \qquad (1)$$

where $d_i(c_1, c_2)$ is the perpendicular distance from the point q_i to the straight line being sought, as shown in Fig. 4(a). This could be a signed distance, depending on which side of the straight line the point q_i lies, and hence the need for taking its modulus in defining the objective function $\sum_{i=1}^{n} |d_i(c_1, c_2)|$. It is clearly a bivariate function of the two parameters c_1 and c_2. There are at least two different ways to represent a straight line, leading to two different ways to parameterize it. The choice of the representation and the associated parameterization will affect the geometric shape of the objective function and the efficacy of a search method to find the minimum. These are discussed in the next Section 4.1.

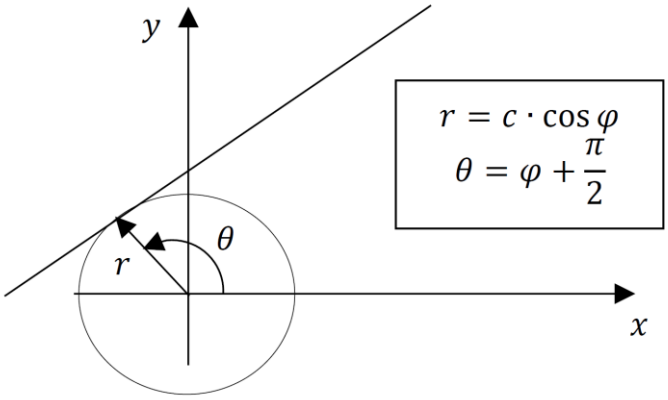
3.1. Explicit and Implicit Representations of a Straight Line in a Plane

An explicit representation of a straight line in a plane is given by the well-known equation $y = mx + c$, with slope m and y-intercept c serving as the two parameters as illustrated in Fig. 7(a). The largest feasible solution space for the straight line under this representation is bounded by $m \in (-\infty, +\infty)$ and $c \in (-\infty, +\infty)$. This representation is not suitable when the straight line is vertical, and it can run into numerical problems if the straight line is near-vertical because the slope and the y-intercept will be very large in absolute values.

This problem can be avoided by choosing an implicit representation of the form $x \cos\theta + y \sin\theta - r = 0$, where a radius r and an angle θ serve as the two parameters. This type of implicit representation is known as the Hesse Normal Form [12]. Fig. 7(b) shows a geometrical interpretation of the two parameters. Here, r is the perpendicular distance from the origin to the straight line; it can be zero if the straight line passes through the origin. This permits a radius vector for a circle to be drawn as shown in Fig. 7(b), with the straight line being tangential to the circle. If the straight line passes though the origin, then the radius vector can be the unit vector normal to the straight line. The angle θ is the angle between the x-axis and the radius vector. The largest feasible solution space for the straight line under this representation is bounded by $0 \leq r < \infty$ and $-\pi \leq \theta < \pi$. There are occasions in which r might take negative values; these will be discussed in Section 4.2. In all real datasets, r can be bounded from above using a bounding box around the input point-cloud.



(a) Explicit representation: $y = mx + c$



(b) Implicit representation: $x\cos\theta + y\sin\theta - r = 0$

FIGURE 7. TWO REPRESENTATIONS FOR A STRAIGHT LINE.

These two representations can be converted from one to the other for the same straight line using the formulas relating the parameters given in Fig. 7. It is important to note that all angles are measured in radians. When an explicit representation is used, the distance of Eq. (1) can be expressed as

$$d_i(m,c) = (y_i - mx_i - c)/\sqrt{m^2+1} \qquad (2)$$

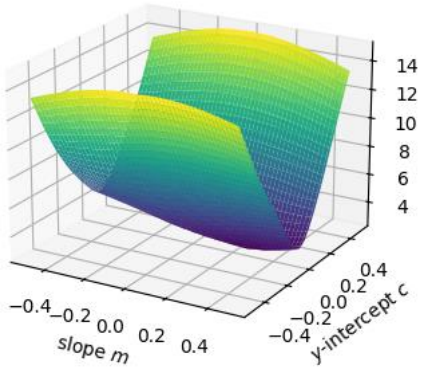
where (x_i, y_i) are the Cartesian coordinates of the point q_i in the point-cloud. This distance is positive if q_i lies above the straight line, zero if it lies on the straight line, and negative if it lies below the straight line. If, on the other hand, an implicit representation is used, the distance formula becomes

$$d_i(r,\theta) = x_i\cos\theta + y_i\sin\theta - r. \qquad (3)$$

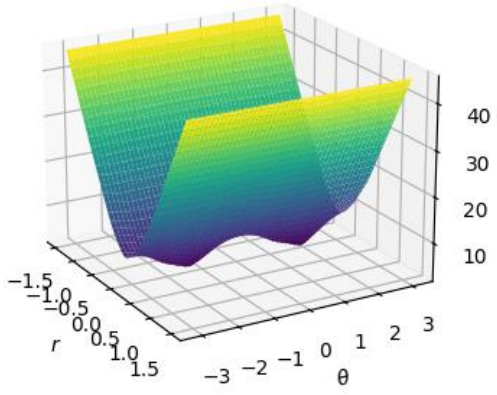
This distance is negative if q_i lies on the same side of the straight line as the origin, zero if it lies on the straight line, and positive otherwise.

Armed with the formulas for d_i in Eqs. (2) and (3), it is possible to visualize the objective function $\sum_{i=1}^{n}|d_i|$ for the point-cloud in Fig. 5. In Fig. 8, the 3D surface plots of the objective functions are shown using both the explicit and implicit representations. These plots are useful to observe the overall shape of the objective functions under the two representations. A more detailed picture emerges in the contour plots of the same objection function in Fig. 9.

Valuable insights about the objective functions under the orthogonal l_1-norm can be drawn from Figs. 8 and 9. First, the objective functions are continuous but not smooth, just like the C^0-continuous functions in Fig. 1 for the univariate case. As a result, optimization schemes reliant on gradients are not applicable since the gradient may not exist in certain locations. Nevertheless, the objective functions display a discernible downhill feature, a desirable characteristic for derivative-free downhill search methods to find a local minimum.

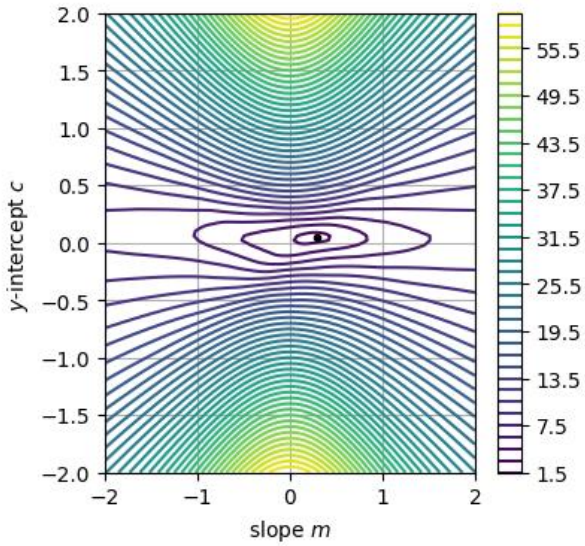


(a) Explicit representation

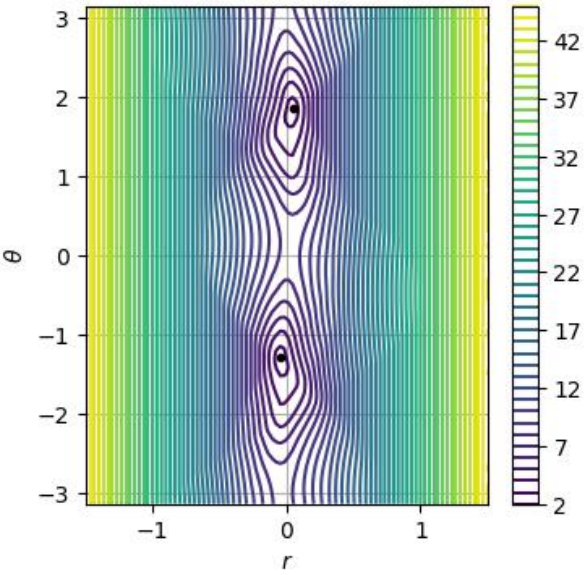


(b) Implicit representation

FIGURE 8. SURFACE PLOTS OF OBJECTIVE FUNCTIONS.



(a) Explicit representation



(b) Implicit representation

FIGURE 9. CONTOUR PLOTS OF OBJECTIVE FUNCTIONS.

Another important insight provided by these figures is about finding a global minimum. Figs. 8(a) and 9(a) indicate that there is only one minimum if the explicit representation is used, leading one to believe that finding a local minimum will also yield the global minimum. Figs. 8(b) and 9(b) suggest the presence of two minima and a saddle point. A closer inspection reveals that the apparent two minima are, in fact, the same. This equivalence arises from the algebraic relation $re^{i\theta} = -re^{i(\theta-\pi)}$, even though (r, θ) and $(-r, \theta - \pi)$ may appear as distinct coordinates in geometric plots using Cartesian coordinates. Additionally, both parameter pairs (r, θ) and $(-r, \theta - \pi)$ satisfy the same implicit equation $x \cos\theta + y \sin\theta - r = 0$, signifying that they correspond to the same straight line. Furthermore, verification from

Eq. (3) confirms that $|d_i(r,\theta)| = |d_i(-r,\theta-\pi)|$, establishing that the objective functions are identical at both locations on the plots.

These insights hold the potential for transferability to other datasets obtained from measuring various straight lines in a plane. The plots in Figs. 8 and 9 are for the point-cloud in Fig. 5, a normalized version of the point-cloud in Fig. 2. When presented with a different point-cloud, assumed to result from measuring a straight line, and subsequently normalized as outlined in Section 3.1, the corresponding objective function is likely to display a similar topography, as illustrated in Figs. 8 and 9. Therefore, the lessons derived from this example offer valuable guidance in selecting appropriate search methods, as discussed in the subsequent Section 4.2, to address the minimization problem posed by Eqs. (1), (2), and (3).

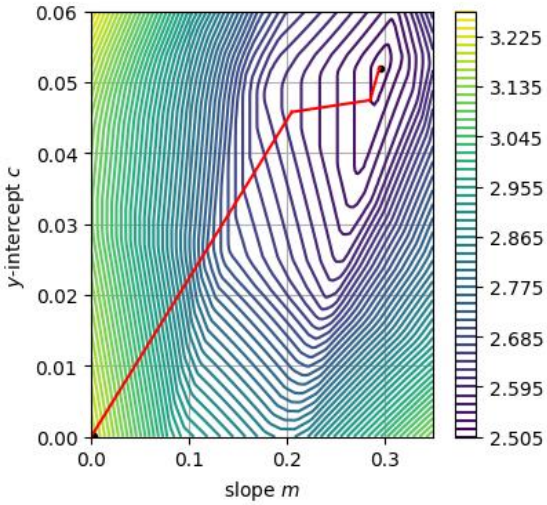
3.2. Computational Details

Finding the minimum when the objective function looks like those shown in Figs. 8 and 9 can be accomplished using commonly available high-quality software packages such as Scipy [9]. Of nearly a dozen minimization methods available in the optimization portfolio of Scipy, at least two are suitable for the type of terrain exhibited by the objective functions to find the median straight line. Both are direct search methods that do not require derivatives; they depend solely on objective function evaluation.

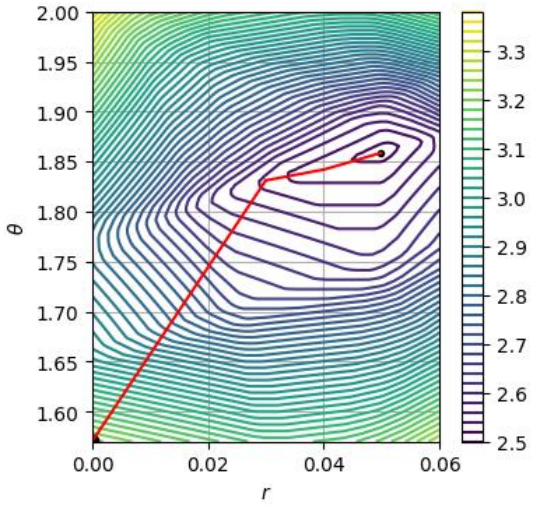
One of them is the Powell method, which belongs to the class of conjugate gradient methods [8]. The other is the Nelder-Mead method, which belongs to the class of polytope methods [8]. Both start with a guess for the solution, which is provided by the user. This task is simplified by the normalization process outlined in Section 3.1, which rotates the point-cloud to a near horizontal position. If an explicit representation is used, then $m = 0$ and $c = 0$ provide a very good start. If an implicit representation is used, then $r = 0$ and $\theta = {}^{\pi}/_{2}$ serve as a good starting guess.

Fig. 10 illustrates how the Powell method descends to the bottom of the valley from the starting point. Fig. 11 provides a similar visualization for the Nelder-Mead method. Table 1 shows the optimization results, the number of iterations, and the total number of objective function evaluations (Fun_evals) for both methods under implicit as well as explicit representations. The Powell method takes a series of one-dimensional search steps and in each step, it evaluates the objective function several times. The Nelder-Mead method employs several downhill sliding simplices and each new simplex involves new objective function evaluations.

The results shown in Table 1 are the ones reported in the Scipy implementations of the Powell and Nelder-Mead methods, using the default convergence criteria for termination of the searches. In all the four cases, the optimization termination was declared successful by Scipy. If there are h function evaluations, then the computational time complexity of both methods can be expected to be $O(nh)$. When the computations were performed in a Jupyter Lab platform [13] on a commonly available laptop computer, the Scipy performance was almost instantaneous in producing the reported results.



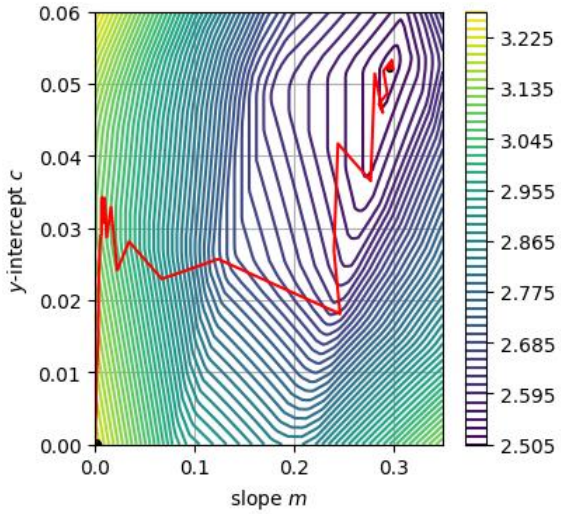
(a) Explicit representation



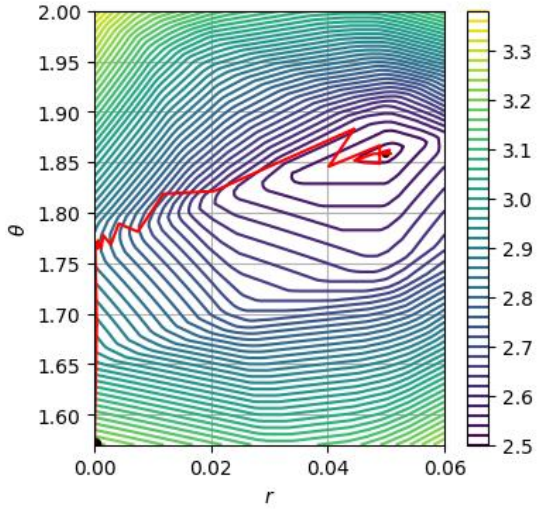
(b) Implicit representation

FIGURE 10. POWELL METHOD IN ACTION.

Figure 12 shows the result for the median straight line as the one computed using the orthogonal l_1-norm. As Table 1 indicates, there is no appreciable difference among the four results and all of them produce the same plot as seen in Fig. 12. For comparison, the computed straight line using the orthogonal l_2-norm is shown in Fig. 12 as the mean straight line. As remarked earlier, it is not surprising that the x-axis is the mean straight line for the normalized point-cloud. The mean and median straight lines of Fig. 12 can be easily transformed back to the original data space by applying the inverse transformations, as outlined in Section 3.1. It is worth noticing in Fig. 12 that the median straight line bisects the point-cloud, which is an inherent consequence of using the l_1-norm.



(a) Explicit representation



(b) Implicit representation

FIGURE 11. NELDER-MEAD METHOD IN ACTION.

TABLE 1. COMPARISON OF OPTIMIZATION METHODS FOR MEDIAN STRAIGHT LINES.

	Powell method	Nelder-Mead method
Explicit representati	f_{min} = 2.511144 m = 0.2961 c = 0.052 Iterations = 6 Fun_ evals = 252	f_{min} = 2.511159 m = 0.2961 c = 0.0521 Iterations = 62 Fun_evals = 119
Implicit representati	f_{min} = 2.511144 r = 0.0499 θ = 1.8586 Iterations = 4 Fun_evals = 176	f_{min} = 2.511162 r = 0.0499 θ = 1.8587 Iterations = 45 Fun_evals = 88

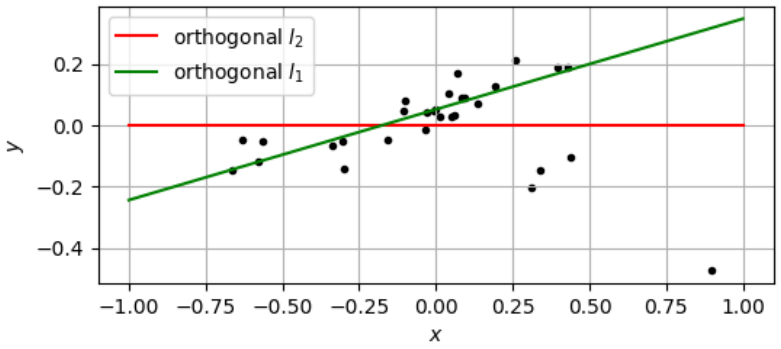


FIGURE 12. MEAN (MOSTLY HORIZONTAL) AND MEDIAN (INCLINED) STRAIGHT LINES.

3.3. Theil-Sen Regression

An alternative approach to computing a median straight line is the use of Theil-Sen regression [4], which can be summarized as follows: For n points (x_i, y_i) for i = 1, …, n, in a plane:

1. Find the median (m) of the slopes of straight lines through every pair of points.
2. Find the median (c) of the y-intercepts of every straight line through every point with the same slope m.

Then, the Theil-Sen straight-line is given by $y = mx + c$.

The Theil-Sen regression is equivariant under uniform scaling and translation, though it lacks equivariance under rotation. Consequently, normalization of an original point-cloud dataset is limited to translation and uniform scaling. This constraint poses a disadvantage for the Theil-Sen method, especially when the point-cloud suggests a near-vertical straight line.

The Theil-Sen regression's computational time complexity is $O(n^2)$ due to the slope determination through every pair of points in Step 1 of the algorithm outlined above. This computational overhead has led to the algorithm being perceived as expensive, particularly for large datasets.

While there is a valid case for using Theil-Sen regression in scenarios where the two-dimensional dataset does not adhere to a Euclidean metric space – such as dealing with (age, income) tuples – it may not be well-suited for coordinate metrology. In addition, Theil-Sen regression is designed for two-dimensional datasets only. In contrast, the orthogonal l_1-norm emerges as a superior alternative, not only for 2D point-clouds but also for three-dimensional problems, as detailed in the following Section 5.

4. Computing Median Plane for a 3d Point-Cloud

A plane in space is part of a 3-parameter family of surfaces [11]. Denoting these three parameters as c_1, c_2, and c_3, the task to find a median plane for a point-cloud $\{q_1, \dots, q_n\}$, like the one shown in Fig. 6, can be formulated as the optimization problem:

$$\min_{c_1,c_2,c_3} \sum_{i=1}^{n} |d_i(c_1, c_2, c_3)|. \qquad (4)$$

Here, $d_i(c_1, c_2, c_3)$ is the perpendicular distance from the point q_i to the plane being sought, as shown in Fig. 4(b). This distance could be signed, depending on side of the plane q_i lies on, necessitating the use of its modulus in defining the objective function $\sum_{i=1}^{n} |d_i(c_1, c_2, c_3)|$. It is evidently a trivariate function of the three parameters c_1, c_2, and c_3. Representing the plane can be achieved in at least two different ways, leading to two distinct parameterizations. The choice of the representation and associated parameterization will influence the geometric shape of the objective function and the effectiveness of a search method in finding the minimum. These considerations are discussed in the following Section 5.1.

4.1. Explicit and Implicit Representations of a Plane in Space

An explicit representation of a plane in space is provided by the well-known equation $z = ax + by + c$, where the partial slopes a and b, and the z-intercept c serve as the three parameters. This mirrors the explicit representation of a straight line in a plane, as discussed in Section 4.1 The largest feasible solution space for the plane under this representation is bounded by $a \in (-\infty, +\infty)$, $b \in (-\infty, +\infty)$, and $c \in (-\infty, +\infty)$. Similar to the case of the straight line explained in Section 4.1, this representation becomes impractical when the plane is vertical. It can also encounter numerical issues if the plane is near-vertical, as the partial slopes and z-intercept will assume very large absolute values.

To circumvent this problem, an alternative is to opt for an implicit representation in the form $x \cos\varphi \sin\theta + y \sin\varphi \sin\theta + z \cos\theta - r = 0$, where a radius r and angles φ and θ serve as the three parameters [14]. This mirrors the implicit representation for a straight line in a plane (the Hesse Normal Form) discussed in Section 4.1. Fig. 13 shows a geometrical interpretation of the three parameters. In this context, r is the perpendicular distance from the origin to the plane; it can be zero if the plane passes through the origin. This allows a radius vector, to be drawn to an imagined sphere centered at the origin, with the plane being tangential to the sphere. If the plane passes through the origin, the radius vector can simply be the unit vector normal to the plane. The angle θ is the angle between the z-axis and the radius vector. The angle φ is the angle between the x-axis and the projection of the radius vector onto the xy-plane. The largest feasible solution space for the straight line under this representation is bounded by $0 \leq r < \infty$, $-\pi \leq \varphi < \pi$, and $0 \leq \theta < \pi$. There are instances where r might assume negative values, which will be elaborated upon in Section 5.2. In real datasets, r can be bounded from above using a bounding box around the input point-cloud. It is important to note that all angles are measured in radians.

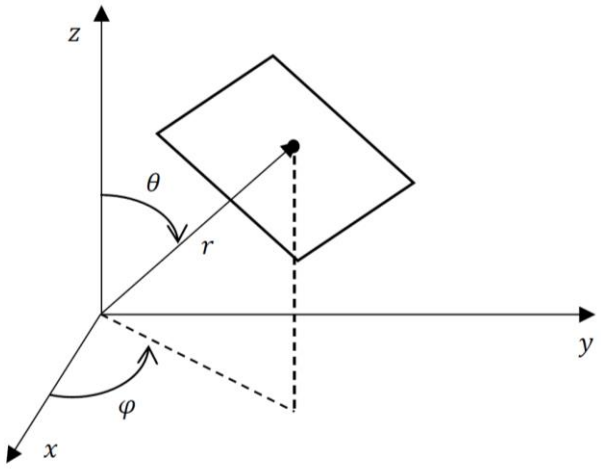


FIGURE 13. IMPLICIT REPRESENTATION FOR A PLANE.

When an explicit representation is used, the distance of Eq. (4) can be expressed as

$$d_i(a,b,c) = (z_i - ax_i - by_i - c) \Big/ \sqrt{a^2 + b^2 + 1} \qquad (5)$$

where (x_i, y_i, z_i) are the Cartesian coordinates of the point q_i in the point-cloud. This distance is positive if q_i lies above the plane, zero if it lies on the plane, and negative if it lies below the plane. If, on the other hand, an implicit representation is used, the distance formula becomes

$$d_i(r,\varphi,\theta) = x_i \cos\varphi \sin\theta + y_i \sin\varphi \sin\theta + z_i \cos\theta - r. \qquad (6)$$

This distance is negative if q_i lies on the same side of the plane as the origin, zero if it lies on the plane, and positive otherwise.

Regrettably, visualizing the objective functions, as demonstrated in Section 4.1 for the case of a straight line, is not feasible for a plane due to the trivariate nature of the objective functions. Nevertheless, many of the insights gained from the two-dimensional investigations in Section 4 can be applied to inform the computation of the median plane for a 3D point-cloud, as elaborated in the following Section 5.2.

4.2. Computational Details

Applying the same strategy outlined in Section 4.2, the Powell and Nelder-Mead methods in Scipy can be employed for the normalized 3D point-cloud depicted in Fig. 6. This allows the optimization problem of Eq. (4) to be solved using the distance formulas provided in Eqs. (5) and (6).

For the explicit representation, a suitable guess to initiate the search process is given by setting $a = b = c = 0$. This simplification is solely due to the normalized nature of the dataset

presented in Fig. 6, leveraging the equivariance and invariance properties of the point-clouds in Euclidean metric space as explained in Section 3.1. It is noteworthy that the mean plane, calculated using the orthogonal l_2-norm, for the normalized point-cloud of Fig. 6 is simply the xy-plane, for which $a = b = c = 0$.

Starting the search process for the implicit representation requires some additional considerations. For normalized 3D point-clouds, such as the one depicted in Fig. 6, it is reasonable to start with $r = \varphi = \theta = 0$. This might result in a negative value for r in the solution. However, as discussed in Section 4.2 for the implicit representation of a straight line, this is an artifact of the periodic nature of the trigonometric functions. It can be verified that the two seemingly different parameter tuples (r, φ, θ) and $(-r, \varphi - \pi, \pi - \theta)$ will both satisfy the same implicit equation $x\cos\varphi\sin\theta + y\sin\varphi\sin\theta + z\cos\theta - r = 0$, leading to the same plane. Therefore, to avoid a negative result for r, the search can be initiated with $\theta = \pi$, which addresses the case when the median plane has a negative z-intercept.

The computational results are summarized in Table 2, with Scipy reporting successful termination of the optimization processes. The Powell and Nelder-Mead methods demonstrate effective performance with the explicit representation. However, it is worth noting that the minimum value reported by the Nelder-Mead method, using the implicit representation, appears slightly higher than the other three results. Although the Nelder-Mead method performed much better with an alternative initial guess under the implicit representation, this raises some doubt about the suitability of this combination for computing median planes with the default initial guess.

TABLE 2. COMPARISON OF OPTIMIZATION METHODS FOR MEDIAN PLANES.

	Powell method	Nelder-Mead method
Explicit representation	f_{min} = 29.582916 a = 0.0003 b = −0.0051 c = −0.0065 Iterations = 3 Fun_ evals = 172	f_{min} = 29.582911 a = 0.0004 b = −0.0051 c = −0.0065 Iterations = 60 Fun_ evals = 103
Implicit representation	f_{min} = 29.582903 r = 0.0065 φ = 1.6472 θ = 3.1467 Iterations = 4 Fun_ evals = 185	f_{min} = 29.671660 r = 0.0061 φ = −0.0011 θ = 3.1423 Iterations = 40 Fun_evals = 73

Applying the same reasoning as for the computation of median straight lines, it can be reasonably anticipated that the computational time complexity of median plane computation is also $O(nh)$, with h the number of function evaluations (Fun_evals in Table 2). A comparison between Tables 1 and 2 unveils a surprising observation: despite the increase in the number of

points from 30 in Table 1 to 900 in Table 2, the number of function evaluations, h, remains in the range of 100 to 200. In essence, h does not appear to significantly increase with n, suggesting that the computational time complexity for both median straight lines and planes might be predominantly linear in n. This aligns it well with the computation of mean straight lines and planes, marking a promising avenue for further research in computational time complexity.

A valuable approach to visualize a median straight line or plane within a point-cloud is to generate a box plot [4, 5], exemplified in Fig. 14 for the 3D point-cloud presented in Fig. 6. By defining the orthogonal distances of points from the median plane as deviations and utilizing them in a box plot, a 3D problem has been transformed into a one-dimensional representation for both visualization and statistical analysis. A box plot is a concise, visual representation of the one-dimensional cumulative distribution function (CDF) of the deviations under investigation. This methodology has already demonstrated its effectiveness in dealing with outliers in geometrical measurements, as illustrated in an earlier study [15] that utilized the 2D point-cloud shown in Fig. 5 as an example.

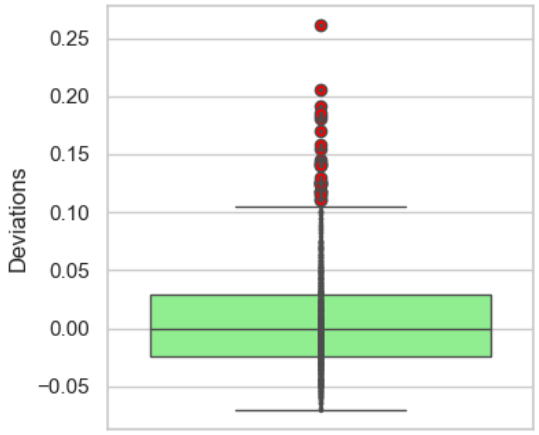


FIGURE 14. BOX PLOT OF POTENTIAL OUTLIERS.

An examination of Fig. 14 reveals at least two interesting facts. Firstly, the median plane, as anticipated, bisects the 3D point-cloud, with the zero-deviation aligning with the median line marker in Fig. 14. Secondly, there are 28 potential outliers, highlighted in red (appearing above the upper horizontal line) in Fig. 14, presenting themselves as candidates for further investigation.

The 900 deviations, each corresponding to a point in the point-cloud of Fig. 6, are densely plotted on one vertical line in Fig. 14. To enhance clarity and gain a more nuanced understanding of the distribution, these deviations are jittered horizontally in the box plot depicted in Fig. 15. This jittering technique provides a clearer visual appreciation of the quartile zones and strongly suggests a more general and potent approach to specify tolerance zones in geometrical product specifications [16, 17] and their metrological verification. The generality of

this approach also signals an expanded role for rank-order statistics in geometrical tolerancing and related metrology standards.

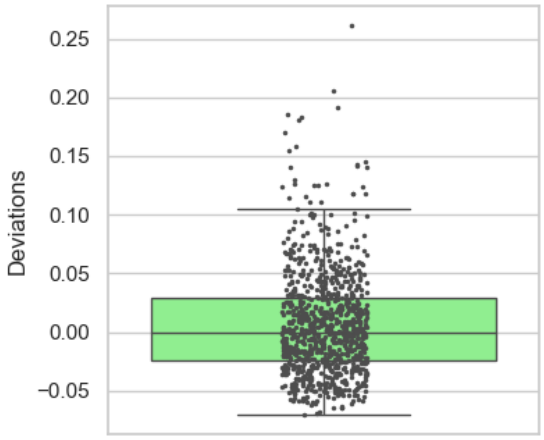


FIGURE 15. BOX PLOT WITH JITTERED POINTS.

5. Summary and Future Directions

This paper establishes a theoretical foundation for computing median straight lines and planes for point-clouds. Derived from the optimization formulation for mean and median in univariate cases, the theory readily extends to multidimensional scenarios. The results emphasize the use of commonly available, high-quality software packages for efficient computation, with the performance of direct solvers for optimization proving comparable to the computation of mean straight lines and planes. This eliminates a significant barrier in computational coordinate metrology, facilitating the inclusion of median and, more broadly, rank-order statistics in a metrologist's computational toolbox. Furthermore, this paper demonstrates how the theory and computations, leveraging equivariant and invariant properties of point-clouds in Euclidean metric space and multivariate optimization, transform multi-dimensional coordinate metrology problems into univariate statistical problems. This transformative approach is applicable to identifying potential outliers in coordinate metrology and defining quartile and percentile tolerance zones for geometrical specifications. The implications of these applications are significant for the development of future standards in geometrical tolerancing and associated metrological practices.

Several future research directions, including a more detailed empirical study of the computational time complexity, are already suggested within the body of the paper. Additionally, there are other promising avenues for future research. The theoretical foundation and computational methods can be extended to encompass median curves and surfaces, such as circles, spheres, cylinders, cones, and tori. This extension is a natural progression from the established methods for computing mean curves and surfaces using total least-squares techniques [2, 3].

Another crucial research direction involves the generalization of computing median curves and surfaces when the input is derived from continuous manifolds rather than discrete point-clouds. This requires replacing the summation in the objective function with an integration of a function of a curve/surface of known type but unknown parameters, which leads to a variational formulation of the problem. This methodology has already been demonstrated for establishing datum planes using constrained total least-squares [18]. Such an extension facilitates standards for the specification of tolerances without explicit reference to the sampling issues inherent in extracting point-clouds from manufactured surfaces. Equally significant is the theoretical insight that enables this approach, placing variational formulation at the core of computational coordinate metrology. It is noteworthy that a similar paradigm shift has already occurred in the development of classical and modern mechanics [19, 20].

References

[1] Srinivasan, V., Shakarji, C. M., Morse, E. P. On the enduring appeal of least-squares fitting in computational coordinate metrology. ASME J. Comput. Inf. Sci. Eng. Mar 2012, 12(1): 011008.
[2] Forbes, A. B. Least-squares best-fit geometric elements. NPL Report DITC 140/89. UK: National Physical Laboratory, 1991.
[3] Shakarji, C. M. Least-squares fitting algorithms of the NIST algorithm testing system. J. Res. Natl. Inst. Stand. Technol. Vol. 103, pp. 633-641, 1998.
[4] Wilcox, R. R. Fundamentals of modern statistical methods, 2nd Edition. USA: Springer, 2010.
[5] Matplotlib: Visualization with Python. https://matplotlib.org/. Accessed Aug. 16, 2023.
[6] Spiegel, M., Stephens, L. Outline of statistics. Sixth Edition, Schaum's outlines, McGraw Hill, New York, 2017.
[7] ASTM E178-21. Standard practice for dealing with outlying observations. USA: ASTM International, 2021.
[8] Gill, P. E., Murray, W., Wright, M. H. Practical optimization. UK: Emerald Group, 1982.
[9] SciPy: Fundamental algorithms for scientific computing in Python. https://scipy.org/. Accessed Aug. 16, 2023.
[10] Nievergelt, Y. Total Least Squares: State-of-the-Art Regression in Numerical Analysis, SIAM Review, Volume 36, Issue 2, June 1994, Pages: 258 – 264, DOI: 10.1137/1036055
[11] Srinivasan, V. Theory of dimensioning: An introduction to parameterizing geometric models, Marcel Dekker, New York, 2004.
[12] Bocher, M. Plane analytic geometry. USA: H Holt, 1915.
[13] Jupyter Lab, jupyter.org, Accessed Dec. 25, 2023.
[14] ISO 80000-2:2019, Quantities and units, Part 2: Mathematics, International Organization for Standardization, Geneva, 2019.
[15] Yin, K., Qi, Q., Morse, E., Shakarji, C., and Srinivasan, V. On dealing with outliers in geometrical measurements, 18th CIRP Conference on Computer Aided Tolerancing, 2024.
[16] ASME Y14.5: 2018. Dimensioning and Tolerancing. American Society of Mechanical Engineers, New York, 2019.
[17] ISO 1101:2017. Geometrical product specifications (GPS) Geometrical tolerancing: Tolerances of form, orientation, location and run-out. International Organization for Standardization, Geneva, 2017.
[18] Shakarji, C., Srinvasan, V. Toward a new mathematical definition of datums in standards to support advanced manufacturing. ASME *J. Comput. Inf. Sci. Eng*. Apr 2023, 23(2): 021013.
[19] Landau, L. D., Lifshitz. Mechanics. Butterworth-Heinemann; 3rd edition, 1976.
[20] Feynman, R., Leighton, R., Sands, M. The Feynman lectures on physics. Basic Books; New Millennium edition, 2011.