Metric and Trigonometric Pruning for Clustering of Uncertain Data in 2D Geometric Space

Wang Kay Ngai[a], Ben Kao[*,a], Reynold Cheng[a], Michael Chau[b], Sau Dan Lee[a], David W. Cheung[a], Kevin Y. Yip[c,d]

[a]*Department of Computer Science, The University of Hong Kong, Hong Kong*
[b]*School of Business, The University of Hong Kong, Hong Kong*
[c]*Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong*
[d]*Department of Molecular Biophysics and Biochemistry, Yale University, New Haven, Connecticut, USA*

Abstract

We study the problem of clustering data objects with location uncertainty. In our model, a data object is represented by an uncertainty region over which a probability density function (pdf) is defined. One method to cluster such uncertain objects is to apply the UK-means algorithm [1], an extension of the traditional K-means algorithm, which assigns each object to the cluster whose representative has the smallest *expected distance* from it. For arbitrary pdf, calculating the expected distance between an object and a cluster representative requires expensive integration of the pdf. We study two pruning methods: pre-computation (PC) and cluster shift (CS) that can significantly reduce the number of integrations computed. Both pruning methods rely on good bounding techniques. We propose and evaluate two such techniques that are based on metric properties (Met) and trigonometry (Tri). Our experimental results show that Tri offers a very high pruning power. In some cases, more than 99.9% of the expected distance calculations are pruned. This results in a very efficient clustering algorithm.[1]

[*]Corresponding author

Email addresses: wkngai@cs.hku.hk (Wang Kay Ngai), kao@cs.hku.hk (Ben Kao), ckcheng@cs.hku.hk (Reynold Cheng), mchau@business.hku.hk (Michael Chau), sdlee@cs.hku.hk (Sau Dan Lee), dcheung@cs.hku.hk (David W. Cheung), kevinyip@cse.cuhk.edu.hk (Kevin Y. Yip)

[1]Part of this paper appears in Ngai et al., 2006 [2], in which the algorithms PC and

Key words: Clustering, data uncertainty

1. Introduction

Clustering is a technique that has been widely studied and used in real applications. Many efficient algorithms, including the well-known and widely applied K-means algorithm, have been devised to solve the clustering problem efficiently. Traditionally, clustering algorithms deal with a set of objects whose positions are accurately known, and do not address situations in which object locations are uncertain. Data uncertainty is, however, inherent in many real-life applications due to factors such as the random nature of the physical data generation and collection processes, measurement errors, and data staling. Recent works (e.g., [3, 4, 5]) have also suggested to protect location privacy by lowering the precision of a user's location, which poses problems for traditional clustering algorithms.

In this paper we study the problem of clustering spatial objects with location uncertainty. In our model, an object's location is represented by a spatial probability density function (pdf). Our objective is to study the computational issues in adapting the traditional K-means algorithm to clustering uncertain objects, and to devise efficient algorithms for solving the clustering problem.

As a motivating example, let us consider the problem of clustering mobile devices. In many wireless network applications, mobile devices report their locations periodically to a remote server [6]. Each device can make low-power short-ranged communication to neighboring devices, or high-power long-ranged communication with the remote server directly. To reduce power consumption, batching protocols have been proposed. Under these protocols, certain devices are elected as leaders, whose job is to collect messages from neighboring devices through short-ranged communication. The leaders then send the collected messages in batch to the server through long-ranged communication [7, 8] (Figure 1). By batching messages, many long-ranged messages are replaced by short-ranged ones. The election of local leaders

CS were described. The idea of trigonometric pruning (Section 6), most of the empirical performance study (Section 7), discussion (Section 8), and all proofs of theorems (Appendixes), have not been previously published. The new materials amount to about 3/4 of the current paper.

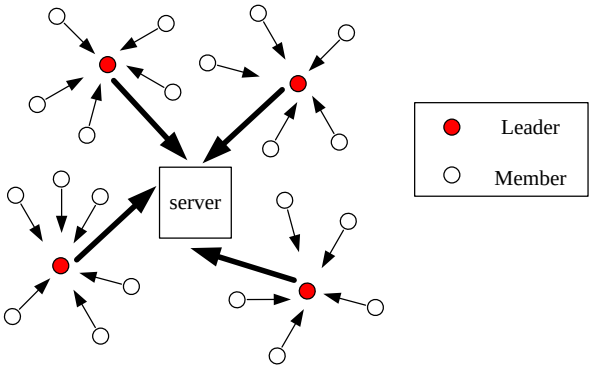


Figure 1: Reporting locations to cluster leaders using short-ranged communication has a much lower power consumption than making long-ranged communication with the server directly.

can be formulated as a clustering problem. The goal is to minimize the distance between every device and its corresponding local leader. This clustering problem differs from the traditional setting in the existence of data uncertainty:

- The physical instruments used for determining the device locations are accurate only up to a certain precision.
- The current locations of the mobile devices can only be estimated based on their last reported values, i.e., the data are always stale. Other practical problems, such as packet loss, could also increase the degree of uncertainty.
- Data uncertainty may also be introduced by the user to protect his location privacy. Particularly, the idea of *location cloaking* has been investigated [4, 5], where the actual location of a user is converted to a larger region, before it is sent to the service provider.

Due to uncertainty, the whereabouts of a mobile device can only be estimated by imposing an uncertainty model on its last reported location [9]. A typical uncertainty model requires knowledge about the moving speed of the device and whether its movement is restricted (such as a car moving in a road network) or unrestricted (such as a tracking device mounted on an

animal moving on plains). Typically, a 2D probability density function is defined over a bounded region to model such uncertainty.

Let us now formally define our uncertain data clustering problem. We consider a set of n objects o_i $(1 \leq i \leq n)$ in a 2-dimensional space. Each object o_i is represented by a probability density function (pdf) $f_i : \mathbb{R}^2 \rightarrow \mathbb{R}$ that specifies the probability density of each possible location of the object. The goal is to partition the objects into k clusters, such that each object o_i is assigned to a cluster c_i and that o_i is *close* to a cluster representative point p_{c_i} of c_i. To measure *closeness*, we define a distance function between an uncertain object and a cluster representative point as the expected distance between them:

$$ED(o_i, p_{c_i}) = \int f_i(x) d(x, p_{c_i}) dx, \tag{1}$$

where d is the Euclidean distance and the integration is taken over the *uncertainty region* (which is assumed to be bounded, as we will discuss below) in which the pdf integrates to one. Given a cluster c_i, its representative p_{c_i} is given by the mean of the centers of mass of all the objects assigned to c_i. The clustering goal is then to find c_i's (and thus p_{c_i}'s) such that the following objective function is minimized:

$$G = \sum_{i=1}^{n} ED(o_i, p_{c_i}) = \sum_{i=1}^{n} \int f_i(x) d(x, p_{c_i}) dx. \tag{2}$$

We assume that the pdf's can take any arbitrary form, which is important when the possible locations of a device are constrained by its dynamic environment, such as the road structure. The only additional requirement we impose on the pdf's is that each of them should integrate to one within a bounded region. This is a reasonable requirement for many applications. For example, the current location of a mobile device is restricted by its last reported location, its maximum speed, and the duration between two location reports [10, 11]. In location cloaking, the actual coordinates of a user's location were replaced by a uniform distribution over a bounded region [4, 5].

In a separate study [1], it was shown that the quality of clustering results could be improved by explicitly considering data uncertainty. An algorithm called UK-means (Uncertain K-means) was proposed to take data uncertainty into account during the clustering process. Experimental results showed

that UK-means consistently produced better clusters than the traditional K-means algorithm. Yet for arbitrary pdf's, expected distance calculations require costly numerical integrations. A straightforward implementation of UK-means for arbitrary pdf's would require a lot of such expected distance calculations, which are computationally impractical.

In this paper we study two pruning algorithms, namely pre-computation (PC) and cluster-shift (CS), which can significantly reduce the number of expected distance calculations of UK-means. The effectiveness of both algorithms relies on good bounds of expected distances. We propose and evaluate two bounding techniques that are based on metric properties (`Met`) and trigonometry (`Tri`). `Met` bounds are derived using the triangle inequality, and `Tri` bounds are obtained by a number of trigonometric rules. Our experimental results show that while the simple (`Met`) bounds are already powerful in pruning expected distance calculations, the more advanced `Tri` bounds provide further pruning power. In some of our experiments, more than 99.9% of the expected distance calculations are pruned. This results in a very efficient clustering algorithm.

The rest of this paper is organized as follows. Section 2 describes some related work on uncertain data mining in general and uncertain data clustering in particular. Section 3 describes the UK-means algorithm. We explain the performance bottleneck in UK-means and introduce a generic pruning framework for reducing the number of expected distance calculations. Based on this framework, we propose and analyze a number of pruning algorithms. In Section 4 we first describe a simple pruning algorithm that uses the idea of minimum bounding rectangles (MBRs). In Section 5, we introduce two pruning algorithms PC and CS. We will show that the effectiveness of these pruning algorithms relies heavily on the tightness of the lower and upper bounds of the expected distances. A major portion of this paper is thus dedicated to the study of such bounding techniques. Section 5 discusses the `Met` bounds, which are derived from the triangle inequality. Section 6 discusses the `Tri` bounds, which are derived from trigonometric rules. In Section 7 we evaluate the effectiveness of the various methods by extensive experiments. Section 8 discusses some observations and potential future developments, and Section 9 concludes the paper. Finally, the appendixes contain detailed mathematical proofs of our theorems.

Before we end this section, we remark that although we focus on 2D spaces (such as those related to geographical and location-based applications), our pruning algorithms and bounding techniques are theoretically applicable to

high-dimensional spaces as well. In particular, all the mathematical proofs of our theorems can be extended to high-dimensional spaces. The computational overheads incurred in the pruning methods, however, are higher in higher dimensional spaces.

2. Related work

There has been significant research interest in uncertain data management in recent years. Data uncertainty has been broadly classified into existential uncertainty and value uncertainty. Existential uncertainty appears when it is uncertain whether an object or a data tuple exists. For example, a data tuple in a relational database could be associated with a probability that represents the confidence of its presence [12, 13]. Value uncertainty, on the other hand, appears when a tuple is known to exist, but its values are not known precisely. A data item with value uncertainty is usually represented by a pdf over a finite and bounded region of possible values [1, 10, 11, 14, 15, 16]. In this paper we study the problem of clustering objects with value uncertainty.

There has been growing interest in uncertain data mining. In [1], the well-known K-means clustering algorithm is extended to the UK-means algorithm for clustering uncertain data. In that study, it is empirically shown that clustering results are improved if data uncertainty is taken into account during the clustering process. One caveat of the approach is the costly computation of numeric integration. To improve the performance of UK-means, pruning techniques have been proposed. Examples include min-max-dist pruning [2], CK-means [17], and Voronoi-diagram-based methods [18]. Our methods are similar to those of min-max-dist pruning [2] in that they all attempt to estimate bounds of expected distances (see Equation 1), which are then used to prune unnecessary integrations. The difference is that we apply trigonometric rules in bound estimations. As we will show later, our methods result in much tighter bounds and hence much more effective pruning algorithms. A recent publication has reported advancements on the topic of MBR-based pruning [19]. Our problem setting represents a special case of their general model. In this degenerate case, though, their proposed new technique reduces back to min-max-dist pruning, and hence cannot be applied to our problem to improve the pruning effectiveness[2].

[2]Please refer to Section 5 and Figures 8–10 of [19].

Voronoi-diagram-based (VD-based) clustering algorithms [18] represent a very different approach. Under VD-based algorithms, a Voronoi diagram of the cluster representatives is first constructed. The algorithms then check whether an uncertain object o's MBR lies entirely within a Voronoi cell c. If so, the cluster representative that corresponds to cell c must be the closest one to object o. Hence o is assigned to that cluster without computing any expected distances. We will briefly compare and contrast the VD-based approach and our bounding approach in Section 8.2.

Apart from studies in partition-based uncertain data clustering, other directions in uncertain data mining include density-based clustering (e.g., FDBSCAN [20]), frequent itemset mining [21] and density-based classification [22]. For density-based clustering, two well-known algorithms, namely, DBSCAN [23] and OPTICS [24] have been extended to handle uncertain data. The corresponding algorithms are called FDBSCAN [20] and FOPTICS [25], respectively. In DBSCAN, the concepts of *core objects* and *reachability* are defined. Clusters are then formed based on these concepts. In FDBSCAN, the concepts are re-defined to handle uncertain data. For example, under FDBSCAN, an object o is a core object if the probability that there is a "good number" of other objects that are close to o exceeds a certain probability threshold. Also, whether an object y is "reachable" from another object x depends on both the probability of y being close to x and the probability that x is a core object. FOPTICS takes a similar approach of using probabilities to modify the OPTICS algorithm to cluster uncertain data.

Subspace clustering [26] is a special kind of density-based clustering where clusters are hidden in (unknown) low-dimensional subspaces. The study in [27] extends the subspace clustering method in [28] to handle uncertain data using expected distances.

Other formulations of the clustering problem have also been studied. In [29] and [30], data uncertainty is represented by value intervals. Different distance measures, such as city-block distance and Minkowski distance, are extended to handle interval data. In [22], each uncertain object is modeled by a kernel function that estimates its errors. The average of the kernel function values of all the objects at any given point gives the *data density* at that point. An algorithm was proposed for using such data densities to solve the classification problem of uncertain data.

Clustering of uncertain data is also related to fuzzy clustering, which has long been studied in fuzzy logic [31]. In fuzzy clustering, a cluster is

represented by a fuzzy subset of objects. Each object has a "degree of belongingness" with respect to each cluster. The fuzzy c-means algorithm is one of the most widely used fuzzy clustering methods [32, 33]. Different fuzzy clustering methods have been applied on normal or fuzzy data to produce fuzzy clusters [34, 35]. A major difference between the clustering problem studied in this paper and fuzzy clustering is that we focus on hard clustering, for which each object belongs to exactly one cluster. Our formulation targets for applications such as mobile device clustering, in which each device should report its location to exactly one cluster leader.

3. The basic UK-means algorithm and Min-max-dist pruning

The problem of clustering uncertain data was first addressed in [1] where the UK-means algorithm was proposed. An objective of the paper was to study whether cluster quality could be improved by considering data uncertainty. The efficiency of UK-means was a secondary issue. In [2], it is shown that UK-means can be very inefficient. Min-max-dist pruning was proposed to significantly reduce clustering time. In this section, we review the basic UK-means algorithm and the min-max-dist pruning approach.

As the name suggests, UK-means is similar to the iterative K-means algorithm, which was designed for clustering conventional point data [36]. To form k clusters, UK-means starts by randomly selecting k points as initial cluster representatives. Each object o_i is then assigned to the cluster whose representative p_j has the smallest expected distance from o_i ($ED(o_i, p_j)$) among all clusters. After the assignment, cluster representatives are recomputed as the mean of the centers of mass of the assigned objects. The two steps form an iteration, which is repeated until the convergence of an objective score. Figure 2 shows the pseudocode of UK-means.

The most time-consuming step of UK-means is line 7, where an expected distance ($ED(o_i, p_j) = \int f_i(x)d(x, p_j)dx$) is calculated for each pair of object o_i and cluster representative p_j. Since cluster representatives shift from one iteration to another, the expected distances have to be recalculated in each iteration. With arbitrary pdf ($f_i()$), the integration has to be computed numerically by sampling values of $f(x)$ at different points x within the bounded uncertainty region of o_i, and computing the distance $d(x, p_j)$. To get an accurate estimate of the integrals, a large number of sample points x are required, making expected distance calculation an expensive operation. Suppose the algorithm runs for t iterations before convergence, a brute force implementa-

Algorithm UK-means (k: target no. of clusters)
1 for $j = 1$ to k do
2 $p_j = random\_point()$
3 $new\_obj\_score = \infty$
4 do {
5 $obj\_score = new\_obj\_score$
6 for $i = 1$ to n do // Assignment
7 $c_i = \arg\min_{j=1}^{k} ED(o_i, p_j)$
8 for $j = 1$ to k do // Re-computing representatives
9 $p_j = \frac{1}{|\{o_i : c_i = j\}|} \sum_{o_i : c_i = j} (\int x f_i(x) dx)$
10 $new\_obj\_score = \sum_{i=1}^{n} ED(o_i, p_{c_i})$
11 }
12 while ($|obj\_score - new\_obj\_score| > stopping\_threshold$)
End

Figure 2: The UK-means algorithm.

tion of the UK-means algorithm requires nkt expected distance calculations, which make the algorithm impractical when the total number of objects n is large. The integration in line 9 computes the centers of mass of objects. These centers need to be computed only once for the whole clustering process since they remain unchanged. The cost is thus relatively insignificant.

Is it possible to reduce the number of expected distance calculations without altering the clustering results? An idea is to use inexpensive distance calculations between some exact points with no uncertainty to setup lower and upper bounds of the expected distance between each object and each cluster representative, so that some representatives can be identified as not the closest one to an object without computing the exact expected distance between them.

Given two cluster representatives p_j and p_r, if a lower bound of $ED(o_i, p_j)$ is larger than an upper bound of $ED(o_i, p_r)$, then p_j is guaranteed not the closest cluster representative of o_i. In other words, the expensive expected distance calculation of $ED(o_i, p_j)$ can be pruned.

More specifically, for each object o_i and each cluster representative p_j, we use a method (to be described in the coming sections) to derive a lower bound $MinDist_{ij}$ and an upper bound $MaxDist_{ij}$ of $ED(o_i, p_j)$. Among all the upper bounds of o_i with respect to different cluster representatives, the smallest one

is called the min-max distance $MinMaxDist_i = \min_j MaxDist_{ij}$. Let p_r be a cluster representative with the smallest upper bound (i.e., $MaxDist_{ir} = MinMaxDist_i$). Any cluster representative p_j with $MinDist_{ij}$ larger than $MinMaxDist_i$ cannot be the one closest to o_i, since

$$\begin{aligned} ED(o_i, p_j) &\geq MinDist_{ij} \\ &> MinMaxDist_i \\ &= MaxDist_{ir} \\ &\geq ED(o_i, p_r), \end{aligned} \tag{3}$$

Thus, the expected distances between o_i and all p_j's such that $MinDist_{ij} > MinMaxDist_i$ are pruned.

If there are two or more cluster representatives that cannot be pruned, their exact expected distances from object o_i are computed in order to determine which one is the closest. We call the whole procedure the Min-max-dist pruning method. The effectiveness of the pruning method depends on the tightness of the bounds $MinDist_{ij}$ and $MaxDist_{ij}$ as well as the overhead incurred by the computation of them. In the coming sections, we will discuss ways to obtain such bounds.

4. MBR-based bounds on expected distances

A simple method to obtain lower and upper bounds of the expected distance $ED(o_i, p_j)$ is to make use of a minimum bounding rectangle (MBR). For each object o_i, we define an MBR that covers the whole bounded uncertainty region of it. For each cluster representative p_j, we compute the minimum and maximum distances between p_j and the MBR of o_i (Figure 3(a)). These distances can be easily computed by simple geometry. It is obvious that such minimum and maximum distances are also lower and upper bounds of $ED(o_i, p_j)$ and thus can be used as $MinDist_{ij}$ and $MaxDist_{ij}$, respectively. For instance, if the minimum distance is $\underline{d_{ij}}$, then

$$\begin{aligned} ED(o_i, p_j) &= \int f_i(x) d(x, p_j) dx \\ &\geq \int f_i(x) \underline{d_{ij}} dx \\ &= \underline{d_{ij}} \int f_i(x) dx \\ &= \underline{d_{ij}}. \end{aligned}$$

(a) An object with a relatively small uncertainty region.

(b) An object with a relatively large uncertainty region.

Figure 3: Minimum and maximum distances for the Min-max-dist pruning method.

For example, in Figure 3(a), we have $MinDist_{i1} = 1$, $MinDist_{i2} = 5$, $MinDist_{i3} = 7$, $MaxDist_{i1} = 6$, $MaxDist_{i2} = 11$ and $MaxDist_{i3} = 14$. Since p_1 gives the smallest maximum distance, we have the min-max distance $MinMaxDist_i = MaxDist_{i1} = 6$. Then, as $MinDist_{i3} = 7$, which is larger than $MinMaxDist_i = 6$, p_3 cannot be the cluster representative closest to o_i and thus the expected distance $ED(o_i, p_3)$ needs not be computed.

The method is effective in saving expected distance calculations of cluster representatives that are much farther away from the closest one. However, it suffers when the MBR gives poor distance bounds. This occurs when the uncertainty region of an object is large relative to the difference in expected distances from the different cluster representatives. For example, in Figure 3(b), the maximum distance of p_1 is increased to 8 due to a larger uncertainty region. The min-max distance $MinMaxDist_i$, now being 8, is no longer smaller than $MinDist_{i3}$. Therefore p_3 cannot be pruned and $ED(o_i, p_3)$ needs to be computed.

In the coming sections, we will discuss ways to tighten the bounds *MinDist* and *MaxDist* in order to achieve more effective pruning.

5. Metric bounds on expected distances

In this section we describe two bounding methods based on the triangle inequality, which takes two forms:

For any three points x, y and z, we have,

$$\begin{aligned} d(x,y) &\leq d(x,z) + d(z,y), \text{ and} \\ d(x,y) &\geq |d(y,z) - d(x,z)|. \end{aligned}$$

We now introduce two pruning methods that make use of the bounds derived from the triangle inequality.

5.1. The pre-computation (PC) method

Our first approach is to pre-compute some expected distances and utilize the triangle inequality to obtain better bounds. A similar idea applied to hierarchical clustering on data without uncertainty was proposed in [37]. Suppose that before the clustering process starts, we pick, for each object o_i, a fixed *anchor point* y in the object space and pre-compute the expected distance $ED(o_i, y)$. By using the triangle inequality, we can derive the following upper bound on the expected distance between object o_i and a cluster representative p_j in terms of y:

$$\begin{aligned} ED(o_i, p_j) &= \int f_i(x) d(x, p_j) dx \\ &\leq \int f_i(x)[d(x, y) + d(y, p_j)]dx \\ &= ED(o_i, y) + d(y, p_j). \end{aligned} \tag{4}$$

Since $ED(o_i, y)$ is pre-computed, the upper bound can be computed by an inexpensive distance calculation of $d(y, p_j)$. One can then compare the bound obtained from Inequality 4 with the bound obtained from the MBR method and use the smaller (i.e., tighter) one of the two as $MaxDist_{ij}$. This will guarantee a pruning effectiveness not worse than that of the MBR method.

Similarly, a lower bound on $ED(o_i, p_j)$ can be derived:

$$\begin{aligned} ED(o_i, p_j) &= \int f_i(x) d(x, p_j) dx \\ &\geq \int f_i(x)|d(y, p_j) - d(x, y)|dx \\ &\geq |\int f_i(x)[d(y, p_j) - d(x, y)]dx| \\ &= |d(y, p_j) - ED(o_i, y)|. \end{aligned} \tag{5}$$

Again, this lower bound can be compared with the one obtained from the MBR method and the larger one is used as $MinDist_{ij}$. We call the pruning procedure based on the bounds given by Inequalities 4 and 5 the pre-computation method, or PC for short.

An interesting issue concerning the PC method is the choice of the anchor point y. Obviously, if y equals p_j, then the bounds are tight since $d(y, p_j)$

would be zero. In general, choosing an anchor point y that is close to a cluster representative p_j would result in good lower and upper bounds of $ED(o_i, p_j)$. Unfortunately, since there are k cluster representatives (p_j) scattered about in the space, it is not possible to pick an anchor point that is close to all p_j's.

To tackle the problem, let us examine the upper bound $ED(o_i, y)+d(y, p_j)$ again. Since p_j is unknown in advance, a best-effort approach is to pick an anchor point y that minimizes the first term, $ED(o_i, y)$. Doing so would also make the inequality $d(y, p_j) > ED(o_i, y)$ more likely to be true, which means minimizing $ED(o_i, y)$ would not only attempt to tighten the upper bound, but also simultaneously attempt to make a tighter lower bound. It is thus important to know where can we find y that minimizes $ED(o_i, y)$. The exact location is in general hard to obtain, but fortunately the following result helps reduce the possible candidates.

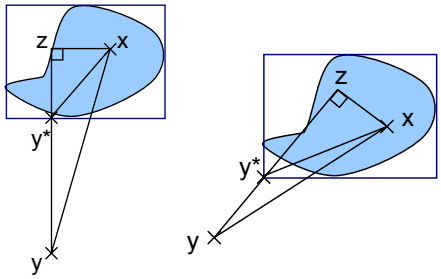
Lemma 5.1. *The anchor point y that minimizes $ED(o_i, y)$ must lie inside the MBR of o_i.*

Proof. Suppose that a point y is outside the MBR of o_i and y^* is the point on the boundary of the MBR that is closest to y. There are two possible cases, either the line yy^* is perpendicular to an edge of the MBR (Figure 4(a)), or y^* is at a corner of the MBR (Figure 4(b)). In both cases, consider any point x inside the uncertainty region of o_i, and let z be the point on the extension of yy^* such that xz is perpendicular to yz. If x lies on the line yy^* (i.e., $x = z$), then $d(x, y) > d(x, y^*)$; Otherwise, $d(x, y) = \frac{d(x,z)}{\cos \angle yxz} > \frac{d(x,z)}{\cos \angle y^*xz} = d(x, y^*)$. Therefore in all cases, $ED(o_i, y) = \int f_i(x)d(x, y)dx > \int f_i(x)d(x, y^*)dx = ED(o_i, y^*)$, which proves the lemma. □

The lemma suggests that we only need to pick anchor points for an object o_i from its MBR. In general, having more points raises the chance of getting tighter bounds, but also increases the pre-computation overhead. As we will show later in Section 7 using our experimental results, the pruning methods that use PC are very effective even with only a single anchor point. We will therefore stay with the one-point scheme with the anchor point at the center of the MBR, unless otherwise stated.

5.2. The cluster shift (CS) method

In the last section, we mentioned the difficulty of choosing an anchor point y that minimizes the second term of the upper bound $ED(o_i, y) + d(y, p_j)$ (Inequality 4), as the location of p_j is unknown in advance. Yet, if by some



(a) (b)

Figure 4: A point that minimizes its expected distance to an Object o_i must lie within the MBR of o_i.

means there exists a point y such that (1) the expected distance $ED(o_i, y)$ has already been computed and (2) y is close to p_j, then y can be used to set up good distance bounds. Such a point indeed exists naturally during the clustering process based on the following observation.

Consider two consecutive iterations h and $h+1$ in the clustering process where the representative of cluster j shifted from point p'_j to point p_j. Since cluster representatives usually shift by small distances especially in the later iterations of the clustering process, $d(p'_j, p_j)$ should be small. If $ED(o_i, p'_j)$ was computed in iteration h because pruning has failed to eliminate the computation, then point p'_j satisfies the two conditions mentioned above and can be used as a good anchor point for bounding $ED(o_i, p_j)$. Again, the upper and lower bounds are derived from the triangle inequality:

$$ED(o_i, p_j) \quad \leq \quad ED(o_i, p'_j) + d(p_j, p'_j), \tag{6}$$

$$ED(o_i, p_j) \quad \geq \quad |ED(o_i, p'_j) - d(p_j, p'_j)|. \tag{7}$$

As in the case of the PC method, the bounds can be computed using inexpensive distance calculations. We call the resulting pruning method the Cluster Shift method, or CS for short. A similar idea was used in [38] for speeding up the K-means algorithm when applied to conventional data.

Note that even if the expected distance $ED(o_i, p'_j)$ was not computed in iteration h, we can still apply the idea of the CS method by considering *the most recent* representative p^*_j of cluster j whose expected distance $ED(o_i, p^*_j)$ from o_i has been computed. An interesting property of the CS method is that if $ED(o_i, p'_j)$ is not available, it means that $ED(o_i, p'_j)$ was pruned by the bounds using p^*_j. In that case, since p'_j and p_j are close, if the bounds led to the pruning of p'_j, it is very likely that $ED(o_i, p_j)$ can be pruned by the bounds using p^*_j as well. On the other hand, if $ED(o_i, p'_j)$ is available, we can use it to bound $ED(o_i, p_j)$. So, either way, chances are that the expected distance $ED(o_i, p_j)$ can be pruned.

There are two main differences between the CS method and the PC method. First, the CS method does not involve any pre-computation of expected distances. It simply uses a previously computed result. There is thus no extra pre-computation overhead. Second, while the anchor point y picked for the PC method is close to the object o_i, the anchor point (e.g., p'_j) used by the CS method is close to the cluster representative p_j. A downside of the CS method is its dependency on the dynamic clustering process, such as whether $ED(o_i, p'_j)$ has been previously computed and thus p' can be used as an anchor point, which is largely unpredictable. Another issue of the CS method is that if the cluster representatives experience large shifts (e.g., during the first few iterations of the clustering process), the derived bounds could be loose. In that case, the algorithm will automatically fall back to those bounds obtained by the basic MBR method (Section 4).

6. Trigonometric bounds on expected distances

The triangle inequality used in the previous section is simple in bounding the expected distances, yet the bounds are not very tight. This is because the method only considers the distances between the anchor point y, the cluster representative p_j and the points x in the MBR of the uncertain object o_i, but not the angles between them. If we make use of these angles, we will be able to find tighter bounds. For instance, let us take a look at Inequality 4. The upper bound of $ED(o_i, p_j)$ is given by the sum of $ED(o_i, y)$ and $d(y, p_j)$. If the vectors $\overrightarrow{xy}$ and $\overrightarrow{yp_j}$ are far from parallel for most of the points x in the MBR of o_i, then $ED(o_i, p_j)$ can be much smaller than $ED(o_i, y) + d(y, p_j)$. That is, the upper bound can be very loose. This problem is illustrated in Figure 5, where p_j is denoted as p in order to simplify the notation. Clearly,

for each point x in object o_i, the distance $d(x, p)$ can be more accurately computed in terms of the angles α, β and γ.

In this section we describe in detail how new bounds of $ED(o_i, p_j)$ can be obtained by using trigonometry. We will show later (see Section 6.5) that the farther away the anchor point y is from the MBR, the tighter the bounds will be (as long as we do not lose much numerical precision). The reason why a far-away anchor is picked is to limit the variation of the three angles (see Figure 5). We may thus consider the trigonometric bounds being complementary to the PC bounds discussed in Section 5, which prefer anchor points inside the MBR.

The new bounds based on trigonometry can be used in place of those based on the triangle inequality under both the PC method and the CS method. For the PC method, for each object, we pick an additional anchor point y that is far away from the object, and pre-compute the expected distance $ED(o_i, y)$. For the CS method, we use the cluster representative of a previous iteration whose expected distance from o_i has been computed as the anchor point. As discussed before, the different bounds can be used together by comparing their tightness and picking the best one. We will study the resulting pruning power and computational overhead empirically in Section 7.

We will first identify a few identities concerning the side lengths and angles of a triangle in Section 6.1. Then, we use these identities to derive bounds of $ED(o_i, p_j)$ in Sections 6.2–6.4. We have discovered that except in some degenerate cases, given a fixed anchor point, the trigonometric bounds are strictly better than those derived from the triangle inequality. This result is formally presented in Section 6.5. Note that the trigonometric bounds require the computation of bounds on certain trigonometric functions on the three angles. In Appendix A, we first describe algorithms for bounding the angles. Then in Appendix B, we show that the bounds on the trigonometric functions can be found easily using the bounds on the angles.

Before we go into the details, let us formally introduce some notations. First, we consider the triangle formed by the three points x, p, y. The corresponding angles in the triangle are α, β, γ, respectively, where all three angles are within $[0, \pi]$. Figure 5 illustrates such a triangle for the case where β is obtuse[3]. We treat y (the anchor point) and p (any cluster representative p_j)

[3]In this article, an angle is defined as obtuse if it is in $(\pi/2, \pi]$.

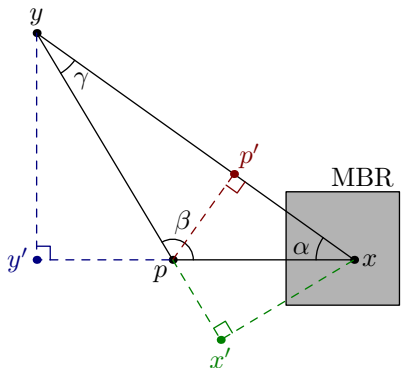


Figure 5: Notations for Section 6

as fixed points, and x as a variable point that is confined within the MBR of an uncertain object o_i. Note that the three angles all vary with x. We assume that x, p, y are all distinct points, so that $\left\|\overrightarrow{xp}\right\|$, $\left\|\overrightarrow{xy}\right\|$ and $\left\|\overrightarrow{yp}\right\|$ are all non-zero.

We use vector notations for our formulas and proofs. We denote the vector that points from a to b as $\overrightarrow{ab}$. If two vectors $\overrightarrow{ab}$ and $\overrightarrow{cd}$ are perpendicular, we write $\overrightarrow{ab} \perp \overrightarrow{cd}$; otherwise, we write $\overrightarrow{ab} \not\perp \overrightarrow{cd}$. For convenience, the distance between two points a and b is written as $\left\|\overrightarrow{ab}\right\| = d(a, b)$. The maximum (resp. minimum) value of a quantity q is denoted as $\overline{q}$ (resp. $\underline{q}$). So, $\underline{\alpha}$ denotes the minimum attainable value of angle α as x varies within the MBR of o_i, and $\overline{\cos \alpha}$ denotes the maximum attainable value of $\cos \alpha$, which happens to be equal to $\cos \underline{\alpha}$ because cosine is a decreasing function in the range $[0, \pi]$.

As a preview, we derive 5 types of bounds, called *Cosine Bounds* (COS), *Secant-A Bounds* (SEC-A), *Secant-B Bounds* (SEC-B), *Cosecant-A Bounds* (CSC-A) and *Cosecant-B Bounds* (CSC-B). These bounds are derived from different trigonometry identities and they are applicable under different conditions. Table 1 shows a summary of the bounds. Note that only upper bounds are shown in the table. The corresponding lower bounds can be obtained by swapping all quantities $\overline{q}$ with $\underline{q}$ in the formulae. First-time readers might want to skip the details of the bound derivations (Sections 6.2–6.4) and proceed to Section 6.5, where some theorems concerning the various bounds are presented.

6.1. Selected Trigonometric Identities

The basic idea of bounding $ED(o_i, p_j)$ is to look for identities that represent $\left\|\overrightarrow{xp}\right\|$ in terms of $\left\|\overrightarrow{xy}\right\|$, $\left\|\overrightarrow{yp}\right\|$, and trigonometric functions of α, β and γ. After taking integration, it is equivalent to representing $ED(o_i, p_j)$ in terms of $ED(o_i, y)$, $\left\|\overrightarrow{yp}\right\|$, and trigonometric functions of α, β and γ. Since $ED(o_i, y)$ is pre-computed and $\left\|\overrightarrow{yp}\right\|$ can be computed easily using an inexpensive distance calculation, by bounding the values of the trigonometric functions, we obtain bounds of $ED(o_i, p_j)$.

We have identified five such identities that can be computed efficiently:

1. COS identity:
$$\left\|\overrightarrow{xp}\right\| = \left\|\overrightarrow{xy}\right\| \cos\alpha + \left\|\overrightarrow{yp}\right\| \cos\beta. \tag{8}$$
2. SEC identities:
   (a) When $\overrightarrow{xy} \not\perp \overrightarrow{xp}$,
$$\left\|\overrightarrow{xp}\right\| = \sec\alpha(\left\|\overrightarrow{xy}\right\| - \left\|\overrightarrow{yp}\right\| \cos\gamma). \tag{9}$$
   (b) When $\overrightarrow{yp} \not\perp \overrightarrow{xp}$,
$$\left\|\overrightarrow{xp}\right\| = \sec\beta(\left\|\overrightarrow{yp}\right\| - \left\|\overrightarrow{xy}\right\| \cos\gamma). \tag{10}$$
3. CSC identities: when x, p, y are not collinear,
$$\left\|\overrightarrow{xp}\right\| = \left\|\overrightarrow{yp}\right\| \sin\gamma \csc\alpha, \tag{11}$$
$$\left\|\overrightarrow{xp}\right\| = \left\|\overrightarrow{xy}\right\| \sin\gamma \csc\beta. \tag{12}$$

Proof. The identities can be readily verified for the triangle shown in Figure 5. Here we give general proofs for all possible values of α, β and γ.

For (8),

$$\begin{aligned}
\overrightarrow{xp} &= \overrightarrow{xy} + \overrightarrow{yp} \\
\overrightarrow{xp} \cdot \overrightarrow{xp} &= \overrightarrow{xp} \cdot (\overrightarrow{xy} + \overrightarrow{yp}) \\
\left\|\overrightarrow{xp}\right\|^2 &= \overrightarrow{xp} \cdot \overrightarrow{xy} + \overrightarrow{xp} \cdot \overrightarrow{yp} \\
&= \left\|\overrightarrow{xp}\right\| \left\|\overrightarrow{xy}\right\| \cos\alpha + \left\|\overrightarrow{xp}\right\| \left\|\overrightarrow{yp}\right\| \cos\beta \\
\left\|\overrightarrow{xp}\right\| &= \left\|\overrightarrow{xy}\right\| \cos\alpha + \left\|\overrightarrow{yp}\right\| \cos\beta.
\end{aligned}$$

Geometrically, referring to Figure 5, Equation 8 states that $\left\|\overrightarrow{xp}\right\|$ is equal to $\left\|\overrightarrow{xy'} + \overrightarrow{y'p}\right\|$, where y' is the point on line xp so that $\overrightarrow{yy'} \perp \overrightarrow{xp}$.

For (9),

$$\begin{aligned}
\overrightarrow{xp} &= \overrightarrow{xy} + \overrightarrow{yp} \\
\overrightarrow{xy} \cdot \overrightarrow{xp} &= \overrightarrow{xy} \cdot (\overrightarrow{xy} - \overrightarrow{py}) \\
\left\|\overrightarrow{xy}\right\| \left\|\overrightarrow{xp}\right\| \cos\alpha &= \left\|\overrightarrow{xy}\right\|^2 - \left\|\overrightarrow{xy}\right\| \left\|\overrightarrow{py}\right\| \cos\gamma \\
\left\|\overrightarrow{xp}\right\| &= \sec\alpha(\left\|\overrightarrow{xy}\right\| - \left\|\overrightarrow{yp}\right\| \cos\gamma).
\end{aligned}$$

Geometrically, the equality states that $\left\|\overrightarrow{xp}\right\|$ is equal to $\left\|\overrightarrow{xp'}\right\| \sec\alpha$, where p' is the point on xy so that $\overrightarrow{pp'} \perp \overrightarrow{xy}$. Equation 10 can be derived in a similar fashion.

Equations 11 and 12 can be derived from the sine rule, which states that

$$\frac{\left\|\overrightarrow{yp}\right\|}{\sin\alpha} = \frac{\left\|\overrightarrow{xy}\right\|}{\sin\beta} = \frac{\left\|\overrightarrow{xp}\right\|}{\sin\gamma}.$$

Rearranging terms leads to (11) and (12) immediately. The geometrical interpretations of (11) and (12) are that $\left\|\overrightarrow{xp}\right\| = \left\|\overrightarrow{pp'}\right\| \csc\alpha$ and $\left\|\overrightarrow{xp}\right\| = \left\|\overrightarrow{xx'}\right\| \csc\beta$, where x' is the point that lies on line yp (projected if necessary) so that $\overrightarrow{x'x} \perp \overrightarrow{yp}$. □

6.2. COS bounds

We derive COS upper and lower bounds based on Identity (8). The bounds are so named because they depend on the cosine values of α and β:

$$\begin{aligned}
\left\|\overrightarrow{xp}\right\| &= \left\|\overrightarrow{xy}\right\| \cos\alpha + \left\|\overrightarrow{yp}\right\| \cos\beta \\
&\leq \left\|\overrightarrow{xy}\right\| \overline{\cos\alpha} + \left\|\overrightarrow{yp}\right\| \overline{\cos\beta}.
\end{aligned}$$

Multiplying both sides by $f_i(x)$ and performing integration, we get:

$$\begin{aligned}
\int f_i(x) \left\|\overrightarrow{xp}\right\| dx &\leq \int f_i(x) \left(\left\|\overrightarrow{xy}\right\| \overline{\cos\alpha} + \left\|\overrightarrow{yp}\right\| \overline{\cos\beta}\right) dx \\
ED(o_i, p) &\leq \overline{\cos\alpha} \int f_i(x) \left\|\overrightarrow{xy}\right\| dx + \overline{\cos\beta} \left\|\overrightarrow{yp}\right\| \int f_i(x) dx \\
&= ED(o_i, y) \overline{\cos\alpha} + \left\|\overrightarrow{yp}\right\| \overline{\cos\beta}. \qquad (13)
\end{aligned}$$

Similarly, we can find a lower bound:

$$ED(o_i, p) \geq ED(o_i, y)\,\underline{\cos\alpha} + \left\|\overrightarrow{yp}\right\|\underline{\cos\beta}. \tag{14}$$

6.3. SEC bounds

The SEC bounds consist of 2 sets of bounds. The SEC-A bounds involve the value of $\sec\alpha$ and are based on Identity (9). The SEC-B bounds involve the value of $\sec\beta$ and are based on Identity (10).

6.3.1. SEC-A bounds

The SEC-A bound is only applicable when there is no point $x \in \text{MBR}$ such that $\alpha = \pi/2$. This guarantees that $\sec\alpha$ is always well defined and hence Identity (9) is applicable.

Let S_{yp} be the sphere with yp being a diameter (Figure 6). Let B_{yp} be the closed ball containing all points inside S_{yp}, inclusively. Note that the chord yp subtends right angles only at points on S_{yp} (see Figure 6(a)). So, if S_{yp} intersects the MBR we know that SEC-A is not applicable, because points in the intersection will make $\alpha = \pi/2$ and hence Identity (9) is not applicable.

When S_{yp} does not intersect the MBR, we consider two cases. In the first case, the whole MBR lies outside S_{yp} (or equivalently, $B_{yp} \cap \text{MBR} = \emptyset$). In this case, α is always acute[4] and hence $\sec\alpha \geq 1$ (see Figure 6(b)). In the second case, the whole MBR lies inside S_{yp} (or equivalently, $\text{MBR} \subseteq B_{yp}$). In this case, α is always obtuse and hence $\sec\alpha \leq -1$ (see Figure 6(c)).

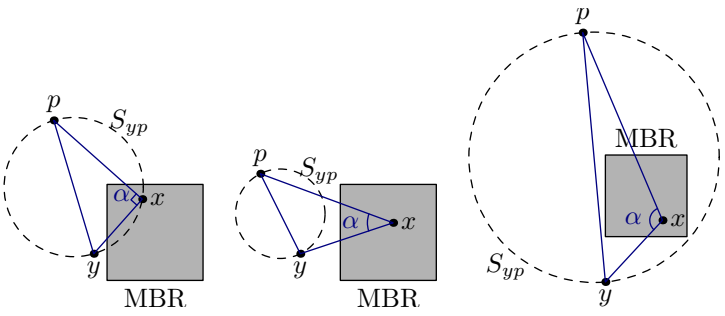
In the first case, since $\sec\alpha$ is positive and the left side of Identity (9) is non-negative, we know that the second factor on the right side of (9) must also be non-negative. So, we have

$$\begin{aligned}\left\|\overrightarrow{xp}\right\| &= \sec\alpha\,(\left\|\overrightarrow{xy}\right\| - \left\|\overrightarrow{yp}\right\|\cos\gamma)\\ &\leq \overline{\sec\alpha}\,(\left\|\overrightarrow{xy}\right\| - \left\|\overrightarrow{yp}\right\|\underline{\cos\gamma}).\end{aligned}$$

Multiplying both sides by $f_i(x)$ and performing integration, we get:

$$\begin{aligned}\int f_i(x)\left\|\overrightarrow{xp}\right\|dx &\leq \int f_i(x)\,\overline{\sec\alpha}\,(\left\|\overrightarrow{xy}\right\| - \left\|\overrightarrow{yp}\right\|\underline{\cos\gamma})\,dx\\ ED(o_i,p) &\leq \overline{\sec\alpha}\left(\int f_i(x)\left\|\overrightarrow{xy}\right\|dx - \left\|\overrightarrow{yp}\right\|\underline{\cos\gamma}\int f_i(x)dx\right)\\ &= \overline{\sec\alpha}\,(ED(o_i,y) - \left\|\overrightarrow{yp}\right\|\underline{\cos\gamma}).\end{aligned} \tag{15}$$

[4]In this article, an angle is acute if it is in $[0, \pi/2)$.



(a) not applicable (b) α: acute (c) α: obtuse

Figure 6: Different Cases for SEC-A

Similarly, a lower bound can be obtained:

$$ED(o_i, p) \geq \underline{\sec \alpha}\,(ED(o_i, y) - \left\|\overrightarrow{yp}\right\| \overline{\cos \gamma}). \tag{16}$$

In the second case, we note that $\sec \alpha$ is negative. The derivations are similar to those of the first case, except that the inequality sign is reversed. We thus get the bounds:

$$ED(o_i, p) \leq \underline{\sec \alpha}\,(ED(o_i, y) - \left\|\overrightarrow{yp}\right\| \overline{\cos \gamma}). \tag{17}$$

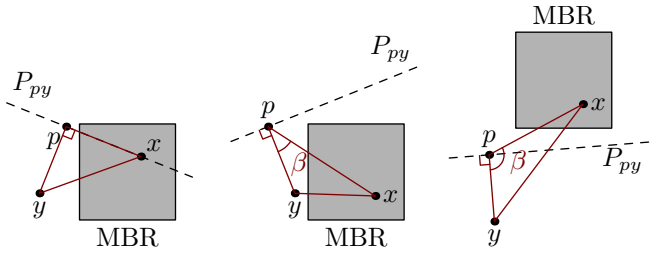
$$ED(o_i, p) \geq \overline{\sec \alpha}\,(ED(o_i, y) - \left\|\overrightarrow{yp}\right\| \underline{\cos \gamma}). \tag{18}$$

6.3.2. SEC-B bounds

Similar to the SEC-A bounds, the SEC-B bounds are only applicable when there is no point $x \in$ MBR such that $\beta = \pi/2$. This guarantees that $\sec \beta$ is always well defined and hence Identity (10) is applicable.

Note that points that make $\beta = \pi/2$ lie exactly on the hyper-plane P_{py} that passes through the point p and is perpendicular to $\overrightarrow{py}$. So, if P_{py} intersects the MBR, SEC-B is not applicable because the intersecting points induce $\beta = \pi/2$ (see Figure 7(a)).

When P_{py} does not intersect the MBR, we again consider two cases. In the first case, the whole MBR lies on the same side of P_{py} as y. In this case β is always acute (see Figure 7(b)). In the second case, the whole MBR lies on the side of P_{yp} opposite to y. In this case, β is always obtuse and hence $\sec \beta \leq -1$ (see Figure 7(c)).



(a) not applicable (b) β: acute (c) β: obtuse

Figure 7: Different Cases for SEC-B

In the first case, we note that $\sec\beta$ is positive and the left side of Identity (10) is non-negative. So, the second factor on the right side must be non-negative. This gives:

$$\left\|\overrightarrow{xp}\right\| \leq \overline{\sec\beta}\,(\left\|\overrightarrow{yp}\right\| - \left\|\overrightarrow{xy}\right\|\underline{\cos\gamma}).$$

Recall that Identity (10) is invalid when $x = y$, which may happen when $y \in$ MBR. However, when that happens,

$$\begin{aligned}\left\|\overrightarrow{xp}\right\| &= \left\|\overrightarrow{yp}\right\| \\ &= 1 \cdot (\left\|\overrightarrow{yp}\right\| - \left\|\overrightarrow{xy}\right\| \cdot 1)\end{aligned}$$

because $\overrightarrow{xy} = \vec{0}$ when $x = y$. Since $\sec\beta \geq 1$, we know that $\overline{\sec\beta} \geq 1$. Furthermore, $\cos\gamma \leq 1$ and hence $\underline{\cos\gamma} \leq 1$. Therefore,

$$\begin{aligned}\left\|\overrightarrow{xp}\right\| &= 1 \cdot (\left\|\overrightarrow{yp}\right\| - \left\|\overrightarrow{xy}\right\| \cdot 1) \\ &\leq \overline{\sec\beta}\,(\left\|\overrightarrow{yp}\right\| - \left\|\overrightarrow{xy}\right\|\underline{\cos\gamma})\end{aligned}$$

when $x = y$. So, whether or not $x = y$, we arrive at the same upper bound. The bound thus holds for all points $x \in$ MBR, regardless of whether $y \in$ MBR or not. Multiplying both sides by $f_i(x)$ and integrating, we get:

$$ED(o_i, p) \leq \overline{\sec\beta}\,(\left\|\overrightarrow{yp}\right\| - ED(o_i, y)\,\underline{\cos\gamma}). \tag{19}$$

A lower bound can be similarly derived:

$$ED(o_i, p) \geq \underline{\sec\beta}\,(\left\|\overrightarrow{yp}\right\| - ED(o_i, y)\,\overline{\cos\gamma}). \tag{20}$$

The case where β is always obtuse can be handled similarly. In that case, $\sec\beta$ is negative. The derivation is similar to the first case except that the inequality sign is reversed. The bounds so obtained are:

$$ED(o_i,p) \leq \underline{\sec\beta}\,(\left\|\overrightarrow{yp}\right\| - ED(o_i,y)\,\overline{\cos\gamma}). \quad (21)$$

$$ED(o_i,p) \geq \overline{\sec\beta}\,(\left\|\overrightarrow{yp}\right\| - ED(o_i,y)\,\underline{\cos\gamma}). \quad (22)$$

6.4. CSC bounds

The CSC bounds are applicable when there is no point $x \in$ MBR such that x, p, y are collinear. (This is to ensure that α and β do not attain the value of 0 or π, thus guaranteeing that $\csc\alpha$ and $\csc\beta$ are well-defined.) Consider the line L_{py} that passes through points p and y. If L_{py} does not intersect the MBR, then p, y, and any point x in the MBR are not collinear and Identities (11) and (12) are applicable. Using (11), we derive the CSC-A upper bound:

$$\begin{aligned}
\left\|\overrightarrow{xp}\right\| &\leq \left\|\overrightarrow{yp}\right\| \overline{\sin\gamma}\,\overline{\csc\alpha} \\
\int f_i(x)\left\|\overrightarrow{xp}\right\| dx &\leq \int f_i(x)\left\|\overrightarrow{yp}\right\| \overline{\sin\gamma}\,\overline{\csc\alpha}\,dx \\
ED(o_i,p) &\leq \left\|\overrightarrow{yp}\right\| \overline{\sin\gamma}\,\overline{\csc\alpha}\int f_i(x)\,dx \\
&= \left\|\overrightarrow{yp}\right\| \overline{\sin\gamma}\,\overline{\csc\alpha}. \quad (23)
\end{aligned}$$

Similarly, the CSC-A lower bound is given by:

$$ED(o_i,p) \geq \left\|\overrightarrow{yp}\right\| \underline{\sin\gamma}\,\underline{\csc\alpha}. \quad (24)$$

The CSC-B bounds are obtained similarly from (12):

$$ED(o_i,p) \leq ED(o_i,y)\,\overline{\sin\gamma}\,\overline{\csc\beta}. \quad (25)$$

$$ED(o_i,p) \geq ED(o_i,y)\,\underline{\sin\gamma}\,\underline{\csc\beta}. \quad (26)$$

6.5. Combining the Bounds Together

Finally, the Trigonometric bounds are obtained by combining the various bounds described above. For a particular pair of cluster representative and uncertain object, we compute the upper (resp. lower) bounds using COS, SEC-A, SEC-B, CSC-A and CSC-B, omitting those that are not applicable. The minimum (maximum) of all the applicable upper (lower) bounds

is used as the trigonometric upper (lower) bound. In case all upper (lower) bounds are inapplicable, we fall back to the ones obtained using the triangle inequality instead (see Inequality 4 and Inequality 5).

Since the trigonometric bounds are those obtained from the triangle inequality in case none of the bounds using COS, SEC-A, SEC-B, CSC-A, CSC-B is applicable, the trigonometric bounds are thus no worse than those given by the triangle inequality. On the other hand, when some specific bounds are applicable, we can show that the trigonometric bounds are *strictly better* than those using the triangle inequality under certain conditions. This interesting result is summarized by the following theorems.

Theorem 6.1. *When COS is applicable (i.e., when $p, y \notin MBR$), the upper bound (13) given by COS is always tighter (i.e., smaller) than or equal to that given by the triangle inequality (4). Equality holds if and only if the (straight) ray emerging from y in the direction of $\overrightarrow{py}$ intersects the MBR.*

Proof. Since the value of cosine is bounded above by 1, we have: $\overline{\cos\alpha} \leq 1$ and $\overline{\cos\beta} \leq 1$. Therefore, $ED(o_i, y)\overline{\cos\alpha} + \left\|\overrightarrow{yp}\right\| \overline{\cos\beta} \leq ED(o_i, y) + \left\|\overrightarrow{yp}\right\|$. Equality holds if and only if $\overline{\cos\alpha} = 1$ and $\overline{\cos\beta} = 1$, or equivalently, $\underline{\alpha} = \underline{\beta} = 0$. For $\underline{\alpha} = 0$, there must be a point x in the MBR such that x lies on the ray emerging from y in the direction of $\overrightarrow{py}$ or the ray emerging from p in the direction of $\overrightarrow{yp}$. For $\underline{\beta} = 0$, there must be a point x in the MBR so that x lies on the ray emerging from p in the direction of $\overrightarrow{py}$. For both of these to happen, we consider the intersection of these rays and deduce that $\underline{\alpha} = \underline{\beta} = 0$ if and only if the ray emerging from y in the direction of $\overrightarrow{py}$ intersects the MBR. □

To compare the trigonometric lower bound with the one obtained by the triangle inequality, we first establish the following lemmas.

Lemma 6.2. *When α is always acute, the lower bound (16) given by SEC-A $\geq ED(o_i, y) - \left\|\overrightarrow{yp}\right\|$. Equality holds if and only if the (straight) ray emerging from p in the direction of $\overrightarrow{yp}$ intersects the MBR.*

Proof. For acute values of α, $\sec\alpha \geq 1$. So, $\underline{\sec\alpha} \geq 1$. Also note that $\overline{\cos\gamma} \leq 1$. Therefore,

$$\begin{aligned} \text{lower bound given by SEC-A} &= \underline{\sec\alpha}\,(ED(o_i, y) - \left\|\overrightarrow{yp}\right\| \overline{\cos\gamma}) \\ &\geq 1 \cdot (ED(o_i, y) - \left\|\overrightarrow{yp}\right\| \cdot 1) \\ &= ED(o_i, y) - \left\|\overrightarrow{yp}\right\|. \end{aligned}$$

Equality holds if and only if $\underline{\sec\alpha} = 1$ and $\overline{\cos\gamma} = 1$, or equivalently, $\underline{\alpha} = \underline{\gamma} = 0$. For $\underline{\alpha} = 0$, there must be a point x in the MBR such that x lies on the ray emerging from y in the direction of $\overrightarrow{py}$ or the ray emerging from p in the direction of $\overrightarrow{yp}$. For $\underline{\gamma} = 0$, there must be a point x in the MBR such that x lies on the ray emerging from y in the direction of $\overrightarrow{yp}$. Considering the intersection of these rays, we have $\underline{\alpha} = \underline{\gamma} = 0$ if and only if the ray emerging from p in the direction of $\overrightarrow{yp}$ intersects the MBR. □

Lemma 6.3. *When β is always acute, the lower bound (20) given by SEC-B $\geq \left\|\overrightarrow{yp}\right\| - ED(o_i, y)$. Equality holds if and only if the line segment yp intersects the MBR.*

The proof is similar and hence omitted. Now, we are ready to prove the following theorem concerning the trigonometric lower bound.

Theorem 6.4. *When* both *SEC-A and SEC-B are applicable, the lower bound given by either of them is tighter (i.e., greater) than or equal to that given by the triangle inequality (5). Equality holds only if the (straight) ray emerging from y in the direction of $\overrightarrow{yp}$ intersects the MBR.*

Proof. Since SEC-A is applicable, α is either always acute or always obtuse. Also, since SEC-B is applicable, β is either always acute or always obtuse. We note that α and β cannot be both obtuse because the sum of the inner angles of a triangle cannot exceed π. We now consider the following cases:

1. α is always obtuse. In this case, β must be always acute. So, by Lemma 6.3, the lower bound given by SEC-B $\geq \left\|\overrightarrow{yp}\right\| - ED(o_i, y)$. (Equality holds if and only if the line segment yp intersects the MBR, which implies that the ray emerging from y in the direction of $\overrightarrow{yp}$ intersects the MBR.) Since α is obtuse, $\left\|\overrightarrow{yp}\right\| \geq \left\|\overrightarrow{xy}\right\|$ (for all $x \in \text{MBR}(o_i)$) and hence $\left\|\overrightarrow{yp}\right\| \geq ED(o_i, y)$. Therefore, the lower bound given by triangle inequality (5) is $\left\|\overrightarrow{yp}\right\| - ED(o_i, y)$. Thus, the lower bound given by SEC-B $\geq$ the lower bound given by the triangle inequality.
2. α is always acute.
   (a) β is always obtuse. Since β is obtuse, we know that $\left\|\overrightarrow{xy}\right\| \geq \left\|\overrightarrow{yp}\right\|$ (for all $x \in \text{MBR}(o_i)$). Hence, $ED(o_i, y) \geq \left\|\overrightarrow{yp}\right\|$. So, the triangle inequality (5) gives $ED(o_i, y) - \left\|\overrightarrow{yp}\right\|$ as the lower bound. But since α is always acute, we know from Lemma 6.2 that the lower

bound given by SEC-A $\geq ED(o_i, y) - \left\|\overrightarrow{yp}\right\|$ = the lower bound given by the triangle inequality. (Equality holds if and only if the (straight) ray emerging from p in the direction of $\overrightarrow{yp}$ intersects the MBR, which implies that the ray emerging from y in the direction of $\overrightarrow{yp}$ intersects the MBR.)

(b) β is always acute. In this case, we do not know which of $\left\|\overrightarrow{yp}\right\|$ and $ED(o_i, y)$ is larger. However, we know that the lower bound given by the triangle inequality is equal to either $\left\|\overrightarrow{yp}\right\| - ED(o_i, y)$ or $ED(o_i, y) - \left\|\overrightarrow{yp}\right\|$ (whichever is greater). Now, using Lemma 6.2, we know that the lower bound given by SEC-A is no less than $ED(o_i, y) - \left\|\overrightarrow{yp}\right\|$. In addition, using Lemma 6.3, we know that the lower bound given by SEC-B is no less than $\left\|\overrightarrow{yp_j}\right\| - ED(o_i, y)$. So, either one of the lower bounds given by SEC-A or SEC-B is no less than the bound given by the triangle inequality. (Equality holds $\Rightarrow$ the line segment yp intersects the MBR or the (straight) ray emerging from p in the direction of $\overrightarrow{yp}$ intersects the MBR $\Rightarrow$ the ray emerging from y in the direction of $\overrightarrow{yp}$ intersects the MBR.)

□

To summarize, given the above theorems, we know that when COS is applicable, the trigonometric upper bound is no worse than the upper bound given by the triangle inequality. In addition, when CSC is applicable, the upper bound given by COS is strictly better than that of the triangle inequality. This is because the applicability of CSC guarantees that the equality condition mentioned in Theorem 6.1 does not hold. On the other hand, when SEC-A and SEC-B are both applicable, the trigonometric lower bound is no worse than the lower bound given by the triangle inequality. The former is strictly better than the latter when all SEC-A, SEC-B and CSC are applicable. Therefore, trigonometric bounds are favorable when the various bounds are applicable. When all of COS, SEC-A, SEC-B, CSC are applicable, the trigonometric bounds are strictly tighter than the metric bounds, thanks to Theorems 6.1 and 6.4. (This is because equality in these theorems holds only if CSC is inapplicable.) To maximize the likelihood that these bounds are applicable, we choose (when possible) an anchor point that is outside the MBR.

Before we end this section, we remark that the CSC-A bounds are in fact not useful, and they were listed just for the sake of completeness. We summarize this finding by the following theorem.

Theorem 6.5. *The CSC-A bounds are no tighter than the bounds determined using the MBR method (see Section 4).*

Proof. We show the proof for the upper bound only. The proof for the lower bound is similar and hence omitted for brevity.

Consider the point $z \in$ the MBR that maximizes $\left\|\overrightarrow{zp}\right\|$. The upper bound on $ED(o_i, p)$ given by the MBR method is $d(z, p) = \left\|\overrightarrow{zp}\right\|$. Let α_z, β_z and γ_z be the values of the angles α, β and γ when $x = z$, respectively. Then, by Identity (11), we have

$$\begin{aligned} \left\|\overrightarrow{zp}\right\| &= \left\|\overrightarrow{yp}\right\| \sin\gamma_z \csc\alpha_z \\ &\leq \left\|\overrightarrow{yp}\right\| \overline{\sin\gamma}\,\overline{\csc\alpha}. \end{aligned}$$

Thus, the upper bound given by the MBR method is tighter than or equal to that given by CSC-A. □

In our implementation of the trigonometric method, we do not compute the CSC-A bounds.

6.6. Summary of Trigonometric Bounds

The trigonometric bounds are obtained by combining the bounds on $ED(o_i, p_j)$ determined using trigonometry. As we have discussed, the trigonometric bounds are not always applicable. Their applicability depends on the relative positions of the MBR, the anchor point, and the cluster representative. The applicability conditions are summarized in Table 1. (The table shows only the upper bounds; lower bounds are obtained by swapping all quantities $\overline{q}$ with $\underline{q}$ in the formulae.) We only compute those bounds that are applicable, and take the tightest upper and lower bounds as the trigonometric bounds. In case none of the bounds are applicable, we fall back to those obtained by the triangle inequality.

In order to make the trigonometric bounds applicable, we need to pick an anchor point y that is outside the MBR, otherwise, Theorems 6.1 and 6.4 do not apply. Also, if y is far from the MBR, the ranges of the angles' values will be small. This leads to tighter bounds. Consequently, it is preferable to choose y that is far away from the MBR, as long as enough precision is maintained for numerical accuracy.

Table 1: Summary of Trigonometric Bounds

Name	Applicable when...	Upper Bound
COS	$p, y \notin$ MBR	$ED(o_i, y)\,\overline{\cos\alpha} + \left\|\overrightarrow{yp}\right\|\,\overline{\cos\beta}$
SEC-A	$B_{yp} \cap \text{MBR} = \emptyset$ (α: acute)	$\overline{\sec\alpha}\,(ED(o_i, y) - \left\|\overrightarrow{yp}\right\|\,\underline{\cos\gamma})$
SEC-A	MBR $\subseteq B_{yp}$ (α: obtuse)	$\underline{\sec\alpha}\,(ED(o_i, y) - \left\|\overrightarrow{yp}\right\|\,\overline{\cos\gamma})$
SEC-B	MBR and y lie on the same side of P_{py} (β: acute)	$\overline{\sec\beta}\,(\left\|\overrightarrow{yp}\right\| - ED(o_i, y)\,\underline{\cos\gamma})$
SEC-B	MBR and y lie on different sides of P_{py} (β: obtuse)	$\underline{\sec\beta}\,(\left\|\overrightarrow{yp}\right\| - ED(o_i, y)\,\overline{\cos\gamma})$
CSC-A	$L_{py} \cap MBR = \emptyset$	$\left\|\overrightarrow{yp}\right\|\,\overline{\sin\gamma}\,\overline{\csc\alpha}$
CSC-B	$L_{py} \cap MBR = \emptyset$	$ED(o_i, y)\,\overline{\sin\gamma}\,\overline{\csc\beta}$

7. Experiments

In this section we report our empirical study on the effectiveness of the various pruning methods.

7.1. Data

We use synthetic datasets and a geographic dataset in our experiments. The synthetic datasets, with controllable parameters, help us study the effectiveness of the pruning methods over a wide spectrum of situations. On the other hand, the geographic dataset allows us to study the practicality of the pruning methods under a real setting.

Synthetic datasets. We generate two types of synthetic datasets, one with objects exhibiting intrinsic cluster patterns, one without. For each type, a number of datasets are generated based on different parameter values.

Each dataset is generated by the following procedure. The object domain is a rectangular 2D region with dimension $[0, 100] \times [0, 100]$. n objects are generated. Each object is associated with an MBR with random side lengths no longer than d units. For datasets without cluster patterns, the objects' MBRs are randomly positioned in the object domain. For datasets with cluster patterns, k points in the domain are randomly chosen as the real cluster centers, with a constraint that these points are at least $\frac{100}{2\sqrt{k}}$ units apart from one another. This distance constraint ensures that the cluster centers are not too close and at the same time allows enough space for the random placements of the centers. Each object is then randomly assigned

to one of the k clusters. The position of an object is randomly fixed at a point that is no farther than $\frac{100}{2\sqrt{k}}$ units from its assigned cluster center. This procedure generates some natural cluster patterns [2].

For each object, we partition its MBR into $\sqrt{s} \times \sqrt{s}$ rectangular grid cells. Each cell's center is used as a sample point, thus each object has s sample points. The probability mass of each cell is first sampled from a uniform distribution independently of other cells, and then normalized by the sum. In our implementation of the UK-means algorithm, expected distances are calculated by summing over discrete sample points of the MBRs. It is therefore sufficient to represent an uncertain object by the probability masses taken at the sample points.

Geographic dataset. Following previous studies [20, 39], we modify a real dataset to mimic location uncertainty of mobile objects. The dataset contains 53,145 2D points that are geographic locations in Long Beach of California in the United States of America, originally extracted from the US census web site, `http://www.census.gov/geo/www/tiger/`.We normalize the locations to the $[0, 100] \times [0, 100]$ domain, and generate a $d \times d$ MBR centering at each location. A pdf is then generated for each MBR in the same way as with the synthetic datasets.

Table 2 summarizes the parameters used in the data generation process. We study the effect of each parameter by varying its value while keeping the other parameters at their default values. For those experiments that measure the number of expected distance calculation (N_{ED}, discussed below), we use a relatively small number of sample points ($s = 1024$) comparing with other studies on uncertain data management (e.g., [40]), in order to control the amount of time taken to complete the experiments. In a real setting where a larger number of sample points is used, the effectiveness of a pruning method would be even more important since the calculation of each expected distance would take longer.

7.2. Setup

We compare the performance of UK-means with various combinations of pruning methods:

- *Brute-force*: the basic UK-means algorithm without expected distance pruning
- *MBR*: UK-means with Min-max-dist pruning using MBRs (Section 4)

Table 2: Experimental parameters and their default values.

Parameter	Description	Default value
n	Number of objects	20,000
k	Number of clusters	49
d	(Maximum) side length of MBR	5
s	Number of sample points	102,400 (1024 for the study of N_{ED})

- *Met:*{*PC/CS/PC+CS*}: UK-means with Min-max-dist pruning using MBRs *and* the PC, CS, or both methods with expected distance bounds given by the triangle inequality (Section 5)
- *Tri:*{*PC/CS/PC+CS*}: UK-means with Min-max-dist pruning using MBRs *and* the PC, CS, or both methods with expected distance bounds determined by trigonometric rules (Section 6)

The primary performance metric is the average number of expected distance calculations per object per iteration, denoted as N_{ED}. It reflects the main performance bottleneck of the UK-means algorithm. Given k clusters, the brute-force implementation of UK-means computes all k expected distances between an object and the k cluster representatives in each iteration of the clustering process. Therefore, $N_{ED} = k$ under Brute-force, which serves as a baseline for comparing the performance of the pruning methods. The goal of the experiments is to study how many expected distance calculations can be saved by the various pruning methods. Whenever the PC method is involved, N_{ED} also includes the number of pre-computed expected distances between objects and their anchor points, and thus in theory could be larger than k if the pruning were completely ineffective.

In some experiments we also use the actual execution time as a secondary performance metric to evaluate the computational overheads of the calculation of the various bounds.

For each clustering process, the initial cluster representatives are picked uniformly from the 2D space. For each set of parameter values, 20 synthetic datasets are randomly generated, and the average results are reported.

For Met:PC, the one-point scheme described in Section 5.1 is used. For Tri:PC, an anchor point located at co-ordinates (100000,100000) is used.

Table 3: Performance of the pruning methods on the synthetic dataset without cluster patterns using default experimental parameter values.

Method	N_{ED}	Clustering time in seconds	Pdf I/O count
Brute-force	49.	>360,000 (100 hours)	1,500,000
MBR	0.6393	66,918	557,534
Met:PC	0.6389	73,675	569,570
Tri:PC	0.2213	20,426	210,982
Met:PC+CS	0.0711	7,414	71,692
Met:CS	0.0598	6,136	59,411
Tri:PC+CS	0.0315	4,089	32,804
Tri:CS	0.0299	3,521	29,124

Our programs are written in Java 1.5. The experiments were conducted on Windows machines with an Intel 3.2GHz Pentium 4 processor and 1GB of memory.

7.3. Results

7.3.1. Default parameter values

In the first set of experiments, we investigate the performance of the various pruning methods using the default parameter values. The results for the dataset without cluster patterns are shown in Table 3. The rows are sorted in descending order of N_{ED}. Note that the Brute-force method did not terminate after 100 hours of execution.

From the N_{ED} column of the table, we clearly see that the pruning methods are very effective. For example, N_{ED} for Brute-force, MBR, and Tri:CS are 49, 0.6393, and 0.0299, respectively. This shows that the MBR method prunes $1 - 0.6393/49 = 98.7\%$ of the expected distances computed by Brute-force. Impressively, Tri:CS prunes yet another $1 - 0.0299/0.6393 = 95.3\%$ of those computed by the MBR method. Notice that N_{ED} is less than one under any pruning method. This implies that the bounding techniques are so effective that in most cases, considering the bounds alone is sufficient to determine the closest cluster representative of an object without resorting to computing expected distances.

Moreover, methods using the trigonometric bounds (prefixed by Tri:) are about 2–3 times more effective than those using the triangle inequality only

(prefixed by Met:). This performance gain justifies the use of the seemingly more complex trigonometric bounds.

Comparing the PC methods and the CS method, we see that in general CS performs better than PC. One reason is that the anchor points used by CS (cluster representatives of previous iterations) are close to the current cluster representatives. This results in very tight bounds. One surprising observation is that combining PC and CS gives a poorer performance than using CS alone. This is due to the large overhead incurred in pre-computing expected distances for the PC method, which could not be compensated for by the savings brought by the method. On the contrary, no pre-computations are done in CS, since anchor points are previous cluster representatives whose expected distances to objects have been computed previously.

The last two columns in Table 3 show the actual execution time and the number of pdf read from disk, respectively. Both measures correlate fairly well with N_{ED}. It is thus reasonable to use N_{ED} as the primary performance metric.

Tables 4 and 5 show the results of the experiments performed on the synthetic dataset with cluster patterns and the ones performed on the geographic dataset, respectively. We observe similar relative performances among the pruning methods as compared to the case when the synthetic dataset without cluster patterns is used. The relative effectiveness of the pruning methods is thus not greatly affected by the data properties. For brevity and clarity, in the following experiments that involve synthetic data, we only report the results for the datasets without intrinsic cluster patterns.

The absolute pruning power appears to be stronger with the synthetic dataset. We believe this is due to the fact that objects are freely distributed in the whole 2D space in the synthetic dataset, but they are subject to geographic and architectural constraints in the geographic dataset. As a result, the clusters in the synthetic dataset are more spread out, which increases the ability to prune the cluster representatives that are not closest.

7.3.2. Scalability with respect to the number of objects

Next we compare the performance of the various methods as the number of objects n increases. Figure 8 shows the results as n increases from 1,000 to 50,000. Figure 8(a) shows the performance curves of all methods. Figure 8(b) shows a magnified version of Figure 8(a) focusing on the best four curves. In the figures, the color of a curve indicates the method used in choosing anchor points (i.e., PC/CS), while the marker shape indicates the bounding

Table 4: Performance of the pruning methods on the synthetic dataset with cluster patterns using default experimental parameter values.

Method	N_{ED}
Brute-force	49
MBR	0.6027
Met:PC	0.6034
Tri:PC	0.2140
Met:PC+CS	0.0878
Met:CS	0.0761
Tri:PC+CS	0.0354
Tri:CS	0.0351

Table 5: Performance of the pruning methods on the geographic dataset using default experimental parameter values (n fixed at original dataset size, 53,145).

Method	N_{ED}
Brute-force	49
MBR	1.4425
Met:PC	1.4222
Tri:PC	0.5963
Met:PC+CS	0.1254
Met:CS	0.1176
Tri:PC+CS	0.0583
Tri:CS	0.0673

strategy.

From Figure 8, we see that the relative rankings of the various methods remain largely the same over the range of n. Our previous observations, such as CS being more effective than PC and trigonometric bounds being better than those obtained from the triangle inequality remain true with different dataset sizes. Another observation is that the curves for CS-based methods (Met:PC+CS, Met:CS, Tri:PC+CS, Tri:CS) drop as n increases. This is because with more objects, the clustering process takes more iterations to terminate. It is thus more likely that an anchor point (which is a cluster representative in a previous iteration such that its expected distance from an object has been computed) can be found for effective pruning. The large overhead in pre-computations once again makes PC-based methods not as efficient as the CS-based method, and the overhead can be so large that Met:PC performs even worse than pure MBR-based pruning when n is small.

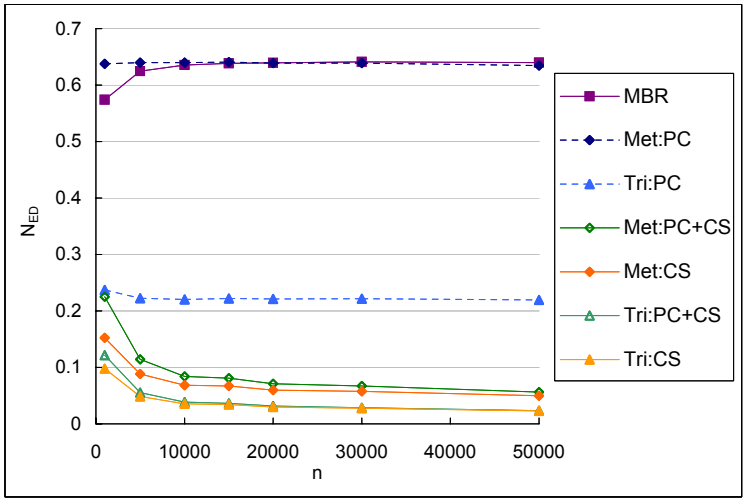
7.3.3. Scalability with respect to the number of clusters

Our next experiment studies the effect of the number of clusters, k. The results are shown in Figure 9. Note that Brute-force computes all k expected distances for each object during each iteration (i.e., no pruning). In order to better express the pruning effectiveness of the algorithms, we report N_{ED} as a percentage of k in Figure 9. For example, the point for MBR at $k = 9$ shows that MBR computes about 2.6% of the expected distances computed by Brute-force, or a pruning effectiveness of 97.4%.

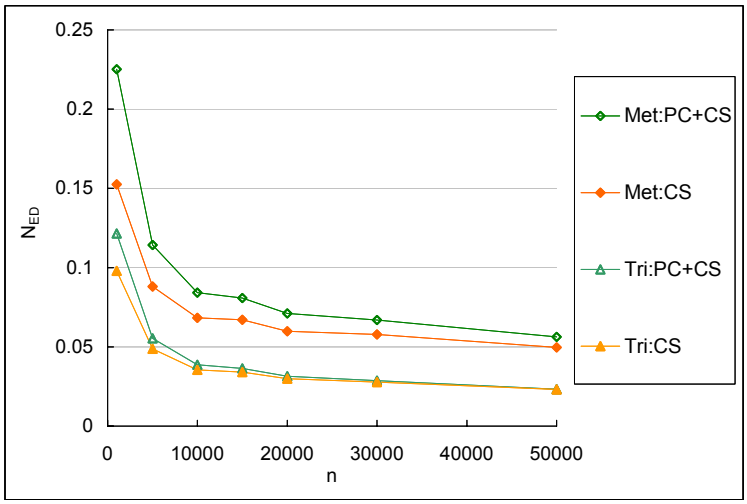
From Figure 9(a), we see that as k increases all curves go down. This is because with more cluster representatives, there is more opportunity for pruning and thus the algorithms can achieve a higher pruning effectiveness. Also, CS-based methods (lower 4-curves) are a lot more effective than the other methods. From Figure 9(b), we see that generally trigonometric pruning (Tri) outperforms triangle-inequality-based methods (Met) by a wide margin. For example, when $k = 25$, Tri-based methods prune at least twice as many expected distances than their Met-based counterparts.

7.3.4. Scalability with respect to the side length of MBRs

Next, we vary the maximum side length of MBRs, d. A larger d gives a larger uncertainty region of an object. The experiment thus studies how the degree of uncertainty affects the algorithms' performance. Figure 10 shows N_{ED} as d increases from 1 to 15 units.

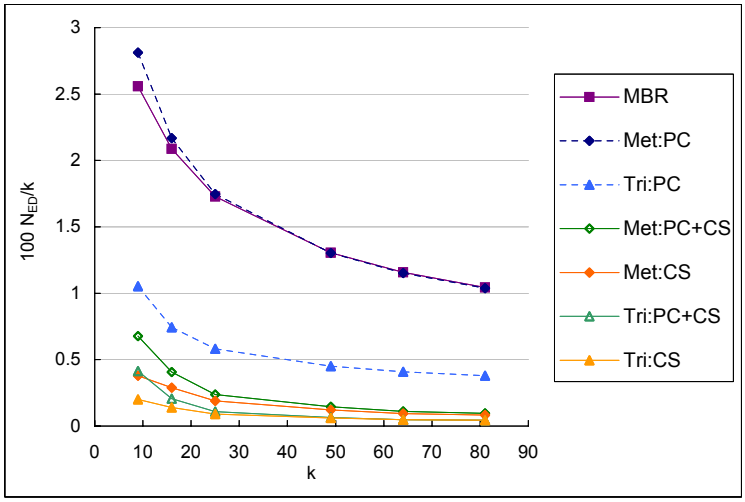


(a)

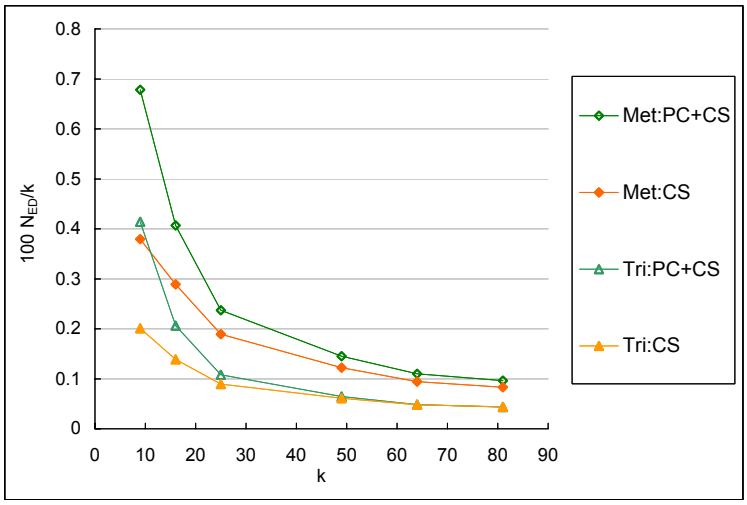


(b)

Figure 8: Performance of the various pruning methods on the synthetic dataset without cluster patterns as n increased.

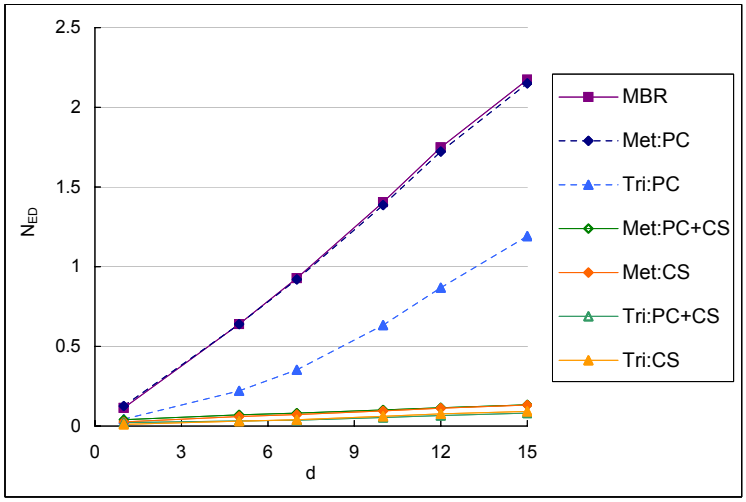


(a)

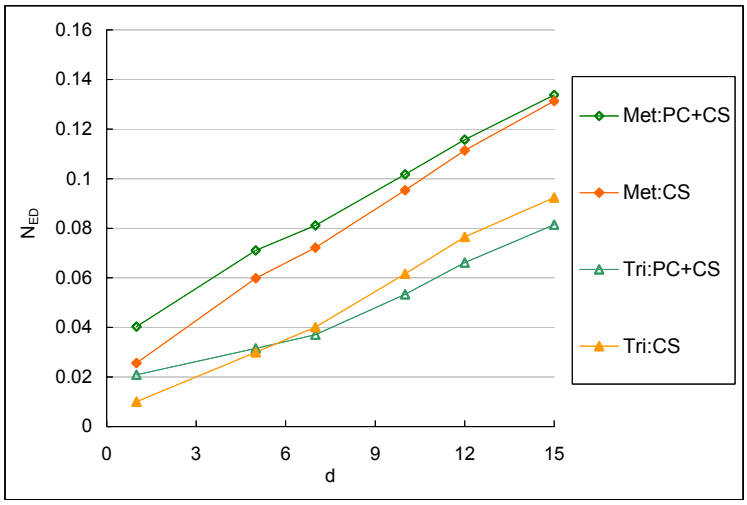


(b)

Figure 9: Performance of the various pruning methods on the synthetic dataset without cluster patterns as k increased.



(a)



(b)

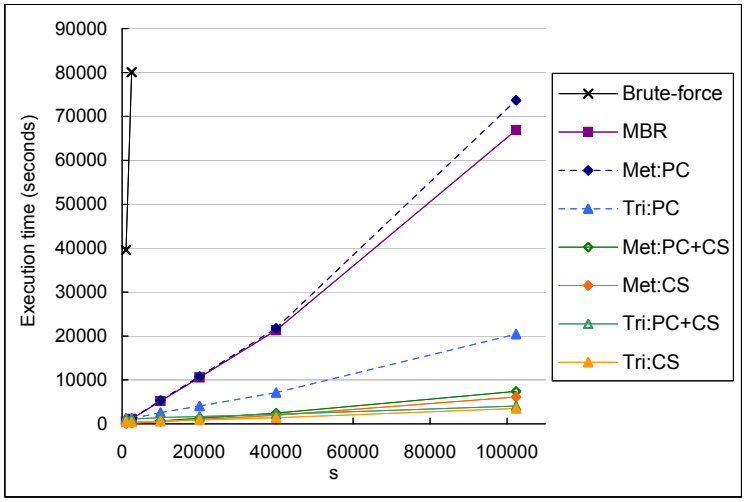
Figure 10: Performance of the various pruning methods on the synthetic dataset without cluster patterns as d increased.

From the figure, we see that all the curves go up with d. This is because a larger d implies that objects have larger uncertainty regions, so that the bounds given by the various methods are looser, and it results in less effective pruning. From Figure 10(b), we observe an interesting trend comparing the relative performance of CS methods against PC+CS methods. We see that when d is small (e.g., when $d = 1$), CS methods outperform their PC+CS counterparts by a wide margin. This is because with small uncertainty, the bounding techniques are very effective and thus the pre-computation overhead incurred by PC makes PC counter-productive. However, as d increases, bounds become looser and pruning becomes less effective. In this situation, combining the bounds obtained from PC and CS allows for a better pruning opportunity. This additional pruning outweighs the pre-computation overhead. The performance of PC+CS thus catches up that of CS. When d is large, we see that Tri:PC+CS even outperforms Tri:CS.

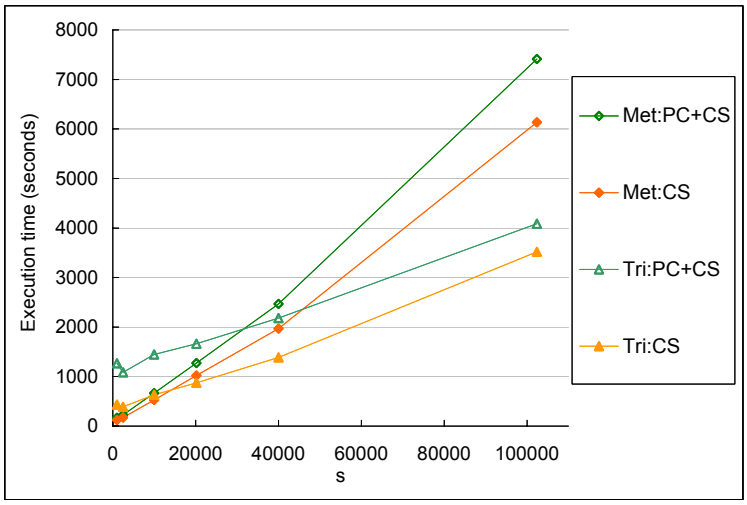
7.3.5. Scalability with respect to the number of sample points from each pdf

Finally, we study the effect of the number of sample points s used in representing a pdf. Basically, the more sample points used (i.e., a larger s), the more expensive it is to compute an expected distance (which involves an integration over the set of sample points). The value of s thus directly affects the execution time of the algorithms. On the other hand, the number of expected distance calculations (N_{ED}) depends mostly on the geometry and locations of objects' uncertainty regions, not so much on s. We therefore report execution times instead of N_{ED} in this experiment. We also report the execution times of Brute-force at small s values for comparison. Figure 11 shows the results.

As expected, the running time increases as s increases for all methods. The brute-force method quickly becomes intractable when s is only moderately large. MBR and Met:PC are again the worst pruning methods. An interesting observation from Figure 11 is that when s is very small, the methods based on the triangle inequality performs better than those based on trigonometric bounds. This is because when s is very small, the cost of computing expected distances is less dominating. Even though Trigonometric bounds (Tri) give a higher pruning effectiveness, the overheads in computing angle bounds become significant relative to the cost of expected distance calculation. Hence, for small s, trigonometric pruning (Tri) is less efficient than those based on the triangle inequality (Met).



(a)



(b)

Figure 11: Performance of the various pruning methods on the synthetic dataset without cluster patterns as s increased.

7.3.6. Scalability results on the geographic dataset

Figures 12, 13 and 14 show the performance of the various pruning methods when applied to the geographic dataset as we vary k, d, and s, respectively. Except being not as smooth, the curves show similar trends and relative performance among the algorithms as those obtained from the synthetic dataset experiments.

7.4. Summary

Based on these experimental results, we have observed the following general trends.

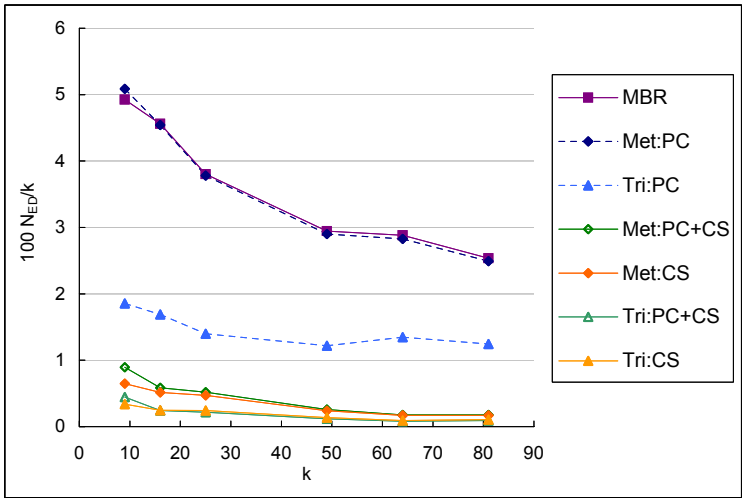
- Tri:CS gives the best pruning effectiveness, shortest execution time and lowest I/O costs over a wide range of parameters.
- However, when the uncertainty region (d) is relatively large (e.g. with side length greater than 5% of the side length of the object domain), Tri:PC+CS may out-perform Tri:CS.
- When the number of pdf sample points (s) is small, Met:CS delivers better performance.
- The PC-based pruning involves non-negligible overhead, which may fail to compensate for the number of ED computations saved from the pruning, especially when k and n are small and s is large.

8. Discussions

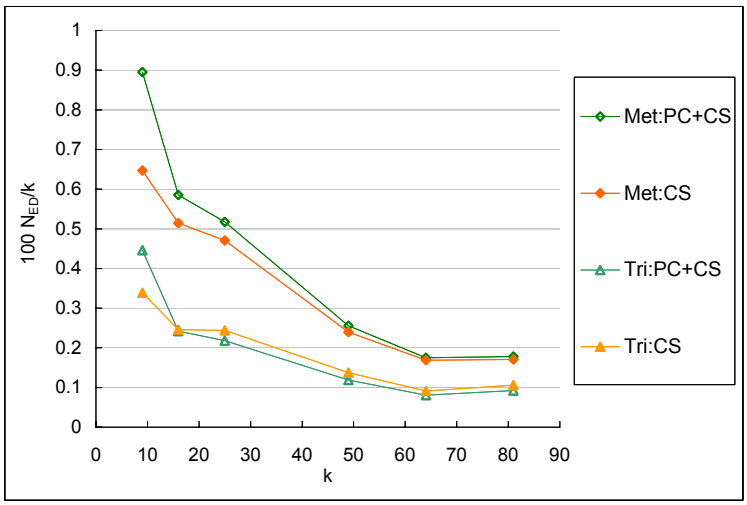
In this section, we briefly discuss a few issues related to our pruning algorithms.

8.1. Interactions between different pruning methods

From the experimental results, we see that CS-based methods perform very well. Also, adding PC to CS (i.e., PC+CS) is generally counter-productive. This is because of the pre-computation overhead incurred by the PC methods, which outweighs the extra pruning power it offers. On a closer examination, we found that if we excluded the pre-computation cost from N_{ED}, PC+CS did perform substantially better than CS (data not shown). Thus, PC can offer some extra mileage in terms of improving pruning effectiveness. It is

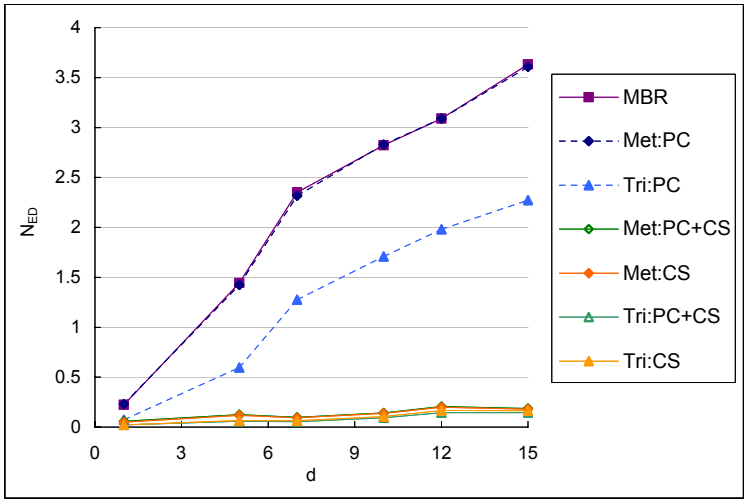


(a)

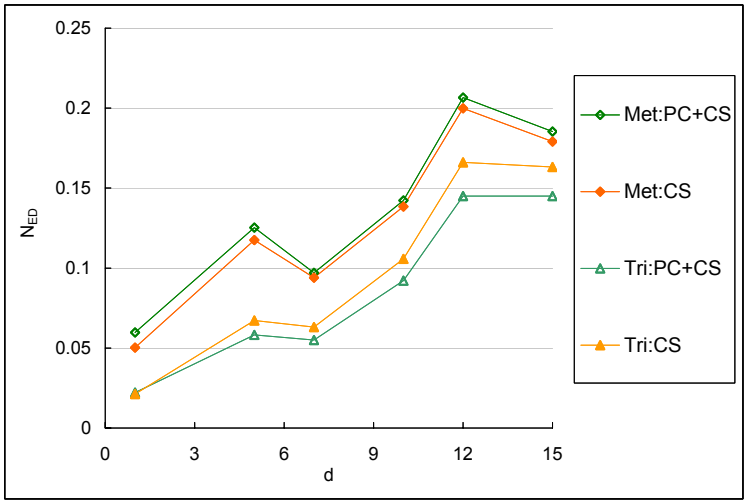


(b)

Figure 12: Performance of the various pruning methods on the geographic dataset as k increased.

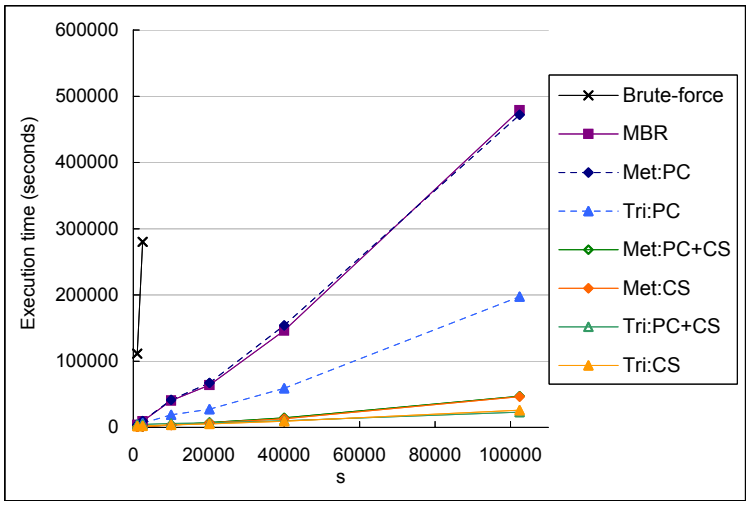


(a)

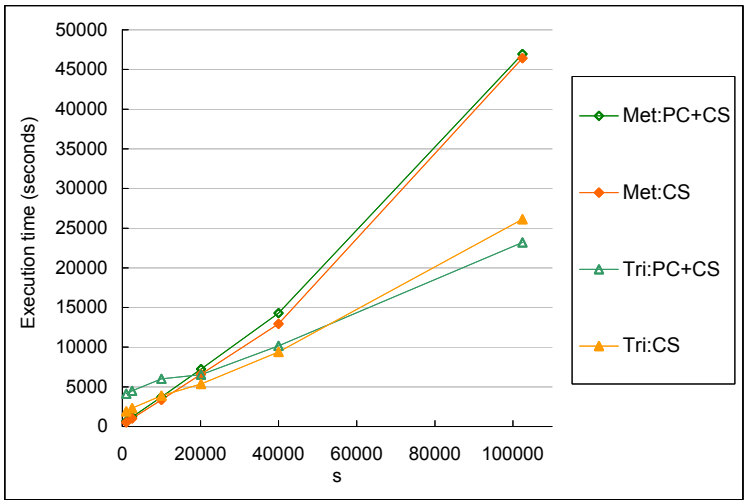


(b)

Figure 13: Performance of the various pruning methods on the geographic dataset as d increased.



(a)



(b)

Figure 14: Performance of the various pruning methods on the geographic dataset as s increased.

thus interesting to find anchor points that provide distance bounds similar to those given by PC, but without pre-computations.

One idea is again to use previous cluster representatives. Given an object o_i and a cluster representative p'_j, we compute the expected distance $ED(o_i, p'_j)$ only if p'_j cannot be pruned. This implies that p'_j is likely to be close to object o_i (so that the lower bound of $ED(o_i, p'_j)$ does not exceed the min-max-dist threshold). As a result, p'_j should serve as a good anchor point for o_i since (1) $ED(o_i, p'_j)$ is available and (2) p'_j is close to o_i. We can thus consider using p'_j as an anchor point for o_i in place of those selected before the clustering process starts (see Section 5.1). Notice that under the CS method, p'_j will also be selected as an anchor point. The difference is that under CS, p'_j will only be used as an anchor point to bound the distance from o_i to a future representative p_j of cluster j, while under PC, p_j is used as a *general* anchor point of o_i and can be used to bound the distance from o_i to *any* future cluster representatives.

We can thus consider a revised PC method: instead of using fixed anchor points, we maintain for each object o_i a list of dynamic anchor points. This list includes all those p'_j whose expected distances from o_i have been computed. The list can be kept small by retaining only the closest points located at different directions of the object (recall the discussion on the selection of anchor points in Section 5.1). The resulting method is expected to have a pruning power similar to the PC method and still be complementary to the CS method, and with the advantage of no pre-computation overhead.

8.2. Comparison with Vornoi-diagram-based pruning

Given a set of cluster centers c_j ($j = 1, \ldots, k$), the Voronoi diagram of them is a partition of the 2D space into k convex cells. Inside each cell V_j, every point is closer to corresponding c_j than any other c_x ($x \neq j$). This nice property has been used to facilitate k-means clustering in the cluster assignment step [41]. In a parallel study, we have extended this technique for use in UK-means [18]. An important theoretical result is that bisector pruning (BP) based on Voronoi diagrams has a pruning power no less than that of Min-max-dist pruning using MBRs (see Section 4), i.e., whenever BP cannot eliminate a cluster candidate, neither can Min-max-dist pruning using MBR. An overhead is incurred, though: the computation of the Voronoi diagram, which can be computed in $O(k \log k)$ time for 2D.

On the other hand, Cluster-Shift pruning (CS, see Section 5) may be able to prune some cluster candidates that BP fails to prune. Here is an

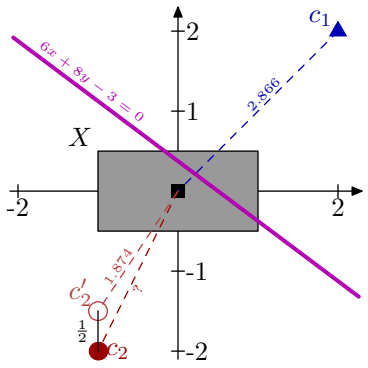


Figure 15: Cluster Shift vs. Bisector pruning

example. Suppose that we have an uncertain object X with an MBR that is 2 units wide and 1 unit tall, centered at the origin (see Figure 15). For simplicity, let X be uniformly distributed within its MBR. Suppose we have only 2 clusters: c_1 at $(2, 2)$ and c_2 at $(-1, -2)$. To which cluster should we assign X? The Voronoi diagram for c_1 and c_2 divides the 2D space into 2 halves, separated by the line $6x + 8y - 3 = 0$, which bisects the line segment joining c_1 and c_2 perpendicularly. This is shown as the thick slanted line in the figure. Since this line intersects the MBR of X, Voronoi-diagram-based pruning is not applicable [18]. Can CS help in this case? Suppose that in the previous iteration, c_2 has been updated, with the old position being c_2' at $(-1, -1.5)$, and that c_1 has remained the same. Now, using the CS technique, we get an upper bound for $ED(X, c_2)$: $MaxDist_{X,2} = ED(X, c_2') + \|c_2 - c_2'\|$. Since $ED(X, c_2') = 1.874$ has been computed and saved in the previous iteration, we know its value without extra cost. Therefore, we get the upper bound $MaxDist_{X,2} = 1.874 + 1/2 = 2.374$. Since c_1 has not changed from the last iteration, we already know that $ED(X, c_1) = 2.866$. Therefore, without computing $ED(X, c_2)$ (which is 2.299), the Cluster-Shift technique can already conclude that $ED(X, c_2) \leq MaxDist_{X,2} < ED(X, c_1)$ and exclude candidate c_1 from consideration, which Voronoi-diagram-based pruning cannot eliminate. A combination of both pruning techniques turns out to be very effective [18].

8.3. Indexing MBRs

In this paper an MBR is used to enclose the uncertainty region of an uncertain object. The purpose of the MBR is to help us tightly bound the

expected distance of an object and a cluster representative. The concept of MBR is also used extensively in spatial indexing structures such as R-tree, in which the MBRs are indexed in a recursive hierarchical structure. We can use the concept of this hierarchical organization to reduce the overheads of pruning rules. More specifically, given a set of nearby uncertain objects S, we can obtain the MBR (M) that encloses the MBRs of all objects in S. If we consider M as the MBR of a super object, we can apply our pruning techniques to M. We remark that although M gives us very loose bounds of the ED values of the objects in S, any pruning achieved applies to *all* the objects in S. This potentially reduced the pruning overheads compared against the case in which the pruning rules are applied repeatedly to each individual object.

8.4. Picking an anchor point for the PC method

In our discussion, the center point of an uncertain object's MBR is used as the anchor point under the PC method. The advantage of this simple approach is that the center of an MBR can be conveniently located. However, if one has pre-computed the center of mass (CG) of each uncertain object (which costs an extra integration per object), then the CG can also be picked as the anchor. This often results in better performance for the PC method. We have conducted an experiment comparing two choices of the anchor points: MBR center and CG. Figure 16 shows N_{ed} for Met:PC under the two choices of anchor points. We see that the CG performs better than the MBR center. For example, when $d = 12$, using CG's requires about 5 times fewer EDs than using the MBR centers. We have also evaluated the performance of the PC method under other choices of anchor points. For example, we have tried a few randomized methods, such as picking a random corner of an object's MBR, and picking a random point within an object's MBR. Our results show that the CG anchor points give better performance than all those other choices. This is because the CG is generally closer to the samples of an object's pdf than other choices of the anchor point. Therefore, the CG generally gives a better estimate of distances via the triangle inequality.

9. Conclusion

In this article we have described the problem of efficient clustering of uncertain data, which finds applications in areas where the exact values of

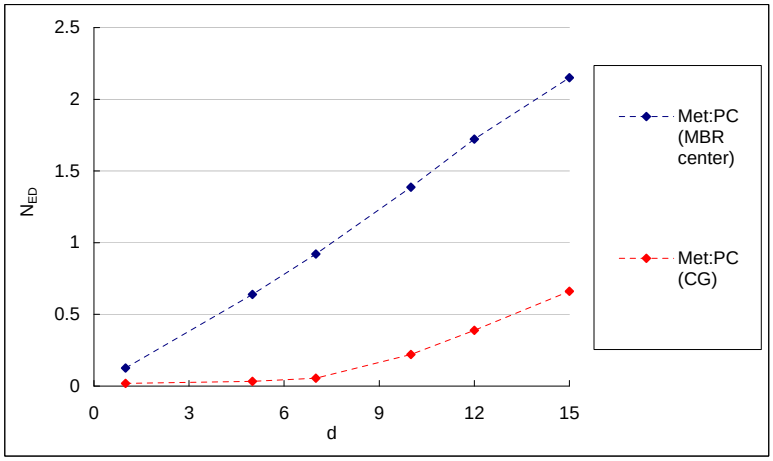


Figure 16: Performance of Met:PC under two choices of anchor points.

some data attributes are not certain, but can be modeled by a probability density function (pdf). We have described the basic UK-means algorithm for uncertain data clustering. We have shown that when objects' pdf's are not confined to a parametric family but can take any arbitrary form, the cost of expected distance calculations is the performance bottleneck of UK-means. We have introduced the basic Min-max-dist pruning framework and have devised a number of pruning algorithms based on the framework. These pruning algorithms include the MBR method, the Pre-Computation (PC) method, and the Cluster-Shift (CS) method. We have studied two bounding techniques: Met bounds are based on the triangle inequality and Tri bounds are based on a number of trigonometric rules. We have studied the theoretical advantages and disadvantages of the different trigonometric bounds, and have proved that under certain easily attainable conditions, trigonometric bounds are strictly better than metric bounds.

In the experiment section, we have illustrated the importance of the pruning methods by comparing the computation cost of UK-means with and without expected distance pruning. We have also compared the performance of the different pruning methods. In general, we have found that the methods that involve trigonometric bounds are superior to those that involve the metric bounds, which are in turn superior to the basic MBR method. In addition, the CS method has been found to perform better than the PC method due to the pre-computation overhead of the latter. Overall, the best

pruning method could have a computational cost (in terms of the number of expected distance calculations) four orders of magnitude lower than the brute-force implementation.

Acknowledgment

We would like to thank Mr. Chun-Kit Chui and Dr. Wai-Shing Ho for helpful discussions.

References

[1] M. Chau, R. Cheng, B. Kao, J. Ng, Uncertain data mining: An example in clustering location data, in: Pacific-Asia Conference on Knowledge Discovery and Data Mining, 2006.

[2] W. K. Ngai, B. Kao, C. K. Chui, R. Cheng, M. Chau, K. Y. Yip, Efficient clustering of uncertain data, in: Proceedings of the Sixth International Conference on Data Mining, 2006, pp. 436–445.

[3] J. Chen, R. Cheng, Efficient evaluation of imprecise location-dependent queries, in: Proc. ICDE, 2007.

[4] R. Cheng, Y. Zhang, E. Bertino, S. Prabhakar, Preserving user location privacy in mobile data management infrastructures, in: Proc. of the 6th Workshop on Privacy Enhancing Technologies, 2006.

[5] M. F. Mokbel, C.-Y. Chow, W. G. Aref, The new casper: Query processing for location services without compromising privacy, in: VLDB, 2006.

[6] P. Misra, P. Enge, Global Positioning System: Signals, Measurements and Performance (2nd Edition), Ganga-Jumuna Press, 2006.

[7] W. R. Heinzelman, A. Chandrakasan, H. Balakrishnan, Energy-efficient communication protocol for wireless microsensor networks, in: Proceedings of IEEE International Conference on System Sciences, 2000.

[8] S. Bandyopadhyay, E. J. Coyle, An energy efficient hierarchical clustering algorithm for wireless sensor networks, in: Proceedings of IEEE INFOCOM, 2003.

[9] O. Wolfson, H. Yin, Accuracy and resource concumption in tracking and location prediction, in: Advances in Spatial and Temporal Databases, 8th International Symposium, SSTD 2003, Vol. 2750 of Lecture Notes in Computer Science, Springer, Santorini Island, Greece, 2003, pp. 325–343.

[10] D. Pfoser, C. S. Jensen, Capturing the uncertainty of moving-object representations, in: Proceedings of the 6th International Symposium on Advances in Spatial Databases, 1999, pp. 111–132.

[11] O. Wolfson, A. P. Sistla, S. Chamberlain, Y. Yesha, Updating and querying databases that track mobile units, Distributed and Parallel Databases 7 (3) (1999) 258–287.

[12] N. Dalvi, D. Suciu, Efficient query evaluation on probabilistic databases, in: Proceedings of the 30th International Conference on Very Large Data Bases, 2004, pp. 864–875.

[13] D. Barbara, H. Garcia-Molina, D. Porter, The management of probabilistic data, IEEE Transactions on Knowledge and Data Engineering 4 (5) (1992) 487–502.

[14] R. Cheng, D. V. Kalashnikov, S. Prabhakar, Evaluating probabilistic queries over imprecise data, in: Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data, 2003.

[15] R. Cheng, D. V. Kalashnikov, S. Prabhakar, Querying imprecise data in moving object environments, IEEE Transactions on Knowledge and Data Engineering 16 (9) (2004) 1112–1127.

[16] J. Chen, R. Cheng, Efficient evaluation of imprecise location-dependent queries, in: ICDE, IEEE, Istanbul, Turkey, 2007, pp. 586–595.

[17] S. D. Lee, B. Kao, R. Cheng, Reducing UK-means to K-means, in: The 1st Workshop on Data Mining of Uncertain Data (DUNE), in conjunction with the 7th IEEE International Conference on Data Mining (ICDM), Omaha, NE, USA, 2007.

[18] B. Kao, S. D. Lee, D. W. Cheung, W.-S. Ho, K. F. Chan, Clustering uncertain data using Voronoi diagrams, in: Proceedings of the 8th IEEE

International Conference on Data Mining (ICDM 2008), 2008, pp. 333–342.

[19] T. Emrich, H.-P. Kriegel, P. Kroger, M. Renz, A. Zufle, Boosting spatial pruning: On optimal pruning of mbrs, in: Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data, 2010, pp. 39–50.

[20] H.-P. Kriegel, M. Pfeifle, Density-based clustering of uncertain data, in: Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, 2005.

[21] C. K. Chui, B. Kao, E. Hung, Mining frequent itemsets from uncertain data, in: Advances in Knowledge Discovery and Data Mining, 11th Pacific-Asia Conference, (PAKDD) Proceedings, Vol. 4426 of Lecture Notes in Computer Science, Springer, Nanjing, China, 2007, pp. 47–58.

[22] C. C. Aggarwal, On density based transforms for uncertain data mining, in: Proceedings of the IEEE International Conference on Data Engineering, 2007.

[23] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, in: Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining, 1996.

[24] M. Ankerst, M. M. Breunig, H.-P. Kriegel, J. Sander, OPTICS: Ordering points to identify the clustering structure, in: Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data, 1999.

[25] H.-P. Kriegel, M. Pfeifle, Hierarchical density-based clustering of uncertain data, in: Proceedings of the Fifth IEEE International Conference on Data Mining, 2005.

[26] R. Agrawal, J. Gehrke, D. Gunopulos, P. Raghavan, Automatic subspace clustering of high dimensional data for data mining applications, in: Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data, 1998.

[27] Z. Yu, H.-S. Wong, Mining uncertain data in low-dimensional subspace, in: Proceedings of the IEEE International Conference on Pattern Recognition, 2006.

[28] A. Gionis, A. Hinneburg, S. Papadimitriou, P. Tsaparas, Dimension induced clustering, in: Proceedings of International Conference on Knowledge Discovery and Data Mining, 2005.

[29] M. Ichino, H. Yaguchi, Generalized Minkowski metrics for mixed feature-type data analysis, IEEE Transactions on Systems, Man and Cybernetics 24 (4) (1994) 698–708.

[30] R. M. C. R. de Souza, F. de A. T. de Carvalho, Clustering of interval data based on city-block distances, Pattern Recognition Letters 25 (2004) 353–365.

[31] E. H. Ruspini, A new approach to clustering, Information Control 15 (1) (1969) 22–32.

[32] J. C. Dunn, A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters, Journal of Cybernetics 3 (1973) 32–57.

[33] J. C. Bezdek, Pattern Recognition with Fuzzy Objective Function Algorithms, Plenum Press, New York, 1981.

[34] M. Sato, Y. Sato, L. C. Jain, Fuzzy Clustering Models and Applications, Physica-Verlag, Heidelberg, 1997.

[35] M. Tabakov, A fuzzy clustering technique for medical image segmentation, in: International Symposium on Evolving Fuzzy Systems, 2006.

[36] J. B. MacQueen, Some methods for classification and analysis of multivariate observations, in: Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability, 1967.

[37] M. Nanni, Speeding-up hierarchical agglomerative clustering in presence of expensive metrics, in: Pacific-Asia Conference on Knowledge Discovery and Data Mining, 2005, pp. 378–387.

[38] C. Elkan, Using the triangle inequality to accelerate k-means, in: Proceedings of the Twentieth International Conference on Machine Learning, 2003.

[39] H.-P. Kriegel, P. Kunath, M. Pfeifle, M. Renz, Probabilistic similarity join on uncertain data, in: Proceedings of the 11th International Conference on Database Systems for Advanced Applications (DASFAA 2006), 2006, pp. 295–309.

[40] Y. Tao, R. Cheng, X. Xiao, W. K. Ngai, B. Kao, S. Prabhakar, Indexing multi-dimensional uncertain data with arbitrary probability density functions, in: Proceedings of the 31st International Conference on Very Large Data Bases, 2005.

[41] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, A. Y. Wu, An efficient k-means clustering algorithm: Analysis and implementation, IEEE Trans. Pattern Anal. Mach. Intell. 24 (7) (2002) 881–892.

A. Finding Bounds on Angles

To find the trigonometric bounds introduced in Section 6, it is necessary to find bounds on several trigonometric functions of the angles α, β and γ (see Figure 5) as the point x varies within the MBR of the uncertain object in question. There are many possible approaches to finding bounds on the trigonometric functions. In this appendix, we provide techniques to find the bounds on the angles α, β and γ. Once the bounds on these angles are found, we can find out the bounds on the trigonometric functions using the methods described in Appendix B. We will be reusing the notations defined in Section 6. It is particularly helpful to revise that section and refer to Figure 5.

The following discussion will be based on a special instance of this problem: 2-dimensional spaces, because it is easier to illustrate the ideas with 2D drawings. This is also what we have implemented in the programs used for the experiments in Section 7. Via coordinate geometry, we have proved that these techniques can be generalized to 3 dimensions and higher. The ideas of α-contours, β-contours and γ-contours are analogous to the 2D case. The proofs are lengthy and hence not included here.

A.1. Bounds on α

To determine the bounds on α, we first introduce the concept of α-contours. Consider any point x in two dimensional space distinct from y and p. The angle $\angle pxy$ is α. (See Figure 17.) If we draw a circumcircle of $\triangle pxy$, we can obtain the arc $\widehat{pxy}$. From elementary geometry, we know that since yp is a chord of this circle, it subtends the same angle α on the arc $\widehat{pxy}$. By symmetry, on the mirror image of this arc reflected about the line yp, yp subtends the same angle α, too. (e.g. see point x' in Figure 17.) Indeed, this arc (excluding the points p and y) and its mirror image is the set of all points that make the same angle α with y and p. Let us denote the mid-point of yp as m. We draw the perpendicular bisector of yp starting from point m and extending in one direction. Then, this line overlaps with the diameter of the circumcircle. Denote the intersection point of this perpendicular bisector and the arc as a. Then, $\angle pay$ is also α.

Now, examine another point a_0 on the perpendicular bisector such that $\left\|\overrightarrow{ma_0}\right\| < \left\|\overrightarrow{ma}\right\|$. Then, $\alpha_0 = \angle pa_0y > \alpha$. We can again draw a circumcircle of $\triangle pa_0y$ and obtain the arc $\widehat{pa_0y}$. Then, for any point x_0 on this arc (or x_0' on it mirror image reflected about yp), $\angle px_0y = \alpha_0$ $(= \angle px_0'y)$. This arc and

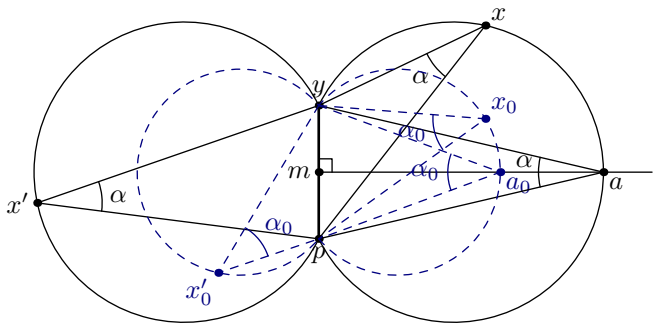


Figure 17: α-contours

its mirror image (excluding the points y and p) is the set of all points in that makes an angle of α_0 with y and p.

Indeed, by varying the distance of a from m (while maintaining a on the perpendicular bisector of yp and on one side of line yp), we can draw a set of contour lines for all a. Each contour line is an arc (excluding the end points y and p) plus its mirror image (reflected about line yp) for the same α. We will call these α-contours. Note that there are two limiting cases:

- $\left\|\overrightarrow{ma}\right\| \to 0$. In this case, the contour is the line segment yp (excluding points y and p) for $\alpha = \pi$.
- $\left\|\overrightarrow{ma}\right\| \to \infty$. The contour for this case is the (infinite) line through y and p excluding all the points at and between y and p. This is the contour for $\alpha = 0$.

It should be obvious that the α-contours do not intersect one another, and (including the two limiting cases) they cover the whole space (except the two singularities y and p). In addition, we know that α decreases as $\left\|\overrightarrow{ma}\right\|$ increases, because $\left\|\overrightarrow{ma}\right\| = \left\|\overrightarrow{my}\right\| \cot(\alpha/2)$ and $\alpha \in [0, \pi]$.

In 3D space, an α-contour is the surface of revolution of arc $\overset{\frown}{pay}$ about the axis yp. Generalisation to higher dimensions can be similarly derived.

A.1.1. Lower bound on α

With the concept of α-contours, it is easy to find $\underline{\alpha}$ as follows.

We first pick an arbitrary point $x \in$ MBR. The circumcircle of $\triangle pxy$ determines the arc $\overset{\frown}{pxy}$. (See Figure 18.) The α-contour for x thus consists of the arc $\overset{\frown}{pxy}$ (excluding points p and y) and its mirror image. This contour

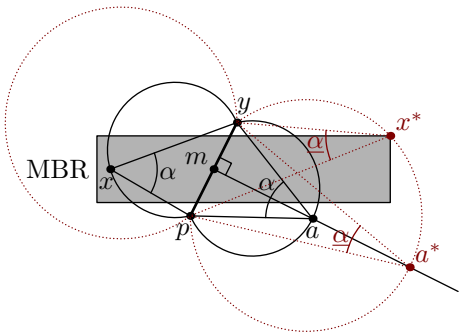


Figure 18: α-contour for $\underline{\alpha}$

intersects the bisector of yp at a. Note that all points in the intersection of the contour and the MBR make the same angle α with y and p. Now, by progressively increasing $\left\|\overrightarrow{ma}\right\|$, we get contours for smaller and smaller α. Points in the intersections of these contours and the MBR thus progressively make smaller and smaller angles with y and p. If any point on the line yp excluding the segment yp lies in the MBR, then, we can keep on increasing $\left\|\overrightarrow{ma}\right\|$ indefinitely and reach the limiting case $\left\|\overrightarrow{ma}\right\| \to \infty$. This corresponds to an α of zero, thus $\underline{\alpha} = 0$.

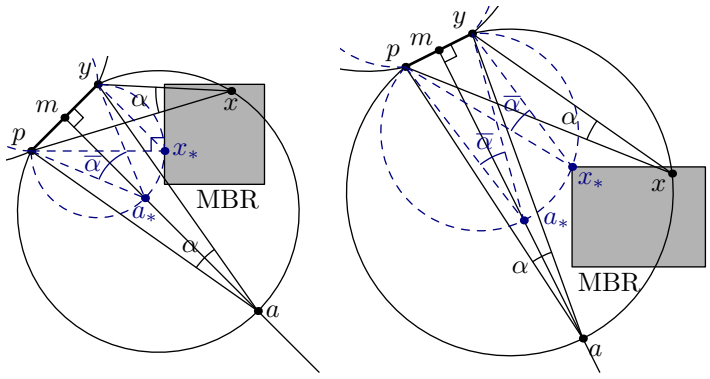
Otherwise, y and p must be outside the MBR. But since the MBR is bounded, as we increase $\left\|\overrightarrow{ma}\right\|$, we will eventually reach a maximum value $\left\|\overrightarrow{ma}\right\| = \left\|\overrightarrow{ma^*}\right\|$ so that the contour still intersects the MBR. This can only occur when the contour intersects the MBR at the corner points of the MBR. At these corners, α is minimized at value $\underline{\alpha}$.

Note that since all we want to find is $\underline{\alpha}$, there is no need to actually locate a^* or compute the arc $\overset{\frown}{pa^*y}$. We have shown that the minimum α can only be attained at the corners of the MBR. So, it suffices to check the values of α at the corners of the MBR.

Here is the procedure to find $\underline{\alpha}$. We first check whether y and p are both outside the MBR. If so, we only need to compute the values of α at all corners of the MBR, and take the minimum among them. This gives $\underline{\alpha}$. However, if either y or p is in the MBR, then, $\underline{\alpha} = 0$. Note that this argument applies to 3-dimensions and higher, too.

A.1.2. Upper bound on α

Similarly, $\overline{\alpha}$ can be found by starting at an arbitrary point in the MBR, and then progressively decreasing $\left\|\overrightarrow{ma}\right\|$ until we reach a minimum value



(a) Touching an edge (b) Intersecting an corner

Figure 19: α-contours for $\overline{\alpha}$

$\left\|\overrightarrow{ma}\right\| = \left\|\overrightarrow{ma_*}\right\|$. There are 3 possible cases, though.

Case 1: When line segment yp intersects the MBR, we will reach the limiting case where $\left\|\overrightarrow{ma_*}\right\| = 0$, which corresponds to $\alpha = \pi$. So, in this case $\overline{\alpha} = \pi$.

Case 2: The arc $\widehat{pa_*y}$ touches an edge of the MBR, as illustrated in Figure 19(a). Let the point of contact be x_*, the point at which the maximum angle $\overline{\alpha}$ is attained. This point x_* is not necessarily at a corner of the MBR. So, unlike the lower bound, we cannot simply check all corners and skip dealing with the α-contours. We need to find the point of contact x_* with an edge of the MBR. Once x_* is found, the corresponding α can be computed directly. Repeating this for all edges of the MBR, we can determine $\overline{\alpha}$.

Case 3: The arc $\widehat{pa_*y}$ intersects only corner points of the MBR. When trying to compute the point of contact of the contours with an edge of the MBR, solutions to the equations may give a point that lies outside the MBR, along the projection of the edge. This happens when the contour for $\overline{\alpha}$ intersects only corner points. (See Figure 19(b).) That corner point lies on the edge being considered. So, in this case, instead of examining the point of contact as given by the solution of the equations, we should check the end points of that edge of the MBR.

Therefore, to find $\overline{\alpha}$, we need to consider every edge of the MBR. For each edge (a line segment), we try to find the α-contour that touches the straight line containing the edge. If the point of contact lies within the edge, examine

the value of α at this contact point; otherwise, we examine the value of α at the end points of the edge. After all edges are considered, the maximum value of α among the examined ones gives the value of $\overline{\alpha}$.

A.2. Bounds on β

Similarly, to find $\underline{\beta}$ and $\overline{\beta}$, we use the concept of contours. In 2D, the β-contour is a ray (a straight line starting from a point and extends to infinity) that originates from p and makes an angle of β with $\overrightarrow{py}$. The point p is excluded. (See Figure 20(a).) Two such rays can be found and they are mirror images of each other, reflected about the line yp. In 3D, the β-contour is the surface of revolution of these rays about the axis yp. Such a surface has the shape of a cone, with the apex removed. The idea of β-contours can be similarly extended to higher dimensions.

Like the α-contours, we have two limiting cases:

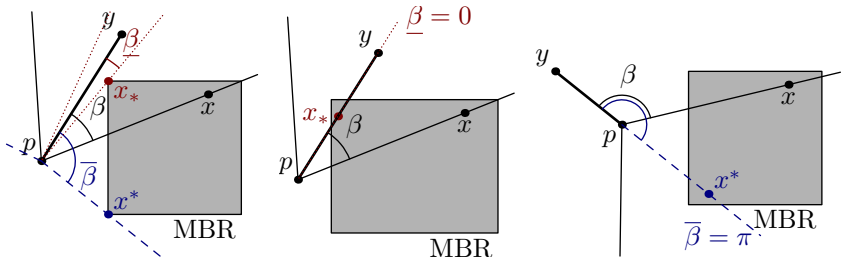
- $\beta \to 0$. In this case, the β-contour reduces to a single ray starting from p and extending in the direction of $\overrightarrow{py}$. Again, the point p is excluded.
- $\beta \to \pi$. The β-contour also reduces to a single ray in this case, starting from p, but extending in the direction of $\overrightarrow{yp}$ instead. The point p is excluded from this contour.

Since the β-contours are simply straight rays that radiate from the point p. The situation is much simpler. The maximum and minimum β-contours always intersect the MBR at the corners of the MBR, because the MBR is convex and the rays are straight. The only thing to care about is to check for the limiting cases.

A.2.1. Lower bound on β

If the ray starting from p extending in the direction of $\overrightarrow{py}$ intersects the MBR (see Figure 20(b)), then the minimum β is attained in that intersection, giving $\underline{\beta} = 0$.

Otherwise, we just need to find the least value of β such that the β-contour intersects MBR. This contour must intersect the MBR at its corners. So, we only need to enumerate through all corner points of the MBR, calculating the value of β at those points, and take the minimum value of them. This gives the value $\underline{\beta}$.



(a) General Case (b) $\underline{\beta} = 0$ (c) $\overline{\beta} = \pi$

Figure 20: β-contours

A.2.2. Upper bound on β

If the ray starting from p extending in the direction $\overrightarrow{yp}$ intersects the MBR (see Figure 20(c)), then the maximum β is attained in that intersection, giving $\overline{\beta} = \pi$.

Otherwise, we just need to find the greatest value of β such that the β-contour intersects MBR. This contour must intersect the MBR at its corners. So, we only need to go through all corner points of the MBR, and calculate the corresponding values of β. The maximum of them gives the value of $\overline{\beta}$.

A.3. Bounds on γ

These can be found in a way similar to the bounds on β as described above. We only need to exchange the roles of p and y and those of β and γ. All the arguments in Section A.2 apply.

B. Finding Bounds on Trigonometric Functions

To compute the trigonometric bounds (Section 6), we need to find the upper and lower bounds of the trigonometric functions sine, cosine, secant and cosecant for angles that vary within a certain interval. The interval of the angles can be determined using the method described in Appendix A. In this appendix, we assume that we are given an interval $[\underline{\theta}, \overline{\theta}] \subseteq [0, \pi]$. Our goal is to find the extreme values of trigonometric functions on any variable angle $\theta \in [\underline{\theta}, \overline{\theta}]$.

For sine, we note that this continuous function is increasing in $[0, \pi/2]$ and decreasing $[\pi/2, \pi]$, with a local maximum at $\pi/2$. So, to find its upper bound we first determine whether $\pi/2 \in [\underline{\theta}, \overline{\theta}]$. If so, then $\overline{\sin\theta} = 1$.

Otherwise, $[\underline{\theta}, \overline{\theta}]$ is either a subset of $[0, \pi/2]$ or $[\pi/2, \pi]$. The sine function is continuous without local extrema in these intervals. So, the maximum value is attained at the end points of the interval being considered. Thus, $\overline{\sin\theta} = \max(\sin\underline{\theta}, \sin\overline{\theta})$. For the lower bound, since sine has no local minimum in $[0, \pi]$, we have: $\underline{\sin\theta} = \min(\sin\underline{\theta}, \sin\overline{\theta})$.

Since cosine is a decreasing, continuous function in $[0, \pi]$, $\overline{\cos\theta} = \cos\underline{\theta}$ and $\underline{\cos\theta} = \cos\overline{\theta}$.

The secant function is undefined at $\pi/2$ and it is unbounded near $\pi/2$. So, when $\pi/2 \in [\underline{\theta}, \overline{\theta}]$, $\overline{\sec\theta} = +\infty$ and $\underline{\sec\theta} = -\infty$. Fortunately, the applicability conditions of SEC bounds (Section 6.3) have eliminated this possibility. So, we only need to handle the cases $[\underline{\theta}, \overline{\theta}] \subseteq [0, \pi/2)$ and $[\underline{\theta}, \overline{\theta}] \subseteq (\pi/2, \pi]$. In either case, $\sec\theta$ is a continuous, increasing function in the interval being considered. Therefore, $\overline{\sec\theta} = \sec\overline{\theta}$ and $\underline{\sec\theta} = \sec\underline{\theta}$.

The cosecant function is undefined at 0 and π. We exclude these points from consideration and only consider the situation where $[\underline{\theta}, \overline{\theta}] \subseteq (0, \pi)$. This is because the applicability conditions of the CSC bounds (Section 6.4) have eliminated the possibility of $\theta = 0$ or π. The cosecant function is continuous and decreasing within $(0, \pi/2]$ and increasing within $[\pi/2, \pi)$, with a local minimum at $\pi/2$. So, if $\pi/2 \in [\underline{\theta}, \overline{\theta}]$, then $\underline{\csc\theta} = 1$. Otherwise, $\csc\theta$ must attain its minimum value at an end point of the interval being considered. Hence, $\underline{\csc\theta} = \min(\csc\underline{\theta}, \csc\overline{\theta})$. Since $\csc\theta$ has no local maximum, we have: $\overline{\csc\theta} = \max(\csc\underline{\theta}, \csc\overline{\theta})$.

The following formulae summarize the results of the above discussions.

$$
\begin{aligned}
\overline{\sin\theta} &= \begin{cases} 1 & \text{if } \pi/2 \in [\underline{\theta}, \overline{\theta}] \\ \max(\sin\underline{\theta}, \sin\overline{\theta}) & \text{otherwise} \end{cases} \\
\underline{\sin\theta} &= \min(\sin\underline{\theta}, \sin\overline{\theta}) \\
\overline{\cos\theta} &= \cos\underline{\theta} \\
\underline{\cos\theta} &= \cos\overline{\theta} \\
\overline{\sec\theta} &= \begin{cases} +\infty & \text{if } \pi/2 \in [\underline{\theta}, \overline{\theta}] \\ \sec\overline{\theta} & \text{otherwise} \end{cases} \\
\underline{\sec\theta} &= \begin{cases} -\infty & \text{if } \pi/2 \in [\underline{\theta}, \overline{\theta}] \\ \sec\underline{\theta} & \text{otherwise} \end{cases} \\
\overline{\csc\theta} &= \max(\csc\underline{\theta}, \csc\overline{\theta}) \\
\underline{\csc\theta} &= \begin{cases} 1 & \text{if } \pi/2 \in [\underline{\theta}, \overline{\theta}] \\ \min(\csc\underline{\theta}, \csc\overline{\theta}) & \text{otherwise} \end{cases}
\end{aligned}
$$

Note that the formulae for $\overline{\csc\theta}$ and $\underline{\csc\theta}$ are valid only if $[\underline{\theta}, \overline{\theta}] \subseteq (0, \pi)$.