Multidimensional unfolding problem solution in the case of a single target

S V Dronov, K A Leongardt
Altai State University, 656049, Lenina, 61, Barnaul, Russia

Abstract. Multidimensional unfolding (statistical deployment, prefscal) is the method for placing simultaneously on the same map a set of observers and a set of targets with fixed distances between each observer and each of the targets. The approximate methods of solving it today suggest that in each of these sets there has at least two elements. In this paper, we propose a method for the exact solution of this problem, when the set of targets is a singleton. Algorithms are easily transferred to the case of only one observer.

Keywords: data visualization, multidimensional unfolding, prefscal methods.

1. Introduction

The solution of problems of visualization of various kinds of statistical data is used practically in any sphere of human activity. A lot of various algorithms have been created, allowing representing objects of observation with points in the space of a small number of measurements (most often on a plane). In the modern fields of application of data analysis, they often do not have a numerical character. At the same time, the visualized data allow somehow to assess the degree of difference in the objects under study, which suggests that the methods of visualization will become more and more popular.

The most known such method, apparently, is the multidimensional scaling [1, 2]. When solving problems of visualization of various kinds of surveys in sociology [3], psychology [4] and some other sciences, a method of the same class is used, but with incomplete data, which was called the multidimensional unfolding.

We give a brief statement of the problem of unfolding. Let there be two sets of objects, one of which is called a set of targets, and the other is a set of observers. Each of the observers reports the distance from themselves to each of the targets. Unlike multidimensional scaling, where all pairwise differences between objects are known, information about distances (differences) between objects, when they both belong to the same set, is missing. A task is to depict the objects of both sets on one map.

There are different approaches to the solution of this problem. For example, one can reduce this problem to the problem of multidimensional scaling by filling the missing cells in the matrix of differences, using special techniques, as suggested in [5]. In [6], the application of the two-stage least squares method to the multidimensional unfolding problem is described. But with any approach known to us, an indispensable requirement of its applicability is the presence of at least two targets and at least two observers.

The main task of the work is to create an algorithm for solving the unfolding problem in the case when we have exactly one target. Since sets of goals and observers actually enter into the formulation of the problem symmetrically, it will be possible to transfer the algorithm to the case of a single observer without any changes.
Let us have one goal and \( n \) observers, each of which must have a known distance \( d_i, i = 1, \ldots, n \) from the only target. We will assume that the observers are numbered, and so the distances are sorted in ascending order, more precisely, in nondecreasing way. The point depicting the target we place at the origin. This can be done without the loss of generality because in the tasks of visualization we are only interested in the location of objects relative to each other. The problem now is to locate the observers at known distances from the target.

Obviously, without additional constraints, the problem has infinitely many solutions, if we eliminate the trivial case of zero distances. Therefore, we will look for such points representing observers that they would be scattered most strongly: we require that the minimum of distances between the points representing the observers be the maximum possible. The emergence of these additional requirements in visualization problems is usually associated with the greatest visibility of the image. It is also possible to explain this otherwise. Let the purpose of the data processing is the detection of all observers on the ground, when the location of the target is known accurately. Then the proposed approach allows us to see the "most unfavorable" situation, which allows estimation of the time and costs of searching from above.

2. One-dimensional solution

Suppose that we need to select points representing observers on a straight line (so we get a one-dimensional visualization). Here we can suggest the following obvious algorithm.

Algorithm \( T_1 \)-line. Arrange the observers by increasing their distances to the target and number them accordingly. We construct points with coordinates \( d_i, i = 1, \ldots, n \), on a straight line, and then all the points with even numbers reflect symmetrically with respect to the origin. The resulting points are declared the locations of the observers. All observers will thus be located at points with coordinates \( (-1)^{i-1} d_i, i = 1, \ldots, n \).

Theorem 1. The \( T_1 \)-line algorithm gives the location of observer points on the line, maximizing the minimum distance between them.

The proof of the theorem is easily carried out by induction on the number \( n \) of observers.

3. Visualization for a larger number of dimensions

Let’s begin with the easier problem of locating points \( Ob_i, i = 1, \ldots, n \) on a plane, i.e. determine for each of them a pair of coordinates. The criterion for optimality of their location is still the maximality of the minimum distance between any two of them. The target is located at the origin. We will continue to assume, as before, that all observers are numbered by increasing their distances to the target. We will study the question of the location of a new point if several points have already been built. Suppose that there are arbitrary \( k \) points \( A_1, \ldots, A_k \) on the plane. For a fixed \( d > 0 \), we construct a system of circumferences of equal radii \( d \) with centers at these points. The following statement seems to be obvious.

Lemma. The locus of points for which the minimum distance to given \( k \) points is \( d \) is the union of those outer parts of the system of constructed circumferences that do not lie inside any of them.
Figure 1. $d$-envelope for a set of six points

An example of such a locus is shown in Figure 1. It is highlighted in bold lines. We agree to call this set the $d$-envelope of points $A_1, ..., A_k$. It is clear that the $d$-envelope of points consists of two parts - the inner and the outer, which is shown in the figure as a dashed bold line. The outer part will be further referred to as the $d$-boundary of the points $A_1, ..., A_k$. Those points of the $d$-envelope, which belong to two of the system of circumferences, will be called deepened points, the meaning of the name is clear from the figure, where the deepened dots are denoted by letters $U$.

Let us turn to the idea of the algorithm. Let several points $Ob_i, i=1,...,k$ have already been constructed. We draw a circumference with center at the origin of coordinates of radius $1kd$ ("the large circumference"). According to the assumption made earlier, all the constructed points lie inside it, and a new point must be sought on this circumference. Taking the $d$-envelope of the points already constructed for sufficiently small $d$, we begin to increase this number $d$ until some nonempty intersection of the $d$-envelope and the "large circumference" appears. It is clear that the inner part of the $d$-envelope is "collapsed", so that only the elements of the $d$-boundary of the points can be in the intersection. We continue increasing $d$ as long as the intersection remains nonempty. For $Ob_{k+1}$ we choose any of the points of this intersection for the greatest possible $d$.

**Theorem 2.** The chosen point has the greatest possible minimum of distances to points $Ob_i, i=1,...,k$ among all points of the "large circumference".

We note that from the proposed idea it follows that each of the points $Ob_i, i=1,...,k$ will eventually be selected from the set of deepened points of the $d$-boundary of points with smaller numbers. Since each deepened point lies on the median perpendicular connecting the centers of some two of the circles, this leads to the following algorithm for adding a new point to the already constructed configuration.

**Algorithm for adding a point**

- Step 1. Let’s draw a "large circumference" $C$ of radius $d_{k+1}$ with center at the origin. Then we find a convex polygon with vertices at some of already constructed points such that it contains all of them.
- Step 2. Restore the median perpendiculars to each side of this polygon until it intersects with $C$. Then we find the point of intersection, the length of the segment of the median
perpendicular at which is greatest. This point we choose for $Ob_{k+1}$. If there are several such points, choose an arbitrary one from them. End of algorithm.

Now we can propose an algorithm for constructing all points representing observers.

**Algorithm T1-plane**

- Step 0. Put the target at the point $(0, 0)$. The point $Ob_1$ is placed on the abscissa axis at the point $(d, 0)$. Then introduce the current number of the constructed point $k = 1$.
- Step 1. Construct $(k+1)$-th point $Ob_{k+1}$ using the algorithm for adding a point.
- Step 2. Are all points constructed? If not, increase $k$ and proceed to step 1. Otherwise, the end of the algorithm.

Note that in the two above algorithms, we can replace the "large circumference" $C$ by the boundary (or part of the boundary) of an arbitrary set containing all the previously constructed points. It is also clear that the T1-line algorithm is a special case of the T1-plane algorithm, when all the "large circumferences" degenerate into two points lying on the abscissa and remote from the origin at distances $d_i, i = 1, ..., n$.

The algorithm T1-plane can be simply generalized to the location of observers in space of any dimension. Then, instead of the system of circumferences, hyperspheres of appropriate dimensions should be used, and the medial perpendiculars should be reconstructed in the midpoints of the edges of a convex polyhedron containing the whole cloud of pre-constructed points. We call this procedure the algorithm T1.

![Figure 2. An example for T1-plane algorithm](image)

An example of the operation of the T1-plane algorithm for the case where a fifth point is constructed on four previously built points is shown in Fig. 2. In it, the point depicting the second observer is located on the abscissa axis on the other side of the origin than the point of the first. The third point lies on the median perpendicular between the first and second, the fourth point lies between them, but on the other side of the axis.

So the work of the algorithm T1 consists in the sequential addition of new points to the already constructed ones. We will call the fixed configuration of the points before the next step of the algorithm, the base one. The algorithm T1 guarantees the optimality of the resulting configuration of points only for the case when it adds one new point, which lies on the hypersphere, inside which the entire basic configuration is located. Now let’s propose a modification of this algorithm in which a new point is selected from the points of an arbitrary hypersurface $C$. The basic configuration with respect to $C$ can be arranged arbitrarily.

**Algorithm T1C**

- Step 1. We split the basic configuration into two parts, each of which lies entirely on one side of the given hypersurface $C$. 
Step 2. For each of the parts obtained, we construct a new point on \( C \) using the algorithm for adding a point. For each of the two constructed points, we calculate the minimum of all distances to the points of the basic configuration.

Step 3. Select the one of the two points for which this minimum distance is greater.

But, it turns out; there are simple examples where you can achieve a better result than the algorithm T1, if you simultaneously change the position of several points. For example, consider the task of visualizing three observers whose distances to the target are the same. Then, in our assumptions, all these points must lie on one circumference with the center at the origin. It is known that the maximum value of the minimum of pairwise distances between points in this case is attained if and only if they are the vertices of an equilateral triangle. But the algorithm T1 first puts two points at the ends of the diameter of the circle, and the third places in the middle of one of the formed semicircles (the point of intersection of the median perpendicular to the diameter and the circumference).

Therefore, to solve our problem correctly, we need to improve the algorithm T1. It should be borne in mind that a new optimal configuration cannot always be built in a finite number of steps as the example below shows. Therefore, we have to build the optimal construction iteratively with some predetermined accuracy.

**Algorithm T1+**

- Step 0. Choose the desired accuracy of the calculation of \( \varepsilon \). Let's build the starting configuration of the points \( Q \) by the algorithm T1. Let us calculate the current minimum of pairwise distances \( D_1 \) in the configuration. Remember the configuration \( Q \).
- Step 1. Let \( D_1 \) be reached on the pair of points \((A, B)\).
- Step 2. We take as the basic configuration the one obtained by removing the point \( A \). After \( C \) we take a hypersphere with center at the origin and radius \( d_A \) corresponding to the point \( A \). Let's construct a new point depicting the same observer that previously was depicted by point \( A \), using the algorithm for adding a point.
- Step 3. We find the minimum distance from the new point to all the others. If it is greater than \( D_1 - \varepsilon \), then go to step 6. Otherwise, we return point \( A \) to its previous place and proceed to the next step.
- Step 4. Remove \( B \). We construct a new point instead of it as in step 2.
- Step 5. We find the minimum distance from the new point to all the others. If it is greater than \( D_1 - \varepsilon \), then proceed to step 6 with a new configuration. Otherwise we return the point \( B \) to its place.
- Step 6. We find the value of the current minimum of distances \( D_2 \) in the configuration. If it is less than \( D_1 - \varepsilon \) or configuration has not changed, then the output from the algorithm. If not, we change \( D_1 \) to \( D_2 \), \( Q \) to the current configuration and go to step 1.

The algorithm T1+ finds the exact solution for the unfolding problem in the case of the only target if it is attainable in a finite number of steps or its approximation up to the accuracy of calculations \( \varepsilon \), if not.

**Example.** Suppose that we had three observers, each of which reports the same distance to the target. For simplicity of calculations, let's assume that this distance is \( 1/2\pi \) (then the length of the circle on which all the points will be located will be 1). Choose the accuracy of the calculations \( \varepsilon = 0.05 \). The successive stages of the operation of the T1+ algorithm are shown in Figures 3-4.
Figure 3. Algorithm T1+ for three observers

In Figure 3, the left configuration shows the starting configuration (the result of the T1plane algorithm). Then according to the algorithm T1+, the first point moves to a new position. It is shown in the middle part of the figure. In its right part you can see how point 2 moves further. The minimum distance between pairs of points under construction does not change at first, then it increases by 0.25, which does not allow the algorithm to finish its work, it is more than the specified accuracy of calculations.

In Figure 4, the algorithm continues to work, as a result of which, in the middle part, the change in the maximum is $1/64 < 0.02 < \varepsilon$. The algorithm is finished. On the right in Figure 4 the final configuration is shown. The lengths of the arcs drawn by the sides of the triangle are $11/32$, $21/64$ and $21/64$. It is clear that this triangle is equilateral within a selected accuracy $\varepsilon$.

4. Discussion and conclusions

In this paper, an algorithm T1+ is proposed for solving the multidimensional unfolding problem for the case when one of the two basic sets is one-element. In this case, as an additional requirement for the desired solution, the condition of maximizing the minimum of pairwise distances between images of objects of the second of the basic sets is put forward. In the authors' opinion, this algorithm should be included in the modules of statistical packages that solve the problems of unfolding and multidimensional scaling. Until now, most statistical packages have not given the user the ability to handle such situations, although in practice the needs for solving such problems certainly arise.

We would also like to note that the proposed algorithm and its parts after a small modification can solve the problem of finding points lying in arbitrary fixed sets so that the minimum distance between
them is maximized. You can also use the proposed algorithm for the point adding to build a point in the given set, the most distant from some finite set of points. These and similar tasks are in demand not only for data visualization, but also for solving some extreme geometric problems.

Reference

[1] Borg I Groenen P 2005 *Modern Multidimensional Scaling: theory and applications* (New York: Springer-Verlag)
[2] Dronov S V 2015 *Methods and Problems of the Multidimensional Statistics* (Barnaul: Altai State University Publishing House) (In Russian)
[3] Price R H Bouffard D L 1974 *Journal of Personality and Social Psychology* v. 30
[4] Nesselroade John R Cattell Raymond B 2013 *Handbook of Multivariate Experimental Psychology* (New York: Springer Science & Business Media)
[5] Leeuv Jan de 2005 Multidimensional Unfolding *The Encyclopedia of Statistics in Behavioral Science* (New York: Wiley)
[6] Heiser W J 1987 Joint Ordination of Species and Sites: The Unfolding Technique *Developments in Numerical Ecology* (Berlin, Heidelberg: Springer-Verlag) pp 189 – 221