THE N-DIMENSIONAL MAP MAKER ALGORITHM

John R Rankin

Department of Computer Science and Computer Engineering
La Trobe University, Australia

ABSTRACT

The Map Maker algorithm which converts survey data into geometric data with 2-dimensional Cartesian coordinates has been previously published. Analysis of the performance of this algorithm is continuing. The algorithm is suitable for generating 2D maps and it would be helpful to have this algorithm generalized to generate 3D and higher dimensional coordinates. The trigonometric approach of the Map Maker algorithm does not extend well into higher dimensions however this paper reports on an algebraic approach which solves the problem. A similar algorithm called the Coordinatizator algorithm has been published which converts survey data defining a higher dimensional space of measured sites into the lowest dimensionalcoordinatization accurately fitting the data. Therefore the Coordinatizator algorithm is not a projection transformation whereas the n-dimensional Map Maker algorithm is.

KEYWORDS

N-dimensional space, Projections, Distance Matrices, Map Maker, Coordinatizations

1. INTRODUCTION

The Map Maker algorithm [1] serves a fundamental purpose. Given geodetic survey data on N sites P_i for i = 1 to N in the form of an NxN distance matrix D the algorithm generates 2D coordinates for each point i.e. it determines x_i and y_i such that $P_i = (x_i,y_i)$ satisfies the distance matrix D, according to

$$D_{ij} = Distance(P_i, P_j) \tag{1}$$

for all i and j = 1 to N. The points P_i however may not in reality all lie on a single flat plane and geodetic survey data is well-known to be non-planar. For example we can test the Map Maker algorithm by starting with a given three dimensional point set $P_i = (x_i,y_i,z_i)$ for i = 1 to N and from these generate a distance matrix D via Eq(1) using the 3-dimensional Euclidean distance measure. Then subsequently we may take this distance matrix D and feed it through the Map Maker algorithm to produce N points Q_i for i = 1 to N in 2 dimensions. The generated points Q_i can then be plotted on the flat screen or output page. Taken together these steps constitute a projection of points P_i in 3D to corresponding points Q_i in 2D. This 'projection' is not a formula applicable to single points as it requires a set of N ≥ 3 points to perform the process. The first point P_1 is fixed at the origin so that $Q_1 = (0,0)$. The second point $P_2$is placed along the x-axis in the positive direction so that $Q_2 = (D_{12},0)$. The third point P_3 is placed such that Q_3 has 2-dimensional distance D_{13} from point Q_1 and 2-dimensional distance D_{23} from point Q_2 and a positive y component. This is equivalent to selecting a plane in 3D space which is the plane of the 3D triangle $P_1P_2P_3$, then selecting an origin in the plane as point P_1, the x-axis direction as the vector P_1P_2 and the y axis perpendicular to this such that P_3 is on the positive y side of the plane. All other points $P_i$for i = 4 to Nare projected onto this plane by rotating each triangle $P_iP_1P_2$in 3D to the triangle $P_i'P_1P_2$ where P_i' lies in the plane of $P_1P_2P_3$. Whether P_i' is placed in the positive y half of the plane or

DOI : 10.5121/ijcga.2014.4402

the negative side is based on which choice has a closer distance value to the distance D_{3i}. In this way all distances D_{1i} and D_{2i} are preserved and D_{3i} is as close to preserved as possible in the projection. Denoting the distance matrix of the projected points as D' we therefore have $D_{1i} = D_{1i}'$ and $D_{2i} = D_{2i}'$ for i = 1 to N. The other elements of the D and D' matrices will in general differ and the Map Maker algorithm tries to minimize these differences. A measure of the effectiveness of this minimization is the root mean square error (RMSE) in the differences of D and D'. Work in reference [1] suggests that reordering the points which results in a permutation of the D matrix can result in lower RMSE values. The RMSE values are also dependent on N, the range of coordinate values that were used to generate D and their dimensionality.

Two other projection operations were proposed by Lee in [2] called the Second Nearest Neighbour approach and the Reference Point approach although he does not attempt to compute coordinates for the mapped points. While the Map Maker algorithm preserves the distances of all points to P_1 and P_2, the Second Nearest Neighbour approach preserves the distances of the next mapped point P_k to the two nearest previously mapped points P_i and P_j and the Reference Point approach preserves distances of the next mapped point P_k to P_1 and the previously mapped nearest point P_i (to point P_k). Thus Lee's methods lead to different spatial locations and therefore coordinatizations of the sites P_i unless the sites are intrinsically two-dimensional from the start. Other authors ([3-12]) have shown the importance of locating objects in space given the distance matrix alone. Rankin in [13] has shown the remarkable result that the distance matrix can determine the dimension of the embedding space and that this space can be determined to be either Euclidean or pseudo-Euclidean from the distance matrix alone. This is achieved through the Coordinatizator algorithm described in that paper.

Youldash and Rankin {14] have shown the methodology adopted for evaluating the speed and accuracy of the Map Maker algorithmand preliminary results were given. Graphing the results was found to be difficult due to the wide statistical variations coming from random point sets so that statistical methods need to be applied. While the 2D Map Maker algorithm generates a 2D coordinatization of the survey sites from their distance matrix, the 3D Map Maker algorithm generates a 3D coordinatization of the survey sites from their distance matrix. It is therefore reasonable to expect that if the survey sites are intrinsically located in a 3-dimensional space, that the 3D Map Maker algorithm should in general produce lower RMSE values. The first aim of this research was to create the 3D Map Maker algorithm and thence test this conjecture. This is achieved in section 2 below with the pseudo-code in section 3 and testing results in section 6. The next aim of the research was to carry this process forward into n-dimensional space and sections 4 and 5 below report on this work.

2. THE 3D MAP MAKER ALGORITHM

Consider N sites P_i with a distance matrix D and we want to create 3D coordinates for each site. We will proceed as in the Map Maker algorithm by declaring the first site P_1 to be located at the origin so that

$$P_1 = (0,0,0) \tag{2}$$

Next P_2 is declared to define the direction of the x-axis preserving the distance D_{12} so that

$$P_2 = (D_{12},0,0) \tag{3}$$

Next site P_3 is declared to define the direction of the y axis by having $P_{3y} > 0$ and $P_{3z} = 0$. As in the Map Maker algorithm we compute the angle $P_2P_1P_3$ as α so that

$$P_3 = (D_{13}cos\alpha, D_{13}sin\alpha, 0) \tag{4}$$

Finally we define the sense of the z-axis perpendicular to the plane of $P_1P_2P_3$ by asserting that P_4 has a positive z component. Figure 1 shows the geometry where C is the point subtended by P_4 on the plane of triangle $P_1P_2P_3$. Thus

$$P_4 = (x_4, y_4, z_4) \tag{5}$$

where $z_4 > 0$.We need to relate the x_4, y_4 and z_4 to the components of D. Thus the problem is to solve the tetrahedron for x_4, y_4 and z_4 when all the side lengths D_{12}, D_{13}, D_{14}, D_{23}, D_{24} and D_{34} of the tetrahedron are known. The solution is:

$$x_4 = \frac{D_{12}^2 - D_{24}^2 + D_{14}^2}{2D_{12}} \tag{6}$$

$$y_4 = \frac{D_{13}^2 - 2D_{13}x cos\alpha - D_{34}^2 + D_{14}^2}{3D_{13} sin\alpha} \tag{7}$$

$$z_4 = \sqrt{D_{14}^2 - x_4^2 - y_4^2} \tag{8}$$

whereα is the same angle used above andalso used in the 2D Map Maker algorithm.

We now enter a loop to place points P_i for i = 5 to N in 3D space. To place point P_ii.e. to determine x_i, y_i and z_i, we use the same formulas as Equations (6) to (8) except replacing index 4 with i. For placing P_4 the positive root is taken in equation (8) but for index i> 4 we must decide for each point $P_i$whether to flip the point and use the negative root instead. As with the 2D Map Maker algorithm, the decision is made by seeing which point P_i' or P_i' flipped has a distance to P_4 which is closer in value to distance D_{4i}.

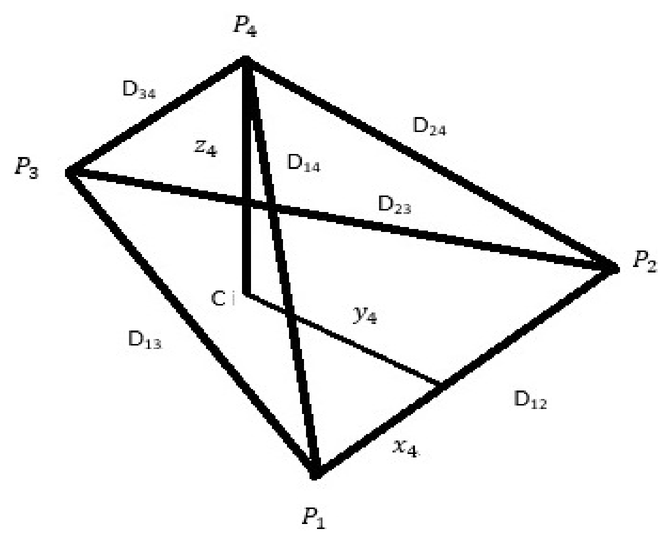

Figure 1. Deriving the coordinates of P_i for i = 4 to n.

3. 3D MAP MAKER PSEUDO-CODE

The 3D Map Maker algorithm can be expressed in the following pseudo-code which is readily converted to C++, Java or other suitable programming language. It assumes only that the NxN real matrix D is available.

```
Set P[1] = (0,0,0)
Set P[2] = (D[1,2],0,0)
Set P[3] = (D[1.3]*cos_alpha,D[1,3]*sin_alpha,0)
for i = 4 to N
Set x = (Sq(D[1,2]) – Sq(D[2,i]) + Sq(D[1,i]))/(2*D[1.2])
Set y = (Sq(P[3].x) + Sq(P[3].y) – 2*P[3].x*x – Sq(D[3,i]) + Sq(D[1,i]))/2*P[3].y)
    Set z = SqRoot(Sq(D[1,i]) – Sq(x) – Sq(y))
Set P[i] = (x,y,z)

   // test for point flipping:
if i> 4 then
Set Flipped = (x,y,-z)
if Abs(D[4,i] – Distance(P[4],Flipped)) < Abs(D[4,i] – Distance(P[4],P[i])) then
Set P[i] = Flipped
```

This pseudo-code assumes that 3D points are stored as C++ structs with components called x, y and z and that the point array is dimensioned to hold elements 1 to N. The terms cos_alpha and sin_alpha are computed trigonometrically (once only for all sites i> 4) as in the Map Maker algorithm. The code has been implemented and undergone extensive testing which is reported in section 6.

4. THE N-DIMENSIONAL MAP MAKER ALGORITHM

In 2D we used the first 3 points to define the 2D coordinate system: P_1 defines the origin of coordinates, P_2 defines the x-axis and P_3 defines the y-axis.In 3D we used the first 4 points to define the 3D coordinate system: P_1 defines the origin of coordinates, P_2 defines the x-axis, P_3 defines the y-axis and P_4 defines the z-axis. So in n dimensions we need to use the first n+1 points to define the coordinate system and then the remaining points can be located inside this coordinate system. Therefore the first n+1 points will be coordinatized as follows:

$$\begin{gathered} P_1 = (0,0,0,\dots 0) \\ P_2 = (x_{21},0,0,\dots 0) \\ P_3 = (x_{31},x_{32},0,\dots 0) \\ P_4 = (x_{41},x_{42},x_{43},\dots 0) \\ :::::::::: \\ P_n = (x_{n1},x_{n2},x_{n3},\dots\ x_{n\,n-1},0) \\ P_{n+1} = (x_{n+1\,1},x_{n+1\,2},x_{n+1\,3},\dots\ x_{n+1\,n}) \end{gathered} \tag{9}$$

In these formulas it is required in defining axial directions that

$$x_{i\,i-1} > 0 \tag{10}$$

fori = 2 to n+1. There is no sign restriction on the other x_{ij} values. It is also clear that in selecting a projection dimension n we are restricted to the requirement:

$$N \geq n + 1 \tag{11}$$

In the case of equality in (11), the problem reduces to the Coordinatizator algorithm solved in [13]. For the n-dimensional Map Maker algorithm therefore we will assume that n < N-1.The components x_{ij} for i = 2 to n+1 and j = 1 to i-1 are n(n+1)/2 unknowns to be determined first. After the x_{ij} are determined then the components of any point P_i for i> n+1 can be determined in terms of the x_{ij} and D. To see how the components of P_i are determined in terms of the x_{ij} and D, for simplicity denote the components of P_i as x_i for i = 1 to n. Then form the n equations:

$$(\boldsymbol{P_i} - \boldsymbol{P_j})^2 \equiv \boldsymbol{D_{ij}^2} = \sum_{\boldsymbol{k=1}}^{\boldsymbol{j-1}} (\boldsymbol{x_k} - \boldsymbol{x_{jk}})^2 + \sum_{k=j}^{n} x_k^2 \tag{12}$$

for j = 1 to n. These are n equations in n unknowns, the x_i for i = 1 to n. The general solution is given by the following recurrence relation:

$$x_k = \left\{ \sum_{j=1}^{k} x_{k+1\,j}^2 - D_{k+1\,i}^2 + D_{1\,i}^2 - 2\sum_{j=1}^{k-1} x_j x_{k+1\,j} \right\} / (2x_{k+1\,k}) \tag{13}$$

for k = 1 to n-1 and finally

$$x_n = \pm \sqrt{D_{1\,i}^2 - \sum_{j=1}^{n-1} x_j^2} \tag{14}$$

The plus or minus sign in equation (14) indicates that a flip decision has to be made for this component to minimize the RMSE. The decision is based on which point flipped or unflipped results in a distance value closer to $D_{i\ n+1}$.Equations (13) and (14) are the general solution to finding the coordinates of P_i for i> n+1 in terms of the components x_{ij} and the given distance matrix D. Equation (13) is the generalization of the cosine rule as used in Map Maker and equation (14) is the generalization of the sine formula used in the Map Maker algorithm. Furthermore this procedure for deriving the coordinates of $P_i$also applies to finding the x_{ij} in the first place by sequentially solving for the x_{ij} down the list given in (9) and always taking the positive root in equation (14). However in doing this care must be taken in computing only the coordinates up to i-2 allowing the remainder to be zero. This process is shown in the algorithm presented in the next section.

5. ALGORITHM

For implementing the n-dimensional Map Maker's algorithm the following pseudo-code represents this new algorithm. For this algorithm each point is a dynamic array of floating point numbers (doubles) of length n and the set of all points is a dynamic array of length N of this point type.

```
for i = 1 to N
    for j = 1 to n
        Set P[i,j] = 0
```

```
Set P[2,1] = D[1,2]
Set P[3,1] = D[1.3]*cos_alpha
Set P[3.2] = D[1,3]*sin_alpha
for i = 4 to n+1
   for j = 1 to n
      Pi[j] = 0
   for k = 1 to i-2
      Set sum1 = 0
      Set sum2 = 0
      for j = 1 to k
         Set sum1 = sum1 + Sq(P[k+1,j])
         Set sum2 = sum2 + Pi[j]*P[k+1,j]
      Set P[I,k] = (sum1 – Sq(D[k+1,i]) + Sq(D[1,i]) – 2*sum2)/(2*P[k+1,k])
      Set sum1 = 0
      for j = 1 to n-1
         Set sum1 = sum1 – Sq(Pi[j])
      Set Pi[i-1] = SqRoot(sum1)
      for j = 1 to n
Set P[i,j] = Pi[j]
for i = n+2 to N
   for j = 1 to n
      Pi[j] = 0
   for k = 1 to n-2
      Set sum1 = 0
      Set sum2 = 0
      for j = 1 to k
         Set sum1 = sum1 + Sq(P[k+1,j])
         Set sum2 = sum2 + Pi[j]*P[k+1,j]
      Set P[i,k] = (sum1 – Sq(D[k+1,i]) + Sq(D[1,i]) – 2*sum2)/(2*P[k+1,k])
      Set sum1 = 0
      for j = 1 to n-1
         Set sum1 = sum1 – Sq(Pi[j])
Set Pi[n] = SqRoot(sum1)
      for j = 1 to n-1
         Set Flipped[j] = Pi[j]
      Set Flipped[n] = -Pi[n]
if Abs(D[n+1,i] – Distance(P[n+1],Flipped)) < Abs(D[n+1,i] – Distance(P[n+1],P[i])) then
         Set Pi = Flipped
      for j = 1 to n
         Set P[I,j] = Pi[j]
```

The algorithm for the n-dimensional Map Maker is clearly longer and more detailed than either the Map Maker or 3D Map Maker algorithms and requires more memory. The next section will present the initial results and implications of this algorithm.

6. DISCUSSION

Normally when making maps from survey distance information we expect that the maps will be drawn on a flat 2D sheet of paper or flat screen and the Map Maker algorithm performs this service. However the concept is generalizable so that given the distance matrix for a set of N sites one can now generate Cartesian coordinates for each site for any specified dimension n decided in advance. For geodetic survey data in our 3-dimensional world we could expect that n = 3 is sufficient. The 3D output when displayed graphically shows the positional relationships between the survey sites and the user can view this 3D data from many different angles and viewpoints. Research is underway to convert the 3D coordinatized point data into a height map that can show the detailed topography and lie of the land.

To test the 3D Map Maker algorithm, many sets of random 3D points were constructed. These sets were each converted into distance matrices and stored in files. The distance matrix files were then given to the 3D Map Maker algorithm which converted the distance matrix input to 3D coordinates for each survey point. The output 3D point sets do not directly compare with the initial 3D point sets that were constructed because the algorithm causes a translation operation to P_1 as origin and then reflections and rotations of the points about P_1so that P_1P_2 aligns with the x-axis, P_3 is in the x-y plane with positive y component and P_4 has positive z component. Therefore instead of comparing the point sets, we compute a new distance matrix based on the output points from the algorithm and compare this with the original distance matrix. The results varied based on the number N of points in the set. As an example with N = 5, a total RMS absolute error value of 0.000104 was obtained with a total RMS relative error of 0.00000581. For N = 10 a sample result was 0.00145 for the absolute error and 0.00337 for the relative error.

A similar testing process was used on the n-dimensional Map Maker algorithm. For N = 10 and n = 5 a total RMS absolute error was 0.0056 with a total RMS relative error of 0.00577. Low errors suggests that the n-dimensional Map Maker algorithm is doing its job correctly. If the distance matrix used in the algorithm was created from a point set of dimensionality different from n then we could expect the error rate to rise. This is subject to future testing.

It should be noted that the n-dimensional Map Maker algorithm produces enantiomorphs because handedness (i.e. parity) cannot be transmitted in the distance matrix. This means that the spatial arrangement of the nodes can be any of 2^{n-1} possible mirror images produced by parity inversions i.e. reflections along any axis other than the first and all enantiomorphs have the same distance matrix D making them indistinguishable in the Map Maker algorithms.

CONCLUSIONS

The n-dimensional Map Maker's algorithm converts spatial distance survey data to point coordinates in n dimensions. The first n+1 points get mapped to prescribed positions to define the positive directions of the n axes. In so doing these first n+1 points contain n(n+1)/2 zeros in their coefficients and preserve n(n-1)/2 distances. If the distance matrix was originally derived not by survey measurement but from N points in a k-dimensional space then the n-dimensional Map Maker's algorithm can be viewed as a projection transformation from k dimensions to n dimensions. If $N > n+1$ and $k > n$ we can expect to see distance preservation failures indicated by a significant positive RMSE. If $N \leq n+1$ or $k \leq n$ there should be no significant systemic errors so that only numerical calculation errors should appear. Continuing work in this area includes testing of the speed of the algorithm and the behaviour of the RMS errors as well as applications of the algorithm to real world problems.

REFERENCES

[1] Rankin J, "Map Making From Tables" IJCGA, April 2013, Vol3, Nr 2 pp 11-17. http://airccse.org/journal/ijcga/current2013.html

[2] Lee RTC et al, "A Triangulation Method for the Sequential Mapping of Points from N-Space to Two-Space", IEEE Transactions on Computers, March 1977, pp288-292.

[3] Yin H, "Nonlinear Dimensionality Reduction and Data Visualization: A Review", International Journal of Automation and Computing 04(3), July 2007, pp294-303.

[4] Sammon J W Jr, "A Nonlinear Mapping for Data Structure Analysis", IEE Transaction on Computers, Vol C-18, No. 5, May 1969, pp401-409.

[5] Shisko JF et al, "IGODS: An Important New Tool for Managing and Visualizing Spatial Data", OCEANS 2010, pp1-10.

[6] Li Q et al, "A Chunking Method for Euclidean Distance Matrix Calculation on Large Dataset Using Multi-GPU", IEEE Machine Learning and Applications (ICMLA) Dec 2010, pp 208-213.

[7] Hyo-Sung A, "Command Coordination In Multi-Agent Formation: A Distance Matrix Approach", IEEE Control Automation and Systems (ICCAS) 2010, pp 1592-1597.

[8] Srinkanthan S, "Accelerating the Euclidean Distance Matrix Computation Using GPUs", IEEE Electronics Computer Technology (ICECT) 2011, pp422-426.

[9] Del Bue A et al, "2D-3D Registration of Deformable Shapes with Manifold Projection", 2009, pp 1061-1064.

[10] Drineas P et al, "Distance Matrix Reconstruction from Incomplete Distance Information for Sensor Network Localization", Sensor and Ad Hoc Communications and Networks 2006, pp 536-544.

[11] Ayhan S et al "Implementing Geospatially Enabled Aviation Web Services", Integrated Communications, Navigation and Surveillance (ICNS) 2008, pp1-8.

[12] Xiao Q et al,"Application of Visualization Technology in Spatial Data Mining", Computing, Control and Industrial Engineering (CCIE), 2010, pp153-157.

[13] Rankin J, "Visualization of General Defined Space Data", International Journal of Computer Graphics and Animation, October 2013, Vol3, Nr 4.http://airccse.org/journal/ijcga/current2013.html

[14] Youldash MM & Rankin J, "Evaluation of A Geometric Approach ToCoordinatization of Metric Spaces", 8th International Conference on Information Technology and Applications, ICITA 2013, July Sydney 2013.