Visualization of General Defined Space Data

John R Rankin

La Trobe University, Australia

Abstract

A new algorithm is presented which determines the dimensionality and signature of a measured space. The algorithm generalizes the Map Maker's algorithm from 2D to n dimensions and works the same for 2D measured spaces as the Map Maker's algorithm but with better efficiency. The difficulty of generalizing the geometric approach of the Map Maker's algorithm from 2D to 3D and then to higher dimensions is avoided by using this new approach. The new algorithm preserves all distances of the distance matrix and also leads to a method for building the curved space as a subset of the N-1 dimensional embedding space. This algorithm has direct application to Scientific Visualization for data viewing and searching based on Computational Geometry.

Categories and Subject Descriptors: I.3.3 Viewing algorithms, I.3.5 Computational Geometry and Object Modelling

1. Introduction

The Map Maker's problem is to convert a measured space into a two-dimensional coordinatized space. This is well achieved by the Map Maker's algorithm [Ran13]. However the assumption in that process is that the distance matrix of the measured space corresponds at least roughly to a flat two-dimensional space. When this is true the Map Maker's algorithm is distance preserving. In general however this would not be the case and then the Map Maker's algorithm does not preserve all distances. A measured space is defined by a set of sites i = 1 to N and their measured separations d_{ij} where d_{ij} is the shortest distance between site i and site j. The set of NxN distances d_{ij} is called the Distance Matrix. This matrix is symmetric and trace-free with zeros down the principal diagonal. Thus a set of known sites and their distance matrix constitutes what is called a defined (or measured) space.

Research based on distance matrices has had wide applications such as in signal processing [PC92], pattern recognition [ZWZ08] and medical imaging [PE*09]. One of the foundational research efforts [LSB77] resulted in Lee's algorithm for projecting n-dimensional distance matrix data to two dimensions for visualization. Lee's algorithm does not determine the n-dimensional coordinates of the points involved but goes straight to a projection of those points in 2D. In the process of this projection most of the distance information is not preserved in the visualization. Other approaches such as LLE preserve distances just in the local neighbourhood of a chosen point [RS00, TSL00]. In the research presented here the coordinatization and projection processes are separated so that one is free to apply any desired projection transformation to the coordinatized data. Surprisingly, from the distance matrix d_{ij} one can determine the dimensionality and signature of the space that it defines as well as the coordinates for the sites and the algorithm presented in this paper demonstrates this.

DOI : 10.5121/ijcga.2013.3402

The algorithm presented here, called the Coordinatizator algorithm, does not require any special ordering of the sites before processing. The given ordering of the sites however determines the directions and orientations of all axes in the building of the n-dimensional space. Re-orderings of the sites therefore may be done for other reasons such as desired axial directions or reduction in numerical errors. The Coordinatizator algorithm coordinatizes by considering the sequence of simplices $S_k = P_1P_2\ldots P_k$ with vertices P_i for i = 1 to k and setting up orthogonal axes for these simplices.

2. Coordinatizator Process

The Coordinatizator algorithm uses the given sites to construct axes and produce coordinates for each of the sites. As in the Map Maker's algorithm, site 1 is taken as the origin so it is coordinatized as the point:

$P_1 = (0, 0, 0, 0...)$

The direction from site 1 to site 2 is taken as the x-axis or the first axis so therefore site 2 is coordinatized as the point:

$P_2 = (x_{21}, 0, 0, 0...)$
and obviously $x_{21} = d_{12}$.

The triangle of site 1 to site 2 to site 3 defines the x-y plane and thus the 2-axis (i.e. the y-axis which is perpendicular to the 1-axis which is the x-axis). Therefore P_3 is coordinatized as the point:

$P_3 = (x_{31}, x_{32}, 0, 0, 0...)$

The tetrahedron formed by the first 4 sites defines the third axial direction perpendicular to the first two. Thus site 4 is coordinatized as the point:

$P_4 = (x_{41}, x_{42}, x_{43}, 0, 0, 0...)$
and so forth so that in general:

$P_i = (x_{i1}, x_{i2}, x_{i3}, \ldots, x_{ii-1}, 0, 0, 0, 0...)$
for i = 1 to N where N is the number of sites and the size of the Distance Matrix.

This process of converting the Distance Matrix into a set of N points in n-dimensional space generally creates a non-Euclidean space of n = N-1 dimensions. We add the points progressively by building up a simplex from 0 dimensions to n = N-1 dimensions. However it is possible that at some step the added point is degenerate and the dimension does not increase for that step so that in general the dimensionality of the measured space is bounded above by N-1, $n \le N-1$. It is interesting that every time another point say the k+1th point is added, we only have to consider k more edges since $d_{ij} = d_{ji}$.

The Coordinatizator algorithm solves the following mathematical equations for the x_{ij} given the d_{ij}. Investigation into these equations found that the algebraic solution was possible for the general n-dimensional case.

Given:
$P_1 = (0, 0, 0, 0...)$
$P_2 = (x_{21}, 0, 0, 0...)$
$P_3 = (x_{31}, x_{32}, 0, 0, 0...)$
$P_4 = (x_{41}, x_{42}, x_{43}, 0, 0, 0...)$
$P_5 = (x_{51}, x_{52}, x_{53}, x_{54}, 0, 0, 0...)$
:::::
$P_i = (x_{i1}, x_{i2}, x_{i3}, \ldots, x_{ii-1}, 0, 0, 0, 0...)$

The unknowns x_{ij} in the above equations can be solved for k = 1 to N by the following procedure. Every coordinatization is extended to higher dimensions by appending the additional required number of zeros in the n-tuple.

For k = 1 we consider only point P_1 and coordinatize it by assign it to the 0 dimensional origin.
For k = 2 we bring in point P_2 and get:
$d_{12}^2 = (P_2 - P_1)(P_2 - P_1) = x_{21}^2$

and without loss of generality we choose:

$x_{21} = d_{12}$
and thereby coordinatize P_2 as the 1 dimensional point (x_{21}).

For k = 3 we bring in point P_3 and coordinatize it by getting:
$d_{13}^2 = (P_3 - P_1)(P_3 - P_1) = x_{31}^2 + x_{32}^2$
$d_{23}^2 = (P_3 - P_2)(P_3 - P_2) = (x_{31}-x_{21})^2 + x_{32}^2$
Subtracting gives:
$d_{13}^2 - d_{23}^2 = 2x_{21}x_{31} - x_{21}^2$
so that
$x_{31} = (d_{13}^2 - d_{23}^2 + x_{21}^2)/(2x_{21})$
and therefore

$$x_{32} = \pm\sqrt{{d_{23}}^2 - (x_{31} - x_{21})^2}$$

Now if the term in the square root is negative then we have non-Euclidean flat space. We will consider this type of situation later on. Here we will take the positive root without loss of generality since this will be the first point with a y component.

For k = 4 we are adding in P_4 and we have these three equations to consider:

$d_{14}^2 = (P_4 - P_1)(P_4 - P_1) = x_{41}^2 + x_{42}^2 + x_{43}^2$
$d_{24}^2 = (P_4 - P_2)(P_4 - P_2) = (x_{41}-x_{21})^2 + x_{42}^2 + x_{43}^2$
$d_{34}^2 = (P_4 - P_3)(P_4 - P_3) = (x_{41}-x_{31})^2 + (x_{42}-x_{32})^2 + x_{43}^2$
Therefore:
$d_{14}^2 - d_{24}^2 = 2x_{21}x_{41} - x_{21}^2$
so that
$x_{41} = (d_{14}^2 - d_{24}^2 + x_{21}^2)/(2x_{21})$
Secondly
$d_{24}^2 - d_{34}^2 = (x_{41}-x_{21})^2 - (x_{41}-x_{31})^2 + 2x_{32}\, x_{42} - x_{32}^2$

so that
$x_{42} = (d_{24}^2 - d_{34}^2 - (x_{41}-x_{21})^2 + (x_{41}-x_{31})^2 + x_{32}^2)/(2x_{32})$
Finally we get:

$$x_{43} = \pm\sqrt{d_{34}{}^2 - (x_{41} - x_{31})^2 - (x_{42} - x_{32})^2}$$

Now if the term in the square root is negative then we have non-Euclidean flat space. We will consider this type of situation later on. Here we will take the positive root without loss of generality since this will be the first point with a z component (i.e. 3rd or kth component).

For k = 5 we are adding in P_5 and we have these four equations to consider:

$d_{15}^2 = (P_5 - P_1)(P_5 - P_1) = x_{51}^2 + x_{52}^2 + x_{53}^2 + x_{54}^2$
$d_{25}^2 = (P_5 - P_2)(P_5 - P_2) = (x_{51}-x_{21})^2 + x_{52}^2 + x_{53}^2 + x_{54}^2$
$d_{35}^2 = (P_5 - P_3)(P_5 - P_3) = (x_{51}-x_{31})^2 + (x_{52}-x_{32})^2 + x_{53}^2 + x_{54}^2$
$d_{45}^2 = (P_5 - P_4)(P_5 - P_4) = (x_{51}-x_{41})^2 + (x_{52}-x_{42})^2 + (x_{53}-x_{43})^2 + x_{54}^2$
Therefore:
$d_{15}^2 - d_{25}^2 = 2x_{21}x_{51} - x_{21}^2$
so that
$x_{51} = (d_{15}^2 - d_{25}^2 + x_{21}^2)/(2x_{21})$
Secondly
$d_{25}^2 - d_{35}^2 = (x_{51}-x_{21})^2 - (x_{51}-x_{31})^2 + 2x_{32}\, x_{52} - x_{32}^2$
so that
$x_{52} = (d_{25}^2 - d_{35}^2 - (x_{51}-x_{21})^2 + (x_{51}-x_{31})^2 + x_{32}^2)/(2x_{32})$
Thirdly
$d_{35}^2 - d_{45}^2 = (x_{51}-x_{31})^2 - (x_{51}-x_{41})^2 + (x_{52}-x_{32})^2 - (x_{52}-x_{42})^2 + 2x_{43}\, x_{53} - x_{43}^2$
so that
$x_{53} = (d_{35}^2 - d_{45}^2 - (x_{51}-x_{31})^2 + (x_{51}-x_{41})^2 - (x_{52}-x_{32})^2 + (x_{52}-x_{42})^2 + x_{43}^2)/(2x_{43})$
Finally we get:

$$x_{54} = \pm\sqrt{d_{45}{}^2 - (x_{51} - x_{41})^2 - (x_{52} - x_{42})^2 - (x_{53} - x_{43})^2}$$

Now if the term in the square root is negative then we have non-Euclidean flat space. We will consider this type of situation later on. Here we will take the positive root without loss of generality since this will be the first point with a kth component.

This forms a clear pattern for any value of k >= 1 and hence the algorithm can be worked for any value of N.

3. Imaginery Components

The quantities under the square root signs in the equations above could be positive, zero or negative giving rise to three important cases to consider. If the quantity is positive we have an extra Euclidean dimension for housing the points. If the quantity is zero then we have sufficient dimensions already and an extra dimension is not (yet) required for embedding the points considered so far. If the quantity is negative then the Euclidean space becomes pseudo-Euclidean [Wik13a] like Minkowski space in Special Relativity where time is an imaginary extra axial dimension for three-dimensional space. As in Relativity, the invariant quadratic form then is no

longer positive definite and is properly termed a pseudo-metric though it is common to continue to call it a metric anyway).
Sylvester's Theorem [Wik13b] states that the matrix for any quadratic form can be diagonalized by a suitable change of coordinates and with rescalings the diagonal elements are only {-1,0,1} and furthermore the number of +1s (called the positive index of inertia n_+) and the number of 0s (n_0)and the number of -1s (called the negative index of inertia n_-) is invariant no matter what recoordinatization is chosen. Clearly

$n_+ + n_0 + n_- = N-1$

The dimensionality of the measured space is given by
$n = n_+ + n_-$

The signature of the measured space is given by
$s = n_+ - n_-$

Knowing the dimensionality, the signature and N is equivalent to knowing n_+, n_0 and n_- which the Coordinatization algorithm provides. The Corrdinatizator algorithm automatically determines these quantities by noting the sign of the quantities under the square root. The presentation of the algorithm below explicitly indicates these quantities.

4. Coding the Algorithm

We have an outer loop for k = 1 to N. In each loop we determine k constants which are the non-zero components of P_k. Computing the next component depends on the values of all previously computed components. Essentially we are constructing a new output NxN matrix called x from a given input NxN matrix called d. The input matrix cannot have negative values, must be symmetric and its principal diagonal should consist only of zeros. The output matrix is lower-triangular so that it also has zeros on its principal diagonal. In both cases the matrix indices run from 1 to N. So the algorithm looks like this:

```
Inputs: The NxN distance matrix d[i,j]
Outputs: The NxN matrix of point components x[i,j].
Initialize all x[i,j] = 0 -:
for i = 1 to N
        for j = 1 to N
                x[i,j] = 0
PositiveIndex = 0
NegativeIndex = 0
for k = 1 to N
        l = k-1
        for i = 1 to l-1
                sum = Sq(d[i,k]) - Sq(d[i+1,k])
                for j = 1 to i-1
                        sum += - Sq(x[k,j] - x[i,j])
                        sum += + Sq(x[k,j] - x[i+1,j])
                sum += Sq(x[i+1,i])
                x[k,i] = sum/2/x[i+1,i]
```

```
sum = Sq(d[l,k])
for i = 1 to l-1
        sum = sum - Sq(x[l,i]-x[k,i])
if sum > 0
        PositiveIndex++
        x[l,k] = SqRoot(sum)
if sum < 0
        NegativeIndex++
        x[l,k] = SqRoot(-sum)
```

5. Curvature, Dimensionality and Degeneracy

We will start by assuming that the N sites define a space of dimensionality n = N-1. If for any k in the algorithm we find x_{k+1k} is zero or close to it then we will reduce the dimensionality n of the data space by 1. The Coordinatizator algorithm does not search for curvature in the space nor the dimensionality of the embedding space, so the defined space be either a flat n-dimensional Euclidean space or a flat n-dimensional pseudo-Euclidean space. When there is degeneracy one has to check whether the negative is the right solution i.e. it gives the right distances to the other points before it according to the distance matrix as described in the Map Maker's algorithm.

6. Conclusions

The Coordinatizator algorithm presented in this paper maps sites to an n-dimensional pseudo-Euclidean space preserving all distances of the given distance matrix. The algorithm returns the positive index of inertia and the negative index of inertia of the pseudo-metric of the pseudo-Euclidean space. If the negative index is zero then the space is pure Euclidean and the triangle inequality [Wik13c] is valid for all triplets of points within it. If the negative index is non-zero then the triangle inequality is reversed but only for the subspace of points spanned by the imaginary axial directions. Once this algorithm is applied to defined space data projection mappings can be applied to view the n-dimensional data on a flat screen. The Coordinatizator algorithm effectively maps one NxN lower triangular matrix (d_{ij}) to another NxN lower triangular matrix (x_{ij}) and this would allow us to reduce memory needs by storing data in a single square NxN real matrix.

References

[1]. [Fer57] Ferrar W, "Algebra – A Textbook of Determinants, Matrices and Algebraic Forms", Oxford University Press, 2nd edition, 1957, p 154.

[2]. [LSB77] Lee R, Slagle J & Blum H, "A Triangulation Method for the Sequential Mapping of Points from N-Space to Two-Space", 1977, IEEE Transaction on Computers, pp 288-292.

[3]. [PC92] Potter L & Chiang D, "Distance Matrices and Modified Cyclic Projections for Molecular Conformation", Acoustics, Speech and Signal Processing 1992, Vol 4, pp 173-176.

[4]. [PE*09] Pham T, Eisenblatter U, Golledge J, Baune B & Berger K, "Segmentation of Medical Images Using Geo-theoretic Distance Matrix in Fuzzy Clustering", Image Processing (ICIP) 2009, pp 3369-3372.

[5]. [Ran13] Rankin J, “Map Making From Tables”, International Journal of Computer Graphics and Animation, vol 3, no 2, 2013, pp 11-19

[6]. [RS00] Rowels S & Saul L, “Nonlinear Dimensionality Reduction by Locally Linear Embedding”, Science, Vol 290, December 2000, pp. 232-233.

[7]. [Sle10] Sledge I, “A Projection Transform for non-Euclidean Relational Clustering”, Neural Networks (IJCNN) 2010, pp 1-8.

[8]. [TSL00] Tenembaum J, de Silva V & Langford J, “A Global Geometric Framework for Nonlinear Diemsionality Reduction”, Science, Vol 290, December 2000, pp. 232-233.

[9]. [TT*12] Tzagkarakis G, Tsagkatakis G, Starck J & Tsakalides P, “Compressive Video Classification in a Low-dimensional Manifold with Learned Distance Matrix”, 20th European Signal Processing Conference [EUSIPCO 2012], 2012, pp 155-159.

[10]. [Wik13a] “PseudoEuclidean spaces” Wikipedia, 2011

[11]. http://en.wikipedia.org/wiki/Pseudo-Euclidean_space

[12]. [Wik13b] “Sylvester’s Law of Inertia, Wikipedia, 2013

[13]. http://en.wikipedia.org/wiki/Sylvester's_law_of_inertia

[14]. [Wik13c] “Triangle Inequality”, Wikepedia

[15]. http://en.wikipedia.org/wiki/Triangle_inequality

[16]. [YR13] Youldash M & Rankin J, “Evaluation of a Geometric Approach to the Coordinatization of Measured Spaces”, 8th International Conference on Information Technology and Applications (ICITA 2013), Sydney, Australia July 2013.

[17]. [ZWZ08] Zuo W, Wang K & Zhang D, “Assembled Matrix Distance Metric for 2DPCA-based Face and Palmprint Recognition”, Machine Learning and Cybernetics 2008, Vol 8, pp4870-4875.