Clustering with an N-Dimensional Extension of Gielis Superformula

AIKED 2008
Cambridge, U.K.

Angel Kuri

akuri@itam.mx

Instituto Tecnológico Autónomo de México

http://www.itam.mx/es/index.php

February 22, 2008

Clustering

Clustering can be considered the most important unsupervised learning problem; so, as every other problem of this kind, it deals with finding a structure in a collection of unlabeled data.

A loose definition of clustering could be “the process of organizing objects into groups whose members are similar in some way”.

A cluster is therefore a collection of objects which are “similar” between them and are “dissimilar” to the objects belonging to other clusters

Clustering

The clustering problem has been addressed in many contexts and by researchers in many disciplines; this reflects its broad appeal and usefulness as one of the steps of exploratory data analysis.

Clustering: Associated Issues

Regardless of the approach selected to tackle this problem, there are 4 basic associated issues:

1. How many clusters are there?
2. Which is the best way to determine similarities? What distance to use?
3. Which is the best algorithm?
4. How representative are the clusters?

How Many Clusters are There?

- This problem has no unique solution. It depends on the type of data, the distance used, etc.
- “Elbow criterion”.
- Extended penalized competitive learning
- GAP statistic
- Bayesian information criterion
- Evolutionary algorithms (entropy based fitness)
- Parsimony and homogeneity

How representative are the clusters?

- There are several measures of “quality”
  - Dunn and Dunn-like validity indices
  - Davies-Bouldin validity index
  - SD validity index
  - S_Dbw validity index
  - DBSCAN algorithm (density based)
  - DENCLUE algorithm (density based)
  - Variance of the nearest neighbor (VNN)

A measure of goodness?

If the validity indices are a good measure of success it would seem natural to try to directly find the clusters from such indices.

That is the approach of DBSCAN, for instance.

But even then, some clusters are poorly evaluated if they present irregular shapes.

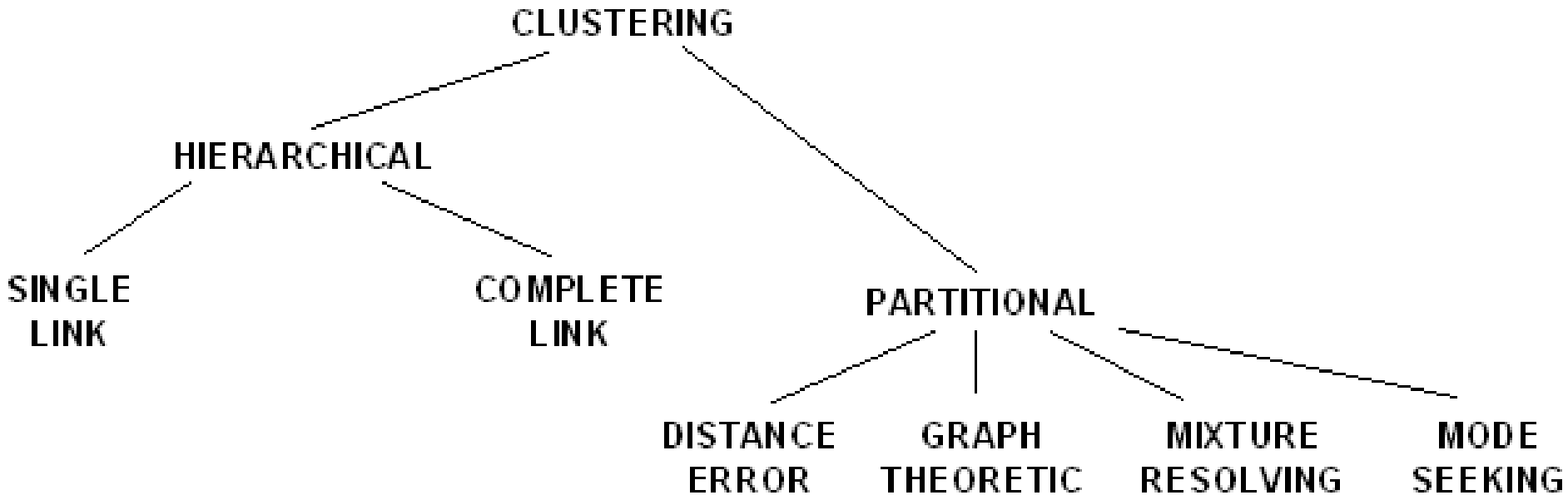
Purportedly, the VNN index is superior to others. Their use as an explicit optimization distance is, however, difficult to implement (but wait a bit).

What distance to use?
Which is the best algorithm?

- This matters are central to our study.
- We discuss them in what follows

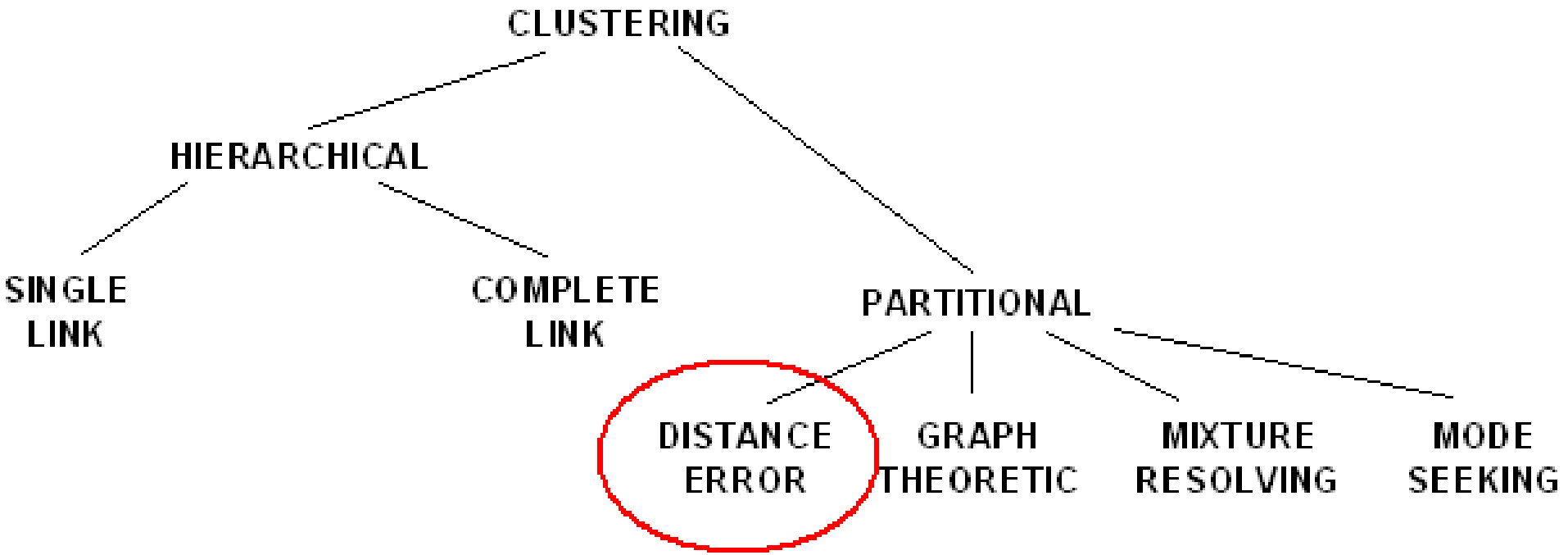
Taxonomy

A taxonomy of clustering approaches is the following.



Taxonomy

We shall focus in clustering with the aim of minimizing some kind of error measure.



Clustering

- The problem is to find an appropriate set of clusters assuming their number is given and the data has be chosen such that it is representative of the population and, therefore, the clustering derived from the sample applies with generality to the population from which the sample has been collected.

The Goals of Clustering

For instance, we could be interested in finding representatives for homogeneous groups (data reduction), in finding “natural clusters” and describe their unknown properties (“natural” data types), in finding useful and suitable groupings (“useful” data classes) or in finding unusual data objects (outlier detection).

The Goals of Clustering

So, the goal of clustering is to determine the intrinsic grouping in a set of unlabeled data. But how to decide what constitutes a good clustering? It can be shown that there is no absolute “best” criterion which would be independent of the final aim of the clustering. Consequently, it is the user which must supply this criterion, in such a way that the result of the clustering will suit their needs.

Distance Measure

An important component of a clustering algorithm is the distance measure between data points.

If the components of the data instance vectors are all in the same physical units then it is possible that the simple Euclidean distance metric is sufficient to successfully group similar data instances.

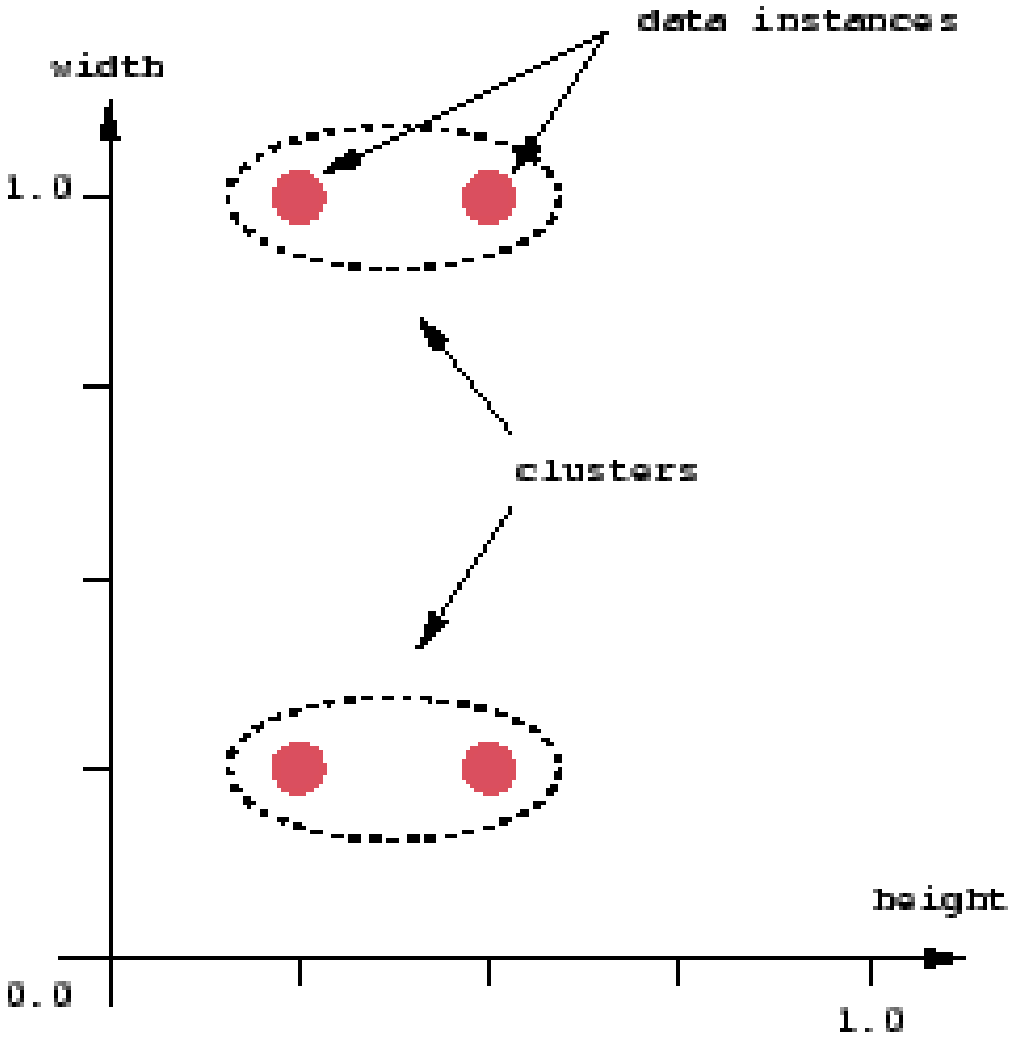
However, even in this case the Euclidean distance can sometimes be misleading.

Distance Measure

The next figure illustrates this with an example of the width and height measurements of an object.

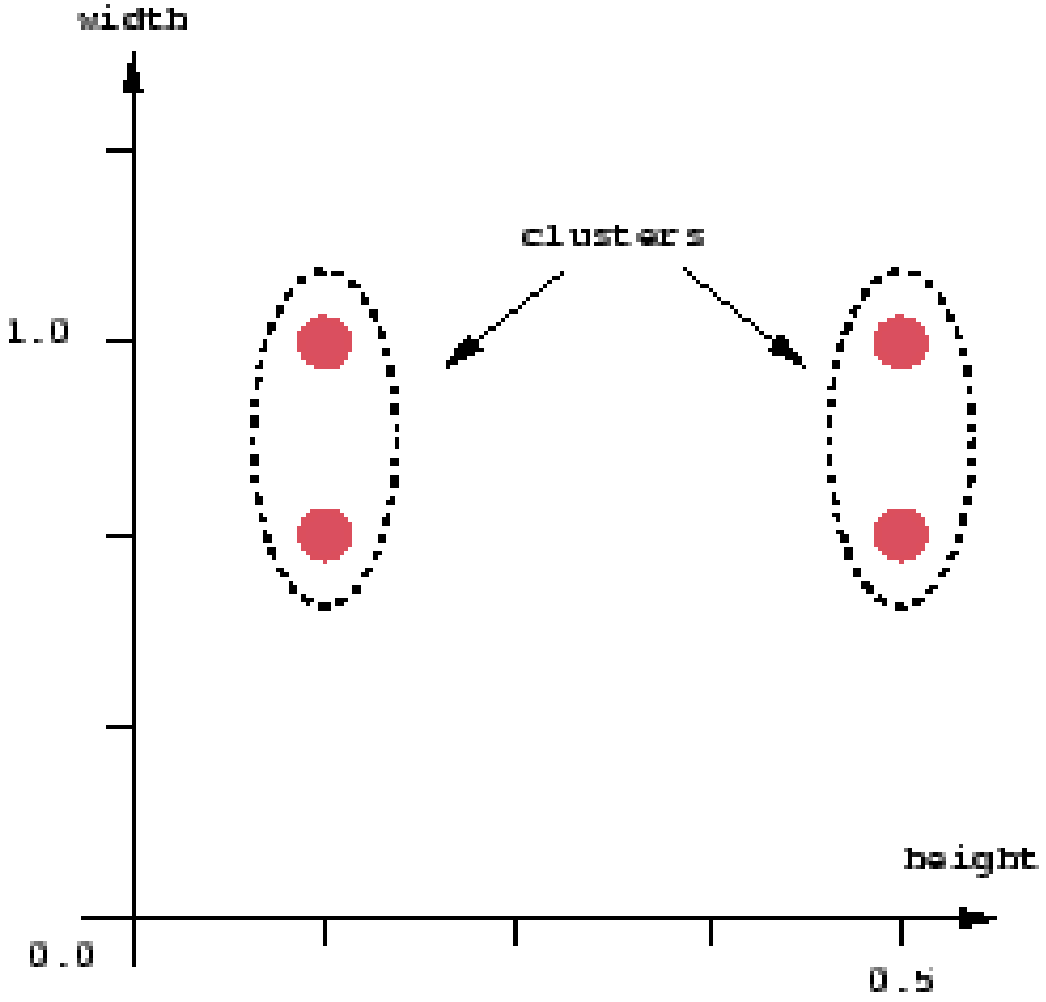
Despite both measurements being taken in the same physical units, an informed decision has to be made as to the relative scaling. As the figure shows, different scalings can lead to different clusterings

Distance Measure



before scaling

Distance Measure



Distance Measure

Notice however that this is not only a graphic issue: the problem arises from the mathematical formula used to combine the distances between the single components of the data feature vectors into a unique distance measure that can be used for clustering purposes: different formulas lead to different clusterings.

Euclidean Distance

Def. The euclidian distance between two points $x = (x_1, \ldots, x_p)^t$ and $y = (y_1, \ldots, y_p)^t$ in the p-dimensional space $\mathbb{R}^p$ is defined as

$$d_E(x, y) = \sqrt{(x_1 - y_1)^2 + \cdots + (x_p - y_p)^2} = \sqrt{(x - y)^t (x - y)}$$

and $d_E(x, 0) = \|x\|_2 = \sqrt{x_1^2 + \cdots + x_p^2} = \sqrt{x^t x}$ is the euclidian norm of x.

Euclidean Distance

It follows, immediately, that all points with the same distance of the origin $\|X\|_2 = c$ satisfy $X_1^2 + \ldots + X_p^2 = c^2$ which is the equation of a spheroid.

This issue is central to the considerations leading to the approach we have taken in our study.

Mahalanobis Distance

We would, alternatively, like to have a distance that for each of the components (the variables) takes the variability of that component into account when determining its distance from the center. Components with high variability should receive less weight than components with low variability. This can be obtained by rescaling the components.

Mahalanobis Distance

Denote

$$u = (\frac{x_1}{s_1}, \ldots, \frac{x_p}{s_p}) \text{ and } v = (\frac{y_1}{s_1}, \ldots, \frac{y_p}{s_p})$$

then define the distance between x and y as

$$d(x,y) = d_E(u,v) = \sqrt{(\frac{x_1 - y_1}{s_1})^2 + \cdots + (\frac{x_p - y_p}{s_p})^2} = \sqrt{(x-y)^t D^{-1}(x-y)}$$

where $D = \text{diag}(s_1^2, \ldots, s_p^2)$. Now the norm of x equals

$$\|x\| = d(x,0) = d_E(u,0) = \|u\|_2 = \sqrt{(\frac{x_1}{s_1})^2 + \cdots + (\frac{x_p}{s_p})^2} = \sqrt{x^t D^{-1} x}$$

Mahalanobis Distance

and all points with the same distance of the origin $\|x\| = c$ satisfy

$$(\frac{x_1}{s_1})^2 + \cdots + (\frac{x_p}{s_p})^2 = c^2$$

which is the equation of an ellipsoid centered at the origin with principal axes equal to the coordinate axes.

Mahalanobis Distance

Finally, we also want to take the correlation between variables into account when computing statistical distances. Correlation means that there are associations between the variables. Therefore, we want the axes of the ellipsoid to reflect this correlation. We allow the axes of the ellipsoid at constant distance to rotate. This yields the following general form for the distance of two points.

Mahalanobis Distance

Def. The statistical distance or Mahalanobis distance between two points $x = (x_1, \ldots, x_p)^t$ and $y = (y_1, \ldots, y_p)^t$ in the p-dimensional space $\mathbb{R}^p$ is defined as

$$d_S(x, y) = \sqrt{(x - y)^t S^{-1} (x - y)}$$

and $d_S(x, 0) = \|x\|_S = \sqrt{x^t S^{-1} x}$ is the norm of x.

Mahalanobis Distance

$\mathbf{y} = (y_1, \ldots, y_n)$

$\mathbf{x} = (x_1, \ldots, x_n)$

$$d = \sqrt{\sum_{x=1}^{N} \sum_{y=1}^{N} b_{xy}(x_i - x_j)(y_i - y_j)}$$

$$\bar{x} = \frac{1}{K} \sum_{k=1}^{K} x_k; \quad \bar{y} = \frac{1}{K} \sum_{k=1}^{K} y_k$$

$$a_{xy} = \frac{1}{K} \sum_{k=1}^{K} (x_k - \bar{x})(y_k - \bar{y})$$

a_{xy} is the covariance between x and y, and

b_{xy} is the corresponding element of its inverse

Mahalanobis Distance

```
 a=new double[V][V];
cov(a,dato); //Calculate covariance
b=new double[V][V];
if (!Inverse(a,b,V)){
    System.out.println("Non-invertible");
    return false;
}//endIf
```

Mahalanobis Distance

CALCULATE THE COVARIANCE MATRIX

```
for (int i=0;i<V;i++){
   for (int j=0;j<V;j++){
      cov[i][j]=0;
      for (int k=0;k<U;k++){
          cov[i][j]=cov[i][j]+(X[k][i]-
          mu[i])*(X[k][j]-mu[j]);
      }cov[i][j]=cov[i][j]/U;
   }//endFor
}//endFor
```

Mahalanobis Distance

Intuitive explanation

Consider the problem of estimating the probability that a test point in N-dimensional Euclidean space belongs to a set, where we are given sample points that definitely belong to that set. Our first step would be to find the average or center of mass of the sample points. Intuitively, the closer the point in question is to this center of mass, the more likely it is to belong to the set.

Mahalanobis Distance

However, we also need to know how large the set is. The simplistic approach is to estimate the standard deviation of the distances of the sample points from the center of mass. If the distance between the test point and the center of mass is less than one standard deviation, then we conclude that it is highly probable that the test point belongs to the set. The further away it is, the more likely that the test point should not be classified as belonging to the set.

Mahalanobis Distance

This intuitive approach can be made quantitative by defining the normalized distance between the test point and the set to be $\frac{x - \mu}{\sigma}$. By plugging this into the normal distribution we get the probability of the test point belonging to the set.

Mahalanobis Distance

The drawback of the above approach was that we assumed that the sample points are distributed about the center of mass in a spherical manner. Were the distribution to be decidedly non-spherical, for instance ellipsoidal, then we would expect the probability of the test point belonging to the set to depend not only on the distance from the center of mass, but also on the direction. In those directions where the ellipsoid has a short axis the test point must be closer, while in those where the axis is long the test point can be further away from the center.

Mahalanobis Distance

Putting this on a mathematical basis, the ellipsoid that best represents the set's probability distribution can be estimated by building the covariance matrix of the samples. The Mahalanobis distance is simply the distance of the test point from the center of mass divided by the width of the ellipsoid in the direction of the test point.

Mahalanobis Distance

This measure (and many others) have been invented to try to circumvent the inherent hyper-spheroidicity of the spaces resulting from classical distance measures

Some Clustering Methods

There are many approaches to minimum distance clustering. Some of them:

- K-means
- Fuzzy C-means
- Hierarchical dynamic clustering
- Self-Organizing Maps
- Genetic algorithmic clustering

Some Clustering Methods

We now briefly describe a few of the attempts at practically solving the problem of clustering a set of arbitrary data.

In the statistically based methods the distance is usually taken to be Euclidean. But this need not be the case, as already discussed.

K-Means

K-means (MacQueen, 1967) is one of the simplest unsupervised learning algorithms that solve the well known clustering problem. The procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume k clusters) fixed a priori. The main idea is to define k centroids, one for each cluster. These centroids should be placed in a cunning way because of different location causes different result. So, the better choice is to place them as much as possible far away from each other.

K-Means

The next step is to take each point belonging to a given data set and associate it to the nearest centroid. When no point is pending, the first step is completed and an early groupage is done. At this point we need to re-calculate k new centroids as barycenters of the clusters resulting from the previous step. After we have these k new centroids, a new binding has to be done between the same data set points and the nearest new centroid.

A loop has been generated. As a result of this loop we may notice that the k centroids change their location step by step until no more changes are done. In other words centroids do not move any more.

K-Means

This algorithm aims at minimizing an *objective function*, in this case a squared error function: the objective function

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} \left\| x_i^{(j)} - c_j \right\|^2$$

where $\left\| x_i^{(j)} - c_j \right\|^2$ is a chosen distance measure between a data point $x_i^{(j)}$ and the cluster center c_j, is an indicator of the distance of the ***n*** data points from their respective cluster centres.

Fuzzy C-Means

Fuzzy c-means (FCM) is a method of clustering which allows one piece of data to belong to two or more clusters. This method is frequently used in pattern recognition. It is based on minimization of the following objective function:

$$J_m = \sum_{i=1}^{N} \sum_{j=1}^{C} u_{ij}^{m} \left\| x_i - c_j \right\|^2, \quad 1 \le m < \infty$$

Fuzzy C-Means

where m is any real number greater than 1, u_{ij} is the degree of membership of x_i in the cluster j, x_i is the ith of d-dimensional measured data, c_j is the d-dimension center of the cluster, and ||*|| is any norm expressing the similarity between any measured data and the center

Fuzzy C-Means

Fuzzy partitioning is carried out through an iterative optimization of the objective function shown above, with the update of membership u_{ij} and the cluster centers c_i by:

$$u_{ij} = \frac{1}{\sum_{k=1}^{C} \left(\frac{\|x_i - c_j\|}{\|x_i - c_k\|} \right)^{\frac{2}{m-1}}}, \qquad c_j = \frac{\sum_{i=1}^{N} u_{ij}^{m} \cdot x_i}{\sum_{i=1}^{N} u_{ij}^{m}}$$

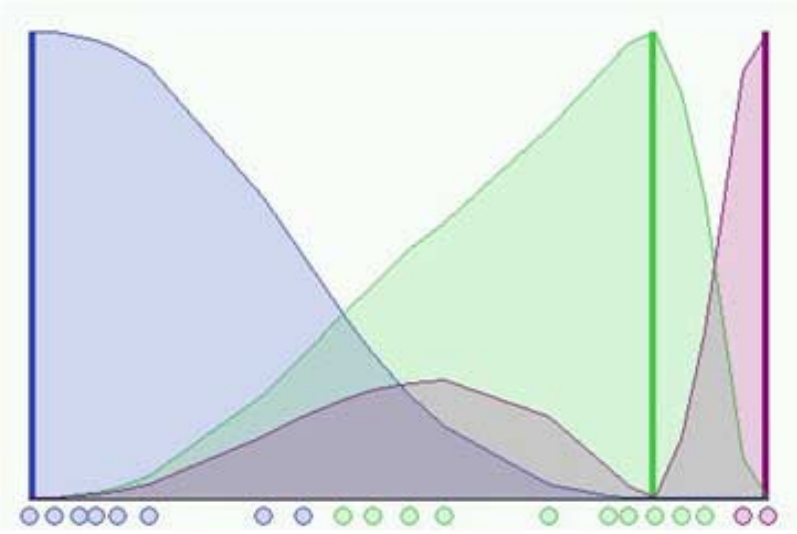
Fuzzy C-Means

The iteration will stop when $\max_{ij}\left\{\left|u_{ij}^{(k+1)} - u_{ij}^{(k)}\right|\right\} < \varepsilon$, where ε is a termination criterion between 0 and 1, whereas *k* are the iteration steps. This procedure converges to a local minimum or a saddle point of J_m.

Fuzzy C-Means

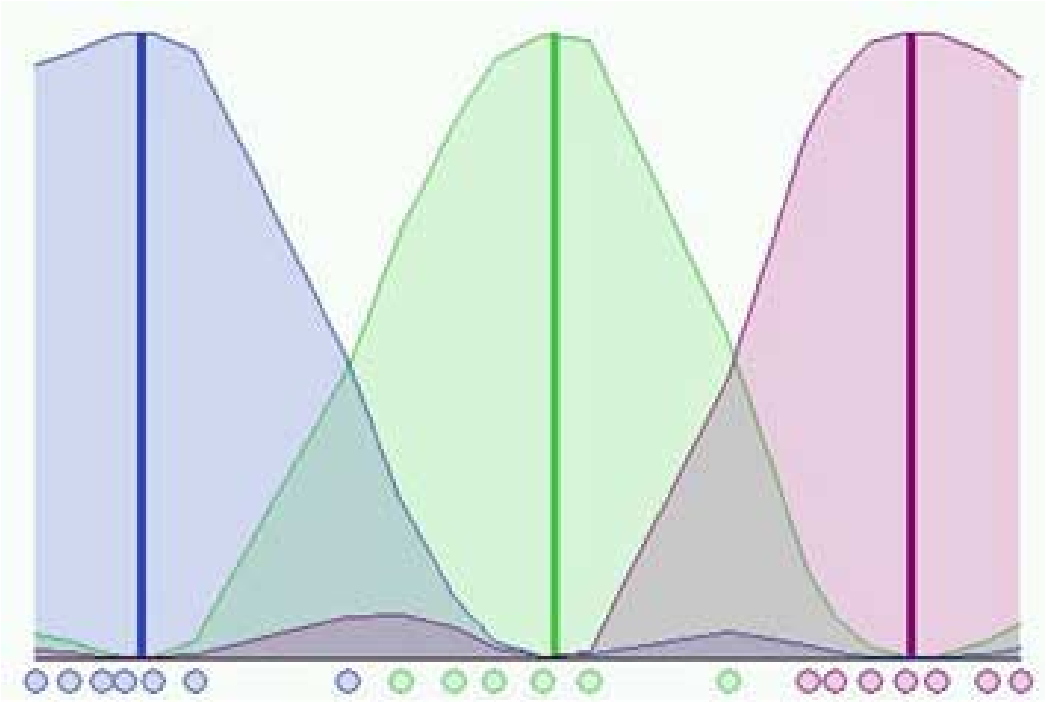
Consider the simple case of a one dimensional application of the FCM. Twenty data and three clusters are used to initialize the algorithm and to compute the U matrix. The figures below show the membership value for each datum and for each cluster. The color of the data is that of the nearest cluster according to the membership function.

Fuzzy C-Means



The picture shows the initial condition where the fuzzy distribution depends on the particular position of the clusters. No step is performed yet so that clusters are not identified very well. Now we can run the algorithm until the stop condition is verified.

Fuzzy C-Means



The next figure shows the final condition reached at the 8th step with $m = 2$ and $\varepsilon = 0.3$:

Hierarchical Clustering

Given a set of N items to be clustered, and an N*N distance (or similarity) matrix, the basic process of hierarchical clustering is this:

1. **Start by assigning each item to a cluster, so that if you have N items, you now have N clusters, each containing just one item. Let the distances (similarities) between the clusters the same as the distances (similarities) between the items they contain.**
2. **Find the closest (most similar) pair of clusters and merge them into a single cluster, so that now you have one cluster less.**

Hierarchical Clustering

3. Compute distances (similarities) between the new cluster and each of the old clusters.

4. Repeat steps 2 and 3 until all items are clustered into a single cluster of size N.

Of course there is no point in having all the N items grouped in a single cluster but, once you have got the complete hierarchical tree, if you want k clusters you just have to cut the k-1 longest links.

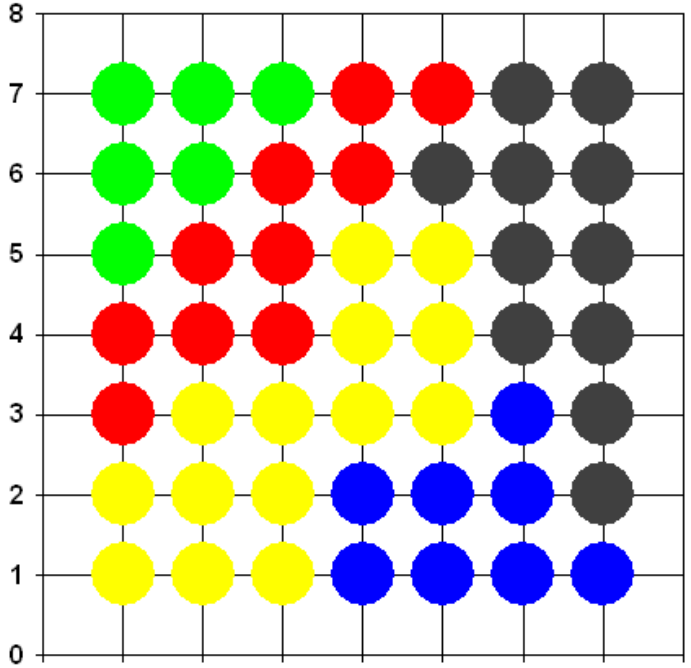
Self Organizing Maps

Competitive neural networks are often used to cluster input data.

SOMs have the desirable characteristic that neighboring neurons define regions of similarity

In the map every color represents a cluster (5 in this example).

Notice that neurons with similar colors (belonging to the same cluster) appear in neighboring regions of the map



Evolutionary Computation

Candidate solutions (an individual) to the clustering problem are initially randomly generated.

Three (typical) operators are used (selection, recombination and mutation).

These transform a set of individuals into a different set.

A fitness function evaluated for every individual determines its probability of survival for the next iteration.

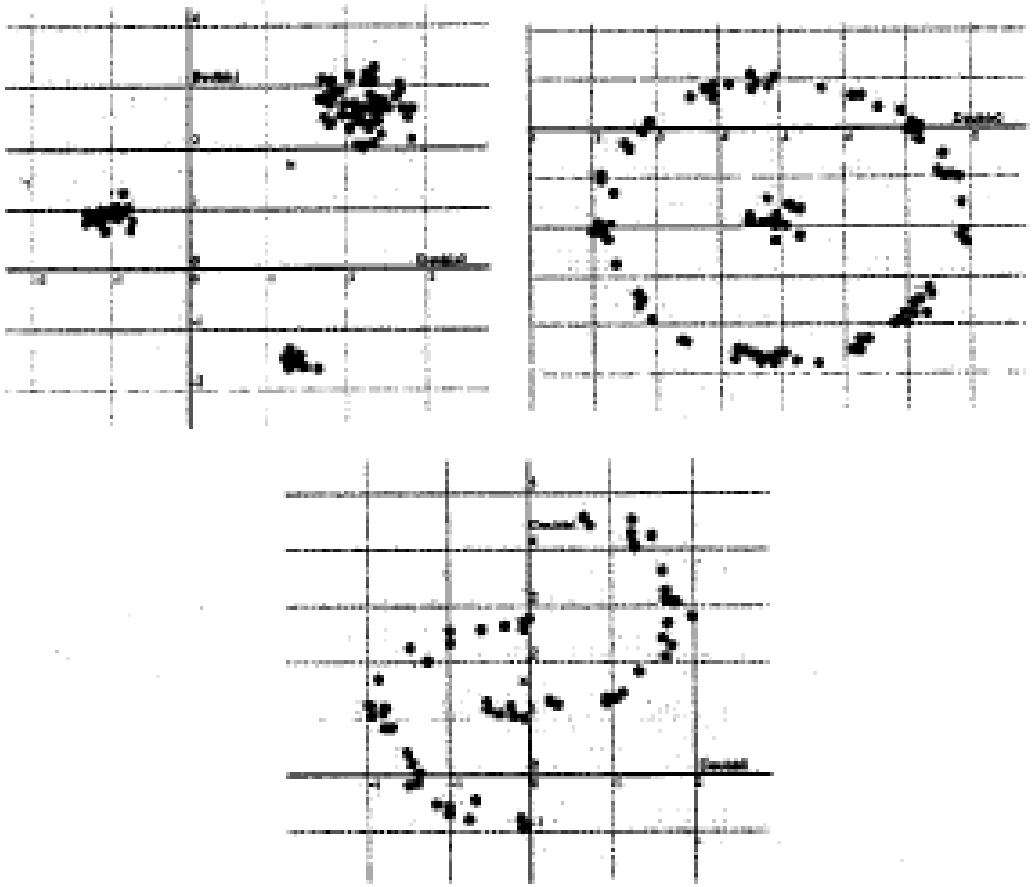
Typically, a distance is minimized as fitness function.

Limitations

All of the methods described so far have one thing in common: they depend on the distance as defined. In general any measure of distance ultimately results in structures which resemble an N-dimensional spheroid. Therefore, clusters with irregular forms are very difficult (at best) to detect.

Irregular Clusters

Clusters such as the ones shown are very difficult to detect



Further Motivation

The interest in finding irregular patterns in a set of data is manyfold.

For instance, in bioinformatics, one of the most urgent needs is to find similarities between known and unknown proteins.

Large genomic databases are now available.

But, in many cases, researchers are unable to find significant relationships between known and “new” proteins because the patterns are highly irregular.

Further Motivation

In data mining the relationships between relevant independent variables are difficult to uncover when the structure of the clusters is simplified.

In data compression, the state of the art methods rely on finding the patterns imbedded in the messages.

Further Motivation

In robotics the identification of patterns is closely associated to the problem of adaptive behavior.

In algorithmic information theory, the amount of information may be approximated by identifying the embedded patters.

…many others

Efficient optimization

A central issue behind all attempts to clusterize efficiently is the capability of finding the subset of the nearest vectors.

Such subset may be searched for in a variety of ways. A few of these have already been mentioned.

However, a general optimization tool must take into consideration that, in general, there is a non-linear relation between the elements of the set which minimize the nearness.

Efficient optimization

During the last 15 years the possibility of applying computing tools with virtually zero cost has made usual the application of computer intensive methodologies.

One such methodology is called “Evolutionary Computation” and relies on the basic idea of refining a plausible solution iteratively by simulating the processes of natural evolution.

Evolutionary Computation

In truth, EC should be more properly seen as an optimization method which constitutes a directed stochastic search.

In particular Genetic Algorithms have enjoyed favor in the AI community because they are rugged and easy to implement.

In fact, their ruggedness has induced the application of some sort of Gas which may be shown not to be the most efficient.

Genetic Algorithms

These algorithms have been proven to:

a) Be independent of the initial population

b) Fully explore the search space when mutation is applied

c) Reach a global optimum when the best individual is retained (elitism)

d) Rapidly converge to near optimal solutions

However, their convergence is not bounded in time. Therefore we would like to make this convergence as efficient as possible.

The Simple Genetic Algorithm

In this regard the so-called “Simple” GA is hampered by:

a) Deceptive functions

b) Hitchhiking

To overcome the limitations of a Simple Genetic Algorithm we worked with *Vasconcelos GA*.

It displays:

a) Deterministic (i -> n-i+1) coupling

b) Annular crossover

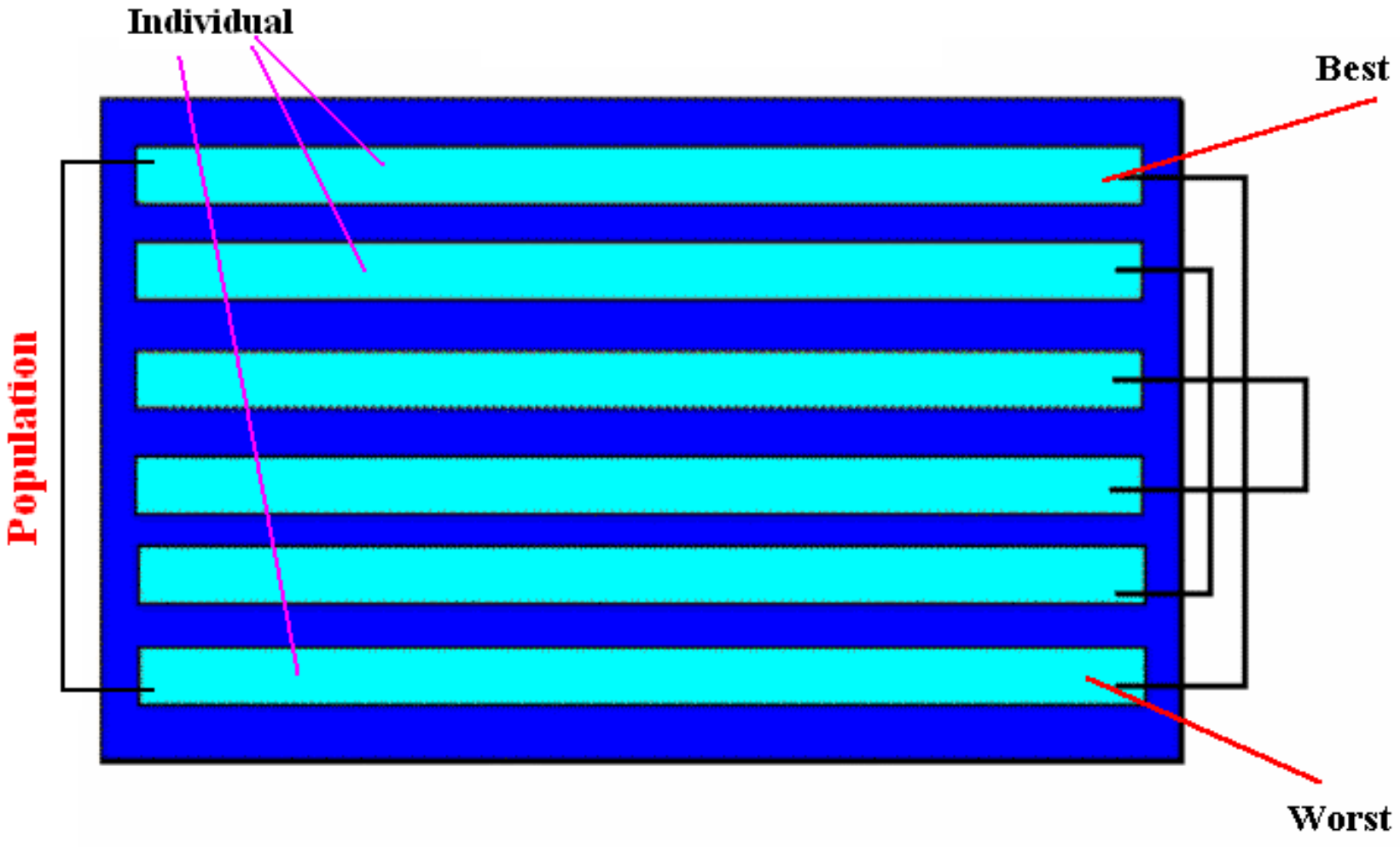
c) Uniform mutation

d) Full elitism

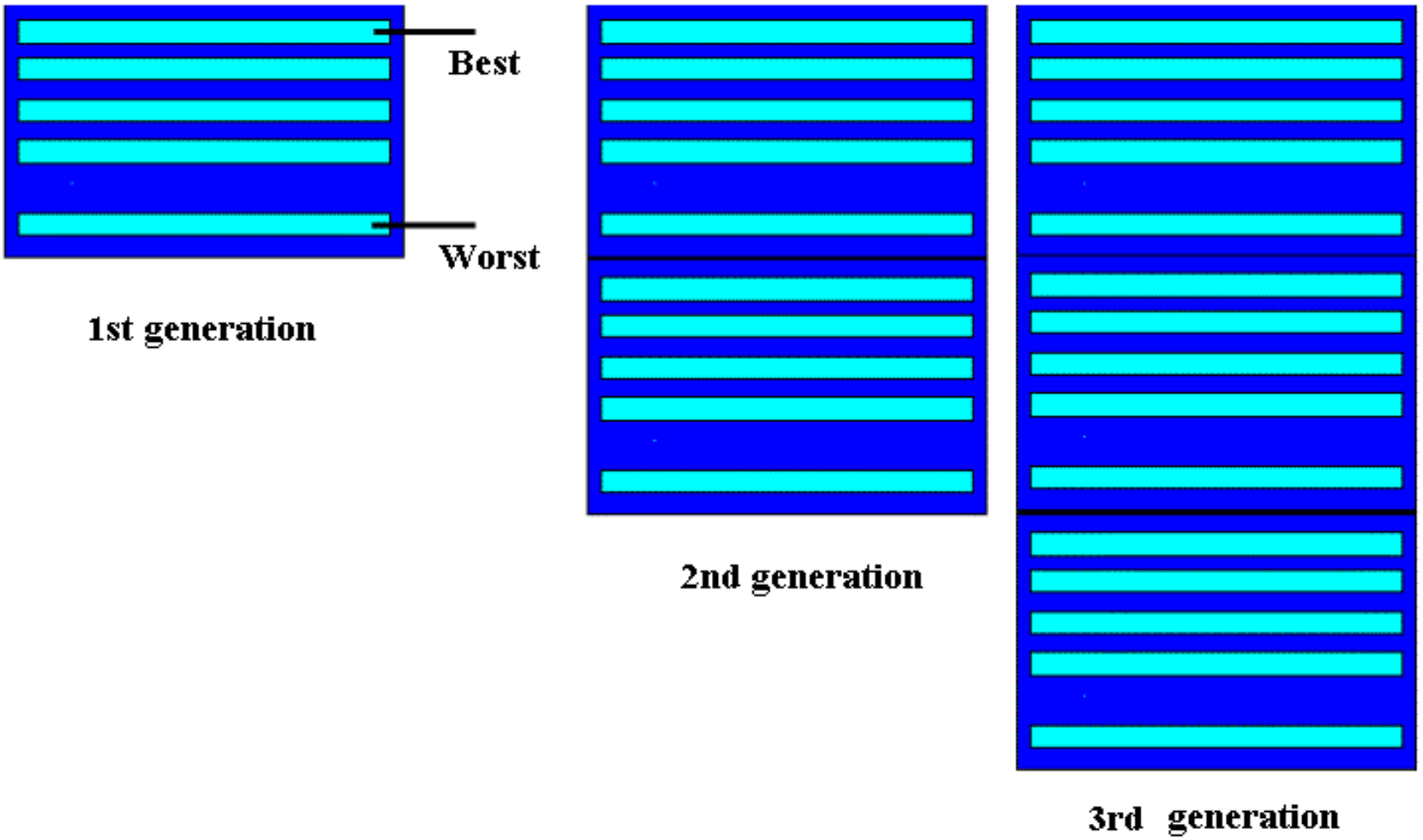
Vasconcelos' GA

A particular GA (called Vasconcelos GA, in honor of the Latin American philosopher José Vasconcelos who postulated the admixture of all races leading to the best possible race, the "Cosmic Race") has been proven to be relatively better than other genetic variations over a wide range of functions.

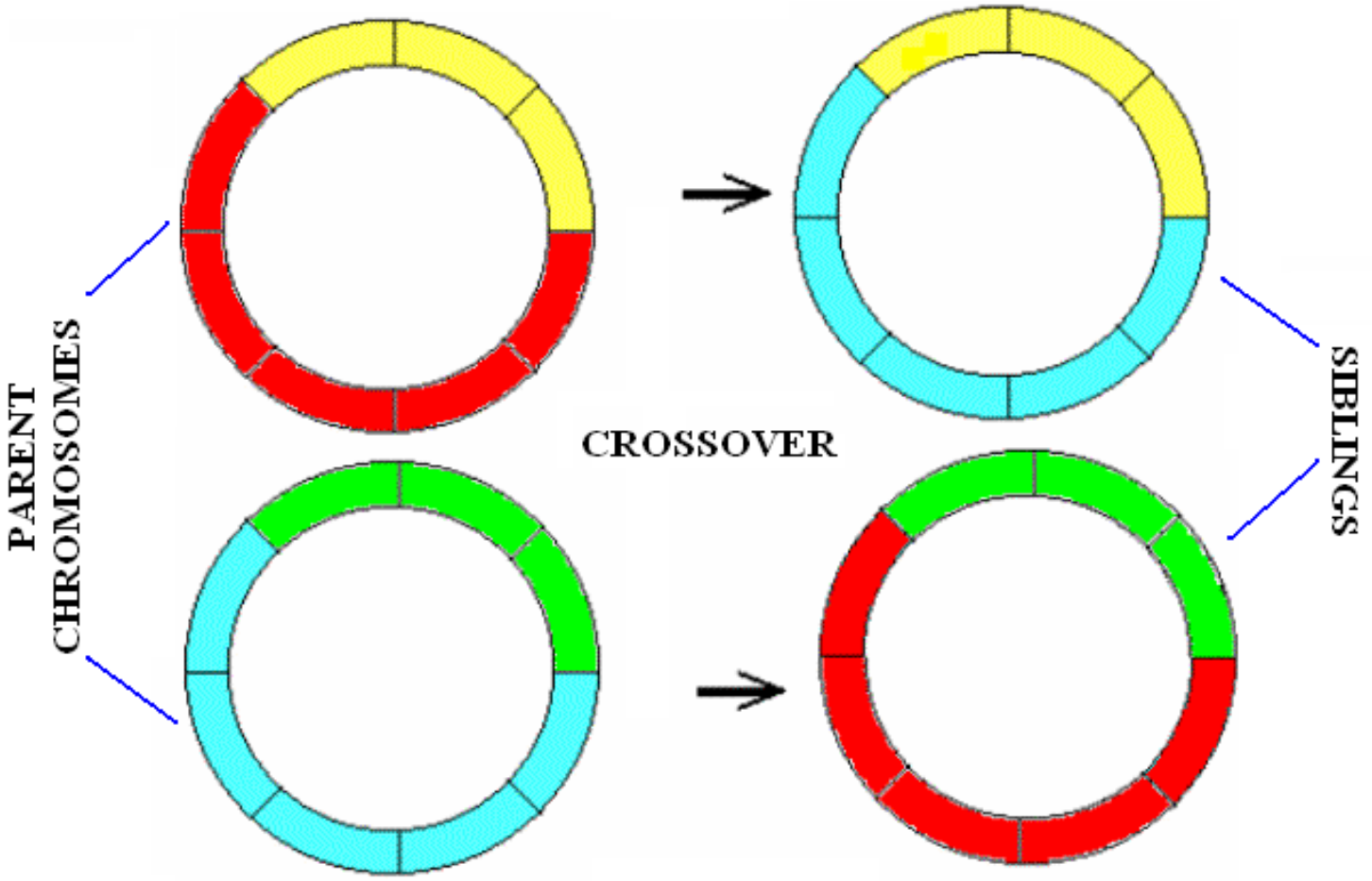
Vasconcelos Genetic Algorithm (Deterministic Coupling)



Vasconcelos Genetic Algorithm (Full Elitism)



Vasconcelos Genetic Algorithm (Annular Crossover)

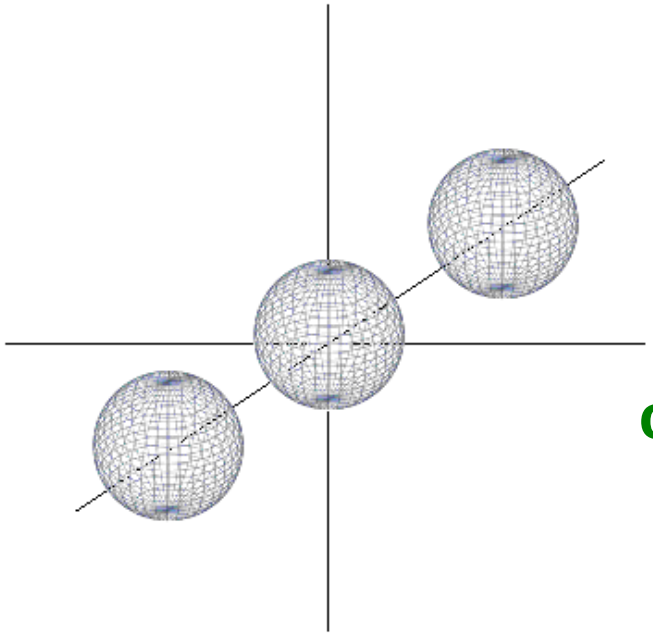


Testing traditional metrics using Vasconcelos GA

Since we know that VGA outperforms most other sorts of Gas we decided to use it as an optimization tool while retaining traditional distance measures.

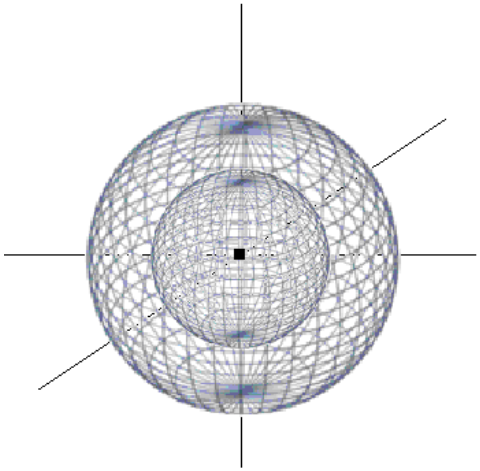
We decided to try its clustring capabilities versus a set of data gotten from a very simple figure.

Spheres

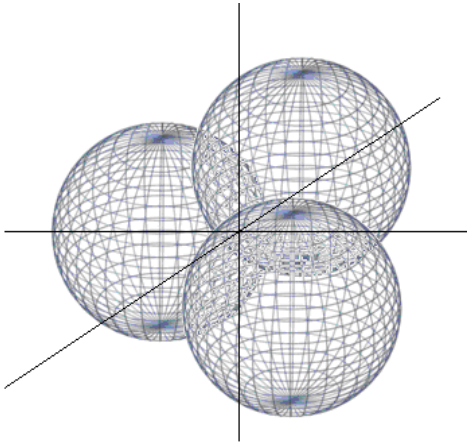


Disjoint

Three combinations of 3D Spheres



Concentric



Overlapping

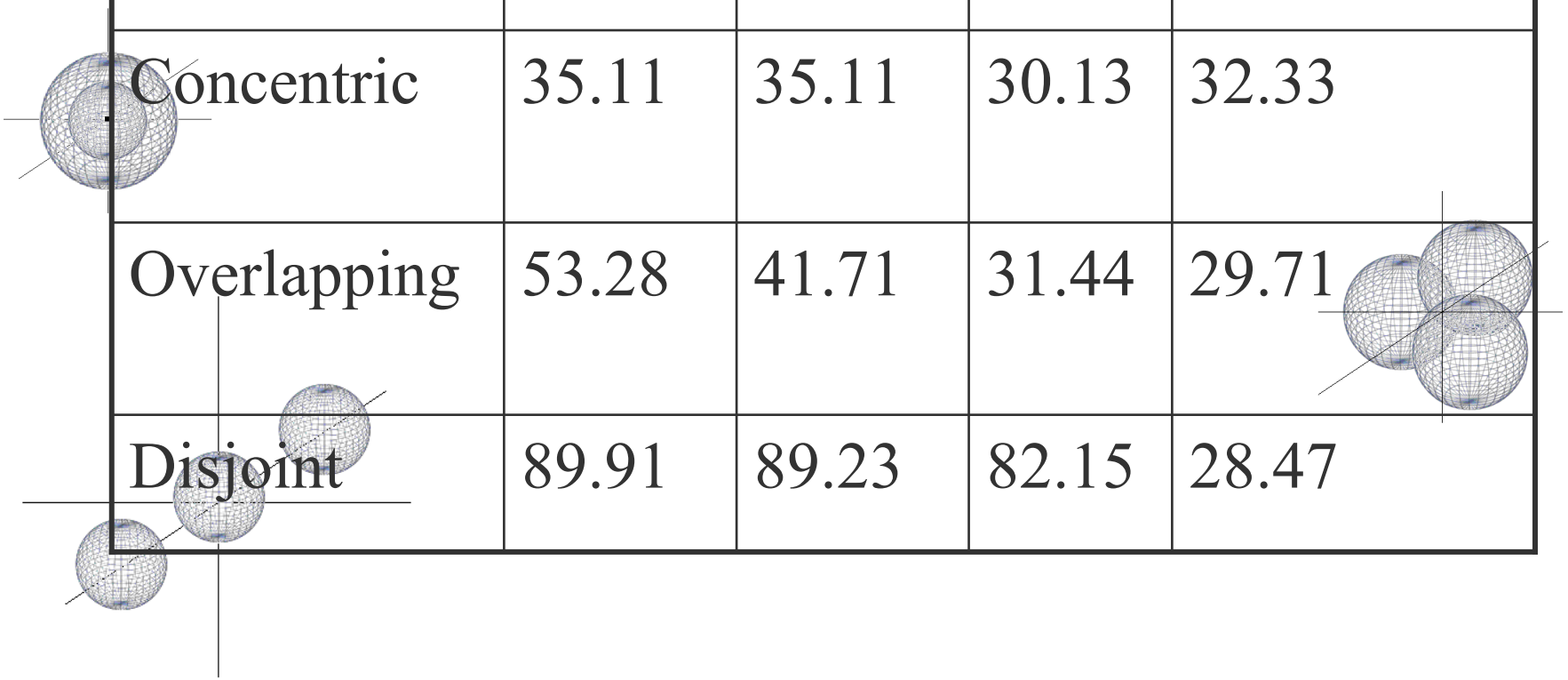
Testing for Spheres

In a set of experiments we tried out genetic clustering algorithms using different types of distances in an effort to see if the limitations mentioned could be overcome.

We tried L_1, L_2, L_∞ and Mahalanobis distances. They where used as the basic fitness fucntion for a Genetic Algorithm.

Clustering Spheres

	Metrics/Efficiency Percentage			
	L_1	L_2	L_∞	Mh
Concentric	35.11	35.11	30.13	32.33
Overlapping	53.28	41.71	31.44	29.71
Disjoint	89.91	89.23	82.15	28.47



Unsatisfactory Results

Clearly, the results are unsatisfactory.
Since GAs allow us to try to optimize some of the validity indices mentioned in the opening slides we used them as fitness functions.
Hence,we tested for the Dunn and VNN validity indices as a fitness functions.
Results were not satisfactory either.

Geometric determination of elements in a cluster

A novel alternative is to explore the *forms* of the cluster and find whether a set of data lies within such objects.

The idea is to express N-dimensional bodies in such a way that their geometric characteristics are mathematically expressed and then find the parameters which best fit into the arbitrary bodies.

Gielis Superformula

One possible alternative is to use:

$$r(\phi) = \left[\left| \frac{\cos\left(\frac{m\phi}{4}\right)}{a} \right|^{n_2} + \left| \frac{\sin\left(\frac{m\phi}{4}\right)}{b} \right|^{n_3} \right]^{-\frac{1}{n_1}}$$

In 3D we have:

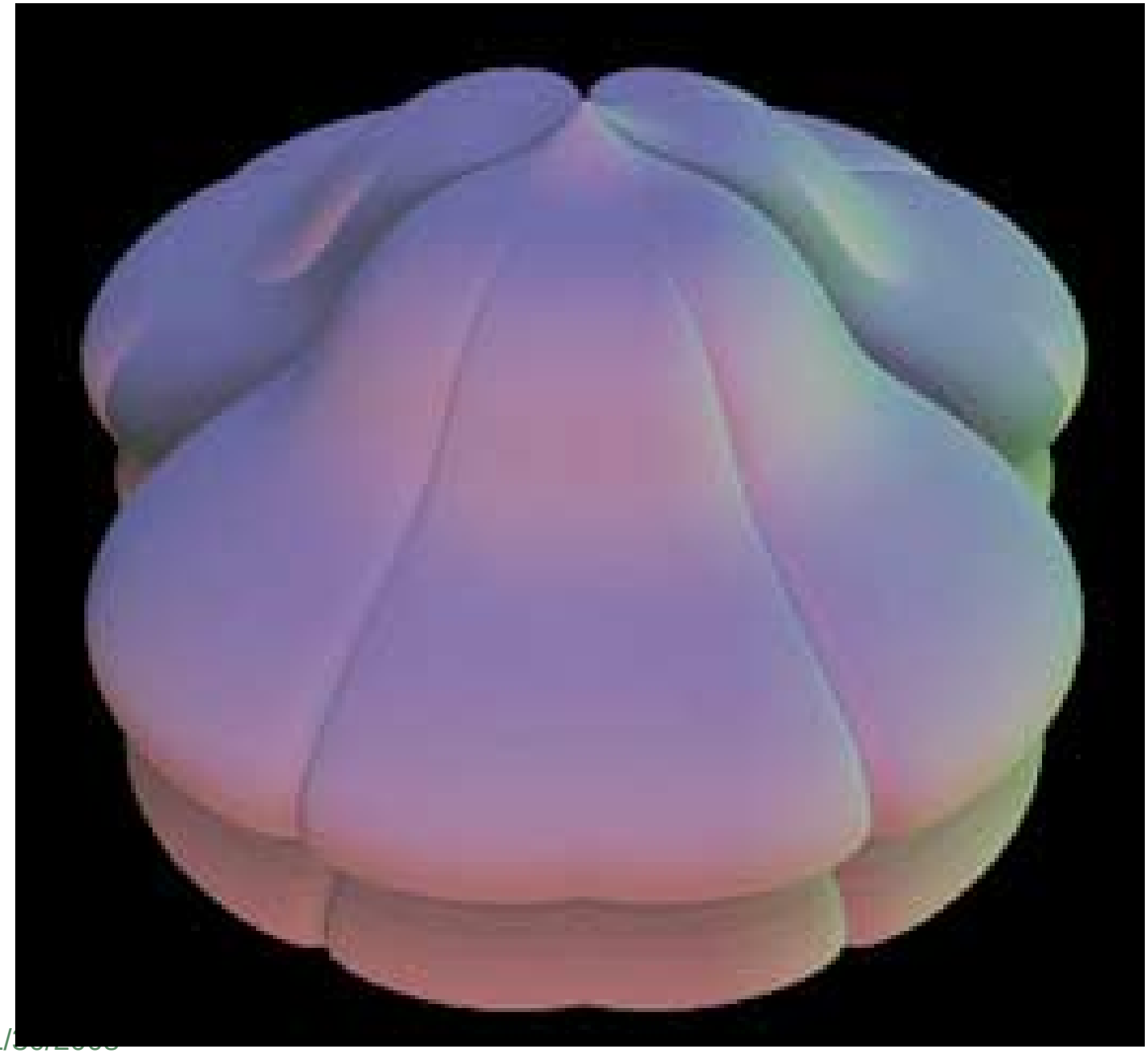
$$\begin{aligned} x &= r_1(\theta)\cos(\theta) r_2(\phi)\cos(\phi) \\ y &= r_1(\theta)\sin(\theta) r_2(\phi)\cos(\phi) \\ z &= r_2(\phi)\sin(\phi) \end{aligned}$$

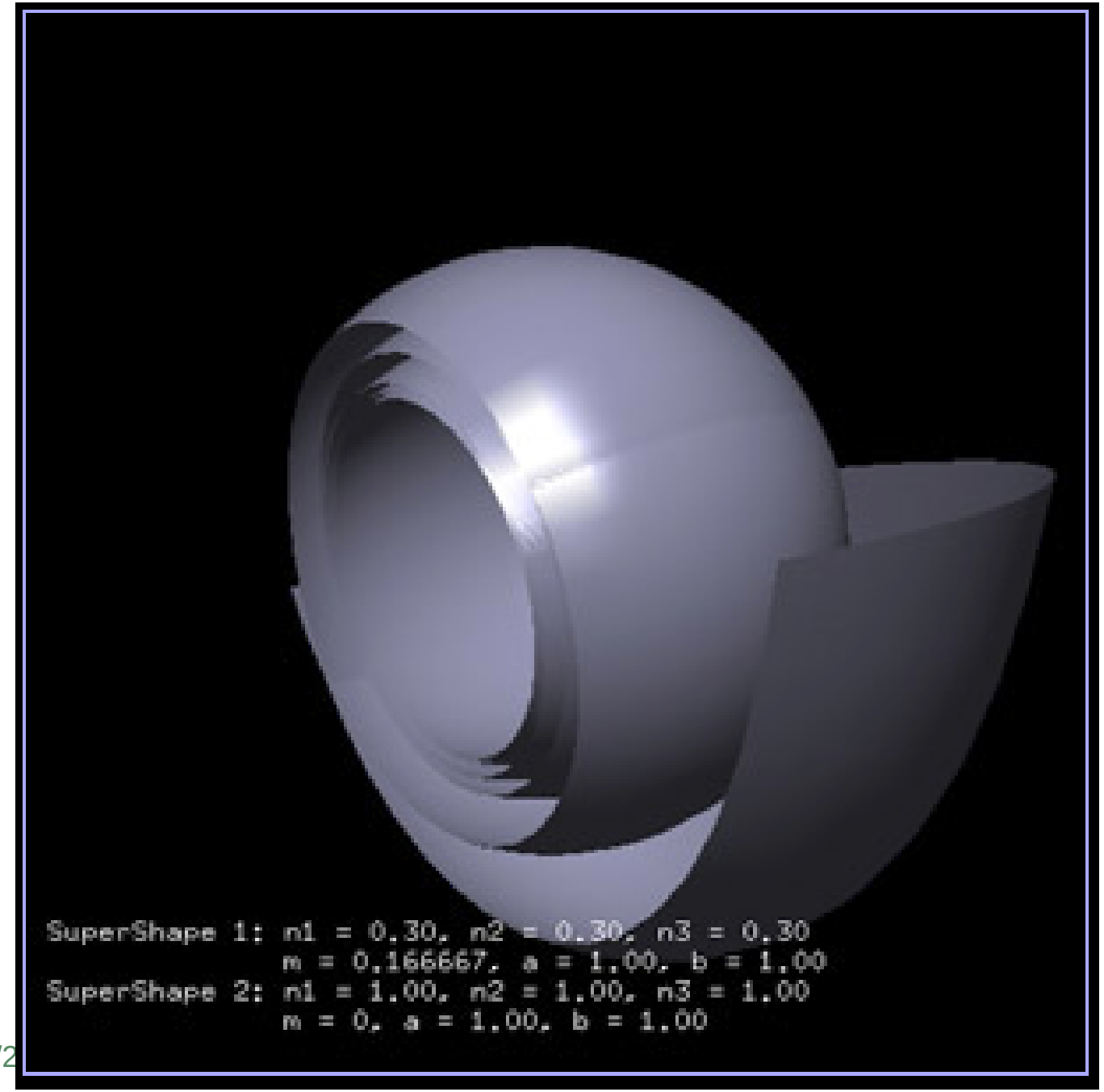
Gielis Superformula (SF)

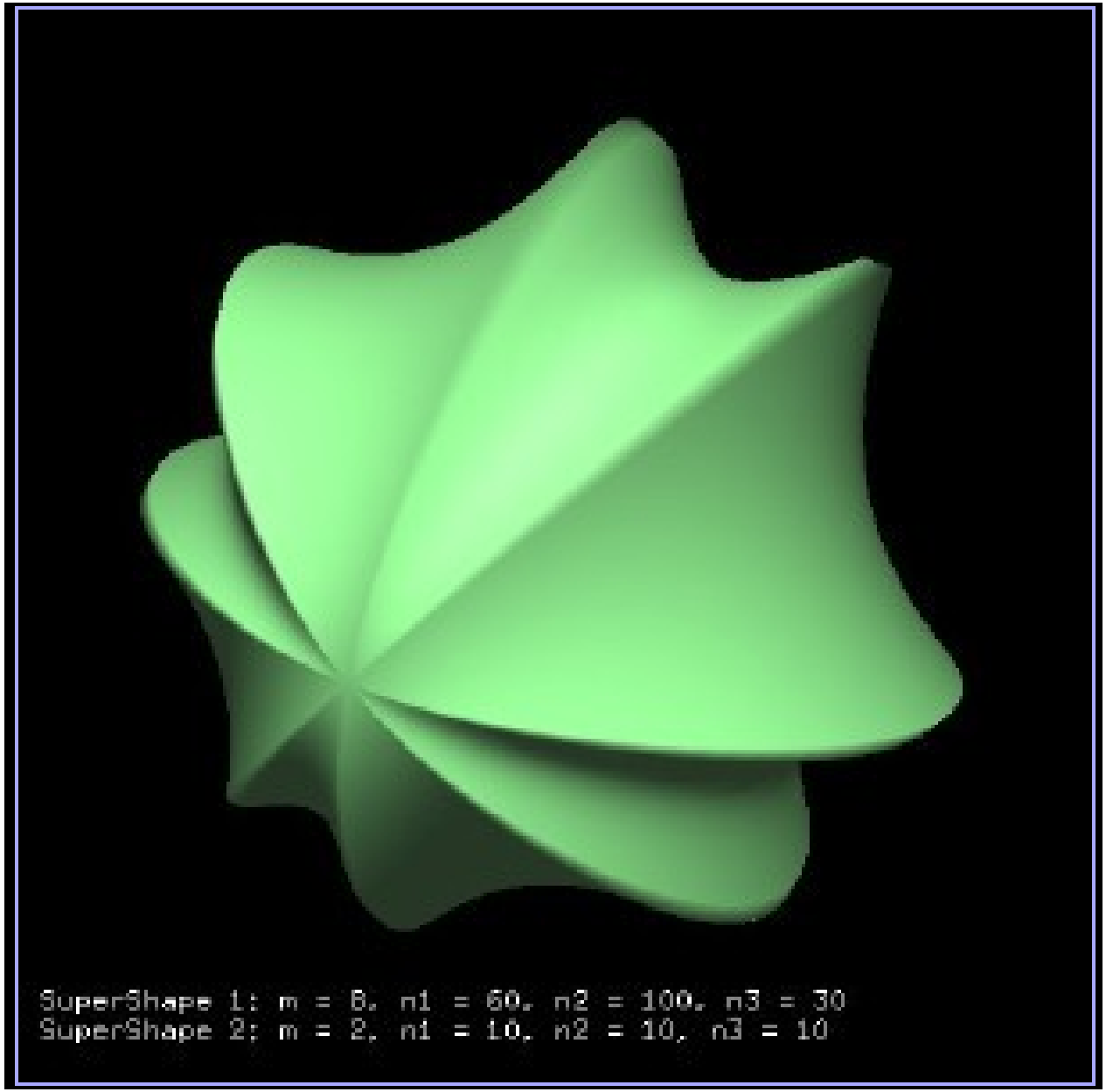
Originally developed by Johan Gielis(*) this SF was originally intended as a tool to analyze forms of bodies arising spontaneously in the natural world.

This formula gives rise to a surprising variety of 3D bodies of which we offer some interesting examples.

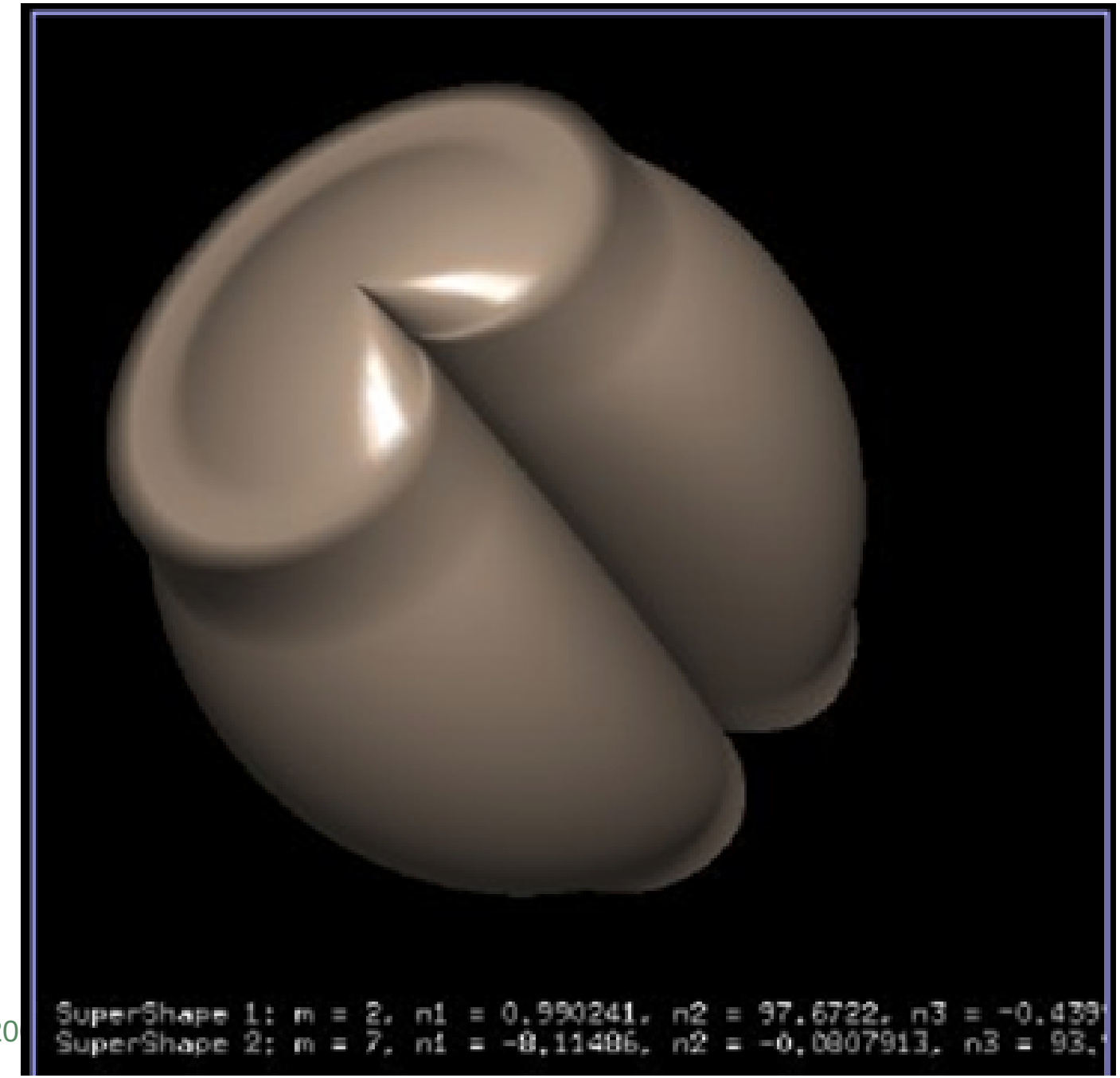
(*)Gielis, J.: "A generic geometric transformation that unifies a wide range of natural and abstract shapes", American Journal of Botany, 90(3): pp. 333–338, 2003.

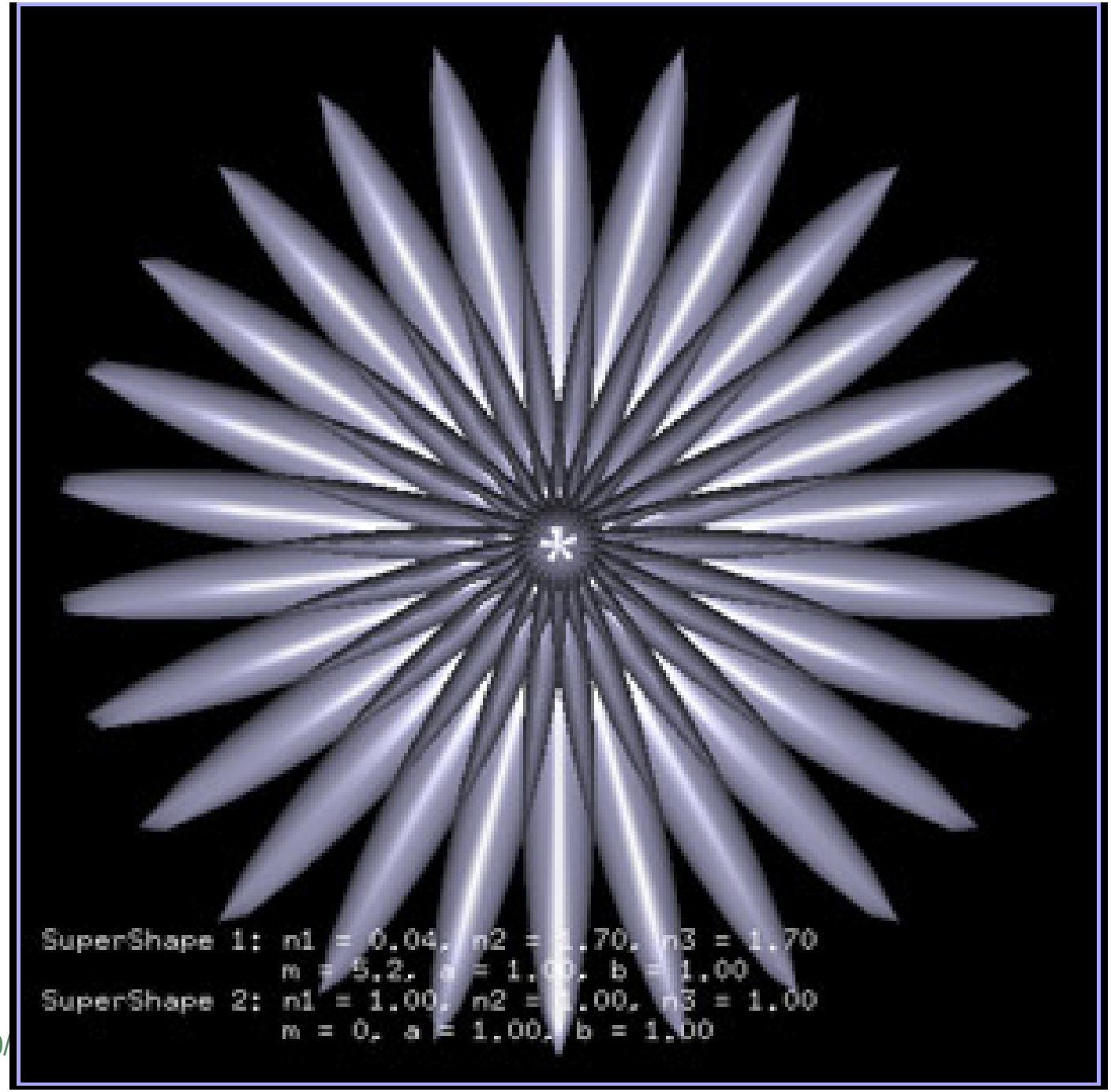


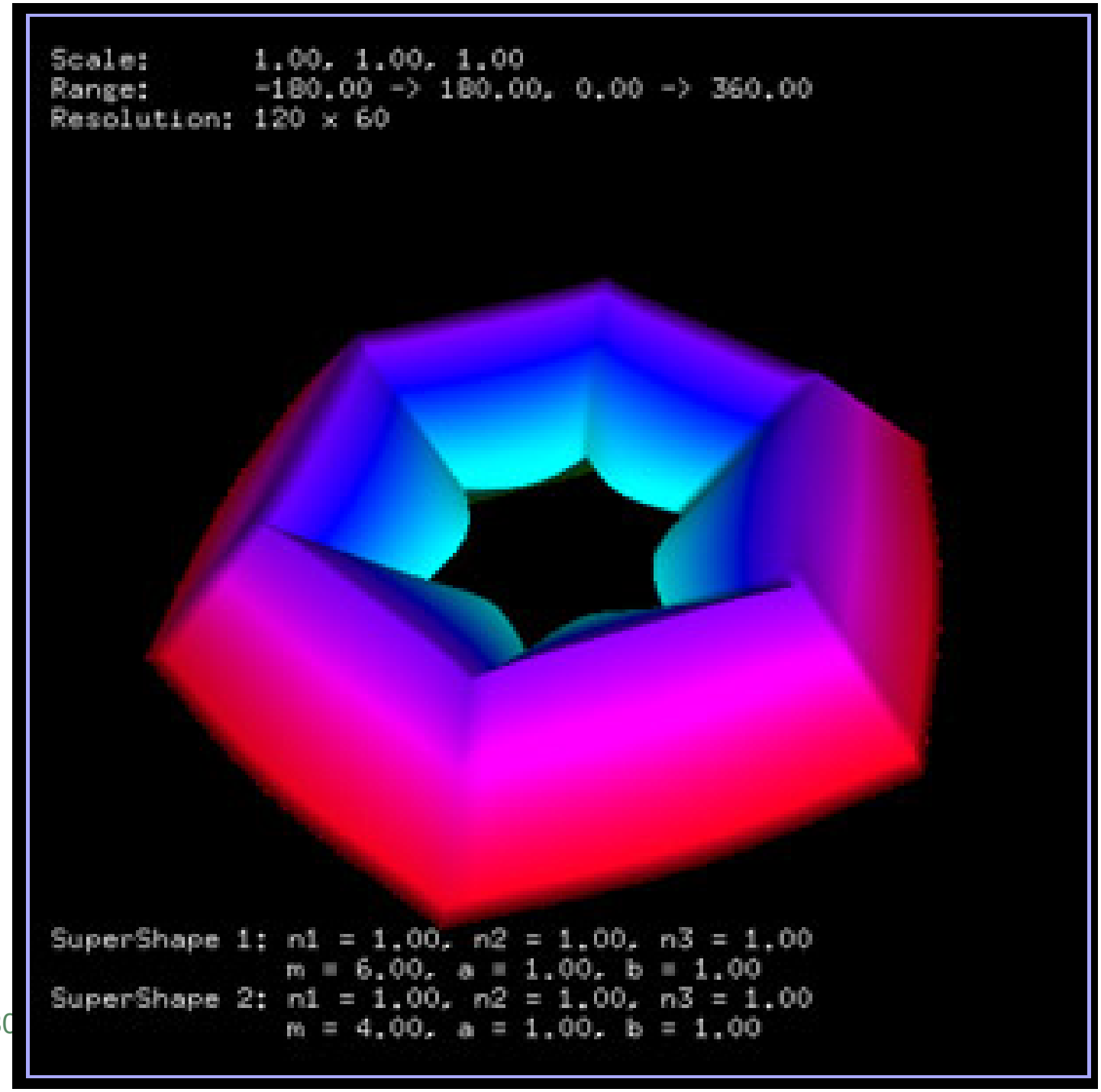

SuperShape 1: n1 = 0.30, n2 = 0.30, n3 = 0.30
m = 0.166667, a = 1.00, b = 1.00
SuperShape 2: n1 = 1.00, n2 = 1.00, n3 = 1.00
m = 0, a = 1.00, b = 1.00

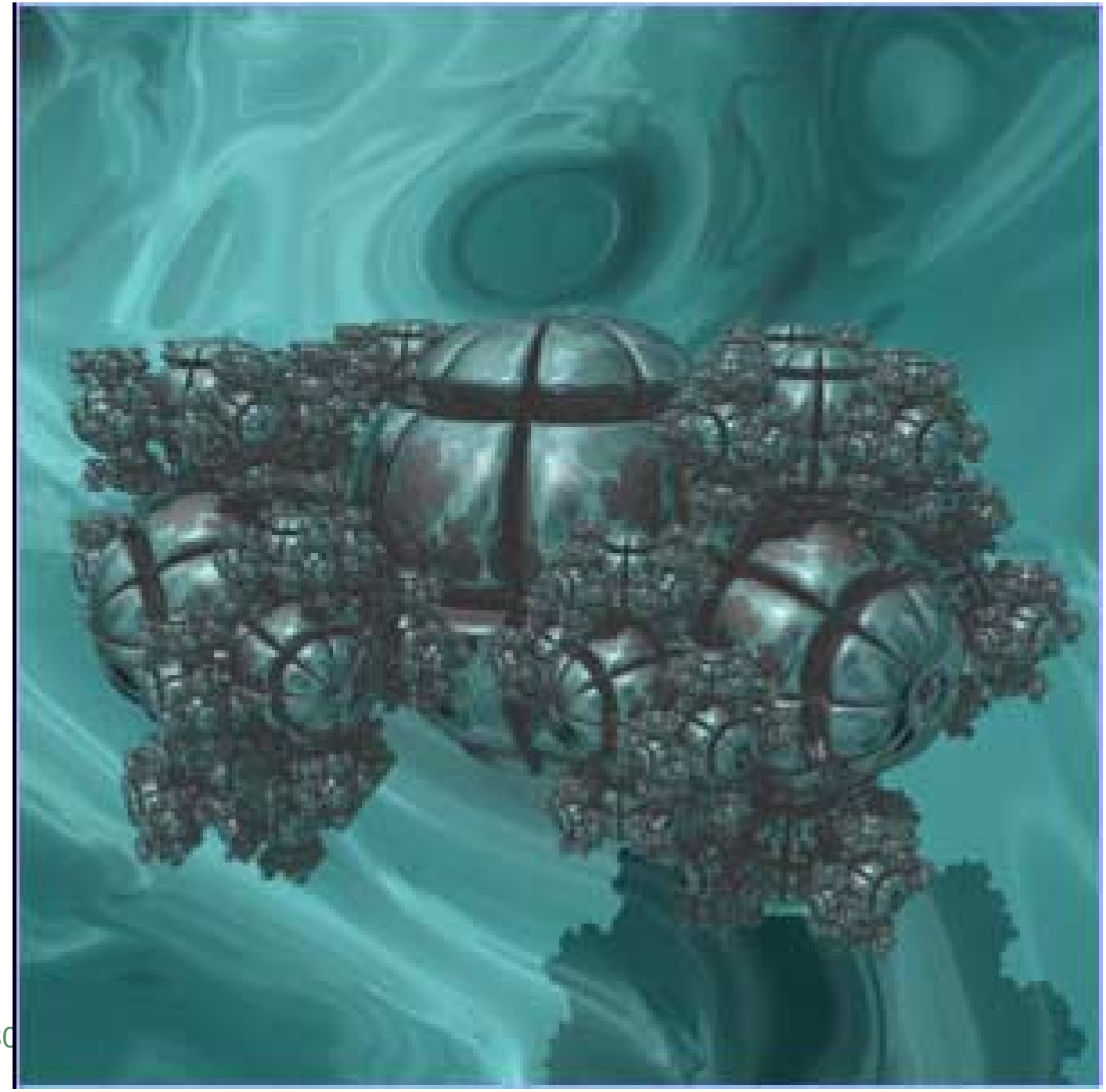

SuperShape 1: m = 6. n1 = 60. n2 = 100. n3 = 30
SuperShape 2: m = 2. n1 = 10. n2 = 10. n3 = 10

11/30/2008


SuperShape 1: m = 2, n1 = 0,990241, n2 = 97,6722, n3 = -0,439
SuperShape 2: m = 7, n1 = -0,11486, n2 = -0,0807913, n3 = 93,


SuperShape 1: n1 = 0.04, n2 = 1.70, n3 = 1.70
m = 5.2, a = 1.00, b = 1.00
SuperShape 2: n1 = 1.00, n2 = 1.00, n3 = 1.00
m = 0, a = 1.00, b = 1.00


Scale: 1.00, 1.00, 1.00
Range: -180.00 -> 180.00, 0.00 -> 360.00
Resolution: 120 x 60
SuperShape 1: n1 = 1.00, n2 = 1.00, n3 = 1.00
m = 6.00, a = 1.00, b = 1.00
SuperShape 2: n1 = 1.00, n2 = 1.00, n3 = 1.00
m = 4.00, a = 1.00, b = 1.00



Clustering with Gielis SF

We decided to try and use Gielis SF “in reverse”.

Instead of plugging in some values for its parameters in order to investigate the properties arising from the Gielis bodies we intend to find the parameters which define the hypothetical bodies which encompass the largest possible data vectors.

How do we find the parameters?

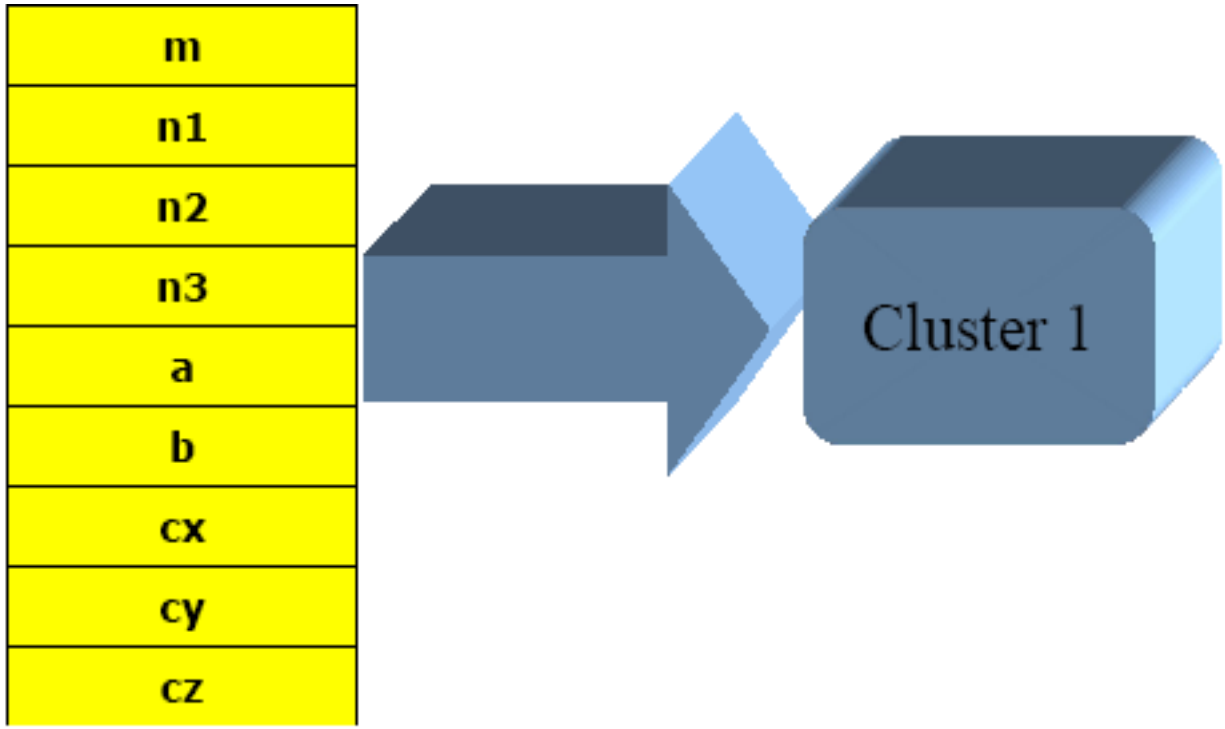
The idea is simple:

a) Determine the number of clusters

b) Encode this in a binary string

c) Evolve the best solution using a Genetic Algorithm

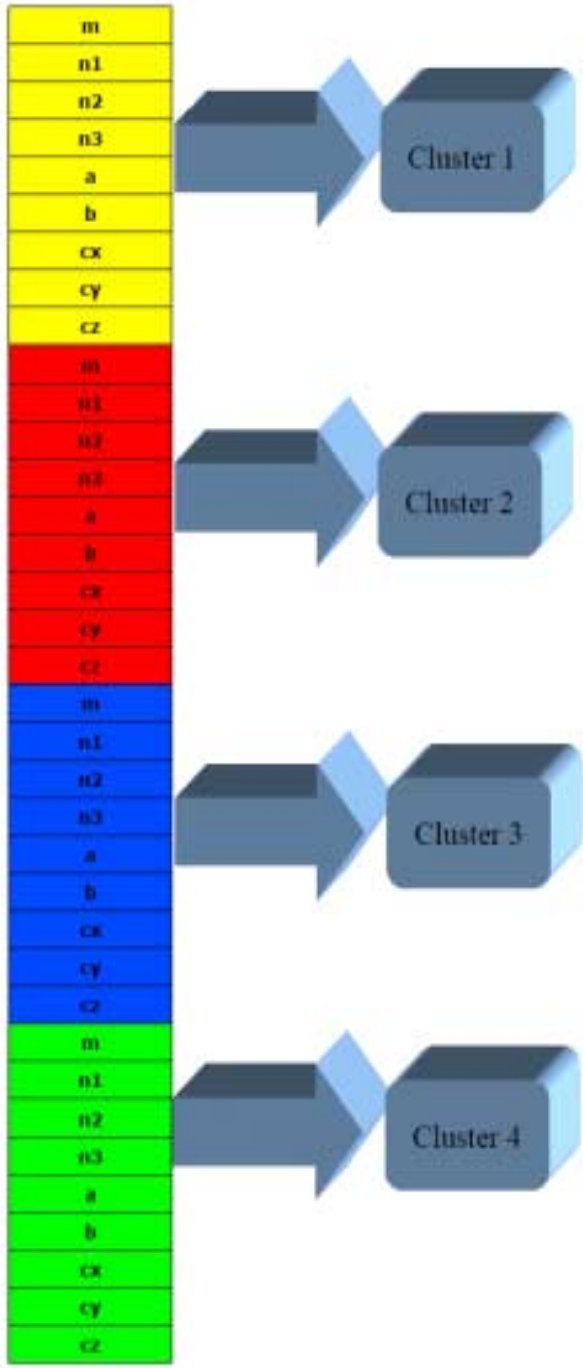
Encoding



The variables to optimize are m, n1, n2, n3, a, b for each of the clusters. Cx, Cy and Cz are the coordinates of the centers.

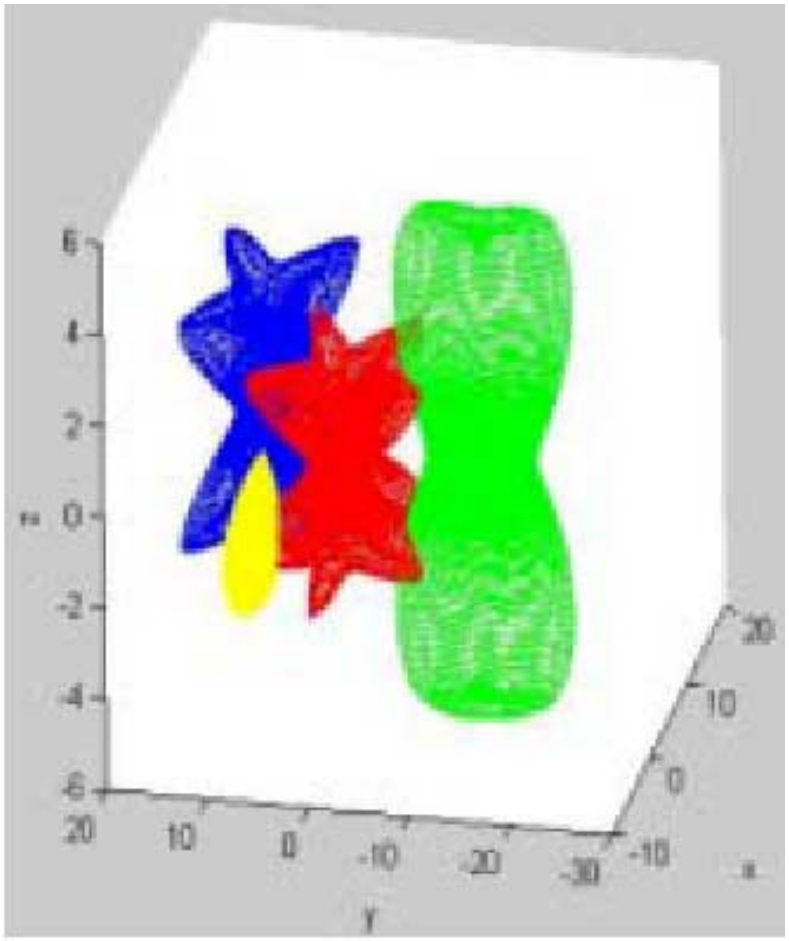
Encoding

The full chromosome (assuming 4 clusters) is shown in the figure.



Geometric Clustering

To test our method we “generated” 4 bodies and obtained 2,500 elements in every cluster.



Geometric Clustering

The number of correctly assigned elements per cluster is as follows.

Cluster 1	2430
Cluster 2	2455
Cluster 3	2463
Cluster 4	2475
Total	9823
% Error	1,77

Relative Performance

We tackled the same data set with other algorithms. In the perceptrons the network was trained with known memberships.

Method	Errors	% Error
Kohonen Maps	2,654	25.51
Fuzzy C-Means	389	3.74
Perceptron Network	310	2.98
VGA-Gielis	177	1.7

Results

Notice that using our strategy we obtained better results than even those gotten from supervised learning algorithms (i.e. multi-layered perceptron networks).

This fact is, of course, encouraging since the form of the clusters was totally irregular.

Conclusions

Geometric clusters are not hampered by the hyper-spherical-cluster-form limitation.

The complexity of the chromosome, however, grows with the number of clusters and dimensions.

The task of generalizing Gielis SF to higher dimensions is not a trivial task.

Conclusions

Furthermore, as the GA's chromosome grows in length so does the expected computational time.

On the other hand, the matter of determining whether a data vector lies within the "Gielis body" becomes computationally challenging.

We expect to report on a much wider set of experiments shortly.