Ensembles of Probabilistic Principal Surfaces and Competitive Evolution on Data: two different approaches to data classification

Roberto Amato[1], Angelo Ciaramella[2], Carmine Del Mondo[1], Lara De Vinco[2],
Ciro Donalek[3,4], Giuseppe Longo[1,4,5], Gennaro Miele[1,4],
Giancarlo Raiconi[2], Antonino Staiano[2], Roberto Tagliaferri[2,4]
1 - Department of Physical Sciences, University Federico II of Napoli, via Cinthia 6, ITALY
2 - Department of Mathematics and Informatics, University of Salerno, Fisciano - ITALY
3 - Department of Mathematics and Applications, University Federico II of Napoli, via Cinthia 6, ITALY
4 - INFN - Italian Institure of Nuclear Physics Unit of Naples, via Cinthia 6, Italy
5 - INAF - Italian Institute for Astrophysics, via Moiariello 16, Napoli, ITALY
astaiano@unisa.it

Abstract

Probabilistic Principal Surfaces (*PPS*) offer very powerful visualization and classification capabilities and overcome most of the shortcomings of other neural tools such as SOM, GTM, etc. More specifically *PPS* build a probability density function of a given data set of patterns lying in a D-dimensional space (with $D >> 3$) which can be expressed in terms of a limited number of latent variables laying in a Q-dimensional space (Q is usually 2-3) which can be used to visualize the data in the latent space. *PPS* may also be arranged in ensembles to tackle very complex classification tasks. Competitive Evolution on Data (*CED*) is instead an evolutionary system in which the possible solutions (cluster centroids) compete to conquer the largest possible number of resources (data) and thus partition the input data set in clusters. We discuss the application of Spherical–*PPS* to two data sets coming, respectively, from astronomy (Great Observatory Origins Deep Survey) and from genetics (microarray data from yeast genoma) and of *CED* to the genetics data only.

Keywords: neural networks, genetic algorithms, data mining, visualization

1. INTRODUCTION

The explosive growth in the quantity, quality and accessibility of data which is currently experienced in all fields of science and human endeavor, has triggered the search for a new generation of computational theories and tools capable to assist humans in extracting useful information (knowledge) from huge amounts of distributed and heterogeneous data. The field of Knowledge Discovery in Databases or KDD is therefore becoming of paramount importance not only in its traditional arena but also as an auxiliary tool for almost all fields of research.

In this paper, we shall first focus on the visualization and classification capabilities of Ensembles of Spherical Probabilistic Principal Surfaces (*PPS*; Sect.2) [1, 2, 3] and then on a new genetic algorithm method: Competitive Evolution on Data (*CED*; Sect.3) which seems to offer several advantages in the field of data classification. We also discuss two applications in the fields of astronomy (Sect.4) and genetics (Sect.5), respectively. All results have been obtained in the framework of the Astroneural [4] collaboration: a joint project between the Department of Mathematics and Informatics of the University of Salerno and the Department of Physical Sciences of the University Federico II in Napoli.

2. ENSEMBLES OF SPHERICAL PROBABILISTIC PRINCIPAL SURFACES

Probabilistic Principal Surfaces or *PPS* [1, 2] are a nonlinear extension of principal components, in that each node on the *PPS* is the average of all data points that projects near/onto it. From a theoretical point of view, *PPS* may be seen as a generalization of the Generative Topographic Mapping (*GTM*) [5], which on the other hand, can be seen as a parametric alternative to Self Organizing Maps (*SOM*) [6]. *PPS* are governed by their latent topology [7] and, owing to their intrinsic flexibility, a large variety of *PPS* topologies can be created. Among these, that of a $3D$ sphere (Spherical-*PPS*) is particularly appealing since a sphere is both finite and unbound and, being the nodes distributed at the edge of the sphere, they are ideal for emulating the sparseness and peripheral property of high-D data. Furthermore, data points projecting near a principal surface node have higher influences on that node than the points which project far away from it (Fig.1). Finally, the sphere topology (with no edges such as, for instance, is the case for *SOM*) can be easily comprehended by humans and thereby be extremely effective for the visualization of high-D data.

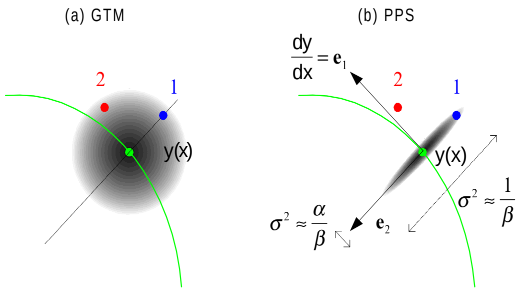

(a) GTM
2
1
y(x)
(b) PPS
$\frac{dy}{dx} = \mathbf{e}_1$
2
1
y(x)
$\sigma^2 \approx \frac{\alpha}{\beta}$
$\mathbf{e}_2$
$\sigma^2 \approx \frac{1}{\beta}$

FIGURE 1: Under a spherical Gaussian model of the *GTM*, points 1 and 2 have equal influences on the center node $y(\mathbf{x})$ (a) *PPS* have an oriented covariance matrix so point 1 is probabilistically closer to the center node $y(\mathbf{x})$ than point 2 (b).

A detailed description of how (Spherical) *PPS* work and of their advantages over more traditional tools such as *GTM* or *SOM* can be found in [8, 9]. Since *PPS* build a probability density function (*pdf*) as a mixture of Gaussian distributions trained through Expectation-Maximization (EM) algorithm, their performances may degrade with increasing data dimensionality due to singularities and local maxima in the log-likelihood function. One way to circumvent this problem is to build committees of spherical–*PPS* to gain improved pdfs and hence classification rates.

Ensembles of learning machines have so far been successfully and extensively applied to neural networks especially in the case of supervised learning algorithms. Fewer instances can instead be found of unsupervised learning methodologies and of density estimations: among these, those introduced by [10] and [11] exploit consolidated techniques such as stacking [12] and bagging [13], and represent the basis of our implementation.

2.1. Ensembles via Stacking

Let us suppose we are given S probabilistic principal surface models (i.e., S density estimators) $\{PPS_s(\mathbf{t})\}_{s=1}^{S}$, where $PPS_s(\mathbf{t})$ is the s-th *PPS* model. Note that in the original formulation given in [11], the S density estimators could also be of different types, for example finite mixtures with a fixed number of component densities or kernel density estimate with a fixed kernel and a single fixed global bandwidth in each dimension.

Now, going back to our model, each of the S *PPS* models can be chosen to be diverse enough, i.e. by considering different π_s values, number of latent variables and latent bases. In order to stack the S *PPS* models, we follow the procedure described below:

1. Let D the training data set, with size $|D| = N$. Partition D v times, as in v-fold cross-validation. The v-th fold contains exactly $(v-1)\frac{N}{v}$ training data points and $\frac{N}{v}$ test data points both from the training set D. For each fold:

- (a) fit each of the S *PPS* models to the training subset of D.
- (b) evaluate the likelihood of each data point in the test partition of D, for each of the S fitted models.

2. At the end of these preliminary steps, we obtain S density estimators for each of the N data points which are organized in a matrix A, of size $N \times S$, where each entry a_{is} is $PPS_s(\mathbf{t}_i)$;
3. Use the matrix A to estimate the combination coefficients $\{\pi_s\}_{s=1}^{S}$ that maximize the log-likelihood at the points $\mathbf{t}_i$ of a stacked density model of the form:

$$\textit{StPPS}(\mathbf{t}) = \sum_{s=1}^{S} \pi_s PPS_s(\mathbf{t}_i)$$

which corresponds to maximize

$$\sum_{i=1}^{N} \ln \left(\sum_{s=1}^{S} \pi_s PPS_s(\mathbf{t}) \right),$$

as a function of the weight vector $(\pi_1, \ldots, \pi_S)$. Direct maximization of this function is a non-linear optimization problem. We can apply the EM algorithm directly, by observing that the stacked mixture is a finite mixture density with weights $(\pi_1, \ldots, \pi_S)$. Thus, we can use the standard EM algorithm for mixtures, except that the parameters of the component densities $PPS_s(\mathbf{t})$ are fixed and the only parameters allowed to vary are the mixture weights.
4. The concluding phase consists in the parameters re-estimation of each of the S component *PPS* models using all of the training data D. The stacked density model is then the linear combination of the so obtained component *PPS* models, with combining coefficients $\{\pi_s\}_{s=1}^{S}$.

2.2. Ensembles via Bagging

A second combining scheme uses bagging as a method to average single density estimators, in our case the *PPS*, in a way similar to the model proposed in [10]. All we have to do is to train a number S of *PPS* with S bootstrap replicates of the original learning data set. At the end of this training process, we obtain S different density estimates which are then averaged to form the overall density estimate model. Formally speaking, let D be the original training set of size N and $\{PPS_s\}_{s=1}^{S}$ a set of *PPS* models:

1. create S bootstrap replicates (with replacement) of D, $\{D_{Boot}(s)\}_{s=1}^{S}$ with size N;
2. train each of the S *PPS* models with a bootstrap replicate D_{Boot};
3. at the end of the training we obtain S density estimates $\{PPS_s\}_{s=1}^{S}$;
4. average the S density estimates $\{PPS_s\}_{s=1}^{S}$ as

$$\textit{BgPPS}(\mathbf{t}) = \frac{1}{S} \sum_{s=1}^{S} PPS_s(\mathbf{t}).$$

3. COMPETITIVE EVOLUTION ON DATA

The Competitive Evolution on Data (*CED*) model [14] is an evolutionary system [15, 16] specifically tailored to perform clustering on noisy data in absence of a priori knowledge. The main feature of CED is that it uses a double metaphoric association: i) one, which is also shared by all evolutionary systems, in which the possible solutions are generic individuals and, ii) a second one initially introduced by [17] and commonly used to solve multi-modal problems [18], in which each datum is associated to a resource for which the individuals will compete. The data set therefore represents the environment in which the individuals (*id est*, the possible solutions) evolve. The evolutionary process picks out those individuals which, being better fit to the environment, are capable to attract a larger number of resources and therefore have also a higher chance to reproduce. The evolutionary dynamics of such a system is such that individuals tend to conquer the regions of the environment which are more densely populated and to neglect those which are empty or scarcely populated. ([19, 14]).

3.1. Evolutive scheme

The general structure of the *CED* model is the following:

- first we randomly generate a set of possible solutions (initial population) and to each individual we attribute a fitness value 0;
- the following steps are iterated (the number of iterations is established via a trial and error procedure):
  - The fitness of each individual is computed from the number of resources they have been able to conquer;
  - Using genetic operators, individual are left free to reproduce with a probability which is proportional to their fitness and a second generation is produced.

At each generation a set of data is randomly selected to become the resources of the virtual environment where individuals are competing. Than for each resource takes place a tournament among a randomly selected number (arbitrarily fixed) of individuals and the individual which is closest to the resource "conquests it" and its fitness is increased by one unit. The distance is defined through a similarity criterion $d(x, z)$. In other words: if D is the input data space, $d : D \times D \rightarrow R^+$ and x and y are two individuals, x is closer to the resource z if $d(x, z) < d(y, z)$. At the end of the process the individuals with higher fitness are those which have conquered the larger number of resources and therefore are those which better represent groups of data very similar to each other. In what follows we make use of three different definitions of d:

Euclidean metric: $d(x, y) = \sqrt{\sum x_i^2 - y_i^2}$

Pearson centered: $d(x, y) = 1/|\sigma(x, y)| - 1$ dove $\sigma(x, y) = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$

Pearson non–centered: $d(x, y) = 1/\sigma(x, y) - 1$ dove $\sigma(x, y) = \frac{\sum x_i y_i}{\sqrt{\sum x_i^2 \sum y_i^2}}$

3.2. Extraction of the results

The number of solutions (id est, cluster centroids) produced during the evolution may in general be quite high and, in order to extract the most significant ones it is necessary to run an extraction phase.

- Inside all populations groups of similar or identical individuals are flagged and a representative is randomly selected;
- each representative is attributed a weight which is proportional to the size of the group to which it belongs;
- all nearby representatives are grouped together and only one is selected using a weighted mean criterium;
- all "average representative" with weights falling below a given treshold are eliminated.
- the remaining representative define the centroids of the clusters.

All parameters are fixed using a trial and error procedure.

4. AN APPLICATION OF *PPS* TO ASTRONOMY: THE GOODS DATA SET

The Great Observatories Origins Deep Survey (or GOODS) is a joint project among the American and the European Space Agencies and several of the most powerful ground-based astronomical facilities. Aim of the project is to survey the distant universe to the faintest flux limits across the broadest range of wavelengths and at the end, GOODS data will cover a total of roughly 320 square arcminutes in two fields centered on the Hubble Deep Field North and the Chandra Deep Field South, respectively [20]. The GOODS catalog used in what follows contained 28405 objects.

Each object being measured in 7 optical bands, namely the *U,B,V,R,I,J,K* bands. For each band 3 different types of parameters, astrometric (positions), geometric (i.e., Kron radius, ellipticity etc.) and photometric (Fluxes and Magnitudes) were measured, adding up to a total of several dozens of parameters. Objects were also classified as angularly resolved (or *galaxies*, in the astronomical jargon) and non resolved (or *stars*).

GOODS (and more in general astronomical surveys) data present a further peculiarity: the majority of the objects are ”drop outs”, id est they are detected only in some bands and not detected in the others due to either instrumental (different detection limits) or intrinsic (different spectral properties) reasons. Without entering into details we must stress that the characterization of an object as a ”dropout” (id est as an object with a strong relative flux difference between two or more spectral regions) is very important from the astronomical point of view since it allows to discriminate among different classes of celestial objects. From our *statistical clustering* point of view, therefore, the data set contains four classes of objects, namely stars, galaxies, stars which are drop outs and therefore present missing data, and drop out galaxies (at this stage, we do not take into account the number of bands for which data are missing).

In Fig. 2, we give an example of the visualization capabilities of *PPS*: colors mark the four classes defined above.

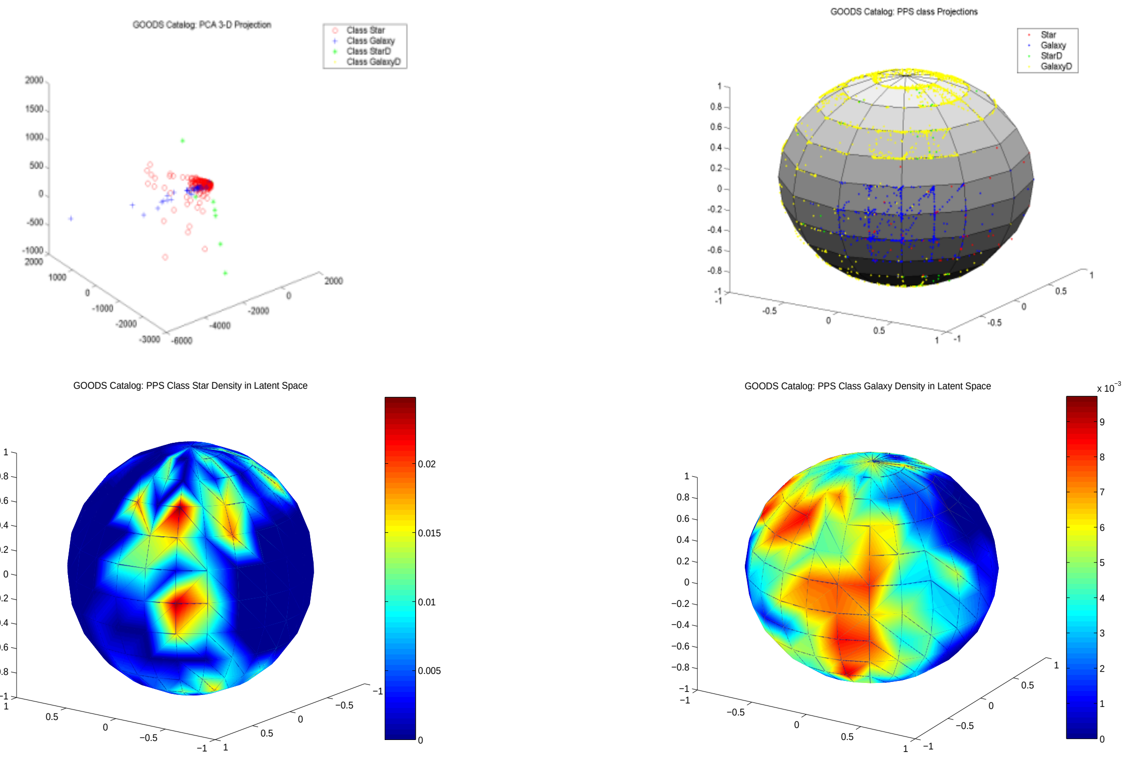


FIGURE 2: Different visualizations of the GOODS catalogue. From top left to bottom right clockwise: $3D$ PCA projections; *PPS* projections on the sphere; *PPS* galaxy density on the sphere and, *PPS* star density on the sphere.

As it is clear, The PCA visualization gives very little information since it displays only a single condensed group of data. The *PPS* projection onto the spherical latent manifold appears much more readable than the PCA and contains much more information and clearly shows several well defined clusters.

4.1. Classification of GOODS data

In the case of *StPSS*, we built a model in which a group of six different *PPS* models, each one with a fixed α value, are put together in an ensemble via stacking. An important parameter for stacking is the number *v* of folds in the cross-validation procedure. In our experiments we tried $5-$fold and $10-$fold cross-validation. In the case of *BgPPS* we used, instead, a single *PPS* model with its own parameters setting and to bag it in order to improve its performance. In our experiments we bag ten *PPS* models (for $\alpha = 0.2, 0.4, ...2.0$) in order to assess the best α value. The *PPS* models are trained on 20 bootstrap replicates of the training data set (hence we have a committee of 20 *PPS* models whose responses are averaged).

In all experiments, the classifiers run 25 times and each time new training and test data partitions (60% for training and 40% for testing) are generated. Moreover, for comparison purposes we accomplished classification by using single *PPS* models as well, to

1. compute the reference manifolds for each class (we denote this classifier as *PPSRM*),
2. compute the posterior class probability (hereinafter denoted as *PPSPR*).

4.2. Application to the GOODS Catalog: *StPPS*

In GOODS catalog the behavior of the stacked model, for which the parameters are set as in Table 1, is inverted in terms of $5-$fold and $10-fold$ cross-validation. In fact here we have better results for $10-fold$ cross-validation (mean classification error 2.87 and standard deviation 0.1344) with respect to $5-$fold cross validation (mean classification error 3.44 and standard deviation 0.4720) as it can be seen from Fig. 3. This is reasonable, since the number of training data for the first three classes (*S*, *G* and *SD*) is much smaller than the number of training data for class *GD*, so a higher number of folds leads to a better fit to data. The confusion matrix corresponding to the minimum error (1.05) is shown in Table 2.

TABLE 1: *GOODS* Catalog: *StPPS* parameter settings

Parameters	*PPS*$_1$	*PPS*$_2$	*PPS*$_3$	*PPS*$_4$	*PPS*$_5$	*PPS*$_6$
α	1.4	1.2	0.8	0.6	1.6	2.0
M	266	266	266	266	615	615
L	18	83	83	83	83	83
L_{fac}	1	2	1.5	1.1	1.3	2
$iter$	100	100	100	100	100	100
ϵ	0.01	0.01	0.01	0.01	0.01	0.01

TABLE 2: *GOODS* Catalog: confusion matrices computed by *StPPS* best model

Classifier	*Confusion Matrix*				
		S	*G*	*SD*	*GD*
	S	92	4	2	0
$StPPS(2.62)$	*G*	76	1234	2	36
	SD	0	0	52	36
	GD	0	8	134	9688

4.3. Application to the GOODS Catalog: *BgPPS*

For *GOODS* catalog the results are more fluctuating for each of the α values. In fact the best results are obtained between the interval $[0.2, 0.6]$ and $[1.4, 2.0]$. The overall best result falls in the second interval, in particular for $\alpha = 1.8$ (mean classification error 2.74 and standard deviation 0.3987) even though *BgPSS* with $\alpha = 0.6$ obtains a lower standard deviation value (0.1725). The minimum classification error with confusion matrix is shown in Table 4.

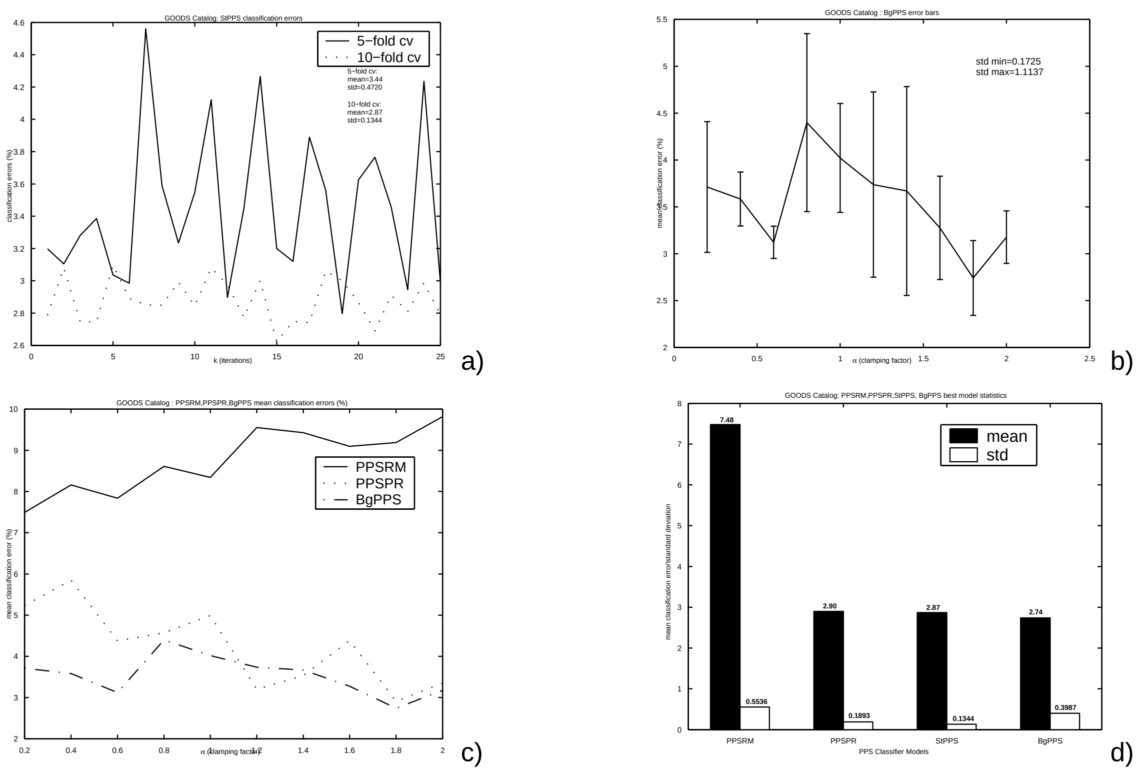


FIGURE 3: Panel a): *GOODS* Catalog: *StPPS* classification errors over 25 iterations; Panel b): BgPPS error bars over 25 iterations for each fixed α; Panel c: *PPSRM*, *PPSPR* and *BgPPS* mean classification errors over 25 iterations for each fixed α; Panel d): *PPSRM*, *PPSPR*, *StPPS* and *BgPPS* best model statistics.

TABLE 3: *GOODS* Catalog: *BgPPS* parameter settings

Parameter	*Value*	*Description*
M	266	number of latent variables
L	83	number of basis functions
L_{fac}	1	basis functions width
$iter$	100	maximum number of iteration
ϵ	0.01	early stopping threshold

TABLE 4: *GOODS* Catalog: confusion matrix computed by *BgPPS* best model

Classifier	*Confusion Matrix*					α
		S	*G*	*SD*	*GD*	
$BgPPS(2.15)$	*S*	155	35	12	5	1.8
	G	8	1160	6	8	
	SD	0	0	64	7	
	GD	5	51	108	9740	

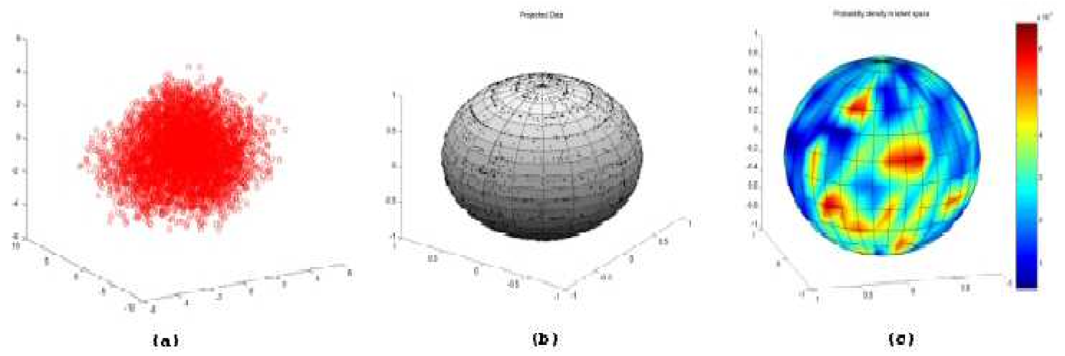


FIGURE 4: Yeast Gene Data Set: (a) $3D$ PCA projection (b) Data point projections in the latent space (c) Data probability density in the latent space

4.3.1. GOODS *Catalog:* PPSRM*,* PPSPR*,* StPPS *and* BgPPS *Comparison*

GOODS catalog classification task is more complex. The four classes are heavily overlapping and even in the best cases there are classes (i.e., *S* and *SD*) whose objects are classified with an error rate about 60%. This is evident from the results obtained by the different classifiers used. However, even in this case *BgPPS* outperforms all the other models (*PPSRM*, *PPSPR* and *StPPS*). Moreover, Stacked PPS here outperforms both *PPSRM* and *PPSPR*. Among the two single PPS classifier models, *PPSPR* is still better than *PPSRM* (see Fig. 3).

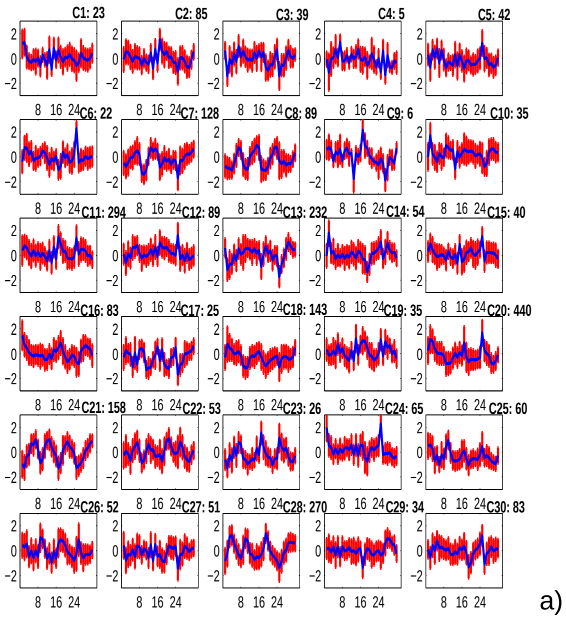
5. APPLICATION OF *PPS* AND *CED* TO GENETICS: MICROARRAY YEAST GENE DATA

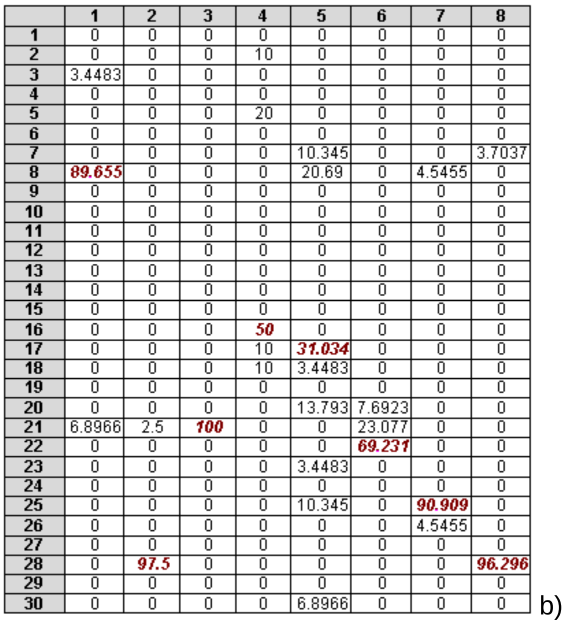
Gene-expression microarrays, whose development started in the second half of the 1990's, which are having a powerful impact on molecular biology. In fact, although the ability to measure transcription of a single gene is not new, the possibility to measure the transcription of all genes in an organism at once is a recent advance and is leading to new methods of diagnosis and of treatment for a large number of diseases. However, it is also becoming increasingly clear that simply generating the data is not enough and that the extraction of the relevant information is a non trivial task. Statistical techniques and other classical methods of data analysis, are not adequate and therefore, in the last decade, much work has focused on the development of machine learning methodologies suited for the analysis of genetic data.

Just to mention a few, support vector machines have been used for the functional classification of genes [21]; clustering techniques were used for grouping similar expression patterns across a number of experiments of all the genes of the yeast Saccharomyces Cerevisiae [22]; Neural networks have been employed both for clustering and visualization of gene microarray data [23, 24]. In order to investigate the capabilities of our methods in this different field of applications, we started from the work of Spellman and his colleagues which is described in [22] and provides a comprehensive catalog of yeast genes whose transcript levels vary periodically within the cell cycle. In order to produce the catalogue, samples from yeast cultures synchronized with different experiments were used. In [22] a type of agglomerative hierarchical clustering [25] was used in order to identify clusters of genes behaving similarly in each experiment and which represent groups of apparently co-regulated genes. These clusters provide a solid basis for understanding the transcriptional mechanism of cell cycle regulation. The data set used by us, consists of a set 6125 genes, subject to four different experiments. Each experiment consists of measurements at different epochs for a total of 73 parameters.

5.1. Results from Spherical *PPS*

In order to make the data set more apt to be processed with *PPS*, we first applied a preprocessing phase in which, through the use of a nonlinear PCA [26], we reduced each experiment to 8 measurements and eliminated the genes whose experiments had to much missing data. Hence,





	1	2	3	4	5	6	7	8
1	0	0	0	0	0	0	0	0
2	0	0	0	10	0	0	0	0
3	3.4483	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0
5	0	0	0	20	0	0	0	0
6	0	0	0	0	0	0	0	0
7	0	0	0	0	10.345	0	0	3.7037
8	***89.655***	0	0	0	20.69	0	4.5455	0
9	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0
16	0	0	0	***50***	0	0	0	0
17	0	0	0	10	***31.034***	0	0	0
18	0	0	0	10	3.4483	0	0	0
19	0	0	0	0	0	0	0	0
20	0	0	0	0	13.793	7.6923	0	0
21	6.8966	2.5	***100***	0	0	23.077	0	0
22	0	0	0	0	0	***69.231***	0	0
23	0	0	0	0	3.4483	0	0	0
24	0	0	0	0	0	0	0	0
25	0	0	0	0	10.345	0	***90.909***	0
26	0	0	0	0	0	0	4.5455	0
27	0	0	0	0	0	0	0	0
28	0	***97.5***	0	0	0	0	0	***96.296***
29	0	0	0	0	0	0	0	0
30	0	0	0	0	6.8966	0	0	0

b)

FIGURE 5: Panel a: *PPS* cluster prototype periodic behaviors and error bars (3σ) showing the standard deviation of genes from the prototypes for a fixed cluster. On the top of each subplot are the cluster number and the number of genes within each cluster. Panel b: *PPS* and Spellman cluster comparisons. On each row are reported the 30 *PPS* clusters, while on the columns are the clusters computed by Spellman. The A_{ij}-th entry of the table correspond to the fraction of Spellman cluster j falling in the *PPS* cluster i

the used data set consists of 5425 genes and 32 features. Furthermore, since, in general, microarray data is noisy, it is necessary to resort to some kind of cleaning procedure, to identify those genes affected from noise process involved in the generation of data from microarrays. At this aim, we decided to train a *PPS* with a high number of latent variables, so that each one is responsible of a limited number of data points, afterward, we apply a clustering procedure on the nodes of the manifold in the data space. So doing, a number of identified clusters containing genes with low variance (i.e. genes whose transcript levels show a poor periodic behavior) were thrown away. The number of remaining genes turned out to be 2761. We then used a *PPS* with 266 latent variables and 40 latent basis functions and a clamping factor α set to 0.5. After the completion of the training phase we projected the data in latent space and computed the responsibility for each latent variable as shown in Fig. 4.

On the basis of the probability density functions visualized in Fig. 4 we decided to identify 30 clusters through a hierarchical clustering procedure. For each cluster we plot the prototype trend with respect to the 32 features, as it can be seen in Fig. 5-a, which highlights the average behavior of genes belonging to the same cluster. In Fig. 5-b we compare the results of the *PPS* clustering procedure with those by Spellman [22]. We wish to stress that the two clustering procedures were completely different: Spellman in fact clustered the gene properties using an a priori knowledge of their characteristics and thus he worked with only 209 genes, while our algorithm made use only of the statistical properties of the data with no a priori knowledge. In spite of this, some remarkable patterns may be detected: Spellman's clusters number 1 falls near entirely in our cluster 8; Spellman's clusters number 2 and 8 are statistically speaking indistinguishable (together they form our cluster number 28); Spellman's cluster number 5 appears to be a sort of statistical waste basket which groups together rather different clusters ($7, 8, 17, 20$ plus several others with lower significance) which, however, are topological neighbors in the *PPS* latent space and can therefore be considered as "substructures" (missed by Spellman) of a larger cluster. Finally, cluster 21 contains entirely the genes belonging to Spellman's cluster 3. The most relevant result, however, seems to be the fact that many (13 out of 30) of our clusters are not mapped by any of the 209 genes in the Spellman's sample. Whether these clusters have or have not any biological significance will be the subject of future studies.

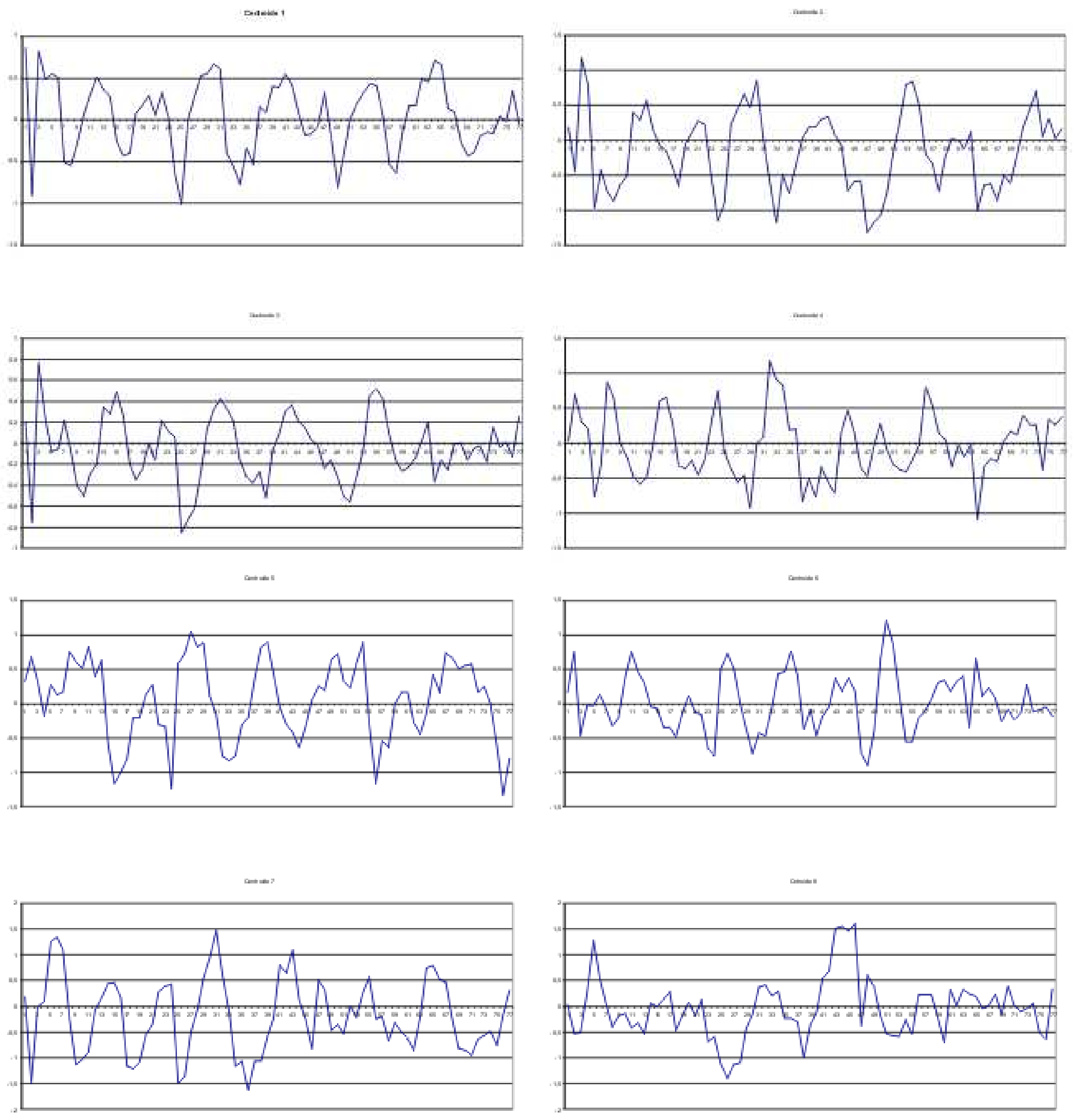


FIGURE 6: Average behavior of genes belonging to the clusters attributed to the centroids 1–8.

	1 Sp.	2 Sp.	3 Sp.	4 Sp.	5 Sp.	6 Sp.	7 Sp.	8 Sp.
1	0	**40**	0	0	2	0	0	0
2	**31**	0	0	0	1	0	0	0
3	0	0	0	0	**29**	0	0	0
4	0	0	0	0	0	0	**22**	0
5	0	0	0	0	0	0	0	**27**
6	0	0	0	0	0	**15**	0	0
7	0	0	**8**	0	1	0	0	0
8	0	0	0	**10**	0	0	0	0

TABLE 5: Results of the clustering of the Spellman data for the CED method. Columns: Spelmann clusters; rows: CED clusters. The figures give the number of genes falling in that cell.

5.2. CED results

If we apply the CED model to the same Spellman data set we have very interesting results. The system automatically extracts 8 centroids. If we then label each gene attributing him to the nearest centroid, we obtain 8 clusters. The average behavior of the genes belonging to each cluster is shown in Fig.6.

In Table 5 we compare the results obtained using the *CED* method on the Spellman sample. The number of genes belonging to the $i-th$ Spellman cluster is obtained by summing along the colums, while the number of genes in the $i-th$ *CED* cluster is obtained by summing along the rows. As it may be clearly seen, the *CED* clusters coincide almost perfectly with the Spellman ones (only 4 genes out of 186 are classified differently).

The Astroneural project is funded by the Italian Ministry for University and Scientific Research (MIUR) and by Regione Campania.

REFERENCES

[1] Chang K.: *Nonlinear Dimensionality Reduction Using Probabilistic Principal Surfaces*, PhD Thesis, Department of Electrical and Computer Engineering, The University of Texas at Austin, USA, (2000)
[2] Chang K., Ghosh J.: *A unified Model for Probabilistic Principal Surfaces*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 23, NO. 1, (2001)
[3] Staiano A.: *Unsupervised Neural Networks for the Extraction of Scientific Information from Astronomical Data*, PhD thesis, University of salerno, ITALY (2003).
[4] http://people.na.infn.it/∼longo/Ricerca/ASTRONEURAL/astroneural_main.html
[5] Bishop C.M., Svensen, M., Williams, C.K.I.: *GTM: The generative topographic mapping. Neural Computation* (1998)
[6] Kohonen T.: *Self-Organizing Maps*, Springer-Verlag, Berlin, (1995)
[7] Bishop C.M.: *Latent variable models*, In M. I. Jordan (Ed.), Learning in Graphical Models, MIT Press, (1999).
[8] Staiano A., Tagliaferri R., Longo G., Benvenuti P.: *Committee of Spherical Probabilistic Principal Surfaces*, in IJCNN'2004, p.58, in press (2004)
[9] Staiano A., De Vinco L., Ciaramella A., Raiconi G., Tagliaferri R., Longo G., Miele G., Amato R., Del Mondo C., Donalek C., Mangano G., Di Bernardo D.: *Probabilistic principal surfaces for yeast gene microarray data-mining*, in ICDM'04 - Fourth IEEE International Conference on Data Mining, in press (2004)
[10] Ormoneit D., Tresp V.: *Averaging, Maximum Likelihood and Bayesian Estimation for Improving Gaussian Mixture Probability Density Estimates*, IEEE Transaction on Neural Networks, Vol.9, NO. 4 (1998)
[11] Smyth P., Wolpert D.H.: *An evaluation of linearly combining density estimators via stacking*, Machine Learning, Vol. 36 (1999)
[12] Wolpert D.H.: *Stacked Generalization*, Neural Networks, 5, 241 (1992)

[13] Breiman L.: Bagging Predictors, Machine Learning, 26 (1996)
[14] Del Mondo C.: *Un metodo evolutivo per la ricerca di predicati “interessanti”:Un nuovo approccio al Data Mining Unsupervised*, Master Thesis in Physics, University Federico II of Napoli (ITALY), 2004
[15] Goldberg D. E.: *Genetic Algorithms in Search Optimization and Machine Learning*. Addison-Wesley (1989)
[16] Koza J.: *Genetic Programming: A Paradigm for Genetically Breeding Population of Computer Programs to Solve Problems*, Stanford University (1990)
[17] Forrest S., Javornik B., Smith R.E., Perelson A.S.: *Using Genetic Algorithms to Explore Patter Recognition in the Immune System*, Evolutionary Computation 1(3): 191-211 (1993)
[18] Horne J., Goldberg D.E., Deb K.: *Implicit Niching in a learning Classifier System: Nature’s Way*. Evolutionary Computation 2(1), 37 - 66 (1994)
[19] De Stefano C., Della Cioppa A., Marcelli A., Matarazzo F.: *Grouping Character Shapes by means of Genetic Programming*, IWVF, pp. 504-513 (2001)
[20] Dickinson, M.: *The great Observatories Origins Deep Survey: An Overview*, http://www.stsci.edu/science/goods/abstract.html
[21] Brown M., Grundy W., Lin D., Cristianini N., Sugnet C., Furey T., Ares M. Jr., Haussler D.: *Knowledge-based Analysis of Microarray Gene Expression Data by Using Support Vector Machines*, Proceedings of the National Academy of Science USA, 97 (1) (2000) 262–267
[22] Spellman P.T., Sherlock G., Zhang M.Q., Iyer V.R., Anders K., Eisen M.B., Brown P.O., Botstein D., Futcher B.: *Comprehensive identification of cell cycle-regulated genes of the yeast saccharomyces cerevisiae by microarray hybridization*, Molecular Biology of the Cell, 9:3273–3297 (1998)
[23] Tamayo P., Slonim D., Mesirov J., Zhu Q., Kitareewan S., Dmitrovsky E., Lander E.S., Golub T.R.: *Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation*, PNAS, 96:2907–2912 (1999)
[24] Toronen P., Kolehmainen M., Wong G., Castren E.: *Analysis of gene expression data using self-organizing maps*, FEBS Letters, 451:142–146 (1999)
[25] Eisen M.B., Spellman P.T., Brown P.O., Botstein D.: *Cluster analysis and display of genome-wide expressionpatterns*, PNAS, 95:14863–14868 (1998)
[26] Tagliaferri R., Ciaramella A., Milano L., Barone F., Longo G.: *Spectral analysis of stellar light curves by means of neural networks*, Astronomy and Astrophysics Supplement Series, 137:391–405 (1999)