# Entropy based selection for spectral clustering

**Technical Report 11-83**

**Fabian Ojeda**
fabian.ojeda@esat.kuleuven.be

**Carlos Alzate**
carlos.alzate@esat.kuleuven.be

**Bart De Moor**
bart.demoor@esat.kuleuven.be

**Johan A.K Suykens**
johan.suykens@esat.kuleuven.be

ESAT-SCD-SISTA Department of Electrical Engineering
Katholieke Universiteit Leuven
Kasteelpark Arenberg 10
B-3001 Heverlee, Belgium
http://www.esat.kuleuven.be/scd

**Abstract**

*Although many clustering algorithms have been proposed for the analysis of gene expression data [30], the validity of the generated partitions is often assessed using the full data set. Therefore future test genes from a subsequent stage, lack of a model assigning them to one of the initial clusters. In this context, we explore the use of a particular spectral clustering formulation that incorporates an extension for out-of-sample points [4]. A clear advantage of the proposed method is the possibility to train and validate the clustering model [3, 4]. We make use of this algorithm to cluster genes in groups based on their expression values. Firstly, the clustering model is built upon a selected subset of informative genes and, in a second stage the model is used to infer cluster memberships for the remaining genes. Informative genes are selected via entropy maximization, related to the underlying density distribution of the sample. This subsampling scheme greatly reduces the computational burden in large number of genes while the out of sample extension provides a clustering model for new samples. Application on both synthetic and real gene expression data supports the usefulness of this approach.*

# 1 Kernel *k*-means and Spectral Clustering

In contrast to the classification or regression tasks earlier discussed, the goal of clustering consists in partitioning a given data set into groups or *clusters* such that the data points belonging to the same group or cluster, will be more similar to each other than points in different clusters. This process is achieved by discovering groups and identifying interesting distributions and patterns in the underlying data in an *unsupervised* manner, that is by forming separated groups of samples within the *complete dataset*. In this scenario, given data $\mathcal{D} = \{\mathbf{x}_i^n\}$ the goal is to learn a function $f : \mathcal{X} \to \{1, \ldots, k\}$, which assigns each object $\mathbf{x}_i$ to a corresponding cluster $c \in \{1, \ldots, k\}$. Objects assigned to the same cluster have a high within-cluster similarity and at the same time low similarity to all other objects not in the same cluster. The data set $\mathcal{D}$ is then broken down into a number of groups $\mathcal{C} = \{\pi_1, \ldots, \pi_k\}$ which are significantly more uniform than the overall dataset.

The assumption behind traditional linear clustering techniques is that the data space consists of elliptical regions. Consequently, these methods can not detect clusters that are non-linearly separable in the input space. The problem of non-linear separability of classes, can be circumvented by mapping the observed data to a higher dimensional space in a nonlinear way so that each cluster for each class unfolds into a simple form. A first approach to tackle non-linearity is kernel $k$-means [9, 35], where points are first mapped to a higher-dimensional feature space using a non-linear function, and the data points are then partitioned in this new space. *Spectral clustering* [3, 4, 24, 29] is another alternative that has strong connections with graph theory [7]. The core algorithm uses the eigenvectors of an affinity matrix to group the data points. These two methodologies can nevertheless be formulated in terms of kernel functions. Introducing these techniques in microarray data analysis allows for dealing with both high-dimensional data and nonlinear relationships in the data, since the kernel trick enables computations for high-dimensional input spaces.

## 1.1 The kernel *k*-means algorithm

Kernel $k$-means clustering can be considered as a generalization of the $k$-means clustering algorithm. As usual, we assume our original data to be mapped to some feature space $\mathcal{F}$ via the function $\varphi(\cdot) : \mathcal{X} \to \mathcal{F}$. Hereby, we follow the formulation earlier exposed in [26]. Given a set of data points $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, where each data point is represented in a $d$ dimensional space, a smooth, continuous non-linear mapping $\varphi(\cdot)$ from the input space to the feature space, the goal is then to partition the data set into $k$ disjoint cluster sets $\boldsymbol{\mathcal{C}} = \{\pi_c\}_{c=1}^k$, while minimizing

the within-cluster sum of squares, that is:

$$\mathcal{J}^{\varphi} = \text{argmin}_{\pi_1,\ldots,\pi_k} \sum_{c=1}^{k} \sum_{i=1}^{n} I_{ic} \|\varphi(\mathbf{x}_i) - \boldsymbol{\mu}_c^{\varphi}\|_2^2 \ , \tag{1}$$

where $I_{ic} = 1$ if $\mathbf{x}_i \in \pi_c$ and zero otherwise. For each cluster, $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k$ represent the centers or centroids. Contrary to $k$-means, here the centroids are implicitly defined in the feature space as

$$\boldsymbol{\mu}_c^{\varphi} = 1/|\pi_c| \sum_{i=1}^{n} I_{ic} \varphi(\mathbf{x}_i) \ ,$$

where $|\pi_c|$ denotes the size of clusters $\pi_c$. Application of the kernel trick results in

$$\|\varphi(\mathbf{x}_i) - \boldsymbol{\mu}_c^{\varphi}\|_2^2 = K(\mathbf{x}_i, \mathbf{x}_i) - \frac{2}{|\pi_j|} \sum_{j=1}^{n} I_{jc} K(\mathbf{x}_i, \mathbf{x}_j) + \frac{1}{|\pi_c|^2} \sum_{j,l=1}^{n} I_{jc} I_{lc} K(\mathbf{x}_j, \mathbf{x}_l) \ . \tag{2}$$

The key issue in extending traditional $k$-means to kernel $k$-means simplifies to computation of distance in the kernel induced space. Assigning a data point $\mathbf{x}_i$ to a cluster is

$$I_{ic} = \begin{cases} 1 & \|\varphi(\mathbf{x}_i) - \boldsymbol{\mu}_c^{\varphi}\|_2^2 \leq \|\varphi(\mathbf{x}_i) - \boldsymbol{\mu}_q^{\varphi}\|_2^2, \forall c \neq q, c, q = 1, \ldots, k \\ 0 & \text{otherwise} \ . \end{cases} \tag{3}$$

With respect to standard $k$-means clustering, the general structure of algorithm is preserved in the kernel version. Nevertheless, there are two main differences between both schemes, namely the non-linear mapping via the kernel trick and the lack of an explicit centroid in the feature space. The mapping from the feature space *back* to the input space is called the *pre-image* problem. Typically the exact pre-image of a given $\varphi(\mathbf{x})$ does not exist and therefore one can only approximate it [19]. Instead, we consider a pseudo-centroid defined as the data point that is closest to the centroid in the feature space [26], i.e.

$$\hat{\boldsymbol{\mu}}_c = \text{argmin}_{\mathbf{x}_i \in \pi_c} \|\varphi(\mathbf{x}_i) - \boldsymbol{\mu}_c^{\varphi}\|_2^2 \ , (1 \leq c \leq k) \ . \tag{4}$$

Similar to standard $k$-means, convergence to the global optimum for kernel $k$-means is not guaranteed. The non-convexity nature of the optimization function $\mathcal{J}^{\varphi}$ is sensible to the initialization. Multiple runs are thus often required.

## 1.2 Spectral clustering

To overcome the non-convexity in greedy $k$-means algorithms, spectral clustering methods relax the original problem in a way that global optimal solutions can be found. Spectral clustering techniques can be regarded as relaxations of graph cut problems, with nodes representing data points and edges the pairwise similarity among them. Clustering then corresponds to partitioning the nodes in the graph into groups. Such a division of the graph nodes in two disjoint sets is called a *graph cut* [7].

In the special case of a partitioning of the data into two groups, we aim to learn a clustering function $f : X \to \{+1, -1\}$. The set of data points is represented by graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and similarity matrix $\mathbf{W}_{ij} = w_{ij} \geq 0$, where $w_{ij}$ can be interpreted as the weights on the edges connecting nodes $i$ and $j$ in a graph. For $w_{ij} = 0$, there is no edge from $i$ to $j$, i.e. $\mathbf{W}$ plays the role of an adjacency matrix often taken as the RBF kernel $K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{W}_{ij} = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|_2^2/\sigma^2)$.

In this way, the problem of clustering consists in minimizing the cost of *cutting* the graph into $k$ disjoint sets $\pi_1, ..., \pi_k$. Let $d_i = \sum_j w_{ij}$, called the degree of node $i$, and $\mathbf{D} \in \mathbb{R}^{n \times n}$ a diagonal matrix with $\mathbf{D}_{ii} = d_i$. Furthermore, let $\mathbf{y} = [y_1, \ldots, y_n]^\top \in \{+1, -1\}^n$ a vector of labels that are assigned to the data items by the clustering algorithm. Then the cut-cost of splitting the graph in two partitions according to a given clustering is given as

$$\sum_{y_i \neq y_j}^{n} w_{ij} = \frac{1}{2}\left(\sum_{i,j=1}^{n} w_{ij} - \sum_{i,j=1}^{n} y_i y_j w_{ij}\right) = \frac{1}{2}\left(\mathbf{1}^\top \mathbf{W}\mathbf{1} - \mathbf{y}^\top \mathbf{W}\mathbf{y}\right) = \frac{1}{2}\mathbf{y}^\top (\mathbf{D}–\mathbf{W})\mathbf{y} \ . \tag{5}$$

In order to favor balanced clusters Shi and Malik [29] proposed the so-called normalized cut NCUT. Minimizing *cut*-like costs is known to be NP complete [7]. Nevertheless, allowing $\mathbf{y}$ to take real values, an optimal $\mathbf{y}^*$ can be computed as the solution of the following optimization problem:

$$\begin{aligned} \min_{\mathbf{y}} \quad & \mathbf{y}^\top (\mathbf{D} - \mathbf{W})\mathbf{y} \\ \text{s. t.} \quad & \mathbf{y}^\top \mathbf{D}\mathbf{y} = 1 \ . \end{aligned} \tag{6}$$

The relaxed solution corresponds to the eigenvector associated to the second smallest eigenvalue of the generalized eigenvector problem

$$(\mathbf{D} - \mathbf{W})\mathbf{y} = \lambda \mathbf{D}\mathbf{y} \ . \tag{7}$$

Clustering is then performed by thresholding $\mathbf{y}^*$ appropriately. The matrix $L = \mathbf{D} - \mathbf{W}$, is the so-called Laplacian matrix of the graph. The top $k$ eigenvectors

provide relaxed clustering indicators $\mathbf{y} \in \mathbb{R}^n$. In this setting, however, there is no clear way as how test points (out-of-sample) should be assigned to the initial clusters, since the embedding eigenvectors are only defined for the full data set. In [13] the authors employed the Nyström method in order to approximate eigenvectors and reduce the computational load for large scale applications. In contrast to this approach, the authors in [3,4] present the spectral clustering methods within a primal-dual optimization framework. This way, the clustering model obtained can be readily applied to new data. Such formulation arises from the context of Weighted Kernel PCA and LS-SVMs [31]. Directions of maximal variance provide a nonlinear embedding where clusters become tighter. Within this formulation one aims at finding directions with maximal variance

$$\begin{aligned} \max_{\mathbf{w},\mathbf{z}} \quad & \gamma \frac{1}{2}\mathbf{z}^\top \mathbf{V}\mathbf{z} - \frac{1}{2}\mathbf{w}^\top \mathbf{w} \\ \text{s.t.} \quad & \mathbf{z} = \Phi \mathbf{w} \ , \ \mathbf{V} = \mathbf{V}^\top > 0 \ , \end{aligned}$$

where $\Phi = [\varphi(\mathbf{x}_1)^\top; ...; \varphi(\mathbf{x}_n)^\top]$. An eigenvalue problem is obtained after elimination of primal variables

$$\mathbf{V}\mathbf{\Omega}\boldsymbol{\alpha} = \lambda \boldsymbol{\alpha} \ , \tag{8}$$

where $\mathbf{\Omega}_{ij} = K(\mathbf{x}_i, \mathbf{x}_j) = \varphi(\mathbf{x}_i)^\top \varphi(\mathbf{x}_j)$ and $\lambda = 1/\gamma$. If $\mathbf{V} = \mathbf{D}^{-1}$, one obtains the random walk spectral clustering [20]. For a new test point $\mathbf{x}$, its projection

$$z(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i K(\mathbf{x}_i, \mathbf{x}) \ ,$$

defines the extension to out-of-sample points. To project on more eigenvectors

$$z^{(r)}(\mathbf{x}) = \sum_{i=1} \alpha_i^{(r)} K(\mathbf{x}_i, \mathbf{x}), \ \ r = 1, \ldots, k \ . \tag{9}$$

The top $k$ eigenvectors $\boldsymbol{\alpha}$, of $\mathbf{D}^{-1}\mathbf{\Omega}$ provide an embedding that contains the clustering information of the original samples. One of the main advantages of this formulation is the possibility to evaluate the model on out-of-sample points. This is important for predictive capabilities, model selection and even large-scale data analysis. Other techniques such as the Nyström method approximate the eigenvectors for out-of-sample points. In general, the relaxed solutions provided by spectral clustering methods are real-valued and do not provide cluster indicators. In [29], the authors propose to apply the normalized cut in a recursive way, that is the current partition is subsequently divided if the NCUT is below some prespecified value. The process is repeated until $k$ clusters have been found. Another

alternative to obtain $k$ clusters is called *re-clustering*. This approach consists of computing the top $k$ eigenvectors of the eigenvalue problem in (7) or (8) and then performing $k$-means onto the eigenvector space. The general approach of spectral methods for clustering of genes expression data is summarized in Figure 1.

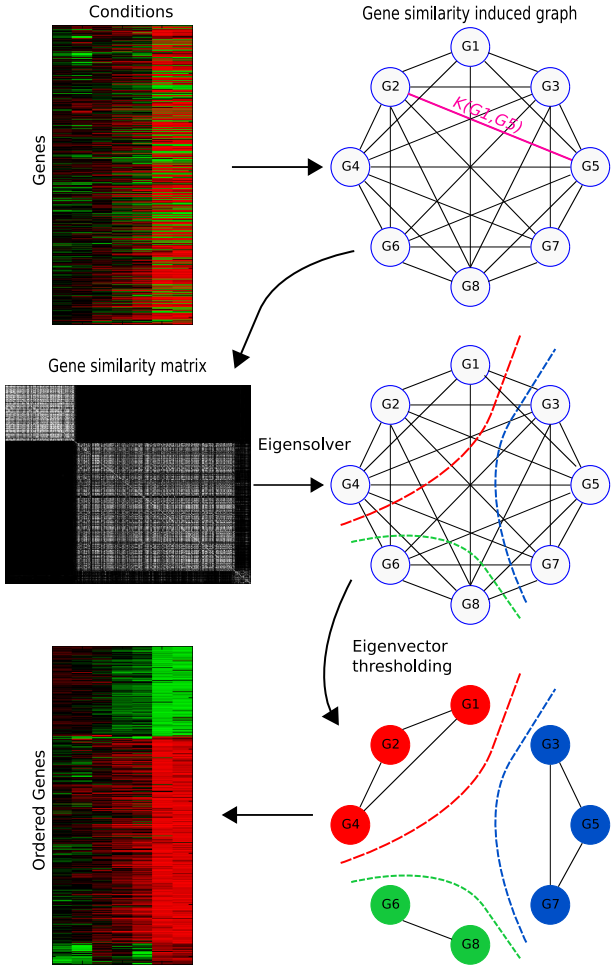


Figure 1: *Top row:* Gene similarities induce a graph. *Middle row:* Genes embedding via $K(\cdot, \cdot)$ similarity function. *Bottom row:* Eigenvectors or score variables provide clustering.

## 2 Selecting informative samples

A major drawback in kernel-based learning is that the amount of computation required to find the solution scales with the number of samples $n$. Hence, using

the entire sample $\mathcal{D}$ to compute the clustering model becomes prohibitive for large values of $n$. Approximation to the eigendecomposition of the Gram matrix can be computed by the Nyström method (which is used for the numerical solution of eigenproblems) [33]. This is achieved by carrying out an eigendecomposition on a smaller system of size $m \ll n$, and then expanding the results back up to $n$ dimensions.

In the context of estimation in the primal space [31, 32], it has been motivated to use a subsample $m \ll n$ to compute an approximation of the feature map $\hat{\varphi}$. In general, the selection of the subsample of size $m$, is performed before the model estimation. Here, instead of directly approximating $\hat{\varphi}$, we aim at selecting a subset of informative data points (genes) to construct the clustering model in (8). This approach leads to a reduced eigenvalue problem. In order to select the working subsample, we seek to maximize the quadratic Renyi $H_R$ [31]. In this case, given a fixed-size $m$, the goal is to select informative genes maximizing

$$\begin{aligned} H_S &= -\log \int p(x)^2 \mathrm{d}x \\ H_S &\approx H_R = \int \hat{p}(x)^2 \mathrm{d}x = \frac{1}{m^2} \mathbf{1}^\top \mathbf{\Omega}_h \mathbf{1} \ . \end{aligned} \tag{10}$$

The use of this active selection procedure can be quite important for large scale problems, as it is related to the underlying density distribution of the sample. Here, $\mathbf{\Omega}_h$ is a $m \times m$ Parzen kernel estimator of width $\hat{h}$. The entropy criterion ensures that the selected subsample is spread over the entire data region and not only concentrated on a certain area of the data set. The kernel bandwidth of the entropy based selection criterion is determined using a plug-in method as described in [8]. Criterion (10) can be iteratively maximized through the greedy strategy summarized in Algorithm 1.

It has been extensively reported that this strategy performs far better than random picking, often requiring a smaller working set [8, 11, 31]. Alternatively, similar to Radial Basis Networks (RBF) applications [23, 25], we also consider the $k$-means algorithm to select the working subsample. That is, first we perform $k$-means clustering on the input space and secondly use the centroids (such as in equation (4)) as the working subsample $\mathcal{A}$. The final proposed spectral clustering algorithm combined with working subset is described in Algorithm 2. For the re-clustering phase, we follow the $k$-means implementation in [10] which uses the triangle inequality to speed up computations[1].

[1] Matlab code available at `http://cseweb.ucsd.edu/~elkan/fastkmeans.html`

**Algorithm 1** Subsample selection maximizing Renyi's entropy.

**Input:** Training set $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^n, \mathbf{x}_i \in \mathbb{R}^d$, size of the subsample $m$.
**Output:** Active set $\mathcal{A} = \{\mathbf{x}_j\}_{j=1}^m \subset \mathcal{D}$.

Select randomly an initial active set of size $m$, $\mathcal{A} = \{\mathbf{x}_j\}_{j=1}^m \subset \mathcal{D}$.
**repeat**
  Compute $H_R(\mathcal{A})$.
  Randomly pick two data points as $\mathbf{x}_* \in \mathcal{A}$, and $\mathbf{x}_+ \in \mathcal{D} \setminus \mathcal{A}$
  Let $\mathcal{W} = \{\mathcal{A} \setminus \{\mathbf{x}_*\}\} \cup \{\mathbf{x}_+\}$
  **if** $H_R(\mathcal{W}) > H_R(\mathcal{A})$ **then**
    swap$(\mathbf{x}_+, \mathbf{x}_*)$
  **end if**
**until** change in $H_R$ value is too small
**return** $\mathcal{A}$

**Algorithm 2** Spectral clustering

**Input:** Training set $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^n, \mathbf{x}_i \in \mathbb{R}^d$, active set $\mathcal{A} = \{\mathbf{x}_j\}_{j=1}^m \subset \mathcal{D}$. Number of clusters $k$. Kernel parameter $\sigma^2$.
**Ouput**: Clustering partitions $\mathcal{C} = \{\pi_c\}_{c=1}^k$ .

Using $\mathcal{A}$, compute kernel matrix $\mathbf{\Omega}_{jl} = K(\mathbf{x}_j, \mathbf{x}_l) \in \mathbb{R}^{m \times m}, j, l = 1, \ldots, m$ .
Obtain top $k$ eigenvectors $\boldsymbol{\alpha}^{(r)} \in \mathbb{R}^{m \times 1}, r = 1, \ldots, k$ from (8).
$\forall \mathbf{x}_j \in \mathcal{A}$, compute the score variables $z_j^{(r)} = \sum_{l=1}^m \alpha_l^{(r)} K(\mathbf{x}_l, \mathbf{x}_i)$ .
Denote $\mathbf{z}_j = [z_j^{(1)} \ldots z_j^{(k)}]$, as the score variables space $\mathbf{Z} = [\mathbf{z}^{(1)} \ldots \mathbf{z}^{(k)}] \in \mathbb{R}^{m \times k}$ .
Perform standard $k$-means on $\mathbf{Z}$ to obtain cluster centroids $\boldsymbol{\mu}_c \in \mathbb{R}^k$, and partitions $\mathcal{C} = \{\pi_1, \ldots, \pi_k\}, c = 1, \ldots, k$ .
For out-of-sample (test) points, $\mathbf{x}_i \in \mathcal{D}$, obtain $\hat{z}_i^{(r)} = \sum_{j=1}^m \alpha_j^{(r)} K(\mathbf{x}_j, \mathbf{x}_i)$ .
Assign $\mathbf{x}_i$ to $\pi_{c^*} \in \mathcal{C}$, where

$$c^* = \text{argmin}_c \|\hat{\mathbf{z}}_i - \boldsymbol{\mu}_c\|_2^2 \ .$$

**return** $\mathcal{C} = \{\pi_c\}_{c=1}^k$ .

## 2.1 Model selection

Opposite to standard clustering settings, we consider in all our simulations a full learning framework. That is, we made a clear distinction between training, validation and test phases. The number of clusters $k$ and the kernel parameter $\sigma^2$ are chosen such that the resulting partition of data points into clusters $\mathcal{C} = \{\pi_1, \ldots, \pi_k\}$ is optimal with respect to a certain index. To validate our results we compute the

Caliński and Harabasz (CH) index [6], which is reported to perform consistently among several other indexes [22]. The CH index is defined as:

$$\text{CH}(k) = \frac{\text{trace}(\mathbf{S}_B)/(k-1)}{\text{trace}(\mathbf{S}_W)/(n-k)} \ , \tag{11}$$

where $\mathbf{S}_B$ represents the between-cluster dispersion matrix and $\mathbf{S}_W$ the within-group dispersion matrix. The value of $k$, which maximizes $\text{CH}(k)$, is regarded as specifying the number of clusters [12].

Contrary to the standard approach in clustering of selecting the parameters on the basis of the full data set, we make use of a randomly selected validation set. Using the out-of-sample extension in (9), we can readily compute the CH on the score variables space $\mathbf{Z}$. Additionally, we tune the value of the $\sigma^2$ parameter as well, thus choosing the pair $(\sigma^{2^*}, k^*)$ that maximizes CH on the validation score variables. In fact, a wide number of clustering quality measures [15] can be readily used in combination with the spectral clustering algorithm [26]. The only difference resides in that the computation of such indexes is carried on the projected variables $\mathbf{Z}$ (also on eigenvectors [24, 26]), rather than in the original data set $\mathbf{X}$. In the case of kernel $k$-means, however, clustering indexes can not be directly used as they are defined for finite dimensional spaces. To evaluate clustering indexes in more general feature spaces, one must rewrite them first in terms of dot products. Indeed, we have previously extended the Caliński and Harabasz and silhouette score [28] to more generalized versions in terms of kernel evaluations [26].

# 3 Experiments

In order to test the proposed methodology, we first create a set of toy examples both including linear and nonlinear clustering problems. Such problems allow us to observe the impact that the subsampling method, subset size $m$, number of clusters $k$ and kernel parameter $\sigma^2$, have on the clustering output and to refine the general approach. Additionally, we test our methodology on a standard yeast microarray data set [5] to cluster genes based on their expression value. In general we will observe that we can set $m < n$ without any significant decrease in the accuracy and quality of the clustering solution.

## 3.1 Toy data

We consider first a two-dimensional artificially generated data set with $k = 15$ pre-defined clusters and $n = 5000$ total number of data points. This data set (S1)[2] presents a varying complexity in terms of data distribution, overlap and noise [14]. Secondly, we vary the subsample size $m \in \{100, 300, 500, 1000\}$ and the number of clusters in the range $k = 2, \ldots, 20$. Finally, we calculate the average maximum CH index over 20 randomizations on validation data. In Figure 2, the original data set S1 is shown, along with the entropy sampled data points. Clustering partitions $k = 2, 3, 15$, with maximum CH index are also illustrated. Optimal values of the CH index are consistently attained $k = 2, 3, 15$, hence indicating the multi-scale nature of the data set. After $k = 15$ clusters, there is an evident decrease in the index performance.

## 3.2 Non-linear cluster structures

In order to illustrate the benefit of using a kernel-based method and quantify whether the entropy subsampling is effective or not, we consider a classical non-linear data set in spectral clustering. The data set consists of a highly dense Gaussian cloud enclosed by two concentric rings-type clusters of lower density. We refer to this data set as CLOUD&RINGS. In the first stage, we vary the subsample size from $m \in \{100, 300, 500, 1000\}$ from a total number of $n = 5000$ data points. Finally, we compare three subsampling methods namely: random, $k$-means and entropy, reporting performance in terms of the CH index and the adjusted rand index (ARI) with respect to the true clusters [17]. The expected value of the adjusted Rand index has value zero and the maximum value of the adjusted Rand index is 1, that is a perfect agreement between the produced clustering and the reference. Table 1 summarizes the obtained results and Figure 3 represents the data structure along with the subsampled data points.

Clearly the entropy selection provides a superior estimation of the underlying data distribution, therefore allowing a better final clustering. The selected data points are nicely spread over the data set. Interestingly, the highly dense Gaussian cloud requires few number of data points to be represented, hence we could think of it as a compressed codebook. On the contrary, random and $k$-means subsampling are somewhat fooled by the Gaussian cloud as they sample almost all the data points from this high density region. These two approaches fail thus at capturing the structure of the two rings (cf. Figure 3).

[2] `http://cs.joensuu.fi/sipu/datasets/`

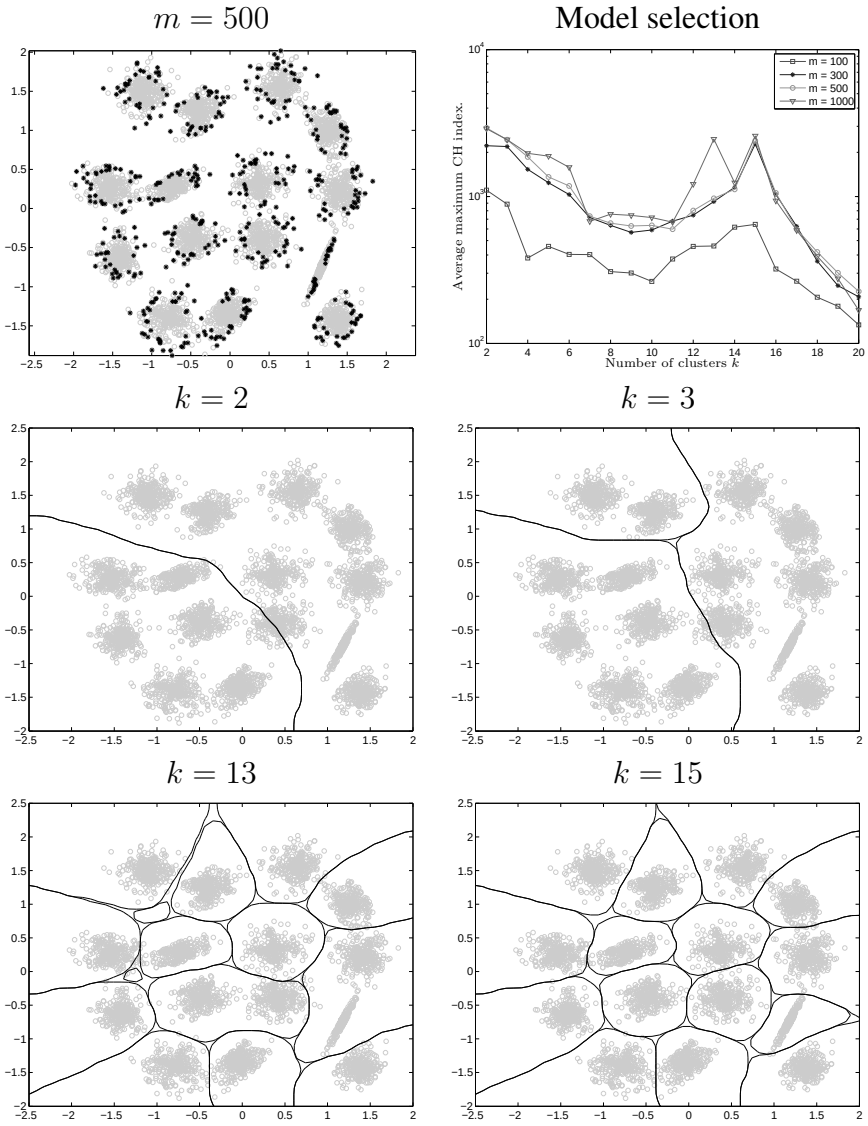


Figure 2: S1 data set clustering results. **Top row:**(*left*) Full data set and entropy sampling with $m = 500$ data points. (*right*) Average maximal CH index for varying subsample size $m$ and number of clusters $k$. Optimal values are consistently attained at $k = 2, 3, 15$ clusters. Performance falls clearly off after $k = 15$ clusters. **Middle row:** Cluster partitions with average maximum CH index for $k = 2, 3$. **Bottom row:** Cluster partitions with average maximum CH index for $k = 13, 15$.

## 3.3 Simulated and real data gene expression data

In order to test our kernel-based clustering approach, we consider a simulated data and a yeast galactose microarray data set that have been extensively reported

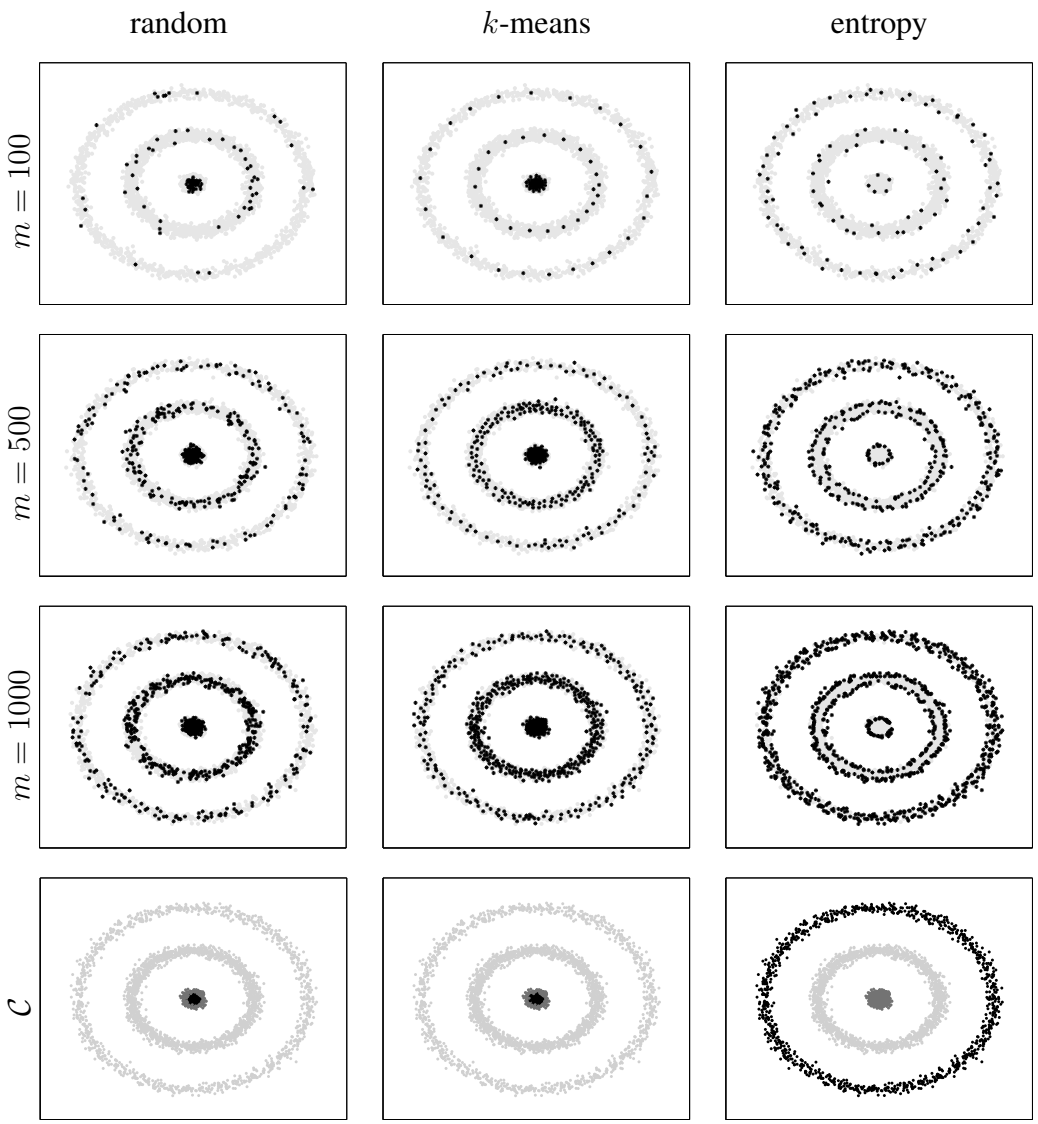


Figure 3: Comparison of three subsampling schemes across different subsample size $m$ on the CLOUD&RINGS problem. **Bottom row:** Best clustering result for each subsampling scheme.

| $m$ | random | | $k$-means | | entropy | |
|---|---|---|---|---|---|---|
| | CH | ARI | CH | ARI | CH | ARI |
| 100 | 1.86E+05 | 0.531 | 2.16E+05 | 0.582 | 2.28E+05 | 1 |
| 300 | 2.17E+05 | 0.507 | 1.39E+05 | 0.588 | 3.04E+05 | 1 |
| 500 | 1.96E+05 | 0.566 | 2.1E+05 | 0.571 | 4.16E+05 | 1 |
| 1000 | 2.1E+05 | 0.561 | 1.99E+05 | 0.581 | 4.43E+05 | 1 |

Table 1: Numerical performance on the CLOUD&RINGS problem with with $n = 5000$, $k = 3$ and $\sigma^2 = 1.8$. The entropy subsampling attains a perfect agreement (ARI = 1) with as few as $m = 100$ samples. Neither random subsampling nor $k$-means reach good performance.

in previous studies. The simulated data (SIMULATED1000) set consists of four clusters of 1000 genes under 100 conditions [21]. The yeast galactose microarray data set (YEASTG) represents a time series showing the expression profiles of yeast growing with 205 genes measured under 20 different perturbations for the GAL pathway [18]. Given that information over the true partitions is known, we employ the adjusted rand index (ARI) to compare our clustering approach against previously reported methods [34]. Table 2 summarizes the results of the synthetic data set for different subsample sizes. We report the CH index for both the entropy subsample and the final projected set of genes.

| $m$ | CH subsample | CH projected | ARI |
|---|---|---|---|
| 100 | 2.4E+10 | 3.3E+11 | 0.99 |
| 200 | 4.9E+10 | 3.3E+11 | 0.99 |
| 300 | 7.4E+10 | 3.3E+11 | 0.99 |
| 400 | 9.9E+10 | 3.3E+11 | 0.99 |
| 500 | 1.2E+11 | 3.3E+11 | 0.99 |

Table 2: SIMULATED1000 data set. CH index for the entropy subsample and the full projected set of genes and ARI. The size of the subsample $m$ is varied, whence only 100 entropy selected genes suffice to attain an almost perfect agreement (ARI = 0.99) with respect to the true partition.

Similarly to the simulated data case, we examine the effect of the subsample size on the YEASTG microarray data set. Again, values for the CH index in both the entropy selected genes and the full set of projected genes are reported along with the ARI scores. Results are summarized on Table 3.

| $m$ | CH subsample | CH projected | ARI |
|---|---|---|---|
| 50 | 58 | 1.5E+02 | 0.14 |
| 100 | 2.1E+03 | 1.5E+04 | 0.92 |
| 150 | 3E+03 | 4.6E+03 | 0.92 |
| 200 | 1.7E+04 | 1.1E+04 | 0.92 |

Table 3: YEASTG microarray data set. CH index for the entropy subsample and the full projected set of genes and ARI. The size of the subsample $m$ is varied, whence 100 entropy selected genes suffice to attain an adjusted rand index ARI = 0.92 with respect to the true partition.

Comparison with other clustering algorithms [34] are reported in Table 4. Note, however, that we employed only half of the genes to construct the clustering model. In order to illustrate the final ordering of the genes and reveal the cluster

| | Simulated (1000, 100, 4) | Yeast galactose (205, 20,4) |
|---|---|---|
| proposed | 0.99 (4) | 0.92 (4) |
| CLIC | 1 (4) | 0.97 (4) |
| $k$-means | 0.68 (4) | 0.87 (4) |
| HPCluster | 1 (4) | 0.83 (4) |
| CRC | 0.90 (6) | 0.97 (4) |
| MCLUST | 1 (4) | 0.97 (4) |
| $k$-boost | 0.72 (3) | 0.95 (4) |
| CLICK | NA (1) | 0.81 (2) |

Table 4: Adjusted rand indexes of several clustering algorithms for the SIMULATED1000 data set and the YEASTG microarray data set.

structure, the affinity (kernel) matrices of the entropy selected genes and the full set of projected are shown in Figure 4. On the left hand side, every row represents each of the 100 entropy selected genes used to build the spectral clustering model. Genes assigned to each of the 4 clusters identified by spectral clustering are shown on the right hand side. We appreciate two prominent clusters, whence particularly cluster (4) reveals a large homogeneous group of under-expressed genes.

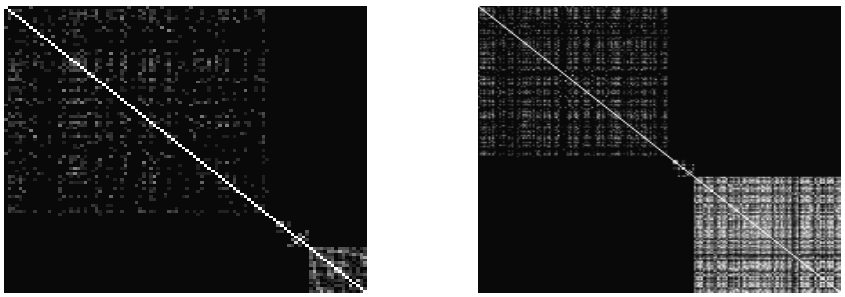

Figure 4: Affinity matrices revealing the clustering structure that is preserved both in the entropy selection phase and the extension to the full group of genes. **Left:** Entropy selected genes. **Right:** Full set of genes. The lower block represents the homogeneous cluster (4) of under-expressed genes.

## 3.4 The NCI60 data set

An example study is presented that uses NCI60 cell line data [27]. This large-scale data set consists of $n = 22283$ gene probes and $d = 108$ samples collected from various human cancer cell lines. As suggested by the authors, before clustering we centred in the gene direction as to make the subsequent analysis independent of the amount of each gene's mRNA in the reference pool. In a first attempt to cluster this large dataset, we let the subsample size to take values $m \in \{100, 200, 500, 1000, 2000, 3000\}$, and construct a grid for the parameters $k \in [2, ..., 20]$, $\sigma \in [10^{-3}, ..., 500] \times \sqrt{d}$. Under these settings, we report in Figure 5, $CH$ values averaged over 20 randomizations.

The monotonic decrease of the average CH values indicates the presence of two large prominent groups of genes. In turn, these results suggest that a global clustering approach might not be enough to reveal other smaller clusters. Moreover, given the relatively noisy character of microarray expression data, the more subtle clusters are likely to be indistinguishable from the noise. To this end, we adapt our algorithm to work in a hierarchical divisive fashion, that is, starting from the full set of genes we recursively partition it into smaller groups forming a tree. At each node of this tree, we construct first a subsample using the entropy method, tune the parameters and finally cluster the remaining genes. A maximum depth of 10 levels is fixed and we restrict our search to a maximum of number of clusters $k_{\max} = 5$. Since clustering of the data occurs at different levels, and thus different resolutions, we reduce the range for $\sigma$ at each level. The size of the subsample is set to half of the samples at each level with $m_{\min} = 100$ and $m_{\max} = 1000$ as extreme values. Again, the best pair $(k, \sigma)$ is selected according to the average

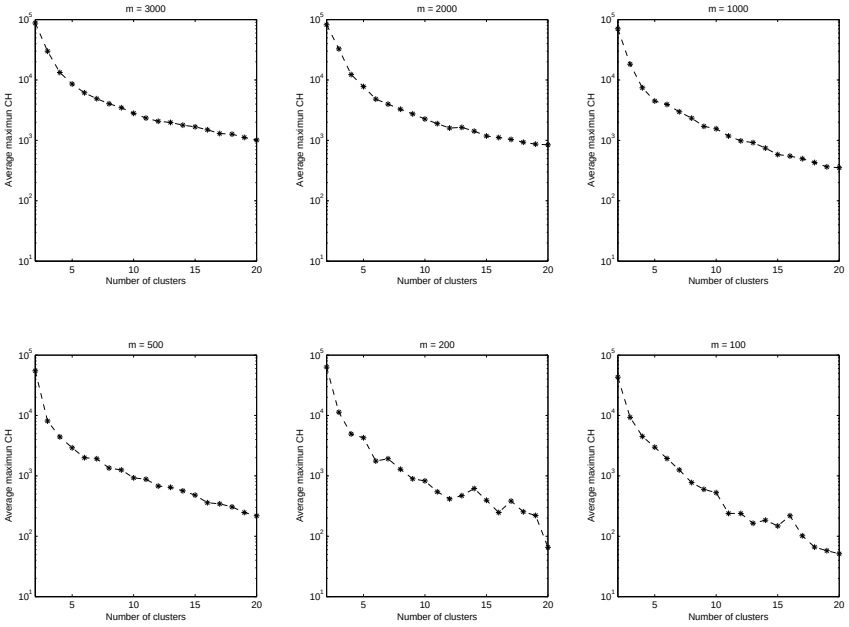


Figure 5: NCI60 microarray data set. We report the average CH index with respect to the number of clusters $k$ and the subsample size $m \in \{100, 200, 500, 1000, 2000, 3000\}$. In all cases, the CH index peaks at $k = 2$, hence suggesting the presence of large groups of genes organized in a hierarchical structure.

maximum CH over 20 randomizations.

Our procedure delivers a hierarchical tree with 40 leaves. At each level the average maximum CH was once more attained at $k = 2$, hence we end up having a binary tree. Figure 6 displays the tree organization in the form of a dendrogram. Leave nodes indicate each of the 40 clusters found by our method, with the top-most node representing the full data set. The lower panel illustrates the distribution of the genes into the 40 clusters. We can readily appreciate three big clusters accounting for about 33% of the data set.

Further, we narrow down our analysis to those cluster groups containing less than 200 genes and perform a standard functional enrichment analysis using the web-tools DAVID[3] [16] and Babelomics[4] [1,2]. The most statistically significant terms involving biological processes are listed in Tables 5 and 6. Notice that the clusters

[3] `http://david.abcc.ncifcrf.gov/`

[4] `http://babelomics.bioinfo.cipf.es/`

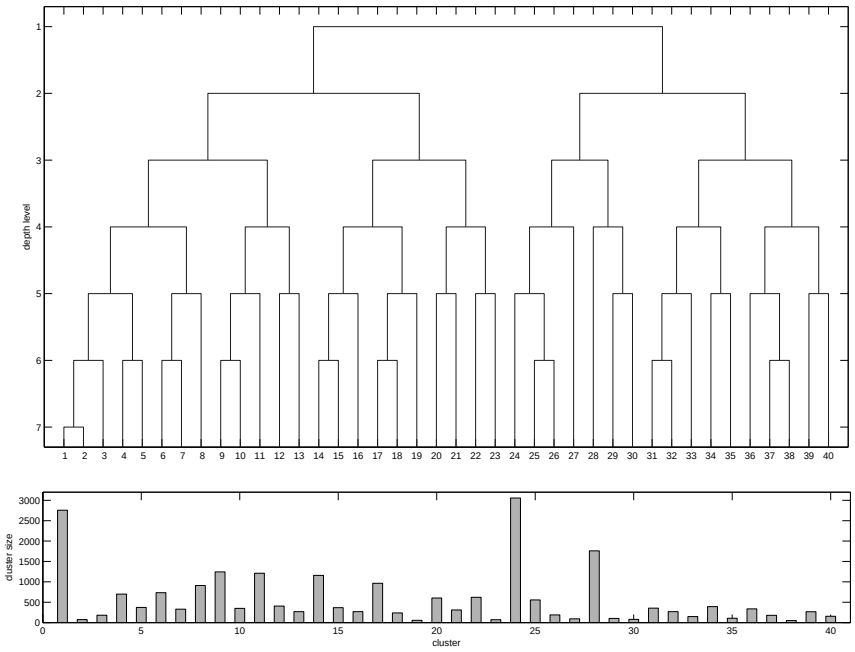


Figure 6: NCI60 data set. **Top.** Dendrogram representation of our hierarchical approach. At each level the optimal number of partitions was found to be $k = 2$. The result is then a binary tree with a total of 40 clusters on the leaves. **Bottom.** Distribution of the genes across the 40 clusters. Three major clusters can be easily distinguished as they account for about 33% of the data set.

analysis reveals meaningful GO annotation associations and, in general, they tend to contain genes which represent specific biological processes. For example, the genes which are members of cluster 38 are involved in immune response mediated by leukocyte activation. Likewise, the members of cluster 23 are also involved in immune response, however, their mode of action is more specific to antigen mediated immune response, as is evident from inspection of the terms presented in Table 5. Cluster 19 contains genes which are associated almost exclusively with neurological processes such as *neuron differentiation* and *cell communication*. Cluster 30 is composed of genes which are predominately involved in the cell cycle and its regulation. The genes in Cluster 27 are involved in g-protein coupled signaling, whereas cluster 35 clearly represents genes exclusively associated to metabolism and catabolic processes.

Finally, we look at the terms in cluster 38 involving immune response and leukocyte activation, and explore how they relate to each other in the gene ontology

annotation. The corresponding tree structure is presented in Figure 7.

# 4 Summary

With the ongoing availability of large microarray data sets, clustering algorithms should move also forward to reduce large computational complexity and memory requirements. This study presents a modified spectral clustering approach, which handles large data sets and it is not limited to microarray data sets. Our method is based on a novel concept in which informative genes (or data points) are first selected using an entropy criterion. This selection step leads to a more compact and better representation of the full data set than a random selection. The entropy genes, in turn, form the support set upon which the spectral clustering model is developed. Moreover, in order to detect subtle clusters in large scale data sets and therefore avoid forming oversimplified clusters, we adapt our methodology to work in a hierarchical fashion with clusters being refined at each level.

# References


[1] F. Al-Shahrour, J. Carbonell, P. Minguez, S. Goetz, A. Conesa, J. Tárraga, I. Medina, E. Alloza, D. Montaner, and J. Dopazo. BABELOMICS: advanced functional profiling of transcriptomics, proteomics and genomics experiments. *Nucleic Acids Res*, 36(Web Server issue):W341–6, July 2008.

[2] F. Al-Shahrour, P. Minguez, J.M. Vaquerizas, L. Conde, and J. Dopazo. BABELOMICS: a suite of web tools for functional annotation and analysis of groups of genes in high-throughput experiments. *Nucleic Acids Res*, 33(Web Server issue):W460–4, July 2005.

[3] C. Alzate and J.A.K. Suykens. A weighted kernel PCA formulation with out-of-sample extensions for spectral clustering method. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN'06)*, pages 138–144, 2006.

[4] C. Alzate and J.A.K. Suykens. Multiway spectral clustering with out-of-sample extensions through weighted kernel pca. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(2):335–347, 2010.

[5] M.P.S. Brown, W. Noble, D. Lin, N. Cristianini, C.W. Sugnet, T.S. Furey, M. Ares, and D. Haussler. Knowledge-based analysis of microarray gene

expression data by using support vector machines. *Proceedings of the National Academy of Sciences*, 97(1):262–267, 2000.

[6] R.B. Caliński and J. Harabasz. A dendrite method for cluster analysis. *Communications in Statistics*, 3:1–27, 1974.

[7] F.R.K. Chung. *Spectral Graph Theory (CBMS Regional Conference Series in Mathematics, No. 92)*. American Mathematical Society, February 1997.

[8] K. De Brabanter, J. De Brabanter, J.A.K. Suykens, and B. De Moor. Optimized fixed-size kernel models for large data sets. *Comput. Stat. Data Anal.*, 54:1484–1504, June 2010.

[9] I.S. Dhillon, Y. Guan, and B. Kulis. Kernel k-means: spectral clustering and normalized cuts. In *KDD '04: Proceedings of the 2004 ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 551–556, New York, NY, USA, 2004. ACM Press.

[10] C. Elkan. Using the triangle inequality to accelerate k-means. In *ICML*, pages 147–153, 2003.

[11] M. Espinoza, J.A.K. Suykens, and B. De Moor. Fixed-size least squares support vector machines: A large scale application in electrical load forecasting. *Computational Management Science*, 3(2):113–129, April 2006.

[12] B.S. Everitt, E. Landau, and M. Leese. *Cluster Analysis*. Arnold, London, 2001.

[13] C. Fowlkes, S. Belongie, F. Chung, and J. Malik. Spectral grouping using the nyström method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, page 2004, 2004.

[14] P. Fränti and O. Virmajoki. Iterative shrinking method for clustering problems. *Pattern Recogn.*, 39:761–775, May 2006.

[15] M. Halkidi, Y. Batistakis, and M. Vazirgiannis. On clustering validation techniques. *Journal of Intelligent Information Systems*, 17(2):107–145, 2001.

[16] D.W. Huang, B.T. Sherman, and R.A. Lempicki. Systematic and integrative analysis of large gene lists using david bioinformatics resources. *Nat Protoc*, 4(1):44–57, 2009.

[17] L. Hubert and P. Arabie. Comparing partitions. *Journal of Classification*, 2(1):193–218, 1985.

[18] T. Ideker, V. Thorsson, J.A. Ranish, R. Christmas, J. Buhler, J.K. Eng, R. Bumgarner, D.R. Goodlett, R. Aebersold, and L. Hood. Integrated genomic and proteomic analyses of a systematically perturbed metabolic network. *Science*, 292(5518):929–34, May 2001.

[19] J.T. Kwok and I.W. Tsang. The pre-image problem in kernel methods. *IEEE Trans Neural Networks*, 15(6):1517–25, November 2004.

[20] M. Meila and J. Shi. A random walks view of spectral segmentation. In *AISTATS*, 2001.

[21] D.J. Michaud, A.G. Marsh, and P.S. Dhurjati. eXPatGen: generating dynamic expression patterns for the systematic evaluation of analytical methods. *Bioinformatics*, 19(9):1140–6, June 2003.

[22] G.W. Milligan and M.C. Cooper. An examination of procedures of determining the number of cluster in a data set. *Psychometrika*, 50(2):159–179, 1985.

[23] I. T. Nabney. Efficient training of rbf networks for classification. *Int. J. Neural Syst.*, 14(3):201–208, 2004.

[24] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *In Advances in Neural Information Processing Systems*, volume 14, 2001.

[25] M.J.L. Orr. Regularisation in the selection of radial basis function centres. *Neural Computation*, pages 606–623, 1995.

[26] N. Pochet, F. Ojeda, F. De Smet, T. De Bie, J.A.K Suykens, and B. De Moor. *Kernel clustering for knowledge discovery in clinical microarray data analysis*, chapter III, pages 64–92. Kernel Methods in Bioengineering, Signal and Image Processing. Idea Group Inc., Hershey, Pennsylvania, 2007.

[27] D.T. Ross, U. Scherf, M.B. Eisen, C.M. Perou, C. Rees, P. Spellman, V. Iyer, S.S. Jeffrey, M. Van de Rijn, M. Waltham, A. Pergamenschikov, J.C. Lee, D. Lashkari, D. Shalon, T.G. Myers, J.N. Weinstein, D. Botstein, and P.O. Brown. Systematic variation in gene expression patterns in human cancer cell lines. *Nat Genet*, 24(3):227–35, March 2000.

[28] P. Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *J. Comput. Appl. Math.*, 20:53–65, November 1987.

[29] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[30] P.T. Spellman, G. Sherlock, M.Q. Zhang, V.R. Iyer, K. Anders, M.B. Eisen, P.O. Brown, D. Botstein, and B. Futcher. Comprehensive identification of cell cycle-regulated genes of the yeast saccharomyces cerevisiae by microarray hybridization. *Mol Biol Cell*, 9:3273–3297, 1998.

[31] J.A.K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, and J. Vandewalle. *Least Squares Support Vector Machines*. World Scientific, Singapore, 2002.

[32] C. Williams and M. Seeger. The effect of the input density distribution on kernel-based classifiers. In Pat Langley, editor, *ICML*, pages 1159–1166. Morgan Kaufmann, 2000.

[33] C. Williams and M. Seeger. Using the Nyström method to speed up kernel machines. volume 13 of *Neural Information Processing Systems (NIPS)*, pages 682–688, 2001.

[34] T. Yun, T. Hwang, K. Cha, and G-S. Yi. CLIC: clustering analysis of large microarray datasets with individual dimension-based clustering. *Nucleic Acids Res*, 38(Web Server issue):W246–53, July 2010.

[35] R. Zhang and A.I. Rudnicky. A large scale clustering scheme for kernel k-means. In *ICPR (4)*, pages 289–292, 2002.

| Cluster (size) | Term ($p$-value) |
|---|---|
| cluster 38 (50) | lymphocyte activation (2.2E-4), B cell differentiation (5.9E-3), B cell activation (5.9E-3), T cell activation (5.9E-3), germ cell development (5.9E-3), elevation of cytosolic calcium ion concentration (8.5E-3), cytosolic calcium ion homeostasis (9.9E-5), ameboidal cell migration (9.9E-3). |
| cluster 19 (55) | negative regulation of neuron differentiation (5.4E-3), synaptic transmission (6.7E-3), multicellular organismal process (7.7E-3), cell-cell signaling (9.7E-3), cell communication (1.0E-2), localization (1.1E-2), transmission of nerve impulse (1.2E-2), neurological system process (1.2E-2), anatomical structure morphogenesis (1.4E-2), ion transport (1.5E-2). |
| cluster 23 (71) | antigen processing and presentation of peptide antigen via MHC class I (3.5E-3), antigen processing and presentation of peptide antigen (9.8E-3), antigen processing and presentation of endogenous peptide antigen via MHC class I (3.3E-2), antigen processing and presentation of endogenous peptide antigen (3.3E-2), antigen processing and presentation (4.0E-2), metabolic process (4.0E-2), primary metabolic process (4.3E-2), regulation of cellular metabolic process (4.3E-2), antigen processing and presentation of endogenous antigen (4.4E-2), regulation of primary metabolic process (4.5E-2). |
| cluster 30 (78) | regulation of cell cycle process (1.9E-2), signal transduction (2.1E-2), cell-cell signaling (2.7E-2), cell surface receptor linked signal transduction (3.1E-2), cell communication (3.1E-2), maintenance of protein location in nucleus (3.7E-2), regulation of mitotic cell cycle (3.8E-2), intracellular signaling cascade (5.8E-2), regulation of cell cycle (7.9E-2), protein localization in nucleus (8.0E-2). |

Table 5: NCI60 data set. Most significant functional terms associated to each the clusters. Corresponding $p$-values are indicated between brackets.

| Cluster (size) | Term ($p$-value) |
|---|---|
| cluster 27 (93) | G-protein coupled receptor protein signaling pathway (1.8E-3), cell surface receptor linked signal transduction (3.6E-3), second-messenger-mediated signaling (3.7E-3), secretion (8.1E-3), behavior (2.4E-2), cation homeostasis (3.1E-2), cellular metal ion homeostasis (3.6E-2), G-protein signaling, coupled to cyclic nucleotide second messenger (3.8E-2), immune response (3.9E-2), metal ion homeostasis (4.2E-2). |
| cluster 35 (106) | aminoglycan catabolic process (6.0E-3), polysaccharide catabolic process (1.2E-2), carbohydrate catabolic process (3.8E-2), translational elongation (3.8E-2), chitin metabolic process (4.6E-2), detection of bacterium (4.6E-2), chitin catabolic process (4.6E-2), translation (5.6E-2), aminoglycan metabolic process (5.7E-2), immune response (7.9E-2) |
| cluster 33 (148) | behavior (7.5E-4), monocarboxylic acid metabolic process (1.0E-2), regulation of synaptic transmission (1.2E-2), associative learning (1.5E-2), cellular homeostasis (1.5E-2), homeostatic process (1.6E-2), regulation of secretion (1.6E-2), regulation of transmission of nerve impulse (1.7E-2), carbohydrate metabolic process (1.9E-2), regulation of neurological system process (2.1E-2) |
| cluster 40 (155) | nucleobase, nucleoside, nucleotide and nucleic acid metabolic process (8.7E-17), cellular nitrogen compound metabolic process (1.3E-14), nitrogen compound metabolic process (7.2E-14), cellular macromolecule metabolic process (4.8E-13), macromolecule metabolic process (1.1E-10), cellular metabolic process (5.5E-9), cellular macromolecule biosynthetic process (3.1E-8), primary metabolic process (3.8E-8), macromolecule biosynthetic process (4.0E-8), DNA metabolic process (8.4E-8) |

Table 6: NCI60 data set. Most significant functional terms associated to each the clusters. Corresponding $p$-values are indicated between brackets.

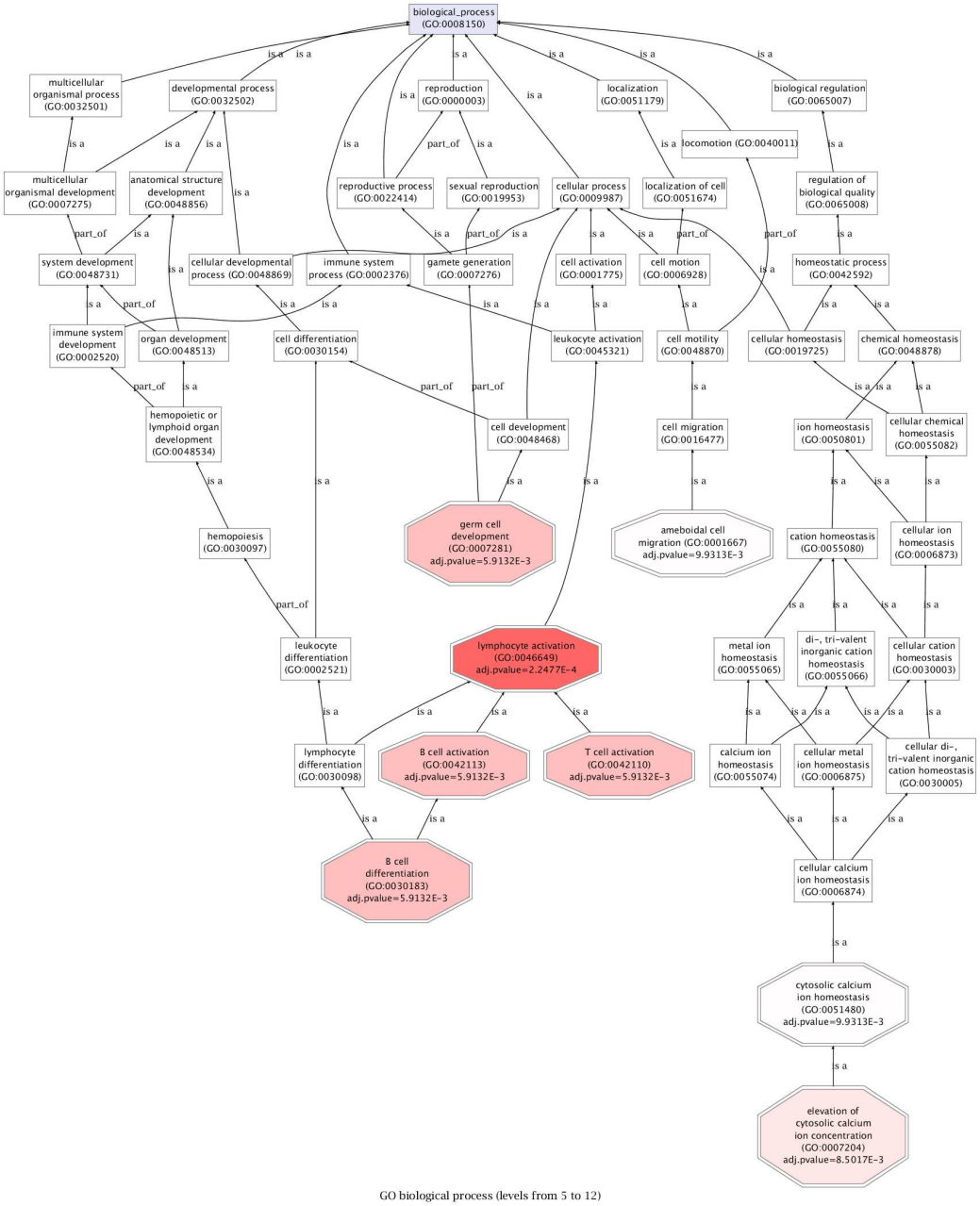


Figure 7: NCI60 data set. Gene ontology tree structure for the most significant terms of the genes in cluster 38.