Technical report from Automatic Control at Linköpings universitet

Just Relax and Come Clustering! A Convexification of k-Means Clustering

Fredrik Lindsten, Henrik Ohlsson, Lennart Ljung

Division of Automatic Control

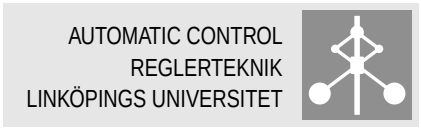
E-mail: lindsten@isy.liu.se, ohlsson@isy.liu.se, ljung@isy.liu.se

1st February 2011

Report no.: LiTH-ISY-R-2992

Address:
Department of Electrical Engineering
Linköpings universitet
SE-581 83 Linköping, Sweden

WWW: http://www.control.isy.liu.se



Abstract

k-means clustering is a popular approach to clustering. It is easy to implement and intuitive but has the disadvantage of being sensitive to initialization due to an underlying non-convex optimization problem. In this paper, we derive an equivalent formulation of k-means clustering. The formulation takes the form of a ℓ_0-regularized least squares problem. We then propose a novel convex, relaxed, formulation of k-means clustering. The sum-of-norms regularized least squares formulation inherits many desired properties of k-means but has the advantage of being independent of initialization.

Keywords: Clustering, k-means, sum-of-norms, group-lasso

Just Relax and Come Clustering!

A Convexification of k-Means Clustering

Fredrik Lindsten, Henrik Ohlsson and Lennart Ljung

***Abstract*—k-means clustering is a popular approach to clustering. It is easy to implement and intuitive but has the disadvantage of being sensitive to initialization due to an underlying non-convex optimization problem. In this paper, we derive an equivalent formulation of k-means clustering. The formulation takes the form of a ℓ_0-regularized least squares problem. We then propose a novel convex, relaxed, formulation of k-means clustering. The sum-of-norms regularized least squares formulation inherits many desired properties of k-means but has the advantage of being independent of initialization.**

***Index Terms*—Clustering, k-means, sum-of-norms, group-lasso**

I. Introduction and related work

Clustering is the problem of dividing a given set of data points into different groups, or clusters, based on some common properties of the points. Clustering is a fundamental cornerstone of machine learning, pattern recognition and statistics and an important tool in *e.g.,* image processing and biology. Clustering has a long history and, naturally, a huge variety of clustering techniques have been developed. We refer to Xu and Wunsch [2005] for en excellent survey of the field.

One of the most well known clustering techniques is *k-means clustering*. The k-means clustering method has been a frequently used tool for clustering since the 1950s. The idea was first proposed by Hugo Steinhaus in 1956 [Steinhaus, 1956] but the algorithm often used today was not published until 1982 [Lloyd, 1982].

One of the weaknesses of k-means clustering is that it is sensitive to initialization. As pointed out in Peña et al. [1999] and shown in Example 1, two different initializations can lead to considerably different clustering results. Heuristically "good" initialization methods for k-means have been proposed, see *e.g.,* Khan and Ahmad [2004], Arthur and Vassilvitskii [2007]. The sensitivity to initialization is due to the non-convex optimization problem underlying k-means clustering. A number of clustering algorithm with convex objectives, which therefore are independent of initialization, have been proposed, see *e.g.,* Lashkari and Golland [2008], Nowozin and Bakir [2008].

In this paper, we present a novel method for clustering, called sum-of-norms (SON) clustering. This method is briefly presented in Section II, where we also point out some of its important properties. In Section III, we then turn our attention to the well known k-means clustering problem. We discuss some of the weaknesses of k-means clustering, and also reformulate the problem in an equivalent form. From this formulation, we find a convex relaxation of the k-means problem, which in Section IV is shown to lead back to SON clustering. Here, we also discuss the properties of SON clustering in more detail. Readers not interested in the relationship between k-means clustering and SON clustering are encouraged to skip Sections III and IV-A. In Section V, we highlight some of the strengths and drawbacks with SON clustering, using numerical examples. Section VI provides some experimental results on real world data. Finally, in Section VII we briefly mention a few possible extensions to SON clustering and in Section VIII we draw conclusions.

A similar formulation to the method proposed in Section IV was previously discussed by Pelckmans et al. [2005]. It was there proposed to use $p = 1$ or $p = \infty$. In particular it was stated that "The use of the 1-norm and the ∞-norm results in a solution vector containing typically many zeros, which yields insight into the data in the form a small set of clusters." and "The use of a 2-norm is computationally more efficient but lacks the interpretability of the result as a form of clustering.". As discussed in Section IV, $p = 2$ will indeed result in clustering. The presentation in Pelckmans et al. [2005] is inspired by *lasso* [Tibsharani, 1996, Chen et al., 1998] while our formulation could be seen as inspired by *group-lasso* [Yuan and Lin, 2006]. No connection to k-means clustering is presented in Pelckmans et al. [2005].

Finally, some related contributions using sum-of-norms regularization are given by Yuan and Lin [2006], Kim et al. [2009], Ohlsson et al. [2010c,a,b].

II. Sum-of-norms clustering

This section will give a short preview of a novel clustering method, called sum-of-norms (SON) clustering. The method will be further discussed in Section IV, where it is derived as a convex relaxation of the well known k-means clustering problem. The reason for this preview is twofold,

1) To provide some insight into the proposed method, to make the discussion in the coming sections more accessible.
2) To show that SON clustering can be motivated as a clustering method on heuristic grounds, without the connection to k-means clustering.

The problem that we are interested in is to divide a set of observations $\{x_j\}_{j=1}^N$ in $\mathbb{R}^d$, into different clusters. Informally, we wish that points close to each other (in the Euclidian sense) are assigned to the same cluster, and vice versa. Also, the number of clusters should not be unnecessary large, but we do not know beforehand what the appropriate number is.

It is natural to think of the clusters as subsets of $\mathbb{R}^d$, such that if any point x_j belongs to this subset it also belongs to the corresponding cluster. Consequently, we can say that each cluster has a centroid in $\mathbb{R}^d$. Now, since we do not want to

specify how many clusters we are dealing with, we let μ_j be the centroid for the cluster containing x_j. Two x's are then said to belong to the same cluster if the corresponding μ's are the same. The sum-of-squares error, or the fit, is given by

$$\sum_{j=1}^{N} \|x_j - \mu_j\|^2. \tag{1}$$

Minimizing this expression with respect to the μ_j's would, due to the over-parameterization, not be of any use. The result would simply be to let $\mu_j = x_j$, $j = 1, \ldots, N$, *i.e.,* we would get N "clusters", each containing just one point. To circumvent this, we introduce a regularization term, penalizing the number of clusters. This leads to the *SON clustering* problem,

$$\min_{\mu} \sum_{j=1}^{N} \|x_j - \mu_j\|^2 + \lambda \sum_{j=2}^{N} \sum_{i<j} \|\mu_i - \mu_j\|_p, \tag{2}$$

for some $p \geq 1$ (see Section IV-B for a discussion on choices of p). The name refers to the sum-of-norms (SON) used as a regularization. The reason for using SON is that it is a well known sparsity regularization, see *e.g.,* Yuan and Lin [2006]. Hence, at the optimum, several of the terms $\|\mu_i - \mu_j\|_p$ will (typically) be exactly zero. Equivalently, several of the centroids $\{\mu_j\}_{j=1}^N$ will be identical, and associated x's can thus be seen as belonging to the same cluster, efficiently reducing the number of clusters. The regularization parameter λ is a user choice that will control the tradeoff between model fit and the number of clusters.

There are three key properties which make SON clustering appealing:

- The optimization problem is convex and the global optima can therefore be found independent of initialization. Many existing clustering methods (there among k-means clustering), are dependent on a good initialization for a good result.
- Convex constraints can easily be added.
- The method does not require the number of clusters to be specified beforehand. Instead, the tradeoff between the number of clusters and the model fit is controlled by a regularization parameter λ. This can be beneficial if we for instance seek to cluster sequential data, in which the "true" number of clusters is time-varying. For such problems, the method will adapt the number of clusters automatically, even if λ is left unchanged. See Lindsten et al. [2011] for an example in which SON clustering is applied to such data.

III. K-Means clustering

We will now turn our attention to the k-means clustering problem. The k-means problem is to minimize the within cluster sum-of-squares error, for a fixed number of clusters. As before, let $\{x_j\}_{j=1}^N$ be a given set of observations in $\mathbb{R}^d$. Let k be the (fixed) number of clusters and let $S_i \subset \{1, \ldots, N\}$, $i = 1, \ldots, k$ be distinct index sets. In particular, let $j \in S_i$ if x_j belongs to cluster i. Let $S = \{S_i\}_{i=1}^k$. The k-means clustering problem is given by,

$$\min_{S} \sum_{i=1}^{k} \sum_{j \in S_i} \|x_j - \theta_i\|^2 \tag{3a}$$

$$\text{s.t.} \quad \theta_i = \frac{1}{\operatorname{card} S_i} \sum_{j \in S_i} x_j, \tag{3b}$$

$$\bigcup_{i=1}^{k} S_i = \{1, \ldots, N\}. \tag{3c}$$

This problem has been shown to be NP hard [Aloise et al., 2009, Dasgupta and Freund, 2009]. An approximate solution is commonly sought by the heuristic method presented in Algorithm 1.

Algorithm 1 k-Means clustering (Lloyd's algorithm [Lloyd, 1982])

Require:
- Data points $\{x_j\}_{j=1}^N$.
- The number of clusters $k \leq N$.
- An initialisation of the centroids $\{\theta_i\}_{i=1}^k$.

Ensure:
- Index sets $\{S_i\}_{i=1}^k$.

1: **loop**
2: **Update index sets:** For fixed centroids $\{\theta_i\}_{i=1}^k$, compute the index sets $\{S_i\}_{i=1}^k$,
$$S_i \leftarrow \{j : \|x_j - \theta_i\| \leq \|x_j - \theta_l\|, l = 1, \ldots, k\}.$$
3: **Update centroids:** For fixed index sets $\{S_i\}_{i=1}^k$, estimate the centroids $\{\theta_i\}_{i=1}^k$,
$$\theta_i \leftarrow \frac{1}{\operatorname{card} S_i} \sum_{j \in S_i} x_j.$$
4: **if** No change in assignment since last iteration **or** maximum number of iterations reached **then**
5: **return**
6: **end if**
7: **end loop**

Of course, there is no guarantee that the algorithm will find any of the minimizers of (3). In fact, the algorithm is known to be sensitive to initialization [Peña et al., 1999], which is one of its major drawbacks. This issue is illustrated in Example 1. Another drawback with Algorithm 1 is that it for certain problems can be slow to converge to a stationary point. In Vattani [2009], it is shown that the algorithm in fact may require exponential time, even for two-dimensional (2D) problems.

Example 1: k-means' initialization problems For illustration purpose, let us consider a simple 2D clustering problem. Let x_j, $j = 1, \ldots, 25$ form the first cluster. These x's are shown as filled squares in the left plot of Figure 1. $\{x_j\}_{j=1}^{25}$ were generated by sampling uniformly from a circle with radius 0.07 centered at $[-0.05 \; -0.05]$ Let further x_j, $j = 26, \ldots, 50$ form the second cluster. These points are shown as circles in the left plot of Figure 1. $\{x_j\}_{j=26}^{50}$ were generated

by sampling uniformly from a circle with radius 0.07 centered at $[0.05\, 0.05]$.

Let us apply k-means clustering to $\{x_j\}_{j=1}^{50}$. Let $k = 2$, use Algorithm 1 and a maximum of 1000 iterations. The two asterisks $(*)$ shown in the left plot of Figure 1 show the initialization of the two means of the k-means clustering algorithm. For this particular initialization, k-means clustering identify the "true" clusters. However, if the algorithm is initialized differently, an erroneous result is obtain, as shown in the right plot of Figure 1.

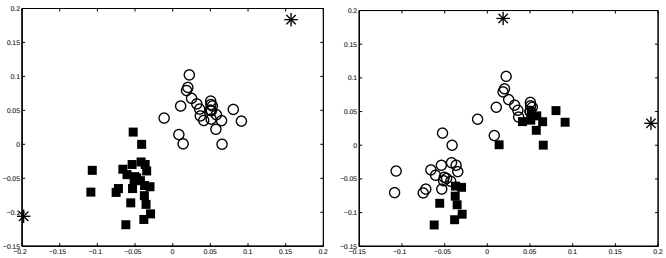


Fig. 1. Illustration of the sensitivity of the k-means clustering algorithm for initial condition.

The minimizing solution of (3) can always be computed by an exhaustive search. However, since the number of combinations grows as $\sum_{i=1}^{\lceil N/2 \rceil} \frac{N!}{(N-i)!}$ (for $k = 2$), approximately $2 \cdot 10^{39}$ different solution candidates have to be checked in this particular example in order to compute the minimizing S. This is clearly impractical.

A. An equivalent formulation

In the coming sections, SON clustering will be derived as a convex relaxation of the k-means problem. To arrive at this relaxation, we shall start by rewriting k-means clustering in an equivalent form.

We first note the following,

Proposition 1. *Assume that there are at least k distinct data points,* i.e., *the collection $\{x_j\}_{j=1}^N$ contains at least k unique vectors. Let S^* be a global optima of* (3)*. Then S^* is a partitioning of the set $\{1, \ldots, N\}$ into k non-empty, disjoint subsets. Furthermore, the θ_i computed according to* (3b) *are all unique.*

Proof: See Appendix A. ∎

In other words, any solution to the k-means clustering problem will assign each data point x_j, $j = 1, \ldots, N$, to only one cluster. Moreover, no cluster will be empty. Due to this, whenever we refer to (3), we shall use an equivalent form of the problem in which S is constrained according to Proposition 1. This is merely a technical detail.

Let us introduce variables $\mu_j \in \mathbb{R}^d$ for $j = 1, \ldots, N$. Let $\mu = \{\mu_j\}_{j=1}^N$ and consider the optimization problem,

$$\min_{\mu} \sum_{j=1}^{N} \|x_j - \mu_j\|^2 \tag{4a}$$

$$\text{s.t. } \{\mu_1, \ldots, \mu_N\} \text{ contains } k \text{ unique vectors.} \tag{4b}$$

Hence, instead of using just k centroids, we increase the number of optimization variables to allow for one vector μ_j corresponding to each data point x_j. A solution to (4) would still produce k clusters, due to the constraint (4b) (using the convention that x's belong to the same cluster if their corresponding centroids are the same).

To formalize the relationship between the problems (3) and (4), we will need mappings between the sets of feasible points for the two problems. Hence, let $\Gamma \subset \mathbb{R}^{d \times N}$ be the set of all centroids, feasible for (4). Similarly, let Ω be the set of all points, feasible for (3), *i.e.,* Ω is the set of all possible partitionings of $\{1, \ldots, N\}$ into k non-empty, disjoint subsets.

Definition 1. *Define the mapping $T : \Gamma \to \Omega$ as follows. Given $\mu = \{\mu_j\}_{j=1}^N \in \Gamma$, compute $\{\bar{S}_i\}_{i=1}^n$, according to,*

$\bar{S}_1 \leftarrow \{1\}$
$n \leftarrow 1$
for $j = 2$ to N **do**
  if $\mu_j = \mu_l$ for some $l = 1, \ldots, j-1$ **then**
    $\bar{S}_i \leftarrow \bar{S}_i \cup \{j\}$ where i is the index for which $l \in \bar{S}_i$.
  else
    $\bar{S}_{n+1} \leftarrow \{j\}$
    $n \leftarrow n + 1$
  end if
end for

Then, after termination, $n = k$, $\bar{S} = \{\bar{S}_i\}_{i=1}^k \in \Omega$ and we take $\bar{S} = T(\mu)$.

Definition 2. *Define the mapping $R : \Omega \to \Gamma$ as follows. For $S = \{S_i\}_{i=1}^k \in \Gamma$, we take $\bar{\mu} = R(S)$ such that, for $\bar{\mu} = \{\bar{\mu}_j\}_{j=1}^N \in \Gamma$, we have $\bar{\mu}_j = \theta_i$ if $j \in S_i$, $j = 1, \ldots, N$, and θ_i is computed according to* (3b)*.*

Hence, the mapping T will take any point μ, feasible for (4), and return a corresponding class of index sets $\bar{S}$. Reversely, the mapping R will take any point S, feasible for (3), and return the corresponding set of mean vectors $\bar{\mu}$, one for each data point x_j (but only k of the vectors are unique). We are now ready to give the following proposition.

Proposition 2. *Given a set of observations $\{x_j\}_{j=1}^N$ and for a given k, the problems* (3) *and* (4) *are equivalent in the sense that,*

1) *if S^* is an optimal point of* (3)*, then $\mu = R(S^*)$ is an optimal point for* (4)*.*
2) *if μ^* is an optimal point of* (4)*, then $S = T(\mu^*)$ is an optimal point for* (3)*.*

Proof: See Appendix B. ∎

Let us now formulate the constraint (4b) using mathematical language. Given the set $\{\mu_1, \ldots, \mu_N\}$ we wish to count the number of unique vectors in the set. Define an $N \times N$ matrix according to

$$\Delta_{ij} = \kappa(\mu_i, \mu_j), \tag{5a}$$

where $\kappa : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is any nonnegative, symmetric function with the property

$$\kappa(\mu_i, \mu_j) = 0 \Leftrightarrow \mu_i = \mu_j \tag{5b}$$

(κ could for instance be any norm of the difference of its arguments). Clearly, Δ is symmetric with zeros on the diagonal and if $\mu_i = \mu_j$, then the element Δ_{ij} will be zero. Consider the upper triangle of Δ (excluding the diagonal),

$$\Delta = \begin{bmatrix} 0 & \times & \cdots & \times \\ & \ddots & \ddots & \vdots \\ & & \ddots & \times \\ & & & 0 \end{bmatrix}. \tag{6}$$

The number of vectors in the set $\{\mu_j\}_{j=2}^N$ that are equal to μ_1 is then the number of zeros in the first row of the upper triangle. Similarly, for $n < N$, the number of vectors in the set $\{\mu_j\}_{j=n+1}^N$ that are equal to μ_n is the number of zeros in the n:th row of the triangle. Also, Δ must have the property that, for $i < j < l$, if $\Delta_{ij} = 0$ then

$$\Delta_{il} = 0 \Leftrightarrow \Delta_{jl} = 0, \tag{7}$$

i.e., if $\mu_i = \mu_j$ then $\mu_i = \mu_l \Leftrightarrow \mu_j = \mu_l$.

To count the number of duplicates among the vectors $\{\mu_j\}_{j=1}^N$, we could thus proceed as follows,

- Count the number of zeros in the first row of the upper triangle.
- Count the number of zeros in the second row of the upper triangle, unless there is a zero in the same column in the first row.
- Proceed with all $N-1$ rows of the upper triangle, count any zero unless there there is a zero in the same column in any row above.

It can be realized that this, rather inconvenient procedure, is equivalent to,

- Count the number of columns in the upper triangle, containing at least one zero.

By using the indicator function

$$I(x) = \begin{cases} 1 & \text{if } x \neq 0, \\ 0 & \text{if } x = 0, \end{cases} \tag{8}$$

the number of a zeros in column j of the upper triangle of Δ is

$$\sum_{i<j} \left(1 - I(\Delta_{ij})\right). \tag{9a}$$

By taking the indicator function of this expression, the presence of any zero in column j can be determined. Finally, the unique number of vectors in the set $\{\mu_j\}_{j=1}^N$ can thus be written

$$N - \sum_{j=2}^{N} I\left(\sum_{i<j}\Big(1 - I\big(\kappa(\mu_i,\mu_j)\big)\Big)\right). \tag{9b}$$

Using the ℓ_0-norm, which simply is the number of non-zero elements of a vector, we can instead write the above expression as

$$N - \|\delta\|_0, \tag{9c}$$

where we have defined the $(N-1)$-vector $\delta = \begin{bmatrix}\delta_2 & \dots & \delta_N\end{bmatrix}^T$ as

$$\begin{aligned}\delta_j &= \sum_{i<j}\Big(1 - I\big(\kappa(\mu_i,\mu_j)\big)\Big) \\ &= j - 1 - \sum_{i<j} I\big(\kappa(\mu_i,\mu_j)\big) = j - 1 - \|\gamma^j\|_0.\end{aligned}$$

Here, the vectors $\gamma^j = \begin{bmatrix}\gamma_1^j & \dots & \gamma_{j-1}^j\end{bmatrix}^T$ for $j = 2, \dots, N$, are given by

$$\gamma_i^j = \kappa(\mu_i, \mu_j). \tag{9d}$$

We can now reformulate problem (4) according to

$$\min_{\mu} \sum_{j=1}^{N} \|x_j - \mu_j\|^2 \tag{10a}$$

$$\text{s.t.} \quad k = N - \|\delta\|_0. \tag{10b}$$

IV. Proposed Clustering Method

A. Relaxation of k-means clustering

In some sense, the convex function closest to the ℓ_0-norm is the ℓ_1-norm. The ℓ_1-norm has been used to approximate the ℓ_0-norm in a number of very successful methods *e.g., Compressed Sensing* (CS, Candès et al. [2006], Donoho [2006]) and *lasso* (least absolute shrinkage and selection operator, Tibsharani [1996], Chen et al. [1998], see also Hastie et al. [2001, p. 64]). In the spirit of these methods, we will in this section propose to convexify (10) by replacing the ℓ_0-norm with the ℓ_1-norm. A relaxed version of (10) is

$$\min_{\mu} \sum_{j=1}^{N} \|x_j - \mu_j\|^2 \tag{11a}$$

$$\text{s.t.} \quad k = N - \sum_{j=2} \Big| j - 1 - \|\gamma^j\|_0 \Big|. \tag{11b}$$

Since $j - 1 - \|\gamma^j\|_0 \geq 0$, $j = 2, \dots, N$, the absolute value can be removed,

$$\min_{\mu} \sum_{j=1}^{N} \|x_j - \mu_j\|^2 \tag{12a}$$

$$\text{s.t.} \quad \sum_{j=2}^{N} \|\gamma^j\|_0 = \frac{3N - N^2}{2} - k. \tag{12b}$$

This is still non-convex and to obtain a convex criterion, we again relax the ℓ_0-norm using an ℓ_1-norm, yielding

$$\min_{\mu} \sum_{j=1}^{N} \|x_j - \mu_j\|^2 \tag{13a}$$

$$\text{s.t.} \quad \sum_{j=2}^{N} \sum_{i<j} \kappa(\mu_i, \mu_j) = \frac{3N - N^2}{2} - k. \tag{13b}$$

If κ is chosen appropriately (13) is a convex problem (recall that, in the original formulation, κ is arbitrary under the constraint (5b)). In this case, by using a Lagrange multiplier

(see *e.g.,* Boyd and Vandenberghe [2004, p. 215]), it can be shown that there exists a $\lambda > 0$ such that (13) is equivalent to

$$\min_{\mu} \sum_{j=1}^{N} \|x_j - \mu_j\|^2 + \lambda \sum_{j=2}^{N} \sum_{i<j} \kappa(\mu_i, \mu_j). \qquad (14)$$

B. Sum-of-norms clustering revisited

Now, let $\kappa(x, y) = \|x - y\|_p$. This particular choice which will yield a convex problem, given by

$$\min_{\mu} \sum_{j=1}^{N} \|x_j - \mu_j\|^2 + \lambda \sum_{j=2}^{N} \sum_{i<j} \|\mu_i - \mu_j\|_p. \qquad (15)$$

We have thus obtained the SON clustering problem (2) by applying an $\ell_0 \to \ell_1$ relaxation of the k-means problem. As previously pointed out, (15) is equivalent to (13) for *some* λ. We could of course choose to address the problem (13) directly, which would also be convex for the current choice of κ, but there is a point in not doing so. The additive term $\sum_{j=2}^{N} \sum_{i<j} \|\mu_i - \mu_j\|_p$, appearing in (15), can be seen as a regularization. Hence, $\lambda > 0$ can be viewed as a parameter that will control the trade-off between model fit (the first term) and the number of clusters (the second term). Note that the number of clusters k, is not present in the criterion (15).

Hence, SON clustering will adapt the number of clusters depending on the data. The user chosen parameter is moved from the number of clusters k, to the regularization parameter λ. This property of SON clustering has several benefits. For instance, if we wish to perform sequential clustering of data that changes over time, SON clustering has the ability to adapt the number of clusters to best describe the data. This can be done without any interference from a user, *i.e.,* for the same value of λ. This would of course not be possible if we instead specify the number of clusters k directly. We study an example of such sequential data in Lindsten et al. [2011].

Since the number of parameters in (15) equals the number of observations, the regularization is necessary to prevent overfitting to the noisy observations. Using the regularization, we prevent overfitting by penalizing the number of distinct μ-values, used in (15).

Remark 1 (Sum-of-norms regularization)**.** *The SON regularization used in* (15) *can be seen as an ℓ_1-regularization of the p-norm of differences $\mu_i - \mu_j$, $j = 1, \ldots, N$, $i < j$. The SON term is the ℓ_1-norm of the vector obtained by stacking $\|\mu_i - \mu_j\|_p$, for $j = 1, \ldots, N$, $i < j$. Hence, this stacked vector, and not the individual μ-vectors, will become sparse.*

We will in general use $p = 2$, but other choices are of course possible. However, to get the properties discussed above, p should be chosen greater than one. With $p = 1$, we obtain a regularization variable having many of its components equal to zero, we obtain a sparse vector. When we use $p > 1$, the whole estimated regularization variable vector often becomes zero; but when it is nonzero, typically all its components are nonzero. Here, $p > 1$ is clearly to be preferred, since we desire the whole parameter vectors μ to be the same if they are not needed to be different. In a statistical linear regression framework, sum-of-norms regularization ($p > 1$) is called *group-lasso* [Yuan and Lin, 2006], since it results in estimates in which many groups of variables are zero.

A last step is also useful. Having found the minimizing μ, say μ^*, of (15) we carry out a constrained least squares, where μ_i is set equal to μ_j if $\mu_i^* = \mu_j^*$. This is done to avoid a biased-solution. In the following, we assume that the procedure of solving (15) is always followed by such a constrained least squares problem, whenever referring to SON clustering. However, note that this last step is relevant only if the actual centroid-values are of interest. To merely compute which x's that belong to which cluster, the last step can be skipped since we only need to know whether or not μ_i^* equals μ_j^*, not the actual values of the individual elements of μ^*.

C. Solution algorithms and software

Many standard methods of convex optimization can be used to solve the problem (15). Systems such as CVX [Grant and Boyd, 2010, 2008] or YALMIP [Löfberg, 2004] can readily handle the sum-of-norms regularization, by converting the problem to a cone problem and calling a standard interior-point method. For the special case $p = 1$, more efficient, special purpose algorithms and software can be used, such as `l1_ls` [Kim et al., 2007]. Recently, many authors have developed fast, first order methods for solving ℓ_1 regularized problems, and these methods can be extended to handle the sum-of-norms regularization used here; see, for example, Roll [2008, §2.2].

Below, we show how to solve the problem in MATLAB using CVX. Let a d-dimensional set of input data vectors $\{x_j\}_{j=1}^N$ be given. In the MATLAB workspace, define `lambda`, let `x` be a $d \times N$-dimensional matrix containing the N x's of $\{x_j\}_{j=1}^N$ as columns and set `p`, `d` and `N`. Set also `eps` to some small value. SON clustering can then be carried out using the CVX-code given in Listings 1.

Listing 1. CVX code for SON clustering

```
i=1;
for t=1:N
    for k=t+1:N
     Q(i,t)=1;Q(i,k)=-1;i=i+1;
    end
end
%%%%%%%%%%%%%% solve (15) %%%%%%%%%%%%%%
cvx_begin
variable mu1(d,N)
minimize(sum(sum((x-mu1).*(x-mu1))) ...
         +lambda*sum(norms(Q*mu1',2,p)))
cvx_end
%%%%% constrained least squares %%%%%%
cvx_begin
variable mu2(d,N)
minimize(sum(sum((x-mu2).*(x-mu2))))
subject to
Q(find(norms(Q*mu1',2,p)<eps),:)*mu2'==0
cvx_end
```

A code-package is also available for download at http://www.control.isy.liu.se/~ohlsson/code.html.

V. Discussion and numerical illustraion

In this section we discuss the properties of SON clustering and exemplify its behavior through a number of examples. We do not have the intention to show, and we do not believe, that SON clustering is the clustering method that should be applied in all situations. On the contrary, it is clear that every approximate solution to the (NP-hard) k-means problem will suffer from some undesired properties or peculiarities, and this will of course hold for SON clustering as well. As a user, it is important to be aware of and understand these properties. The goal of this section is therefore to point at both weaknesses and advantages and to give advise concerning when SON clustering can be suitable.

We start by comparing SON clustering with two related methods, Lloyd's algorithm and hierarchical clustering, to see how SON clustering circumvent some of the problems with these methods. We then turn to the weaknesses of SON clustering to point out the pitfalls that one should be aware of.

A. SON clustering and Lloyd's algorithm

In Example 1, it was shown that Lloyd's algorithm was sensitive to initialization. Let us return to this example to see how SON clustering performs.

Example 2: To apply SON clustering we need to choose a value for the regularization parameter λ. Once this value is fixed, the clustering result is independent of initialization, due to the convexity of SON clustering. However, the result will of course depend on λ. As previously pointed out, the regularization parameter will control the trade-off between model fit and the number of clusters. In Figure 2 the number of clusters is plotted as a function of λ. Any λ between $4.5\cdot 10^{-3}$ and $5.1 \cdot 10^{-3}$ gives two clusters and the "true" S. It is clear that one of the main difficulties in applying SON clustering is to find a "good" value for λ. We comment further on this in Section V-B.

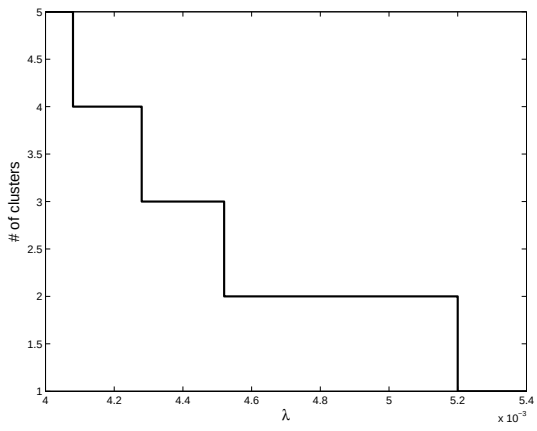


Fig. 2. Illustration of the effect of λ on the number of clusters for the data in Example 1.

B. SON clustering and hierarchical clustering

In Example 2 we saw how the number of clusters produced by SON clustering depended on the regularization parameter λ. Choosing λ is related to deciding on model complexity and the problem of order selection. Hence, methods like cross validation (see *e.g.,* Ljung [1999]) can aid in finding a proper value. Another way, guided by Example 2, would be to solve the problem for *all* values of λ and draw a plot similar to Figure 2. From this plot, we see that the number of clusters remains at two for a wide range of λ-values, indicating that there is evidence for two clusters in the data.

However, assuming that we can solve the problem for all values of λ, a more sensible choice would be to provide the entire "solution set" to the user. In other words, instead of making a hard decision on λ (implying, for a given problem, a hard decision on the number of clusters) all solutions are given to the user as a function of λ. Whether or not a hard choice on the number of clusters should be made is then left as an *a posteriori* decision to the user. In some sense, this is the most that we can ask of any method for clustering (or more generally, model identification). Unavoidable, there needs to be a trade-off between model fit and model complexity, and it is up to the user to decide on this trade-off. The most we can expect from any clustering/identification method is thus to provide as much and as clear information as possible to guide in this decision.

To solve the SON clustering problem for all values of λ may seem like a hopeless task, but this is not necessarily the case. Ideas similar to Roll [2008] could be used to find a solution path, *i.e.,* to find a functional expression for the solution to the SON clustering problem as a function of λ. In the general case, this would be a nonlinear function. However, for $p = 1$ in the regularization term, the solution path is piecewise affine and can be efficiently computed [Roll, 2008].

The idea of providing a full set of solutions, instead of a single one, appears also in hierarchical clustering (see *e.g.,* Hastie et al. [2001], Chapter 14.3.12). The main difference between SON clustering (with solution paths in λ) and hierarchical clustering, is that the latter is inherently greedy. Take for instance an agglomerative (bottom up) method. Starting with a cluster for each data point, it successively merges the two clusters that minimizes the objective function the most, and then continues in that fashion. Hence, a poor choice at an early stage can not be repaired and can lead to a poor result.

This problem is illustrated in the example below, in which SON clustering is compared with hierarchical clustering.

Example 3: The MATLAB function `linkage` of the Statistics Toolbox is an implementation of hierarchical clustering. We apply this function to synthetic data consisting of $N = 107$ point, shown in the upper left plot of Figure 4. We use the default settings, *i.e.,* a single linkage agglomerative method. A tree diagram, a *dendrogram*, showing the merging of clusters is given in in Figure 3. The algorithm starts with one clusters for each data point, illustrated by the $N = 107$ bars at the bottom of figure. As we "move" upward in the figure, existing clusters are merged successively until we reach a minimum of two clusters at the top. Hence, the dendrogram provides a

full "solution set" as discussed above. The user can choose to "cut" the tree at any desired depth, yielding a solution with the corresponding number of clusters. As previously mentioned,

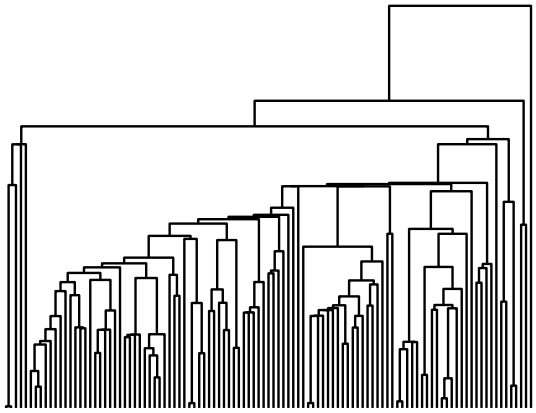


Fig. 3. Dendrogram from hierarchical clustering method (see text for details).

a problem with hierarchical clustering is that it is greedy. If an error is made early in the algorithm (*i.e.,* at the bottom of the dendrogram), this error cannot be repaired as we move upward. This problem is illustrated in Figure 4, where the result from the function `linkage` is given. The lower left plot shows the result when the dendrogram is cut at a level yielding two clusters. One of the clusters contains a single data point, and the remaining points belong to the second cluster, which is clearly not optimal *w.r.t.* the within cluster sum-of-squares error. To see why we obtain this result, we choose to stop the merging of clusters at a much earlier stage, namely when we have as many as 32 clusters. The result from this level is shown in the lower right plot of Figure 4. To avoid cluttering of the figure, we choose to show only the largest cluster. The data points in this cluster are shown as circles, whereas all point in the remaining 31 clusters are shown as dots. Since the merging is based only on local properties, closeness of the point in the "middle" create a cluster which stretches into both groups of points. The errors made at this early stage can then not be repaired as we continue toward two clusters.

Since it is not a greedy approach, SON clustering works a bit differently. By solving the clustering problem for all λ-values, a plot can be created in the same fashion as Figure 2. This plot hold similar information as the dendrogram in Figure 3. For any given value of λ, we can extract a solution contained a certain number of clusters. However, it is not possible to draw a dendrogram over the SON clustering results. The reason is that when moving from one level to the next, no merging of existing clusters is made. Instead we may get an entirely new configuration, which means that an error made at an early stage (small λ) indeed can be repaired as we increase λ. In the top right plot of Figure 4, the result of SON clustering for $\lambda = 0.036$ is shown.

C. Weaknesses of SON clustering

SON clustering can be viewed as approximately solving the k-means problem. As mentioned above, we should expect any such approximative method to suffer from some undesired

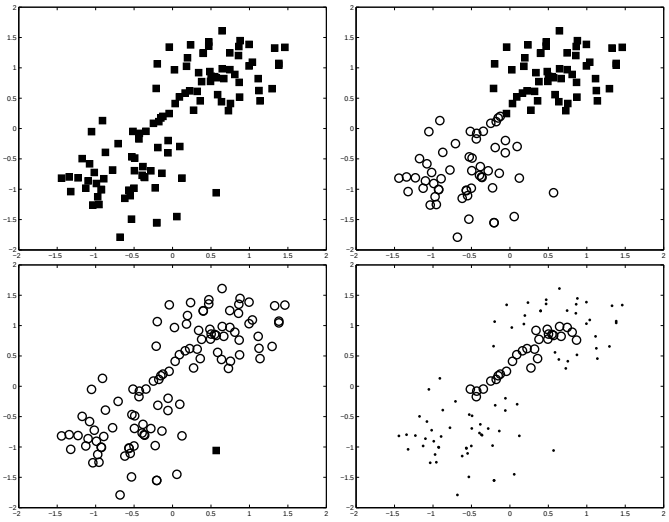


Fig. 4. (Top left) Data set with two "visible" clusters. (Top right) SON clustering with $\lambda = 0.036$. (Bottom left) Hierarchical clustering result for two clusters. (Bottom right)The largest cluster (shown as circles) from the hierarchical clustering result for 32 clusters.

properties. We will in this section highlight a peculiarity with SON clustering, namely that it prefers non-equally sized clusters. This is illustrated in the following example.

Example 4: Assume that λ is chosen such that the solution to the SON clustering problem contains $k = 2$ clusters. For simplicity, let the data points be ordered so that $\{x_j\}_{j=1}^{n}$ belongs to cluster 1 and $\{x_j\}_{j=n+1}^{N}$ belongs to cluster 2. Furthermore, let r be the distance between the two cluster centroids (under the chosen norm). Now, since the regularization term in (15) is said to control the number of clusters, its value should remain constant as long as $k = 2$. In other words, we would like the value of the regularization term to be independent of how many data points that are assigned to each cluster, *i.e.,* to be independent of n. However, it is easily checked that the regularization term in this case is given by

$$\lambda \sum_{j=2}^{N} \sum_{i<j} \|\mu_i - \mu_j\|_p = \lambda r(N-n)n, \tag{16}$$

which indeed depends on n. Hence, the regularization penalty will reach its maximum if $n = \lfloor N/2 \rfloor$ or $n = \lceil N/2 \rceil$, and its minimum when $n = 1$ or $n = N - 1$, see Figure 5. This means that the regularization term of SON clustering will prefer one small and one large cluster, over two equally sized ones. However, it is important to remember that we in this analysis only considered the influence of the regularization term. When also the fit term is taken into account, the optimal solution of the SON clustering problem might very well be to assign equally many points to each cluster, if there is evidence for this in the data (see *e.g.,* Example 3).

To analyze the reason for this behavior, consider the matrix Δ as defined in (5), for this particular example with $k = 2$,

$$\Delta = \begin{bmatrix} 0_{n\times n} & r1_{n\times(N-n)} \\ r1_{(N-n)\times n} & 0_{(N-n)\times(N-n)} \end{bmatrix}. \tag{17}$$

Here, $0_{i\times j}$ and $1_{i\times j}$ refers to matrices filled with zeros

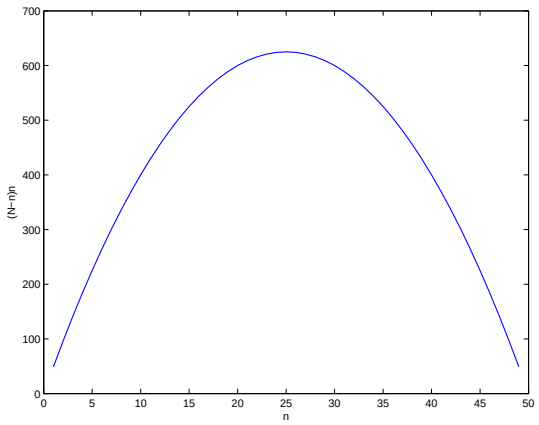


Fig. 5. Value of SON regularization term for different cluster sizes, *i.e.,* $(N-n)n$ plotted against n, for $N = 50$ and $n = 2, \ldots, N-1$.

and ones, respectively. To compute the number of clusters we should, according to Section III-A, count the number of columns in the upper triangle of this matrix, containing at least one zero. For any choice of $n \in \{1, \ldots, N-1\}$, this will, as exprected, result in $k = 2$ clusters. However, the approximation underlying SON clustering means that we instead will sum over all elements of the upper triangle. Hence, the sum will be proportional to the "volume" of the upper right block matrix, which has $n(N-n)$ elements.

One way to remove the influence of this issue, is to apply a kernel based weighting to the SON regularization, see Section VII.

VI. Experimental results

In this example we demonstrate how SON-clustering can be used for vector quantization (VQ, see *e.g.,* Gersho and Gray [1991]). Figure 6 shows part of a scan of a handwritten document. The letter 'P' and part of an 'e' are visible. The

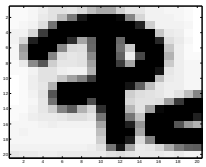


Fig. 6. A part of a scanned paper showing a handwritten 'P' and part of an 'e'.

VQ implementation used here breaks down the image into blocks of 4 pixels. The gray-scale values within a block is stacked and represented as a 4-dimensional vector. The image showed in Figure 6 can hence be represented as 100 4-dimensional vectors. The idea behind VQ is now to find an approximation by replacing the original vectors by a low number of basis vectors, or as it is called in VQ, *codewords*. To find these codewords we here chose to use SON clustering and replace vectors of a cluster by their centroid. Figure 7 shows approximations of the image showed in Figure 6 using 49, 31, 16 and 8 codewords ($\lambda = 0.00001,\ 0.02,\ 0.025,\ 0.057$ in SON clustering). That corresponds to 0.18, 0.15, 0.13 and 0.09 of the required memory needed for the original image.

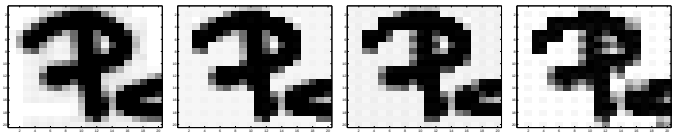


Fig. 7. From left to right, 49, 31, 16 and 8 clusters.

VII. Extensions

The problem (2) can be seen as the basic formulation of SON clustering. Here, we briefly mention two possible extensions to the method. First, since it is based on the Euclidian distance between data points, (2) can only handle linearly separable clusters. To address nonlinear clustering problems, the "kernel trick" can be used.

Second, it may be beneficial to add weights to the regularization term in (2). Since the sum in the regularization term ranges over all pairs of point, it will penalize distinct μ-values even if the corresponding data points are far apart. To circumvent this, we can localize the regularization penalty by adding data dependent weights. A modified optimization problem is then,

$$\min_{\mu_1 \cdots \mu_N} \sum_{j=1}^{N} \|x_j - \mu_j\|^2 + \lambda \sum_{j=2}^{N} \sum_{i<j} \kappa(x_i, x_j) \|\mu_i - \mu_j\|_p, \tag{18}$$

where κ is a local kernel. Note that, since κ depends only on the (fixed) data points $\{x_i\}_{i=1}^N$ and not on the optimization variables $\{\mu_i\}_{i=1}^N$, it does not change the convexity or the dimension of the problem.

Any local kernel (*e.g.,* Gaussian) can of course be used. However, from a computational point of view, it can be beneficial to use a kernel with bounded support, since this can significantly reduce the number of nonzero terms in the regularization sum. This can for instance be a simple kNN-kernel (k nearest neighbours), *i.e.,*

$$\kappa(x_i, x_j) = \begin{cases} 1 & \text{if } x_i \in k\text{NN}(x_j) \text{ or } x_j \in k\text{NN}(x_i), \\ 0 & \text{otherwise,} \end{cases} \tag{19}$$

where kNN(x) is the set of x's k nearest neighbours.

VIII. Conclusion

We have proposed a novel clustering method, formulated as a convex optimization problem. We show that the problem can be seen as a relaxation of the popular k-means clustering problem, which is known to be NP-hard. Due to the convexity of the proposed problem, it can be efficiently solved and it is also straightforward to add constraints. The problem is over-parameterized, and overfitting is prevented using a sum-of-norms regularization. The method does not require the number of clusters to be specified beforehand. Instead, a regularization parameter is used to control the trade-off between model fit and the number of clusters. This feature gives the method the ability to dynamically adapt the number of clusters, *e.g.,* if it is applied to sequential data with a varying number of clusters.

APPENDIX

A. Proof of Proposition 1

Assume that S_i^* is empty for some i. Then at least one of the remaining index sets S_l^*, $l \in \{1, \ldots, k\} \setminus i$, contains two elements m, n, for which $x_m \neq x_n$. Assume that $x_n \neq \theta_l$ (if not, let n and m change place). Consider the contribution to the cost from cluster l,

$$\begin{aligned}\sum_{j \in S_l^*} \|x_j - \theta_l\|^2 &= \sum_{j \in S_l^* \setminus n} \|x_j - \theta_l\|^2 + \|x_n - \theta_l\|^2 \\ &> \sum_{j \in S_l^* \setminus n} \|x_j - \theta_l\|^2 \\ &\geq \sum_{j \in S_l^* \setminus n} \|x_j - \frac{1}{\operatorname{card} S_l^* \setminus n} \sum_{j \in S_l^* \setminus n} x_j\|^2.\end{aligned} \tag{20}$$

Hence, the cost is strictly lower if we remove index n from the set S_l^*. By letting $S_i^* = \{n\}$, the value of the cost function is not increased ($\theta_i = x_n$), and we get a new feasible point. Hence, no optimal solution can have empty index sets.

For the uniqueness of θ, assume that $\theta_i = \theta_l$ for some i, l at S^*. Then,

$$\begin{aligned}\operatorname{card} S_i^* \theta_i + \operatorname{card} S_l^* \theta_l &= \sum_{j \in S_i^*} x_j + \sum_{j \in S_l^*} x_j \\ \Rightarrow \theta_i &= \frac{1}{\operatorname{card} S_i^* + \operatorname{card} S_l^*} \sum_{j \in S_i^* \cup S_l^*} x_j.\end{aligned} \tag{21}$$

Hence, we would not change θ_i if we were to remove all points in S_l^* and add them to S_i^*, and would thus not affect the cost. But in that case, S_l^* would be empty and by the above reasoning, we could strictly lower the cost by making some new assignment. Hence, no optimal solution can yield any two $\theta_i = \theta_l$, for some i, l.

Finally, for the disjointedness of the elements of S^*. Assume that, for some i, l, $S_i^* \cap S_l^* \neq \emptyset$ and take $j \in S_i^* \cap S_l^*$. Assume that $x_j \neq \theta_i$ (if not, let i and l switch places). Then, the cost will be strictly lower if index j is removed from S_i^* and this will still yield a feasible solution, since $\bigcup_{i=1}^k S_i* = \{1, \ldots, N\}$ still holds. Hence, the $\{S_i^*\}_{i=1}^k$ are all disjoint. ■

B. Proof of Proposition 2

Let f and g be the objective functions in problems (3) and (4), respectively, *i.e.,*

$$f(S) = \sum_{i=1}^{k} \sum_{j \in S_i} \|x_j - \theta_i\|^2, \tag{22a}$$

$$g(\mu) = \sum_{j=1}^{N} \|x_j - \mu_j\|^2. \tag{22b}$$

Let S^* be an optimal point for (3) and let $\{\theta_i^*\}_{i=1}^k$ be the corresponding centroids as in (3b). Then $\bar{\mu} = R(S^*)$ is feasible for (4). It follows that, for μ^* any optimal point for (4),

$$g(\mu^*) \leq g(\bar{\mu}). \tag{23}$$

Since S^* is a partitioning of $\{1, \ldots, N\}$ we may write,

$$g(\bar{\mu}) = \sum_{j=1}^{N} \|x_j - \bar{\mu}_j\|^2 = \sum_{i=1}^{k} \sum_{j \in S_i^*} \|x_j - \bar{\mu}_j\|^2. \tag{24}$$

From Definition 2 it follows that $\bar{\mu}_j = \theta_i^*$ for all $j \in S_i^*$, and consequently

$$g(\bar{\mu}) = f(S^*). \tag{25}$$

Similarly, $\bar{S} = T(\mu^*)$ is feasible for (3) and it follows that

$$f(S^*) \leq f(\bar{S}). \tag{26}$$

Define $\bar{\theta}_i$, $i = 1, \ldots, k$, such that $\bar{\theta}_i = \mu_j^*$ for some $j \in \bar{S}_i$, (according to Definition 1, we can choose any $j \in \bar{S}_i$). Since μ^* is optimal for (4), it must be that

$$\bar{\theta}_i = \arg\min_{\theta} \sum_{j \in \bar{S}_i} \|x_j - \theta\|^2 = \frac{1}{\operatorname{card} \bar{S}_i} \sum_{j \in \bar{S}_i} x_j, \tag{27}$$

(if this would not be the case, it would be possible to lower the cost in (4) by updating all μ_j^* for $j \in \bar{S}_i$). Hence,

$$f(\bar{S}) = \sum_{i=1}^{k} \sum_{j \in \bar{S}_i} \|x_j - \bar{\theta}_i\|^2 = \sum_{j=1}^{N} \|x_j - \mu_j^*\|^2 = g(\mu^*). \tag{28}$$

Combining (23), (25), (26) and (28) we get

$$g(\mu^*) \leq g(\bar{\mu}) = f(S^*) \leq f(\bar{S}) = g(\mu^*). \tag{29}$$

It follows that the inequalities are attained, meaning that $g(\bar{\mu}) = g(\mu^*)$ and $f(\bar{S}) = f(S^*)$, which completes the proof. ■

REFERENCES

D. Aloise, A. Deshpande, P. Hansen, and P. Popat. NP-hardness of Euclidean sum-of-squares clustering. *Mach. Learn.*, 75:245–248, May 2009.

D. Arthur and S. Vassilvitskii. k-means++: the advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, SODA '07, pages 1027–1035, Philadelphia, PA, USA, 2007. Society for Industrial and Applied Mathematics.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

E. J. Candès, J. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52:489–509, February 2006.

S. S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20(1):33–61, 1998.

S. Dasgupta and Y. Freund. Random projection trees for vector quantization. *Information Theory, IEEE Transactions on*, 55 (7):3229–3242, July 2009.

D. L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, April 2006.

A. Gersho and R. M. Gray. *Vector quantization and signal compression*. Kluwer Academic Publishers, Norwell, MA, USA, 1991.

M. Grant and S. Boyd. Graph implementations for nonsmooth convex programs. In V. D. Blondel, S. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, pages 95–110. Springer-Verlag Limited, 2008. http://stanford.edu/~boyd/graph_dcp.html.

M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming, version 1.21. http://cvxr.com/cvx, August 2010.

T. Hastie, R Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

S. Khan and A. Ahmad. Cluster center initialization algorithm for $k-means$ clustering. *Pattern Recognition Letters*, 25 (11):1293–1302, 2004.

S.-J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky. An interior-point method for large-scale l1-regularized least squares. *IEEE Journal of Selected Topics in Signal Processing*, 1(4):606–617, December 2007.

S.-J. Kim, K. Koh, S. Boyd, and D. Gorinevsky. ℓ_1 trend filtering. *SIAM Review*, 51(2):339–360, 2009.

D. Lashkari and P. Golland. Convex clustering with exemplar-based models. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 825–832. MIT Press, Cambridge, MA, 2008.

F. Lindsten, H. Ohlsson, and L. Ljung. Clustering using sum-of-norms regularization; with application to particle filter output computation. In *Submitted to the 2011 IEEE Workshop on Statistical Signal Processing (SSP)*, 2011.

L. Ljung. *System identification, Theory for the user*. System sciences series. Prentice Hall, Upper Saddle River, NJ, USA, second edition, 1999.

S. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, March 1982.

J. Löfberg. Yalmip : A toolbox for modeling and optimization in MATLAB. In *Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004. URL http://control.ee.ethz.ch/~joloef/yalmip.php.

S. Nowozin and G. Bakir. A decoupled approach to exemplar-based unsupervised learning. In *Proceedings of the 25th international conference on Machine learning*, ICML '08, pages 704–711, New York, NY, USA, 2008. ACM.

H. Ohlsson, F. Gustafsson, L. Ljung, and S. Boyd. State smoothing by sum-of-norms regularization. In *Proceedings of the 49th IEEE Conference on Decision and Control*, Atlanta, USA, December 2010a. To appear.

H. Ohlsson, F. Gustafsson, L. Ljung, and S. Boyd. Trajectory generation using sum-of-norms regularization. In *Proceedings of the 49th IEEE Conference on Decision and Control*, Atlanta, USA, December 2010b. To appear.

H. Ohlsson, L. Ljung, and S. Boyd. Segmentation of ARX-models using sum-of-norms regularization. *Automatica*, 46 (6):1107–1111, 2010c.

J. M. Peña, J. A. Lozano, and P. Larrañaga. An empirical comparison of four initialization methods for the k-means algorithm. *Pattern Recognition Letters*, 20:1027–1040, October 1999.

K. Pelckmans, J. De Brabanter, J.A.K. Suykens, and B. De Moor. Convex clustering shrinkage. In *Statistics and Optimization of Clustering Workshop (PASCAL)*, London, U.K., July 2005. Lirias number: 181608.

J. Roll. Piecewise linear solution paths with application to direct weight optimization. *Automatica*, 44:2745–2753, 2008.

H. Steinhaus. Sur la division des corp materiels en parties. *Bull. Acad. Polon. Sci*, 1:801–804, 1956.

R. Tibsharani. Regression shrinkage and selection via the lasso. *Journal of Royal Statistical Society B (Methodological)*, 58(1):267–288, 1996.

A. Vattani. k-means requires exponentially many iterations even in the plane. In *Proceedings of the 25th Annual Symposium on Computational Geometry (SCG)*, Aarhus, Denmark, June 2009. doi: 10.1145/1542362.1542419.

R. Xu and II Wunsch, D. Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, 16(3):645–678, May 2005.

M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society, Series B*, 68:49–67, 2006.