Minimum sum-of-squares clustering

Pierre Hansen and Daniel Aloise

GERAD, HEC Montréal and
LIX, École Polytechnique, Palaiseau.

Bordeaux, Mars 2009

Outline

Outline

Outline

Clustering

- Clustering is a powerful tool for automated analysis of data.
- Given a set of entities, find subsets, or clusters, which are homogeneous and/or well separated.
- Among many criteria used in cluster analysis, the minimum sum-of-squares is one of the most used.

Minimum Sum-of-Squares Clustering (MSSC)

Input:

- A set of n entities $O = \{o_1, o_2, \ldots, o_n\}$ at given points $p_i = (p_{ik}, k = 1, \ldots, s)$ of $\mathbb{R}^s$ for $i = 1, \ldots, n$;
- A fixed number c of clusters.

The objective is to find c cluster centers located in points $z_j \in \mathbb{R}^s$ for $j = 1, \ldots, c$ in order to minimize

$$\sum_{i=1}^{n} \sum_{j=1}^{c} w_{ij} \|p_i - z_j\|^2,$$

where the binary variables w_{ij} express the assignment of the entity o_i to the cluster j.

Minimum Sum-of-Squares Clustering (MSSC)

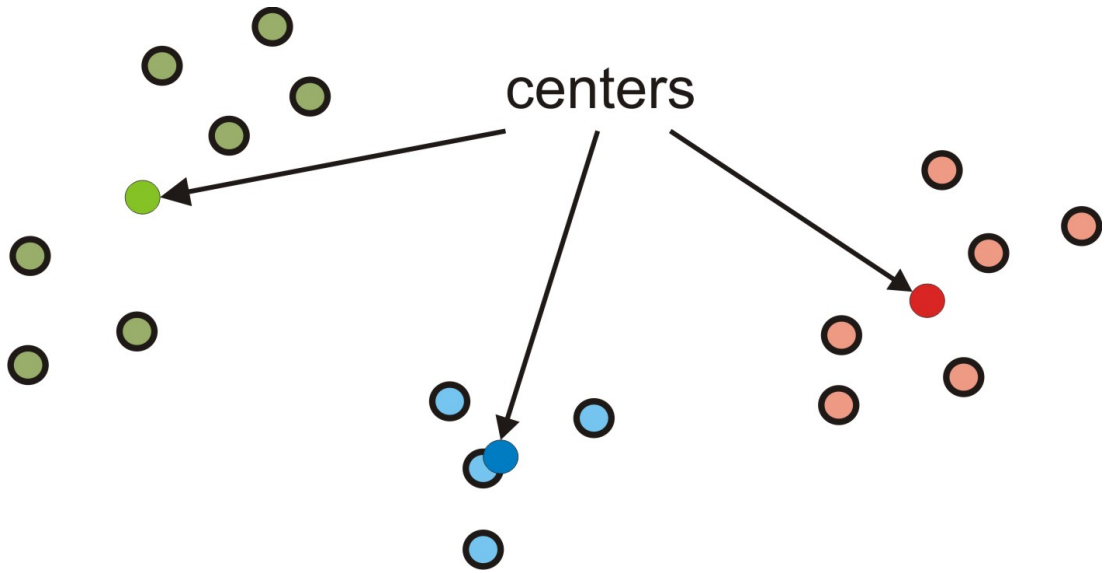
Each entity can be assigned to exactly one cluster

$$\sum_{j=1}^{c} w_{ij} = 1, \quad \forall i = 1, \ldots, n.$$

Minimum Sum-of-Squares Clustering (MSSC)

Each entity can be assigned to exactly one cluster

$$\sum_{j=1}^{c} w_{ij} = 1, \quad \forall i = 1, \ldots, n.$$



Minimum Sum-of-Squares Clustering (MSSC)

- For one cluster the problem is easy.
- The first order KKT conditions require that

$$\sum_{i=1}^{n}(z_{jk} - p_{ik}) = 0, \quad \forall k,$$

i.e.,

$$z_{jk} = \frac{\sum_{i=1}^{n} p_{ik}}{n}, \quad \forall k.$$

Minimum Sum-of-Squares Clustering (MSSC)

- For one cluster the problem is easy.
- The first order KKT conditions require that

$$\sum_{i=1}^{n}(z_{jk} - p_{ik}) = 0, \quad \forall k,$$

i.e.,

$$z_{jk} = \frac{\sum_{i=1}^{n} p_{ik}}{n}, \quad \forall k.$$

cluster center = centroid

Minimum Sum-of-Squares Clustering (MSSC)

MSSC has several properties:

- Given the assignments, the cluster centers are located in the centroids of the classes;
- Given the centroids, each entity is assigned to its closest centroid;
- It expresses both homogeneity and separation;
- Monotonicity;
- Sphere-like clusters;
- Several thousand papers have been written on exact methods, heuristics and applications.

We focus here on exact methods.

Computational complexity

- The computational complexity of a clustering problem depends on the criterion used.
- For instance, split maximization is polynomially solvable (Delattre and Hansen, 1980) while diameter minimization is NP-hard (Brücker, 1978; Hansen and Delattre, 1978).
- Regarding computational complexity,
  - for $c \geq 2$ and one dimensional data, MSSC can be solved in $O(n^3)$ time (Späth, 1980).
  - The problem is NP-hard in the plane for general values of c (Mahajan, Nimbhorkar and Varadarajan, 2009).
  - In general dimension, MSSC is NP-hard even for $c = 2$ (Aloise, Deshpande, Hansen and Popat, 2009).
  - If both c and dimension s are fixed, the problem can be solved in $O(n^{sc+1})$ time (Inaba, Katoh and Imai, 1994), which may be very time-consuming even for instances in the plane.

Computational complexity (Aloise, Deshpande, Hansen and Popat, 2009)

Theorem

MSSC in general dimension is NP-hard for $k = 2$

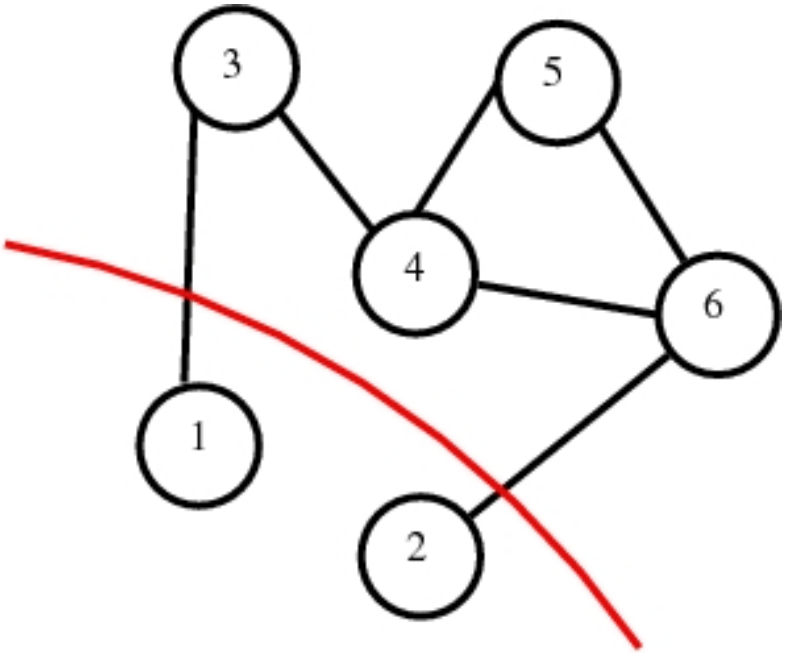
- The reduction is from the densest cut problem, whose objective is to maximize for a given graph $G = (V, E)$ the ratio $|E(P, Q)|/|P| \cdot |Q|$ over all bipartitions (P, Q) of the vertices in G, where $E(P, Q)$ denotes the edge set of the cut.
- The problem is equivalent to the sparsest cut problem on the complement graph, which was shown to be NP-hard by (Matula and Shahrokhi, 1990).

Computational complexity (Aloise, Deshpande, Hansen and Popat, 2009)

- Given a graph G with no parallel edges, let us define a $|V|$ by $|E|$ matrix M
- An entry (v, e) in M is equal to 0, if edge $e \in E$ is not incident to vertex $v \in V$. Otherwise, it is $+1$ for one endpoint of e and -1 for the other.
- Thus, each column of M has exactly one entry equal to $+1$ and exactly one entry equal to -1.

Computational complexity (Aloise, Deshpande, Hansen and Popat, 2009)

Example



$$M = \left[\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ \hline -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 1 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 & -1 & -1 \end{array}\right]$$

Computational complexity (Aloise, Deshpande, Hansen and Popat, 2009)

- Let us suppose that the rows of M are points in $\mathbb{R}^{|E|}$ and compute the value of the MSSC criterion for a bipartition into two clusters P and Q, with $|P| = p$, $|Q| = q$ and $p + q = n$.
- The centroid of cluster P has in its e-th coordinate a value equal to either $+1/p$ or $-1/p$ if $e \in E(P, Q)$, or 0 otherwise.
- The same holds for the coordinates of the centroid of cluster Q.

Computational complexity (Aloise, Deshpande, Hansen and Popat, 2009)

Then, by computing the total cost of the bipartition, we have that

$$\sum_{e \in E} \text{cost of } P \text{ due to the } e\text{-th coordinate} + \text{cost of } Q \text{ due to the } e\text{-th coordinate}$$

$$\begin{aligned}
&= \sum_{e \in E(P,Q)} (p-1)\frac{1}{p^2} + \left(1 - \frac{1}{p}\right)^2 + (q-1)\frac{1}{q^2} + \left(1 - \frac{1}{q}\right)^2 + \sum_{e \notin E(P,Q)} 2 \\
&= \left(2 - \frac{1}{p} - \frac{1}{q}\right) |E(P,Q)| + 2|E(P,P)| + 2|E(Q,Q)| \\
&= 2|E| - \frac{n}{p \cdot q} |E(P,Q)|,
\end{aligned}$$

by using $p + q = n$. The MSSC for $k = 2$ minimizes the above, which means that it maximizes $|E(P,Q)|/p \cdot q$ and hence finds the densest cut in the given graph G.

k-means

- The most popular heuristic for MSSC is k-means.
- Voted by the IEEE Computer Society as the #2 most important algorithm in Data Mining.
- It does not provide a global optimum. It can lead to very bad solutions (more than twice the optimal value if there are many entities and clusters, Hansen and Mladenović, *Pattern Recognition*, 2001).
- Popular due to its simplicity and fast local optimum convergence observed in practice.
- A much better algorithm is j-means.

k-means

Algorithm

1. Start from an initial partition;
2. *while* stability is not reached *do*
    1. Reassign the entities to their closest centroids;
    2. Update the position of centroids;

Because reassignments are performed only if profitable and the number of partitions is finite, we can conclude that *k*-means always converges to a local minimum.

Exact Methods

Reformulation-Linearization Technique (Sherali and Desai, 2005)

- The underlying nonlinear, discrete optimization problem is transformed into an equivalent 0-1 MIP by optimizing the convex envelope of the objective function (i.e., replacing products of two variables by a new one), taking products of variables and constraints from the convex hull of entities, linearizing again and performing branch-and-bound.
- Such procedure is done by the reformulation-linearization technique (RLT), Sherali & Adams (1990).
- A branch-and-bound algorithm is devised with lower bounds obtained by RLT.

RLT model

The MSSC objective function is equivalent to

$$\begin{aligned}
& \sum_{i \in I_j}^{n} \sum_{j=1}^{c} \sum_{k=1}^{s} w_{ij}(z_{jk} - p_{ik})^2 \\
= & \sum_{i \in I_j}^{n} \sum_{j=1}^{c} \sum_{k=1}^{s} w_{ij}(z_{jk} - p_{ik}) z_{jk} - \sum_{i=1}^{n} \sum_{j=1}^{c} \sum_{k=1}^{s} w_{ij}(z_{jk} - p_{ik}) p_{ik} \\
= & \sum_{i \in I_j}^{n} \sum_{j=1}^{c} \sum_{k=1}^{s} w_{ij} p_{ik}^2 - \sum_{i=1}^{n} \sum_{j=1}^{c} \sum_{k=1}^{s} p_{ik} w_{ij} z_{jk}.
\end{aligned}$$

$\overset{RLT}{\Rightarrow}$ Maximize $\sum_{i \in I_j}^{n} \sum_{j=1}^{c} \sum_{k=1}^{s} p_{ik} y_{ijk}$

RLT model

First-order KKT conditions

$$z_{jk} = \frac{\sum_{i \in I_j} w_{ij} p_{ik}}{\sum_{i \in I_j} w_{ij}}, \quad \forall j, k$$

$$\overset{RLT}{\Rightarrow} \sum_{i \in I_j} y_{ijk} - \sum_{i \in I_j} p_{ik} w_{ij} = 0$$

RLT model

Variables z_j for $j = 1, \ldots, c$ can be confined to lie in the convex hull of the points that can be assigned to cluster j, denoted I_j.

$$\overline{H}(I_j) = \{z_j : \alpha_j^k \leq z_{jk} \leq \beta_j^k, k = 1, \ldots, s\},$$

where, $\alpha_j^k = \min\{p_{ik} : o_i \in I_j\}$ and $\beta_j^k = \max\{p_{ik} : o_i \in I_j\}$, $\forall k = 1, \ldots, s$.

$$\overset{RLT}{\Rightarrow} \quad \alpha_j^k w_{ij} \leq z_{jk} w_{ij} \leq \beta_j^k w_{ij} \quad \forall i \in I_j \ \forall j, k$$
$$\alpha_j^k (1 - w_{ij}) \leq z_{jk}(1 - w_{ij}) \leq \beta_j^k (1 - w_{ij}) \quad \forall i \in I_j \ \forall j, k$$

$$\overset{RLT}{\Rightarrow} \quad \alpha_j^k w_{ij} \leq y_{ijk} \leq \beta_j^k w_{ij} \quad \forall i \in I_j \ \forall j, k$$
$$\alpha_j^k (1 - w_{ij}) \leq z_{jk} - y_{ijk} \leq \beta_j^k (1 - w_{ij}) \quad \forall i \in I_j \ \forall j, k$$

RLT model

$$\text{Max} \sum_{i \in I_j} \sum_{j=1}^{c} \sum_{k=1}^{s} p_{ik} y_{ijk}$$

$$\sum_{i \in I_j} y_{ijk} - \sum_{i \in I_j} p_{ik} w_{ij} = 0, \qquad \forall j, k$$

$$\alpha_j^k w_{ij} \leq y_{ijk} \leq \beta_j^k w_{ij} \qquad \forall i \in I_j \;\; \forall j, k$$

$$\alpha_j^k (1 - w_{ij}) \leq z_{jk} - y_{ijk} \leq \beta_j^k (1 - w_{ij}) \qquad \forall i \in I_j \;\; \forall j, k$$

$$\sum_{j=1}^{c} w_{ij} = 1, \qquad \forall i$$

$$\sum_{i \in I_j} w_{ij} \geq 1, \qquad \forall j$$

$$z_{jk} \leq \textstyle\sum_{i \in I_j} y_{ijk} \leq (n - c + 1) z_{jk} \qquad \forall j, k$$

constraints for breaking symmetry

w binary

Attempted Reproduction of Results

- Except for the platform used, a Pentium IV 512 MB RAM under Linux, the implementations were supposed to be equivalent.
- We could not reproduce the results reported in Sherali and Desai (2005), i.e., problems with n up to 1000. Data were not described precisely and are no more available, having been deleted.
- With both direct application of CPLEX 10.1 and a software we wrote, we could solve problems with n up to 22 only. We believe that either there are some errors in the Desai and Sherali results or instances generated are extremely easy (arbitrarily large instances can be solved without branching provided the clusters are compact and pairwise far one from the other).

Cutting plane algorithm (Peng and Xia, 2005)

- Peng and Xia, 2005 proved that the objective function of MSSC is concave in the relaxed feasible domain.
- A large number of approaches for concave minimization are traced back to Tuy's cutting algorithm (Tuy, 1964) for minimizing a concave function over a full dimensional polytope.
- Basic ideas of Tuy's cuts:
  - Let x^0 be a local minimum and a vertex of the feasible domain. Denote $\gamma = f(x^0)$. If D is full-dimensional, x^0 has at least n adjacent vertices. Let $y^1, \ldots, y^p$ denote the vertices adjacent to x^0 ($p \geq n$).
  - For $i = 1, \ldots, n$, let

$$\theta_i \stackrel{def}{=} \sup\{t : t \geq 0, f(x^0 + t(y^i - x^0)) \geq \gamma\},$$

and denote

$$r^i \stackrel{def}{=} x^0 + \theta_i(y^i - x^0).$$

Tuy cuts

- Since f is concave, any point in the simplex $Spx \stackrel{def}{=} conv\{x^0, r^1, \ldots, r^n\}$ has objective value greater than γ, and therefore, it can be cut off from further search for a global minimum.
- Since x^0 is a vertex of a full dimensional space, one can always find n binding constraints at x^0, and x^0 has n linearly independent edges.
- Define

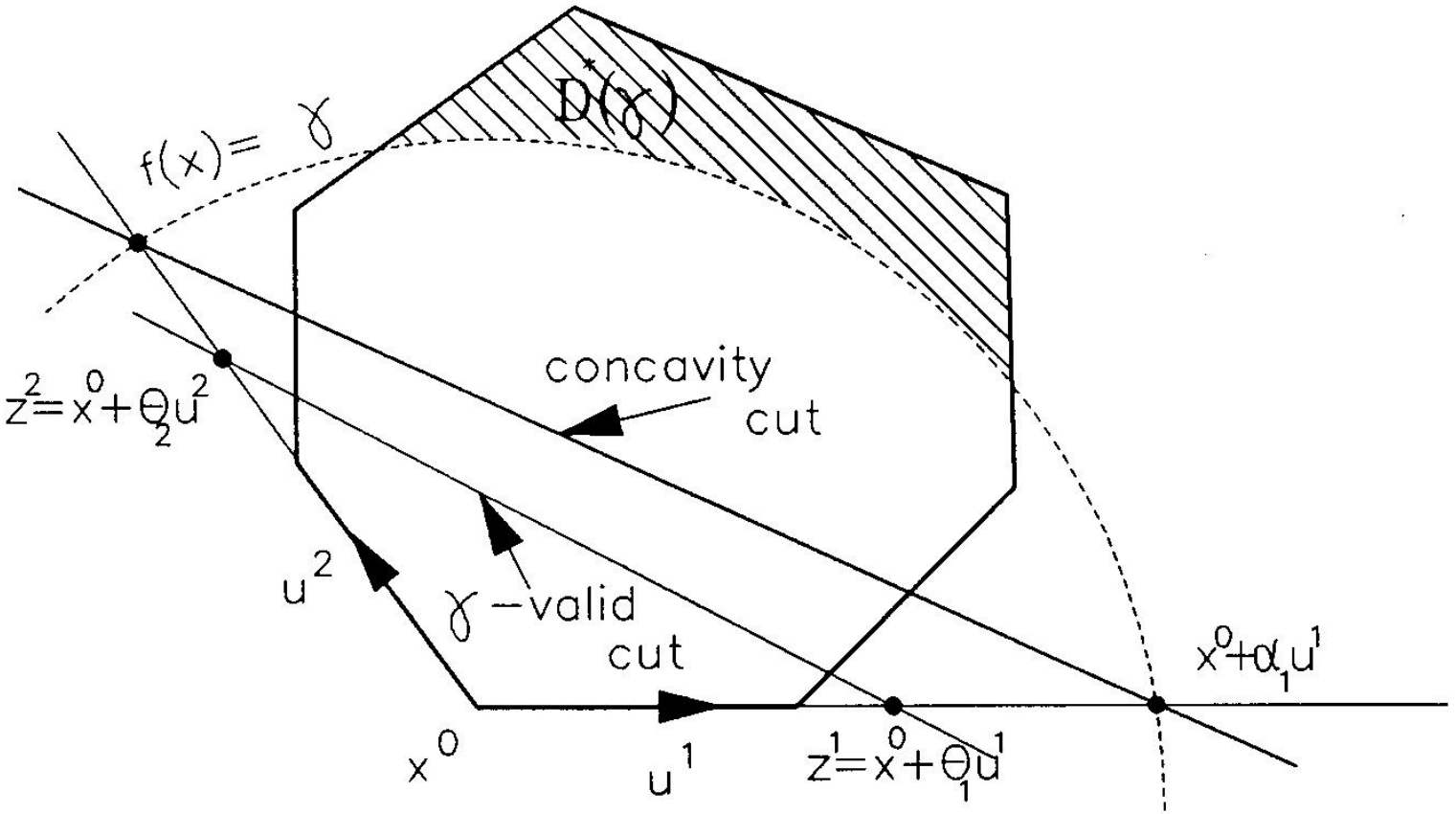
$$\pi = e^T Diag(\frac{1}{\theta_1}, \ldots, \frac{1}{\theta_n}) U^{-1}, U = [y^1 - x^0, \ldots, y^n - x^0].$$

Then the inequality

$$\pi(x - x^0) > 1$$

provides a γ-valid concavity cut for f in the feasible domain.

Tuy cuts


f(x)= γ
D*(γ)
concavity
cut
z²=x⁰+θ₂u²
u²
γ −valid
cut
x⁰+α₁u¹
x⁰
u¹
z¹=x⁰+θ₁u¹

Tuy cutting algorithm

- Search for a vertex x^0 which is a local minimum. Set $\gamma = f(x^0)$, $D_0 = D$.
- Iteration $i = 0, 1, \ldots$
  1. At x^i construct a γ-valid concavity cut π^i for (f, D_i).
  2. Solve the linear program

$$\max \pi^i(x - x^i) \quad s.t.\ x \in D_i.$$

     Let ω^i be an optimum of this LP. If $\pi^i(\omega^i - x^i) \leq 1$, then stop: x^0 is a global minimum. Otherwise, go to step 3.
  3. Let $D_{i+1} = D_i \cap \{x : \pi^i(x - x^i) \geq 1\}$. Starting from ω^i find a vertex x^{i+1} of D_{i+1} which is a local minimum of $f(x)$ over D_{i+1}.
  4. If $f(x^{i+1}) \geq \gamma$, then go to iteration $i + 1$. Otherwise, set $\gamma = f(x^{i+1})$, $x^0 \leftarrow x^{i+1}$, $D_0 \leftarrow D_{i+1}$, and go to iteration 0.

Tuy cutting algorithm for MSSC

- An adaptation of Tuy's cutting algorithm is proposed in Peng and Xia, 2005.
- This is required since the feasible domain does not have full dimension – each vertex is adjacent to only $n \times (c - 1)$ other vertices in the feasible domain.

Adjacent vertices

- Peng and Xia provide a method to build the matrix U, by considering infeasible directions from the local minimum.
- Let $X^0 \in \mathbb{R}^{n \times k}$ be the current feasible local minimum in step 3 of Tuy's cutting algorithm. Then, let $E_{i,j}$ denote the matrix whose (i, j) entry is 1, while the other entries are zero.
- For $\ell = 1, \ldots, n$, assume o_ℓ is assigned to cluster ℓ_j. Let $Y^{\ell,k}(k = 1, \ldots, c; k \neq \ell_j)$ be the matrix different from X^0 only by the assignment of o_ℓ to cluster k.
- Choose $1 \leq \ell_p \leq c$, $\ell_p \neq \ell_j$, and let Y^{ℓ,ℓ_j} denote the matrix different from X^0 only in its (ℓ, ℓ_p) entry being 1 as well, i.e.,

$$Y^{\ell,k} = \begin{cases} X^0 - E_{\ell,\ell_j} + E_{\ell,k} & k \neq \ell_j \\ X^0 + E_{\ell,\ell_p} & k = \ell_j \end{cases}$$

Adjacent vertices

- Then, $Y^{\ell,k}$ $(\ell = 1, \ldots, n,\ k = 1, \ldots, c)$ are $n \times c$ adjacent vertices of X^0.
- Vector x^0 is formed by stacking all the columns of X^0 together. Similarly, the vectors $y^{\ell,k}$ are formed.
- Thus, $U = [y^{1,1} - x^0, \ldots, y^{1,c} - x^0, \ldots, y^{n,c} - x^0]$ has full rank.

Tuy cutting algorithm for MSSC

- Local minima are obtained by a local search in the 1-exchange neighborhood followed by a procedure for restoring feasibility. Thus, the algorithm guarantees its global convergence.
- The computational experiments refer to a heuristic algorithm which is halted after either: (i) a global optimum is found; (ii) the objective cannot be improved after 5 succeeding cuts; (iii) the total number of cuts exceeds 20.
- We tested running the algorithm up to global convergence, but only some fairly small data sets ($\approx$ 22 to 25 entities) could be solved to optimality.

Dynamic Programming (van Os & Meulman, 2004)

- $O(n^3)$ algorithm for one dimensional data (e.g. Bellman, 1973, Späth, 1980).
- MSSC can also be solved by non-serial dynamic programming as shown by Jensen, 1969.
- Let $f_j^*(O')$ denote the optimal value for partitioning the set $O' \subseteq O$ into j clusters.

$$f_j^*(O') = \begin{cases} f_j^*(O') & = \underset{O_j \subset O'}{opt} \left(f_{j-1}^*(O' - O_j) + h(O_j)\right) \text{ for } j > 1, \\ f_1^*(O') & = h(O'). \end{cases}$$

- An improved dynamic programming implementation (van Os & Meulman, 2004) appears to present a limit of $n \approx 25$ on the size of benchmark instances exactly solved.

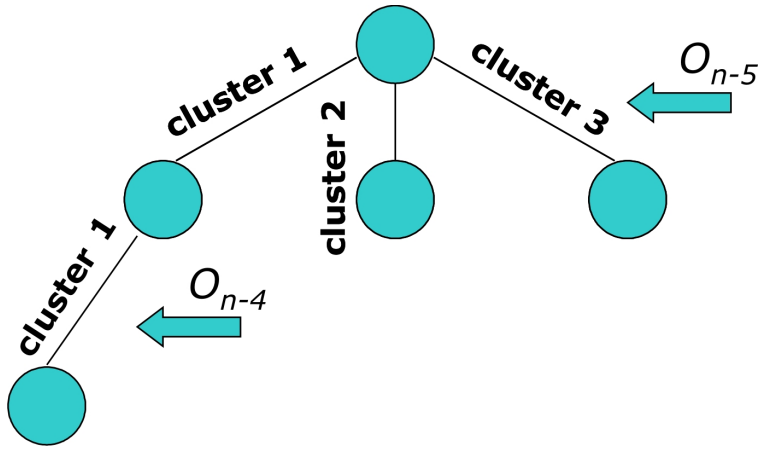
Repetitive Branch-and-Bound Algorithm (Brusco, 2006)

- Branch-and-bound algorithms were previously proposed in the literature (Koontz et al., 1975; Diehr, 1985).
- Their main characteristic is that lower bounds are obtained from values due to assigned entities plus loose bounds on values due to those which remain to be assigned.
- The major contribution of RBBA is that after ordering the entities, subproblem with the $c + 1$ last entities, then with one more entity each time, are solved in sequence by repeating a branch-and-bound procedure.
- In RBBA, the resolution of a given subproblem by branch-and-bound depends on the optimal solutions of smaller subproblems already solved by the algorithm, as look-ahead component in the bound.

Branch-and-bound tree

Let us suppose we are solving a subproblem with six entities.

- At this point, RBBA has already exactly solved the subproblem with five entities $\{o_{n-4}, o_{n-3}, o_{n-2}, o_{n-1}, o_n\}$ and that with four entities $\{o_{n-3}, o_{n-2}, o_{n-1}, o_n\}$.



- The branch-and-bound nodes are explored by a depth-first strategy.
- Each level of the tree is associated to an entity, and each branching associated to a particular cluster assignment.
- If there is no possible assignment of the remaining entities that can produce an optimal solution, the node is pruned, i.e., $LB \geq UB$.

RBBA

- *UB* for the r-th subproblem is obtained from the optimal solution of the $(r-1)$-th subproblem.
- *LB*s are defined for each branch-and-bound node ν as

$$ASSIGN_{\ell,\nu} + OPT_{r-\ell},$$

where $ASSIGN_{\ell,\nu}$ refers to the cost of the partition that results from the ℓ assignments done up to node ν, and $OPT_{r-\ell}$ is the optimal solution of the $(r-\ell)$-th subproblem.

- *LB*s may actually decrease while we walk down in the branch-and-bound tree.

Semidefinite Programming (Peng and Xia, 2005)

- Peng and Xia, 2005 used matrix arguments to model MSSC as a so-called 0-1 semidefinite programming (SDP).
- Canonical SDP

$$(SDP)\begin{cases} \min & Tr(WZ) \\ s.t. & Tr(B_i Z) = b_i \quad \text{for } i = 1, \ldots, m \\ & Z \succeq 0 \end{cases}$$

- General 0-1 SDP

$$(0-1SDP)\begin{cases} \min & Tr(WZ) \\ s.t. & Tr(B_i Z) = b_i \quad \text{for } i = 1, \ldots, m \\ & Z^2 = Z, Z = Z^T \end{cases}$$

SDP

Theorem (Peng and Xia, 2005)

Solving the 0-1 SDP problem

$$\begin{array}{ll} \min & Tr(W_p W_p^T (I - Z)) \\ s.t. & Ze = e, Tr(Z) = c, \\ & Z \geq 0, Z = Z^T, Z^2 = Z. \end{array}$$

is equivalent to finding a global solution of the MSSC.

$(\Rightarrow)$

By rearranging the MSSC cost function which results from Huygens' theorem, we have that

$$\begin{aligned}\sum_{j=1}^{c} \frac{\sum_{i=1}^{n-1} \sum_{k=i+1}^{n} \|p_i - p_k\|^2 x_{ij} x_{kj}}{\sum_{i=1}^{n} x_{ij}} &= \sum_{i=1}^{n} \|p_i\|^2 - \sum_{j=1}^{c} \frac{\| \sum_{i=1}^{n} x_{ij} p_i \|^2}{\sum_{i=1}^{n} x_{ij}} \\ &= Tr(W_p^T W_p) - \sum_{j=1}^{c} \frac{\| \sum_{i=1}^{n} x_{ij} p_i \|^2}{\sum_{i=1}^{n} x_{ij}},\end{aligned}$$

where $W_p \in \mathbb{R}^{n \times s}$ is the matrix whose ith row is the vector p_i. Suppose X a feasible assignment matrix, let

$$Z = X(X^T X)^{-1} X^T$$

Then, the objective function can be rewritten as

$$Tr(W_p W_p^T (I - Z)) = Tr(W_p^T W_p) - Tr(W_p^T W_p Z).$$

We can notice that Z is a matrix that satisfies $Z^2 = Z$ and $Z = Z^T$ with nonnegative elements. We can also write the constraint $\sum_{j=1}^{c} x_{ij} = 1$ as

$$Xe^c = e^n,$$

which implies that

$$Ze^n = ZXe^c = Xe^c = e^n$$

Moreover, the trace of Z should be equal to c

$$Tr(Z) = c \quad \square$$

($\Leftarrow$) (see Peng and Xia, 2005).

Valid inequalities

Since Z is s.d.p. there exists an index $i_1 \in 1, \ldots, n$ such that

$$Z_{i_1 i_1} = \max_{i,j} Z_{ij} > 0.$$

Let us define the index set

$$\mathcal{I}_1 = \{j : Z_{i_1 j} > 0\}.$$

Since $Z^2 = Z$, we have

$$\sum_{j \in \mathcal{I}_1} (Z_{i_1 j})^2 = Z_{i_1 i_1},$$

which implies that

$$\sum_{j \in \mathcal{I}_1} \frac{Z_{i_1 j}}{Z_{i_1 i_1}} Z_{i_1 j} = 1.$$

From the choice of i_1 and the constraint

$$\sum_{j=1}^{n} z_{i_1 j} = \sum_{j \in \mathcal{I}_1} z_{i_1 j} = 1,$$

we conclude that

$$z_{i_1 j} = z_{i_1 i_1}, \quad \forall j \in \mathcal{I}_1.$$

By repeating the process, the following valid inequalities are obtained

$$z_{i_\beta j} = z_{i_\beta i_\beta}, \quad \forall j \in \mathcal{I}_\beta, \beta = 1, \ldots, c.$$

LP relaxation

- Peng and Xia, 2005 have proposed a LP relaxation for the MSSC 0-1 SDP formulation by removing the constraint that $Z^2 = Z$.
- In its place, triangle inequalities are used to strengthen the model.

$$
\begin{aligned}
\min \quad & Tr(W_p W_p^T (I - Z)) \\
s.t. \quad & Ze = e, Tr(Z) = c, \\
& Z \geq 0 \\
& Z_{ij} \leq Z_{ii} \quad \forall i, j \\
& Z_{ij} + Z_{ik} \leq Z_{ii} + Z_{jk} \quad \forall i, j, k
\end{aligned}
$$

LP relaxation

- The lower bounds provided by this LP relaxation are exceptionally good.
- The tightness of the triangle inequalities had already been assessed by Grötschel and Wakabayashi, 1989 in the context of the clique partitioning problem.
- However, Peng and Xia claim that its resolution is unpractical for large-sized data due to the huge amount $O(n^3)$ of triangular inequalities.
- We proposed in Aloise and Hansen, 2008 to tackle this limitation via a cutting plane procedure which adds triangular inequalities only if they are violated.
- A branch-and-cut algorithm was devised.

Branching scheme

Although their paper is not focused on exact methods, Peng and Xia, 2005 also suggest a simple branching scheme for a still to come exact method to the 0-1 SDP MSSC model.
Suppose that for the optimal solution of the LP relaxation there are indices i, j such that

$$Z_{ij}(Z_{ii} - Z_{ij}) \neq 0,$$

then we can produce a branch with $Z_{ii} = Z_{ij}$ and another one with $Z_{ij} = 0$.

In our exact method implementation, indices i, j are chosen as the $\text{argmax}_{i,j} \min\{Z_{ij}, Z_{ii} - Z_{ij}\}$.

Column generation and Interior point algorithm (du Merle et al., 2000)

- In du Merle et al. 2000, a column generation and interior point algorithm is proposed in order to solve exactly larger instances than previous exact methods.
- Partitioning problems in cluster analysis can be formulated by considering all possible clusters.
- From Huygens' theorem, the cost of a cluster C_t is equal to

$$c_t = \frac{1}{|C_t|} \sum_{i,j:o_i,o_j \in C_t} \|p_i - p_j\|^2$$

Then let

$$a_{kt} = \begin{cases} 1 & \text{if entity } o_k \text{ belongs to cluster } C_t \\ 0 & \text{otherwise} \end{cases}$$

A column generation formulation for the MSSC is given by

$$
\begin{array}{ll}
\text{Minimize } \sum_{t \in T} c_t y_t & \\
\text{subject to} & \\
\sum_{t \in T} a_{kt} y_t \geq 1 & k = 1, \ldots, n \\
\sum_{t \in T} y_t \leq c & \\
y_t \in \{0, 1\} & t \in T,
\end{array}
$$

where $T = \{1, \ldots, 2^n - 1\}$.

Restricted master problem

- A standard column generation method for solving the linear relaxation suffers from very slow convergence.
- Solutions are massively primal degenerate.
- du Merle et al. used the weighted version of the analytic center cutting plane method (ACCPM) of Goffin, Haurie, and Vial, 1992 to stabilize column generation (there are other methods, to be tested, e.g. using CPLEX and stabilized column generation, du Merle et al. 1999).

Auxiliary problem

- It is a hyperbolic (or fractional) program in 0-1 variables with quadratic numerator and linear denominator.

$$\sigma + \min_{a_k \in \{0,1\}} \frac{\sum_{k=1}^{n-1} \sum_{l=k+1}^{n} (\|p_k - p_l\|^2 - \lambda_k - \lambda_l) a_k a_l - \sum_{k=1}^{n} \lambda_k a_k}{\sum_{k=1}^{n} a_k}.$$

- Solved by Dinkelbach's algorithm.
- The computational bottleneck of the algorithm is the step that solves an unconstrained quadratic 0-1 program.
  - Variable Neighborhood Search (approximate solutions)
  - Exact resolution (check stopping condition)

Branching rule

- The branching rule is the standard one due to Ryan and Foster, 1981.
- It consists of finding two rows r_1 and r_2 such that there are two columns c_1 and c_2 with fractional variables at the optimum and such that

$$a_{r_1 c_1} = a_{r_2 c_1} = 1 \quad \text{and} \quad a_{r_1 c_2} = 1,\ a_{r_2 c_2} = 0.$$

- The branching is done by imposing on the one hand the constraint

$$x_{r_1} = x_{r_2},$$

i.e., both entities will be in the same cluster, and on the other hand

$$x_{r_1} + x_{r_2} \leq 1,$$

i.e., the two entities cannot be in the same cluster.

The branch-and-price algorithm of du Merle et al. solved for the first time fairly large benchmark instances, including Fisher's 150 iris.

Comparison of ACCPM with Kelley and Bundle methods

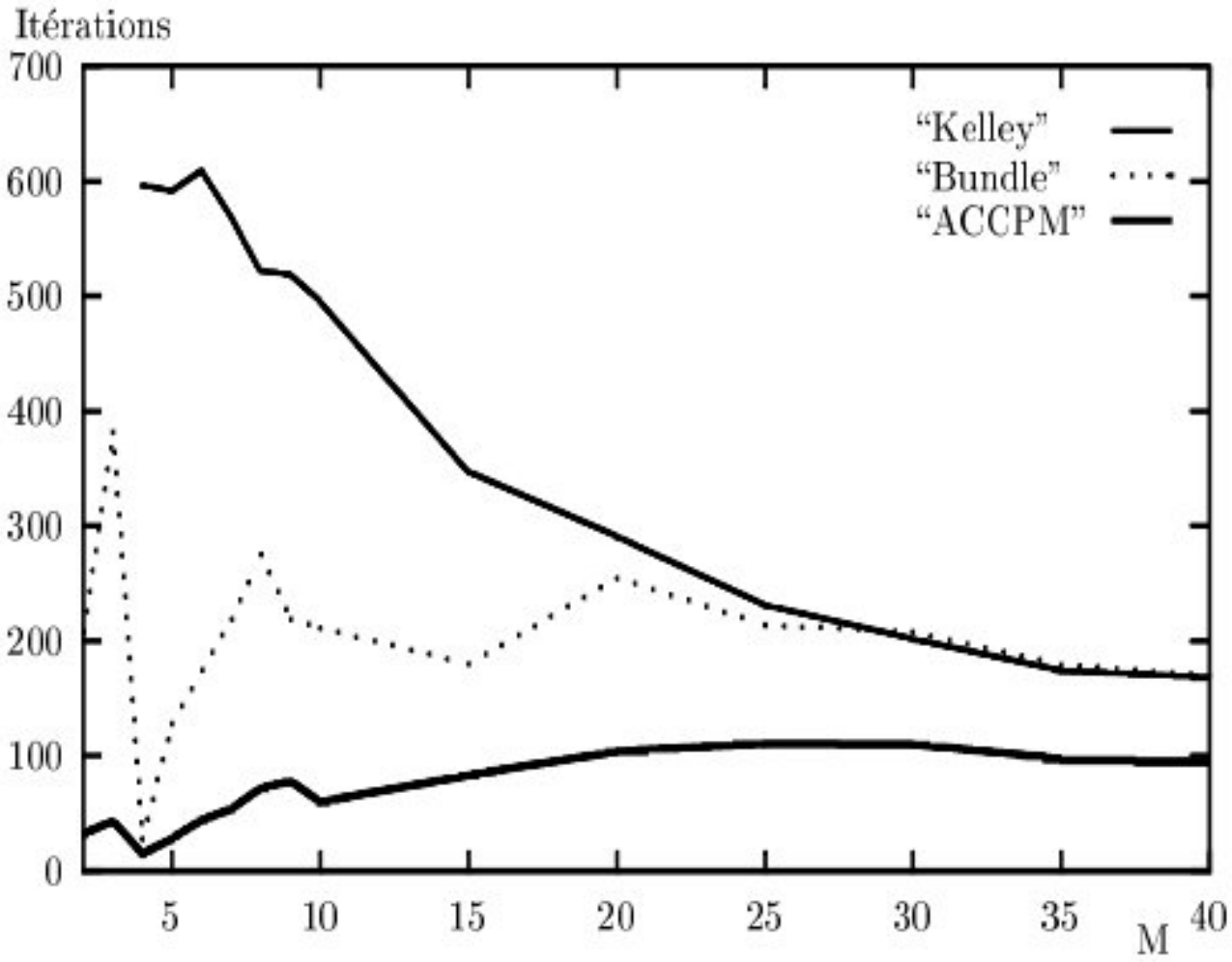


FIG. 2. *Strategies without information.*

Comparison of ACCPM with Kelley and Bundle methods

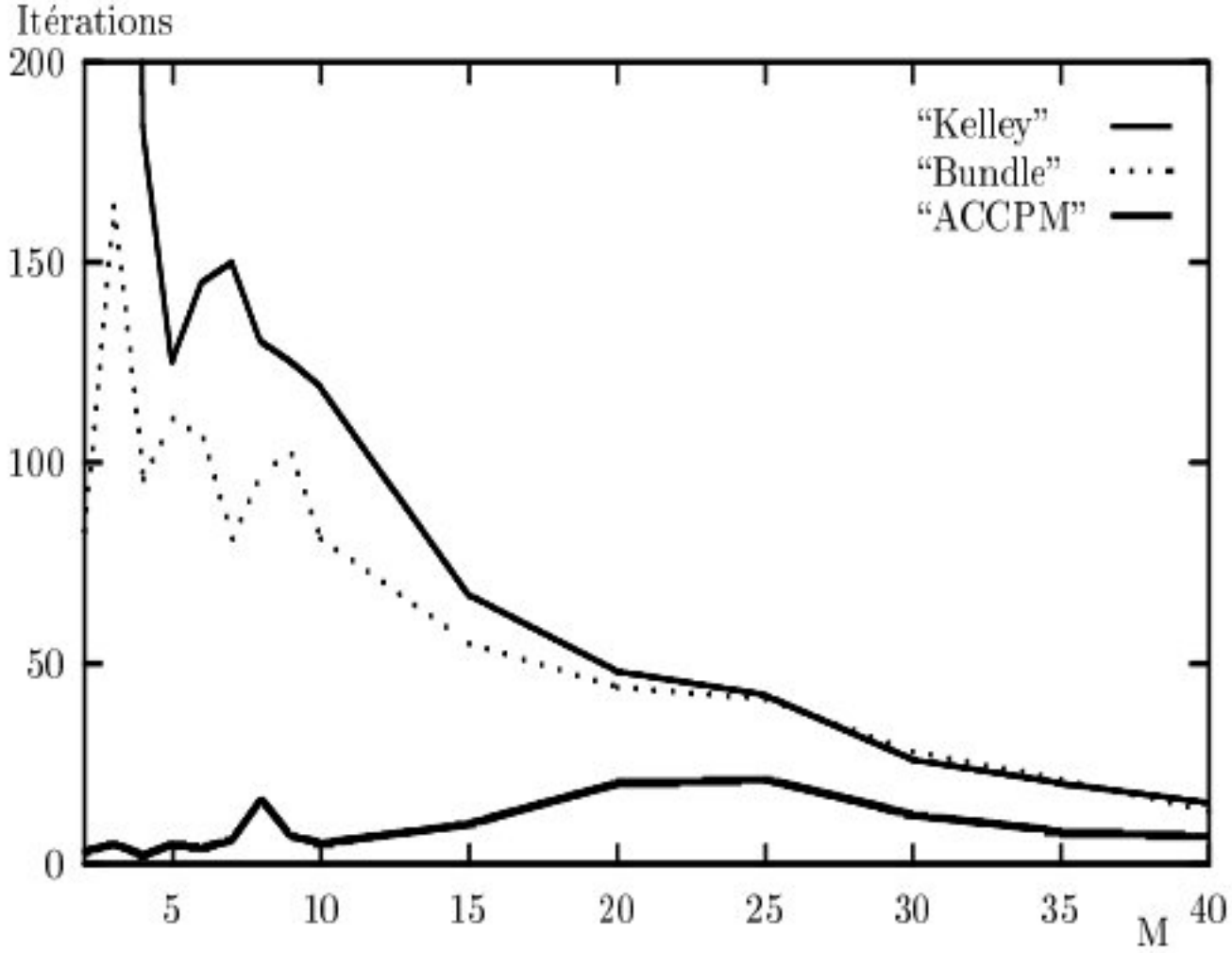


FIG. 3. *Strategies with information.*

A geometric approach in the Euclidean plane

- Recall the auxiliary problem of the column generation approach

$$\min_{a_i \in \{0,1\}} c_t + \sigma - \sum_{i=1}^{n} \lambda_i a_i = \sigma + \min_{a_i \in \{0,1\}} \sum_{i=1}^{n} a_i (\|p_i - y\|^2 - \lambda_i) \quad (1)$$

- Clearly, for a given centroid location y, a_i is equal to 1 if $\|p_i - y\|^2 \leq \lambda_i$, and to 0 otherwise.
- Geometricaly, this is equivalent to y belonging to a disc with radius $\sqrt{\lambda_i}$ centered at p_i.

A geometric approach in the Euclidean plane

$$\begin{aligned} &\min_y \ \textstyle\sum_{i \in S} \|p_i - y\|^2 \\ &\text{subject to} \\ &\|p_i - y\|^2 \leq \lambda_i \qquad\qquad \forall i \in S, \end{aligned} \tag{2}$$

Proposition 1

Let y^*, a^* be the optimal solution to (1). Then, y^*, a^* is the optimal solution to a subproblem of type (2) with a set S for which $\|p_i - y^*\|^2 > \lambda_i$ for all $i \notin S$.

A geometric approach in the Euclidean plane

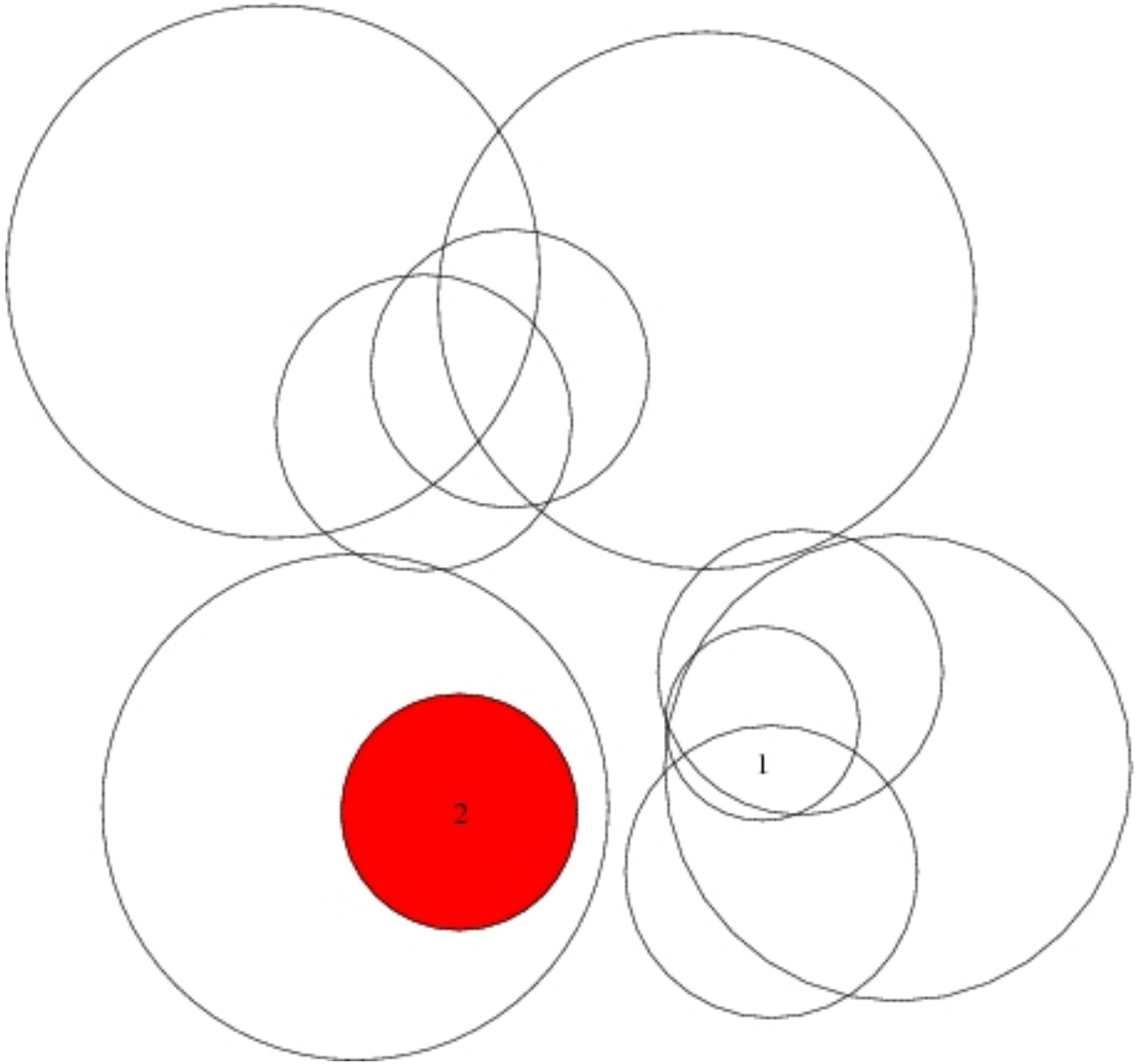
Proposition 2

Let y^*, a^* be the optimal solution to problem (1). Then, y^* must be in the centroid of points p_i for which $a_i^* = 1$ in the optimal solution.

Proposition 3

The number of distinct areas which are intersection of discs $\|p_i - y\|^2 \leq \lambda_i$ and are not reduced to a point is bounded by $2n(n-1)$.

A geometric approach in the Euclidean plane



A geometric approach in the Euclidean plane

The principle of the algorithm is to look at least one point from each area. To this effect intersection points of boundaries of discs are considered as well as one point in each isolated disc or nested discs. Then, we consider the sets of indices corresponding to each area, find the corresponding centroid and its value. The best one is kept.

A geometric approach in the Euclidean plane

Algorithm

1. Search for intersection points of pairs of convex regions in the plane as well as
   (a) Store in $\mathcal{I}_2$ sets S corresponding to areas of type (2);
   (b) Store in $\mathcal{I}_3$ sets S corresponding to areas of type (3).
2. Consider, in turn, all the intersection points of pairs of discs in the plane.
3. For intersection point $p = (p^1, p^2)$ defined by discs of points p_i and p_j, find the set S of all k such that $k \neq i, j$ and $\|p_k - p\|^2 \leq \lambda_k$.
4. Consider four sets: S, $S \cup \{i\}$, $S \cup \{j\}$, and $S \cup \{i, j\}$.
5. Solve subproblems of type (2) defined by each of these sets.
6. Update the best solution if an improving one is found.
7. Consider, in turn, all index sets in $\mathcal{I}_2$ and $\mathcal{I}_3$.
8. Solve subproblems of type(2) defined by each of these sets.
9. Update the best solution if an improving one is found.

A geometric approach in the Euclidean plane

This algorithm solves the auxiliary problem in the column generation approach. It leads to solve much larger problems than before, i.e., with up to 2392 entities and 90 clusters.

A geometric approach in the Euclidean space

- The approach in the plane can be generalized for the intersection graph of hyperspheres.
- Condition for intersection of two hyperspheres are:

$$d_{ij} \leq \sqrt{\lambda_i} + \sqrt{\lambda_j}$$

- The optimal solution of the subproblem is a clique.
- Coefficients of non-edges can be made arbitrarily large.
- The quadratic 0-1 program arising at the last iteration for the subproblem can be shrunk: includes a minimum degree vertex, delete all non-neighbors and iterates. Before next iteration, simplify the whole graph by removing that vertex and adjacent edges.

A geometric approach in the Euclidean space

This algorithm solves the auxiliary problem in the column generation approach. It leads to solve much larger problems than before, i.e., with up to 2310 entities and 230 clusters.

Conclusions

Many approaches have been explored. The problem is difficult and only fairly small instances can be solved exactly. There appear to be three groups of methods:

- RLT, dynamic programming and concave programming can only solve small instances.
- Repetitive branch-and-bound, 0-1 SDP programming and column generation solve instances about 10 times larger.
- Column generation with *geometric reasoning* solves instances about 10 times larger than these last ones.

Algorithm	**Instance Limit**	
RLT	22 German towns (Späth, 1980)	$c = 3$
Concave minimization	22 German towns (Späth, 1980)	$c = 4$
Dynamic programming	26 European jobs (Hand et al., 1994)	$n = 26$
RBBA	59 German towns (Späth, 1980)	$c = 7$
	89 Bavarian postal codes (Späth, 1980)	$c = 4$
	Fisher's 150 Iris data (Anderson, 1935)	$c = 5$
Column generation (CG)	Fisher's 150 Iris data (Anderson, 1935)	$n = 150$
SDP	Fisher's 150 Iris data (Anderson, 1935)	$n = 150$
	TSPLIB 202 cities (Grötschel and Holland, 1991)	$c = 10$
Geometric CG in $\mathbb{R}^2$	TSPLIB 2392 cities (Padberg and Rinaldi, 1991)	$c \geq 2$
Geometric CG in $\mathbb{R}^s$	2310 image segmentation in 19 dimensions (UCI)	$c = 230$