Partially Optimal Cubic Subspace Clustering

Research Project Machine Learning

Volodymyr Drobitko

Technische Universität Dresden

21.07.2025

Partial Optimality for Cubic Clique Partition Problem

Extended cost function $c\colon \binom{S}{3} \cup \binom{S}{2} \cup \emptyset \to \mathbb{R}$

Partial Optimality for Cubic Clique Partition Problem

Extended cost function $c\colon \binom{S}{3} \cup \binom{S}{2} \cup \emptyset \to \mathbb{R}$

Instance of the extended cubic clique partition problem:

$$\min_{y\colon \binom{S}{2} \to \{0,1\}} \sum_{\{a,b,c\} \in \binom{S}{3}} c_{\{a,b,c\}}\, y_{\{a,b\}}\, y_{\{b,c\}}\, y_{\{a,c\}} + c_{\{a,b\}}\, y_{\{a,b\}} + c_{\emptyset}$$

Partial Optimality for Cubic Clique Partition Problem

Extended cost function $c\colon \binom{S}{3} \cup \binom{S}{2} \cup \emptyset \to \mathbb{R}$

Instance of the extended cubic clique partition problem:

$$\min_{y\colon \binom{S}{2} \to \{0,1\}} \sum_{\{a,b,c\} \in \binom{S}{3}} c_{\{a,b,c\}}\, y_{\{a,b\}}\, y_{\{b,c\}}\, y_{\{a,c\}} + c_{\{a,b\}}\, y_{\{a,b\}} + c_{\emptyset}$$

Construct **Improving Maps** for the clustering y

Partial Optimality for Cubic Clique Partition Problem

Extended cost function $c\colon \binom{S}{3} \cup \binom{S}{2} \cup \emptyset \to \mathbb{R}$

Instance of the extended cubic clique partition problem:

$$\min_{y\colon \binom{S}{2} \to \{0,1\}} \sum_{\{a,b,c\} \in \binom{S}{3}} c_{\{a,b,c\}}\, y_{\{a,b\}}\, y_{\{b,c\}}\, y_{\{a,c\}} + c_{\{a,b\}}\, y_{\{a,b\}} + c_{\emptyset}$$

Construct **Improving Maps** for the clustering y

$\rightarrow$ **Partial Optimality Conditions:**

Partial Optimality for Cubic Clique Partition Problem

Extended cost function $c\colon \binom{S}{3} \cup \binom{S}{2} \cup \emptyset \to \mathbb{R}$

Instance of the extended cubic clique partition problem:

$$\min_{y\colon \binom{S}{2} \to \{0,1\}} \sum_{\{a,b,c\} \in \binom{S}{3}} c_{\{a,b,c\}}\, y_{\{a,b\}}\, y_{\{b,c\}}\, y_{\{a,c\}} + c_{\{a,b\}}\, y_{\{a,b\}} + c_{\emptyset}$$

Construct **Improving Maps** for the clustering y

$\rightarrow$ **Partial Optimality Conditions:**

1. Subproblem-CUT-condition (cut subset from its complement)

Partial Optimality for Cubic Clique Partition Problem

Extended cost function $c \colon \binom{S}{3} \cup \binom{S}{2} \cup \emptyset \to \mathbb{R}$

Instance of the extended cubic clique partition problem:

$$\min_{y \colon \binom{S}{2} \to \{0,1\}} \sum_{\{a,b,c\} \in \binom{S}{3}} c_{\{a,b,c\}}\, y_{\{a,b\}}\, y_{\{b,c\}}\, y_{\{a,c\}} + c_{\{a,b\}}\, y_{\{a,b\}} + c_{\emptyset}$$

Construct **Improving Maps** for the clustering y

$\rightarrow$ **Partial Optimality Conditions:**

1. Subproblem-CUT-condition (cut subset from its complement)
2. CUT-conditions (cut pairs and triples)

Partial Optimality for Cubic Clique Partition Problem

Extended cost function $c\colon \binom{S}{3} \cup \binom{S}{2} \cup \emptyset \to \mathbb{R}$

Instance of the extended cubic clique partition problem:

$$\min_{y\colon \binom{S}{2} \to \{0,1\}} \sum_{\{a,b,c\} \in \binom{S}{3}} c_{\{a,b,c\}}\, y_{\{a,b\}}\, y_{\{b,c\}}\, y_{\{a,c\}} + c_{\{a,b\}}\, y_{\{a,b\}} + c_{\emptyset}$$

Construct **Improving Maps** for the clustering y

$\rightarrow$ **Partial Optimality Conditions:**

1. Subproblem-CUT-condition (cut subset from its complement)
2. CUT-conditions (cut pairs and triples)
3. JOIN-conditions (join subsets, pairs and triples)

Partial Optimality for Cubic Clique Partition Problem

Extended cost function $c \colon \binom{S}{3} \cup \binom{S}{2} \cup \emptyset \to \mathbb{R}$

Instance of the extended cubic clique partition problem:

$$\min_{y \colon \binom{S}{2} \to \{0,1\}} \sum_{\{a,b,c\} \in \binom{S}{3}} c_{\{a,b,c\}} \, y_{\{a,b\}} \, y_{\{b,c\}} \, y_{\{a,c\}} + c_{\{a,b\}} \, y_{\{a,b\}} + c_{\emptyset}$$

Construct **Improving Maps** for the clustering y

$\rightarrow$ **Partial Optimality Conditions:**

1. Subproblem-CUT-condition (cut subset from its complement)
2. CUT-conditions (cut pairs and triples)
3. JOIN-conditions (join subsets, pairs and triples)

CUT-conditions can be applied simultaneously
JOIN-conditions cannot be applied simultaneously!

Partial Optimality for Cubic Clique Partition Problem

Extended cost function $c\colon \binom{S}{3} \cup \binom{S}{2} \cup \emptyset \to \mathbb{R}$

Instance of the extended cubic clique partition problem:

$$\min_{y\colon \binom{S}{2} \to \{0,1\}} \sum_{\{a,b,c\} \in \binom{S}{3}} c_{\{a,b,c\}}\, y_{\{a,b\}}\, y_{\{b,c\}}\, y_{\{a,c\}} + c_{\{a,b\}}\, y_{\{a,b\}} + c_{\emptyset}$$

Construct **Improving Maps** for the clustering y

→ **Partial Optimality Conditions:**

1. Subproblem-CUT-condition (cut subset from its complement)
2. CUT-conditions (cut pairs and triples)
3. JOIN-conditions (join subsets, pairs and triples)

CUT-conditions can be applied simultaneously
JOIN-conditions cannot be applied simultaneously!

Apply partial optimality connditions → solve subproblems

Partial Optimality Algorithm

Partial Optimality Algorithm:

Input: clustering y without fixed labels

while condition applied **do**

    apply subproblem-CUT-condition exhaustively

    apply one of JOIN-conditions (in effective order)

end while

apply CUT-conditions exhaustively

Output: partially optimal clustering y with some fixed labels

Partial Optimality Algorithm:

Input: clustering y without fixed labels

while condition applied **do**

  apply subproblem-CUT-condition exhaustively

  apply one of JOIN-conditions (in effective order)

end while

apply CUT-conditions exhaustively

Output: partially optimal clustering y with some fixed labels

Reduction to subproblems:

1. Subproblem-CUT-condition: fix CUT labels for element pairs from different sample subsets; solve each subset as an independent problem and accumulate the results in $c_{\emptyset}$;

Partial Optimality Algorithm

Partial Optimality Algorithm:

Input: clustering y without fixed labels

while condition applied **do**

apply subproblem-CUT-condition exhaustively

apply one of JOIN-conditions (in effective order)

end while

apply CUT-conditions exhaustively

Output: partially optimal clustering y with some fixed labels

Reduction to subproblems:

1. Subproblem-CUT-condition: fix CUT labels for element pairs from different sample subsets; solve each subset as an independent problem and accumulate the results in $c_\emptyset$;
2. JOIN-Conditions: fix JOIN labels for elements of the sample subset; add the join-cost to $c_\emptyset$; solve the problem where the subset is considered as one sample;

Subproblem-CUT and Subset-JOIN

Subproblem-CUT: cut sample subsets $R_1, R_2, \ldots, R_k$ that are only connected via non-negative costs (applied if $k > 1$)

Subproblem-CUT and Subset-JOIN

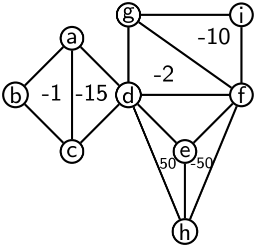
Subproblem-CUT: cut sample subsets $R_1, R_2, \ldots, R_k$ that are only connected via non-negative costs (applied if $k > 1$)

Subset-JOIN: join sample subset R with only non-positive costs if its worst bipartition joining cost is less than or equal to the reward of joining R with $\overline{R}$ (applied if $|R| > 1$)
(The worst bipartition joining cost $\approx$ min-cut)

Subproblem-CUT and Subset-JOIN

Subproblem-CUT: cut sample subsets $R_1, R_2, \ldots, R_k$ that are only connected via non-negative costs (applied if $k > 1$)

Subset-JOIN: join sample subset R with only non-positive costs if its worst bipartition joining cost is less than or equal to the reward of joining R with $\overline{R}$ (applied if $|R| > 1$)
(The worst bipartition joining cost $\approx$ min-cut)

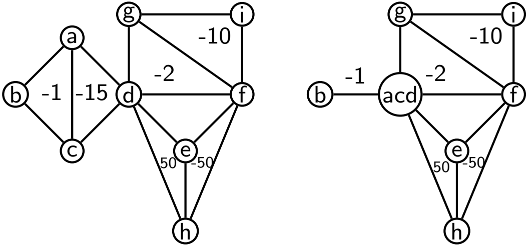


$c_\emptyset = 0$

Subproblem-CUT and Subset-JOIN

Subproblem-CUT: cut sample subsets $R_1, R_2, \ldots, R_k$ that are only connected via non-negative costs (applied if $k > 1$)

Subset-JOIN: join sample subset R with only non-positive costs if its worst bipartition joining cost is less than or equal to the reward of joining R with $\overline{R}$ (applied if $|R| > 1$)
(The worst bipartition joining cost $\approx$ min-cut)

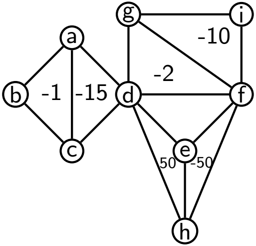


$c_\emptyset = 0$ $c_\emptyset = -15$

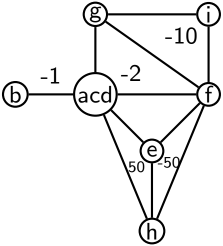
Subproblem-CUT and Subset-JOIN

Subproblem-CUT: cut sample subsets $R_1, R_2, \ldots, R_k$ that are only connected via non-negative costs (applied if $k > 1$)

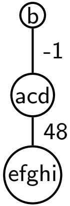
Subset-JOIN: join sample subset R with only non-positive costs if its worst bipartition joining cost is less than or equal to the reward of joining R with $\overline{R}$ (applied if $|R| > 1$)
(The worst bipartition joining cost $\approx$ min-cut)



$c_\emptyset = 0$



$c_\emptyset = -15$

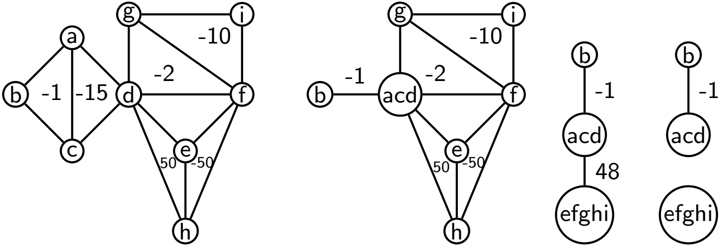


$c_\emptyset = -75$

Subproblem-CUT and Subset-JOIN

Subproblem-CUT: cut sample subsets $R_1, R_2, \ldots, R_k$ that are only connected via non-negative costs (applied if $k > 1$)

Subset-JOIN: join sample subset R with only non-positive costs if its worst bipartition joining cost is less than or equal to the reward of joining R with $\overline{R}$ (applied if $|R| > 1$)
(The worst bipartition joining cost $\approx$ min-cut)

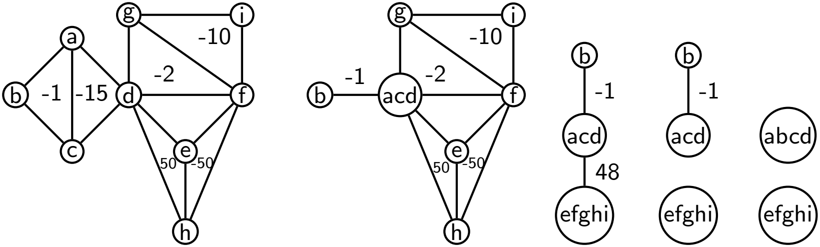


$c_\emptyset = 0$ $c_\emptyset = -15$ $c_\emptyset = -75$

Subproblem-CUT and Subset-JOIN

Subproblem-CUT: cut sample subsets $R_1, R_2, \ldots, R_k$ that are only connected via non-negative costs (applied if $k > 1$)

Subset-JOIN: join sample subset R with only non-positive costs if its worst bipartition joining cost is less than or equal to the reward of joining R with $\overline{R}$ (applied if $|R| > 1$)
(The worst bipartition joining cost $\approx$ min-cut)



$c_\emptyset = 0$ $\quad$ $c_\emptyset = -15$ $\quad$ $c_\emptyset = -75$ $\quad$ $c_\emptyset = -76$

Other JOIN-conditions and CUT-conditions

Pair-JOIN-1: join samples i, j if their overall joining reward $\geq$ the sum of rewards and penalties for joining some subset R with $i \in R$ and $\overline{R}$ with $j \in \overline{R}$ (rhs $\approx$ min-cut)

Other JOIN-conditions and CUT-conditions

Pair-JOIN-1: join samples i, j if their overall joining reward $\geq$ the sum of rewards and penalties for joining some subset R with $i \in R$ and $\overline{R}$ with $j \in \overline{R}$ (rhs $\approx$ min-cut)

Pair-JOIN-2: join samples i, k if there exists a sample triple ijk that fulfills 3 conditions (2 conditions $\approx$ min-cut, 1 explicit condition)

Other JOIN-conditions and CUT-conditions

Pair-JOIN-1: join samples i, j if their overall joining reward $\geq$ the sum of rewards and penalties for joining some subset R with $i \in R$ and $\overline{R}$ with $j \in \overline{R}$ (rhs $\approx$ min-cut)

Pair-JOIN-2: join samples i, k if there exists a sample triple ijk that fulfills 3 conditions (2 conditions $\approx$ min-cut, 1 explicit condition)

Pair-JOIN-3: join samples i, j if $\mathrm{c}_{\{i,j\}} \leq$ the sum of reward costs for joining pairs and triples containing i or j

Other JOIN-conditions and CUT-conditions

Pair-JOIN-1: join samples i, j if their overall joining reward $\geq$ the sum of rewards and penalties for joining some subset R with $i \in R$ and $\overline{R}$ with $j \in \overline{R}$ (rhs $\approx$ min-cut)

Pair-JOIN-2: join samples i, k if there exists a sample triple ijk that fulfills 3 conditions (2 conditions $\approx$ min-cut, 1 explicit condition)

Pair-JOIN-3: join samples i, j if $\mathrm{c}_{\{i,j\}} \leq$ the sum of reward costs for joining pairs and triples containing i or j

Pair-JOIN-4: join samples i, k if there exists a sample triple ijk such that 7 explicit conditions hold

Other JOIN-conditions and CUT-conditions

Pair-JOIN-1: join samples i, j if their overall joining reward $\geq$ the sum of rewards and penalties for joining some subset R with $i \in R$ and $\overline{R}$ with $j \in \overline{R}$ (rhs $\approx$ min-cut)

Pair-JOIN-2: join samples i, k if there exists a sample triple ijk that fulfills 3 conditions (2 conditions $\approx$ min-cut, 1 explicit condition)

Pair-JOIN-3: join samples i, j if $\mathrm{c}_{\{i,j\}} \leq$ the sum of reward costs for joining pairs and triples containing i or j

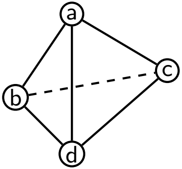
Pair-JOIN-4: join samples i, k if there exists a sample triple ijk such that 7 explicit conditions hold

Triple-JOIN: join samples i, j, k if a condition holds (similar to Pair-JOIN-1) (rhs $\approx$ min-cut)

Pyramid Example and CUT-conditions

$\mathsf{c}_{\{b,c,d\}} = 10$

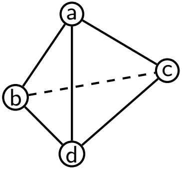
$\mathsf{c}_{\{a,b,c\}} = \mathsf{c}_{\{a,b,d\}} = \mathsf{c}_{\{a,c,d\}} = -50$

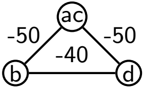


Pyramid Example and CUT-conditions

$\mathsf{c}_{\{b,c,d\}} = 10$

$\mathsf{c}_{\{a,b,c\}} = \mathsf{c}_{\{a,b,d\}} = \mathsf{c}_{\{a,c,d\}} = -50$



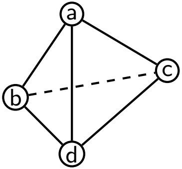


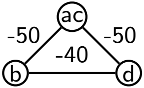
Pair-JOIN-2

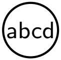
Pyramid Example and CUT-conditions

$c_{\{b,c,d\}} = 10$

$c_{\{a,b,c\}} = c_{\{a,b,d\}} = c_{\{a,c,d\}} = -50$



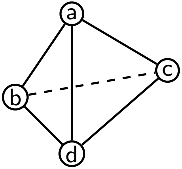


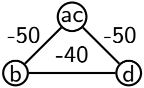


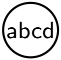
Pair-JOIN-2 Subset-JOIN $c_{\emptyset} = -140$

Pyramid Example and CUT-conditions

$c_{\{b,c,d\}} = 10$
$c_{\{a,b,c\}} = c_{\{a,b,d\}} = c_{\{a,c,d\}} = -50$







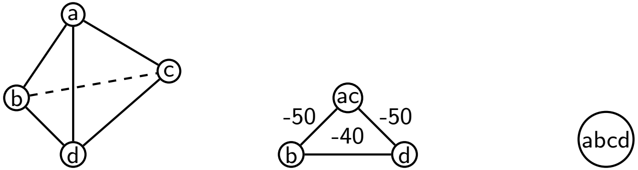
Pair-JOIN-2 Subset-JOIN $c_\emptyset = -140$

Pair-CUT: cut samples i, j if the direct joing penalty $\geq$ the sum of rewards for joining some subset R with $i \in R$ and $\overline{R}$ with $j \in \overline{R}$ (rhs $\approx$ min-cut)

Pyramid Example and CUT-conditions

$c_{\{b,c,d\}} = 10$

$c_{\{a,b,c\}} = c_{\{a,b,d\}} = c_{\{a,c,d\}} = -50$



Pair-JOIN-2 Subset-JOIN $c_\emptyset = -140$

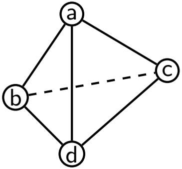
Pair-CUT: cut samples i, j if the direct joing penalty $\geq$ the sum of rewards for joining some subset R with $i \in R$ and $\overline{R}$ with $j \in \overline{R}$ (rhs $\approx$ min-cut)

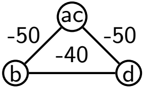
Triple-CUT:join samples i, j, k if a condition holds (similar to Pair-CUT) (rhs $\approx$ min-cut)

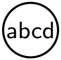
Pyramid Example and CUT-conditions

$c_{\{b,c,d\}} = 10$

$c_{\{a,b,c\}} = c_{\{a,b,d\}} = c_{\{a,c,d\}} = -50$







Pair-JOIN-2 Subset-JOIN $c_{\emptyset} = -140$

Pair-CUT: cut samples i, j if the direct joing penalty $\geq$ the sum of rewards for joining some subset R with $i \in R$ and $\overline{R}$ with $j \in \overline{R}$ (rhs $\approx$ min-cut)

Triple-CUT: join samples i, j, k if a condition holds (similar to Pair-CUT) (rhs $\approx$ min-cut)

Samples in pyramid with $c_{\{b,c,d\}} = 100$ are unjoinable!
Triple-CUT is applied to the triple *bcd*

Program Structure

TODO: class Diagram TODO: features Algorithm implementation in ClusteringProblem Features: ClusteringProblem is generally defined for all types of Cubic Clique Partition Problem (not necessarily points), cost function + sparse costs!, label computation, cut triples, logs joins and cuts! (add screenshots)