Partially Optimal Cubic Subspace Clustering

Research Project Machine Learning

Volodymyr Drobitko

Technische Universität Dresden

21.07.2025

Problem Statement (1)

Finite sample set S, cost function $\mathrm{c}\colon \binom{S}{3} \to \mathbb{R}$.
Instance of the **Cubic Clique Partition Problem**:

$$\min_{\mathrm{y}\colon \binom{S}{2} \to \{0,1\}} \sum_{abc \in \binom{S}{3}} \mathrm{c}_{abc}\, \mathrm{y}_{ab}\, \mathrm{y}_{bc}\, \mathrm{y}_{ac}$$

subject to $\mathrm{y}_{ab} + \mathrm{y}_{bc} - 1 \leq \mathrm{y}_{ac}$ for all distinct $a, b, c \in S$.

Problem Statement (1)

Finite sample set S, cost function $\mathrm{c}\colon \binom{S}{3} \to \mathbb{R}$.
Instance of the **Cubic Clique Partition Problem**:

$$\min_{\mathrm{y}\colon \binom{S}{2} \to \{0,1\}} \sum_{abc \in \binom{S}{3}} \mathrm{c}_{abc}\, \mathrm{y}_{ab}\, \mathrm{y}_{bc}\, \mathrm{y}_{ac}$$

subject to $\mathrm{y}_{ab} + \mathrm{y}_{bc} - 1 \leq \mathrm{y}_{ac}$ for all distinct $a, b, c \in S$.

Find a **partially optimal solution**, i.e. fix some labels y_{ab} for distinct $a, b \in S$

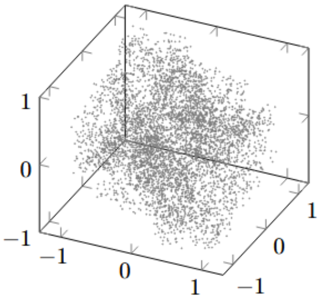
$$\begin{cases} \mathrm{y}_{ab} = 1 & \text{join } a, b \\ \mathrm{y}_{ab} = 0 & \text{cut } a, b \\ \mathrm{y}_{ab} = ? & \text{unknown} \end{cases}$$

in such way that there still exists an optimal solution.

Problem Statement (2)

Subspace Instances of the Cubic Clique Partition Problem

Samples S: points $S \subset \mathbb{R}^3$

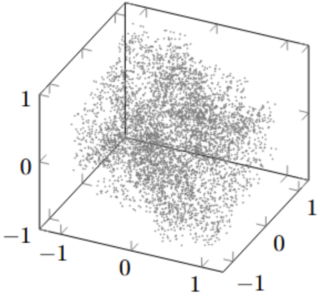


(a) Samples S

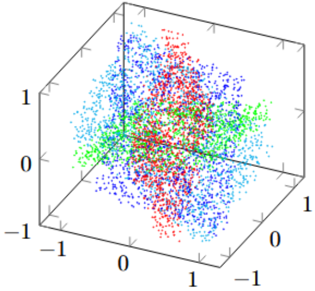
Problem Statement (2)

Subspace Instances of the Cubic Clique Partition Problem

Samples S: points $S \subset \mathbb{R}^3$
Point generation: 3 distinct planes containing the origin, noise σ



(a) Samples S

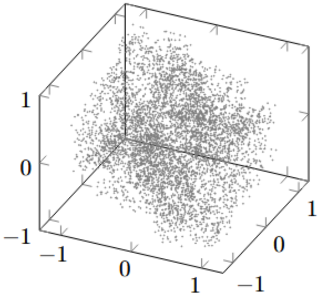


(b) Optimal labeling y^*

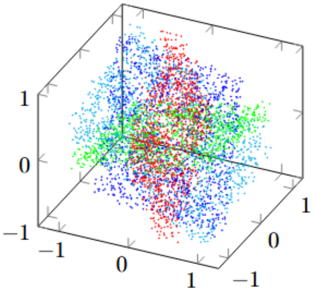
Problem Statement (2)

Subspace Instances of the Cubic Clique Partition Problem

Samples S: points $S \subset \mathbb{R}^3$
Point generation: 3 distinct planes containing the origin, noise σ
Optimal labeling y*: original planes



(a) Samples S

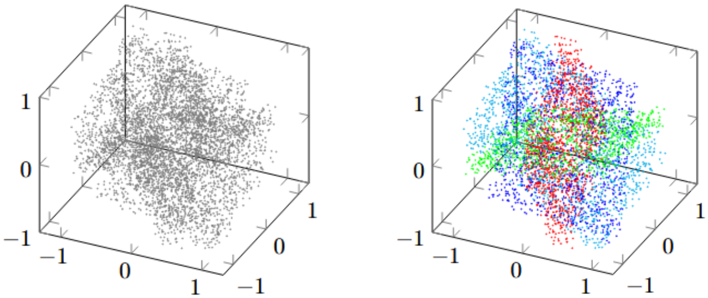


(b) Optimal labeling y*

Problem Statement (2)

Subspace Instances of the Cubic Clique Partition Problem

Samples S: points $S \subset \mathbb{R}^3$
Point generation: 3 distinct planes containing the origin, noise σ
Optimal labeling y^*: original planes
Cost function c? (no concrete plane information given)



(a) Samples S (b) Optimal labeling y^*

Research Goals and Contributions

Related Work:

TODO: $1 + 2$ (citation at the end!)

Research Goals and Contributions

Related Work:
TODO: 1 + 2 (citation at the end!)

Tasks and Solutions:

1. Read [?], implement the partial optimality algorithm
   $\rightarrow$ implementation in C++ (with some adjustments)

Research Goals and Contributions

Related Work:

TODO: 1 + 2 (citation at the end!)

Tasks and Solutions:

1. Read [**?**], implement the partial optimality algorithm
   $\rightarrow$ implementation in C++ (with some adjustments)
2. Construct subspace instances of increasing difficulty
   $\rightarrow$ point generation, appropriate cost function c
   (high accuracy, significant noise tolerance)

Research Goals and Contributions

Related Work:

TODO: 1 + 2 (citation at the end!)

Tasks and Solutions:

1. Read **[?]**, implement the partial optimality algorithm
   → implementation in C++ (with some adjustments)
2. Construct subspace instances of increasing difficulty
   → point generation, appropriate cost function c
   (high accuracy, significant noise tolerance)
3. Apply algorithm to the subspace instances, assess partial optimality, accuracy and computation time
   → experiments and evaluation (prove the quality of c)

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:

$$\min_{y\colon \binom{S}{2} \to \{0,1\}} \sum_{abc \in \binom{S}{3}} c_{abc}\, y_{ab}\, y_{bc}\, y_{ac} + \sum_{ab \in \binom{S}{2}} c_{ab}\, y_{ab} + c_{\emptyset}$$

subject to $y_{ab} + y_{bc} - 1 \leq y_{ac}$ for all distinct $a, b, c \in S$.

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:

$$\min_{y\colon \binom{S}{2} \to \{0,1\}} \sum_{abc \in \binom{S}{3}} c_{abc}\, y_{ab}\, y_{bc}\, y_{ac} + \sum_{ab \in \binom{S}{2}} c_{ab}\, y_{ab} + c_{\emptyset}$$

subject to $y_{ab} + y_{bc} - 1 \leq y_{ac}$ for all distinct $a, b, c \in S$.

Construct **Improving Maps** for the labeling 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:

$$\min_{y\colon \binom{S}{2} \to \{0,1\}} \sum_{abc \in \binom{S}{3}} c_{abc}\, y_{ab}\, y_{bc}\, y_{ac} + \sum_{ab \in \binom{S}{2}} c_{ab}\, y_{ab} + c_{\emptyset}$$

subject to $y_{ab} + y_{bc} - 1 \leq y_{ac}$ for all distinct $a, b, c \in S$.

Construct **Improving Maps** for the labeling 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:

$$\min_{y\colon \binom{S}{2} \to \{0,1\}} \sum_{abc \in \binom{S}{3}} c_{abc}\, y_{ab}\, y_{bc}\, y_{ac} + \sum_{ab \in \binom{S}{2}} c_{ab}\, y_{ab} + c_{\emptyset}$$

subject to $y_{ab} + y_{bc} - 1 \leq y_{ac}$ for all distinct $a, b, c \in S$.

Construct **Improving Maps** for the labeling 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:

$$\min_{y\colon \binom{S}{2} \to \{0,1\}} \sum_{abc \in \binom{S}{3}} c_{abc}\, y_{ab}\, y_{bc}\, y_{ac} + \sum_{ab \in \binom{S}{2}} c_{ab}\, y_{ab} + c_{\emptyset}$$

subject to $y_{ab} + y_{bc} - 1 \leq y_{ac}$ for all distinct $a, b, c \in S$.

Construct **Improving Maps** for the labeling 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:

$$\min_{y\colon \binom{S}{2} \to \{0,1\}} \sum_{abc \in \binom{S}{3}} c_{abc}\, y_{ab}\, y_{bc}\, y_{ac} + \sum_{ab \in \binom{S}{2}} c_{ab}\, y_{ab} + c_{\emptyset}$$

subject to $y_{ab} + y_{bc} - 1 \leq y_{ac}$ for all distinct $a, b, c \in S$.

Construct **Improving Maps** for the labeling 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)

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:

$$\min_{y\colon \binom{S}{2} \to \{0,1\}} \sum_{abc \in \binom{S}{3}} c_{abc}\, y_{ab}\, y_{bc}\, y_{ac} + \sum_{ab \in \binom{S}{2}} c_{ab}\, y_{ab} + c_{\emptyset}$$

subject to $y_{ab} + y_{bc} - 1 \leq y_{ac}$ for all distinct $a, b, c \in S$.

Construct **Improving Maps** for the labeling 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!

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:

$$\min_{y\colon \binom{S}{2} \to \{0,1\}} \sum_{abc \in \binom{S}{3}} c_{abc}\, y_{ab}\, y_{bc}\, y_{ac} + \sum_{ab \in \binom{S}{2}} c_{ab}\, y_{ab} + c_{\emptyset}$$

subject to $y_{ab} + y_{bc} - 1 \leq y_{ac}$ for all distinct $a, b, c \in S$.

Construct **Improving Maps** for the labeling 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: labeling 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 labeling y with some fixed labels

Partial Optimality Algorithm:

Input: labeling 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 labeling 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: labeling 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 labeling 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

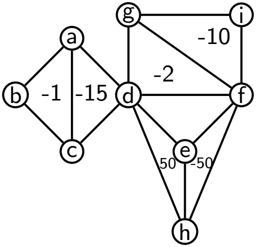
Subproblem-CUT: cut sample subsets $R_1, R_2, \ldots, R_k$ that are only connected via non-negative costs (applied if $k > 1$)

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)

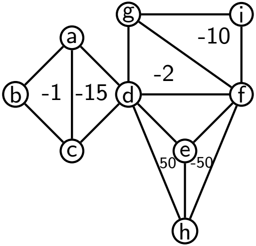


$c_\emptyset = 0$

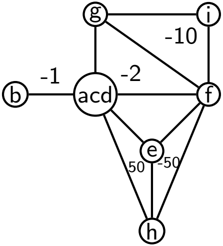
Subproblem-CUT and Subset-JOIN

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)



$c_\emptyset = 0$

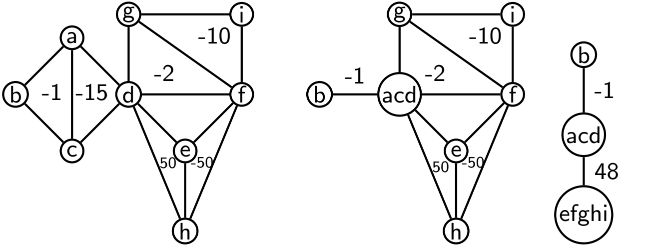


$c_\emptyset = -15$

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)

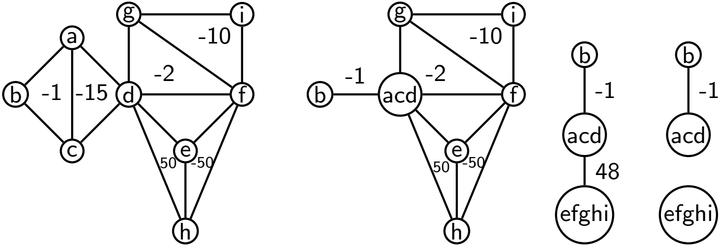


$c_\emptyset = 0$ $c_\emptyset = -15$ $c_\emptyset = -75$

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)

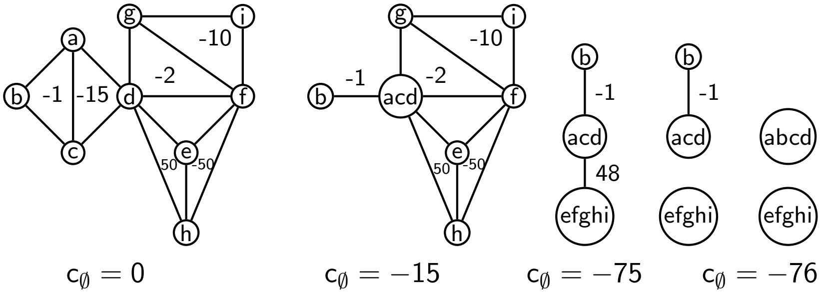


$c_\emptyset = 0$ $c_\emptyset = -15$ $c_\emptyset = -75$

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)



Other JOIN-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 $j \in \overline{R}$ ($\approx$ i-j min-cut)

Other JOIN-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 $j \in \overline{R}$ ($\approx$ i-j min-cut)

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

Other JOIN-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 $j \in \overline{R}$ ($\approx$ i-j min-cut)

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

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

Other JOIN-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 $j \in \overline{R}$ ($\approx$ i-j min-cut)

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

Pair-JOIN-3: join samples i, j if $\mathrm{c}_{ij} \leq$ 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

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 $j \in \overline{R}$ ($\approx$ i-j min-cut)

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

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

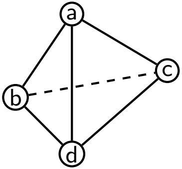
Pair-JOIN-4: join samples i, k if there exists a sample triple ijk such that 7 explicit conditions hold

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

Pyramid Instance and CUT-conditions

$c_{bcd} = 10$

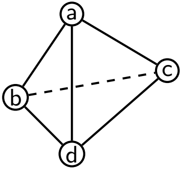
$c_{abc} = c_{abd} = c_{acd} = -50$

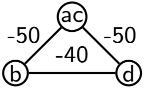


Pyramid Instance and CUT-conditions

$c_{bcd} = 10$

$c_{abc} = c_{abd} = c_{acd} = -50$



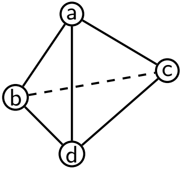


Pair-JOIN-2

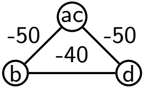
Pyramid Instance and CUT-conditions

$c_{bcd} = 10$

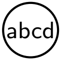
$c_{abc} = c_{abd} = c_{acd} = -50$



Pair-JOIN-2



Subset-JOIN

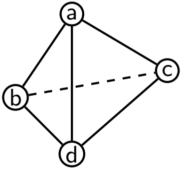


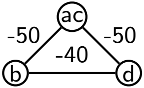
$c_{\emptyset} = -140$

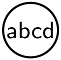
Pyramid Instance and CUT-conditions

$c_{bcd} = 10$

$c_{abc} = c_{abd} = c_{acd} = -50$







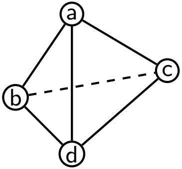
Pair-JOIN-2 Subset-JOIN $c_{\emptyset} = -140$

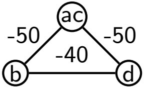
Pair-CUT: cut samples i, j if the direct joing penalty $\geq$ the sum of rewards for joining some subset R with $i \in R$ and $\overline{R}$ with $j \in \overline{R}$ ($\approx$ i-j min-cut)

Pyramid Instance and CUT-conditions

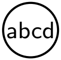
$c_{bcd} = 10$

$c_{abc} = c_{abd} = c_{acd} = -50$





Pair-JOIN-2



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

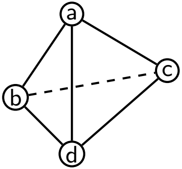
Pair-CUT: cut samples i, j if the direct joing penalty $\geq$ the sum of rewards for joining some subset R with $i \in R$ and $\overline{R}$ with $j \in \overline{R}$ ($\approx$ i-j min-cut)

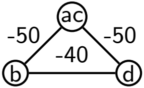
Triple-CUT: cut samples i, j, k if the condition holds (similar to Pair-CUT) ($\approx$ i-jk min-cut)

Pyramid Instance and CUT-conditions

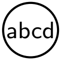
$c_{bcd} = 10$

$c_{abc} = c_{abd} = c_{acd} = -50$





Pair-JOIN-2



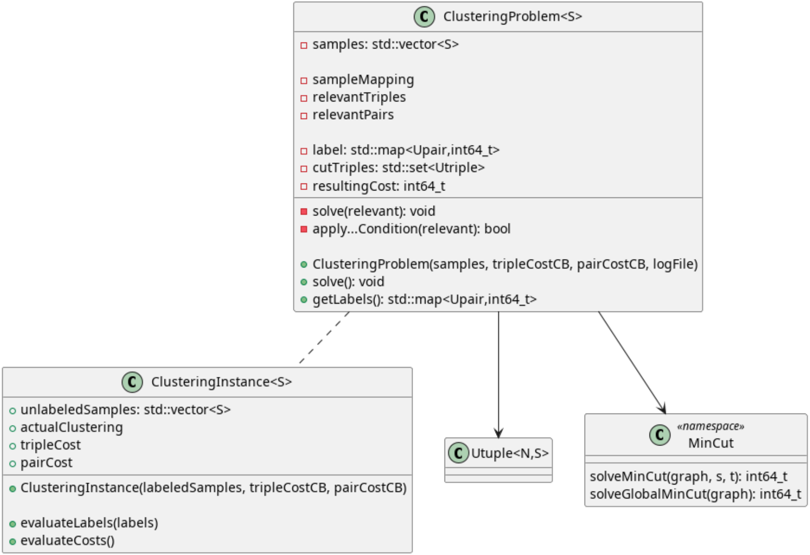
Subset-JOIN $c_{\emptyset} = -140$

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 $j \in \overline{R}$ ($\approx$ i-j min-cut)

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

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

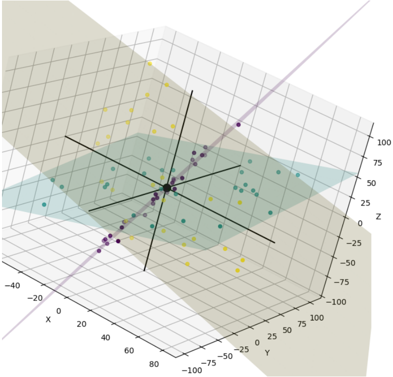
Program Structure



Plane and Point Generation

Plane Generation:

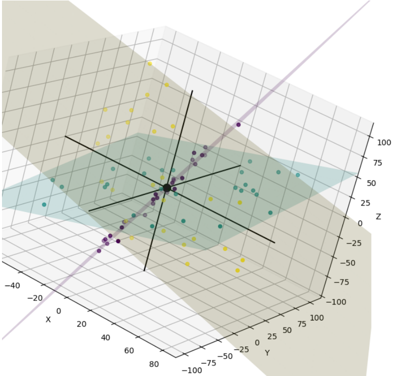
- generate 3 planes
  as distinct normal vectors
  $\vec{n}_1, \vec{n}_2, \vec{n}_3$ (normalized)



Plane and Point Generation

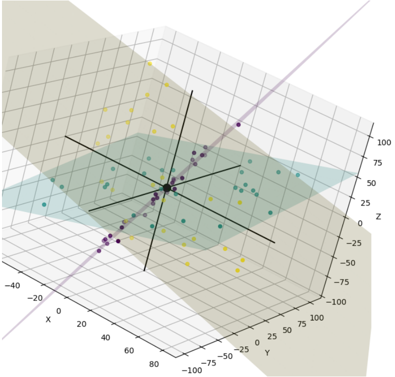
Plane Generation:

- generate 3 planes as distinct normal vectors $\vec{n}_1, \vec{n}_2, \vec{n}_3$ (normalized)
- compute a $\vec{r}_{i,1}$ (normalized) orthogonal to $\vec{n}_i$ $(i \in \{1, 2, 3\})$



Plane Generation:

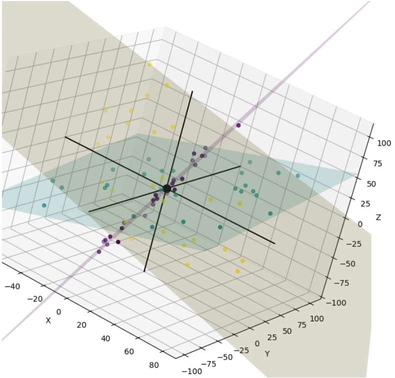
- generate 3 planes as distinct normal vectors $\vec{n}_1, \vec{n}_2, \vec{n}_3$ (normalized)
- compute a $\vec{r}_{i,1}$ (normalized) orthogonal to $\vec{n}_i$ ($i \in \{1, 2, 3\}$)
- compute the $\vec{r}_{i,2}$ (normalized) orthogonal to $\vec{n}_i$ and $\vec{r}_{i,1}$



Plane and Point Generation

Plane Generation:

- generate 3 planes as distinct normal vectors $\vec{n}_1, \vec{n}_2, \vec{n}_3$ (normalized)
- compute a $\vec{r}_{i,1}$ (normalized) orthogonal to $\vec{n}_i$ ($i \in \{1, 2, 3\}$)
- compute the $\vec{r}_{i,2}$ (normalized) orthogonal to $\vec{n}_i$ and $\vec{r}_{i,1}$



Point Generation on the plane $(\vec{n}, \vec{r}_1, \vec{r}_2)$, parameters (D, σ):

- random variables $k_1, k_2 \in [-D, D]$ (uniform distribution)
- random variable k_n (normal distribution based on σ)
- generate point $p = k_1\vec{r}_1 + k_2\vec{r}_2 + k_n\vec{n}$

Cost Function

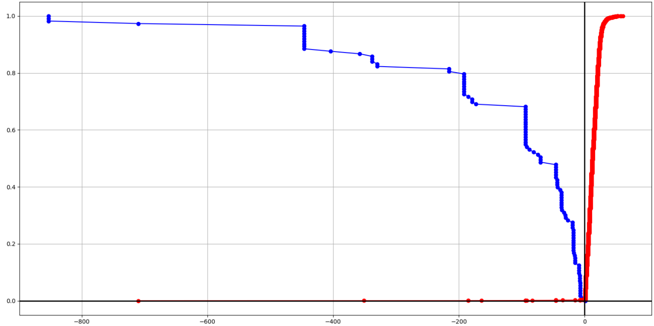
Triangle $abc \in \binom{S}{3}$

1. Smallest side $s < D/2 \rightarrow c_{abc} = 0$
2. Largest angle $\alpha > 150^\circ \rightarrow c_{abc} = 0$
3. ha, hb, hc: distances to the best fitting plane
   $ha + hb + hc > 3\sigma + 10^{-6}$
   $\rightarrow c_{abc} = \frac{(ha+hb+hc)-(3\sigma+10^{-6})}{3D}$
4. ho: distance from the origin to the triangle plane
   $ho > \frac{10\sigma}{\#points} + 10^{-6} \rightarrow c_{abc} = 0$
5. for all points p: hp: distance to the best fitting plane
   choose p if $hp < \sigma + 10^{-6}$ and $|\vec{p}| > 0.3D$
   hp': distance to the best fitting plane of all chosen points
   $\delta_p = \frac{hp' - (\sigma+10^{-6})}{D}$, $SAME = \{p\colon \delta_p < 0\}$, $rew = \sum_{p \in SAME} \delta_p$,
   $|SAME| \leq 3 \rightarrow c_{abc} = 0$
   else $\rightarrow c_{abc} = 2^{|SAME|-4} rew$

Cost Function Evaluation (3x15 points, $\sigma = 1$)

Blue: same plane

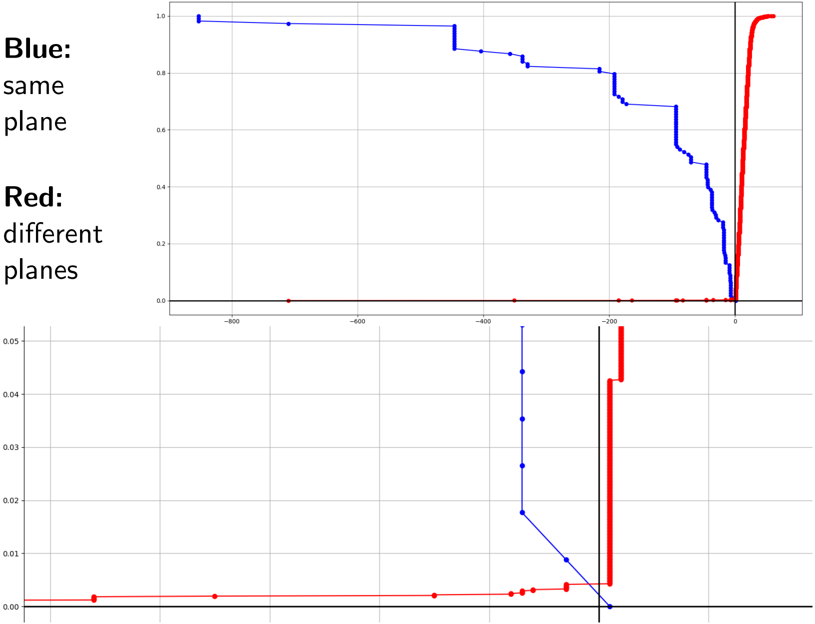
Red: different planes



Cost Function Evaluation (3x15 points, $\sigma = 1$)

Blue: same plane

Red: different planes



Experiments

- WSL2 Ubuntu, Intel Core i7-11370H (3.30 GHz), 16 GB RAM

Experiments

- WSL2 Ubuntu, Intel Core i7-11370H (3.30 GHz), 16 GB RAM
- Apply the algorithm to the random cubic subspace instances with $D = 100$ and fixed $\sigma = 0, 1, 2, 3, 4, 5$:
  - 3x7 points (solve 15 instances)
  - 3x10 points (solve 15 instances)
  - 3x15 points (solve 7 instances)
  - 3x20 points (solve 1 instance)

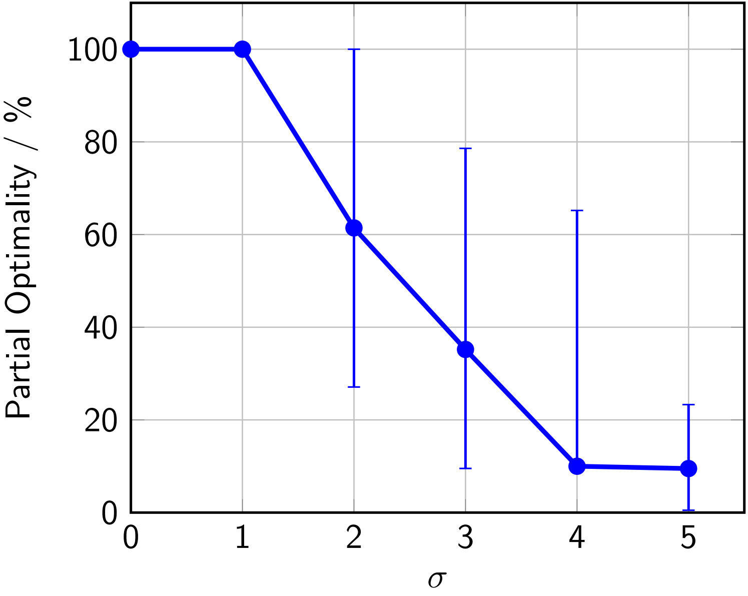
Experiments

- WSL2 Ubuntu, Intel Core i7-11370H (3.30 GHz), 16 GB RAM
- Apply the algorithm to the random cubic subspace instances with $D = 100$ and fixed $\sigma = 0, 1, 2, 3, 4, 5$:
  - 3x7 points (solve 15 instances)
  - 3x10 points (solve 15 instances)
  - 3x15 points (solve 7 instances)
  - 3x20 points (solve 1 instance)
- Track
  - computation time (s)
  - partial optimality (%)
  - accuracy (%) with respect to the truth (correct labeling)

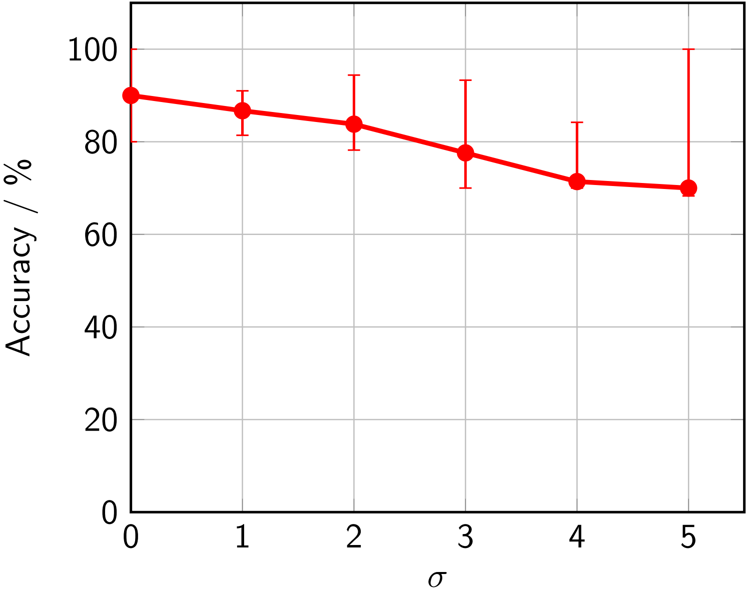
Experiments

- WSL2 Ubuntu, Intel Core i7-11370H (3.30 GHz), 16 GB RAM
- Apply the algorithm to the random cubic subspace instances with $D = 100$ and fixed $\sigma = 0, 1, 2, 3, 4, 5$:
  - 3x7 points (solve 15 instances)
  - 3x10 points (solve 15 instances)
  - 3x15 points (solve 7 instances)
  - 3x20 points (solve 1 instance)
- Track
  - computation time (s)
  - partial optimality (%)
  - accuracy (%) with respect to the truth (correct labeling)
- Capture
  - 1.quartile (Q1)
  - median (Q2)
  - 3.quartile(Q3)
  - the worst computation time

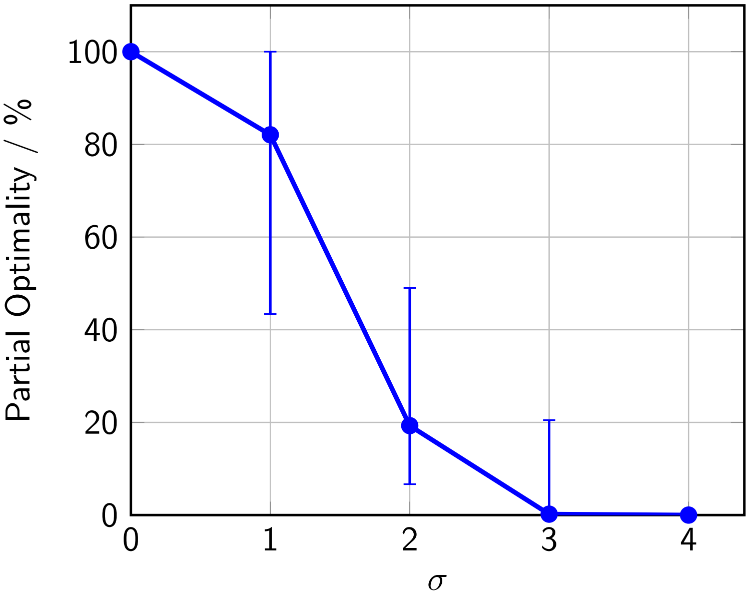
Partial Optimality (3x7 points)



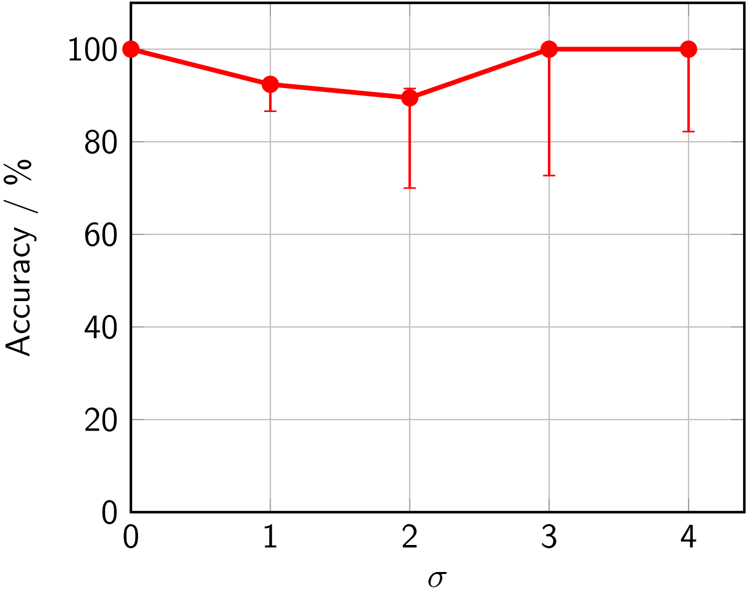
Accuracy (3x7 points)



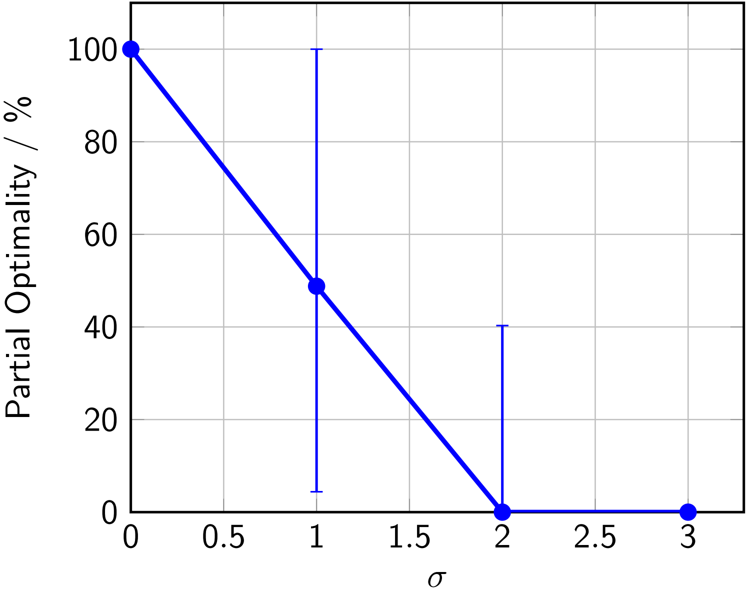
Partial Optimality (3x10 points)



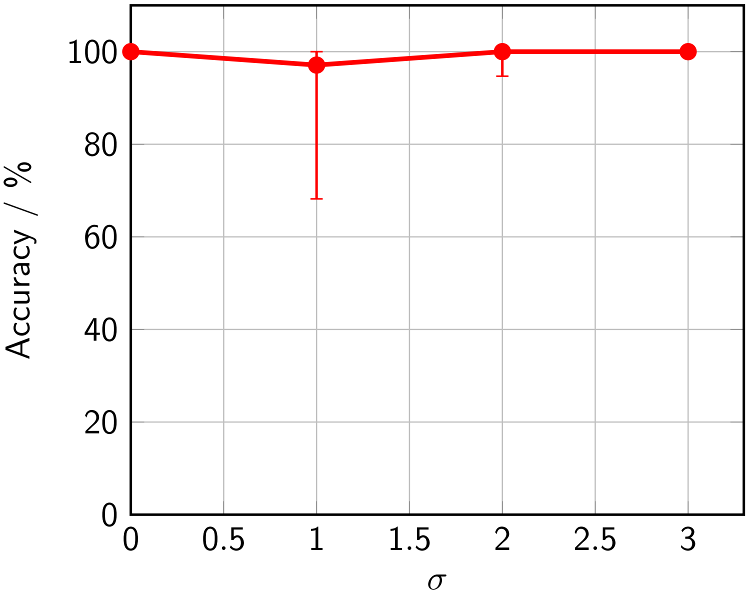
Accuracy (3x10 points)



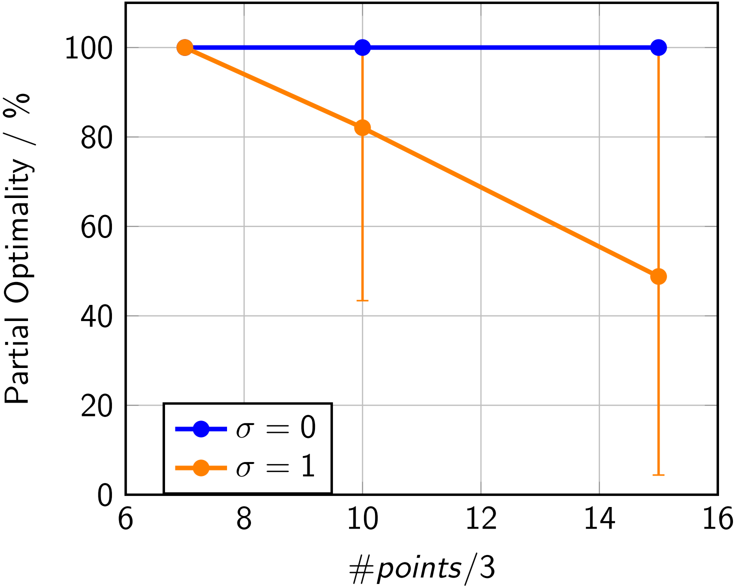
Partial Optimality (3x15 points)



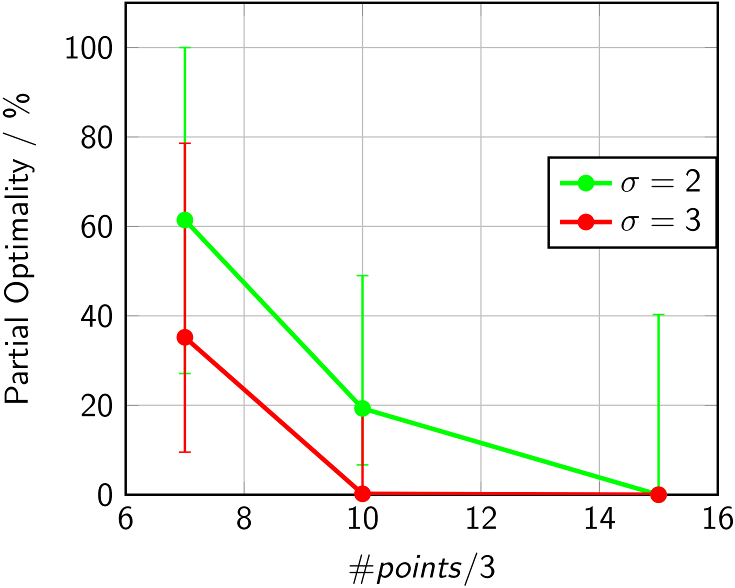
Accuracy (3x15 points)

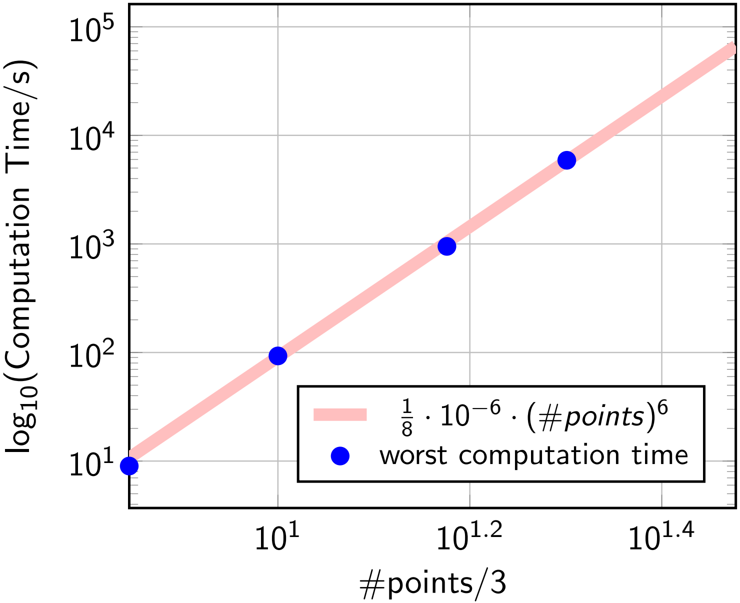


Partial Optimality



Partial Optimality





10^5
10^4
10^3
10^2
10^1
$\log_{10}$(Computation Time/s)
10^1
$10^{1.2}$
$10^{1.4}$
#points/3
$\frac{1}{8} \cdot 10^{-6} \cdot (\#points)^6$
worst computation time

Conclusion

1. Implementation of the partial optimality algorithm:
   - arbitrary sample type
   - sparse cost representation
   - pair labeling and triple cuts
   - reasonable adjustments of the partial optimality conditions
   - self-explaining logs

Conclusion

1. Implementation of the partial optimality algorithm:
   - arbitrary sample type
   - sparse cost representation
   - pair labeling and triple cuts
   - reasonable adjustments of the partial optimality conditions
   - self-explaining logs
2. Subspace instance generation using linear algebra methods, geometric cost function c:
   - high accuracy (over 75%)
   - significant noise tolerance ($\sigma \geq 1$)
   - $O(k \cdot n^6)$ for $n = \#$points and a small k

Conclusion

1. Implementation of the partial optimality algorithm:
   - arbitrary sample type
   - sparse cost representation
   - pair labeling and triple cuts
   - reasonable adjustments of the partial optimality conditions
   - self-explaining logs
2. Subspace instance generation using linear algebra methods, geometric cost function c:
   - high accuracy (over 75%)
   - significant noise tolerance ($\sigma \geq 1$)
   - $O(k \cdot n^6)$ for $n = \#$points and a small k

Future Work:

- optimize and parallelize the algorithm
- overcome the partial optimality loss for c
- determine better parameters for c
- update c with advanced criteria

References

My program, scripts and presentation:

`https://github.com/Vovsanka/ResearchProjectML`

TODO: citation