Optimisation in Multiple View Geometry: The L-infinity Way (Part 2)

Tat-Jun Chin

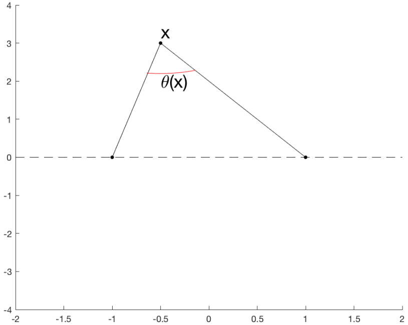
The University of Adelaide

CVPR 2018 Tutorial

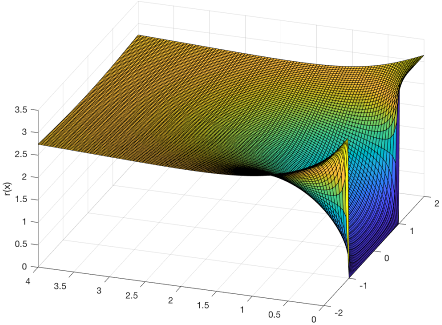
Table of Contents

Table of Contents

Recap: Quasiconvexity



$r(\mathbf{x}) = \pi - \theta(\mathbf{x})$
$\mathbf{x} \in$ upper halfplane



Animation of sublevel sets.

Recap: Quasiconvexity

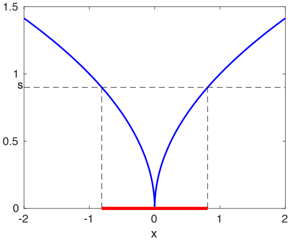
- Let $\{r_i\}_{i=1}^N$ be a set of quasiconvex functions, where each

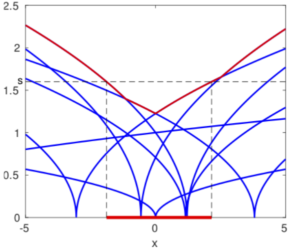
$$r_i : \mathcal{D} \mapsto \mathbb{R}_{\geq 0}$$

  and $\mathcal{D}$ is a *convex domain*.
- Then, the point-wise maximum function

$$Q(\mathbf{x}) = \max_i \; r_i(\mathbf{x}) = \left\| \begin{bmatrix} r_1(\mathbf{x}) \\ \vdots \\ r_N(\mathbf{x}) \end{bmatrix} \right\|_\infty$$

  is also quasiconvex for $\mathbf{x} \in \mathcal{D}$.

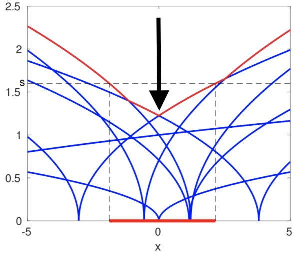




Quasiconvex programming

Minimising the point-wise maximum of a set of quasiconvex functions $\{r_i\}_{i=1}^N$ defined over a convex domain $\mathcal{D}$, i.e.,

$$\min_{\mathbf{x}\in\mathcal{D}} \; \max_i \; r_i(\mathbf{x})$$



D. Eppstein. Quasiconvex programming. Combinatorial and Computational Geometry 25 (2005).

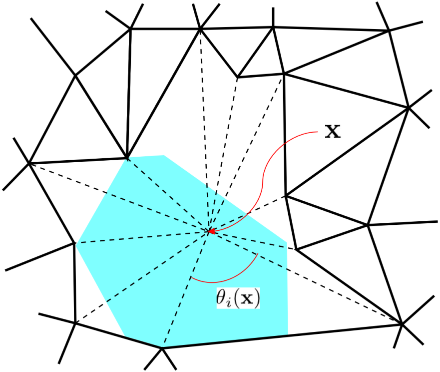
Example: Mesh smoothing

- Function $r_i(\mathbf{x}) = \pi - \theta_i(\mathbf{x})$ decreases with $\theta_i(\mathbf{x})$.
- Solving

$$\min_{\mathbf{x}\in\mathcal{D}} \;\max_{i}\; r_i(\mathbf{x})$$

finds mesh vertex $\mathbf{x}$ that is as close as possible to all sides.

- Solution can only lie in $\mathcal{D}$ (the convex region shaded in blue).



Amenta et al. Optimal point placement for mesh smoothing. Journal of Algorithms 30.2 (1999), pp. 302–322.

Example: Multi-view triangulation (L-infinity)

- Given camera matrix $\mathbf{P}$ and image point $\mathbf{u}$,

$$r(\mathbf{x}) = \left\| \frac{\mathbf{P}^{(1:2)}\tilde{\mathbf{x}}}{\mathbf{P}^{(3)}\tilde{\mathbf{x}}} - \mathbf{u} \right\|_2 = \frac{\left\| \left[\mathbf{P}^{(1:2)} - \mathbf{u}\mathbf{P}^{(3)} \right] \tilde{\mathbf{x}} \right\|_2}{\mathbf{P}^{(3)}\tilde{\mathbf{x}}}$$

  gives the *reprojection error* of 3D point $\mathbf{x} \in \mathbb{R}^3$. If $\mathbf{P}^{(3)}\tilde{\mathbf{x}} \geq 0$, then $\mathbf{x}$ lies in front of the camera.

- Given N 2D points $\{\mathbf{u}_i\}_{i=1}^N$ from N views $\{\mathbf{P}_i\}_{i=1}^N$, solving

$$\min_{\mathbf{x} \in \mathcal{D}} \quad \max_i \quad r_i(\mathbf{x})$$

  finds $\mathbf{x}$ that minimises the largest reprojection error. Here

$$\mathcal{D} = \left\{ \mathbf{x} \in \mathbb{R}^3 \mid \mathbf{P}_i^{(3)}\tilde{\mathbf{x}} \geq 0 \right\}.$$

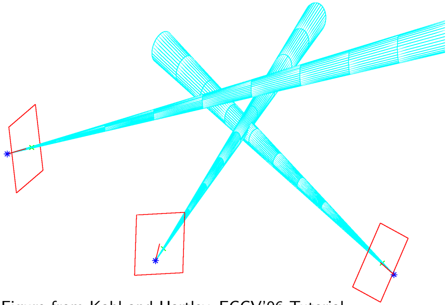
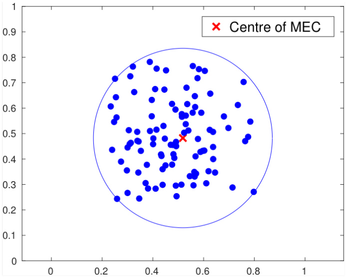


Figure from Kahl and Hartley, ECCV'06 Tutorial.

Example: Minimum enclosing circle (MEC)

- Given a set of points $\{\mathbf{p}_i\}_{i=1}^{N}$ on the plane, find the smallest circle that encloses all the points:

$$\min_{\mathbf{x}\in\mathbb{R}^2} \; \max_{i} \; \|\mathbf{x} - \mathbf{p}_i\|_2$$



Example: Chebyshev approximation

- Given a set of points $\{(\mathbf{a}_i, b_i)\}_{i=1}^{N}$ in $(d+1)$ dimensions, find the hyperplane $\mathbf{x} \in \mathbb{R}^d$ that minimises the largest residual

$$\min_{\mathbf{x}\in\mathbb{R}^d} \max_i |\mathbf{a}_i^T\mathbf{x} - b_i|$$

Special case: Minimum enclosing slab (MES)

- Setting $\mathbf{a}_i = [p_i \; 1]^T$ and $b_i = q_i$, where (p_i, q_i) are points on the plane, performing Chebyshev approximation amounts to finding the *thinnest slab* that encloses all the points.
- Width of slab is measured vertically (i.e., along the q-axis).

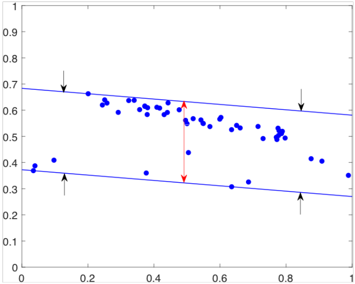


Table of Contents

Linear programming (LP)-type problems

- An LP-type problem consists of
  - A set of "constraints" $\mathcal{S}$;
  - An objective function f that measures the cost of any subset of $\mathcal{S}$.
- The goal is to compute $f(\mathcal{S})$.
- Two main properties:

Monotonicity

For any two subsets $\mathcal{A}$ and $\mathcal{B}$ of $\mathcal{S}$, if $\mathcal{A} \subset \mathcal{B}$ then $f(\mathcal{A}) \leq f(\mathcal{B})$.

Locality

For any $\mathcal{A} \subseteq \mathcal{S}$, $p \in \mathcal{S}$ and $q \in \mathcal{S}$, if $f(\mathcal{A}) = f(\mathcal{A} \cup \{p\}) = f(\mathcal{A} \cup \{q\})$, then $f(\mathcal{A}) = f(\mathcal{A} \cup \{p, q\})$.

D. Eppstein. Quasiconvex programming. Combinatorial and Computational Geometry 25 (2005).

Examples: MEC and MES

Minimum enclosing circle (MEC):

- $\mathcal{S} = \{\mathbf{p}_i\}_{i=1}^{N}$ and $f(\mathcal{S})$ gives the radius of MEC($\mathcal{S}$).

Monotonicity

If $\mathcal{A} \subset \mathcal{B} \subseteq \mathcal{S}$, then MEC($\mathcal{A}$) cannot be larger than MEC($\mathcal{B}$).

Locality

If $f(\mathcal{A}) = f(\mathcal{A} \cup \{\mathbf{p}_i\})$, then point $\mathbf{p}_i$ lies in MEC($\mathcal{A}$).
If $f(\mathcal{A}) = f(\mathcal{A} \cup \{\mathbf{q}_i\})$, then point $\mathbf{q}_i$ also lies in MEC($\mathcal{A}$).
Clearly $\{\mathbf{p}_i, \mathbf{q}_i\}$ also lie in MEC($\mathcal{A}$), hence $f(\mathcal{A}) = f(\mathcal{A} \cup \{\mathbf{p}_i, \mathbf{q}_i\})$.

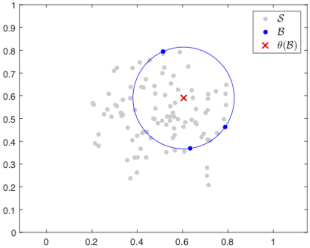
Minimum enclosing slab (MES):

- $\mathcal{S} = \{(\mathbf{a}_i, b_i)\}_{i=1}^{N}$ and $f(\mathcal{S})$ gives the width of MES($\mathcal{S}$).
- Replace MEC with MES in the arguments above.

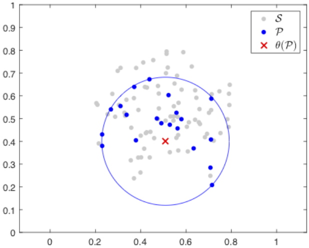
Basis

A basis of an LP-type problem $(\mathcal{S}, f)$ is a subset $\mathcal{B} \subseteq \mathcal{S}$ such that $f(\mathcal{A}) < f(\mathcal{B})$ for every $A \subset \mathcal{B}$.

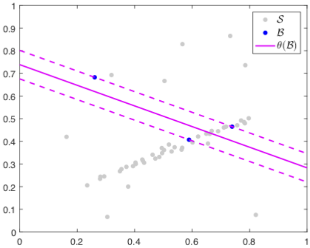
Examples: MEC and MES



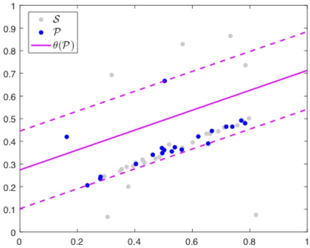
(a) $\mathcal{B}$ is a basis.



(b) $\mathcal{P}$ is not a basis.



(a) $\mathcal{B}$ is a basis.



(b) $\mathcal{P}$ is not a basis.

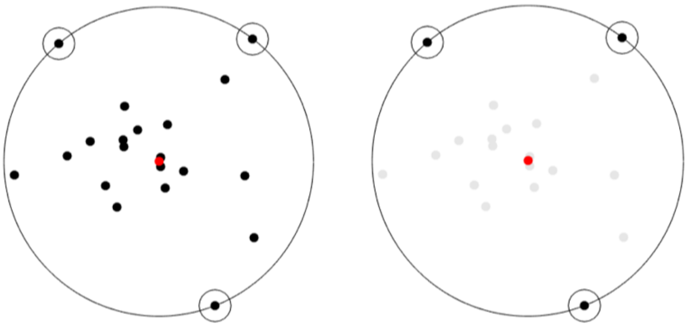
Support set

The support set $\mathcal{K}$ of an LP-type problem $(\mathcal{S}, f)$ is a basis of the problem such that $f(\mathcal{S}) = f(\mathcal{K})$.

Further properties:

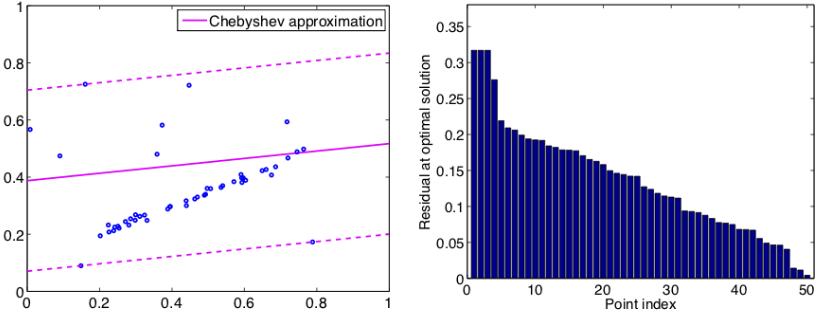
- Solving an LP-type problem amounts to finding its support set.
- The cost/residual of all items in $\mathcal{K}$ are equal w.r.t. the estimate.
- Items not in the support set (the subset $\mathcal{S} \setminus \mathcal{K}$) can be ignored.

Example: MEC



Example: MES

Points in the support set $\mathcal{K}$ have equal (maximum) distance to the Chebyshev fit.



Combinatorial dimension

The upper bound on the size of a basis in an LP-type problem $(\mathcal{S}, f)$.

- Any quasiconvex program forms an LP-type problem of combinatorial dimension is $2d+1$.
- If each component r_i is *continuously shrinking*, then the combinatorial dimension is $d+1$.
- Sim and Harley showed that functions of the form

$$r(\mathbf{x}) = \frac{\|\mathbf{A}\mathbf{x} + \mathbf{b}\|_2}{\mathbf{p}^T\mathbf{x} + q}$$

are continuously shrinking for $\mathbf{p}^T\mathbf{x} + q > 0$. Call problems of this form quasiconvex geometric problems.

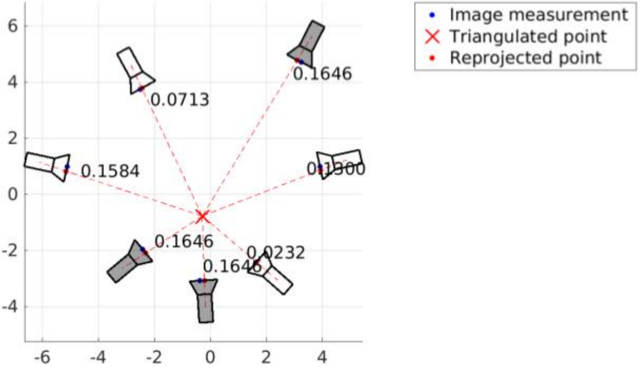
Amenta et al. Optimal point placement for mesh smoothing. Journal of Algorithms 30.2 (1999), pp. 302–322.

K. Sim and R. Hartley. Removing outliers using the l-infinity norm. CVPR 2006.

Example: L-infinity triangulation

$\mathcal{S} = \{(\mathbf{P}_i, \mathbf{u}_i)\}_{i=1}^{N}$ is a set of N camera matrices and image points, and

$$f(\mathcal{S}) = \min_{\mathbf{x} \in \mathcal{D}} \ \max_{i \in \mathcal{S}} \ \frac{\left\| \left[\mathbf{P}_i^{(1:2)} - \mathbf{u}_i \mathbf{P}_i^{(3)} \right] \tilde{\mathbf{x}} \right\|_2}{\mathbf{P}_i^{(3)} \tilde{\mathbf{x}}}$$



Solving LP-type problems

function solve_lptype($\mathcal{S}$,f,$\mathcal{C}$):

---

if $\mathcal{S} = \mathcal{C}$ **then**
    return $\mathcal{C}$.
end if
$c \leftarrow$ A random item from $\mathcal{S} \setminus \mathcal{C}$.
$\mathcal{B} \leftarrow$ solve_lptype($\mathcal{S} \setminus \{c\}$, f, $\mathcal{C}$).
if $f(\mathcal{B}) \neq f(\mathcal{B} \cup \{c\})$ **then** // Core 1: violation test
    $\mathcal{B} \leftarrow$ basis($\mathcal{B} \cup \{c\}$). // Core 2: basis updating
    $\mathcal{B} \leftarrow$ solve_lptype($\mathcal{S}$,f,$\mathcal{B}$).
end if
return $\mathcal{B}$

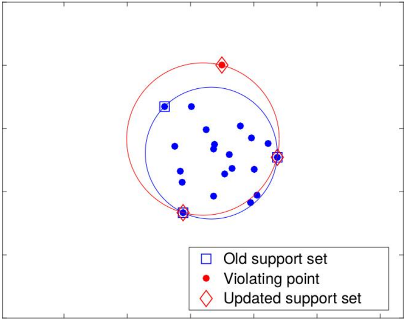
Matoušek et al. A subexponential bound for linear programming. Algorithmica 16 (1996), pp. 498–516.

Solving LP-type problems

function solve_lptype($\mathcal{S}$,f,$\mathcal{C}$):

---

if $\mathcal{S} = \mathcal{C}$ **then**
    return $\mathcal{C}$.
end if
$c \leftarrow$ A random item from $\mathcal{S} \setminus \mathcal{C}$.
$\mathcal{B} \leftarrow$ solve_lptype($\mathcal{S} \setminus \{c\}$, f, $\mathcal{C}$).
if $f(\mathcal{B}) \neq f(\mathcal{B} \cup \{c\})$ **then**
    $\mathcal{B} \leftarrow$ basis($\mathcal{B} \cup \{c\}$).
    $\mathcal{B} \leftarrow$ solve_lptype($\mathcal{S}$,f,$\mathcal{B}$).
end if
return $\mathcal{B}$



H. Li. Efficient reduction of L-infinity geometry problems. CVPR 2009.

Table of Contents

Another algorithm

- Keep expanding $\mathcal{C}$ and updating $\hat{\mathbf{x}}_{\mathcal{C}}$ using the point with the largest residual until all points are covered.

function solve_lptype_mv($\mathcal{S}$,f):

$\mathcal{C} \leftarrow$ Randomly select one item from $\mathcal{S}$.
while true **do**
  $\hat{\mathbf{x}}_{\mathcal{C}} \leftarrow \arg\min_{\mathbf{x}\in\mathcal{D}} \max_{i\in\mathcal{C}} r_i(\mathbf{x})$. // Solve on current coreset $\mathcal{C}$.
  $i^* \leftarrow \arg\max_{i\in\mathcal{S}} r_i(\hat{\mathbf{x}}_{\mathcal{C}})$. // Find most violating point.
  if $r_{i^*}(\hat{\mathbf{x}}_{\mathcal{C}}) \leq f(\mathcal{C})$ **then**
    Break. // Most violating point already covered; done.
  end if
  $\mathcal{C} \leftarrow \mathcal{C} \cup \{i^*\}$. // Insert one point.
end while
return $\hat{\mathbf{x}}_{\mathcal{C}}$

Clarkson. Coresets, sparse greedy approximation, and the Frank-Wolfe algorithm. SODA 2008.

Seo and Hartley. A Fast Method to Minimize L Error Norm for Geometric Vision Problems. ICCV 2007.

Another algorithm

- Keep expanding $\mathcal{C}$ and updating $\hat{\mathbf{x}}_\mathcal{C}$ using the point with the largest residual until all points are covered.

function solve_lptype_mv($\mathcal{S}$,f):

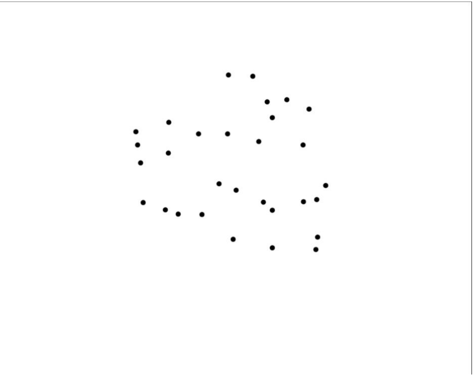
$\mathcal{C} \leftarrow$ Randomly select one item from $\mathcal{S}$.
while true **do**
  $\hat{\mathbf{x}}_\mathcal{C} \leftarrow \arg\min_{\mathbf{x}\in\mathcal{D}} \; \max_{i\in\mathcal{C}} \; r_i(\mathbf{x})$. // Warm start using prev. $\hat{\mathbf{x}}_\mathcal{C}$.
  $i^* \leftarrow \arg\max_{i\in\mathcal{S}} \; r_i(\hat{\mathbf{x}}_\mathcal{C})$. // Find most violating point.
  if $r_{i^*}(\hat{\mathbf{x}}_\mathcal{C}) \leq f(\mathcal{C})$ **then**
    Break. // Most violating point already covered; done.
  end if
  $\mathcal{C} \leftarrow \mathcal{C} \cup \{i^*\}$. // Insert one point.
end while
return $\hat{\mathbf{x}}_\mathcal{C}$

Clarkson. Coresets, sparse greedy approximation, and the Frank-Wolfe algorithm. SODA 2008.

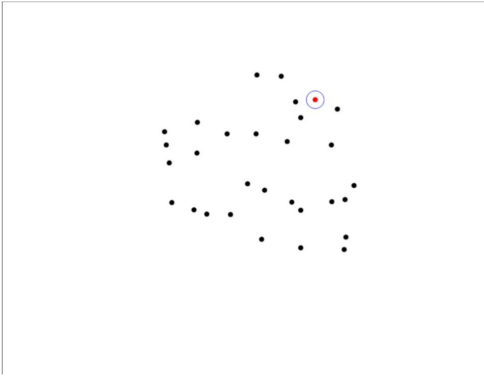
Seo and Hartley. A Fast Method to Minimize L Error Norm for Geometric Vision Problems. ICCV 2007.

Example: MEC

Input data $\mathcal{S}$:

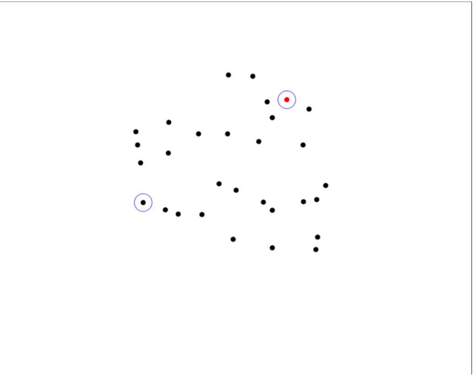


Initial $\mathcal{C}$:

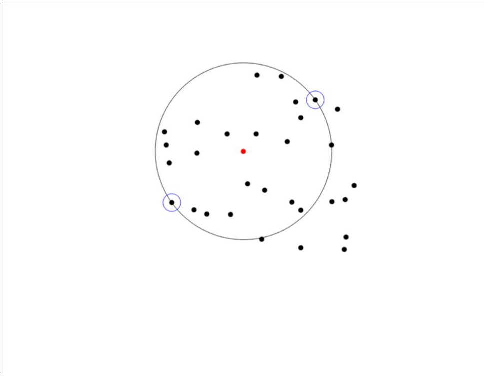


Example: MEC

Found most violating point:

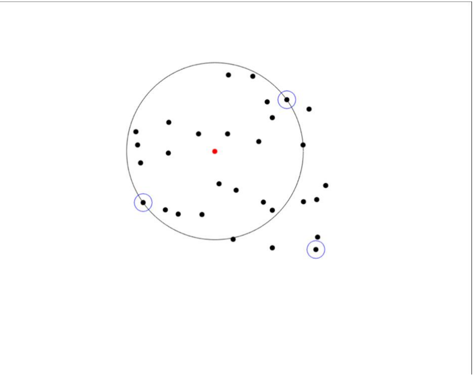


Insert into $\mathcal{C}$ and update MEC:

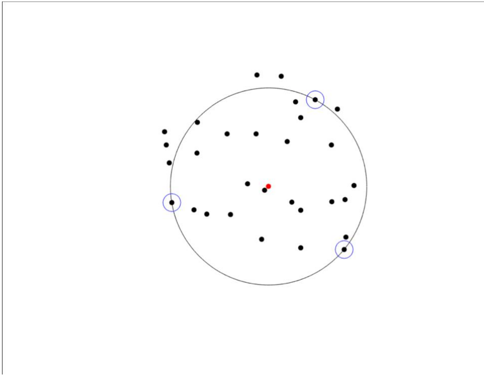


Example: MEC

Found most violating point:

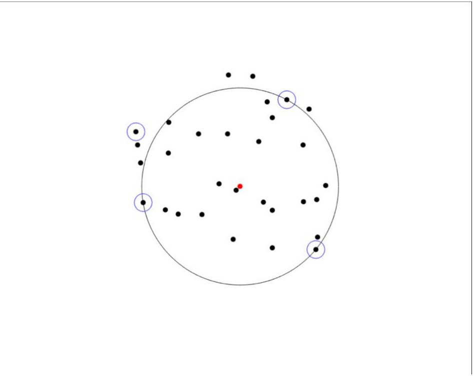


Insert into $\mathcal{C}$ and update MEC:

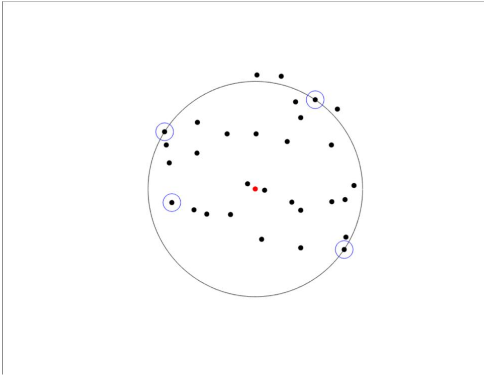


Example: MEC

Found most violating point:

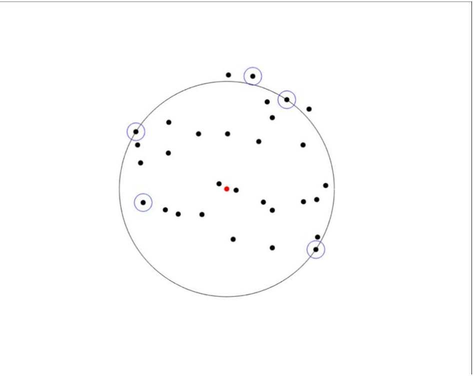


Insert into $\mathcal{C}$ and update MEC:

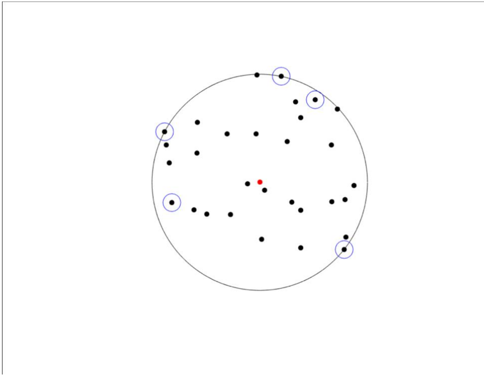


Example: MEC

Found most violating point:

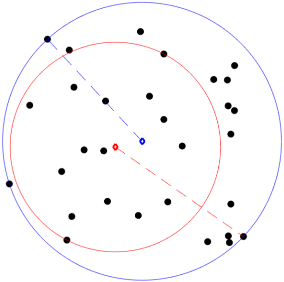


Insert into $\mathcal{C}$ and update MEC:



No more violating points—done.

Coresets

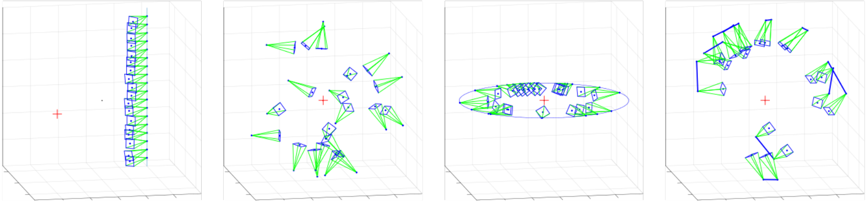


Red circle = MEC on current $\mathcal{C}$.
Blue circle = MEC on $\mathcal{S}$.

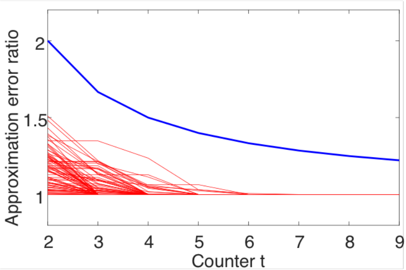
- $\text{error}_{\mathcal{C}} := \max_{i \in \mathcal{S}} \; r_i(\hat{\mathbf{x}}_{\mathcal{C}})$.
- $\text{error}_{\mathcal{S}} := f(\mathcal{S}) = \max_{i \in \mathcal{S}} \; r_i(\hat{\mathbf{x}}_{\mathcal{S}})$.
- By construction $\text{error}_{\mathcal{C}} \geq \text{error}_{\mathcal{S}}$, but we can also guarantee that

$$\frac{\text{error}_{\mathcal{C}}}{\text{error}_{\mathcal{S}}} \leq (1 + \epsilon), \qquad \text{where} \quad \epsilon = \frac{2}{|\mathcal{S}|}$$

Coresets



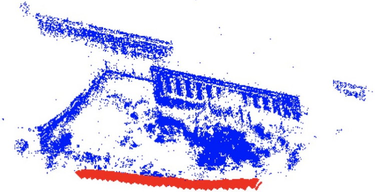
(a) Type A (b) Type B (c) Type C (d) Type D

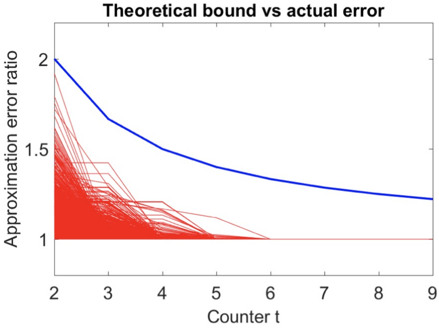


Blue = Theoretical bound
Red = Actual error ratios

Q. Zhang and T.-J. Chin. Coresets for triangulation. IEEE TPAMI (2017). doi: 10.1109/TPAMI.2017.2750672.

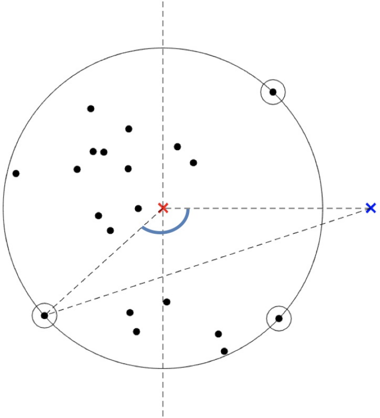
Coresets

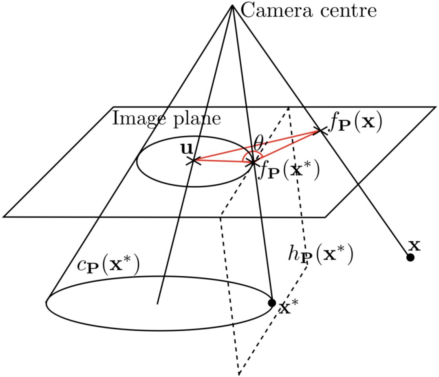




Q. Zhang and T.-J. Chin. Coresets for triangulation. IEEE TPAMI (2017). doi: 10.1109/TPAMI.2017.2750672.

Coresets



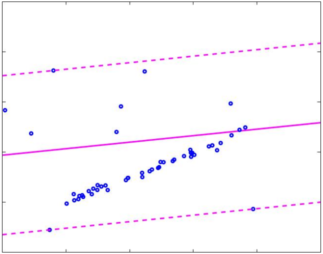


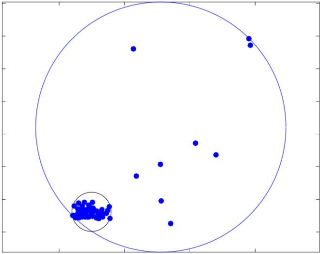
Q. Zhang and T.-J. Chin. Coresets for triangulation. IEEE TPAMI (2017). doi: 10.1109/TPAMI.2017.2750672.

Table of Contents

Sensitivity to outliers

- Minimising the largest residual amounts to fitting on the outliers. Hence, L-infinity estimation is not robust.





- Nonetheless, the L-infinity approach provides a good framework to analyse and develop robust estimation algorithms.

Robust estimation

- Let $(\mathcal{S}, f)$ be an LP-type problem with quasiconvex geometric residuals $\{r_i\}_{i=1}^N$, where some of the residuals correspond to outliers.
- Solve for $\mathbf{x}$ by ignoring residuals that are greater than ϵ:

$$\begin{aligned} \max_{\mathbf{x}\in\mathcal{D},\ \mathcal{I}\subseteq\mathcal{S}} \quad & |\mathcal{I}| \\ \text{s.t.} \quad & r_i(\mathbf{x}) \leq \epsilon,\ \ i \in \mathcal{I}. \end{aligned} \qquad \text{(Consensus maximisation)}$$

- Let $\mathcal{I}^*$ be the maximum consensus set. Observe that

$$\min_{\mathbf{x}\in\mathcal{D}} \ \max_{i\in\mathcal{I}^*} \ r_i(\mathbf{x})$$

is the LP-type problem $(\mathcal{I}^*, f)$, and by construction

$$f(\mathcal{I}^*) \leq \epsilon.$$

Simple algorithm (enumeration)

- Let $\mathcal{K}$ be the support set of $(\mathcal{I}^*, f)$. Since $\mathcal{I}^* \subseteq \mathcal{S}$, $\mathcal{K}$ is also a basis of $(\mathcal{S}, f)$. $\implies$ Find $\mathcal{K}$ by enumerating all bases of $(\mathcal{S}, f)$.

Require: LP-type problem $(\mathcal{S}, f)$, inlier threshold ϵ.

$\psi \leftarrow 0$, $\mathcal{K} \leftarrow \emptyset$.
for all $(d+1)$-subsets $\mathcal{B}$ of $\mathcal{S}$ **do**
    if $f(\mathcal{B}) \leq \epsilon$ **then**
        if coverage($\mathcal{B}$)$> \psi$ **then**
            $\psi \leftarrow$ coverage($\mathcal{B}$). // Needs a bit of explanation.
            $\mathcal{K} \leftarrow \mathcal{B}$.
        end if
    end if
end for
return $\mathcal{K}$.

- Number of iterations $= \binom{N}{d+1}$. Including checking coverage, runtime is $\mathcal{O}(N^{d+2}T(d))$, where $T(d)$ is time to solve $f(\mathcal{B})$.

Olsson et al. A polynomial-time bound for matching and registration with outliers. CVPR 2008.
Enqvist et al. Robust fitting for multiple view geometry. ECCV 2012.

Computational hardness

NP-hardness

Consensus maximisation is NP-hard.

Proof: Turing reduction from Least Median Squares.

$\implies$ There are no polynomial time algorithms.

XP (slice-wise polynomial)

Consensus maximisation is XP in the dimension d.

Proof: Use enumeration method to get runtime of $\mathcal{O}(N^{d+2}T(d))$.

$\implies$ For a fixed d, can get polynomial time in N.

W[1]-hardness

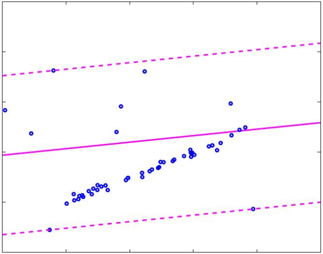
Consensus maximisation is W[1]-hard in the dimension d.

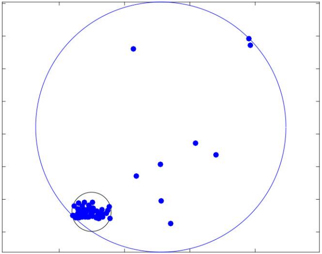
Proof: FPT reduction from K-clique.

$\implies$ There are no algorithms that are faster than $\mathcal{O}(N^{f(d)})$.

Chin et al. Robust Fitting in Computer Vision: Easy or Hard? CoRR abs/1802.06464 (2018).

The model is fitted to the outliers...





Recursive support set removal

- For any instance $(\mathcal{S}, f)$, if $f(\mathcal{S}) > \epsilon$, then there are outliers.
- Since $f(\mathcal{K}) = f(\mathcal{S}) > \epsilon$, where $\mathcal{K}$ is the support set, it is reasonable to suspect that $\mathcal{K}$ contains outliers.
- Idea: Recursively remove support sets of $\mathcal{S}$ until $f(\mathcal{S}) \leq \epsilon$.

Require: LP-type problem $(\mathcal{S}, f)$, inlier threshold ϵ.

while true **do**

    $\mathcal{K} \leftarrow$ Support set of $\mathcal{S}$.

    if $f(\mathcal{K}) > \epsilon$ **then**

        $\mathcal{S} \leftarrow \mathcal{S} \setminus \mathcal{K}$.

    else

        Break.

    end if

end while

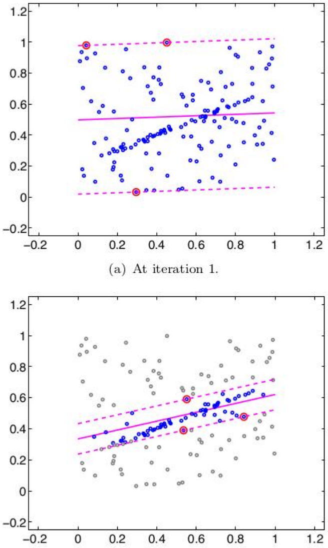
return $\mathcal{K}$.

Sim and Hartley. Removing Outliers Using The L-infty Norm. CVPR 2006.

Olsson et al. Outlier removal using duality. CVPR 2010.

Yu et al. An adversarial optimization approach to efficient outlier removal. ICCV 2011.

Recursive support set removal



(a) At iteration 1.

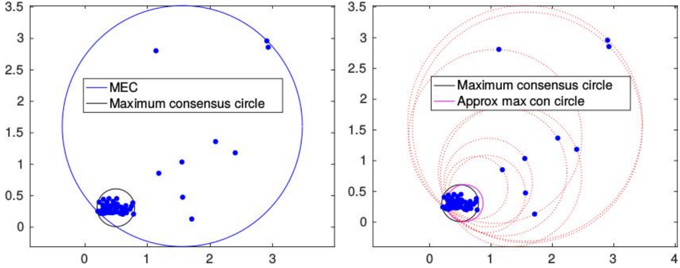
(b) At iteration 10.

(c) At iteration 30.

(d) Final result.

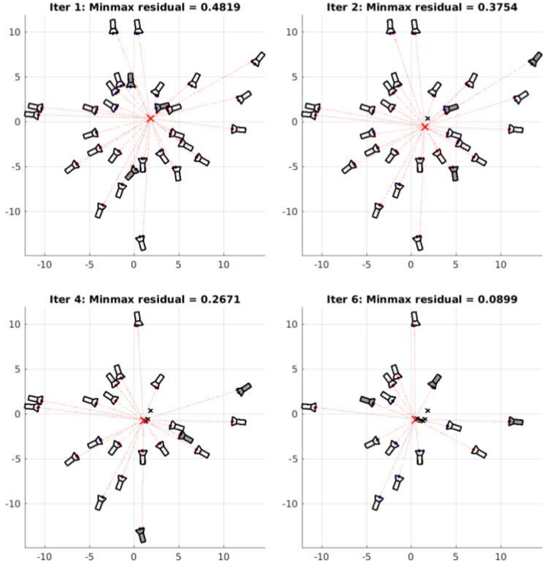
Chin and Suter. The maximum consensus problem: recent algorithmic advances. Morgan & Claypool Publishers, Feb. 2017.

Recursive support set removal



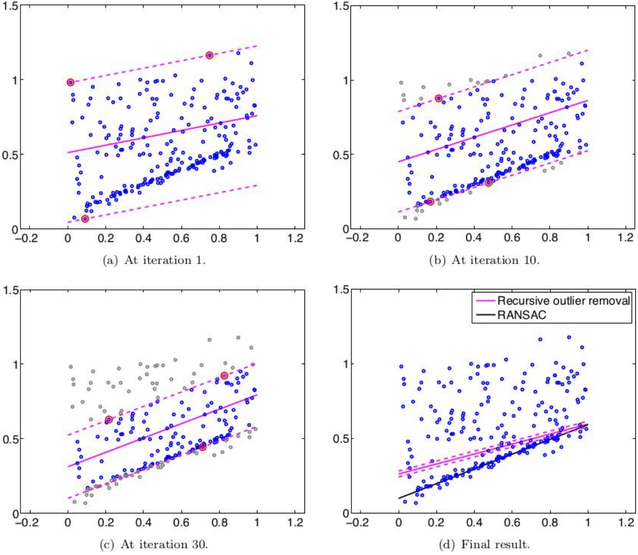
Chin and Suter. The maximum consensus problem: recent algorithmic advances. Morgan & Claypool Publishers, Feb. 2017.

Recursive support set removal



Chin and Suter. The maximum consensus problem: recent algorithmic advances. Morgan & Claypool Publishers, Feb. 2017.

Recursive support set removal (bad case)



(a) At iteration 1. (b) At iteration 10. (c) At iteration 30. (d) Final result.

Chin and Suter. The maximum consensus problem: recent algorithmic advances. Morgan & Claypool Publishers, Feb. 2017.

"True" inliers and outliers

- Call items in $\mathcal{I}^*$ the "true" inliers, and items in $\mathcal{O}^* = \mathcal{S} \setminus \mathcal{I}^*$ the "true" outliers.

Existence of true outliers

Let $\mathcal{C}$ be a subset of $\mathcal{S}$. If

$$f(\mathcal{C}) > \epsilon,$$

then $\mathcal{C}$ contains at least one item from $\mathcal{O}^*$.

- The LP-type properties allow us to prove the above easily: if $f(\mathcal{C}) > \epsilon$, then by monotonicity

$$f(\mathcal{I}^* \cup \mathcal{C}) \geq f(\mathcal{C}) > \epsilon. \tag{1}$$

If $\mathcal{C}$ is a subset of $\mathcal{I}^*$, then by locality

$$f(\mathcal{I}^* \cup \mathcal{C}) = f(\mathcal{I}^*) \leq \epsilon \tag{2}$$

which violates (1). $\implies$ $\mathcal{C}$ contains at least one item from $\mathcal{O}^*$.

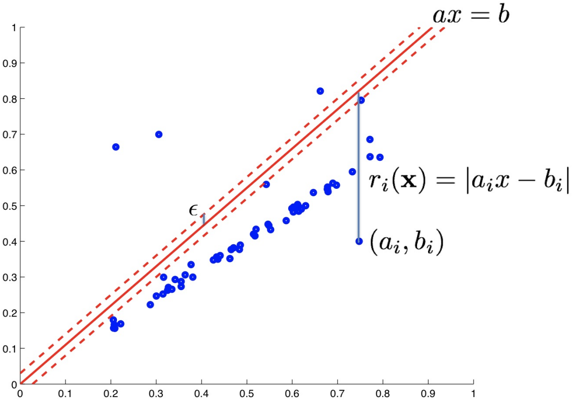
"True" inliers and outliers

- Recursively removing support sets is *sort of* justified, but it also removes true inliers in each iteration.
- For combinatorial dimension $d + 1$, in the worst case, d true inliers are removed in each iteration.
- To avoid losing too many inliers, the inlier proportion should be greater than

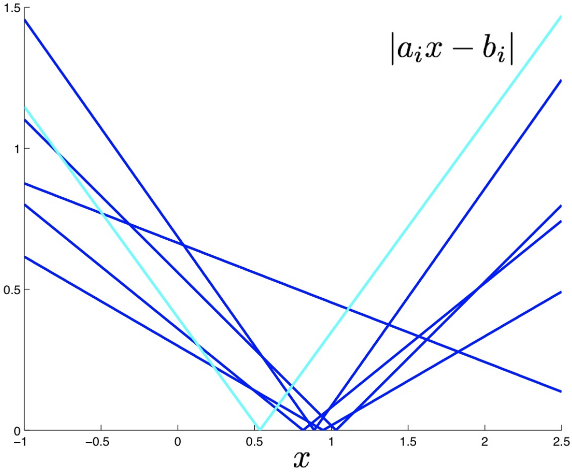
$$\frac{d}{d+1}.$$

$\Longrightarrow$ Method is not advisable for high-dimensional or high-outlier rate problems.

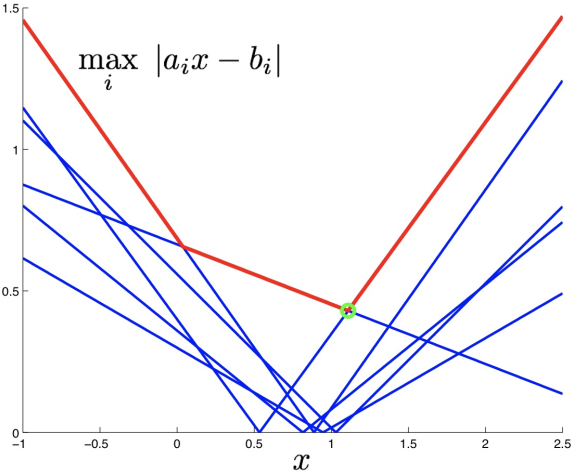
Running example: Robust subspace fitting (1D)



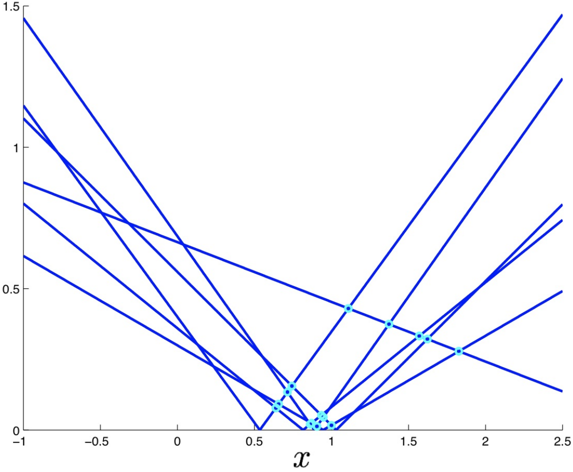
In parameter space



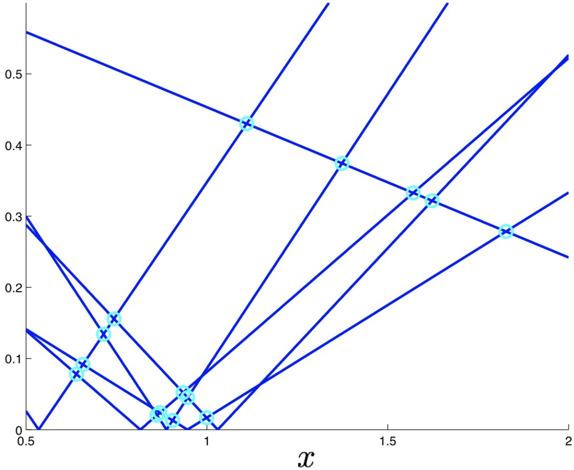
In parameter space



Bases for LP-type problem



Bases for LP-type problem (closeup)

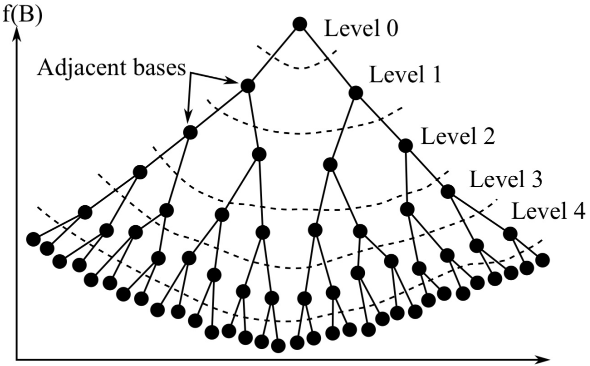


Every basis can be reached from the support set via a directed path.

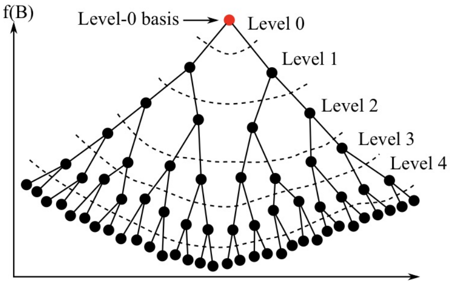
Matoušek. On geometric optimization with few violated constraints. Discrete comput. geom. 14.1 (1995), pp. 365–384.

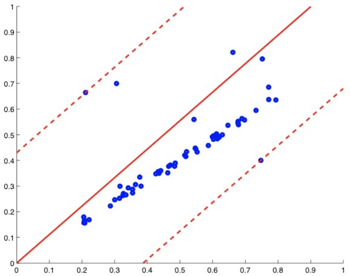
H. Li. A practical algorithm for l-infinity triangulation with outliers. CVPR 2007.

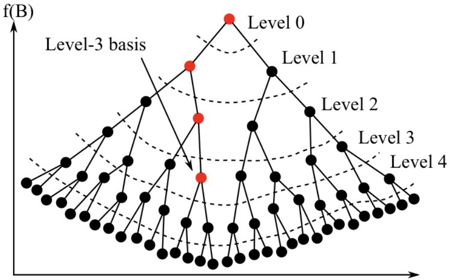
Tree structure

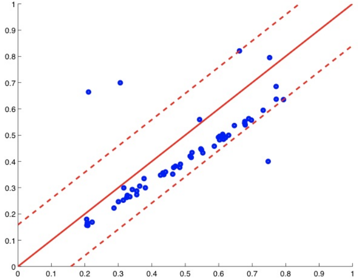


Level of basis

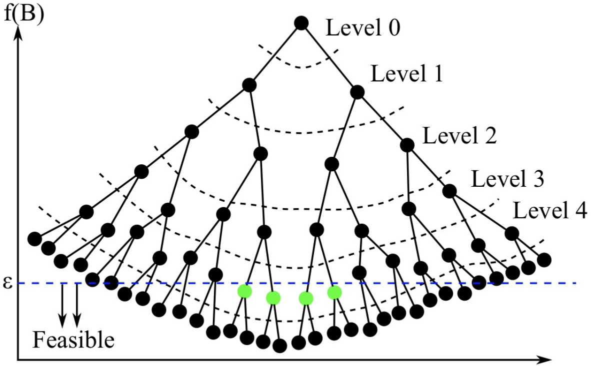








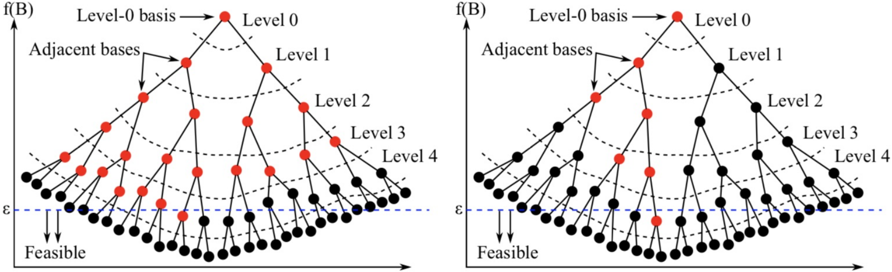
Consensus maximisation



Chin et al. Efficient globally optimal consensus maximisation with tree search. CVPR 2015.

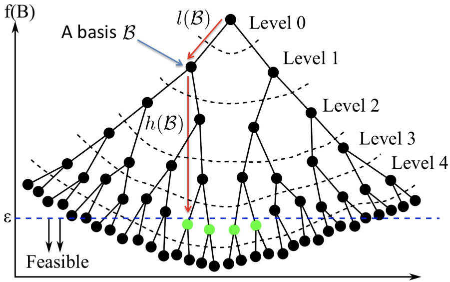
Tree search

Breadth first search (BFS) and A* search.



Chin et al. Efficient globally optimal consensus maximisation with tree search. CVPR 2015.

A* search



- A* search prioritises the bases to expand by

$$\text{eval}(\mathcal{B}) = \text{level}(\mathcal{B}) + h(\mathcal{B}),$$

  where h is a heuristic function.
- h estimates (but cannot overestimate) number of levels remaining.
- Performing recursive support set removal with $\mathcal{B}$ as the initial support set, and counting the number of removals, gives a valid heuristic.

Chin et al. Efficient globally optimal consensus maximisation with tree search. CVPR 2015.

Fixed parameter tractability

Fixed parameter tractability

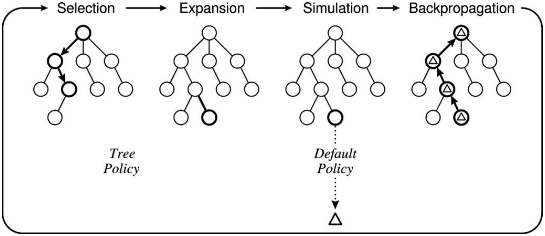
Consensus maximisation is FPT in the dimension d and number of outliers $|\mathcal{O}^*|$.

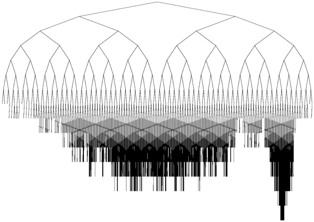
Proof: Use BFS to search the basis tree to get runtime of $\mathcal{O}((d+1)^{|\mathcal{O}^*|}L(N,d))$, where $L(N,d)$ is time to solve LP-type problem.

Chin et al. Robust Fitting in Computer Vision: Easy or Hard? CoRR abs/1802.06464 (2018).

Monte Carlo Tree Search

Use randomisation to explore the tree quickly.





Leads to asymmetric tree growth (tends to explore more promising branches).

Browne et al. A survey of Monte Carlo tree search methods. IEEE Trans. Computational Intelligence and AI in Games, 2012.

Le et al. RATSAC - Random Tree Sampling for Maximum Consensus Estimation. DICTA 2017.

Approximability

APX-hardness

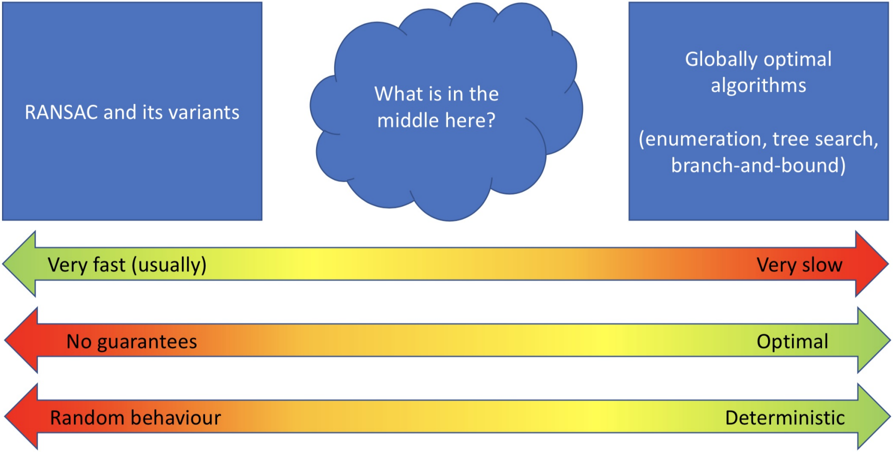
Consensus maximisation is APX-hard.

Proof: L-reduction from MAX-2SAT.

$\Longrightarrow$ There are no polynomial time algorithms that can solve consensus maximisation up to $(1-\delta)|\mathcal{I}^*|$ for any pre-determined δ.

Chin et al. Robust Fitting in Computer Vision: Easy or Hard? CoRR abs/1802.06464 (2018).

So what can be done?



So what can be done?

- RANSAC and variants (no guarantees, not necessarily efficient).
- Use M-estimators and IRLS.
- Deterministic refinement methods:
  - Penalty formulation + bilinear programming (Le et al.).
  - Iteratively reweighted L1 (Purkait et al.).

Le et al. An exact penalty method for locally convergent maximum consensus. CVPR 2017.

Purkait et al. Maximum consensus parameter estimation by reweighted L1 methods. EMMCVPR 2017.

Table of Contents

References I

N. Amenta, M. Bern, and D. Eppstein. "Optimal point placement for mesh smoothing". In: *Journal of Algorithms* 30.2 (1999), pp. 302–322.

C. B. Browne et al. "A survey of Monte Carlo tree search methods". In: *IEEE Trans. Computational Intelligence and AI in Games* 4.1 (Mar. 2012), pp. 1–43.

T.-J. Chin, Z. Cai, and F. Neumann. "Robust Fitting in Computer Vision: Easy or Hard?" In: *CoRR* abs/1802.06464 (2018).

T.-J. Chin and D. Suter. *The maximum consensus problem: recent algorithmic advances*. Ed. by G. Medioni and S. Dickinson. Synthesis Lectures on Computer Vision. Morgan & Claypool Publishers, San Rafael, CA, U.S.A., Feb. 2017.

T.-J. Chin et al. "Efficient globally optimal consensus maximisation with tree search". In: *CVPR*. 2015.

K. L Clarkson. "Coresets, sparse greedy approximation, and the Frank-Wolfe algorithm". In: *SODA*. 2008.

K. L. Clarkson. "Coresets, sparse greedy approximation, and the Frank-Wolfe algorithm". In: *ACM Trans. Algorithms* 6.4 (2010), 63:1–63:30.

P.-A. Coquelin and R. Munos. "Bandit algorithms for tree search". In: *UAI*. 2007.

O. Enqvist et al. "Robust fitting for multiple view geometry". In: *European Conference on Computer Vision (ECCV)*. 2012.

References II

D. Eppstein. “Quasiconvex programming”. In: *Combinatorial and Computational Geometry* 25 (2005).

H. Le, T.-J. Chin, and D. Suter. “An exact penalty method for locally convergent maximum consensus”. In: *CVPR*. 2017.

H. Le, T.-J. Chin, and D. Suter. “RATSAC - Random Tree Sampling for Maximum Consensus Estimation”. In: *DICTA*. 2017.

H. Li. “A practical algorithm for l_∞ triangulation with outliers”. In: *CVPR*. 2007.

H. Li. “Efficient reduction of L-infinity geometry problems”. In: *CVPR*. 2009.

J. Matoušek. “On geometric optimization with few violated constraints”. In: *Discrete comput. geom.* 14.1 (1995), pp. 365–384.

J. Matoušek, M. Sharir, and E. Welzl. “A subexponential bound for linear programming”. In: *Algorithmica* 16 (1996), pp. 498–516.

C. Olsson, O. Enqvist, and F. Kahl. “A polynomial-time bound for matching and registration with outliers”. In: *CVPR*. 2008.

C. Olsson, A. Eriksson, and R. Hartley. “Outlier removal using duality”. In: *CVPR*. 2010.

P. Purkait, C. Zach, and A. Eriksson. “Maximum consensus parameter estimation by reweighted L1 methods”. In: *EMMCVPR*. 2017.

References III

Y. Seo and R. I. Hartley. "A Fast Method to Minimize L Error Norm for Geometric Vision Problems". In: *ICCV*. 2007.

K. Sim and R. Hartley. "Removing outliers using the l_∞ norm". In: *CVPR*. 2006.

J. Yu et al. "An adversarial optimization approach to efficient outlier removal". In: *ICCV*. 2011.

Q. Zhang and T.-J. Chin. "Coresets for triangulation". In: *IEEE TPAMI* (2017). DOI: 10.1109/TPAMI.2017.2750672.