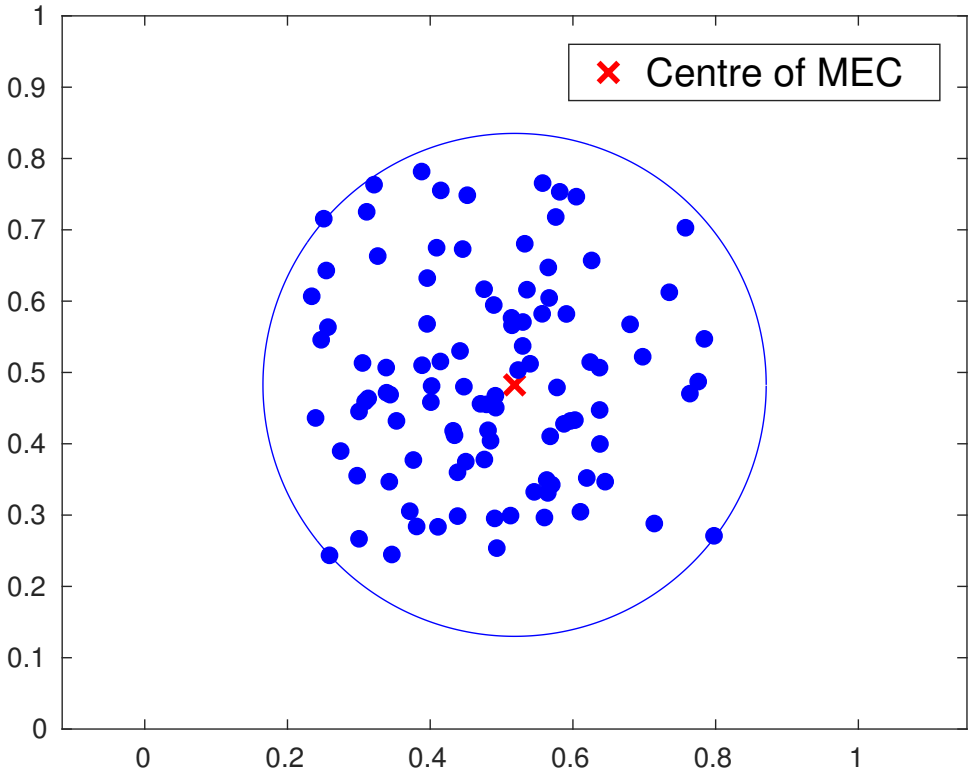
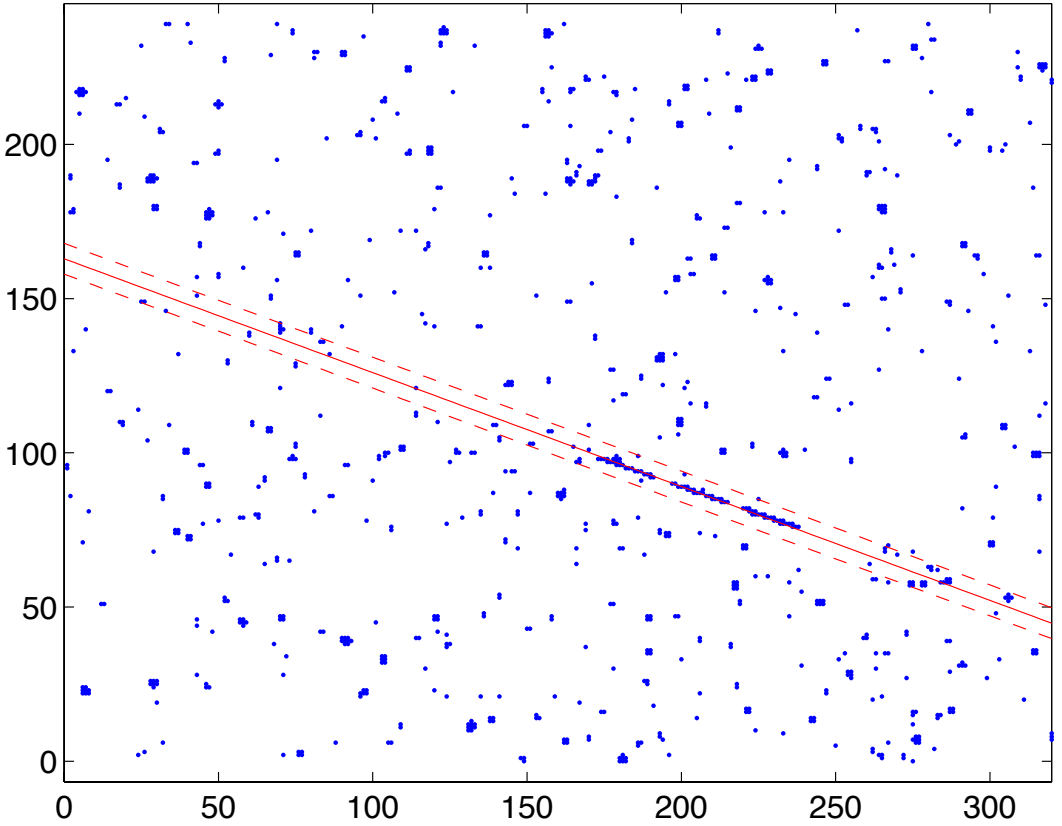
Monotone Boolean Functions: LP-Type Problems, MaxCon, feasibility/infeasibility

David Suter

Based on work with: Ruwan Tennakoon, Erchuan Zhang, Tat-Jun Chin, Ali Bab-Hadiashar

Ruwan Tennakoon, David Suter, Erchuan Zhang, Tat-Jun Chin, and Alireza Bab-Hadiashar. Consensus maximisation using influences of monotone boolean functions. In CVPR2021, 2021. Oral Presentation (17% of accepted papers, roughly 3% of sumitted papers)


Centre of MEC
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.2
0.4
0.6
0.8
1


200
150
100
50
0
0
50
100
150
200
250
300

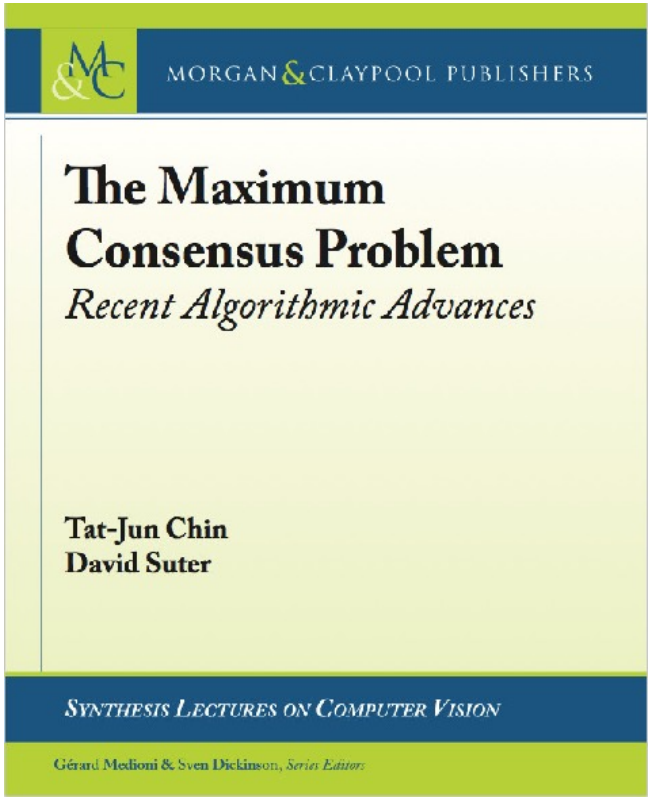
LP-type Problem

Generalisation of Linear Program – linear objective, linear inequality constraints.

LP-type problem and MaxCon – the CVPR2015 paper….

First to recognize that MaxCon (in most forms used) is an example of an LP-type problem

Used ideas from solving LP-type problem - fused with AI 101 “Search” – A* - to give the first *provably optimal* solution of MaxCon that will ”often” (in restricted situations) run efficiently – but……..
Unfortunately…….
Can take “forever”…..(exponential worst case…).



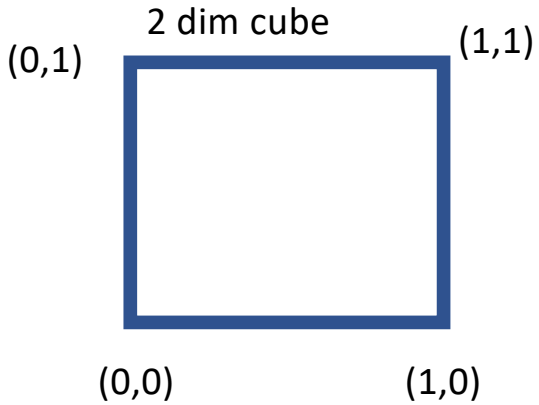
Ongoing work – ARC Discovery projects 2020-2022
Mathematics of clustering of data - how to prove statements about quality of result by various algorithms.
Huge number of applications – visual tracking, visual segmentation, robust fitting in the presence of multi-structure.

New Line of Work – Montone Boolean Functions (MBF) and LP-type Problems

So what is the recent MBF work all about…..

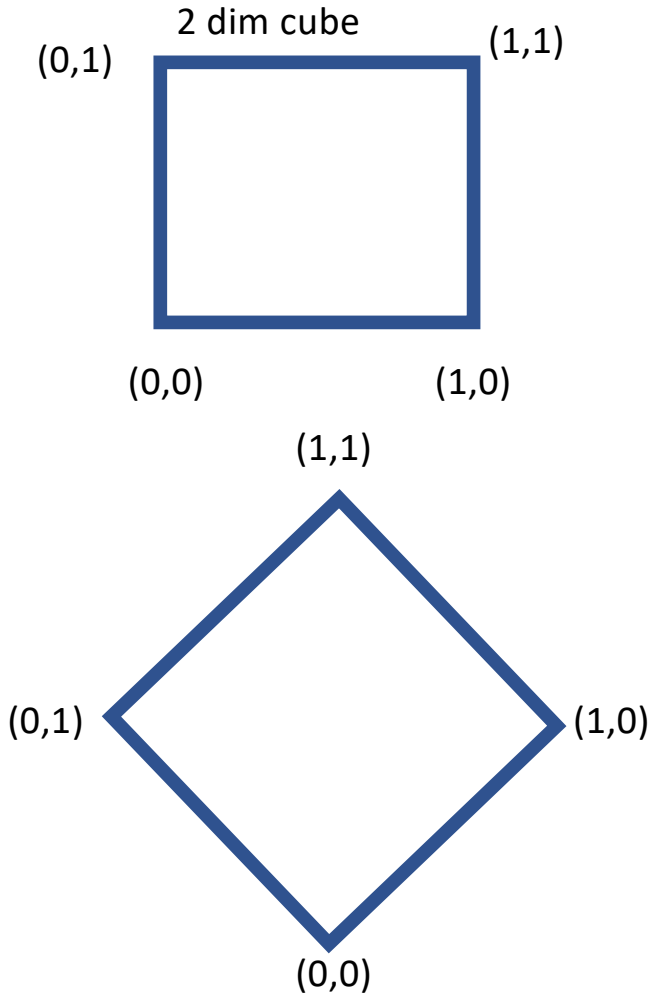
What's a Monotone Boolean Function??

Boolean Cube (without arrows)

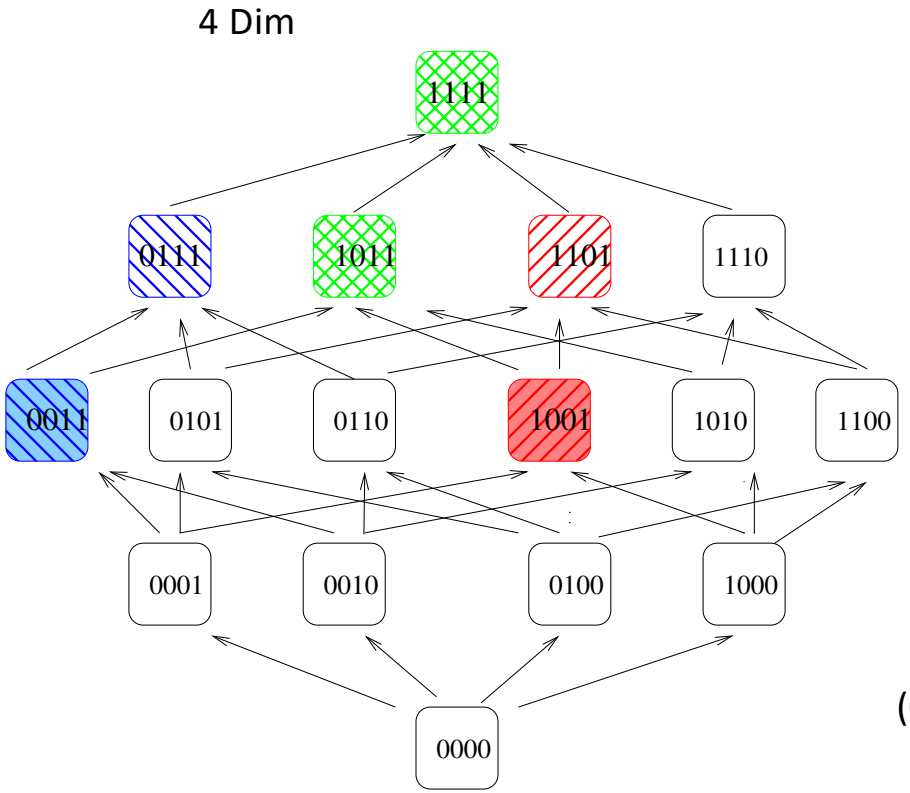


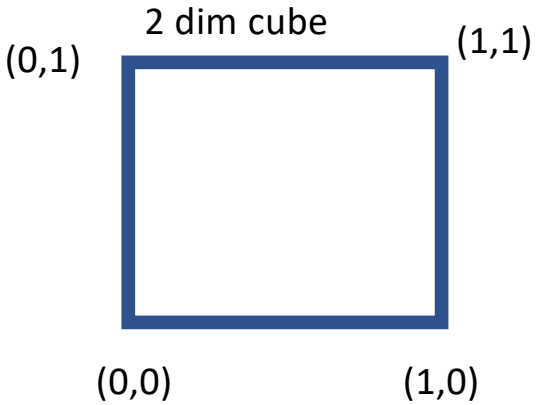
What's a Monotone Boolean Function??

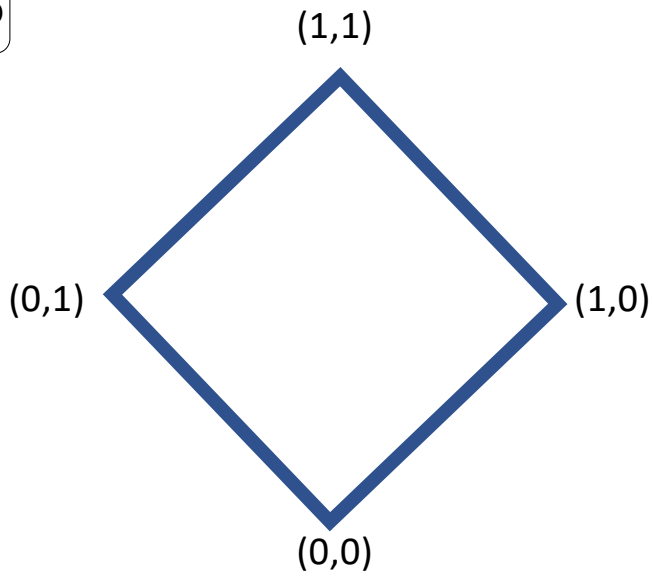
Boolean Cube (without arrows)



What's a Monotone Boolean Function??



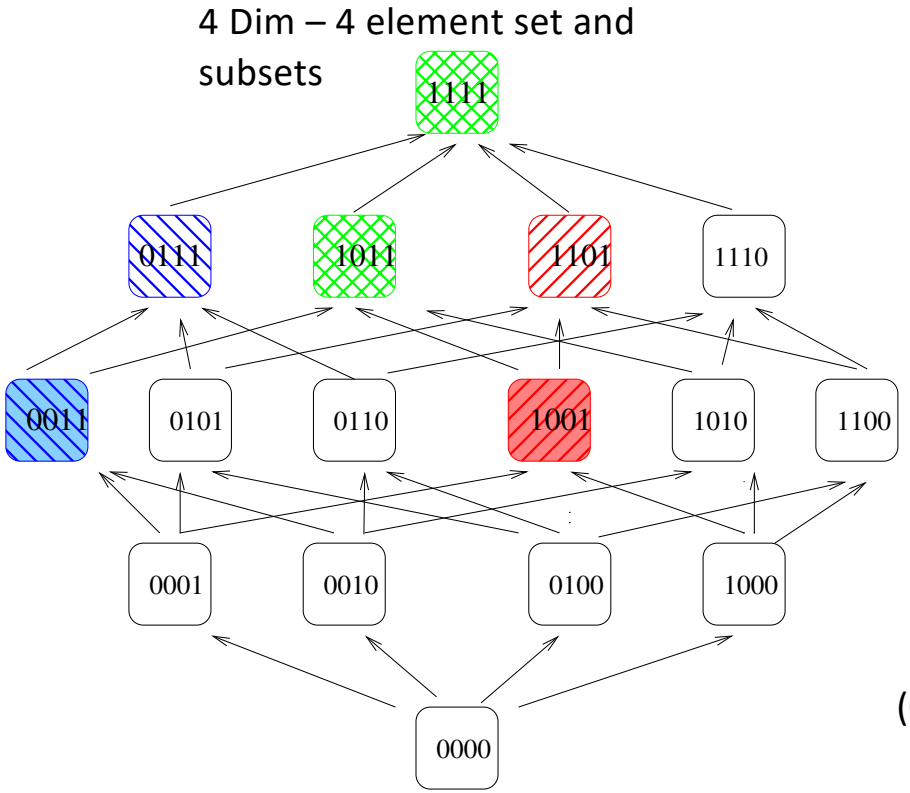


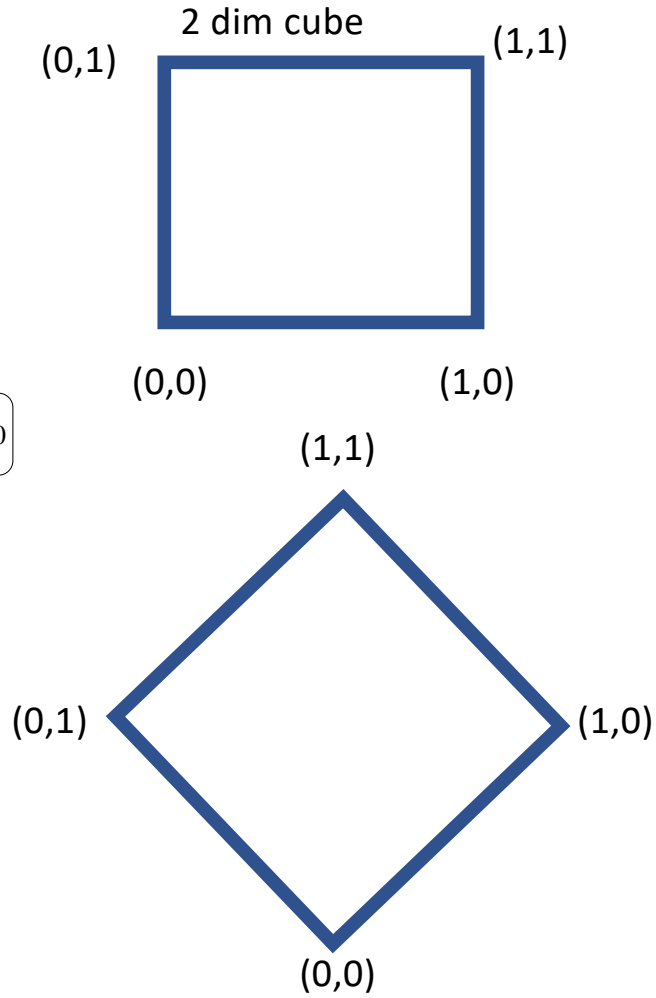


What's a Monotone Boolean Function??

Boolean Cube (without arrows)

Hasse Diagram (with arrows:
and interpreting 0's in position i as absence of element i; 1's as presence Of element i in *that subset (that vertex in the Boolean Cube/Hasse Diagram).*





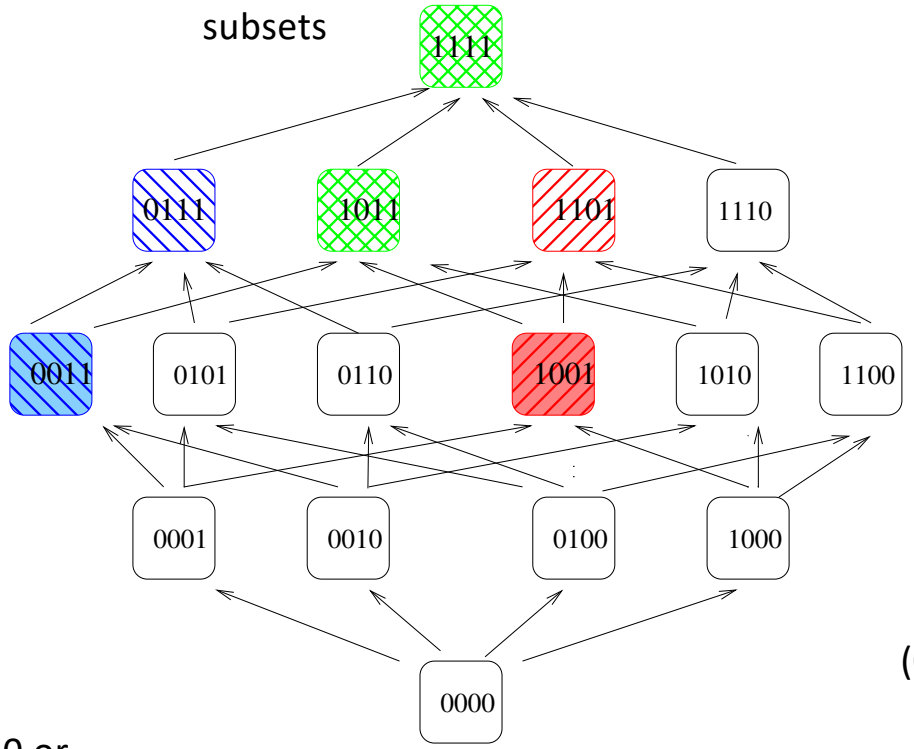
What's a Monotone Boolean Function??

Boolean Cube (without arrows)

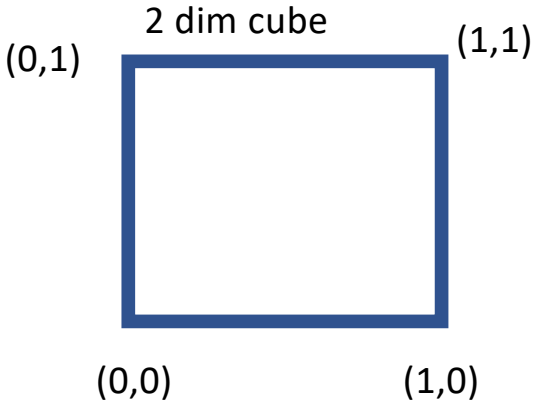
Hasse Diagram (with arrows:
and interpreting 0's in position i as absence of element i; 1's as presence Of element i in *that subset (that vertex in the Boolean Cube/Hasse Diagram).*

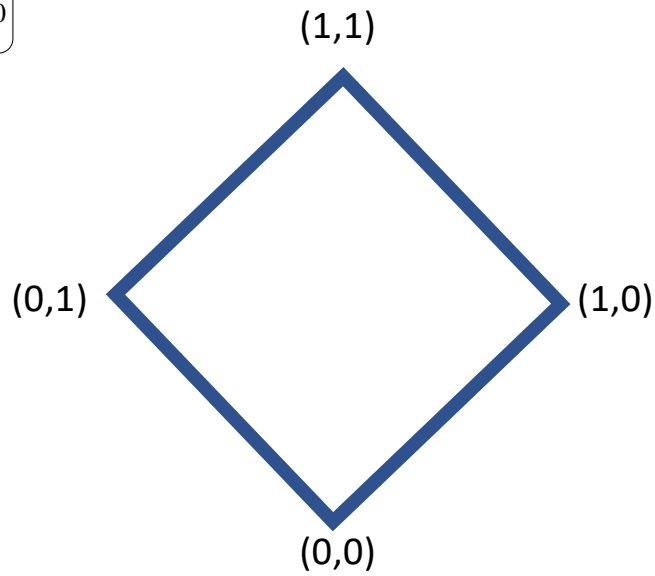
Boolean Function over this cube/Hasse Diagram – assign 0 or 1 to VALUE of function at each vertex. Here use colours/shading – uncoloured/unshaded f()=0

4 Dim – 4 element set and subsets



2 dim cube





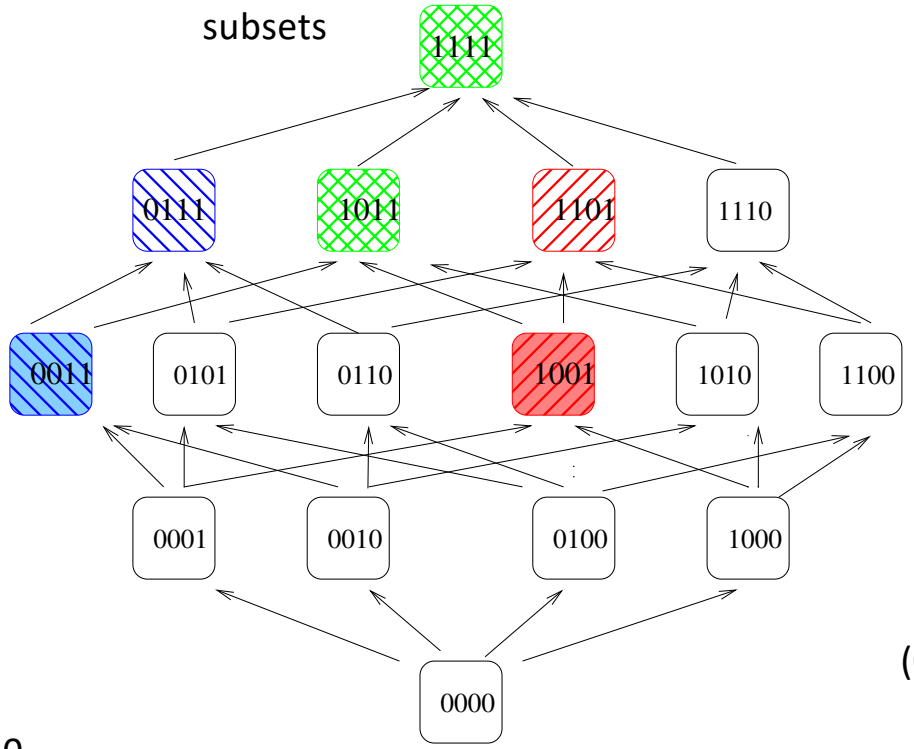
What's a Monotone Boolean Function??

Boolean Cube (without arrows)

Hasse Diagram (with arrows:
and interpreting 0's in position i as absence of element i; 1's as presence Of element i in *that subset (that vertex in the Boolean Cube/Hasse Diagram).*

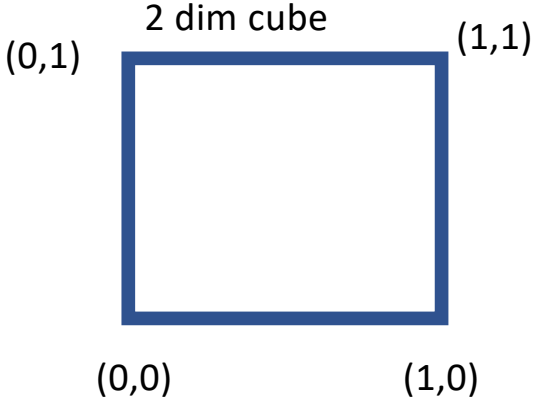
Boolean Function over this cube/Hasse Diagram – assign 0 or 1 to VALUE of function at each vertex.

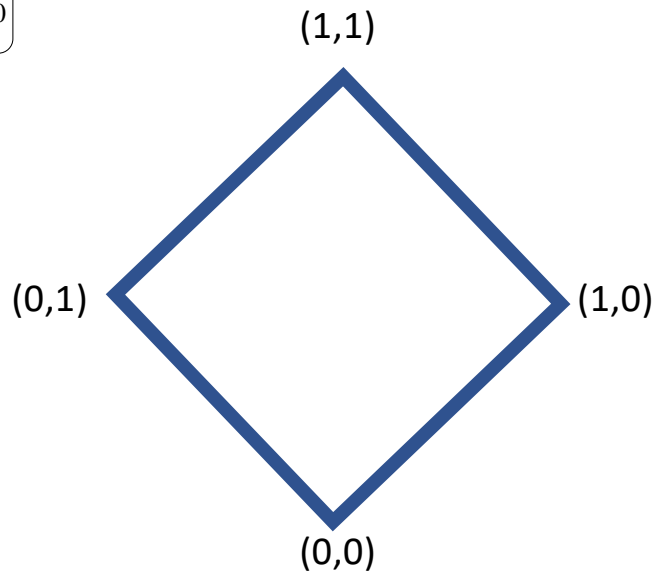
4 Dim – 4 element set and subsets



Monotone Boolean Function – function can only increase (0->1) on going up – NEVER decrease (1->0)

2 dim cube





What does this have to do with MaxCon and Computer Vision?

The previously presented “Tree Search” (on which we used A*) is REALLY “Cube Search” from the top of a Boolean Cube.

Admittedly, that “Tree Search” only considered downward edges that correspond to elements of a basis (for the parent vertex).

Moreover, that “Tree Search” effectively “turned” the Boolean Cube into a tree by duplicating vertices that are arrived at by more than one path. (This is the distinction between state space graph – the Boolean Cube; and the search graph – the Tree. A general distinction we made in those “Search” lectures).

OK – the search is search on the Boolean Cube – but what does MBF have to do with MaxCon?

Moreover….

Infeasiblity is a monotone Boolean function (!)

Why?

If a set of points is infeasible then adding extra points (going up) can only remain infeasible.

If a set of points is feasible then deleting points (going down) can only remain feasible.

So….going up the cube you can only transition at most once (0->1; feasible -> infeasible)

Why do LP-type problems so readily connect with Monotone Boolean Functions??

Another viewpoint… LP-type problems are DEFINED by stating they are related to a monotone function.

Property 1 (Monotonicity). *For every sets, $\mathcal{P}$, $\mathcal{Q}$, $\mathcal{S}$: $\mathcal{P} \subseteq \mathcal{Q} \subseteq \mathcal{S}$, the inequalities $f(\mathcal{P}) \leq f(\mathcal{Q}) \leq f(\mathcal{S})$ hold.*

In the original motivation (generalizing Linear Programs) – the function f corresponded to the function that returns the optimal value of a LP that is minimizing some objective function over a set of constraints. If you add extra constraints the optimal value can only get WORSE (higher).

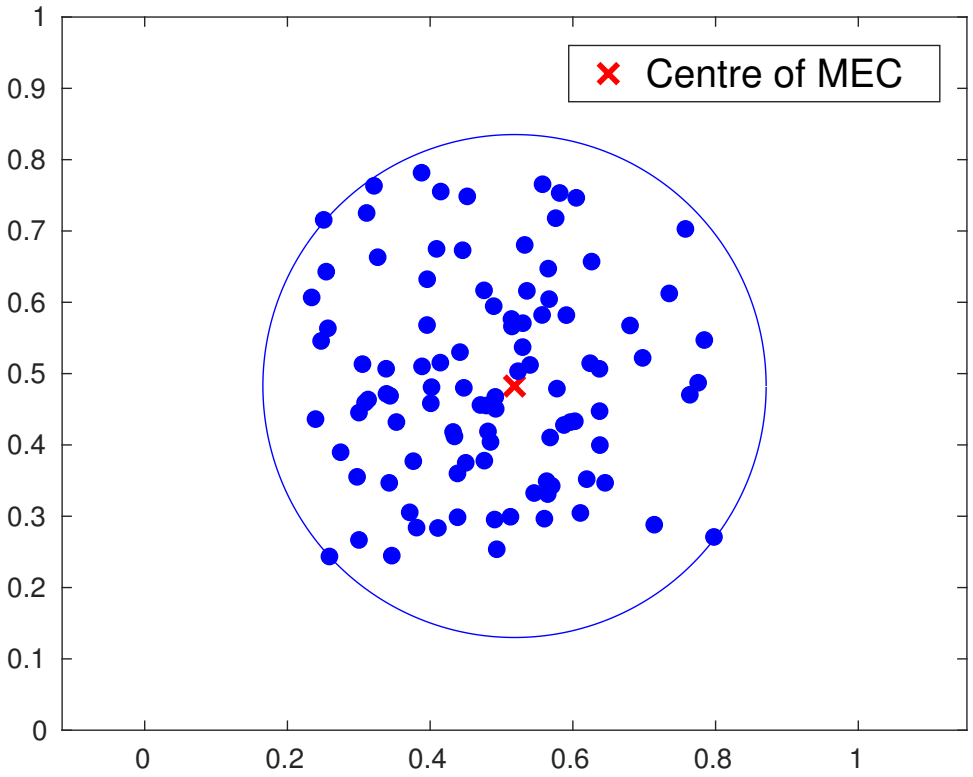
In our fitting problem, f would be the minimum tolerance (epsilon) required for the all the data to fit the model.

So the f is typically REAL VALUED (not Boolean).

But this isn't a big deal….

Indeed, if we were to THRESHOLD f (saying for example – as we do in MaxCon – that we only allow epsilon to be a certain size (and feasible solution exists only if can be fit by that epsilon or less) then we arrive at a MONOTONE Boolean Function.

Additional properties of LP-type problems – Basis!!



For line fitting the basis has size 3.

For other problems the basis typical has a fixed size but usually different to 3.

For Circle/Ball fitting – the 2D version actually has basis with size mostly 3 but sometimes 2…..we'll ignore the complication where the Basis size can take on more than one value for that model fitting problem - it doesn't change the story a lot…

Additional properties of LP-type problems – Basis!!

So this means we are actually dealing with SPECIAL Monotone Boolean Functions…..WHY?

1. Any subset (of data points) smaller than the basis size is automatically feasible.
2. Any subset that infeasible MUST contain a basis sized subset of points that is itself infeasible

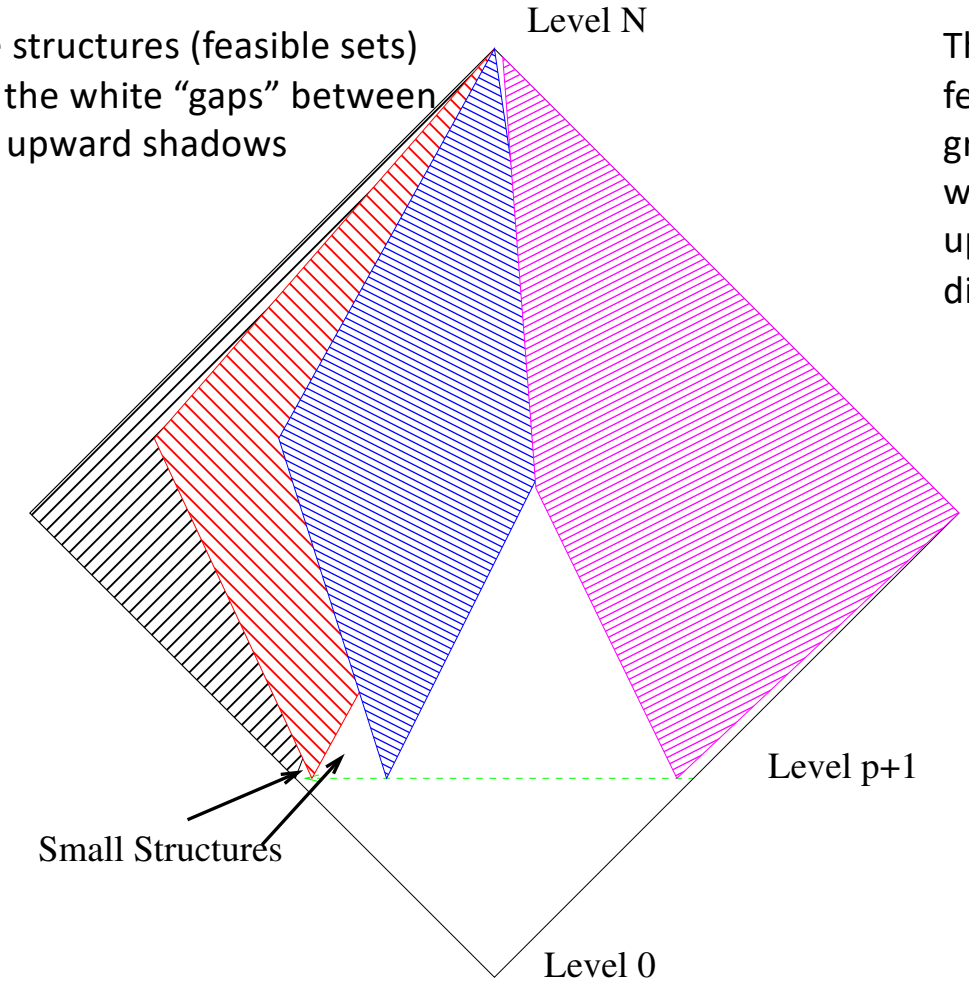
So……..

The minimum sized infeasible sets (all of which have the same size – with some exceptions mentioned before – which we ignore for simplicity) COMPLETELY DETERMINE the Montone Boolean Feasibility function over the whole cube.

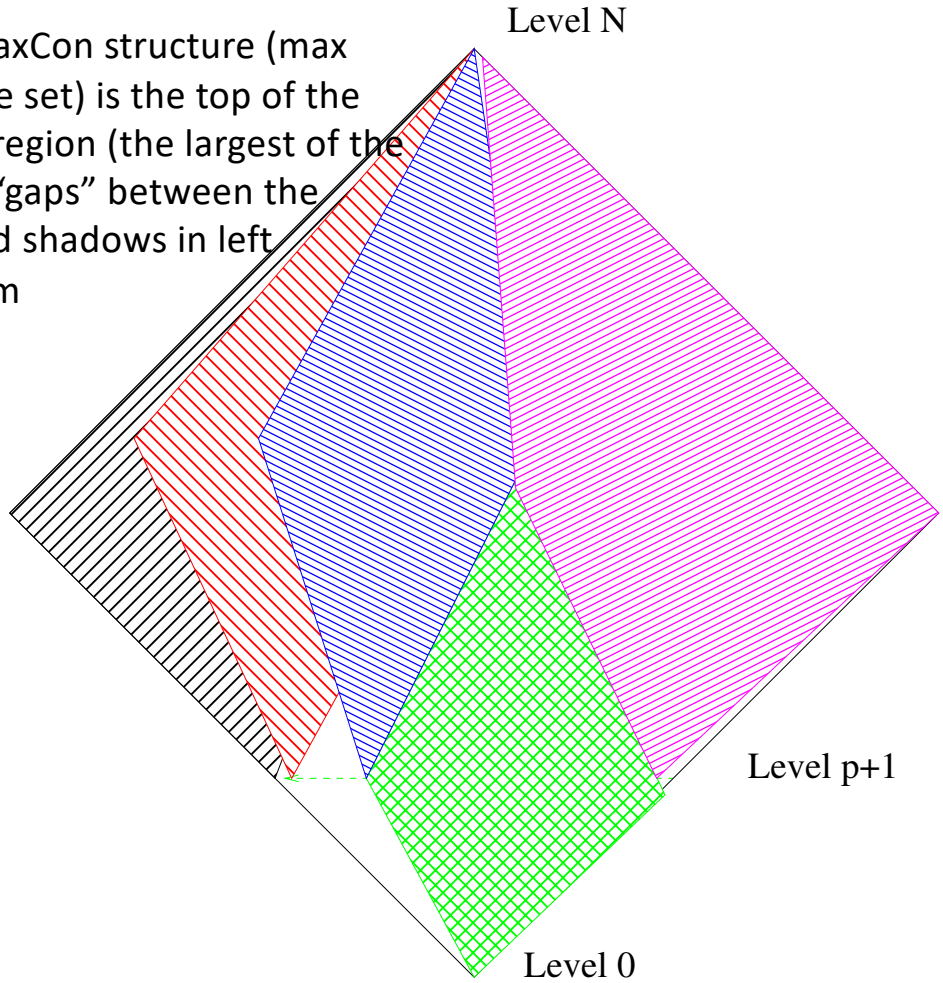
In fact – every infeasible subset (node) in the cube is only feasible because it is the “upward shadow” of at least one minimal infeasible subset.

Min sized infeasible sets and shadows….

The structures (feasible sets) are the white “gaps” between the upward shadows



The MaxCon structure (max feasible set) is the top of the green region (the largest of the white “gaps” between the upward shadows in left digaram



Studying MaxCon is older than we thought....

AUSTRALIA ECU EDITH COWAN UNIVERSITY

AN ALGORITHM FOR FINDING THE MAXIMAL UPPER ZERO OF AN ARBITRARY MONOTONIC FUNCTION OF THE ALGEBRA OF LOGIC*

E. G. KUL'YANOV

Moscow

(*Received* 13 *December* 1973)

AN ALGORITHM for finding the zero of maximum norm of an arbitrary monotonic function of the algebra of logic is presented. The algorithm is written as an operational scheme.

1. Formulation of the problem

Many applied problems can be formalized by means of the approximation of the phenomena considered by monotonic functions of the algebra of logic. A number of papers exist [1–3], devoted to an analysis of the class of monotonic functions, and to an investigation of their general properties. A search of the extremal characteristics of the phenomena studied leads to a search of the individual extremal points of the monotonic function, for example, to a search of the zero of maximal norm of the monotonic function. Our note is devoted to a description of the algorithm of this search.

Russian Math. Surveys **64**:5 787–803 © 2009 RAS(DoM) and LMS
Uspekhi Mat. Nauk **64**:5 3–20 DOI 10.1070/RM2009v064n05ABEH004640

Some unsolved problems in discrete mathematics and mathematical cybernetics

A. D. Korshunov

Abstract. There are many unsolved problems in discrete mathematics and mathematical cybernetics. Writing a comprehensive survey of such problems involves great difficulties. First, such problems are rather numerous and varied. Second, they greatly differ from each other in degree of completeness of their solution. Therefore, even a comprehensive survey should not attempt to cover the whole variety of such problems; only the most important and significant problems should be reviewed. An impersonal choice of problems to include is quite hard. This paper includes 13 unsolved problems related to combinatorial mathematics and computational complexity theory. The problems selected give an indication of the author's studies for 50 years; for this reason, the choice of the problems reviewed here is, to some extent, subjective. At the same time, these problems are very difficult and quite important for discrete mathematics and mathematical cybernetics.

Bibliography: 74 items.

Keywords: graph reconstruction by subgraphs, Hamiltonian cycles, disjunctive normal forms, the snake-in-the-box problem, graph isomorphism, lower bounds, NP-completeness, polynomial problems, Boolean functions hard to compute, cube piercing, perfect binary codes, Steiner triple systems.

Russian Math. Surveys **58**:5 929–1001 ©2003 RAS(DoM) and LMS
Uspekhi Mat. Nauk **58**:5 89–162 DOI 10.1070/RM2003v058n05ABEH000667

Monotone Boolean functions

A. D. Korshunov

Abstract. Monotone Boolean functions are an important object in discrete mathematics and mathematical cybernetics. Topics related to these functions have been actively studied for several decades. Many results have been obtained, and many papers published. However, until now there has been no sufficiently complete monograph or survey of results of investigations concerning monotone Boolean functions. The object of this survey is to present the main results on monotone Boolean functions obtained during the last 50 years.

Contents

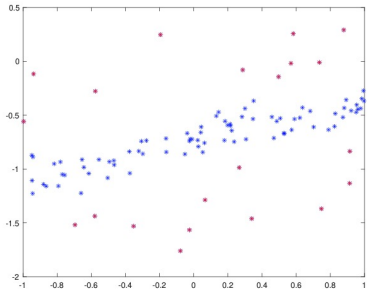
OK – but can you *do* anything???
Influence…. (well it's a start…)

We have proved (Erchuan Zhang) that a property of Boolean Functions (called Influence) is related to outlier/inlier….

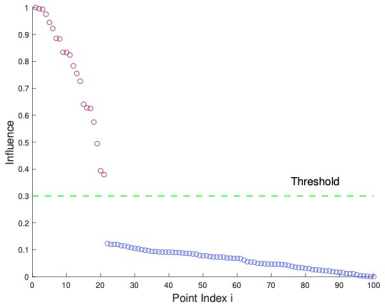
Put in short, larger influence corresponds with outlier.

Indeed, if we could calculate the influence EXACTLY then we can simply threshold the influence and we are done.

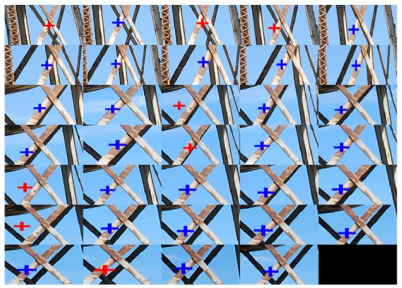
Those points with influence below a certain value will be inliers and the rest outlier…



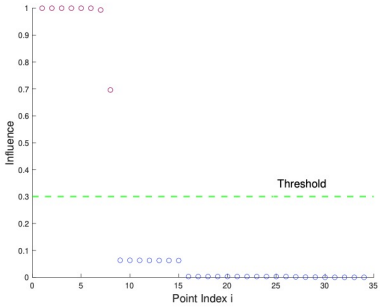
(a) Points on a plane.



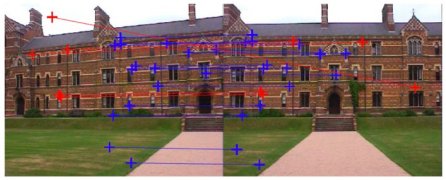
(b)



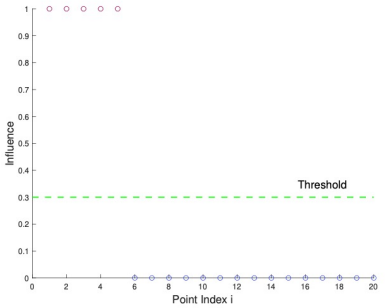
(c) Feature correspondences across multiple calibrated views.



(d)

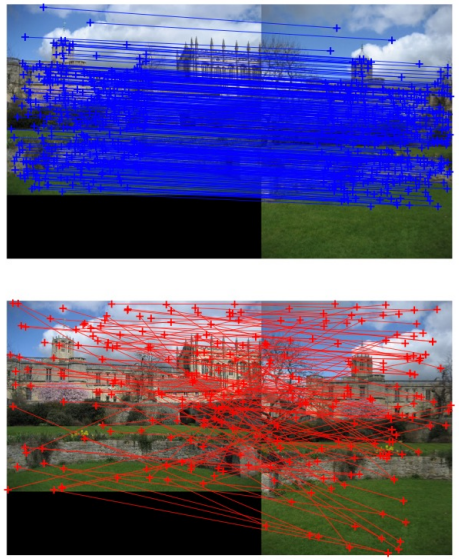


(e) Feature correspondences across two views.

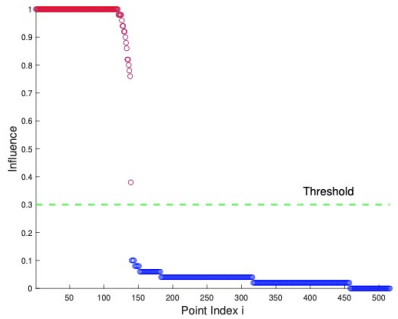


(f)

Figure 1. Data instances $\mathcal{D}$ with outliers (left column) and their normalised influences (right column). Row 1 shows a line fitting instance with $d = 2$ and $N = 100$; Row 2 shows a triangulation instance with $d = 3$ and $N = 34$; Row 3 shows a homography estimation instance with $d = 8$ and $N = 20$. In each result, the normalised influences were thresholded at 0.3 to separate the inliers (blue) and outliers (red).



(a)



(b)

Figure 3. (a) A homography estimation instance with $N = 516$ correspondences, separated into inliers (blue) and outliers (red) according to the normalised approximate influences in (b), which were computed using Algorithm 1. Note that only about $M = 800$ iterations were used in Algorithm 1 to achieve this result.

Algorithm 1 Classical algorithm to compute influence.

Require: N input data points $\mathcal{D}$, combinatorial dimension k, inlier threshold ϵ, number of iterations M.

for $m = 1, \ldots, M$ **do**
  $\mathbf{z}^{[m]} \leftarrow$ Randomly choose k-tuple from $\mathcal{D}$.
  for $i = 1, \ldots, N$ **do**
    if $f(\mathbf{z}^{[m]} \oplus \mathbf{e}_i) \neq f(\mathbf{z}^{[m]})$ **then**
      $X_i^{[m]} \leftarrow 1$.
    else
      $X_i^{[m]} \leftarrow 0$.
    end if
  end for
end for
for $i = 1, \ldots, N$ **do**
  $\hat{\alpha}_i \leftarrow \frac{1}{M} \sum_{m=1}^{M} X_i^{[m]}$.
end for
return $\{\hat{\alpha}_i\}_{i=1}^{N}$.

But calculating influence more or less means counting all edges between the feasible and infeasible regions of the cube – the cube has size 2^N - that is not tractable!!

If we can't calculate influence exactly then MAYBE we can *estimate* influence well enough (random sampling the cube) for these *estimates* to be a bit like a heuristic. The higher they are the more likely the point is an outlier – the more likely it is that discarding it is on the optimal search path(!)

So we are examining ideas around this….

Note, for a general Boolean Function, the influence isn't quite so "nice". For a MONOTONE Boolean Function the influence is actually the 1^{st} order Fourier Coefficients and is a much "nicer" quantity to work with.

At the moment we are using the influence as a GREADY cost function – remember back in the search lectures – we don't maintain a fringe because we ONLY choose the top influence data point (from a basis) as "run with that".

Thus we loose optimality guarantees for several reasons:

1) We only have approx. influence not exact
2) We only use a greedy algorithm not A*
3) Even if we decided to use A* - how do we ensure the estimated influences would give an ADMISSABLE heuristic?

An algorithm!!

Data: $\mathcal{X}$ indexed by $[1 \dots n]$
Result: Maximum consensus set $\mathcal{C}$.
$\mathcal{C} \leftarrow \{1, \dots, n\}$;
while $\mathcal{X}[\mathcal{C}]$ *is not feasible* **do**
    $\mathbf{x} \leftarrow$ The $p+1$ edge points from L_∞ fit to $\mathcal{X}[\mathcal{C}]$;
    Calculate $\hat{f}(\{i\}) \quad \forall i \in \mathbf{x}$;
    $r \leftarrow \underset{i \in x}{argmax} \left\{ \hat{f}(\{i\}) \right\}$;
    $\mathcal{C} \leftarrow \mathcal{C} \setminus r$;
end

Algorithm 1: Computing maximum consensus set (BMF-maxcon).

New - Refined Algorithm

Algorithm 1 Algorithm for finding the maximum consensus set using influences of BMFs.

Require: $\{\mathcal{X}_i\}_{i=1}^n$, method $\in$ {'max', 'L∞'}, m.

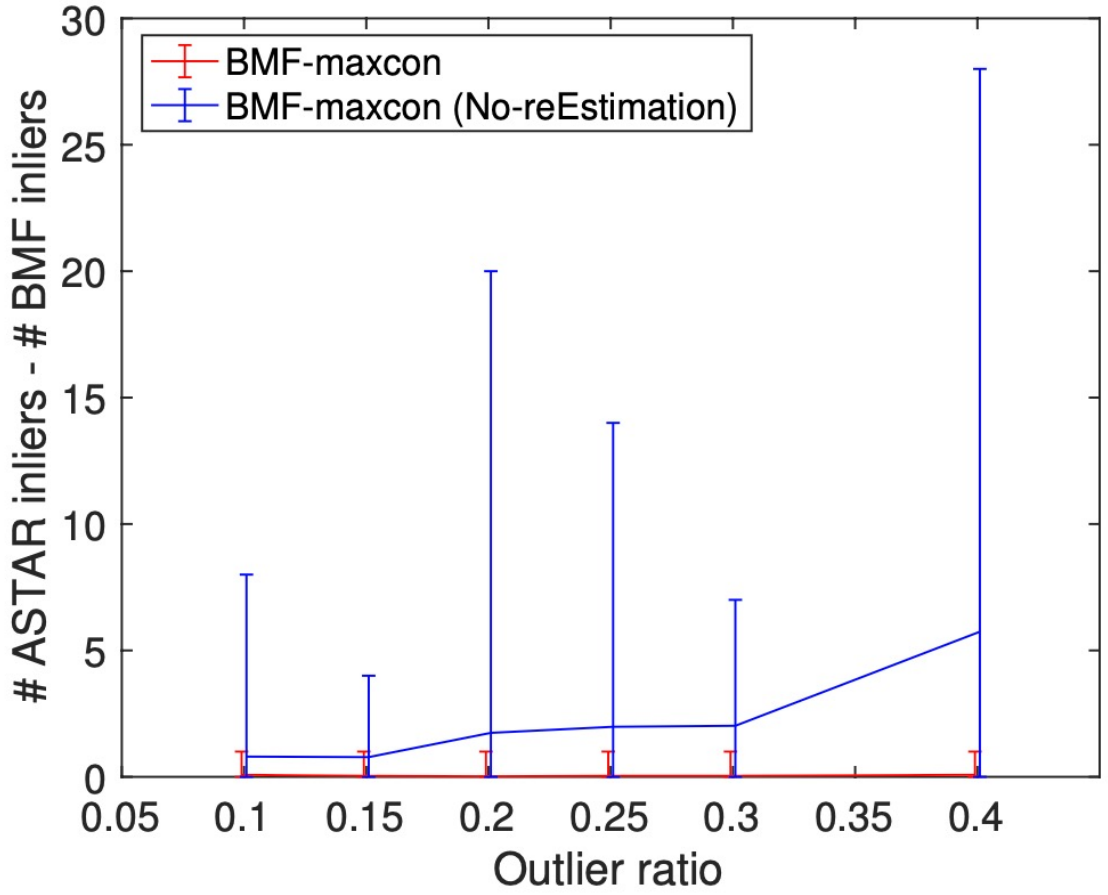
$\mathbf{x} \leftarrow [1, \ldots, 1]_{[1\times n]}$
repeat
  $b \leftarrow \{i : x_i = 1\}$
  if method = 'L∞' **then**
    $b \leftarrow$ The $p+1$ edge points from L_∞ fit to $\bar{\mathcal{X}} = \{\mathcal{X}_i : i \in b\}$
  end if
  Estimate $\hat{f}(\{i\})$ $\forall i \in b$ using m function queries
  $r \leftarrow \underset{i\in b}{argmax}\ \hat{f}(\{i\})$
  $x_r \leftarrow 0$
until $f(\mathbf{x}) = 0$
$\mathbf{x} \leftarrow$ Run local expansion step in (Algorithm 2)
return maximum consensus set $s \leftarrow \{i : x_i = 1\}$

Algorithm 2 Local expansion step.

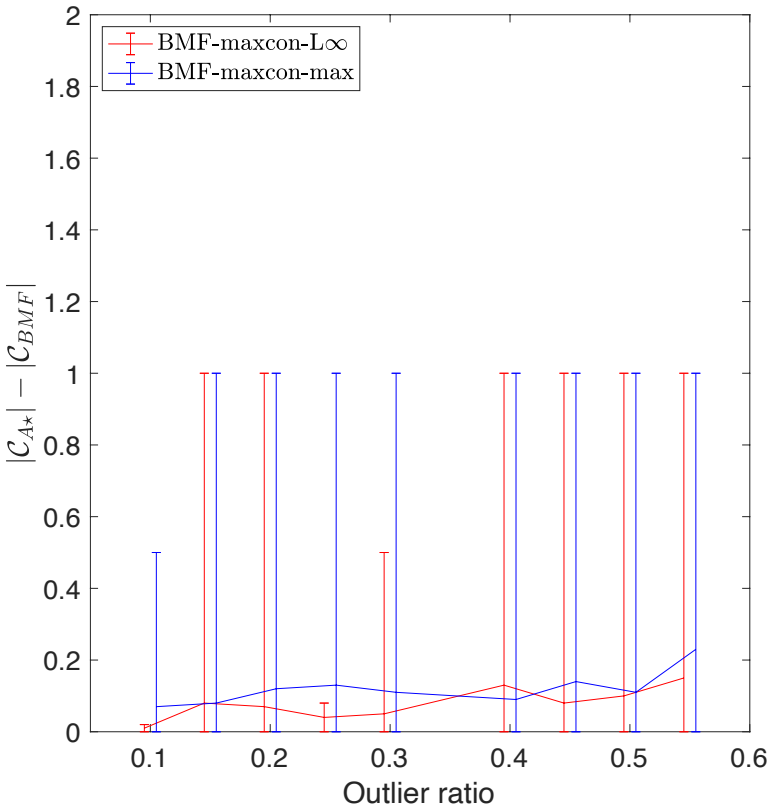
Require: $\{\mathcal{X}_i\}_{i=1}^n$, m, initial feasible set $\mathbf{x}$.

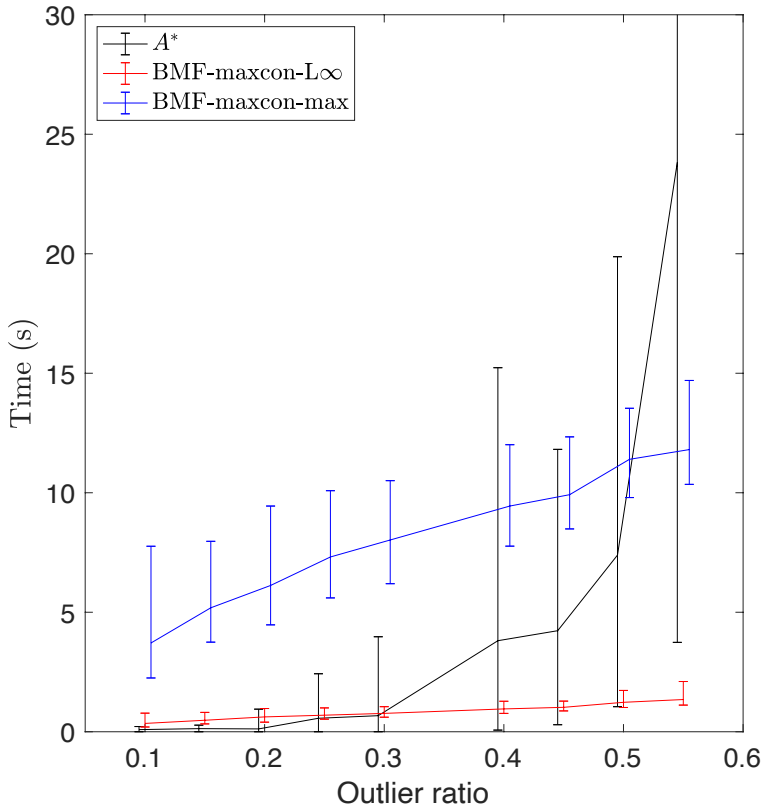
$\bar{b} \leftarrow \{i : x_i = 0\}$
repeat
  solFound = false
  for all $z \in \bar{b}$ **do**
    $\bar{\mathbf{x}} \leftarrow \mathbf{x}$; $\bar{x}_z \leftarrow 1$
    if $f(\bar{\mathbf{x}}) = 0$ **then**
      $\mathbf{x} \leftarrow \bar{\mathbf{x}}$; solFound = true; break;
    end if
  end for
until solFound=true
return $\mathbf{x}$

Re-estimating influence matters!!



OK – show me some results!





“New” results

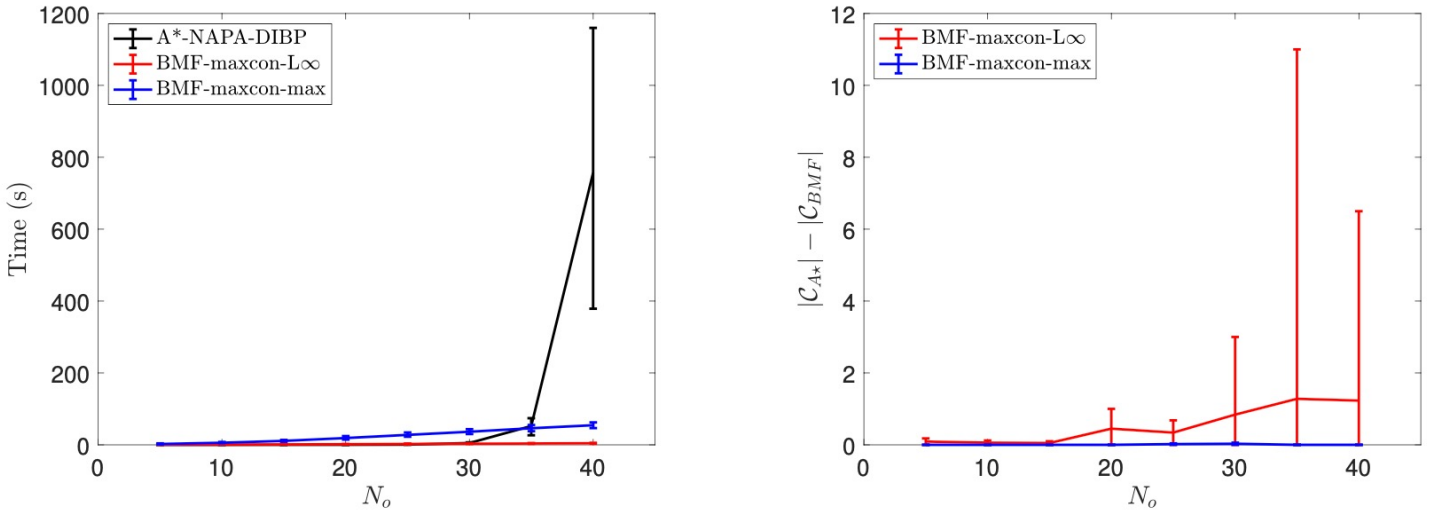


Figure 7: Results for 8 dimensional robust linear regression with synthetic data (a) Variation of computational time with number of outliers and (b) Number of inliers found compared with the global optimal (obtained using A^*). The experiemets were repeated 100 times and the errorbars indicate the 0.05-th and 0.95-th percentile.

"New" results

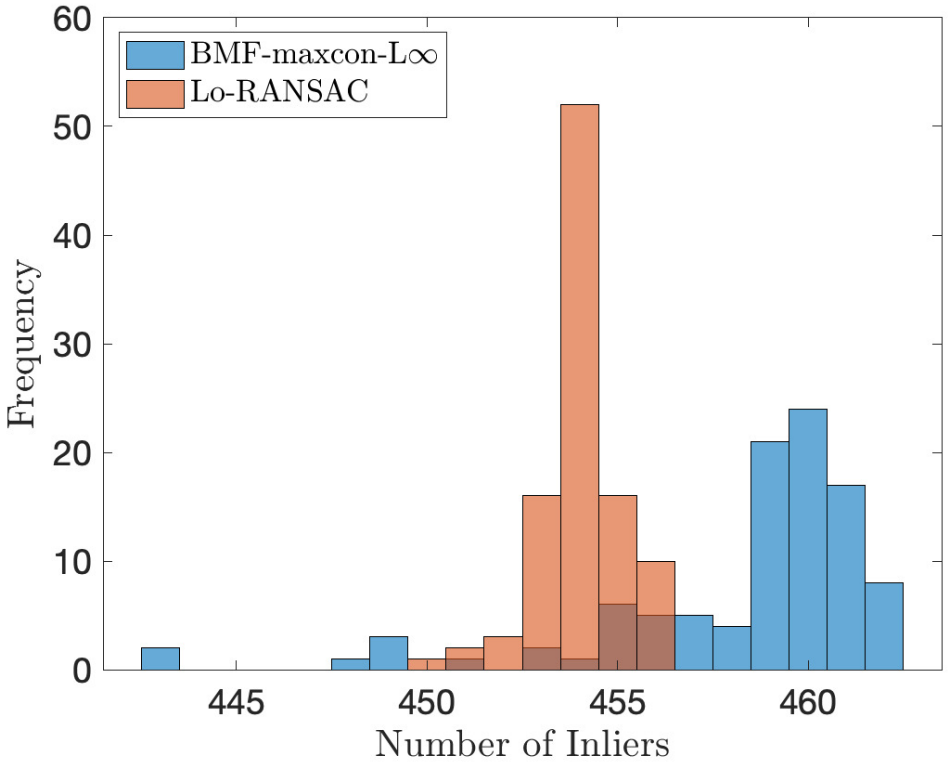
Table 1: Linearized fundamental matrix estimation result.

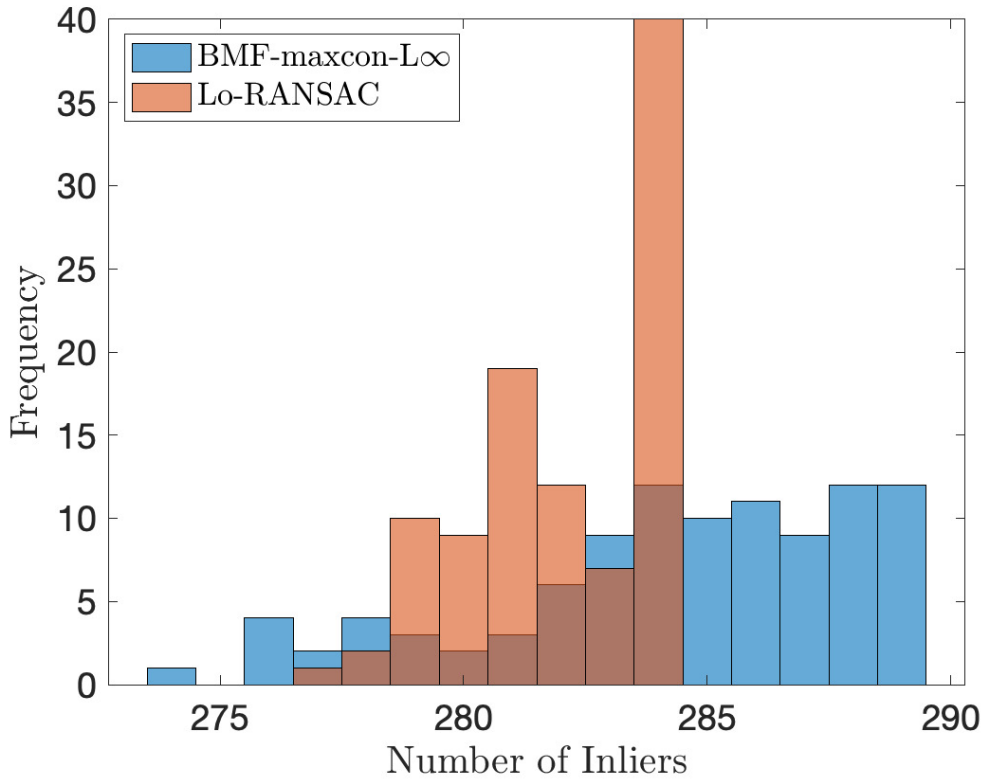
Frame		A*-NAPA DIBP	BMF-maxcon-L∞	BMF-maxcon-max	Lo-RANSAC
104-108	N_i	289	285.10 (274, 289)	283.40 (271, 289)	281.66 (277, 284)
	time (s)	9.627	1.697	9.904	2.21
198-201	N_i	296	291.50 (278, 296)	287.10 (277, 292)	290.3 (288, 292)
	time (s)	6.047	1.739	10.996	2.22
417-420	N_i	366	362.15 (334, 366)	345.90 (286, 365)	360.4 (359, 361)
	time (s)	8.037	2.694	19.341	2.28
579-582	N_i	523	514.00 (498, 523)	523.00 (523, 523)	519.26 (514, 520)
	time (s)	6.942	4.032	24.688	2.26
738-742	N_i	462	458.50 (443, 462)	420.50 (403, 459)	453.9 (450, 456)
	time (s)	4.151	1.884	17.872	2.25

Table 2: Linearized fundamental matrix estimation result for cases with multipl structures.

		A*-NAPA-DIBP	BMF-maxcon-L\infty	Lo-RANSAC
breadcube	N_i	~	63.25 (57, 67)	61.5 (57, 65)
	time (s)	>36000	32.9	31.35
breadtoy	N_i	~	103.25 (96, 111)	103.4 (100, 107)
	time (s)	>36000	29.29	31.04

“New” results





Two examples of Homography Estimation – multiple estimation runs (100)

I’ve covered nowhere near the possible things to investigate….

Property testing

Noise sensitivity/stability

Distance to nearest XXX (Junta, ….well known monotonic functions)

Applications to other LP-type problems and beyond….

What about even more general than LP-type? – violator spaces….

Polymatriods, simplices, discrete topology, sources and sinks (linear orientation of lattices…)

Whole plethora of learning models (so far simple “oracle” type compute cost – not description length, Decision Tree complexity. etc..

But isn’t deep learning the way to go?.....

Look at other work of my student:

Giang Truong, Huu Le, David Suter, Erchuan Zhang, and Syed Zulqarnain Gilani. Unsupervised learning for robust fitting: a reinforcement learning approach. In *CVPR2021*, 2021.