Using Influence to Solve MaxCon (Robust Fitting)

CVPR2021 and CVPR2022 papers

The Idea…

- It started when I saw a paper on Monotone Boolean Functions and asked myself what they were and if they might be relevant to my interests…
- I had also been thinking about the Boolean Cube since my colleagues had the ideas for the CVPR2015 paper (A* search on the Boolean Cube – they called it a tree search)
- I soon realized the the feasibility function was a monotone Boolean function – for reasons covered already
- OK so what might be useful from that theory?

The Idea…

- Influence was mentioned a lot in the MBF literature and I got a hunch that maybe outliers have more influence than inliers…(note term influence already exists in robust fitting literature but it is *different* to influence of Boolean function theory)
- But where to go from there…influence is way too expensive to calculate….
- So what if we approximate it – instead of an exhaustive search/count on the cube…what if we used a smallish number of random samples of the cube? For *monotone* Boolean functions the influences are also the first order fourier coefficients – can we estimate influence more efficiently knowing that? (so far…doesn't seem to help)?
- Can we show *mathematically* - at least for a simple situation, that the hunch is right?

I need help….

- My postdoc – Erchuan Zhang – has a PhD in Maths….we discussed a simple situation and how we might algebraically prove the hunch for that simple situation – the “ideal single structure data” – see later. (He succeeded)
- I invited colleagues from RMIT University Melbourne to visit and asked if they wanted to help….Ruwan Tennakoon got interested in estimating influence and ran a large number of experiments….it sort of worked….but certain non-obvious things had to be done to make it work…see later.
- We rushed off a paper to ECCV2020…it got rejected!
- Improved it a bit – especially the writing…and it got in to CVPR2021 as an oral presentation

First the theory…what is ideal?

- **No outliers**. Yeah, but that's not interesting….

- **Outliers but "ideal single structure"**
- Defined in CVPR2021 (Tennakoon et al.)
- "only one upper zero of monotone Boolean function (0 feasible, 1 infeasible)….. of size larger than trivial…"

What I didn't know then, but know now, is that this means the hypergraph if *infeasible* minimal subsets of data has to be a special type of split graph. The K part (clique of infeasible edges – every outlier forms a clique with every other outlier), S part (independent set of inliers – contain no outliers), maximally connected between K and S part.

I now know this is called (though not studied much) "the complete split (hyper)graph".

Put another way – the outliers have to be ideal – make edges *of infeasibility* with everyone – other outliers AND all inliers… (in the *complement* graph – graph of feasibility – they make NO edges with anyone).

I also now know that this hypergraph is {0,1}*-constructible hypergraph with construction string 00…011…1 (every inlier added as isolated vertices then every outlier added to the graph in a completely connected hypergraph).

What Erchuan Zhang proved….

Theorem 3.1. If a monotone Boolean function f is ideal with respect to $\mathbf{x}^k \in L_k$, then

$$\mathrm{Inf}_i[f] = \begin{cases} \frac{C_p^{n-1} - C_p^{k-1}}{2^n} & \text{if } i \text{ is inlier} \\ \frac{C_p^{n-1} + \sum_{l=p+1}^{k} C_l^k}{2^n} & \text{if } i \text{ is outlier} \end{cases} \tag{6}$$

Corollary 3.1.1.

$$\mathrm{Inf}_j[f] - \mathrm{Inf}_i[f] = \frac{1}{2^n} \sum_{l=p+1}^{k} C_l^k + C_p^{k-1} > 0 \tag{7}$$

where i is any inlier and j is any outlier.

In fact, he also derived an expression for the influences in ANY situation – more than one (what we know would call…maximum faces of the independence complex) BUT it is very hard to draw any conclusions from the formula… So….we had shown the hunch was correct BUT only for this VERY special case…which almost no data would satisfy.

So we had to appeal to "continuity" – for data "close" to ideal – in some sense – surely the influences of outliers is still larger than inliers? (????)

What Ruwan Tennakoon discovered about *estimating influences*

- The naïve approach of “single step” estimating all influences, sorting them and looking for a gap (we don’t know how many outliers there are!) doesn’t work too well. (Actually, Tat-Jun Chin – who I had also invited to visit and think about my idea – and a student of his *did* show they could do this for *some* data…but my hunch is they “got lucky or spent way too much effort on estimating the influences – after all, their paper was mainly on how this could sit within a quantum computing approach and weren’t so concerned with traditional computing costs…ironically, this paper became the first published – ACCV2020 – on the idea of using MBF theory/influences of MaxCon – in between the rejected ECCV2020 paper and the CVPR2021 paper).

So what did work reasonably well?

- Firstly, adopt an “estimate – eliminate highest influence data point, test if remaining data is feasible if not *re-estimate influences on the remaining data*” approach. We justified this by firstly not wanting have the difficult task of finding a gap (possibly very small/noisy), but also because one wants to use relatively few samples to estimate – and so the noise in the estimates is larger but likely the highest influence outlier will still be high if not the highest in the rough estimates.

  (I now know other possible reasons!)
- Secondly, don’t sample over the whole Boolean cube…this is likely very wasteful (if the number of inliers is not close to the number of data then most of the upper part of the cube is infeasible – no transitions from feasible to infeasible or visa versa. So sample around p+1, p+2, p+3….in subset size. This we saw as biased, but perhaps more efficient estimates of the “true” influences.

So what did work reasonably well?

- Thirdly, since our estimates are possibly wrong enough for us to "miss" the "top" face of the independence complex (the MaxCon solution) – once the first stage of iteratively removing the highest estimate influence data point, we attempt a "greedy" recovery – look at each excluded data point one by one and see if adding it back in leaves the data still feasible.
- LOTS of "fine tuning" of the idea – how many samples to use in the influence estimation, what level of the cube to bias the sampling towards etc. etc.

Results?

The A* search (actually and improved version of the CVPR2015 version) always gives the right answer if it terminates…but look at the runtime for large number of outliers…
Popular methods – Ransac/Loransac give reasonable results but not as good – even when run for the same amount of time!

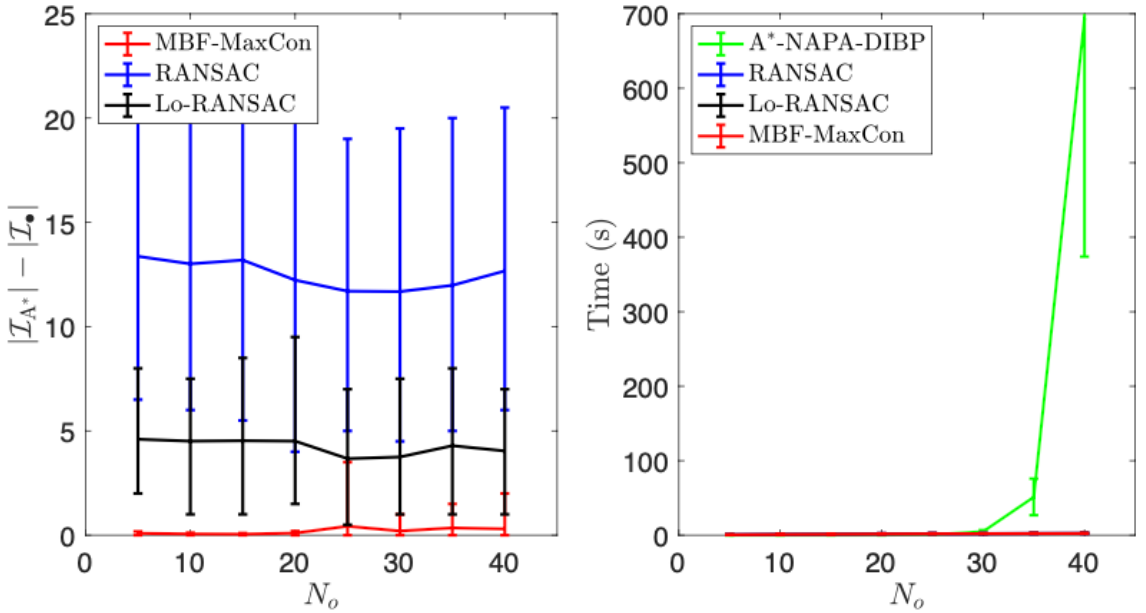

Figure 6: Results for 8 dimensional robust linear regression with synthetic data (a) Number of inliers found compared with the global optimal (obtained using A^*) and (b) Variation of computational time with number of outliers. The experiments were repeated 100 times and the error-bars indicate the 0.05-th and 0.95-th percentile.

Real data – real problems?

Table 1: Linearized fundamental matrix estimation result. "Same Comp." refers to running RANSAC with the same time budget as MBF-MaxCon and, "sp=0.99" refers to running RANSAC with the success probability 0.99.

		A*-NAPA -DIBP	MBF-MaxCon	RANSAC Same Comp.	Lo-RANSAC Same Comp.	RANSAC sp=0.99	Lo-RANSAC sp=0.99
104-108	n_i	289	288.52 (289, 285)	282.03 (286, 277)	284.54 (287, 283)	271.18 (282, 254)	281.41 (284, 276)
	Time (s)	10.96	1.78	1.78	1.78	0.004	0.04
198-201	n_i	296	293.10 (296, 291)	291.88 (294, 290)	292.87 (294, 291)	287.00 (293, 272)	290.35 (293, 287)
	Time (s)	4.04	2.05	2.05	2.05	0.004	0.04
417-420	n_i	366	364.44 (366, 359)	363.54 (365, 361)	364.02 (365, 363)	357.38 (364, 343)	362.33 (364, 359)
	Time (s)	7.93	2.59	2.59	2.59	0.004	0.06
579-582	n_i	523	520.68 (523, 514)	518.04 (521, 511)	520.89 (522, 520)	502.31 (519, 463)	517.30 (512, 497)
	Time (s)	4.28	3.22	3.22	3.22	0.004	0.11
738-742	n_i	462	460.97 (462, 457)	455.02 (459, 451)	457.24 (460, 455)	438.55 (455, 409)	451.35 (459, 435)
	Time (s)	2.97	2.72	2.72	2.72	0.005	0.1
breadcube	n_i	$\sim$	64.36 (68, 58)	61.77 (65, 59)	63.66 (66, 61)	59.04 (65, 55)	62.17 (66, 59)
	Time (s)	>3600	19.07	19.07	19.07	0.993	1.07
breadtoy	n_i	$\sim$	107.48 (115, 102)	105.24 (111, 102)	107.23 (111, 104)	100.31 (107, 93)	105.21 (109, 101)
	Time (s)	>3600	38.37	38.37	38.37	0.526	0.68
cubetoy	n_i	$\sim$	58.51 (61, 52)	54.54 (58, 52)	56.14 (58, 55)	52.20 (56, 48)	55.24 (57, 52)
	Time (s)	>3600	15.38	15.38	15.38	0.45	0.65

Real data – real problems…

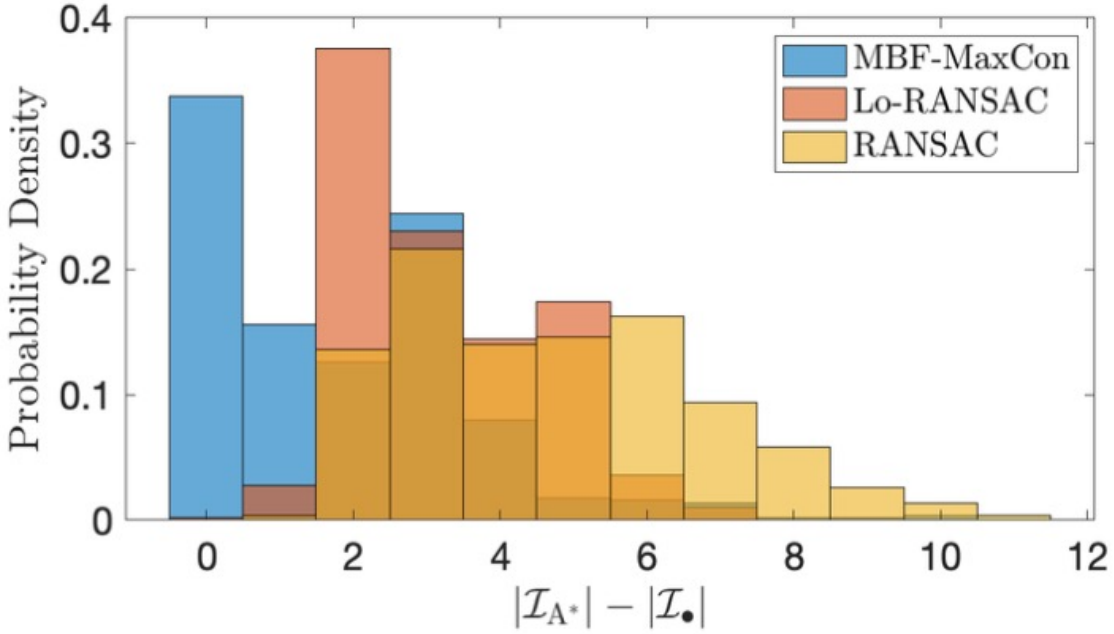

Figure 7: The distribution of errors by each algorithm over 100 repeated runs for all the frames in sequence "00" of the KITTI Odometry data set.

Summary….

- The method works reasonably well
- Unfortunately, it is not as fast as the popular methods (and would give inferior results if run for such short times as those methods would usually run for…
- But the popular methods and this method give better and better results the longer they run for
- This method eventually (if you can afford the time) give BETTER results than the popular methods when they run for that same time
- The method doesn't suffer the sudden exponential growth of cost that can happen for the known best (but not generally fast methods – our A* and a few others such as Branch and Bound), when the number of outliers increases

But what about the second paper (CVPR2022)?

- That was mainly theoretical
- It studied biased sampling estimation of influence and showed for the ideal single structure the hunch was also right in biased sampling approaches (Bernoulli sampling) – and that one could get small gains by doing this “right”.