Computing Plurality Points and Condorcet Points in Euclidean Space

Philipp Gabler

Outline

1. Informal introduction
2. Notation
3. Idea of algorithm, neccessary lemmata by case
4. Algorithm
5. Analysis of algorithm

Informal description

We can represent opinion standpoints by points in space. There are several ways define a (spacial) equilibrium between these quantified “opinions”.

- Voters: Multiset in Euclidean space ($\mathbb{R}^d$ with ℓ_2 norm)
- Plurality point: Closer to at least as many voters as any other point
- Condorcet point: No other point is closer to an absolute majority of voters

Problem: given the voters, find plurality and Condorcet points.

Notation

Definition (Voters)

Voters: $V = \{v_1, \ldots, v_n\} \subseteq \mathbb{R}^d$ (a multiset, in general). The components are denoted as $v_i = (v_i^{(1)}, \ldots, v_i^{(d)})$.

Definition (Multidimensional medians)

Informally: at least half the voters are on each side, per dimension:

$$\mathcal{M}_V = \{\theta : |\{v \in V : \forall j : v^{(j)} \leq \theta^{(j)}\}| \geq n/2 \\ \wedge |\{v \in V : \forall j : v^{(j)} \geq \theta^{(j)}\}| \geq n/2\}$$

Notation

Definition (Median corners)

Let $V^{(i)}$ be the i-th coordinate of points in V, and $x_l^{(i)}$ and $x_h^{(i)}$ denote the $\lceil n/2 \rceil$-th and $\lceil (n+1)/2 \rceil$-th smallest numbers in $V^{(i)}$. Then the set

$$C = \{(x_{t_1}^{(i)}, \ldots, x_{t_d}^{(i)}) : t_1, \ldots, t_d \in \{l, h\}\}$$

collects all corners of $\mathcal{M}_V$. Clearly, $|C| \leq 2^d$.

Definition (Half spaces)

Given a line L, we can partition $\mathbb{R}^d = L^+ \cup L \cup L^-$, where L^+ and L^- are the open half spaces separated by L. We define:

$$V_L = L \cap V, \quad V_L^+ = L^+ \cap V, \quad V_L^- = L^- \cap V$$
$$n_L = |V_L|, \quad n_L^+ = |V_L^+|, \quad n_L^- = |V_L^-|$$

Formal definition

Definition (Preference)

- v_i prefers θ_1 to θ_2 if $d(v_i, \theta_1) < d(v_i, \theta_2)$
- $[\theta_1 \succ \theta_2] := \{v \in V : d(v, \theta_1) < d(v, \theta_2)\}$, and accordingly for other relations

Definition (Plurality point (PP))

$\Delta \in \mathbb{R}^d$ is a plurality point iff

$$\forall \theta \in \mathbb{R}^d : |[\theta \succ \Delta]| \leq |[\Delta \succ \theta]|$$

Definition (Condorcet point (CP))

$\Delta \in \mathbb{R}^d$ is a Condorcet point iff

$$\forall \theta \in \mathbb{R}^d : |[\theta \succ \Delta]| \leq n/2$$

Equivalence of PPs and CPs

Lemma (Equivalence)

In $\mathbb{R}^d$, a point is a plurality point iff it is a Condorcet point.

Proof.

1. A PP is a CP by definition.
2.
   - Assume Δ is a CP, but not a PP, so $\exists \delta : |[\Delta \succ \delta]| < |[\delta \succ \Delta]|$.
   - Let m be the midpoint of Δ, δ. Then for each $v \in [\delta \succeq \Delta]$, $d(v, m) < d(v, \Delta)$.
   - Thus, $|[m \succ \Delta]| > n/2$, which is a contradiction.
   - Therefore, a CP is also a PP.

From now on, we constrain ourselves to PPs in $\mathbb{R}^2$.

Idea behind algorithm

The presented algorithm works by successively cutting down the space of possible PPs via a series of case distinctions, until only $\mathcal{O}(1)$ candidate points remain, which can be checked exhaustively. For this purpose, the following series of lemmata is employed.

Furthermore:

Definition

Δ_V denotes the set of all PPs of V

Collinear case

Lemma (Collinear points)

If all voters in V are collinear, then

1. $\Delta_V = \mathcal{M}_V$, *and*
2. $|\Delta_V| \geq 1$.

Proof.

1. has been shown by Hansen & Thisse (1981) for CCs in $\mathbb{R}^2$. By the equivalence lemma, this holds also for PPs.
2. By definition, $\mathcal{M}_V \neq \emptyset$. When $|V|$ is even, we may have $|\mathcal{M}_V| = |\Delta_V| > 1$.

"Tukey depth condition"

This neccessary and sufficient condition is used to decide whether a candidate point is a valid PP.

Lemma

In $\mathbb{R}^2$, Δ is a plurality point iff for any line L through Δ, $n_L^+ \leq n/2$ and $n_L^- \leq n/2$.

It is equivalent checking if

$$\min_{L \in \mathcal{L}_\Delta} \{|V \cup \gamma| : \gamma \text{ is a closed halfspace separated by } L\} \geq n/2,$$

where $\mathcal{L}_\Delta$ is the space of lines through Δ. The quantity on the left is called *Tukey depth* of Δ with respect to V.

Non-collinear case & medians

Lemma (Non-collinear candidates)

If not all voters in V are collinear, then

1. $\Delta_V \subseteq \mathcal{M}_V$, *and*
2. $|\Delta_V| = 0$, *or* $|\Delta_V| = 1$.

So, a plurality point must be a median, but not vice versa. We can use this to cut down the searched space to the median; however, this is still infinite.

Δ is in $\mathcal{M}_V$

These lemmata reduce the possibilities in the non-collinear case further to $\mathcal{O}(1)$ candidate points.

Lemma ($\Delta \in V$)

If Δ is a PP of V, and $\Delta \in V$, then $\Delta \in C$, with $V \cup C \leq 2^d$.

Lemma ($\Delta \notin V$)

If Δ is a PP of V, $\Delta \notin V$, and L is a separating line through Δ (ie., $n_L = 0$), then Δ is the intersection of the internal tangents of the convex hulls of V_L^+ and V_L^-.

Subroutines

- SELECT(S, n) returns the n-th smallest point in S. This can be calculated in $\mathcal{O}(|S|)$ time by the median-of-medians selection algorithm from Cormen et. al. Used for median calculation.
- VERIFYCANDIDATES(C, V) returns all points from C whose Tukey depth in is greater or equal $|V|/2$ (ie., satisfying the condition in the "Tukey depth lemma" above). Tukey depth of one point can be computed in $\mathcal{O}(|V| \log |V|)$ time with an algorithm by Rousseeuw and Struyf (1998).

Subroutines

- InternalTangentIntersection(V, W) returns the intersection of the internal tangents between the convex hulls of V and W. The tangents can be found by solving the linear program

$$\max_{k,d} F(k,d) = k, \quad \text{s.t.}$$
$$\forall\, v \in V : v_y \leq k v_x + d \quad \text{and}$$
$$\forall\, w \in W : w_y \geq k w_x + d,$$

and the opposite formulation (minimizing with V and W swapped). Linear programming with fixed dimension can be done in linear time in the number of constraints, by Megiddo (1984).

Pseudocode

```
1:  procedure PluralityPoint2D(V)
2:      if all voters are collinear then
3:          return medians of V
4:      else                                   ▷ Corners of medians
5:          x_h = Select(V_x, ⌈(n + 1)/2⌉)
6:          x_l = Select(V_x, ⌈n/2⌉)
7:          y_h = Select(V_y, ⌈(n + 1)/2⌉)
8:          y_l = Select(V_y, ⌈n/2⌉)
9:          C = {(x_h, y_h), (x_h, y_l), (x_l, y_h), (x_l, y_l)}

10:         if |C| = 1 then ▷ n odd, or C degenerate
11:             return VerifyCandidates(C, V)
12:         else
13:             P = VerifyCandidates(V ∩ C, V)
14:             if P ≠ ∅ then                  ▷ Δ ∈ V
15:                 return P
16:             else                           ▷ Δ ∉ V
17:                 if x_h = x_l then
18:                     V_a = {v ∈ V : v^(2) ≤ y_h}
19:                     V_b = {v ∈ V : v^(2) ≥ y_h}
20:                 else
21:                     V_a = {v ∈ V : v^(1) ≤ x_h}
22:                     V_b = {v ∈ V : v^(1) ≥ x_h}
23:                 end if

24:                 p = InternalTangentIntersection(V_a, V_b)
25:                 return VerifyCandidates({p}, V)
26:             end if
27:         end if
28:     end if
29: end procedure
```

Analysis

- Detecting collinearity (line 2): $\mathcal{O}(n)$
- Selecting median corners (lines 5–8): $\mathcal{O}(n)$
- Verifying candidates by calculating Tukey depth (lines 11, 13, 25): $\mathcal{O}(n \log n)$
- Computing internal tangent intersection (line 24): $\mathcal{O}(n)$
- Therefore: $\mathcal{O}(n \log n)$ overall complexity

Conclusion

- In Euclidean space, plurality points and Condorcet points are equivalent notions
- In $\mathbb{R}^2$, they can be calculated in $\mathcal{O}(n \log n)$ time, using the presented algorithm
- This can be generalized to $\mathbb{R}^d$ in $\mathcal{O}(n^{d-1} \log n)$ time

Working implementation (in Julia)

`https://github.com/phipsgabler/plurality-points`

Bibliography


Nimrod Megiddo. “Linear programming in linear time when the dimension is fixed”. In: *Journal of the ACM (JACM)* 31.1 (1984), pp. 114–127. DOI: 10.1145/2422.322418.


Peter J. Rousseeuw and Ida Ruts. “Bivariate Location Depth”. In: *Applied Statistics* 45 (1996), pp. 516–526. DOI: 10.2307/2986073.


Yen-Wei Wu et al. “Computing Plurality Points and Condorcet Points in Euclidean Space”. In: *International Symposium on Algorithms and Computation*. Springer, 2013, pp. 688–698. DOI: 10.1007/978-3-642-45030-3˙64.