Computing Sparse Subsets of the Delaunay Triangulation in High-Dimensions for Interpolation and Graph Problems

T.H. Chang[a], L.T. Watson[b], and T.C.H. Lux[b]

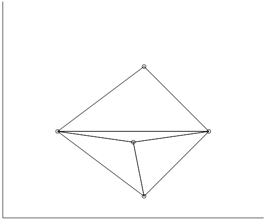
[a]Argonne National Laboratory
[b]Virginia Polytechnic Institute and State University

March, 2021

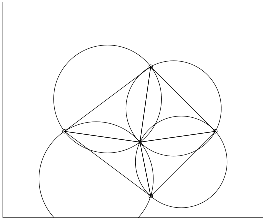
Outlines

About Delaunay Triangulations

- The *Delaunay triangulation* is an unstructured simplicial mesh defined by an arbitrary vertex set $P = \{x^{(1)}, \dots, x^{(n)}\} \subset \mathbb{R}^d$
- The defining property of the Delaunay triangulation $DT(P)$ is that for every simplex $S \in DT(P)$, the circumball $B^{(S)}$ must have empty intersection with P: $B^{(S)} \cap P = \emptyset$.



✗



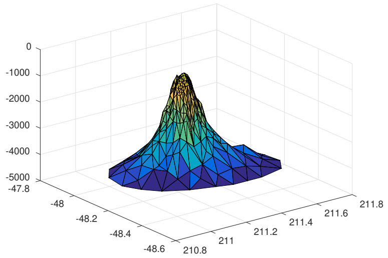
✓

- $DT(P)$ exists and is unique when P is in *general position*.

Delaunay Interpolation

Let $y \in S \in DT(P)$. S has vertex set $\{s^{(1)}, \ldots, s^{(d+1)}\}$ and there exist convex weights $\{w_1, \ldots, w_{d+1}\}$ such that $y = \sum_{i=1}^{d+1} w_i s^{(i)}$.

$$\hat{F}_{DT}(y) = \sum_{i=1}^{d+1} w_i F(s^{(i)}).$$



Advantages in data science/ML settings

- Interpolates data (when that is a desirable property)
- Like a feed-forward fully-connected ReLU net, this is a piecewise linear model. But this model is provably considered "optimal" (in a sense) for interpolation w.r.t. other piecewise linear models.
- Can be used to interpolate functional response variables, e.g., PDFs.

Scalability Issues

- Oweing to Klee, the size of the Delaunay triangulation is

$$\mathcal{O}\left(n^{\lceil d/2 \rceil}\right)$$

  - For $d > 4$, this is expensive!
  - For $d > 8$, this is not scalable!

Scalability Issues

- Oweing to Klee, the size of the Delaunay triangulation is

$$\mathcal{O}\left(n^{\lceil d/2 \rceil}\right)$$

  - For $d > 4$, this is expensive!
  - For $d > 8$, this is not scalable!

Observation: For interpolation at a single point y, we only need the vertices $(\{s^{(1)}, \dots, s^{(d+1)}\})$ of $S \in DT(P)$ such that $y \in S$

$$\hat{F}_{DT}(y) = \sum_{i=1}^{d+1} w_i \; F(s^{(i)}).$$

Scalability Issues

- Oweing to Klee, the size of the Delaunay triangulation is

$$\mathcal{O}\left(n^{\lceil d/2 \rceil}\right)$$

  - For $d > 4$, this is expensive!
  - For $d > 8$, this is not scalable!

Observation: For interpolation at a single point y, we only need the vertices $(\{s^{(1)}, \ldots, s^{(d+1)}\})$ of $S \in DT(P)$ such that $y \in S$

$$\hat{F}_{DT}(y) = \sum_{i=1}^{d+1} w_i \; F(s^{(i)}).$$

Question: Can we find S containing y in polynomial time?

Algorithm outline

Algorithm to locate Delaunay simplex containing y:

- Grow an initial Delaunay simplex (greedy algorithm) that is "nearby" to y
- "Flip" accross facets from which y is visible to a new Delaunay simplex (closer to y)
- This "visibility walk" converges to y in finite steps (Edelsbrunner's acyclicity theorem)

Chang, Watson, Lux, Li, Xu, Butt, Cameron, and Hong. "A polynomial time algorithm for multivariate interpolation in arbitrary dimension via the Delaunay triangulation." In Proc. 2018 ACMSE Conf.

Growing the First Simplex

ϕ is the vertex set for the initial Delaunay simplex:

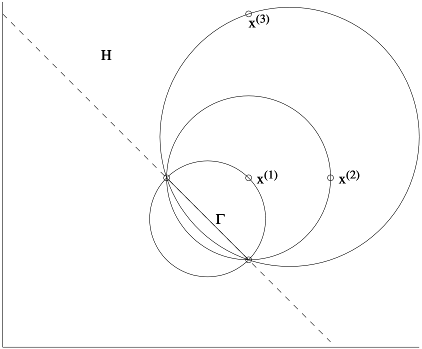
- Start with ϕ containing just the nearest neighbor to y in P;
- For all $x \in P \setminus \phi$, compute the radius r_{min} of the smallest circumball about $\{x\} \cup \phi$ and select the x^* that minimizes r_{min};
- $\phi \leftarrow \phi \cup \{x^*\}$;
- Repeat until $|\phi| = d + 1$;

Lemma

Let P be in general position, and let Γ be a Delaunay j-face with vertices $\phi \subset P$ where $j < d$. Let $x^* \in P \setminus \phi$ minimize the radius of the smallest $(d-1)$-sphere through the points in $\phi \cup \{x\}$, over all $x \in P \setminus \phi$. Then $\Gamma^* = ConvexHull(\phi \cup \{x^*\})$ is a Delaunay $(j+1)$-face. **Proof in Paper.**

Flipping

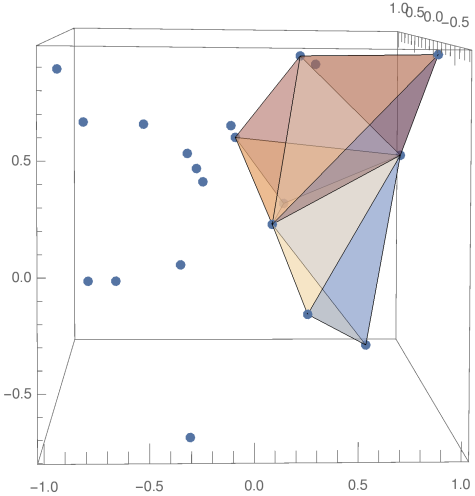
- Let ϕ be the vertices for a facet of a Delaunay simplex;
- Let Γ be the facet with vertices in ϕ;
- Let H be the halfspace containing y, w.r.t. the hyperplane containing Γ
- Unless Γ is a facet of the convex hull, there exists $x^* \in P \setminus \phi$ such that $\phi \cup \{x^*\}$ is the vertex set for a Delaunay simplex;



Proof in Paper.

Visibility walk

- Grow an initial simplex $S^{(0)}$;
- While $y \notin S^{(k)}$, generate $S^{(k+1)}$ by flipping across a facet of $S^{(k)}$ from which y is "visible";
- Terminate when $y \in S^{(k)}$, otherwise, $k \leftarrow k+1$;



Converges in finite flips by *Edelsbrunner's Acyclicity Theorem*.

Algorithm Complexity

- To grow the first simplex: $\mathcal{O}(nd^3)$ to apply n rank-1 updates to the QR factorization of $d \times j$ matrix for $j = 1, \ldots, d$
- To compute a flip: $\mathcal{O}(nd^2)$ to apply n rank-1 updates to the QR factorization of a $d \times d$ matrix
- ℓ total flips

	$n = 2K$	$n = 8K$	$n = 16K$	$n = 32K$
$d = 2$	3.05	2.90	3.25	3.10
$d = 8$	23.75	24.75	24.30	23.10
$d = 32$	95.25	125.60	131.85	150.10
$d = 64$	171.95	221.85	248.35	280.60

Algorithm Complexity

- To grow the first simplex: $\mathcal{O}(nd^3)$ to apply n rank-1 updates to the QR factorization of $d \times j$ matrix for $j = 1, \ldots, d$
- To compute a flip: $\mathcal{O}(nd^2)$ to apply n rank-1 updates to the QR factorization of a $d \times d$ matrix
- ℓ total flips

	$n = 2K$	$n = 8K$	$n = 16K$	$n = 32K$
$d = 2$	3.05	2.90	3.25	3.10
$d = 8$	23.75	24.75	24.30	23.10
$d = 32$	95.25	125.60	131.85	150.10
$d = 64$	171.95	221.85	248.35	280.60

Overall complexity: $\mathcal{O}(nd^2\ell)$

Algorithm Complexity

- To grow the first simplex: $\mathcal{O}(nd^3)$ to apply n rank-1 updates to the QR factorization of $d \times j$ matrix for $j = 1, \ldots, d$
- To compute a flip: $\mathcal{O}(nd^2)$ to apply n rank-1 updates to the QR factorization of a $d \times d$ matrix
- ℓ total flips

	$n = 2K$	$n = 8K$	$n = 16K$	$n = 32K$
$d = 2$	3.05	2.90	3.25	3.10
$d = 8$	23.75	24.75	24.30	23.10
$d = 32$	95.25	125.60	131.85	150.10
$d = 64$	171.95	221.85	248.35	280.60

Overall complexity: $\mathcal{O}(nd^2\ell)$

Unresolved question: $\ell \approx d$? ℓ independent of n?

Linear programming interpretation

$$\tilde{A} = \begin{bmatrix} (-x^{(1)})^T & 1 \\ (-x^{(2)})^T & 1 \\ \vdots & \vdots \\ (-x^{(n)})^T & 1 \end{bmatrix}, \; \tilde{b} = \begin{bmatrix} \|x^{(1)}\|_2^2 \\ \|x^{(2)}\|_2^2 \\ \vdots \\ \|x^{(n)}\|_2^2 \end{bmatrix}, \text{ and } \tilde{c} = \begin{bmatrix} -y \\ 1 \end{bmatrix}.$$

Linear programming interpretation

$$\tilde{A} = \begin{bmatrix} (-x^{(1)})^T & 1 \\ (-x^{(2)})^T & 1 \\ \vdots & \vdots \\ (-x^{(n)})^T & 1 \end{bmatrix}, \ \tilde{b} = \begin{bmatrix} \|x^{(1)}\|_2^2 \\ \|x^{(2)}\|_2^2 \\ \vdots \\ \|x^{(n)}\|_2^2 \end{bmatrix}, \text{ and } \tilde{c} = \begin{bmatrix} -y \\ 1 \end{bmatrix}.$$

Primal prob: $\max_{\tilde{u}} \tilde{c}^T \tilde{u}$ such that $\tilde{A}\tilde{u} \leq \tilde{b}, \tilde{u}$ free.

Ext pts: $\tilde{u} = (-2\text{circumcenter}, \text{circumradius}^2 - \|\text{circumcenter}\|_2^2)$

Linear programming interpretation

$$\tilde{A} = \begin{bmatrix} (-x^{(1)})^T & 1 \\ (-x^{(2)})^T & 1 \\ \vdots & \vdots \\ (-x^{(n)})^T & 1 \end{bmatrix}, \ \tilde{b} = \begin{bmatrix} \|x^{(1)}\|_2^2 \\ \|x^{(2)}\|_2^2 \\ \vdots \\ \|x^{(n)}\|_2^2 \end{bmatrix}, \text{ and } \tilde{c} = \begin{bmatrix} -y \\ 1 \end{bmatrix}.$$

Primal prob: $\max_{\tilde{u}} \tilde{c}^T \tilde{u}$ such that $\tilde{A}\tilde{u} \leq \tilde{b}, \tilde{u}$ free.

Ext pts: $\tilde{u} = (-2\text{circumcenter}, \text{circumradius}^2 - \|\text{circumcenter}\|_2^2)$

Dual prob: $\min_{\tilde{v}} \tilde{b}^T \tilde{v}$ such that $\tilde{A}^T \tilde{v} = \tilde{c}, \tilde{v} \geq 0.$

Ext pts: $\Rightarrow$ $\tilde{v}$ are convex weights for y

Linear programming interpretation

$$\tilde{A} = \begin{bmatrix} (-x^{(1)})^T & 1 \\ (-x^{(2)})^T & 1 \\ \vdots & \vdots \\ (-x^{(n)})^T & 1 \end{bmatrix}, \ \tilde{b} = \begin{bmatrix} \|x^{(1)}\|_2^2 \\ \|x^{(2)}\|_2^2 \\ \vdots \\ \|x^{(n)}\|_2^2 \end{bmatrix}, \text{ and } \tilde{c} = \begin{bmatrix} -y \\ 1 \end{bmatrix}.$$

Primal prob: $\max_{\tilde{u}} \tilde{c}^T \tilde{u}$ such that $\tilde{A}\tilde{u} \leq \tilde{b}, \tilde{u}$ free.

Ext pts: $\tilde{u} = (-2\text{circumcenter}, \text{circumradius}^2 - \|\text{circumcenter}\|_2^2)$

Dual prob: $\min_{\tilde{v}} \tilde{b}^T \tilde{v}$ such that $\tilde{A}^T \tilde{v} = \tilde{c}, \tilde{v} \geq 0.$

Ext pts: $\Rightarrow \tilde{v}$ are convex weights for y

Primal $+$ dual feasible $\Rightarrow$ Delaunay simplex containing y

Linear programming interpretation

$$\tilde{A} = \begin{bmatrix} (-x^{(1)})^T & 1 \\ (-x^{(2)})^T & 1 \\ \vdots & \vdots \\ (-x^{(n)})^T & 1 \end{bmatrix}, \tilde{b} = \begin{bmatrix} \|x^{(1)}\|_2^2 \\ \|x^{(2)}\|_2^2 \\ \vdots \\ \|x^{(n)}\|_2^2 \end{bmatrix}, \text{ and } \tilde{c} = \begin{bmatrix} -y \\ 1 \end{bmatrix}.$$

Primal prob: $\max_{\tilde{u}} \tilde{c}^T \tilde{u}$ such that $\tilde{A}\tilde{u} \leq \tilde{b}, \tilde{u}$ free.

Ext pts: $\tilde{u} = (-2\text{circumcenter}, \text{circumradius}^2 - \|\text{circumcenter}\|_2^2)$

Dual prob: $\min_{\tilde{v}} \tilde{b}^T \tilde{v}$ such that $\tilde{A}^T \tilde{v} = \tilde{c}, \tilde{v} \geq 0$.

Ext pts: $\Rightarrow$ $\tilde{v}$ are convex weights for y

Primal + dual feasible $\Rightarrow$ Delaunay simplex containing y

LP basic solution in polynomial time is an open problem!

DELAUNAYSPARSE Package

Standalone software package `DELAUNAYSPARSE`:

- Robust against degeneracy
- Runs in $\mathcal{O}(mnd^2\ell)$ time
- Parallel and serial implementations

Runtime (secs) for interpolating a single point ($m = 1$) with n pts in $\mathbb{R}^d$

	d				
n	2	8	32	64	128
250	0.005	0.013	0.150	3.404	27.078
500	0.021	0.042	0.325	6.479	59.511
1000	0.083	0.152	0.791	14.020	124.320
2000	0.344	0.583	2.230	28.984	242.066
4000	1.314	2.284	7.165	62.494	502.620
8000	5.580	9.027	26.210	151.177	905.711
16,000	22.086	35.725	109.448	386.596	2190.362
32,000	82.915	145.115	421.934	1097.060	5024.675

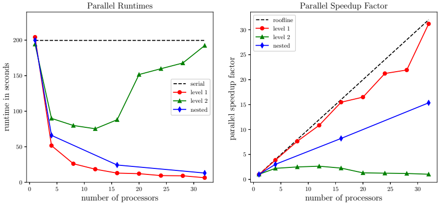
Chang, Watson, Lux, Butt, Cameron, and Hong. 2020. Algorithm 1012: DELAUNAYSPARSE: Interpolation via a sparse subset of the Delaunay triangulation in medium to high dimensions. ACM Trans. Math. Softw. 46(4).

Parallel implementation

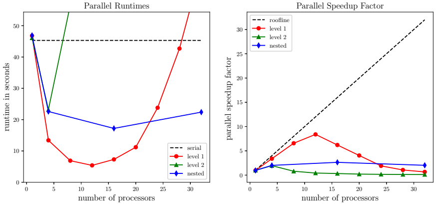
Distributed memory: Run the serial algorithm on small batches of interpolation pts

Shared memory: Multiple levels

- Level 1: loop over multiple interpolation points (like distributed case)
- Level 2: loop(s) over data points – results in additional work; preferrable only when m is small and n is large



$d = 50,\ n = 500,\ m = 64$



$d = 10,\ n = 1000,\ m = 1024$

The Delaunay Graph

- Delaunay graph of $P = DG(P)$
- Connect 2 vertices iff they are shared by a single Delaunay simplex
- Used for:
  - Neighbor structure in spatial data
  - Topological shape analysis
- There are at most $n(n-1)/2$ edges
- Current state-of-the-art implementation in CGAL computes $DG(P)$ from $DT(P)$ – scales well for large n, infeasible for $d \geq 10$

Getting the Delaunay Graph

- The number of connections in $DG(P)$ is upper bounded by $n(n-1)/2$
- Can recover $DG(P)$ by interpolating the midpoint between each pair of points in P
  - If the simplex containing the midpoint between $x^{(1)}$ and $x^{(2)}$ also contains both $x^{(1)}$ and $x^{(2)}$, then they are clearly connected
  - If not, then it certifies that they are not connected in *a* Delaunay triangulation (in case degenerate)
- Using `DELAUNAYSPARSE`, requires $\mathcal{O}(n^3 d^2 \ell)$ time — better than current state-of-the-art for d large, worse for n large

Implementation currently under review for publication.

Full proof/description in

T.H. Chang. Mathematical Software for Multiobjective Optimization Problems. Ph.D. Thesis, Virginia Tech, 2020.

Questions

This material is based upon work supported by the U.S. Dept. of Energy, Office of Science, Office of Advanced Scientific Computing Research, SciDAC program under contract number DE-AC02-06CH11357, and the Office of Science Graduate Student Research (SCGSR) program. The SCGSR program is administered by the Oak Ridge Institute for Science and Education (ORISE), which is managed by ORAU under contract number DE-SC0014664. All opinions in this paper are the authors' and do not necessarily reflect the policies and views of the DOE, ORAU, or ORISE.

This work was also supported in part by NSF Grants CNS-1565314 and CNS-1838271.