Algorithms and Software for Delaunay Interpolation and Multiobjective Optimization

Tyler H. Chang

Dept. of Computer Science
Virginia Polytechnic Institute and State University

February 16, 2020

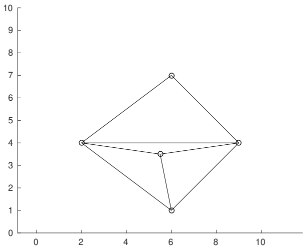
About me

- Ph.D. candidate at Virginia Tech
- Advisor: Dr. Layne Watson
- Interests: Analysis! (Numerical, Functional, Stochastic)
- Skills: Algorithms, Parallel Computing, Low-Level Languages
- Application areas: Data Science, Engineering Design, Quantum Computing

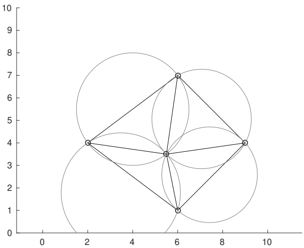
Table of Contents

About Delaunay Triangulations

- The *Delaunay triangulation* is an unstructured simplicial mesh defined by an arbitrary vertex set $P = \{p_1, \ldots, p_n\} \subset \mathbb{R}^d$
- 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: $B_S \cap P = \emptyset$.



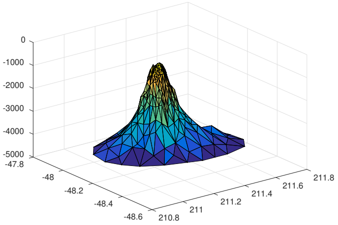
✗



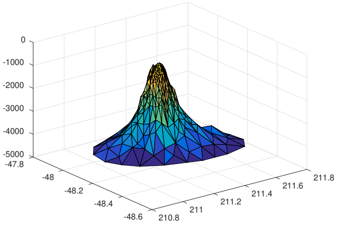
✓

Applications of Delaunay Triangulations

- Interpolation mesh for
  - Finite element method,
  - data science,
  - GIS, and
  - computer graphics
- Delaunay graph



- Interpolation mesh for
  - Finite element method,
  - data science,
  - GIS, and
  - computer graphics
- Delaunay graph



Piecewise linear interpolation: Let $f : \mathbb{R}^d \to \mathbb{R}$, and let $q \in S \in DT(P)$. S has vertex set $\{s_1, \ldots, s_{d+1}\}$ and there exist convex weights $\{w_1, \ldots, w_{d+1}\}$ such that $q = \sum_{i=1}^{d+1} w_i s_i$.

$$\hat{f}_{DT}(q) = \sum_{i=1}^{d+1} w_i f(s_i).$$

Problem: 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, we only need the vertices $(\{s_1, \ldots, s_{d+1}\})$ of $S \in DT(P)$ such that $q \in S$

$$\hat{f}_{DT}(q) = \sum_{i=1}^{d+1} w_i \, f(s_i).$$

Question: Can we find S containing q in polynomial time (without computing the whole mesh)?

Algorithm outline

- Grow an initial simplex (greedy algorithm)
- "Flip" accross a facet from which q is visible
- This "visibility walk" converges to q in k steps (Edelsbrunner's acyclicity theorem)

Full algorithm published in *Tyler H. Chang, et al. "A polynomial time algorithm for multivariate interpolation in arbitrary dimension via the Delaunay triangulation." In the ACMSE 2018 Conf.*

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

DELAUNAYSPARSE Package

Standalone software package `DELAUNAYSPARSE`:

- Robust against degeneracy
- Runs in $\mathcal{O}(kmnd^2)$ time, where k is the number of "flips", n is the numer of data points, m is the number of interpolation points, and d is the input dimension
- Typically, $k \approx \mathcal{O}(d \log d)$
- Parallel and serial implementations

Under review: *Tyler H. Chang, et al. "Algorithm XXX: DELAUNAYSPARSE: Interpolation via a sparse subset of the Delaunay triangulation in medium to high dimensions." Submitted to ACM Transactions on Mathematical Software (2019)*.

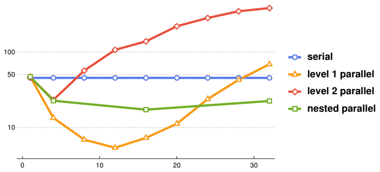
Serial performance

Runtime in seconds for interpolating a single point $(m = 1)$ with n points in d dimensions

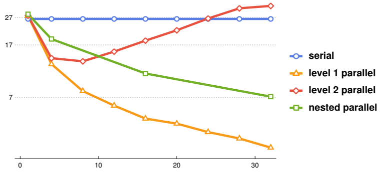
n	d = 2	8	32	64	128
100	0.001	0.004	0.060	0.820	n/a
500	0.021	0.042	0.325	6.479	59.511
2000	0.344	0.583	2.230	28.984	242.066
8000	5.580	9.027	26.210	151.177	905.711
16,000	22.086	35.725	109.448	386.596	2190.362
32,000	82.915	145.115	421.934	1097.060	slow

Parallel implementation

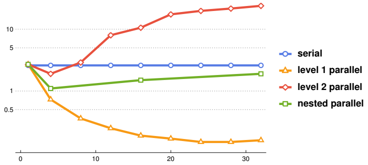
- Level 1: loop over multiple interpolation points
- Level 2: loop(s) over data points – imperfect scaling



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



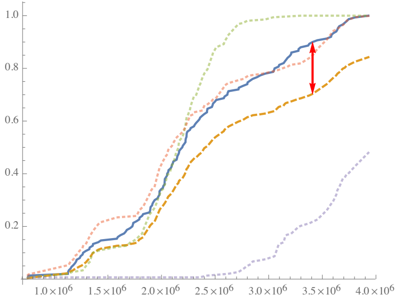
$d = 10,\ n = 10,000,\ m = 64$



$d = 20,\ n = 200,\ m = 64$

Applications of DELAUNAYSPARSE

- HPC system data interpolation
  - Nonparametric distribution interpolation
- Aerospace engineering – oblique shock calculation
  - Surrogate model to "warm start" calculations
- Data science applications
  - Comparison with neural network and support vector regressors



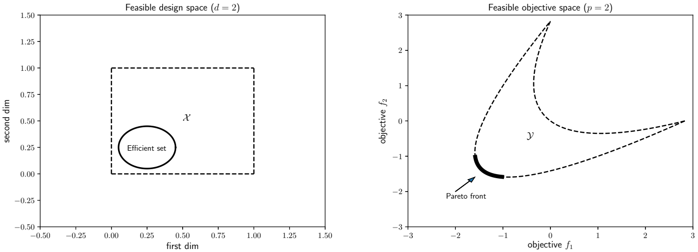
Future work

- Delaunay interpolation in an arbitrary metric space
- Other sparse subsets, such as umbrella neighborhood
  Tyler H. Chang, et al. "Computing the umbrella neighbourhood of a vertex in the Delaunay triangulation and a single Voronoi cell in arbitrary dimension." In IEEE SoutheastCon 2018.

Questions about Delaunay interpolation?

What is a MOP?

- The Multiobjective Optimization Problem (MOP) generalizes the Single Objective (Scalar) Optimization Problem (SOP);
- The MOP attempts to balance the tradeoff between multiple conflicting objectives;
- Whereas the SOP generally has a unique solution, the solution to a MOP is a *set* of *Pareto optimal* solutions;



Goals:

Find a discrete set of approximately nondominated objective points that describes the Pareto front, and the corresponding efficient designs

Types of MOPs

functions are "cheap" to evaluate derivative info is available	functions are "cheap" to evaluate no derivative info is available
functions are costly to evaluate derivative info is available	functions are costly to evaluate no derivative info is available

Focus on bottom right: **expensive blackbox MOPs**!

Motivating Example: VarSys

VarSys: Managing performance variance

- For multiple runs of the same I/O task on the same HPC system, we get varying throughputs
- This presents issues for load balancing and performance guarantees
- Needs to be balanced against other concerns such as energy consumption and mean throughput
- Evaluation expense: 1+ minutes to build distributions

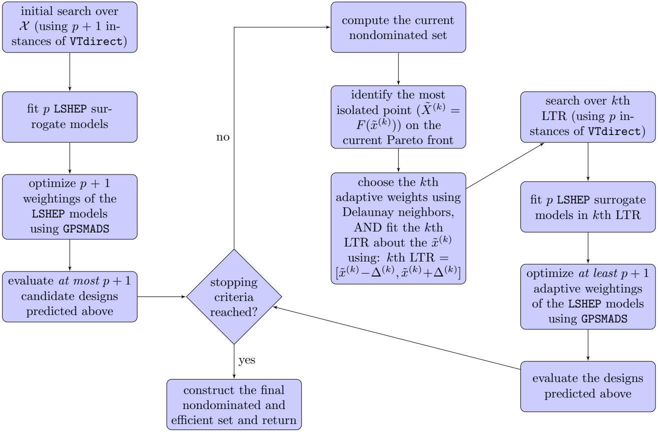
VTMOP

`VTMOP` is a Fortran 2008 blackbox MOP solver and framework, based on an algorithm by *Shubhangi Deshpande, et al. "Multiobjective optimization using an adaptive weighting scheme." Optimization Methods and Software 31.1 (2016): 110-133.*

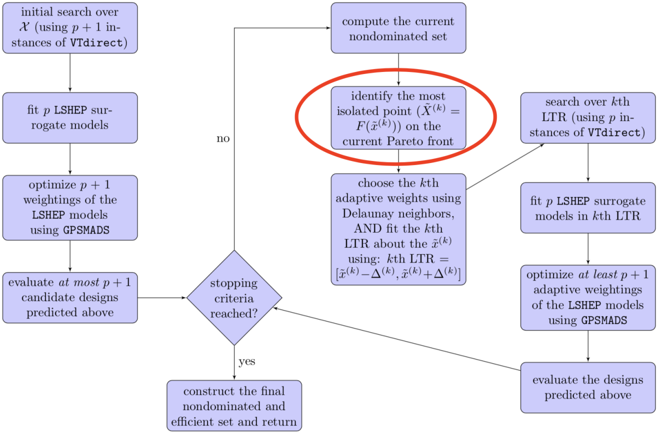
`VTMOP` is meant to be flexible, scalable, portable, robust, and efficient for solving expensive blackbox MOPs

Combines adaptive weighting scheme, response surface modeling, and trust region methods

The Algorithm Outline



Key component



Identifying an Isolated Point

Let $P^{(k)}$ be the kth set of nondominated objective points $P^{(k)} = \{X^{(1,k)}, \ldots, X^{(N_k,k)}\}$. Define the projected set

$$H^{(k)} = \left\{ \left(\frac{X_1^{(n,k)}}{X_p^{(n,k)}}, \ldots, \frac{X_{p-1}^{(n,k)}}{X_p^{(n,k)}} \right) \;\middle|\; n = 1, \ldots, N_k \right\}$$

The most isolated point is identified by considering the average Euclidean distance to all neighbors in the Delaunay graph of $H^{(k)}$

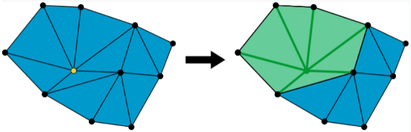


Image from Wikipedia

Compute the Delaunay neighborhood of $X^{(1,k)}$, ..., $X^{(N_k,k)}$ with respect to the projected set $H^{(k)}$.

- Only need the Delaunay graph G_{DT}
- Number of connections in G_{DT} is upper bounded by $N_k(N_k - 1)/2$
- Can recover G_{DT} by interpolating the midpoint between each pair of points in $H^{(k)}$
- Using `DELAUNAYSPARSE`, requires $\mathcal{O}(N_k^3 p^3 \log p)$ time

Parallel implementations

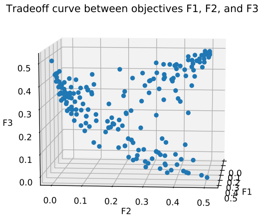
Focus on achieving parallel function evaluations:

- One "unmodified" implementation – distributes function evaluations that can be done asynchronously without changing the original algorithm
- The `libEnsemble` implementaion – integrates with Argonne's `libEnsemble` library (part of the Exascale Computing Project) to achieve increased levels of concurrency
  - Required significant modification to the underlying algorithm

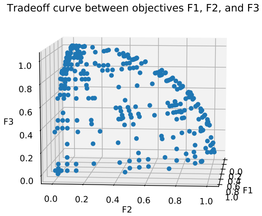
Tyler H. Chang, et al. "Managing computationally expensive blackbox multiobjective optimization problems with libEnsemble." Submitted to SpringSim 2020, 28th HPC Symposium.

Benchmark problems

- $F_c(x) = (\|x - e_1\|_2^2, \ldots, \|x - e_p\|_2^2)$
- Convex Pareto front $\Rightarrow$ "easier" problem



- `DTLZ2` from Deb et al.
- Concave Pareto front $\Rightarrow$ "harder" problem



Number of solutions, RMSE, and Delaunay discrepancy (respectively) for F_c and `DTLZ2`, after a budget of 2000 function evaluations, with $d = 5$ (Averaged over 5 runs).

Performance metrics:

1. the *cardinality* of the solution set (num pts)
2. the *convergence* of the solution points to the true Pareto front (RMSE)
3. the *relative spacing/coverage* of the solution set (Delaunay discrep)

Prob/Meth	$p = 2$	$p = 3$	$p = 4$
F_c / `bVTdir`	73, .00100, .207	173, .0505, .579	288, .101, NA
F_c / `libE`	78, .0127, .158	189, .0560, .429	283, .104, .551
`DTLZ2` / `bVTdir`	139, .00713, .109	354, .0401, .230	658, .0443, NA
`DTLZ2` / `libE`	66, .103, .201	258, .175, .691	548, .201, .793

Runtimes for VTMOP with 2000 function evaluations (either 1 second or in range [0.5 s, 1.5 s]), for bVTdirect and libEnsemble with $d = 5$. Shows CPU time / wall time in seconds, for 36 core machine.

p	Method	F_c, no var	F_c, w/ var	DTLZ2, no var	DTLZ2, w/ var
2	bVTdir	2008 / 1037	2007 / 1039	2007 / 1093	2004 / 1082
	libE	2051 / 112	2070 / 142	2060 / 111	2064 / 143
3	bVTdir	2012 / 717	2012 / 719	2021 / 797	2018 / 797
	libE	2077 / 133	2066 / 144	2054 / 99	2057 / 126
4	bVTdir	2026 / 582	2029 / 586	2177 / 807	2149 / 782
	libE	2134 / 190	2124 / 186	2182 / 227	2185 / 257

Spectrum of blackbox problems

Computationally cheap ⟶ expensive

- Eval: ≈ 1 sec
- Budget: $\approx 10,000$
- Software: `NSGA-II`

- Eval: ≈ 1 min
- Budget: ≈ 1000
- Software: `VTMOP`

- Eval: ≈ 1 hr
- Budget ≈ 100 (at most)
- Software: `FUN3D`? `NASTRAN`?

Spectrum of blackbox problems

Computationally cheap $\longrightarrow$ expensive

- Eval: ≈ 1 sec
- Budget: $\approx 10,000$
- Software: `NSGA-II`

- Eval: ≈ 1 min
- Budget: ≈ 1000
- Software: `VTMOP`

- Eval: ≈ 1 hr
- Budget ≈ 100 (at most)
- Software: `FUN3D`? `NASTRAN`?
- **Future work!**

Significant work

Peer-Reviewed:

T. H. Chang, et al. "Least-squares solutions to polynomial systems of equations with quantum annealing." Springer, QINP 18:374 (2019).

T. H. Chang, et al. "Computing the umbrella neighbourhood of a vertex in the Delaunay triangulation and a single Voronoi cell in arbitrary dimension." In IEEE SoutheastCon 2018.

T. H. Chang, et al. "A polynomial time algorithm for multivariate interpolation in arbitrary dimension via the Delaunay triangulation." In the ACMSE 2018 Conf.

T. H. Chang, et al. "Predicting system performance by interpolation using a high-dimensional Delaunay triangulation." In SpringSim 2018, 26th HPC Symp.

Under Review:

T. H. Chang, et al. "Algorithm XXX: DELAUNAYSPARSE: Interpolation via a sparse subset of the Delaunay triangulation in medium to high dimensions." Submitted to ACM TOMS (2019).

T. H. Chang, et al. "Managing computationally expensive blackbox multiobjective optimization problems with libEnsemble." Submitted to SpringSim 2020, 28th HPC Symp.

Major Awards:

Cunningham Fellow. Virginia Tech, Grad School. 2016–Present

SCGSR award. DOE, Office of Sci. Jun–Dec, 2019

Various CS/Eng. dept. fellowships. Virginia Tech. 2016–Present

Projects:

VarSys project at Virginia Tech. NSF grant #1565314

Professional:

Reviewer for *IEEE SoutheastCon* and *JMLR*