PREDICTING SYSTEM PERFORMANCE BY INTERPOLATION USING A HIGH-DIMENSIONAL DELAUNAY TRIANGULATION

Tyler Chang[a], Layne Watson[abc], Thomas Lux[a], Jon Bernard[a], Bo Li[a], Li Xu[d], Godmar Back[a], Ali Butt[a], Kirk Cameron[a], and Yili Hong[d]

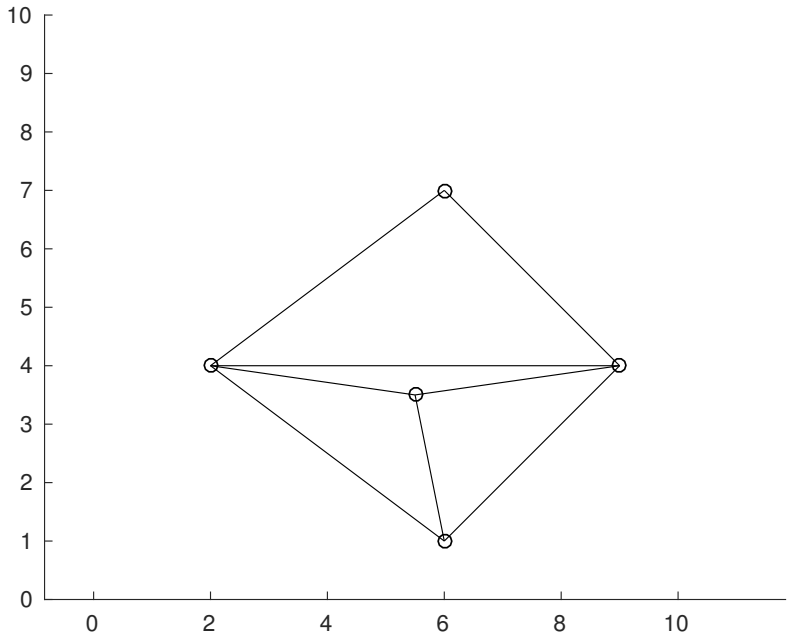
Departments of Computer Science[a], Mathematics[b], Aerospace & Ocean Engineering[c], and Statistics[d]

Virginia Polytechnic Institute and State University
Blacksburg, VA 24061-0106 USA

COMPUTER SCIENCE
VIRGINIA TECH

What is a Triangulation?

A *triangulation* of a (finite) set of points P in $\mathbb{R}^d$ is a space filling simplical-mesh, whose elements are disjoint except along shared boundaries, whose vertices are points in P, and whose union is the *convex hull* of P.



Note: The above triangulation makes for a poor interpolation mesh. Think about why...

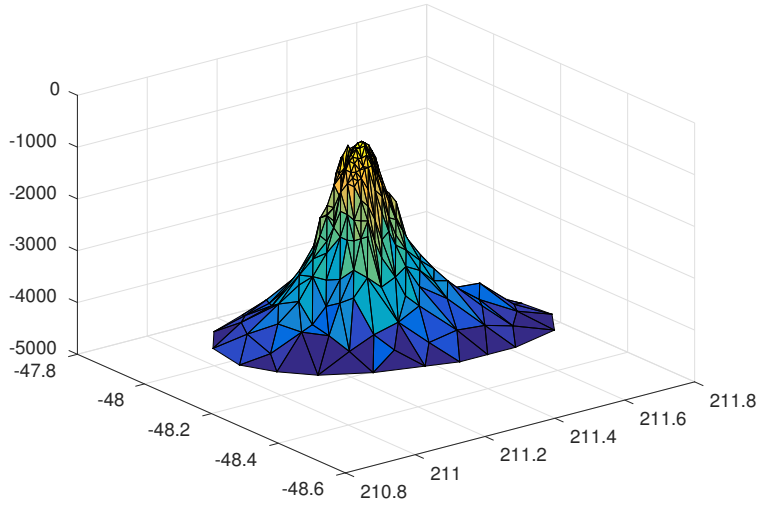
Multivariate Interpolation via Triangulation

Given a triangulation $T(P)$, define a piecewise linear multivariate interpolant as follows:

- Let $T(P)$ be a d-dimensional triangulation of P.
- Let $q \in CH(P)$ be an interpolation point, and let S be a simplex in $T(P)$ with vertices $s_1, \ldots, s_{d+1}$ such that $q \in S$.
- Then there exist unique *convex weights* $w_1, \ldots, w_{d+1}$ such that $q = \sum_{i=1}^{d+1} w_i s_i$.

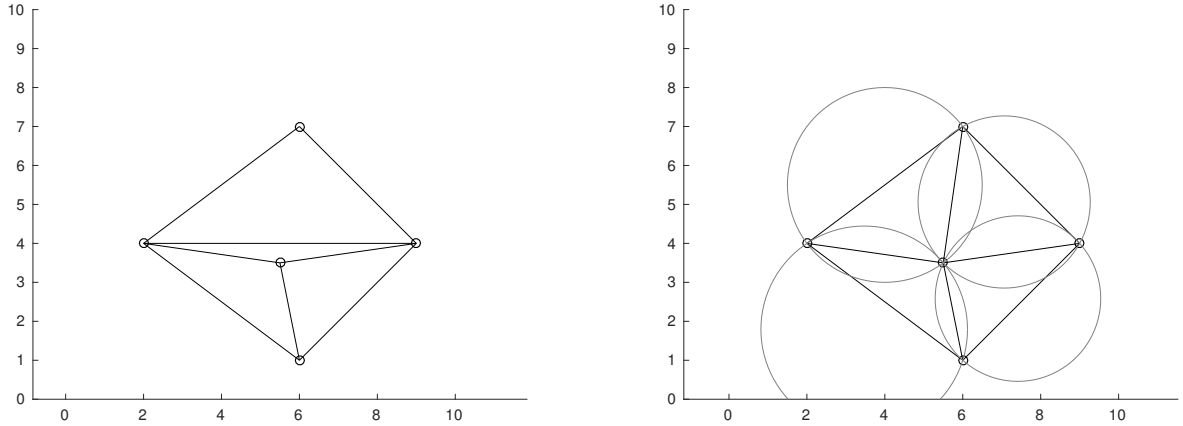
Then the interpolant $\hat{f}_T$ is given by

$$\hat{f}_T(q) = f(s_1)w_1 + f(s_2)w_2 + \ \ldots \ + f(s_{d+1})w_{d+1}.$$



The Delaunay Triangulation

The Delaunay triangulation is a particular triangulation (usually defined in terms of the *empty circumsphere property*) that is considered optimal for interpolation.



Recall the above (left) arbitrary triangulation of points in the plane. The shape of the skinny triangle in the middle makes this a bad interpolatory mesh. The Delaunay triangulation (right) satisfies the empty circumsphere property, making all its elements as “well shaped” as possible.

The VarSys Problem

- A deterministic program is run with the same inputs on a machine with fixed system-level configurations, but different run times are observed.
- Using a single summary statistic such as the mean observed run time does not capture the true nature of this *distribution* of run times.
- Another performance statistic that might be helpful is the variance between runs, which together with the mean run time, gives a better understanding of the true distribution.
- To this end, a model is needed that describes the *performance variance* of a system as a function of application and system level parameters.

VarSys Methodology

This experiment attempts to model *throughput variance* as a function of application and system *parameters*.

To do so, the *IOZone* benchmark is used to read files of varying sizes on a homogeneous system. The variance in the throughput for each read is modelled as a function of several parameters.

The parameters chosen and the values used for them are in the table below:

Parameters	Values
file size being read (in KB)	64, 256, 1024
record size of file system (in KB)	32, 128, 512
number of threads for IOZone reader	1, 2, 4, 8, 16, 32, 64, 128, 256
CPU frequency (in GHz)	1.2, 1.4, 1.5, 1.6, 1.8, 1.9, 2.0, 2.1, 2.3, 2.4, 2.5, 2.7, 2.8, 2.9, 3.0, 3.001

For each combination of the above parameters, 40 runs of IOzone were done and the observed throughput variance was computed using:

$$\sigma^2 = \left(\sum_{i=1}^{40} (t_i - \mu)^2 \right) / 39$$

where t_i is the ith observed throughput and μ is the observed mean over all 40 runs.

Challenges: Using Delaunay to Compute the Model

Due to the nonlinearity of the underlying throughput variance function, it is very difficult to model.

Recall that the Delaunay triangulation is only piecewise linear, so it can effectively model nonlinear functions.

Goal: To use the Delaunay triangulation to model this function.

Challenge: Unfortunately, the Delaunay triangulation grows exponentially large with the *dimension*.

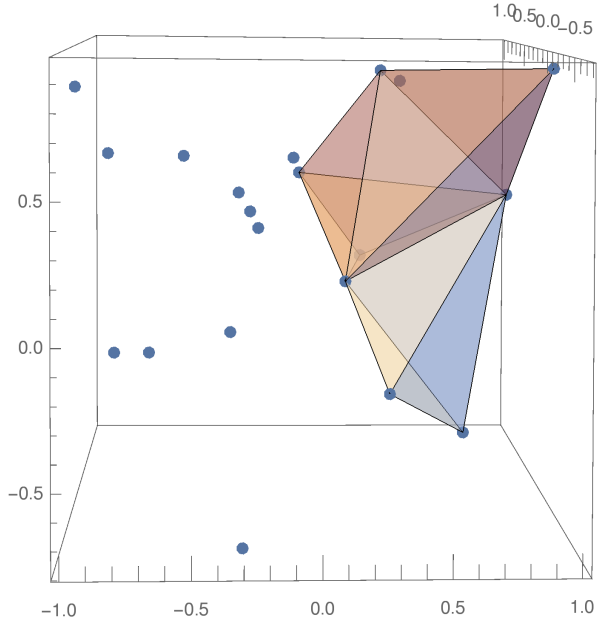
In the worst case, the total size of the Delaunay triangulation of n points in $\mathbb{R}^d$ is $\mathcal{O}\left(n^{\lceil d/2 \rceil}\right)$.

Solution: A New Algorithm for Delaunay Interpolation

Recall that only the simplex containing the interpolation point q is needed to compute the Delaunay interpolant $\hat{f}_{DT}(q)$.

So, the following steps are taken to obtain *only* the simplex containing q, and *not* the whole triangulation.

- Grow an initial simplex close to q (through a sequence of least squares problems).
- Flip that simplex toward q (by dropping the vertex "opposite" q)
- Iterate until q is found ("walking" toward q by flipping simplices)



Algorithm Analysis

- Takes $\mathcal{O}\left(nd^4\right)$ time to compute the first simplex.
- Takes $\mathcal{O}\left(nd^3\right)$ time to perform each "flip."
- Experimentally, takes $\mathcal{O}\left(d \log d\right)$ flips to "walk" from first simplex to q.
- After locating simplex, time to compute interpolant $\hat{f}_{DT}(q)$ is trivial.

So, total time to interpolate at a single point q is $\mathcal{O}\left(nd^4 \log d\right)$! (sub-exponential)

Model Evaluation

Tested accuracy of Delaunay interpolant by using various "training" percentages to predict various points.

- I.e., predict the variance at a point $q \in P$ using $x\%$ of the remaining data: $P \setminus \{q\}$.

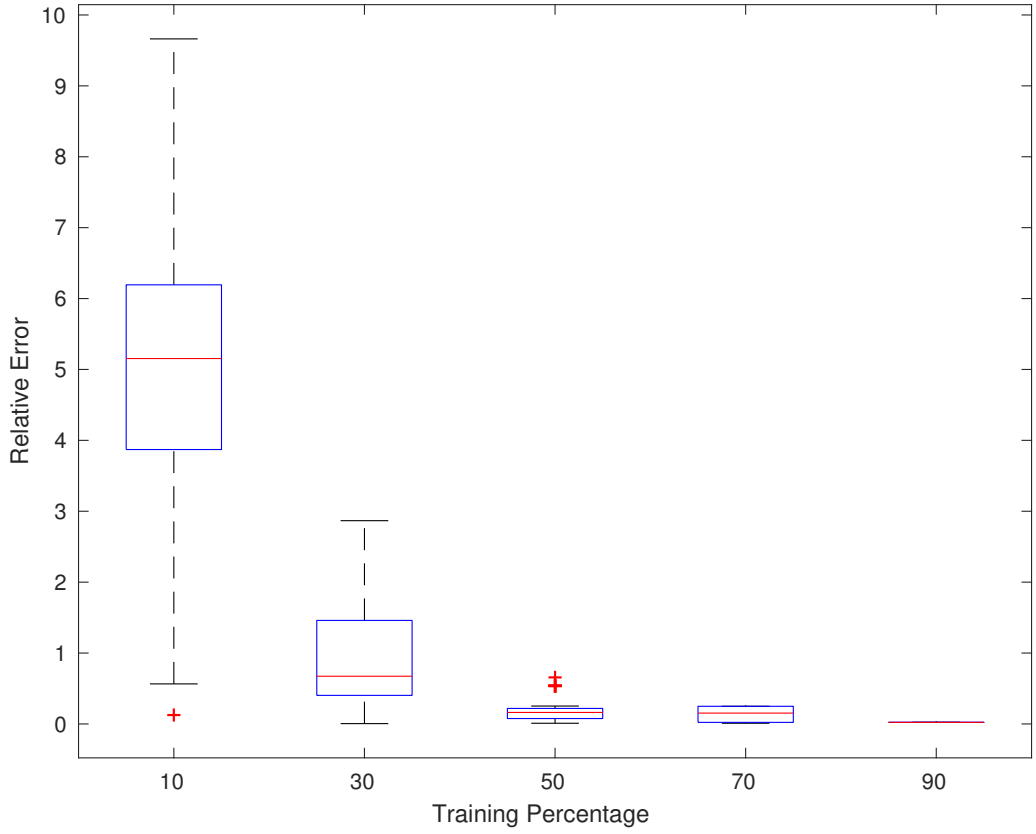
For each point q and each training percentage x, 200 unique random samplings of $P \setminus \{q\}$ are done to obtain $x\%$, with a bias toward uniformly distributed samplings.

The relative error for each prediction was computed using

$$|\hat{f}_{DT}(q) - f(q)|/f(q).$$

Results: Boxplot of Errors

For the point q corresponding to a file size of 1024 KB, record size of 32 KB, 4 threads, and CPU frequency of 2.9 GHz, a boxplot of the prediction error is shown:



Conclusion

A new scalable algorithm for computing Delaunay interpolants (without computing the complete Delaunay triangulation) was presented.

Empirically, it performed well on a real world systems problem concerning performance variance.

Future Work

The above algorithm can be used for many types of interpolation and meshing problems.

Similar ideas could also be extended to other applications for Delaunay triangulations.

Acknowledgements

The data shown here was collected for the VarSys project at Virginia Tech.