PREDICTIVE MODELING OF I/O CHARACTERISTICS IN HIGH PERFORMANCE COMPUTING SYSTEMS

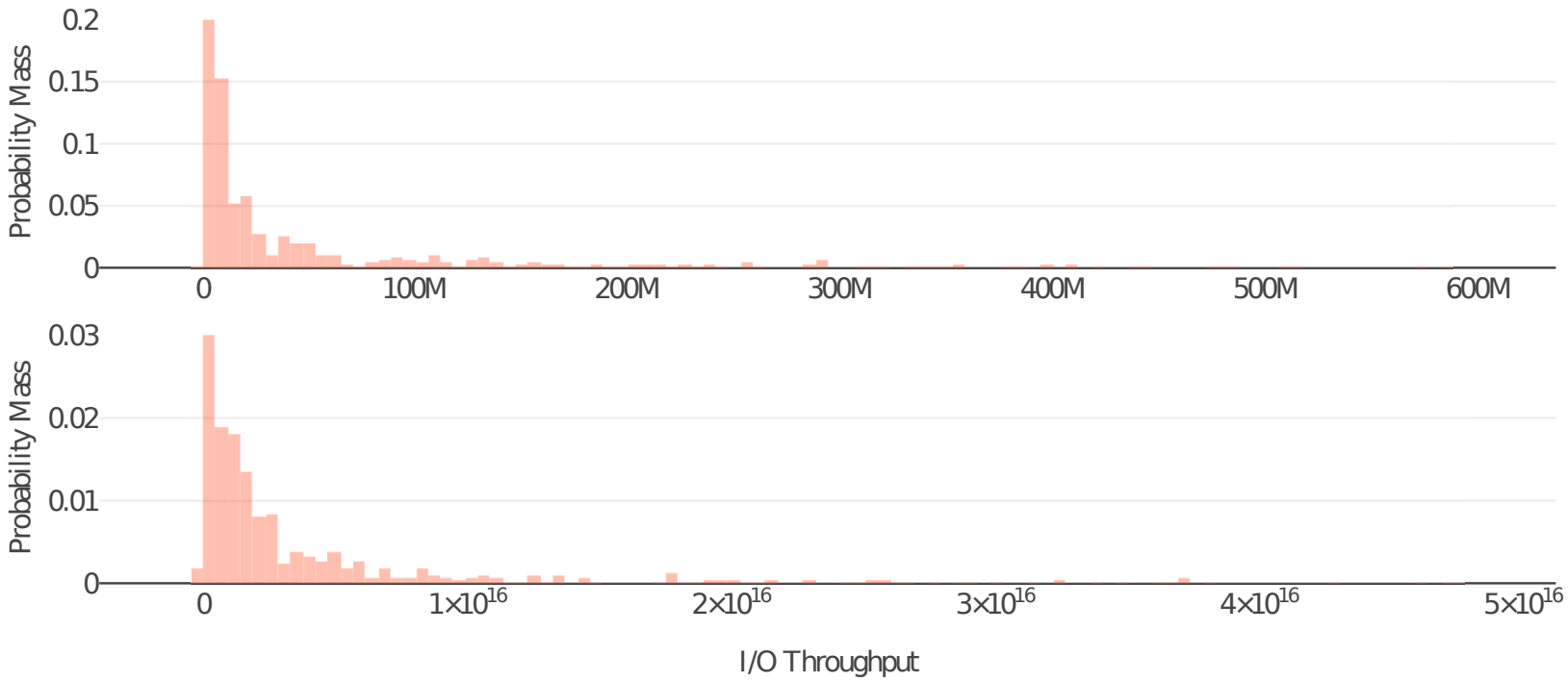
Thomas Lux[1], Layne Watson[123], Tyler Chang[1], Jon Bernard[1], Bo Li[1], Li Xu[4], Godmar Back[1], Ali Butt[1], Kirk Cameron[1], and Yili Hong[4]

Departments of Computer Science[1], Mathematics[2], Aerospace & Ocean Engineering[3], and Statistics[4]

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

Problem Setting

When the same computer is used to execute the same program repeatedly, most measurable performance characteristics will vary. This paper will consider the execution of IOzone, a file I/O benchmark for HPC systems.



This variance in performance can make it difficult to identify an optimal configuration that meets minimum requirements.

Proposed Approach

Use multivariate interpolation and regression to model the performance of a computer with respect to the system and application configuration.

Multivariate *interpolation* and *regression* are defined when there exists $f : \mathbb{R}^d \rightarrow \mathbb{R}$ and a set X of n points in $\mathbb{R}^d$ along with response values $f(x)$ for all $x \in X$.

Interpolation constructs an approximation $\hat{f} : \mathbb{R}^d \rightarrow \mathbb{R}$ such that $\hat{f}(x) = f(x)$ for all $x \in X$. The form of the true underlying function f is often unknown, however it is still desirable to construct an approximation $\hat{f}$ with minimum approximation error at $y \notin X$.

Regression relaxes the conditions of interpolation by minimizing the error in $\hat{f}$ at $x \in X$ while maintaining some parametric form with parameters P. This can be written as $\min_P ||\hat{f}(X) - f(X)||$, where $f(X)$ is a vector of $f(x)$ for all $x \in X$ and $|| \cdot ||$ is an appropriate measure.

Regression Algorithms

Multivariate Adaptive Regression Splines (MARS) is an iterative approximation of the form

$$B_{2s-1}(x) = B_l(x)[c(x_i - v)]_+,$$

$$B_{2s}(x) = B_k(x)[c(x_i - v)]_-,$$

where s is the iteration number, $B_l(x)$ and $B_k(x)$ are basis functions from the previous iteration, $c, v \in \mathbb{R}$,

$$w_+ = \begin{cases} w, & w \geq 0 \\ 0, & w < 0 \end{cases},$$

and $w_- = (-w)_+$. After iteratively constructing a model, MARS then iteratively removes basis functions that do not contribute to goodness of fit.

In effect, MARS creates a locally component-wise tensor product approximation of the data. The computational complexity of MARS is $\mathcal{O}(ndm^3)$ where m is the maximum number of underlying basis functions.

Regression Algorithms

Multilayer Perceptron Regressor (MLP Regressor)

$$l(u) = \left(u^t W_l\right)_+,$$

where W_l is the i by j weight matrix for layer l.

The multilayer perceptron (MLP) produces a piecewise linear model of the input data. The computational complexity is $\mathcal{O}(ndm)$, where m is determined by the sizes of the layers of the network and the stopping criterion of the error minimizer.

Support Vector Regressor (SVR)

$$p(x) = \sum_{i=1}^{n} a_i K(x, x^{(i)}) + b,$$

where K is the selected kernel function, $a \in \mathbb{R}^n$, $b \in \mathbb{R}$ are coefficients to be solved for simultaneously. The computational complexity of the SVR is $\mathcal{O}(n^2 dm)$, with m being determined by the minimization convergence criterion.

Interpolation Algorithms

Delaunay constructs a simplicial mesh such that for a d-simplex S with vertices $v^{(0)}$, $v^{(1)}$, ..., $v^{(d)}$, $x \in S$, and data values $f(v^{(i)})$, $i = 0, \ldots, d$, x is a unique convex combination of the vertices, and the interpolant is given by

$$p(x) = \sum_{i=0}^{d} w_i f(v^{(i)}),$$

where w_i are the convex weights. The computational complexity of the Delaunay triangulation (for the implementation used here) is $\mathcal{O}(n^{\lceil d/2 \rceil})$.

Linear Shepard (LSHEP) blends local linear interpolants and has the form

$$p(x) = \frac{\sum_{k=1}^{n} W_k(x) P_k(x)}{\sum_{k=1}^{n} W_k(x)},$$

where $W_k(x)$ is a locally supported weighting function and $P_k(x)$ is a local linear approximation to the data satisfying $P_k\left(x^{(k)}\right) = f\left(x^{(x)}\right)$. The computational complexity of LSHEP is $\mathcal{O}(n^2 d^3)$.

IOzone Data

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.

Analyzing Predictability: Multi Dimensional Analysis

1. For all $k = 1, \ldots, d$ and for all nonempty subsets $F \subset \{1, 2, \ldots, d\}$, reduce the input data to points $(z, f_F(z))$ with $z \in \mathbb{R}^k$ and $f_F(z) = E\left[\left\{f\left(x^{(i)}\right) \mid \left(x_F^{(i)} = z\right)\right\}\right]$, where $E[\cdot]$ denotes the mean and $x_F^{(i)}$ is the subvector of $x^{(i)}$ indexed by F.

2. For all r in $\{5, 10, \ldots, 95\}$, generate N random splits $(train, test)$ of the reduced data with r percentage for training and $100 - r$ percentage for testing.

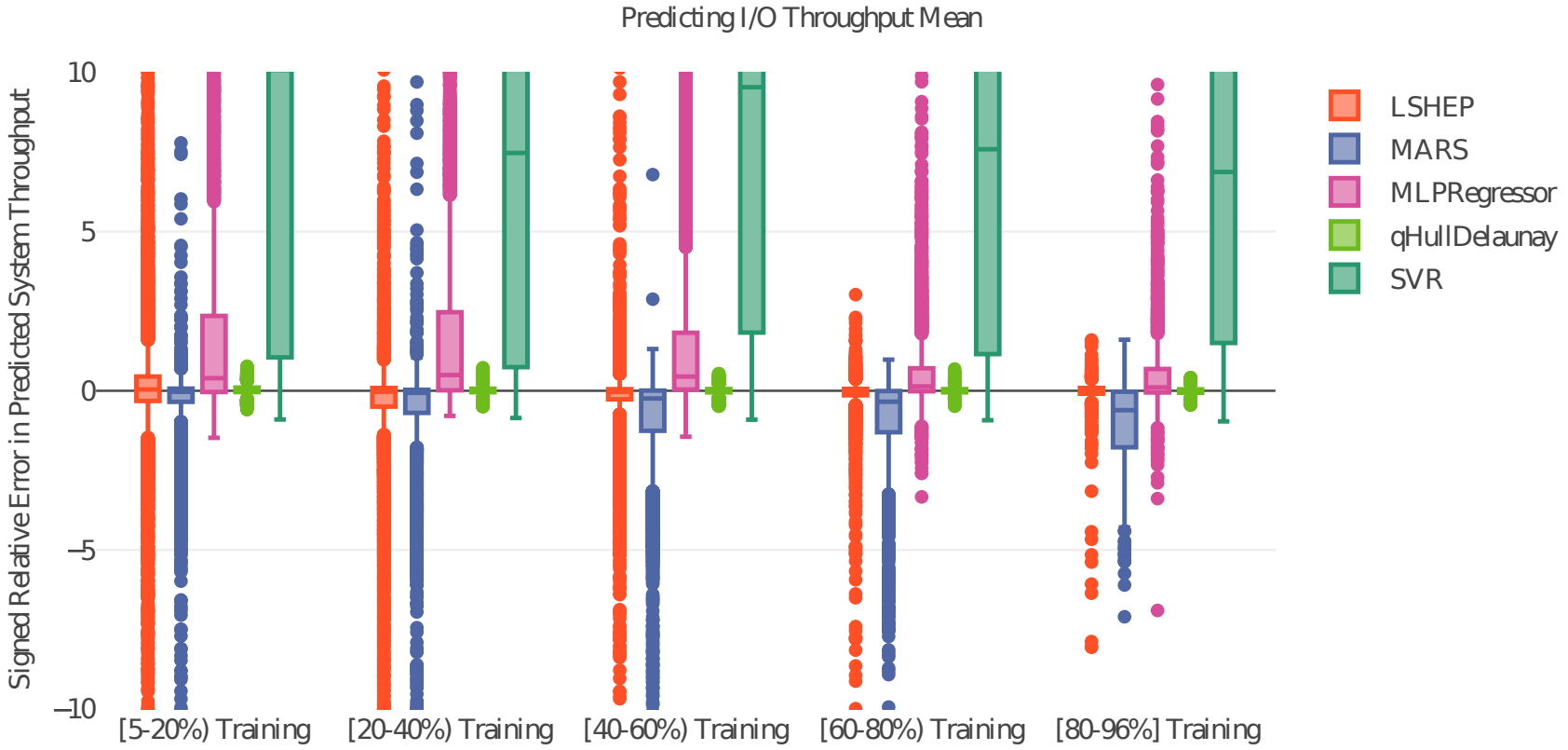
3. When generating each of N random $(train, test)$ splits, ensure that all points from $test$ are in the convex hull of points in $train$ (to prevent extrapolation); also ensure that the points in $train$ are well spaced.

In order to ensure that training points are well spaced, a statistical method for picking points is used

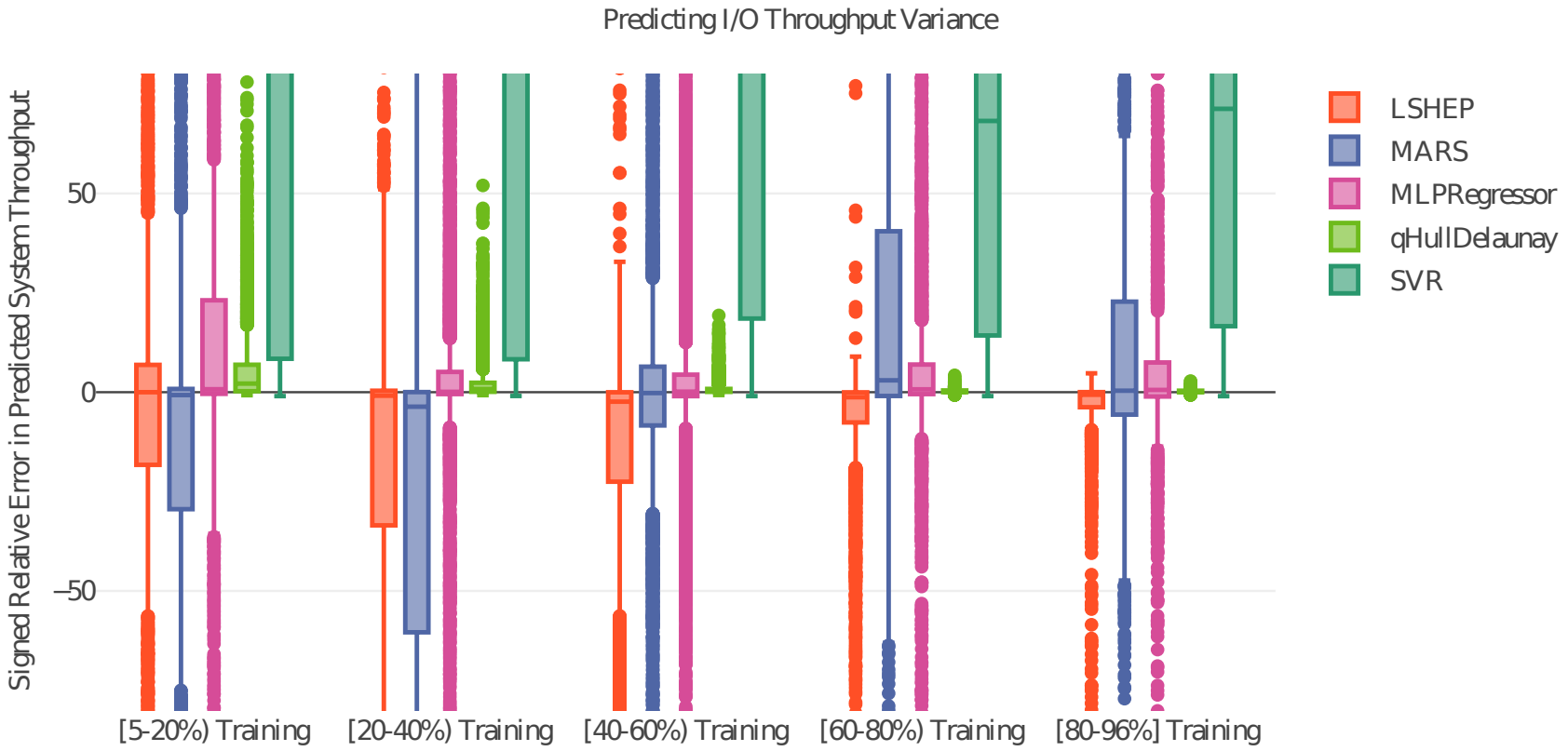
1. Generate a sequence of all pairs of points $\left(z^{(i_1)}, z^{(j_1)}\right), \left(z^{(i_2)}, z^{(j_2)}\right), \ldots$ sorted by ascending pairwise Euclidean distance between points, so that $\left|\left|z^{(i_k)} - z^{(j_k)}\right|\right|_2 \leq \left|\left|z^{(i_{k+1})} - z^{(j_{k+1})}\right|\right|_2$.

2. Sequentially remove points from candidacy until only $|train|$ remain by randomly selecting one point from the pair $\left(z^{(i_m)}, z^{(j_m)}\right)$ for $m = 1, \ldots$ if both $z^{(i_m)}$ and $z^{(j_m)}$ are still candidates for removal.

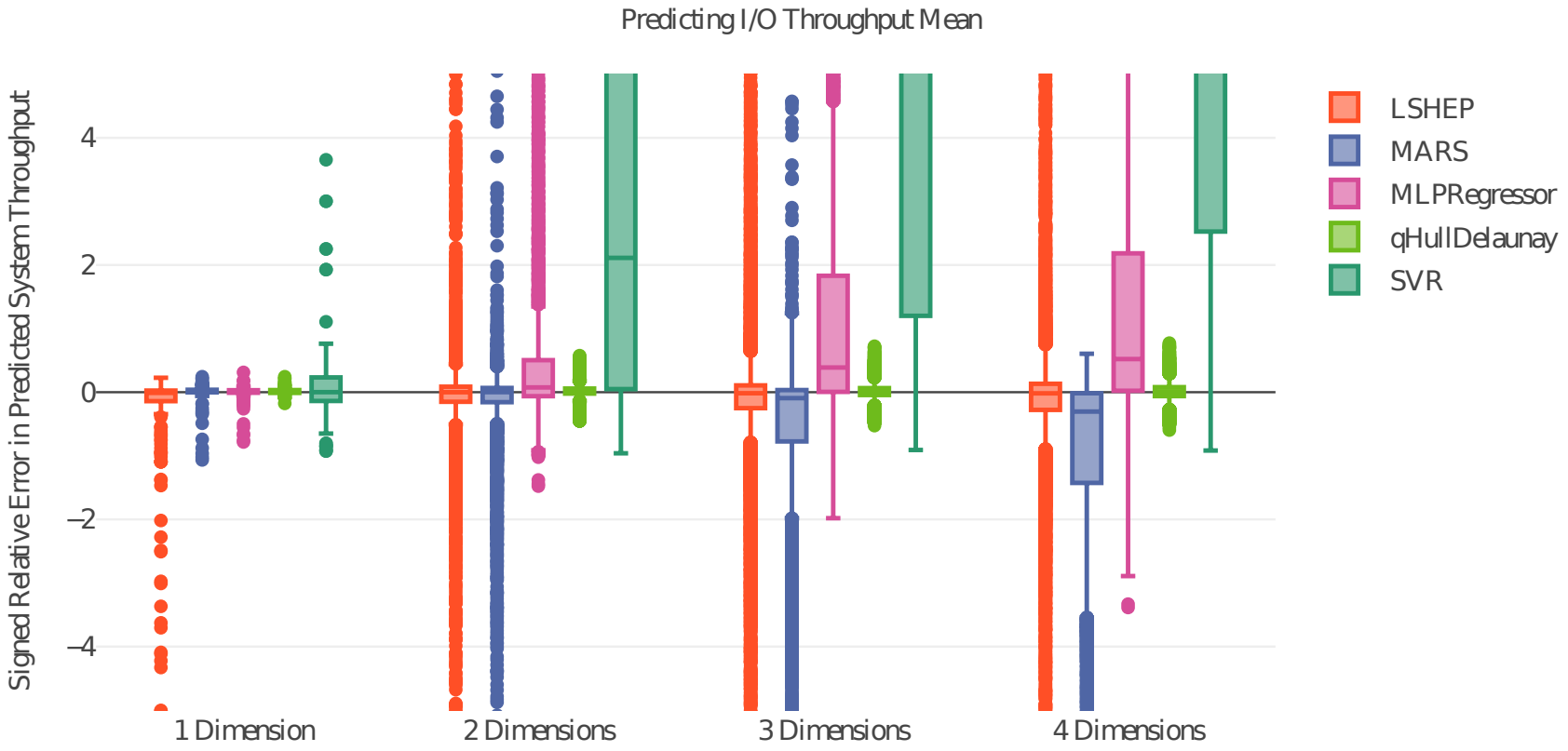
Increasing Samples: Results for Predicting Throughput Mean



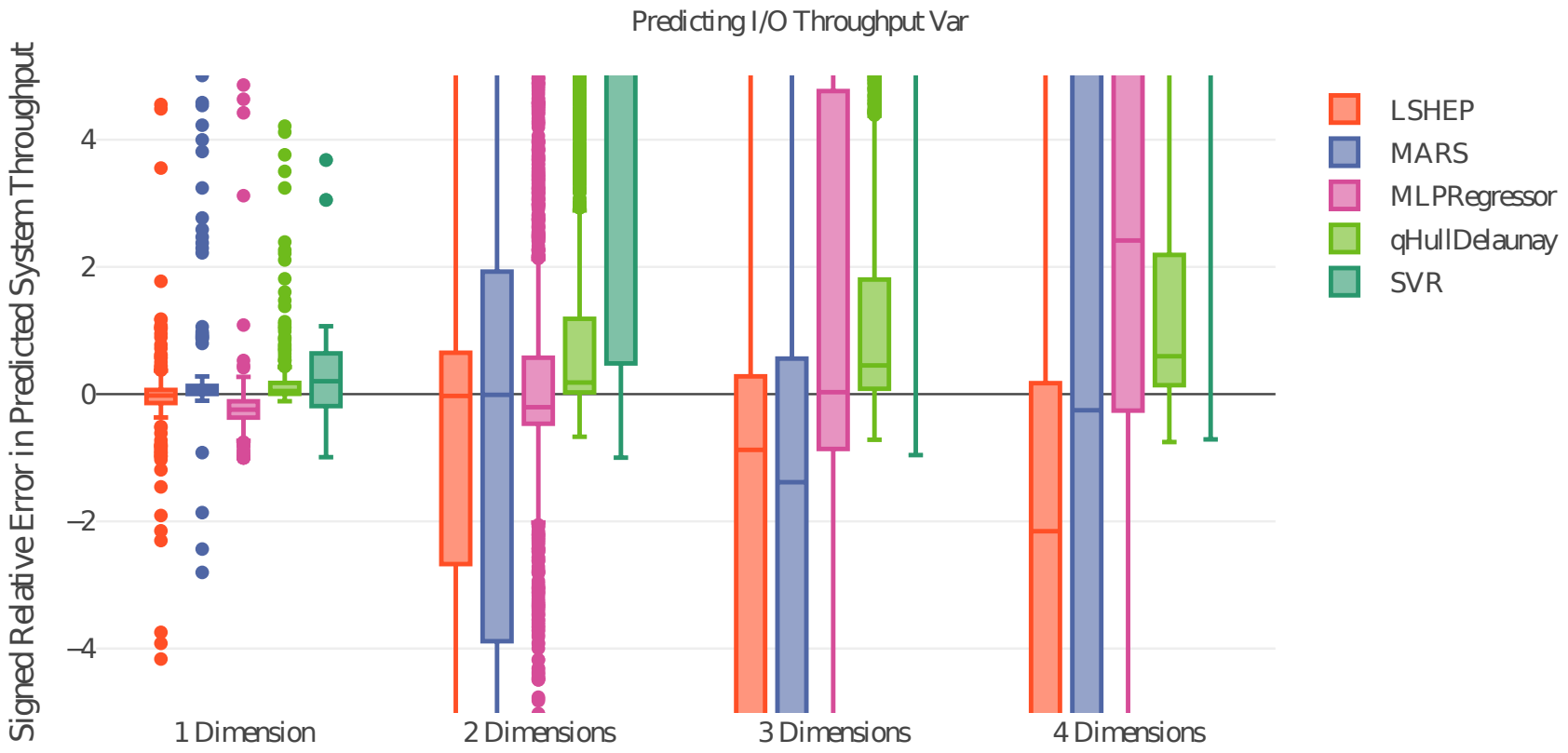
Increasing Samples: Results for Predicting Throughput Variance



Increasing Dimension: Results for Predicting Throughput Mean



Increasing Dimension: Results for Predicting Throughput Variance



Conclusion and Future Work

Multivariate models of HPC system performance can effectively predict I/O throughput mean and variance. These multivariate techniques significantly expand the scope and portability of statistical models for predicting computer system performance over previous work.

Delaunay model can make predictions for 821 system configurations with less than 5% error when trained on only 43 configurations.

Multivariate methods should be applied to predict distributions instead of summary statistics. This would allow for much deeper insight.

Acknowledgements

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