Interpolants Scalable with Dimension*

Thomas C. H. Lux
Dept. of Computer Science
Virginia Polytechnic Institute and State University
Blacksburg, Virginia
tchlux@vt.edu

Layne T. Watson
Depts. of Computer Science, Mathematics, and Aerospace & Ocean Engineering
Virginia Polytechnic Institute and State University

Tyler H. Chang
Dept. of Computer Science
Virginia Polytechnic Institute and State University

ACM Reference Format:
Thomas C. H. Lux, Layne T. Watson, and Tyler H. Chang. 2018. Interpolants Scalable with Dimension*. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

ABSTRACT

A rapid increase in the quantity of data available is allowing all fields of science to generate more accurate models of multivariate phenomena. Regression and interpolation become challenging when the dimension of data is large, especially while maintaining tractable computational complexity. While regression is a popular approach to solving approximation problems with high dimension, there are often advantages to interpolation. This paper presents error bounds for, and analyzes the performance of interpolants on moderately high dimension problems and contrasts their approximations with popular regression techniques. Empirical results demonstrate the viability of interpolants for high dimension approximation problems, suggesting that these techniques are capable of effectively modeling multivariate phenomena while maintaining flexibility in different application domains.

Keywords: Approximation, regression, interpolation, high dimension, error bound

1 INTRODUCTION

Regression and interpolation are problems of considerable importance that find applications across many fields of science. Pollution and air quality analysis [1], energy consumption management [2], and student performance prediction [3, 4] are a few of many interdisciplinary applications of multivariate regression for predictive analysis. As discussed later, these techniques can also be applied to prediction problems related to high performance computing (HPC) file input/output (I/O) [5], Parkinson's patient clinical evaluations [6], and forest fire risk assessment [7].

Regression and interpolation have a considerable theoretical base in one dimension [8]. Splines in particular are well understood as an interpolation technique in one dimension [9], particularly B-splines. Fewer techniques have been discussed and evaluated for two dimensional problems. Though in two and three dimensions tensor product splines remain computable [10]. Tensor products have an unfortunate exponential scaling in parameterization with increasing dimension. Exponential scaling prohibits tensor products from being reasonably applied beyond three-dimensional data. In order to address this dimensional scaling challenge, C. de Boor and others proposed box splines [11], of which one of the approximation techniques in this work is composed [12].

The theoretical foundation of low dimension interpolation allows the construction of strong error bounds that are absent from high dimension problems. This work extends some known results regarding the secant method [13] to construct an interpolation error bound for problems of arbitrary dimension. These error bounds are useful, considering the same cannot be said for regression algorithms in general. The maximum complexity of an interpolant is bounded by the amount of data available, while the maximum complexity of a regressor is bounded by both the amount of data and the chosen parametric form. For this reason, generic uniform bounds are largely unobtainable for regression techniques on arbitrary approximation problems, even when the approximation domain is bounded.

Aside from theoretical motivation for the use of interpolants, there are often computational advantages as well. Many interpolants do not have the need for *fitting* data, or minimizing error with respect to parameters. In applications where the amount of data is large and the relative number of predictions that need to be made for a given collection of data is small, the direct application of an interpolant is much less computationally expensive.

In this work, multivariate interpolation is defined given a metric space Y with metric s that is closed under convex combinations when there exists some function $f : \mathbb{R}^d \to (Y, s)$ and a set of points $X = \{x^{(1)}, \ldots, x^{(n)}\} \subset \mathbb{R}^d$, along with associated function values $f(x)$ for all $x \in X$. The goal is to construct an approximation $\hat{f} : \mathbb{R}^d \to (Y, s)$ such that $\hat{f}(x) = f(x)$ for all $x \in X$. It is often the case that the form of the true underlying function f is unknown, however it is still desirable to construct an approximation $\hat{f}$ with minimum approximation error at $y \notin X$. The two metric spaces that will be discussed in this work are the real numbers under absolute difference and the space of Cumulative Distribution Functions (CDFs) under the Kolmogorov-Smirnov (KS) statistic.

Multivariate regression is often used when the underlying function is presumed to be stochastic, or stochastic error is introduced in the evaluation of f. Hence, multivariate 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 $\min_P \|s(\hat{f}(x^{(1)}), f(x^{(1)})), \ldots, s(\hat{f}(x^{(m)}), f(x^{(m)}))\|$, where

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

$\|\cdot\|$ is an appropriate measure. The difficult question in the case of regression is often what parametric form to adopt for any given application.

The most challenging problem when scaling in dimension is that the number of possible interactions between dimensions grows exponentially. Quantifying all possible interactions becomes intractable and hence beyond three-dimensional data, mostly linear models are used. That is not to say nonlinear models are absent, but nonlinearities are often either preconceived or model pairwise interactions between dimensions at most. Even globally nonlinear approximations such as neural networks are constructed from compositions of summed low-interaction functions [14].

Provided the theoretical and practical motivations for exploring interpolants, this work aims to study the empirical performance differences between a set of scalable interpolation techniques and a set of common regression techniques. These techniques are applied to a collection of moderately high dimension problems ($d \in [5, 50]$) and the empirical results are discussed.

The remainder of this paper is organized as follows. Section **??** presents the multivariate models. Section **???** presents the error measuring methodology that is used for collecting empirical results. Section **???** presents the theoretical error bounds for interpolation. Section **???** presents the data sets for empirical approximation analysis and presents the approximation results for all models on these data sets. Section **???** analyzes and discusses the results, their implications, and their limitations. Finally, Section **???** concludes.

1.1 Multivariate Regression

Multivariate regressors are capable of accurately modeling a complex dependence of a response (in Y) on multiple variables (represented as a points in $\mathbb{R}^d$). The approximations to some (unknown) underlying function $f : \mathbb{R}^d \to Y$, s are chosen to minimize some error measure related to data samples $f(x^{(i)})$. For example, least squares regression uses the sum of squared differences between modeled function values and true function values as an error measure. In this section and the next, some techniques are limited to approximating single real values. These techniques are extended to real vector-valued functions by repeating the construction for each component of the vector outputs.

1.1.1 Multivariate Adaptive Regression Splines. This approximation was introduced in [15] and subsequently improved to its current version in [16], called fast multivariate adaptive regression splines (Fast MARS). In Fast MARS, a least squares fit model is iteratively built by beginning with a single constant valued function and adding two new basis functions at each iteration of the form

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

where j 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 overall computational complexity of Fast MARS is $O(ndm^3)$ where m is the maximum number of underlying basis functions. This paper uses an implementation of Fast MARS [17] with $m = 200$.

1.1.2 Multilayer Perceptron Regressor. The neural network is a well studied and widely used method for both regression and classification tasks [18]. When using the rectified linear unit (ReLU) activation function [19] and training with the BFGS minimization technique [20], the model built by a multilayer perceptron uses layers $l : \mathbb{R}^i \to \mathbb{R}^j$ defined by

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

where W_l is the i by j weight matrix for layer l. In this form, the multilayer perceptron (MLP) produces a piecewise linear model of the input data. The computational complexity of training a multilayer perceptron is $O(ndm)$, where m is determined by the sizes of the layers of the network and the stopping criterion of the BFGS minimization used for finding weights. This paper uses the scikit-learn MLP regressor [21], a single hidden layer with 100 nodes, ReLU activation, and BFGS for training.

1.1.3 Support Vector Regressor. Support vector machines are a common method used in machine learning classification tasks that can be adapted for the purpose of regression [22]. How to build a support vector regressor (SVR) is beyond the scope of this summary, but the resulting functional fit $p : \mathbb{R}^d \to \mathbb{R}$ has the form

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

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 $O(n^2dm)$, with m being determined by the minimization convergence criterion. This paper uses the scikit-learn SVR [21] with a polynomial kernel function.

1.2 Multivariate Interpolation

The following interpolation techniques demonstrate a reasonable variety of approaches to interpolation. All of the presented interpolants produce approximations that are continuous in value, which is often a desirable property in applied approximation problems.

1.2.1 Linear Shepard. The linear Shepard method (LSHEP) is a blending function using local linear interpolants, a special case of the general Shepard algorithm [23]. The interpolant 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 $O(n^2d^3)$. This paper uses the Fortran#95 implementation of LSHEP in SHEPPACK [23].

1.2.2 Delaunay. The Delaunay triangulation is a well-studied geometric technique for producing an interpolant [24]. The Delaunay triangulation of a set of data points into simplices is such that there are no data points inside the sphere defined by the vertices of each simplex. For a d-simplex S with vertices $x^{(0)}, x^{(1)}, \ldots, x^{(d)}$, and function values $f(x^{(i)})$, $i = 0, \ldots, d$, $y \in S$ is a unique convex combination of the vertices,

$$y = \sum_{i=0}^{d} w_i x^{(i)}, \quad \sum_{i=0}^{d} w_i = 1, \quad w_i \geq 0, \quad i = 0, \ldots, d,$$

and the Delaunay interpolant $\hat{f}(y)$ at y is given by

$$\hat{f}(y) = \sum_{i=0}^{d} w_i f(x^{(i)}).$$

The computational complexity of Delaunay interpolation (for the implementation used) is $O(nd^4 \log d)$, which is capable of scaling reasonably to $d \leq 50$. In the present application, a Delaunay simplex S containing y is found, then the $d+1$ vertices (points in X) of S are used to assign weights to each vertex and produce the predicted function value for point y.

1.2.3 Iterative Box Mesh. The Iterative Box mesh is an interpolation technique that utilizes overlapping box splines [11] as basis functions that are shifted and scaled to have support over box shaped regions. The boxes are constructed in a way to guarantee a covering of the domain. Given a set of box splines $\left\{b^{x^{(i)}}\right\}$ with the iterative box properties anchored at interior points $\left\{x^{(i)}\right\}$,

$$\hat{f}(y) = \frac{\sum_i b^{x^{(i)}}(y) f(x^{(i)})}{\sum_i b^{x^{(i)}}(y)}.$$

Note that the box splines always satisfy $b^{x^{(i)}}(y) \geq 0$. The computational complexity of interpolation via the Iterative Box mesh is $O(n^2 d)$.

1.2.4 Voronoi Mesh. A well-studied technique for classification and approximation is the nearest neighbor algorithm [25]. Nearest neighbor inherently utilizes the convex region $C^{x^{(i)}}$ (Voronoi cell [26]) consisting of all points closer to $x^{(i)}$ than any other point $x^{(j)}$. The Voronoi mesh smooths the nearest neighbor approximation by utilizing the Voronoi cells to define support via a generic basis function $v : \mathbb{R}^d \to \mathbb{R}_+$ given by

$$v^{x^{(i)}}(y) = \left(1 - \frac{||y - x^{(i)}||_2}{2\, h(y - x^{(i)} \mid x^{(i)})}\right)_+,$$

where $h(w \mid x^{(i)})$ is the distance between $x^{(i)}$ and the boundary of the Voronoi cell $C^{x^{(i)}}$ in the direction w. $v^{x^{(i)}}\left(x^{(j)}\right) = 0$ and $v^{x^{(i)}}$ has local support, giving the interpolated value

$$f(y) = \frac{\sum_i v^{x^{(i)}}(y) f(x^{(i)})}{\sum_i v^{x^{(i)}}(y)},$$

where $0 \leq v^{x^{(i)}}(y) \leq 1$. The computational complexity of interpolation via the Voronoi mesh is $O(n^2 d)$. All of the approximations are an interpolant involving a convex combination of known function values $f(x^{(i)})$.

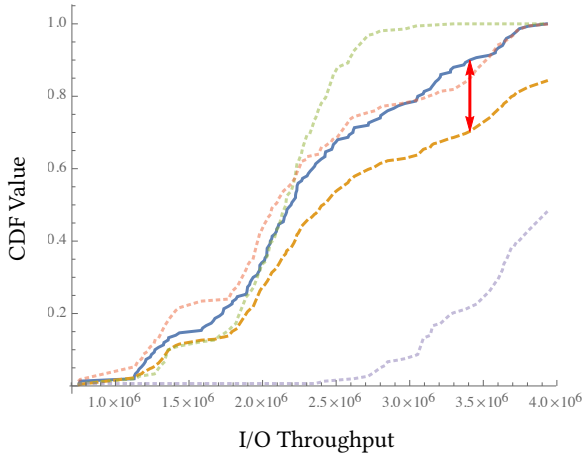

Figure 1: In this HPC I/O example, the general methodology for predicting a CDF and evaluating error can be seen. The Delaunay method chose three source distributions (dotted lines) and assigned weights {.3, .4, .3} (top to bottom at red arrow). The weighted sum of the three known CDFs produces the predicted CDF (dashed line). The KS Statistic (red arrow) computed between the true CDF (solid line) and predicted CDF (dashed line) is 0.2 for this example. The KS test null hypothesis is rejected by p-value 0.01, however it is not rejected by p-value 0.001.

1.3 Measuring Error

When the range of an approximation is the real numbers, error is reported with summary statistics including: min absolute error, max absolute error, absolute error quartiles, and mean absolute error. In general, all errors are recorded as signed relative error and any figures depicting distributions of errors will present these values.

A hurdle when modeling function-valued outputs such as Cumulative Distribution Functions (CDFs) or Probability Density Functions (PDFs) is that certain properties must be maintained. It is necessary that a PDF $f : \mathbb{R} \to \mathbb{R}$ have the properties $f(x) \geq 0$ and $\int_{-\infty}^{\infty} f(x)dx = 1$. However, for a CDF $F : \mathbb{R} \to \mathbb{R}$ the properties are $F(x) \in [0, 1]$ and $F(x)$ is right continuous and nondecreasing. This work utilizes the fact that a convex combination of CDFs (or PDFs) results in a valid CDF. Given $G(x) = \sum_i w_i F_i(x)$, $\sum_i w_i = 1$, $w_i \geq 0$, and each F_i is a valid CDF, G must also be a valid CDF. A demonstration of how this is applied can be seen in Figure 1.

The performance of approximation techniques that predict probability functions can be analyzed through a variety of summary statistics. Mean error, mean absolute error, mean squared error, and the max absolute error are all popular measures. This work uses the max absolute error, also known as the Kolmogorov-Smirnov (KS) statistic [27] for its compatibility with the KS test.

The two-sample KS test is a useful nonparametric test for comparing two CDFs while only assuming stationarity, finite mean, and

finite variance. The null hypothesis (that two CDFs come from the same underlying distribution) is rejected at level $p \in [0, 1]$ when

$$KS > \sqrt{-\frac{1}{2}\ln\left(\frac{p}{2}\right)}\sqrt{\frac{1}{n_1} + \frac{1}{n_2}},$$

with distribution sample sizes $n_1, n_2 \in \mathcal{N}$. For all applications of the KS test presented in this work $n_1 = n_2$. An example of the round-trip prediction methodology from known and predicted distributions to the calculation of error can be seen in Figure 1.

2 THEORETICAL ERROR BOUND

This section presents the theoretical results bounding the error of Delaunay interpolation. The error analysis relies on the Delaunay interpolation for two reasons: (1) second order results can be obtained utilizing a Lipschitz constant on the gradient of a function, rather than standard Lipschitz bounds and (2) multiple other interpolants analyzed compute predictions as convex combinations of observed function values, which may allow for straight-forward extensions of this error bound.

Lemma 1. Let $S \subset \mathbb{R}^d$ be open and convex, $f : \mathbb{R}^d \to \mathbb{R}$, and $\nabla f \in Lip_{(\gamma, \|\cdot\|_2)}(S)$, the set of γ-Lipschitz continuous functions in the 2-norm. Then for all $x, y \in S$

$$\left| f(y) - f(x) - \langle \nabla f(x), y - x \rangle \right| \leq \frac{\gamma \|y - x\|_2^2}{2}.$$

Proof. Consider the function $g(t) = f\Big((1-t)x + ty\Big)$, $0 \leq t \leq 1$, whose derivative $g'(t) = \Big\langle \nabla f\Big((1-t)x + ty\Big), y - x \Big\rangle$ is the directional derivative of f in the direction $(y - x)$.

$$\begin{aligned}
&\left| f(y) - f(x) - \langle \nabla f(x), y - x \rangle \right| \\
&\quad = \left| g(1) - g(0) - g'(0) \right| \\
&\quad = \left| \int_0^1 g'(t) - g'(0)\, dt \right| \\
&\quad \leq \int_0^1 \left| g'(t) - g'(0) \right| dt \\
&\quad = \int_0^1 \left| \Big\langle \nabla f\Big((1-t)x + ty\Big) - \nabla f(x), y - x \Big\rangle \right| dt \\
&\quad \leq \int_0^1 \left\| \nabla f\Big((1-t)x + ty\Big) - \nabla f(x) \right\|_2 \|y - x\|_2\, dt \\
&\quad \leq \int_0^1 \Big(\gamma\, \|y - x\|_2\Big)\, \Big(\|y - x\|_2\Big) t\, dt \\
&\quad = \frac{\gamma \|y - x\|_2^2}{2}.
\end{aligned}$$

□

Lemma 2. Let $x, y, v_i \in \mathbb{R}^d$, $c_i \in \mathbb{R}$, and $|\langle y - x, v_i \rangle| \leq c_i$ for $i = 1, \ldots, d$. If $M = (v_1, \ldots, v_d)$ is nonsingular, then

$$\|y - x\|_2^2 \leq \frac{1}{\sigma_d^2} \sum_{i=1}^{d} c_i^2,$$

where σ_d is the smallest singular value of M.

Proof. Using the facts that M and M^t have the same singular values, and $\|M^t w\|_2 \geq \sigma_d \|w\|_2$, gives

$$\begin{aligned}
\|y - x\|_2^2 &\leq \frac{\|M^t (y - x)\|_2^2}{\sigma_d^2} \\
&= \frac{1}{\sigma_d^2} \sum_{i=1}^{d} \langle y - x, v_i \rangle^2 \\
&\leq \frac{1}{\sigma_d^2} \sum_{i=1}^{d} c_i^2.
\end{aligned}$$

□

Lemma 3. Given f, γ, S as in *Lemma 1*, let $X = \{x_0, x_1, \ldots, x_d\} \subset S$ be the vertices of a d-simplex, and let $\hat{f}(x) = \langle c, x - x_0 \rangle + f(x_0)$, $c \in \mathbb{R}^d$ be the linear function interpolating f on X.

Let σ_d be the smallest singular value of the matrix $M = (x_1 - x_0, \ldots, x_d - x_0)$, and $k = \max\limits_{1 \leq j \leq d} \|x_j - x_0\|_2$. Then

$$\left\| \nabla f(x_0) - c \right\|_2 \leq \sqrt{d} \frac{\gamma k^2}{\sigma_d}.$$

Proof. Consider $f(x) - \hat{f}(x)$ along the line segment $z(t) = (1 - t)x_0 + t x_j$, $0 \leq t \leq 1$. By Rolle's Theorem, for some $0 < \hat{t} < 1$, $\Big\langle \nabla f\Big(z(\hat{t})\Big) - c, x_j - x_0 \Big\rangle = 0$. Now

$$\begin{aligned}
&\left| \Big\langle \nabla f(x_0) - c, x_j - x_0 \Big\rangle \right| \\
&\quad = \left| \Big\langle \nabla f(x_0) - \nabla f\Big(z(\hat{t})\Big) + \nabla f\Big(z(\hat{t})\Big) - c, x_j - x_0 \Big\rangle \right| \\
&\quad = \left| \Big\langle \nabla f(x_0) - \nabla f\Big(z(\hat{t})\Big), x_j - x_0 \Big\rangle \right| \\
&\quad \leq \left\| \nabla f(x_0) - \nabla f\Big(z(\hat{t})\Big) \right\|_2 \|x_j - x_0\|_2 \\
&\quad \leq \gamma \|x_0 - z(\hat{t})\|_2\, \|x_j - x_0\|_2 \\
&\quad \leq \gamma \|x_j - x_0\|_2^2 \leq \gamma k^2,
\end{aligned}$$

for all $1 \leq j \leq d$. Using *Lemma 2*,

$$\left\| \nabla f(x_i) - c \right\|_2^2 \leq \frac{d}{\sigma_d^2} \Big(\gamma k^2\Big)^2 \implies \left\| \nabla f(x_i) - c \right\|_2 \leq \sqrt{d} \frac{\gamma k^2}{\sigma_d}.$$

□

Theorem. Under the assumptions of *Lemma 1* and *Lemma 3*, for $z \in S$,

$$\left| f(z) - \hat{f}(z) \right| \leq \frac{\gamma \|z - x_0\|_2^2}{2} + \sqrt{d} \frac{\gamma k^2}{\sigma_d} \|z - x_0\|_2.$$

Proof. Let $v = \nabla f(x_0) - c$, where $\|v\|_2 \leq \sqrt{d} \gamma k^2 / \sigma_d$ by *Lemma 3*. Now

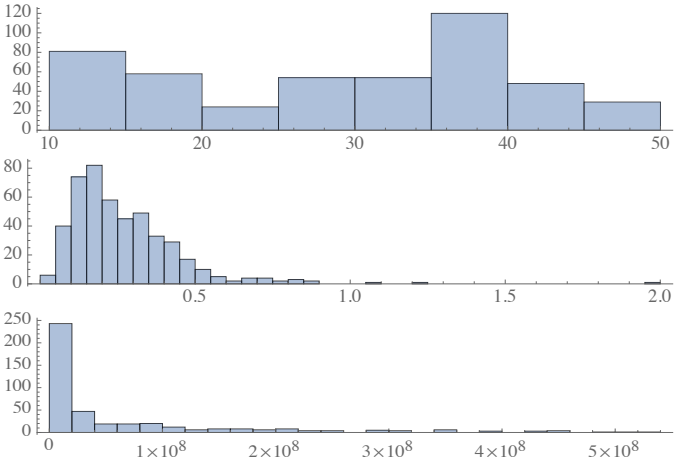

Figure 2: Histograms of Parkinsons (total UPDRS), forest fire (area), and HPC I/O (mean throughput) response values respectively. Notice that both the forest fire and HPC I/O data sets are heavily skewed.

$$
\begin{aligned}
\left| f(z) - \hat{f}(z) \right| &= \left| f(z) - f(x_0) - \langle c, z - x_0 \rangle \right| \\
&= \left| f(z) - f(x_0) - \langle \nabla f(x_0) - v, z - x_0 \rangle \right| \\
&= \left| f(z) - f(x_0) - \langle \nabla f(x_0), z - x_0 \rangle + \langle v, z - x_0 \rangle \right| \\
&\leq \left| f(z) - f(x_0) - \langle \nabla f(x_0), z - x_0 \rangle \right| + \left| \langle v, z - x_0 \rangle \right| \\
&\leq \left| f(z) - f(x_0) - \langle \nabla f(x_0), z - x_0 \rangle \right| + \|v\|_2 \, \|z - x_0\|_2 \\
&\leq \left| f(z) - f(x_0) - \langle \nabla f(x_0), z - x_0 \rangle \right| + \frac{\gamma k^2 \sqrt{d}}{\sigma_d} \|z - x_0\|_2 \\
&\leq \frac{\gamma \|z - x_0\|_2^2}{2} + \sqrt{d} \frac{\gamma k^2}{\sigma_d} \|z - x_0\|_2,
\end{aligned}
$$

where the last inequality follows from *Lemma 1.*

□

3 DATA AND ANALYSIS

In order to evaluate the proposed interpolation and approximation techniques, this paper utilizes three data sets of varying dimension and application. In the following subsections the sources and targets of each data set are described, as well as challenges and limitations related to interpolating and approximating these data sets. The distributions of response values being modeled can be seen in Figure 2. The preprocessing and approximation processes are described in Section 3.2.

3.1 Data Summary

3.1.1 High Performance Computing I/O ($n = 532, d = 4$). The first of three data sets is a four-dimensional data set produced by executing the IOzone benchmark from [28] on a homogeneous cluster of computers. The system performance data was collected by executing IOzone 40 times for each of a select set of system configurations. A single IOzone execution reports the max I/O file-read throughput seen. The 40 executions for each system configuration are converted to their mean, which is capable of being modeled by each of the multivariate approximation techniques presented in Section **??**. The four dimensions being modeled to predict throughput mean are file size, record size, thread count, and CPU frequency.

Data Set	Technique	Tolerance	Average Error
HPC I/O	MBM	1.2	0.597
Forest Fire	MBM	1.8	3.517
Parkinson's	MBM	0.6	0.114
HPC I/O	IBM	0.4	0.419
Forest Fire	IBM	1.8	3.615
Parkinson's	IBM	1.8	0.121
HPC I/O	VM	0.2	0.382
Forest Fire	VM	1.0	4.783
Parkinson's	VM	2.0	1.824

Table 1: The optimal error tolerance bootstrapping parameters for each technique and each data set as well as the average absolute relative errors achieved by that tolerance. Notice that large relative error tolerances occasionally yield even lower evaluation errors, demonstrating the benefits of approximation over interpolation for noisy data sets.

3.1.2 Forest Fire ($n = 517, d = 12$). The forest fire data set [7] describes the area of Montesinho park burned on specific days and months of the year in terms of the environmental conditions. The twelve dimensions being used to model burn area are the x and y spatial coordinates of burns in the park, month and day of year, the FFMC, DMC, DC, and ISI indices (see source for details), the temperature in Celsius, relative humidity, wind speed, and outdoor rain. The original analysis of this data set demonstrated it to be difficult to model, likely due to the skew in response values.

3.1.3 Parkinson's Telemonitoring ($n = 468, d = 16$). The final data set for evaluation [6] is derived from a speech monitoring study with the intent to automatically estimate Parkinson's disease symptom development in Parkinson's patients. The function to be predicted is a time-consuming clinical evaluation measure referred to as the UPDRS score. The total UPDRS score given by a clinical evaluation is estimated through 16 real numbers generated from biomedical voice measures of in-home sound recordings.

3.2 Performance Analysis

The performance of the approximation techniques varies considerably across the three evaluation data sets. Relative errors for the most naïve approximators such as nearest neighbor can range from zero to $\left(\max_x f(x) - \min_x f(x)\right) / \min_x f(x)$ when modeling a positive function $f(x)$ from data. Each of the approximation techniques presented remain within these bounds and all errors are presented in signed relative form $(\hat{f}(x) - f(x))/f(x)$. Before the models are constructed all data values (components $x_r^{(i)}$ of $x^{(i)} \in X$) are shifted and scaled to be in the unit cube $[0, 1]^d$, while the response values are taken in their original form. All models are evaluated with 10 random 80/20 splits of the data.

Each of the approximation techniques presented incorporates bootstrapping based on an allowable error tolerance t. An analysis of the effects of bootstrapping error tolerances on validation accuracy can be seen in Figure 4. The approximation meshes perform

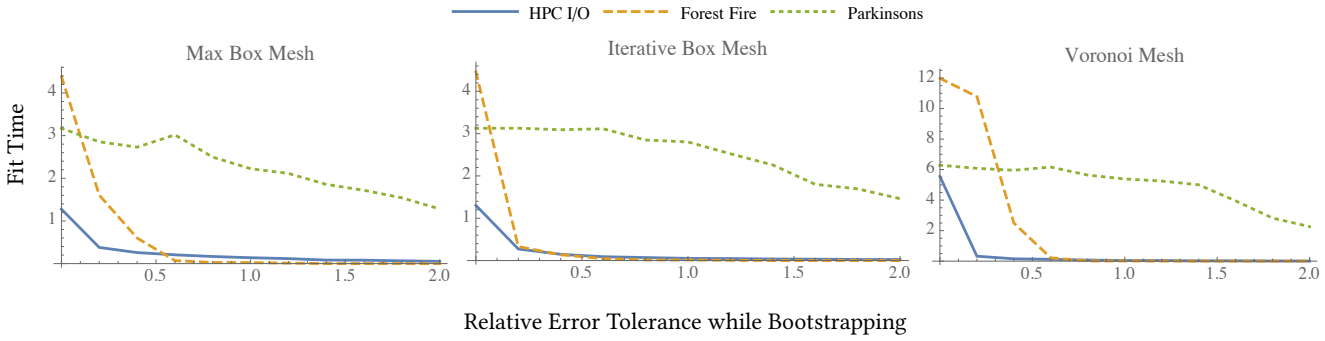

Figure 3: Time required to generate model fits for each technique with varying relative error tolerance during bootstrapping.

best on the forest fire and Parkinson's data sets when the error tolerance used for fitting is large (smoothing rather than interpolating), while near-interpolation generally produces the most accurate models for HPC I/O. Another performance result of note is that the *MBM* and *IBM* have very similar basis functions with largely different outputs.

The selection of bootstrapping error tolerance also effects the computation time required to fit each of the models to data. Figure 3 presents the time required to construct approximations for each model and each data set with varying t. The rapid reduction in computation time required for the forest fire and HPC I/O data sets suggests that large reductions in error can be achieved with relatively few basis functions. The Parkinson's data set however presents a more noisy response, with increasing number of basis functions reducing error much less quickly.

The distributions of errors experienced by each approximation technique when the optimal bootstrapping relative error tolerance is selected can be seen in Figure 5. HPC I/O exhibits the most normal approximation errors, which suggests that the models are converging on the random noise of the response for the data set. The worst relative approximation errors are produced by the Voronoi mesh on the forest fire data set. The small magnitude true response values contribute to the larger relative errors. Regardless, the *VM* errors are unacceptably large.

4 DISCUSSION

The bootstrapping procedure presented for each approximation technique still has much room for improvement. Initial analysis suggests that the appropriate relative error tolerance needs to be discovered empirically for each application of a modeling technique. Further analytic studies could arrive at methods for determining optimal error tolerances at runtime, however increases in runtime complexity may not be afforded in many applications.

The box-shaped basis functions and the construction algorithms used for the *MBM* and *IBM* could become a source of error when d (the dimension of the data X) is comparable to n (the number of known points). The blending regions in which multiple basis functions overlap are always axis aligned and in applications such as image analysis, any single dimension may be unsuitable for approximating the true underlying function. The Voronoi mesh attempts to address this problem by utilizing boundaries between points in multiple dimensions simultaneously. However, it is empirically unclear whether the true benefits of the *VM* are seen in applications where $d \ll n$.

Each of the case studies presented have fewer than 1000 points. The complexities of the presented approximation techniques are suitable for large dimension, but the increased complexity associated with brute-force bootstrapping currently prohibits their use on larger data sets. The Voronoi mesh in particular has a large complexity with respect to n which could be significantly improved via more greedy bootstrapping. While each technique requires less than ten seconds on average to produce a fit in the presented case studies, the fit time required quickly grows into minutes around 1000 points. While these initial results appear somewhat limiting, they demonstrate the viability of each mesh and leave room for further theoretical exploration of techniques to reduce the runtime complexity while maintaining the approximation power and flexibility.

5 CONCLUSION

The Max Box Mesh, Iterative Box Mesh, and Voronoi Mesh each provide novel strategies for effectively approximating multivariate phenomonon. The underlying constructions are theoretically straightforward, yet powerful and flexible. The computational complexities of each make them particularly suitable for applications in many dimensions, while the bootstrapping error tolerance parameter allows a balance between smoothing and interpolation to be explored empirically with each application.

5.1 Future Work

A thorough comparison with constituent multivariate approximation techniques including but not limited to, linear Shepard interpolation, multivariate adaptive regression splines, multilayer perceptron regression, and Delaunay triangulation constitutes future work. A more detailed study of alternative bootstrapping techniques may also provide valuable insight.

6 DATA

This paper presents a variability modeling case study with a five-dimensional dataset produced by executing the IOzone benchmark [28] on a homogeneous cluster of computers. Each node contains two Intel Xeon E5-2637 CPUs offering a total of 16 CPU cores with

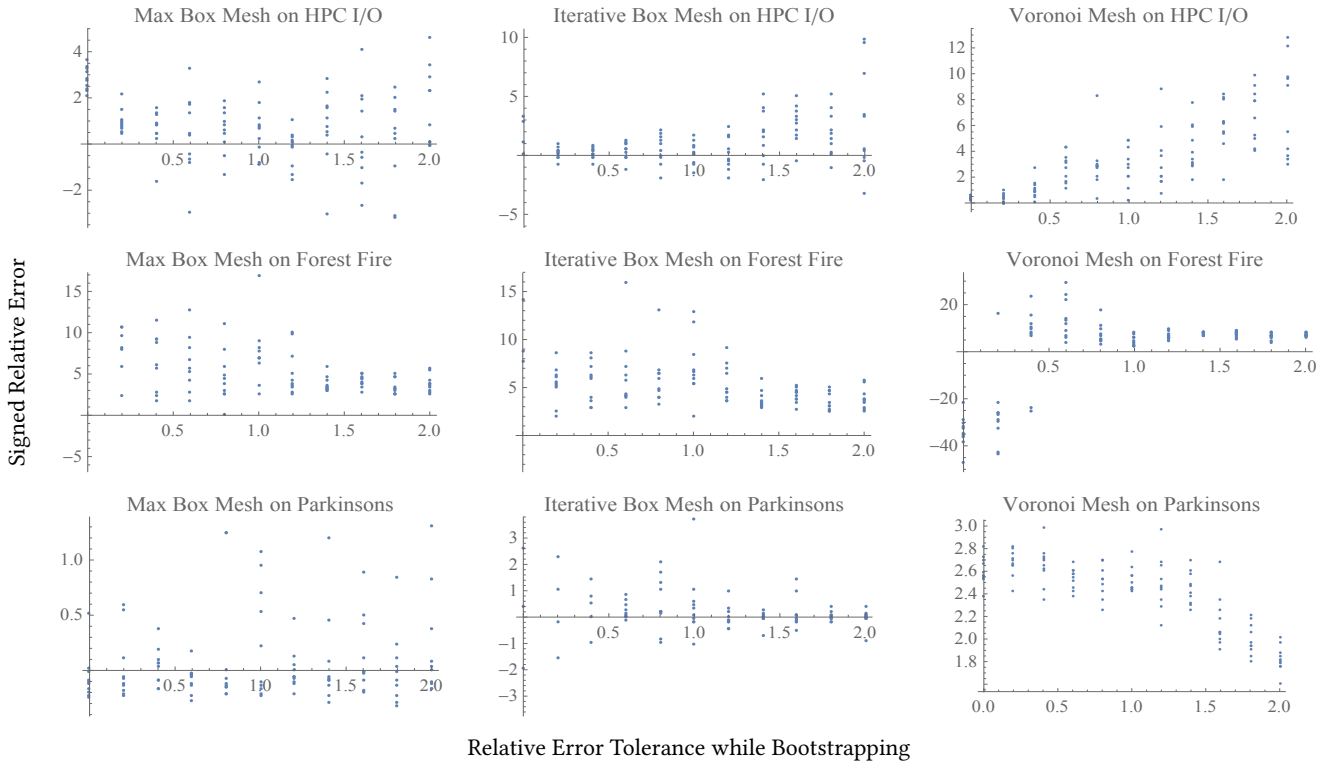

Figure 4: The performance of all three techniques with varied relative error tolerance for the bootstrapping parameter. The columns are for Max Box Mesh, Iterative Box Mesh, and Voronoi Mesh, respectively. The rows are for HPC I/O, Forest Fire, and Parkinson's respectively. Notice the techniques' behavior on the Parkinson's and Forest Fire data sets, performance increases with larger error tolerance.

System Parameter	Values
File Size (KB)	4, 16, 64, 256, 1024, 4096, 8192, 16384
Record Size (KB)	4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096, 8192, 16384
Thread Count	1, 8, 16, 24, 32, 40, 48, 56, 64
Frequency (GHz)	1.2, 1.6, 2, 2.3, 2.8, 3.2, 3.5
Test Type	Readers, Rereaders, Random Readers, Initial Writers, Rewriters, Random Writers

Table 2: A description of system parameters considered for IOzone. Record size must be $\leq$ file size during execution.

16GB of DRAM. While the CPU frequency varies depending on the test configuration, the I/O from IOzone is performed by an ext4 filesystem sitting above an Intel SSDSC2BA20 SSD drive. At the time of data collection, Linux kernel Version 4.13.0 was used. The system performance data was collected over two weeks by executing IOzone 150 times for each of a select set of approximately 18K system configurations, for a total of approximately 2.7M executions of IOzone. A single IOzone execution reports the max I/O throughput in kilobytes per second seen for the selected test type. The summary of the data components in $x^{(i)}$ for the experiments for this paper can be seen in Table 2. Distributions of raw throughput values being modeled can be seen in Figure 6.

Some mild preprocessing was necessary to prepare the data for modeling and analysis. All features were shifted by their minimum value and scaled by their range, mapping each feature independently into $[0, 1]$. This normalization ensures each feature is treated equally by the interpolation techniques and should be performed on all data before building models and making predictions regardless of application. All 150 repeated trials for a system configuration were grouped with that configuration. The only nonordinal feature in this data is the test type. All test types were treated as different applications and were separated for modeling and analysis, i.e., predictions for the "readers" test type were made using only known configurations for the "readers" test type.

7 RESULTS

All three interpolation techniques are used to predict the distribution of I/O throughput values at previously unseen system configurations. In order to improve robustness of the error analysis, ten random selections of 80% of the IOzone data are used to train each model and the remaining 20% provide approximation error for each model. The recorded errors are grouped by unique system configuration and then averaged within each group. The samples are identical for each interpolation technique, ensuring consistency in the training and testing sets.

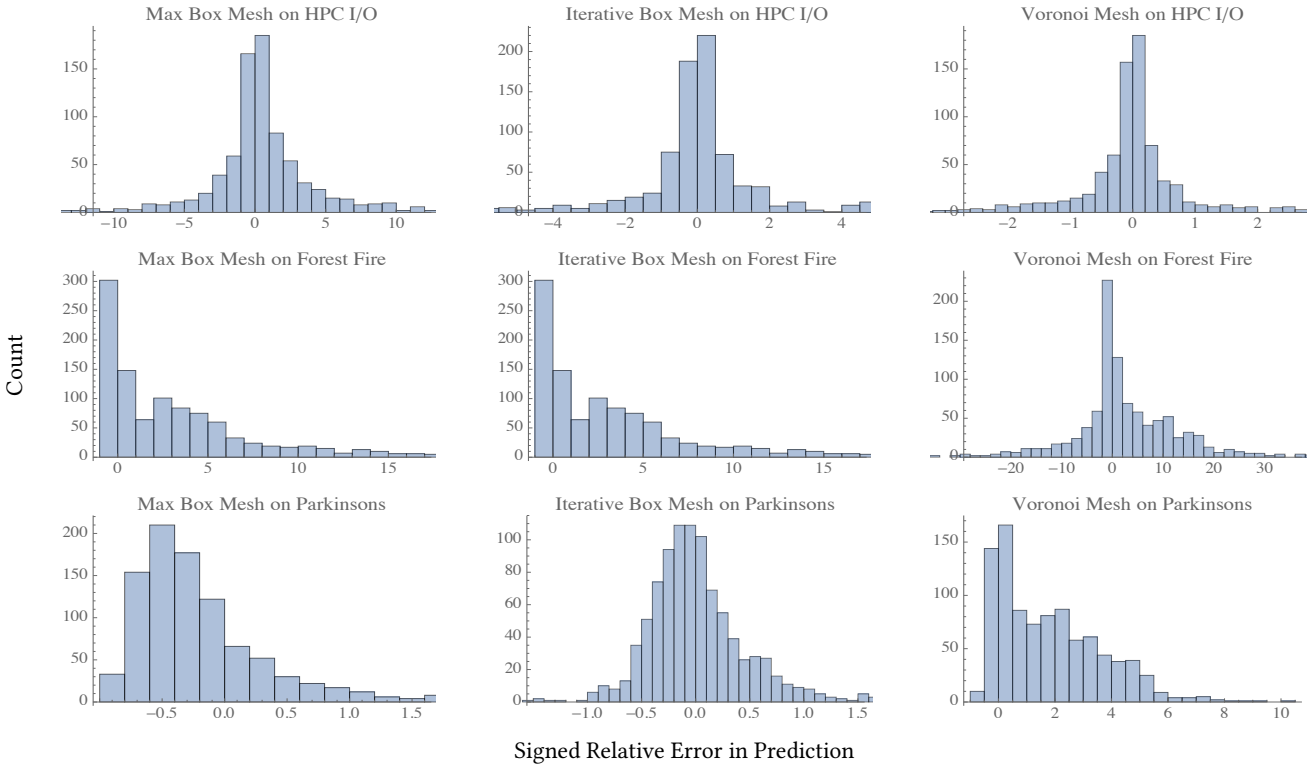

Figure 5: A sample of relative errors for all three techniques with optimal selections of error tolerance. The columns are for Max Box Mesh, Iterative Box Mesh, and Voronoi Mesh, respectively. The rows are for HPC I/O, Forest Fire, and Parkinson's respectively.

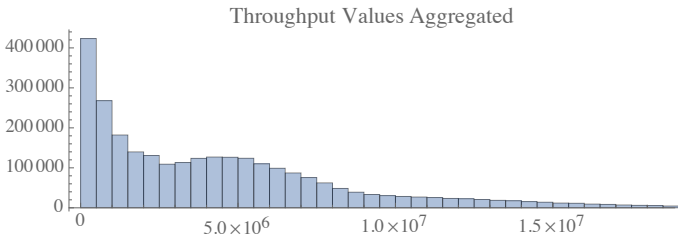

Figure 6: Histogram of the raw throughput values recorded during all IOzone tests across all system configurations. The distribution is skewed right, with few tests having significantly higher throughput than most others.

The aggregation of errors across all IOzone tests given 80% of the data as training can be seen in Figure 7. Agglomerate errors for each technique resemble a Gamma distribution. The percentages of significant prediction errors with varying p-values are on display in Table 3. The primary p-value used for analyses in this work is 0.001, chosen because close to 2K predictions are made for each test type. Also, applications executed in cloud and HPC systems that could benefit from statistical modeling will be executed at least thousands of times. In line with this knowledge, it is important to ensure that only a small fraction of interpretable results could occur solely under the influence of random chance. When considering the $p = 0.001$ results for each technique, a little under half of the predicted CDFs are significantly different from the measured (and presumed) correct CDFs. A rejection rate of 45% would seem a poor result, however in this situation the complexity of the problem warrants a slightly different interpretation. These predictions are a very *precise* characterization of performance variability, in fact the cumulative distribution function of a random variable is the strongest possible characterization of variability that can be predicted. Globally, only a little under half of the predictions fail to capture *all* of the characteristics of performance variability at new system configurations. It is also demonstrated later in this Section that this result can likely be improved.

While interpreting null hypothesis rejection rates for these interpolation techniques, it is important to consider how the rejection rate reduces with increasing amounts of training data. Figure 8 displays the change in $p = 0.001$ null hypothesis rejection rate with increasing density of training data up to the maximum density allowed by this set. Delaunay interpolation provides the best results with the least training data by about 5%, but these low density rejection rates are unacceptably high (90%). Figure 8 clearly shows that this data set and/or the system variables used in the models of performance variability is inadequate to capture the full variability map from system parameters to performance CDF. Which or both obtains is not clear. A few well chosen data points can significantly improve the interpolants, and thus a careful study of the rejection

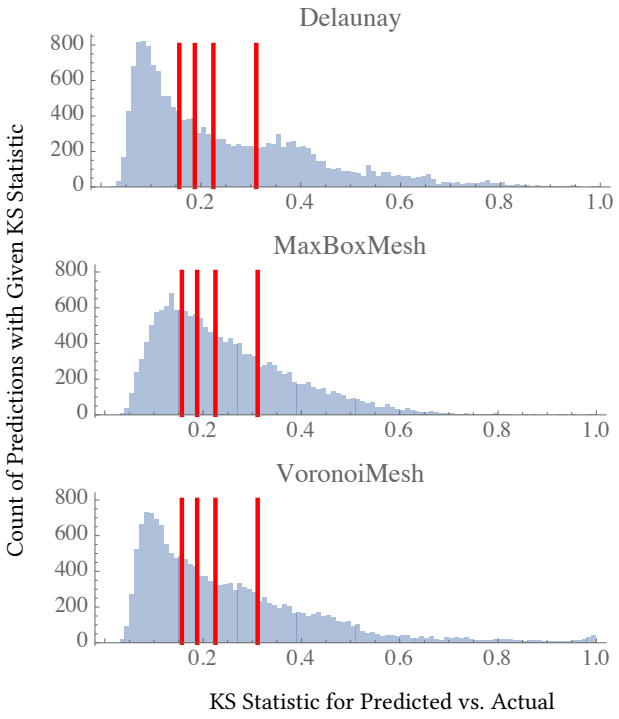

Figure 7: Histograms of the prediction error for each modeling algorithm from ten random splits when trained with 80% of the data aggregated over all different test types. The distributions show the KS statistics for the predicted throughput distribution versus the actual throughput distribution. The four vertical red lines represent commonly used p-values {0.05, 0.01, 0.001, 1.0e-6} respectively. All predictions to the right of a red line represent CDF predictions that are significantly different (by respective p-value) from the actual distribution according to the KS test.

Algorithm	P-Value	% N.H. Rejections
Delaunay	.05	58.4
Max Box Mesh		69.3
Voronoi Mesh		61.9
Delaunay	.01	51.1
Max Box Mesh		58.4
Voronoi Mesh		53.4
Delaunay	.001	44.1
Max Box Mesh		46.9
Voronoi Mesh		45.1
Delaunay	1.0e-6	31.4
Max Box Mesh		26.6
Voronoi Mesh		28.7

Table 3: Percent of null hypothesis rejections rate by the KS-test when provided different selections of p-values. These accompany the percent of null hypothesis rejection results from Figure 7.

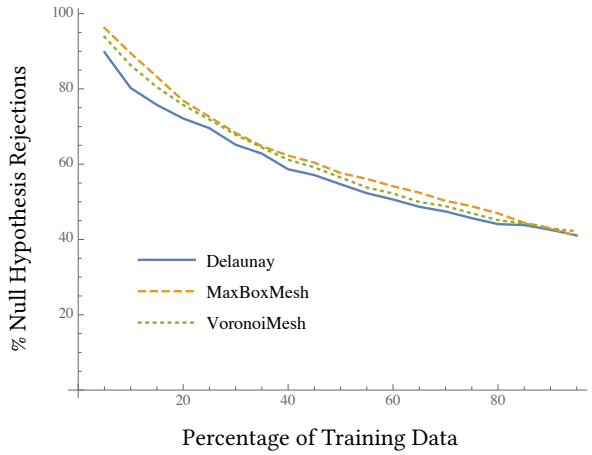

Figure 8: The performance of each algorithm on the KS test ($p = 0.001$) with increasing amounts of training data averaged over all IOzone test types and ten random splits of the data. The training percentages range from 5% to 95% in increments of 5%. Delaunay is the best performer until 95% of data is used for training, at which Max Box mesh becomes the best performer by a fraction of a percent.

instances is warranted, besides enlarging the set of system variables being modeled.

It may be misleading to consider the global performance of each prediction technique across all test types, as some test types are more difficult than others to predict and have more apparent latent variables. In Figure 9, the relative difficulty of each IOzone test type can be compared. The I/O test types analyzing reads are typically approximated with lower error than those test types analyzing writes. Regardless of test type, in the aggregate results the KS statistics hover consistently around 0.15, demonstrating an impressively low KS statistic for predictions. In order to address the opacity of aggregate analysis, another case study and an application of the methodology from Section **??** is presented in Table 4.

The results presented in Table 4 are achieved by permitting each approximation technique 300 iterations of simulated annealing. In each iteration, the impact of potential weights on the average KS statistic were considered. All weights were kept in the range [0,2], and were applied to the normalized features for frequency, file size, record size, and number of threads. All three approximation techniques had similar optimal weights achieved by simulated annealing of approximately $(.001, 2, 1.7, 1.5)$ for frequency, file size, record size, and number of threads, respectively. Recall that each interpolation technique uses small distances to denote large influences on predictions, meaning that frequency was the most important feature when predicting variability for the "readers" test type, followed not-so-closely by number of threads, then record size.

The "readers" test type results demonstrate that the underlying prediction techniques work and are capable of seeing rejection rates below 5% when tuned for a given application. It is important to emphasize that the roughly 95% of predictions for which the null hypothesis was not rejected are predicting the *precise* distribution of I/O throughput that will be witnessed at a previously unseen

Algorithm	P-Value	Unweighted % N.H. Rejection	Weighted % N.H. Rejection
Delaunay	.05	24.9	30.2
Max Box Mesh		21.3	21.2
Voronoi Mesh		18.7	11.3
Delaunay	.01	21.6	27.4
Max Box Mesh		16.4	16.4
Voronoi Mesh		14.9	7.0
Delaunay	.001	19.7	25.4
Max Box Mesh		13.1	13.1
Voronoi Mesh		12.3	4.6
Delaunay	1.0e-6	17.9	23.4
Max Box Mesh		11.3	11.3
Voronoi Mesh		8.5	2.3

Table 4: The null hypothesis rejection rates for various p-values with the KS-test. These results are strictly for the "readers" IOzone test type and show unweighted results as well as the results with weights tuned for minimum error (KS statistic) by 300 iterations of simulated annealing. Notice that the weights identified for the Delaunay model cause data dependent tuning, reducing performance. MaxBoxMesh performance is improved by a negligible amount. VoronoiMesh performance is notably improved.

system configuration. To the authors' knowledge, there is no existing methodology that is generically applicable to any system performance measure, agnostic of system architecture, and capable of making such powerful predictions.

8 DISCUSSION

The results of the IOzone case study indicate that predicting the CDF of I/O throughput at previously unseen system configurations is a challenging problem. The KS statistic captures the worst part of any prediction and hence provides a conservatively large estimate of approximation error. The average absolute errors in the predicted CDFs are always lower than the KS statistics. However, the KS statistic was chosen because of the important statistical theory surrounding it as an error measure. Considering this circumstance, a nonnegligible volume of predictions provide impressively low levels of error. Powerful predictive tools such as those presented in this work allow for more in-depth analysis of system performance variability. For example, system configurations that are most difficult to predict in these tests are likely "outlier" configurations that do not resemble those configurations that share many similar parameters. Analysis of these configurations may provide valuable insight into effective application specific operation of computer systems.

As mentioned in Section 1, no prior work has attempted to model an arbitrary performance measure for a system to such a high degree of precision. All previous statistical modeling attempts capture a few (< 3) ordinal performance measures. Generating models that have such high degrees of accuracy allows system engineers to identify previously unused configurations that present desired characteristics. Service level agreements (SLAs) in cloud computing environments are cause for capital competition that is affected heavily by system performance [29]. Users prefer SLAs that allow the most computing power per monetary unit, incentivizing service providers to guarantee the greatest possible performance. Overscheduling and irregular usage patterns force cloud service providers to occasionally overload machines, in which case precise models of system performance can be used to statistically minimize the probability of SLA violation. Similar targeted performance tuning techniques can be applied to HPC system configuration to maximize application throughput or minimize system power consumption.

A final application domain affected by this methodology is computer security. Collocated users on cloud systems have received attention recently [30]. If a malicious collocated user is capable of achieving specific insight into the configuration of the system, or the activity of other collocated users by executing performance evaluation programs (i.e., IOzone), a new attack vector may present itself. Malicious users could be capable of identifying common performance distributions of vulnerable system configurations and vulnerable active user jobs. This knowledge may allow targeted exploits to be executed. Light inspection of raw IOzone I/O throughputs provides substantial evidence that distinct performance distributions coincide closely with specific system configuration parameters. Conversely, a service provider may defend against such attacks by deliberately obfuscating the performance of the machine. Models such as those presented in this paper could identify optimal staggering and time-delay whose introduction into the system would prevent malicious users from identifying system configurations and active jobs.

Results presented in Table 4 are particularly interesting, demonstrating that Delaunay appears most vulnerable to data dependent tuning, Max Box mesh is largely insensitive to such tuning, and Voronoi mesh benefits (for this data set) from the tuning.

There are many avenues for extending this modeling methodology. One extension is to add categorical variables to the models. Presently the rejection rate of distribution predictions can only be reduced with large volumes of performance data, however the judicious choice (via experimental design, e.g.) of new data points may be able to effectively reduce the amount of training data required. Finally, more case studies need to be done to test the robustness of the present modeling techniques to changes in domain and performance measure.

9 CONCLUSION

The methodology presented is capable of providing new insights, extending existing analyses, and improving the management of computational performance variability. Delaunay, Max Box mesh, and Voronoi mesh interpolation are viable techniques for constructing approximations of performance cumulative distribution functions. A case study on I/O throughput demonstrated that the models are capable of effectively predicting CDFs for most unseen system configurations for any of the available I/O test types. The present methodology represents a notable increase in the ability to statistically model arbitrary system performance measures involving the interaction of many ordinal system parameters.

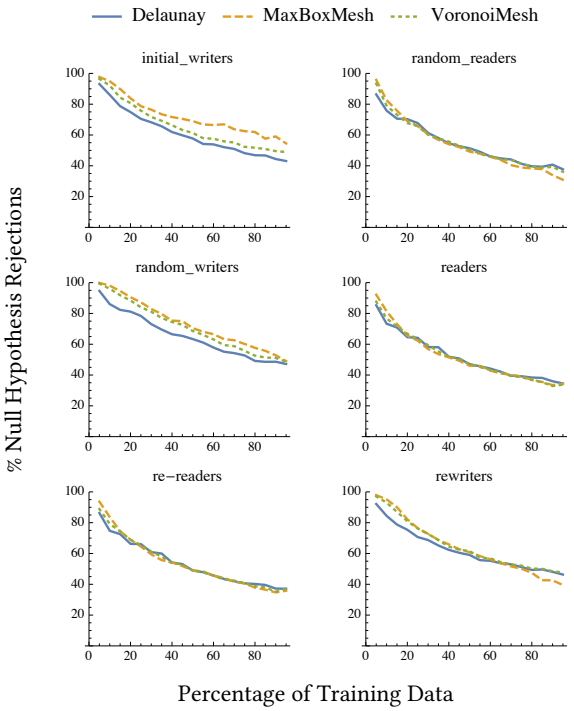

Figure 9: The percentage of null hypothesis rejections for predictions made by each algorithm on the KS test ($p = 0.001$) over different IOzone test types with increasing amounts of training data. Each percentage of null hypothesis rejections is an average over ten random splits of the data. The training percentages range from 5% to 95% in increments of 5%. The read test types tend to allow lower rejection rates than the write test types.

REFERENCES

[1] S. De Vito, E. Massera, M. Piga, L. Martinotto, and G. Di Francia, "On field calibration of an electronic nose for benzene estimation in an urban pollution monitoring scenario," *Sensors and Actuators B: Chemical*, vol. 129, no. 2, pp. 750–757, 2008.

[2] D. Lazos, A. B. Sproul, and M. Kay, "Optimisation of energy management in commercial buildings with weather forecasting inputs: A review," *Renewable and Sustainable Energy Reviews*, vol. 39, pp. 587–603, 2014.

[3] P. Cortez and A. M. G. Silva, "Using data mining to predict secondary school student performance," *Proceedings of 5th Annual Future Business Technology Conference, Porto*, 2008.

[4] T. Lux, R. Pittman, M. Shende, and A. Shende, "Applications of supervised learning techniques on undergraduate admissions data," in *Proceedings of the ACM International Conference on Computing Frontiers*. ACM, 2016, pp. 412–417.

[5] T. C. Lux, L. T. Watson, T. H. Chang, J. Bernard, B. Li, X. Yu, L. Xu, G. Back, A. R. Butt, K. W. Cameron *et al.*, "Nonparametric distribution models for predicting and managing computational performance variability," in *SoutheastCon 2018*. IEEE, 2018, pp. 1–7.

[6] A. Tsanas, M. A. Little, P. E. McSharry, and L. O. Ramig, "Accurate telemonitoring of parkinson's disease progression by noninvasive speech tests," *IEEE Transactions on Biomedical Engineering*, vol. 57, no. 4, pp. 884–893, 2010.

[7] P. Cortez and A. d. J. R. Morais, "A data mining approach to predict forest fires using meteorological data," *13th Portuguese Conference on Artificial Intelligence*, 2007.

[8] E. W. Cheney and W. A. Light, *A Course in Approximation Theory*. American Mathematical Soc., 2009, vol. 101.

[9] C. De Boor, C. De Boor, E.-U. Mathématicien, C. De Boor, and C. De Boor, *A practical guide to splines*. Springer-Verlag New York, 1978, vol. 27.

[10] G. unther Greiner and K. Hormann, "Interpolating and approximating scattered 3d-data with hierarchical tensor product b-splines," in *Proceedings of Chamonix*, 1996, p. 1.

[11] C. De Boor, K. Höllig, and S. Riemenschneider, *Box splines*. Springer Science & Business Media, 2013, vol. 98.

[12] T. C. Lux, L. T. Watson, T. H. Chang, J. Bernard, B. Li, X. Yu, L. Xu, G. Back, A. R. Butt, K. W. Cameron *et al.*, "Novel meshes for multivariate interpolation and approximation," in *Proceedings of the ACMSE 2018 Conference*. ACM, 2018, p. 13.

[13] J. E. Dennis Jr and R. B. Schnabel, *Numerical methods for unconstrained optimization and nonlinear equations*. Siam, 1996, vol. 16.

[14] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," *arXiv preprint arXiv:1511.07289*, 2015.

[15] J. H. Friedman, "Multivariate adaptive regression splines," *The Annals of Statistics*, pp. 1–67, 1991.

[16] J. H. Friedman and the Computational Statistics Laboratory of Stanford University, *Fast MARS.*, 1993.

[17] J. Rudy and M. Cherti. (2017) Py-earth: A python implementation of multivariate adaptive regression splines. Https://github.com/scikit-learn-contrib/py-earth [Online; accessed 2017-07-09]. [Online]. Available: https://github.com/scikit-learn-contrib/py-earth

[18] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.

[19] G. E. Dahl, T. N. Sainath, and G. E. Hinton, "Improving deep neural networks for lvcsr using rectified linear units and dropout," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2013*. IEEE, 2013, pp. 8609–8613.

[20] M. F. Møller, "A scaled conjugate gradient algorithm for fast supervised learning," *Neural networks*, vol. 6, no. 4, pp. 525–533, 1993.

[21] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[22] D. Basak, S. Pal, and D. C. Patranabis, "Support vector regression," *Neural Information Processing-Letters and Reviews*, vol. 11, no. 10, pp. 203–224, 2007.

[23] W. I. Thacker, J. Zhang, L. T. Watson, J. B. Birch, M. A. Iyer, and M. W. Berry, "Algorithm 905: Sheppack: Modified shepard algorithm for interpolation of scattered multivariate data," *ACM Transactions on Mathematical Software (TOMS)*, vol. 37, no. 3, p. 34, 2010.

[24] D.-T. Lee and B. J. Schachter, "Two algorithms for constructing a delaunay triangulation," *International Journal of Computer & Information Sciences*, vol. 9, no. 3, pp. 219–242, 1980.

[25] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967.

[26] G. L. Dirichlet, "Über die reduction der positiven quadratischen formen mit drei unbestimmten ganzen zahlen." *Journal für die Reine und Angewandte Mathematik*, vol. 40, pp. 209–227, 1850.

[27] H. W. Lilliefors, "On the kolmogorov-smirnov test for normality with mean and variance unknown," *Journal of the American Statistical Association*, vol. 62, no. 318, pp. 399–402, 1967.

[28] W. D. Norcott. (2017) Iozone filesystem benchmark. Http://www.iozone.org [Online; accessed 2017-11-12]. [Online]. Available: http://www.iozone.org

[29] P. Patel, A. H. Ranabahu, and A. P. Sheth, "Service level agreement in cloud computing," *Wright State University CORE Scholar Repository*, 2009.

[30] M. Ali, S. U. Khan, and A. V. Vasilakos, "Security in cloud computing: Opportunities and challenges," *Information Sciences*, vol. 305, pp. 357–383, 2015.