Response to Reviewers

April 1, 2020

Dear reviewers,

We have carefully considered all of your comments and feedback, and are grateful for your time and effort. The following revisions have been made to account for the feedback given:

1. A description and set of references to sparse grid methodologies has been placed in the introduction. (Page 2)

2. The descriptions of approximation algorithms have been modified for greater clarity, and three new visuals depicting relevant Delaunay, Voronoi, and Modified Shepard properties have been added. (Pages 4–8)

3. Slight textual modifications were made to Lemma 3 for clarity, and a new visual was constructed to illustrate the logic of the accompanying proof. (Pages 11–12)

4. A new visual was added to the analytic test subsection to demonstrate the link between the theory and practice, accompanying text was added as well. (Pages 15–16)

5. Minor rephrasing and additional references were added throughout the text to address specific concerns and/or misunderstandings. (Pages 3, 9–10, 13–14, 17, and 28)

For the reviewers' convenience, all newly introduced modifications in this revision are marked with red text and have a red bar in the margin accompanying them. In addition, all colored links in this response point directly to the associated positions in the revised paper. Responses to individual requests / comments follow.

* *It is not clear why only Delaunay and the MLP are compared in the analytic test.*

The reason for only studying these two methods is twofold. First and foremost, only Delaunay and MLP are strictly piecewise linear approximations, which means the theorem directly applies to these techniques. Secondly, there is a lot of potential for clutter / overabundance of information in the visuals that follow and little for a reader to gain from seeing all techniques compared on this test. The text was modified to clarify this point at the end of the first paragraph of Section 5.1.

* *Why does the error of Delaunay not decrease with more points in Figure 7?*

  This question has multiple answers, but in short there are not enough data points to see a meaningful decrease in error. It is our hope that item 4 answers this question thoroughly in the paper.

* *The connection between the analytic test and the theorem is unclear. A simulation in which the error is studied with respect to $d(z, x_0)$ for the Delaunay interpolant should be included.*

  Addressed by item 4.

* *It is not clear why the performed simulation is in Section 5.1, and not in a new section.*

  As more clearly shown by item 4, the analytic test specifically demonstrates the validity of the theorem. This is why the analytic test is considered part of Section 5.

* *Report the mean computational time in the main text and leave only the details at the appendix.*

  The mean computation time is included in the main text in Table 2.

* *The statement, "In summary, the regime of interpolation is much greater . . . " is way too ambitious.*

  The phrasing has been reworked (page 16) to emphasize the underlying fact: the transition point at which regression algorithms become better than interpolation algorithms is (at least partially) a function of the signal-to-noise ratio (SNR) and the data sparsity. When data is more sparse, the range of SNRs for which interpolants are useful grows.

* *Add a mention of piecewise linear and Lagrangian interpolation to the list of "well understood" interpolation techniques.*

  Done on page 2.

* *A high dimension interpolation survey is not complete without mention of sparse grids.*

  Addressed by item 1.

* *"... generic uniform bounds are largely unobtainable for regression techniques on arbitrary approximation problems." Not true, bounds for regression techniques can be found using projection onto orthogonal polynomials.*

  We have reworded the statement and referenced the sparse grid error bounds in the same paragraph (page 2).

* *"... interactions are either preconceived or model pairwise interactions between dimensions at most." This statement is not true for sparse grids.*

  The wording has been clarified to say that this statement applies to problems with more than tens of dimensions (page 3).

* *What is the range of j for MARS? Mention that the choice of the MLP network architecture is crucial. Add a picture of a mesh that satisfies the Delaunay criterion and one that doesn't. Comment on the singularity at $W^{(k)}(x^{(k)})$ for modified Shepard and plot the shape of $W^{(k)}(x)$. Please remove the sparse grid reference from Box Splines. Add a plot showing the Voronoi tessellation and $v^{x^{(i)}}(y)$.*

  All of these have been addressed in each individual subsection and in part by item 2 above. Regarding the shape of W_k in Modified Shepard and LSHEP, the weight equation was rewritten to provoke a more clear understanding in accompaniment with the visual.

* *How would you measure the error if the range of approximation is the space of continuous functions?*

  This is answered in the last paragraph of Section 7, the Discussion.

* *Rewrite as "Instead, for a CDF ..." on page 9. Move the sentence in Figure 4 regarding the KS test to the main text. Add "empirical" before "CDF" where appropriate.*

  All addressed on page 9.

* *I think that the Appendix provided on the meaning of confidence intervals and null hypothesis is too short for someone without a background in statistics.*

  A reference to an introductory statistics textbook has been provided for readers that need a more thorough introduction to statistics (page 28).

* *I don't understand the expression "round-trip prediction methodology".*

  The phrasing has been changed (page 10). This is a reference the process of converting source data into predictions and then measuring the error.

* *Add a picture of $g(t)$, $g'(t)$, and the line w to Lemma 3. Give details on these equalities*

$$\frac{g'(0)\tilde{t}}{2} = \int_0^{\tilde{t}} \big(g'(0) - \gamma_g t\big) dt$$
$$\frac{-g'(0)^2}{2\gamma_g} = g'(0) - \frac{-g'(0)^2}{2\gamma_g} - \frac{\gamma_g}{2}$$

The first equation is the definition of an integral recalling $\tilde{t} = g'(0)/\gamma_g$. The second equation is misunderstood. In Lemma 3 it is correctly written as

$$\frac{-g'(0)^2}{2\gamma_g} < g'(0) - \frac{+g'(0)^2}{2\gamma_g} - \frac{\gamma_g}{2},$$

which follows directly from the statement that $g'(0) > \gamma_g/2$. Some text has been added to the lemma to attempt to make this more clear. The requested picture has been added as per item 3.

* *You should also try the analytic test with other space-filling points.*

The tests were additionally run with a Latin hypercube design, however nearly identical results were achieved (mentioned on page 14). The purpose of this test is not to provide a review of space filling / well-conditioned designs, but rather to pick a reasonable design and observe the connections between the observed error and the theorem. Fekete points were specifically requested as such a reasonable design by reviewers in the first revision of this paper.

* *Add in Figure 6 the reference slopes that show linear and quadratic decay of the error with respect to the number of points.*

Notice that the number of points is increasing in this visual. There is no obvious "reference" for linear and quadratic decay here. This is only possible when something approaches 0 in log-log plots. Instead, readers are given the relative decay in longest edge length to compare with decrease in error per item 4. In that figure, it is acknowledged that the approximation error is decreasing at a faster rate than the longest edge length.

* *Plot the Fekete points used in the case $d = 2$ (e.g. the case with 128 or 512 points).*

While potentially interesting for those unaware of Fekete points, this visual is easy to find elsewhere and would distract readers from the purpose of this subsection in the paper. The specific sample design methodology is not as important as decreasing the longest edge length of all simplices at roughly the same rate as the decrease in σ_d as seen in Figure 8.

* *On the test where $d = 20$: could it be that you just need more points before you can see convergence?*

  Yes certainly more points would help, but in order to achieve meaningful density in 20 dimensions at least millions of points would be needed. Consider that it takes roughly a million points to cover all corners of a box in $d = 20$, and that's with no points on the interior! The intent behind this example is to show that sparsity is very common in tens of dimensions and up.

* *Are the Fekete points in $d = 20$ just bad and you should use another design?*

  In general it is very difficult to construct well-spaced point sets in high dimension. This is a known problem. The Fekete points are a reasonable choice and are expected to perform similarly to other well-spaced point sets.

* *On the test with $d = 2$: could it be that the MLP stagnation is due to not having done enough epochs of training?*

  No. To check for this the amount of training was extended by five times. No meaningful improvement in performance was achieved.

* *What do you need the polynomials of order $n+1$ for (the Fekete points are obtained by the polynomials of order n)?*

  This is a misunderstanding. The paper reads "...polynomials of degree $n+1$." and the degree of a polynomial is one less than the order. So order n polynomials are used here.

* *Don't you perform both interpolation and regression on both the data sets with and without noise?*

  Yes, the phrasing in the parenthetical statements was unclear. It is been reworded to emphasize that having no noise poses an interpolation problem and having noise poses a regression problem (bottom of page 14 and top of page 15).

* *The statement, "...exponentially increasing the number of data points should result in a decreasing error." is imprecise: increasing even exponentially won't help if you place the points badly.*

  The original statement was made in the context of using Fekete points. In general yes, the points must be well spaced for convergence to be achieved. The wording has been adjusted to make this more clear (page 15).

* *Test on forest fire: you should mention that timings and lowest absolute prediction error can be checked in the Appendix (same for the other tests).*

  The wording in the paragraph that introduces the empirical tests has been modified to make this more clear (page 17).

* *Add at least for the forest fire test a scatter plot of predicted values versus correct values, at least for the two/three best methods.*

  This suggestion is appreciated. Such specific outcomes are not obviously related to the error bound theorem, which is the central component of this paper. For this reason, we would like to abstain from including information that might distract readers from the main purposes of the text.

Numerical Algorithms manuscript No.
(will be inserted by the editor)

Interpolation of Sparse High-Dimensional Data

Thomas C. H. Lux · Layne T. Watson · Tyler H. Chang · Yili Hong · Kirk Cameron

Received: date / Accepted: date

Abstract Increases in the quantity of available data have allowed 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. Regression is a popular approach to solving approximation problems with high dimension, however there are often advantages to interpolation. This paper presents a novel and insightful error bound for (piecewise) linear interpolation in arbitrary dimension and contrasts the performance of some interpolation techniques with popular regression techniques. Empirical results demonstrate the viability of interpolation for moderately high dimensional approximation problems, and encourage broader application of interpolants to multivariate approximation in science.

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 [19], energy consumption management [30], and student performance prediction [16,33] 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 forest fire risk assessment [15],

This work was supported by the National Science Foundation Grant CNS-1565314.

Thomas Lux
Virginia Polytechnic Institute & State University (VPI&SU)
Blacksburg, VA 24061
E-mail: tchlux@vt.edu

Parkinson's patient clinical evaluations [49], local rainfall and weather [51], credit card transactions [43], and high performance computing (HPC) file input/output (I/O) [34].

Regression and interpolation have a considerable theoretical base in one dimension [11]. Common techniques include Lagrangian interpolation, piecewise linear interpolation, and more generally spline interpolation via B-splines [5]. Tensor products of B-splines [50] or other basis functions have an unfortunate exponential scaling 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 [6], of which one of the approximation techniques in this work is composed [35].

The theoretical foundation of low dimensional interpolation allows the construction of strong error bounds that are lacking in high dimensional problems. A large body of literature exists studying the construction and application of sparse grids [8,12], which attempt to address the problem of dimensional scaling. The sparse grid point sets combined with specific approximation functions are used to construct total error bounds by considering projections onto orthogonal polynomials [36,39]. However, sparse grids have exponential size in dimension making them intractable for more than tens of dimensions and do not immediately apply to the problem of scattered data interpolation. This work extends some known results regarding the secant method [20] to construct a uniform error bound for interpolants over arbitrary scattered data in any dimension. These error bounds are useful, considering similar uniform bounds cannot be constructed 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. Interpolants do not have the need for *fitting* data, or minimizing error with respect to model 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 closed convex subset Y of a metrizable topological vector space with metric s, some function $f : \mathbb{R}^d \to Y$ and a set of points $X = \left\{x^{(1)}, \ldots, x^{(n)}\right\} \subset \mathbb{R}^d$, along with associated function values $f\left(x^{(i)}\right)$. The goal is to construct an approximation $\hat{f} : \mathbb{R}^d \to Y$ such that $\hat{f}\left(x^{(i)}\right) = f\left(x^{(i)}\right)$ for all $i = 1, \ldots, n$. 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 small approximation error at $y \notin X$. The two metric spaces that will be discussed in this work are the real numbers with metric $s(x, y) = |x - y|$, and the set of cumulative

distribution functions (CDFs) with the Kolmogorov-Smirnov (KS) statistic [32] as a metric.

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 choosing parameters P defining $\hat{f}(x;P)$ to minimize the error vector $\left(\left|\hat{f}\left(x^{(1)};P\right)-f\left(x^{(1)}\right)\right|,\ldots,\left|\hat{f}\left(x^{(n)};P\right)-f\left(x^{(n)}\right)\right|\right)$ in some norm. 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 problems beyond ten to twenty dimensions usually rely on linear models. That is not to say nonlinear models are absent, but nonlinearities are often either preconceived or model interactions between small subsets of 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, the main objective of this work is to study the empirical performance differences in both accuracy and computational expense between a set of interpolation techniques and a set of common regression techniques. The main contributions of this work are a sharp theoretical error bound for (piecewise) linear interpolation in arbitrary dimension and multiple empirical comparisons that illustrate potential advantages of interpolation.

The remainder of this paper is organized as follows. Sections 2 and 3 present the multivariate models. Section 4 gives the error measuring methodology that is used for collecting empirical results. Section 5 contains the theoretical error bounds for interpolation as well as an analytic test for demonstration. Section 6 presents the data sets for empirical approximation analysis and the approximation results for all models on these data sets. Section 7 analyzes and discusses the results, their implications, and their limitations. Finally, Section 8 concludes.

2 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 $f:\mathbb{R}^d\to Y$ are chosen to minimize some error measure related to data samples $f\left(x^{(i)}\right)$. 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 real valued functions ($Y\subset\mathbb{R}$). These techniques can be extended to real vector-valued ranges by repeating the construction for each component of the vector output.

Throughout the following, x denotes a d-tuple, x_i the ith component of x, and $x^{(i)}$ the ith d-tuple data point. Different symbols are used to represent the approximation function $\hat{f}$.

2.1 Multivariate Adaptive Regression Splines

This approximation was introduced in [22] and subsequently improved to its current version in [23], 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

$$B_{2j-1}(x) = B_l(x)\big(x_i - x_i^{(p)}\big)_+,$$
$$B_{2j}(x) = B_k(x)\big(x_i - x_i^{(p)}\big)_-,$$

where $j \leq m$ is the iteration number, m is the maximum number of underlying basis functions, $1 \leq p \leq n$, and $B_l(x)$, $B_k(x)$ are basis functions from the previous iteration,

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

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 $\mathcal{O}(ndm^3)$. This paper uses an implementation of Fast MARS [45] with $m = 200$ throughout all experiments.

2.2 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 [3]. 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

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

where K is the selected kernel function, $a_i \in \mathbb{R}^n$, $b \in \mathbb{R}$ are coefficients to be solved for simultaneously. The computational complexity of the SVR is $\mathcal{O}(n^2dm)$, with m being determined by the minimization convergence criterion. This paper uses the scikit-learn SVR [42] with a radial basis function kernel.

2.3 Multilayer Perceptron Regressor

The neural network is a well studied and widely used method for both regression and classification tasks [27,46]. When using the rectified linear unit (ReLU) activation function [18] and fitting the model with stochastic gradient descent (SGD) or BFGS minimization techniques [25,37,44], the model built by a multilayer perceptron uses layers $l : \mathbb{R}^i \to \mathbb{R}^j$ defined by

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

where $c_l \in \mathbb{R}^j$ and W_l is the $i \times j$ 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 $\mathcal{O}(ndm)$, where m is determined by the sizes of the layers of the network and the stopping criterion of the minimization used for finding weights. This paper uses an MLP built from Keras and Tensorflow to perform regression [1,13] with ten hidden layers each having thirty two nodes (a total of approximately ten thousand parameters), ReLU activation, and one hundred thousand steps of SGD for training. It should be noted that the choice of neural network architecture impacts approximation performance, but no architecture tuning is performed here.

3 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.

3.1 Delaunay

The Delaunay triangulation is a well-studied geometric technique for producing an interpolant [31]. 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 $x^{(0)}$, $x^{(1)}$, ..., $x^{(d)}$, and function values $f\left(x^{(i)}\right)$, $i = 0, \ldots, d$, $y \in S$ 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

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

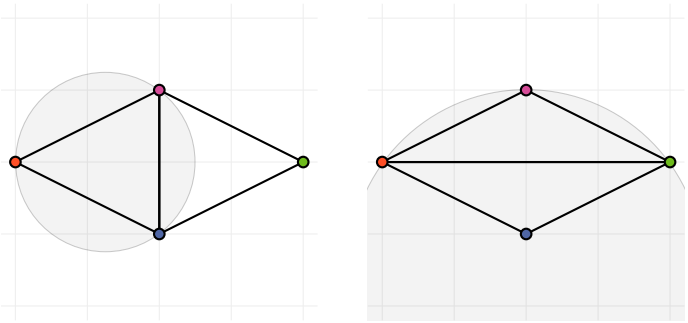

Fig. 1 On the left above is a depiction of a Delaunay triangulation over four points, notice that the circumball (shaded circle) for the left simplex does not contain the fourth point. On the right above, a non-Delaunay mesh is depicted. Notice that the circumball for the top simplex (shaded circle, clipped at bottom edge of the visual) contains the fourth point which violates the Delaunay condition for a simplex.

The computational complexity of Delaunay interpolation (for the implementation used [10]) is $\mathcal{O}(knd^3)$. Given pathological data the entire triangulation could be computed with $k = n^{\lceil d/2 \rceil}$, but the average case yields $k = d \log d$ and 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.

3.2 Modified Shepard

The modified Shepard method used here (ShepMod) generates a continuous approximation based on Euclidean distance and resembles a nearest neighbor interpolant [17]. This model is a type of *metric interpolation*, also called a Shepard method [26,47]. The interpolant has the form

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

where $W_k(x^{(k)}) = 1$, $W_k(x) = \left(\left(r_k - \left\|x - x^{(k)}\right\|_2\right)_+ / \left(r_k \left\|x - x^{(k)}\right\|_2\right)\right)^2$ for $x \neq x^{(k)}$, and $r_k \in \mathbb{R}$ is the smallest radius such that at least $d+1$ other points $x^{(j)}$, $j \neq k$, are inside the closed Euclidean ball of radius r_k about $x^{(k)}$. The computational complexity of ShepMod is $\mathcal{O}(n^2 d)$. This paper uses a Fortran 95 implementation of ShepMod derived from SHEPPACK [48].

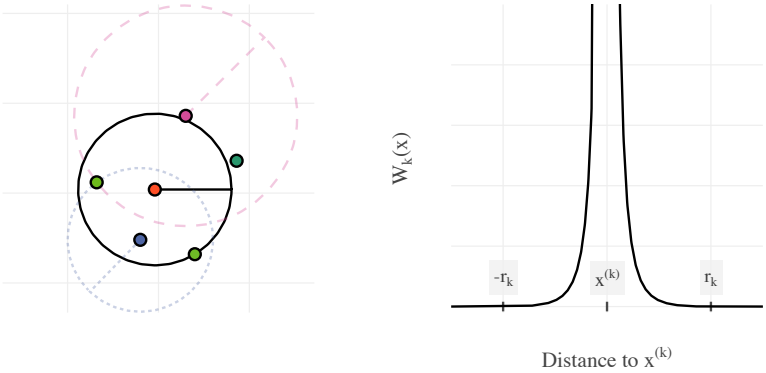

Fig. 2 On the left above is a depiction of the radius of influence for three chosen points of a collection in two dimensions using the modified Shepard criteria. On the right is the shape of the weight function W_k for all data points, where $W_k(x) = 0$ when $\|x - x^{(k)}\|_2 \geq r_k$. It can also be seen that in the limit as W_k tends towards infinity at x_k, the relative weight of x_k tends to one.

3.3 Linear Shepard

The linear Shepard method (LSHEP) is a blending function using local linear interpolants, a special case of the general Shepard algorithm [48]. 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 the same as for the modified Shepard method 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 $\mathcal{O}(n^2 d^3)$. This paper uses the Fortran 95 implementation of LSHEP in SHEPPACK [48].

3.4 Box Splines

The box spline model used here is an interpolation technique built from overlapping box splines [6]. The box splines serve 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 [35]. Given a set of box splines $\left\{b^{x^{(i)}}\right\}$ with the iterative box properties outlined in [35] and anchored at interior points $\left\{x^{(i)}\right\}$,

$$\hat{f}(y) = \frac{\sum_{i=1}^{n} b^{x^{(i)}}(y) f\left(x^{(i)}\right)}{\sum_{i=1}^{n} b^{x^{(i)}}(y)}.$$

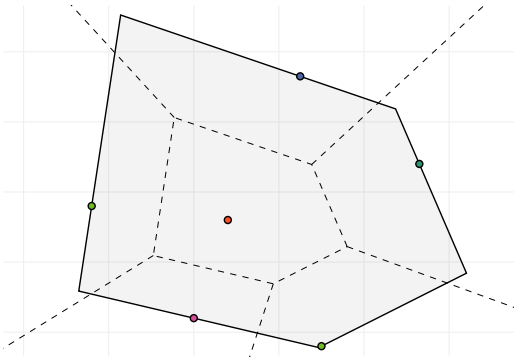

Fig. 3 Above is a depiction of the Voronoi cell boundaries (dashed lines) about a set of interpolation points (dots) in two dimensions. In this example, the Voronoi model basis function about the center most point has nonzero weight in the shaded region and transitions from a value of one at the point to zero at the boundary of the twice expanded Voronoi cell (solid line).

Note that the box splines always satisfy $b^{x^{(i)}}\left(x^{(j)}\right) = \delta_{ij}$ and $b^{x^{(i)}}(y) \geq 0$. The computational complexity of interpolation via the box spline model is $\mathcal{O}(n^2 d)$.

3.5 Voronoi

The nearest neighbor algorithm [17] is a well-studied technique for classification and approximation. Nearest neighbor inherently utilizes the convex region $C^{x^{(i)}}$ (Voronoi cell [21]) consisting of all points closer to $x^{(i)}$ than to any other point $x^{(j)}$. The Voronoi model 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

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

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

$$f(y) = \frac{\sum_{i=1}^{n} v^{x^{(i)}}(y) f\left(x^{(i)}\right)}{\sum_{i=1}^{n} v^{x^{(i)}}(y)},$$

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

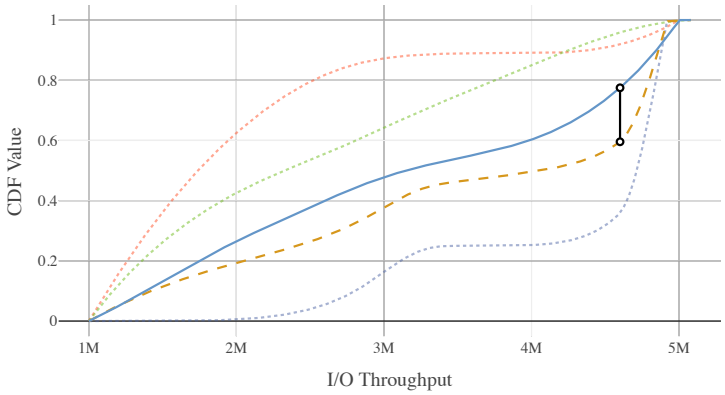

Fig. 4 In this HPC I/O example, the general methodology for predicting a CDF and evaluating error can be seen, where M means $\times 10^6$. The Delaunay method chose three source distributions (dotted lines) and assigned weights {.1, .3, .6} (top to bottom at middle). The weighted sum of the three known CDFs produces the predicted CDF (dashed line). The KS Statistic (vertical line) computed between the true CDF (solid line) and predicted CDF (dashed line) is 0.2 for this example. For this example the KS test null hypothesis is rejected at p-value 0.01, however it is not rejected at p-value 0.001.

4 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, and absolute error quartiles. When the range of an approximation is the space of cumulative distribution functions, the Kolmogorov-Smirnov statistic (max-norm difference between the functions) is used.

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 $\int_{-\infty}^{\infty} f(x)dx = 1$. Instead, for a CDF $F : \mathbb{R} \to \mathbb{R}$ the properties are $F(x) \in [0, 1]$ and $F(x)$ is absolutely continuous and nondecreasing. This work utilizes the fact that a convex combination of CDFs (or PDFs) results in a valid CDF (or PDF). 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 4. In this example the KS test null hypothesis is rejected at p-value 0.01, however it is not rejected at p-value 0.001.

The performance of approximation techniques that predict probability functions can be analyzed through a variety of summary statistics. This work uses the max absolute difference, also known as the Kolmogorov-Smirnov (KS) statistic [32] for its compatibility with the KS test.

The two-sample KS test is a useful nonparametric test for comparing two empirical CDFs while only assuming stationarity, finite mean, and finite variance. The null hypothesis (that two empirical 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 process of generating predicted distributions from known distributions and the subsequent calculation of error can be seen in Figure 4. A brief listing of relevant statistical terms used throughout this work is provided in the Appendix (Section A).

5 Theoretical Error Bound

This section presents the theoretical results bounding the error of (piecewise) linear interpolation. The error analysis relies on linear interpolation for three reasons: (1) second order results can be obtained utilizing a Lipschitz constant on the gradient of a function, rather than standard Lipschitz bounds; (2) the results directly apply to Delaunay interpolation; and (3) multiple other interpolants in this paper compute predictions as convex combinations of observed function values, which may allow for straightforward extensions of this error bound.

Lemma 1 Let $S \subset \mathbb{R}^d$ be open and convex, $f : S \to \mathbb{R}$, and let $\nabla f(x)$ be γ-Lipschitz continuous 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 $(y - x)$.

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

□

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 $k = \max_{1 \leq j \leq d} \|x_j - x_0\|_2$. Then

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

Proof. Consider $g(t) = f(x(t)) - \hat{f}(x(t))$ along the line segment $x(t) = (1 - t)x_0 + tx_j$, $0 \leq t \leq 1$, from x_0 to x_j. Observe that $g(0) = f(x_0) - \hat{f}(x_0) = 0$, $g(1) = f(x_j) - \hat{f}(x_j) = 0$, and $g'(t)$ is γ_g-Lipschitz continuous with $\gamma_g = \gamma\|x_j - x_0\|_2^2$. The following is visualized in Figure 5.

Suppose $g'(0) > \gamma_g/2$. Then $|g'(0) - g'(t)| \leq \gamma_g t \implies g'(0) - \gamma_g t \leq g'(t)$, and the line $w = g'(0) - \gamma_g t$ intersects the t-axis at $\tilde{t} = g'(0)/\gamma_g > 1/2$. $\tilde{t} < 1$ necessarily, since by Rolle's Theorem there exists $0 < z < 1$ such that $g'(z) = 0$ and so $g'(0) - \gamma_g z \leq g'(z) = 0$. Now by integrating

$$\frac{g'(0)^2}{2\gamma_g} = \frac{g'(0)\tilde{t}}{2} = \int_0^{\tilde{t}} (g'(0) - \gamma_g t)dt \leq \int_0^{\tilde{t}} g'(t)dt = g(\tilde{t}),$$

and acknowledging $g'(0) > \gamma_g/2$

$$\begin{aligned} \frac{-g'(0)^2}{2\gamma_g} < g'(0) - \frac{g'(0)^2}{2\gamma_g} - \gamma_g/2 = \frac{(1 - \tilde{t})(g'(0) - \gamma_g)}{2} &= \int_{\tilde{t}}^{1} (g'(0) - \gamma_g t)dt \\ &\leq \int_{\tilde{t}}^{1} g'(t)dt = -g(\tilde{t}), \end{aligned}$$

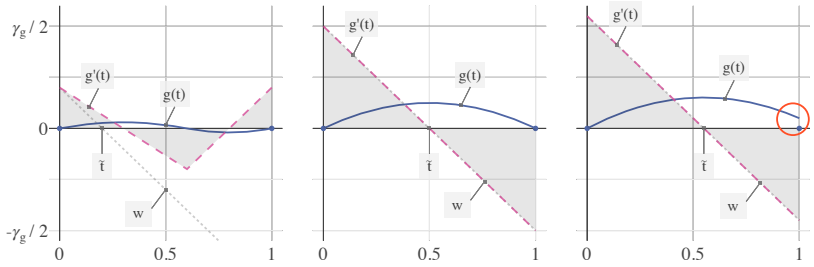

Fig. 5 Three different scenarios visualizing *Lemma 3*, where $g(t)$ is the difference between a piecewise linear interpolant and the approximated function along a normalized line segment between interpolation points, $g'(t)$ is γ_g-Lipschitz continuous, and w and $\tilde{t}$ are defined in the proof. Leftmost is a randomly chosen permissible shape of g and g'. The middle is the only possible shape of g and g' given $g'(0) = \gamma_g/2$, establishing the case of equality in the lemma. Rightmost is the resulting contradiction when $g'(0) > \gamma_g/2$, notice it is impossible to ensure $g'(t)$ is γ_g-Lipschitz continuous and satisfy $g(1) = 0$ (highlighted with red circle on the right).

a contradiction for the value of $g(\tilde{t})$. A similar contradiction arises for $g'(0) < -\gamma_g/2$. Therefore $|g'(0)| \leq \gamma_g/2$. In terms of f,

$$\left|\langle \nabla f(x_0) - c, x_j - x_0 \rangle\right| = |g'(0)| \leq \gamma \|x_j - x_0\|_2^2/2 \leq \gamma k^2/2,$$

which holds for all $1 \leq j \leq d$. Finally, using *Lemma 2*,

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

□

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

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

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

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

where the last inequality follows from *Lemma 1*. □

In summary, the approximation error of a linear (simplicial) interpolant tends quadratically towards zero when approaching observed data only when the diameter of the simplex is also reduced at a proportional rate. Only linear convergence to the true function is achieved in practice without the incorporation of additional observations, by moving closer to interpolation points. Notice that the approximation error is largely determined by the spacing of observed data. Predictions made by simplices whose vertices are not well-spaced (i.e., have large diameter, or are nearly contained in a hyperplane) have higher error.

That the theoretical error bound is sharp can be observed with the test function $q(x) = \|x\|_2^2$, $x \in \mathbb{R}^d$, and the simplex defined by vertices $X = \{0, e_1, \ldots, e_d\}$, where e_i is the i-th standard basis vector in $\mathbb{R}^d$, $e = (1, \ldots, 1) \in \mathbb{R}^d$, and 0 denotes the zero vector in any dimension. The constants relevant to the error bound are

$$\gamma = 2, \qquad \sigma_d = 1, \qquad k = 1, \qquad x_0 = 0, \qquad \hat{q}(x) = \langle e, x - 0 \rangle + q(0).$$

Noting that $q(0) = 0$, the approximation error at $z = -(1/2)e$ is

$$\left|q(z) - \hat{q}(z)\right| = \left|\|z\|_2^2 - \langle e, z\rangle\right| = \left|d/4 + d/2\right| = 3d/4,$$

while the error bound from the theorem gives

$$\begin{aligned}\left|q(z) - \hat{q}(z)\right| &\leq \frac{\gamma\|z - x_0\|_2^2}{2} + \frac{\sqrt{d}\,\gamma k}{2\sigma_d}\|z - x_0\|_2 \\ &= \|z\|_2^2 + \sqrt{d}\|z\|_2 \\ &= d/4 + d/2 = 3d/4.\end{aligned}$$

Acknowledging that the error bound is sharp, it may be of interest to observe the error of piecewise linear approximation techniques on an analytic test function.

5.1 Demonstration on an Analytic Test Function

The theoretical results constructed in Section 5 for (piecewise) linear interpolation are promising and apply directly to Delaunay interpolation, however they are difficult to interpret in context with approximation algorithms that do not have similar known uniform error bounds. For that reason, an analytic function is used to measure the error of a piecewise linear interpolant (Delaunay) and a piecewise linear regressor (the MLP) when provided an increasing amount of data with varying levels of random noise. Only Delaunay and the MLP are considered in this demonstration because all other mentioned techniques are not strictly piecewise linear approximations.

The test function chosen here for analysis resembles the *oscillatory* function used by [2]. However, a slight modification is made to remove the simple dot product structure (which is favorable for the MLP). Let $f(x) = \cos(\|x\|_2)$ for

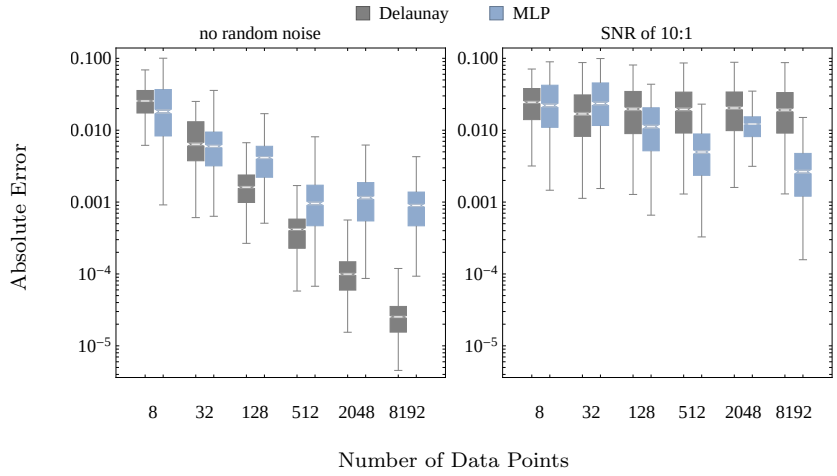

Fig. 6 Delaunay and MLP approximations are constructed from Fekete points over the unit cube evaluating the test function $f(x) = \cos(\|x\|_2)$ for $x \in \mathbb{R}^2$. The figure shows the first/third quartiles at the box bottom/top, the second quartile (median) at the white bar, median 95% confidence interval (cones, barely visible in figure), and whiskers at 3/2 of the adjacent interquartile ranges, for the absolute prediction error for each model at 1000 random evaluation points. The left plot observes a perfect interpolation problem with exact evaluations of f. The right plot observes a regression problem with uniform random noise giving values in $[.9f(x),\ 1.1f(x)]$ for each x. Both axes are log scaled.

$x \in \mathbb{R}^d$ where d is either 2 (Fig. 6) or 20 (Fig. 7). This function has a bounded change in gradient 2-norm and hence meets the necessary Lipschitz condition for the error bound. Data points and approximation points for this test will be within the unit cube $[0, 1]^d$.

The error of a linear interpolant constructed from well spaced points is dominated by the distance to the nearest data point. In order to uniformly decrease the distance to the nearest data point across the unit cube, exponentially more data points must be available for approximation. For this experiment $N = 2(4^i)$, $i = 1, \ldots, 6$, chosen points are kept well spaced by computing approximate Fekete points. The following experiment was also run with a Latin hypercube design [41], which produced almost identical results that are not reported here. Fekete points have a history in potential theory [29] and are most generally defined as those points that maximize the absolute value of a Vandermonde determinant. Here, the QR method outlined in [7] is used to identify approximate Fekete points from a Vandermonde matrix. A multivariate polynomial basis of size $2N$ is constructed containing all polynomials of degree $\leq n$ for the largest n such that $\binom{d+n}{n} \leq 2N$ and $2N - \binom{d+n}{n}$ arbitrarily selected polynomials of degree $n+1$. The Vandermonde matrix for these $2N$ polynomials is used to select N approximate Fekete points.

Finally an additional aspect is added to this test problem by incorporating random uniform noise into the evaluations of f. For each test two experiments are executed, one with exact function evaluations (an interpolation problem)

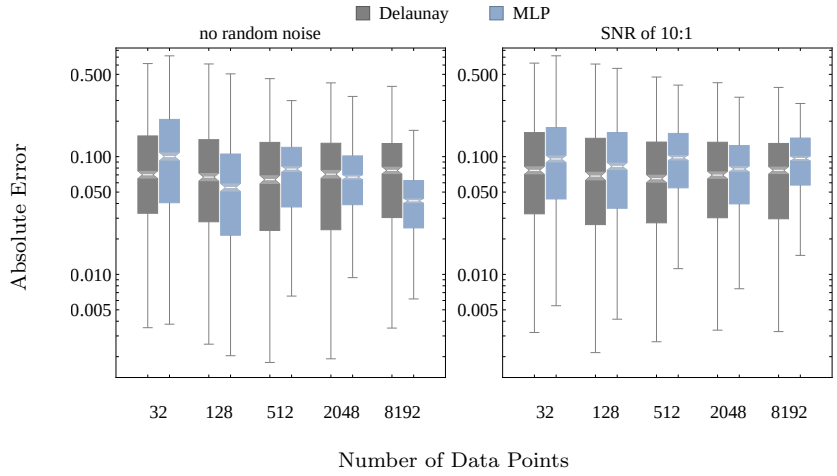

Fig. 7 Delaunay and MLP approximations are constructed from Fekete points over the unit cube evaluating the test function $f(x) = \cos(\|x\|_2)$ for $x \in \mathbb{R}^{20}$. The details are the same as for Fig. 6.

and one with a constant signal-to-noise ratio (SNR) of 10:1 (a regression problem).

The bound from the theorem suggests that by increasing the number of well-spaced data points, approximation error can be reliably decreased. The $d = 2$ test seen on the left half of Figure 6 shows a consistent decrease in error for Delaunay and also shows the eventual accuracy plateau obtained by a parametric regression form (the MLP, at roughly 500 points). On the right hand side of Figure 6 the random noise clearly prohibits Delaunay from converging to f, while the MLP is able to improve its approximation with more data points on average. The convergence result for a very low-dimensional problem like this is expected. However, intuition fails for higher dimensional problems.

Figure 7 shows the test function with $d = 20$ presents a significantly more challenging approximation problem than its counterpart in low dimension. The same increase in number of data points from 32 to 8192 causes no apparent improvement in approximation for either the noise-free or the noisy problems. Perhaps unexpectedly, the interpolation technique performs better than the regression technique on the noisy data (right) in Figure 7, and worse than the regression technique on the noise-free data (left). This result emphasizes the relevance of interpolation for problems in high dimension. It also reveals that the outcome of MLP regressions can get worse when adding more data points.

The results achieved for both $d = 2$ and $d = 20$ align with the theoretical error bound as can be seen in Figure 8. In low dimension, thousands of data points meaningfully reduce the sparsity of the approximation problem. However, in higher dimension it takes many more points to achieve a reduction in data sparsity while simultaneously being more difficult to produce well-

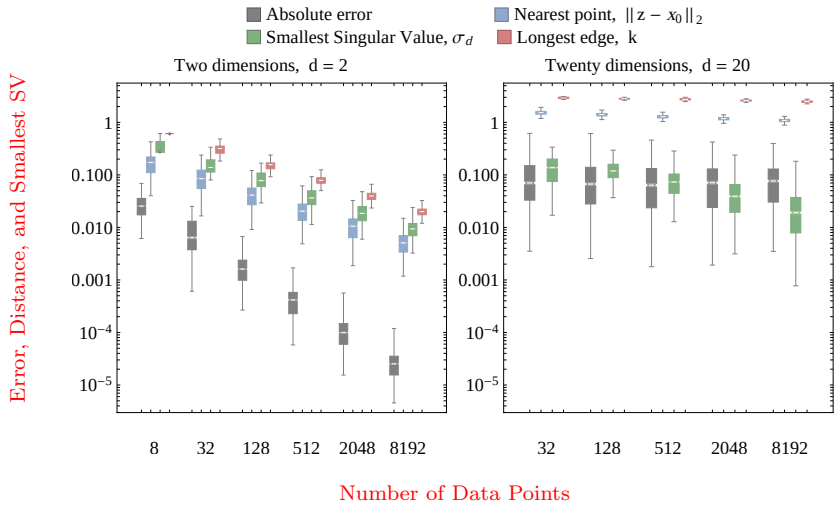

Fig. 8 The distribution of absolute error, distance to the nearest data point, smallest singular value (SV) and the longest edge of the simplex containing each approximation point in the tests from Fig. 6 and Fig. 7 for Delaunay. In two dimensions it can be seen that $\|z - x_0\|_2$, σ_d, and k all shrink at the same rate for well-spaced approximation points, resulting in a faster rate of decrease for approximation error. Notice that in higher dimension the data remains sparse even with thousands of data points, and the decay in data spacing is more prominant. The relatively small reduction in k along with the decrease in σ_d explain the minimal reduction in error seen by Delaunay in Fig. 7.

conditioned simplices. This evidences the inherent challenge of data sparsity in high dimension approximation problems. The analytic results presented are not caused by the chosen test function, but rather the exponential increase in complexity that accompanies increased dimension.

In summary, the regime of signal-to-noise ratios that result in competitively accurate interpolants is greater for moderate to high dimensional problems due to increased sparsity. Acknowledging the viability of interpolation for problems of moderate dimension, the next section will consider real-world problems of similar proportion (thousands of examples in tens of dimensions).

6 Data and Empirical Analysis

This section extends the comparison of interpolation and regression algorithms to a sample of real-world problems. Five different data sets of varying dimension and application are utilized to construct approximations and compare the accuracy of different techniques.

In the following five subsections the sources and targets of each test data set are described, as well as challenges and limitations related to approximating these data. The distribution of response values being modeled is presented followed by the distribution of approximation errors for each algorithm. The plots for all five data sets have the same format.

All five data sets are rescaled such that the domain of approximation is the unit hypercube. The range of the first four data sets is the real numbers, while the range of the fifth data set is the space of cumulative distribution functions. All approximation techniques are applied to the first four data sets, while only those interpolants whose approximations are convex combinations of observed data are applied to the final data set.

All approximations are constructed using k-fold cross validation as described in [28] with $k = 10$. This approach randomly partitions data into k (nearly) equal sized sets. Each algorithm is then evaluated by constructing an approximation over each unique union of $k-1$ elements of the partition, making predictions for points in the remaining element. As a result, each observed data point is used in the construction of $k-1$ different approximations and is approximated exactly once. The k-fold cross validation method is data-efficient and provides an unbiased estimate of the expected prediction error [28], however it should be noted that neither this method nor others can provide a universally unbiased estimator for the variance of prediction error [4].

In addition to the figures displaying approximation results for each data set, tables of accompanying numerical results including timings and exact prediction errors are located in the Appendix (Section A). All of the test data sets capture underlying functions that are almost certainly stochastic. As described in Section 1, regression techniques appear most appropriate for these problems. However, typically data grows exponentially more sparse with increasing dimension. Given that sparse data regressors tend towards interpolation and, as demonstrated in Section 5.1, interpolants produce similar (if not identical) results, it is presumed that interpolants are equally viable approximation techniques on these problems.

6.1 Forest Fire ($n = 504$, $d = 12$)

The forest fire data set [15] describes the area of Montesinho park burned over months of the year along with environmental conditions. The twelve dimensions being used to model burn area are the x and y spatial coordinates of burns in the park, month of year (mapped to x, y coordinates on a unit circle), the FFMC, DMC, DC, and ISI indices (see source for details), the temperature, 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.

As suggested by Figure 10, the SVR has the lowest absolute prediction errors for 80% of the data, with MLP and Delaunay being the nearest overall competitors. The effectiveness of SVR on this data suggests the underlying function can be defined by relatively few parameters, as well as the importance of capturing the low-burn-area data points.

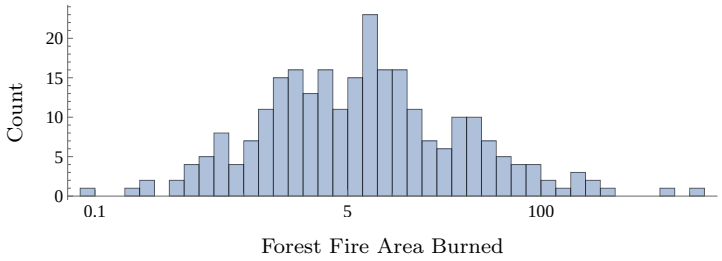

Fig. 9 Histogram of forest fire area burned under recorded weather conditions. The data is presented on a ln scale because most values are small with exponentially fewer fires on record that burn large areas.

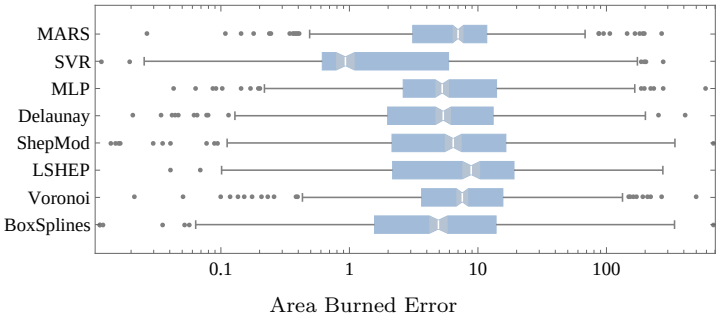

Fig. 10 All models are applied to approximate the amount of area that would be burned given environment conditions. 10-fold cross validation as described in the beginning of Section 6 is used to evaluated each algorithm. This results in exactly one prediction from each algorithm for each data point. These boxes depict the median (middle bar), median 95% confidence interval (cones), quartiles (box edges), fences at 3/2 interquartile range (whiskers), and outliers (dots) of absolute prediction error for each model. Similar to Figure 9, the errors are presented on a ln scale. The numerical data corresponding to this figure is provided in Table 3 in the Appendix.

6.2 Parkinson's Telemonitoring ($n = 5875$, $d = 19$)

The second data set for evaluation [49] 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 approximated 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 19 real numbers generated from biomedical voice measures of in-home sound recordings.

Figure 12 shows the ShepMod algorithm has the lowest minimum, first quartile, and median of absolute errors for this problem, while providing the best prediction 66% of the time. The MLP has the lowest third quartile and provides the best prediction for 32% of approximations. The dominance of ShepMod may be due in part to regular-interval total UPDRS scores provided by clinicians, favoring a nearest-neighbor strategy of prediction.

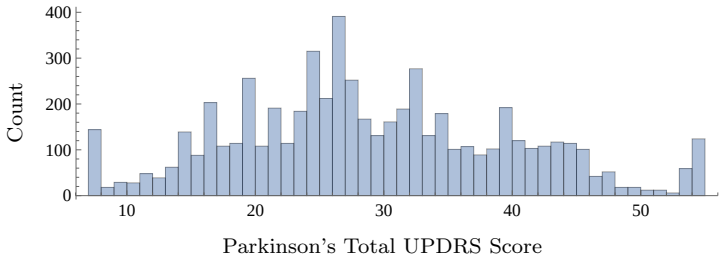

Fig. 11 Histogram of the Parkinson's patient total UPDRS clinical scores that will be approximated by each algorithm.

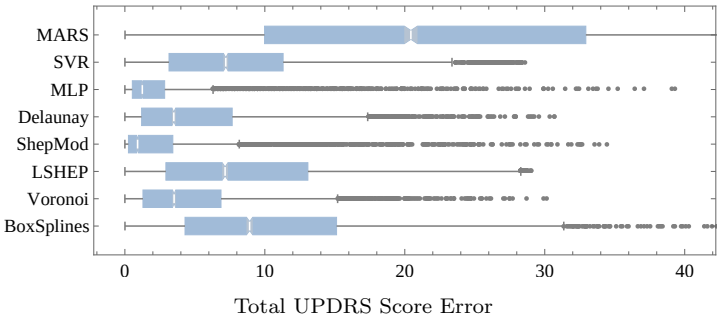

Fig. 12 All models are applied to approximate the total UPDRS score given audio features from patients' home life, using 10-fold cross validation. These boxes depict the median (middle bar), median 95% confidence interval (cones), quartiles (box edges), fences at 3/2 interquartile range (whiskers), and outliers (dots) of absolute prediction error for each model. The numerical data corresponding to this figure is provided in Table 5 in the Appendix.

6.3 Australian Daily Rainfall Volume ($n = 2609$, $d = 23$)

The third data set for the total daily rainfall in Sydney, Australia [51] provides a slightly higher dimensional challenge for the interpolants and regressors. This data is composed of many meteorological readings from the region in a day including: min and max temperatures, sunshine, wind speed directions (converted to coordinates on a circle), wind speeds, and humidities throughout the day, day of the year (converted to coordinates on a circle), and the model must predict the amount of rainfall tomorrow.

While Figure 13 makes MARS look far better than other techniques, it only provides the best prediction for 11% of points. The MLP has the lowest absolute error for 56% of points and LSHEP is best for 28%. MARS likely achieves such a low first quartile due to the high occurrence of the value zero in the data.

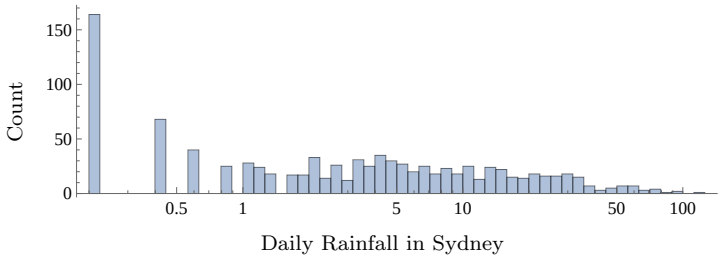

Fig. 13 Histogram of daily rainfall in Sydney, Australia, presented on a ln scale because the frequency of larger amounts of rainfall is significantly less. There is a peak in occurrence of the value 0, which has a notable effect on the resulting model performance.

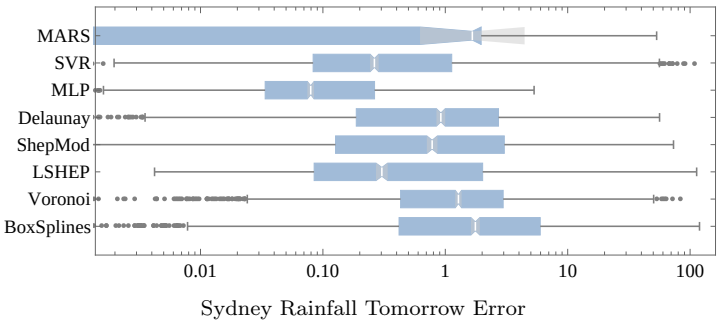

Fig. 14 All models are applied to approximate the amount of rainfall expected on the next calendar day given various sources of local meteorological data, using 10-fold cross validation. These boxes depict the median (middle bar), median 95% confidence interval (cones), quartiles (box edges), fences at 3/2 interquartile range (whiskers), and outliers (dots) of absolute prediction error for each model. The errors are presented on a ln scale, mimicking the presentation in Figure 13. The numerical data corresponding to this figure is provided in Table 7 in the Appendix.

6.4 Credit Card Transaction Amount ($n = 5562$, $d = 28$)

The fourth test data set, and the final with a real-valued range, is a collection of credit card transactions [43]. The provided data carries no direct real-world meaning, being the output of a principle component analysis on the original hidden source data. This obfuscation is done to protect the information of the credit card users. This data has the largest dimension of all considered, at 28. A model for this data predicts the transaction amount given the vector of principle component information.

As can be seen in Figure 16, the MLP outperforms all other algorithms at the first, second, third, and fourth quartiles. The MLP produces the lowest absolute error prediction for 80% of transactions, Delaunay bests the remaining 20%. It is likely that with less data, Delaunay would be the best performer.

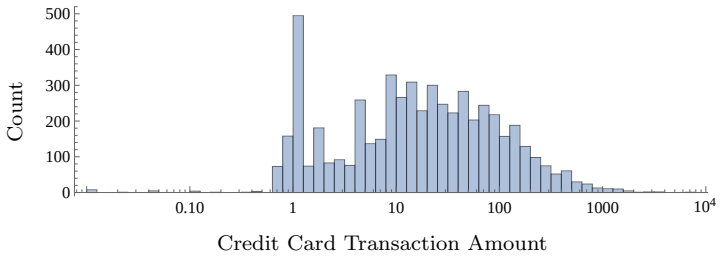

Fig. 15 Histogram of credit card transaction amounts, presented on a ln scale. The data contains a notable frequency peak around $1 transactions. Fewer large purchases are made, but some large purchases are in excess of five orders of magnitude greater than the smallest purchases.

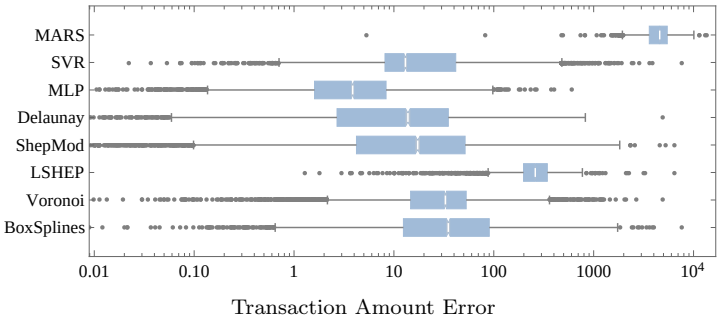

Fig. 16 All models are applied to approximate the expected transaction amount given transformed (and obfuscated) vendor and customer-descriptive features, using 10-fold cross validation. These boxes depict the median (middle bar), median 95% confidence interval (cones), quartiles (box edges), fences at 3/2 interquartile range (whiskers), and outliers (dots) of absolute prediction error for each model. The absolute errors in transaction amount predictions are presented on a ln scale, just as in Figure 15. The numerical data corresponding to this figure is provided in Table 9 in the Appendix.

6.5 High Performance Computing I/O ($n = 3016$, $d = 4$)

The final of five data sets is derived from [9], which provides four-dimensional distribution data by executing the IOzone benchmark [40] on a computer system and varying the system's file size, record size, thread count, and CPU frequency. At each configuration, IOzone samples the I/O file-read throughput (in bytes per second) 150 times. Empirical distribution function points are computed from each set of 150 executions, which are interpolated with a piecewise cubic Hermite interpolating polynomial [24] to approximate the CDF. All interpolation algorithms with the exception of LSHEP are used to approximate these CDFs from system configurations.

Delaunay achieves the lowest KS statistic (max norm difference) for 62% of approximations, while Voronoi is best for the remaining 38%. Figure 18 shows that while Delaunay may have more best predictions, the behavior of Voronoi may be preferable.

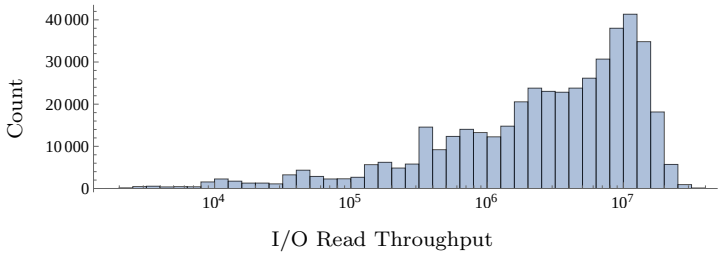

Fig. 17 Histogram of the raw throughput values recorded during all IOzone tests across all system configurations. The distribution is skewed right, with few tests having significantly higher throughput than most others. The data is presented on a ln scale.

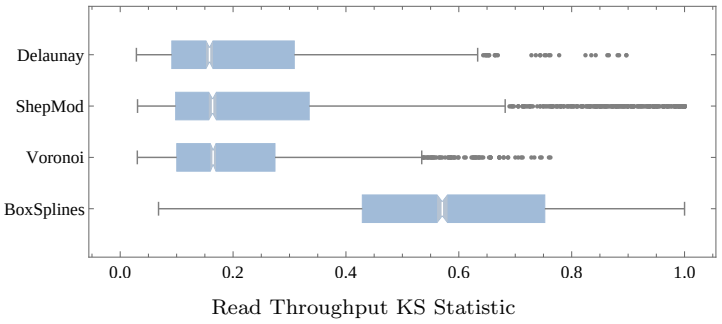

Fig. 18 The models directly capable of predicting distributions are applied to predicting the expected CDF of I/O throughput at a previously unseen system configuration, using 10-fold cross validation. The KS statistic (max norm) between the observed distribution at each system configuration and the predicted distribution is recorded and presented above. Note that the above figure is *not* log-scaled like Figure 17. The numerical data corresponding to this figure is provided in Table 11 in the Appendix.

	$p = .05$	$p = .01$	$p = .001$	$p = 10^{-6}$
Delaunay	**50.3**%	*43.5*%	*36.2*%	*24.7*%
ShepMod	*51.4*%	44.8%	38.1%	27.7%
Voronoi	52.6%	**43.4**%	**34.4**%	**19.1**%
BoxSplines	99.4%	98.5%	96.6%	89.3%

Table 1 Numerical counterpart of the histogram data presented in Figure 19. The columns display the percent of null hypothesis rejections by the KS-test when provided different selections of p-values for each algorithm. The algorithm with the lowest rejection rate at each p is boldface, while the second lowest is italicized.

Figure 19 expands on the KS statistic results presented in Figure 18. Agglomerate errors for each algorithm resemble a Gamma distribution. The percentages of significant prediction errors with varying p-values are on display in Table 1. When considering the $p = 0.001$ results for each technique, a little over one third of the predicted CDFs are significantly different from the measured (and presumed) correct CDFs. However, it should be noted that

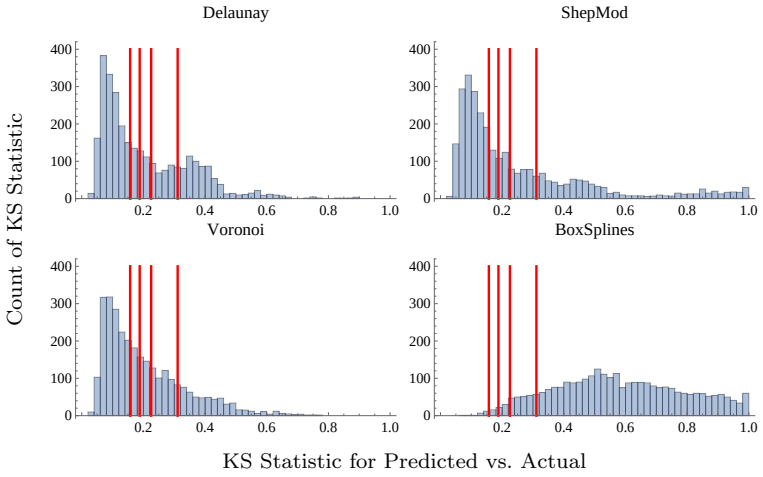

Fig. 19 Histograms of the prediction error for each interpolant that produces predictions as convex combinations of observed data, using 10-fold cross validation. The histograms show the KS statistics for the predicted throughput distribution versus the actual throughput distribution. The four vertical lines represent cutoff KS statistics given 150 samples for commonly used p-values 0.05, 0.01, 0.001, 10^{-6}, respectively. All predictions to the right of a vertical line represent CDF predictions that are significantly different (by respective p-value) from the actual distribution according to the KS test. The numerical counterpart to this figure is presented in Table 1.

with 150 samples, the error of an empirical distribution function (EDF) can reasonably be upwards of .1, which serves as a rough estimate for the lower limit of achievable error. Globally, only a third of Voronoi predictions fail to capture *all* of the characteristics of the CDFs at new system configurations.

7 Discussion

Table 2 summarizes results across the four test data sets with real-valued ranges. The interpolants discussed in this paper produce the *best* approximations roughly one third of the time, and produce competitive approximations for almost all data sets. These test problems are almost certainly *stochastic* in nature, but the high dimension leads to greater data sparsity and model construction cost, making interpolation more competitive.

The major advantages to interpolation lie in the near absence of *fit* time. Delaunay, LSHEP, and ShepMod all require pairwise distance calculations, for numerical robustness (Delaunay) and to determine the radii of influence for data points (LSHEP and ShepMod). At least hundreds, and sometimes hundreds of thousands of predictions can be made by the interpolants before the most widely used regressor (MLP) finishes fitting these relatively small data sets. However, the computational complexities of all interpolants presented are greater than linear in either dimension or number of points, whereas the

Algorithm	Avg. % Best	Avg. Fit or Prep. Time (s)	Avg. App. Time (s)
MARS	4.5	20.0s	0.001s
SVR	*19.5*	**0.5**s	**0.0001**s
MLP	**43.1**	200.0s	0.001s
Delaunay	5.2	1.0s	1.0s
ShepMod	18.0	*0.7*s	**0.0001**s
LSHEP	8.4	2.0s	**0.0001**s
Voronoi	0.5	1.0s	0.04s
BoxSplines	3.5	0.8s	*0.0005*s

Table 2 This average of Appendix Tables 4, 6, 8, and 10 provides a gross summary of overall results. The columns display (weighted equally by data set, *not* points) the average frequency with which any algorithm provided the lowest absolute error approximation, the average time to fit/prepare, and the average time required to approximate one point. The times have been rounded to one significant digit, as reasonably large fluctuations may be observed due to implementation hardware. Interpolants provide the lowest error approximation for nearly one third of all data, while regressors occupy the other two thirds. This result is obtained without any customized tuning or preprocessing to maximize the performance of any given algorithm. In practice, tuning and preprocessing may have large effects on approximation performance.

regressors' nonlinear complexity in dimension generally comes from the model fitting optimization.

The new theoretical results presented in Section 5 directly apply to Delaunay interpolation, however the performance of Delaunay does not appear significantly better than other algorithms on these data sets. This observation may be due to the stochastic nature of the data, but it also speaks to the power of the approximations generated by the different interpolation methods. The strong performance of other interpolants suggests that theoretical results similar to those presented in this work can be achieved for the other interpolants under reasonable assumptions.

Finally, most of the interpolants presented in this work benefit from the ability to model *any* function over real d-tuples with a range that is closed under convex combinations. In general, error can be quantified by any measure (particularly of interest may be L^2, L^∞, etc.). The results of the distribution prediction case study indicate that interpolants can effectively predict CDFs. The error analysis for that work relies on the KS statistic, which 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 as a metric because of the important surrounding statistical theory. A nonnegligible volume of predictions provide impressively low levels of average absolute error in that final case study.

8 Conclusion

The major contributions of this work are: 1) new uniform theoretical error bounds for piecewise linear interpolation in arbitrary dimension (Section 5);

2) an empirical evaluation across real-world problems that demonstrates interpolants produce competitively accurate models of multivariate phenomenon when compared with common regressors for sparse, moderately high dimensional problems (Section 6); and 3) a demonstration that some interpolants generalize to interpolation in function spaces (Section 4), preserving monotonicity (with CDFs, e.g.), neither of which common regressors can do.

The various interpolants discussed in this paper have been demonstrated to effectively approximate multivariate phenomena up to 30 dimensions. The underlying constructions are theoretically straightforward, interpretable, and yield reasonably accurate predictions. Most of the interpolants' computational complexities make them particularly suitable for applications in even higher dimension. The major benefits of interpolation are seen when only a small number of approximations (≤ 1000) are made from data and when there are relatively few data points for the dimension (for empirical results presented, $\log_d n \leq 5$). These findings encourage broader application of interpolants to multivariate approximation in science.

References

1. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X.: TensorFlow: Large-scale machine learning on heterogeneous systems (2015). URL https://www.tensorflow.org/. Software available from tensorflow.org
2. Barthelmann, V., Novak, E., Ritter, K.: High dimensional polynomial interpolation on sparse grids. Advances in Computational Mathematics **12**(4), 273–288 (2000)
3. Basak, D., Pal, S., Patranabis, D.C.: Support vector regression. Neural Information Processing-Letters and Reviews **11**(10), 203–224 (2007)
4. Bengio, Y., Grandvalet, Y.: No unbiased estimator of the variance of k-fold cross-validation. Journal of machine learning research **5**(Sep), 1089–1105 (2004)
5. de Boor, C.: A Practical Guide to Splines, vol. 27. Springer-Verlag New York (1978)
6. de Boor, C., Höllig, K., Riemenschneider, S.: Box Splines, vol. 98. Springer Science & Business Media (2013)
7. Bos, L., De Marchi, S., Sommariva, A., Vianello, M.: Computing multivariate fekete and leja points by numerical linear algebra. SIAM Journal on Numerical Analysis **48**(5), 1984–1999 (2010)
8. Bungartz, H.J., Griebel, M.: Sparse grids. Acta numerica **13**, 147–269 (2004)
9. Cameron, K.W., Anwar, A., Cheng, Y., Xu, L., Li, B., Ananth, U., Bernard, J., Jearls, C., Lux, T.C.H., Hong, Y., Watson, L.T., Butt, A.R.: Moana: Modeling and analyzing i/o variability in parallel system experimental design. IEEE Transactions on Parallel and Distributed Systems (2019). DOI 10.1109/TPDS.2019.2892129
10. Chang, T.H., Watson, L.T., Lux, T.C.H., Li, B., Xu, L., Butt, A.R., Cameron, K.W., Hong, Y.: A polynomial time algorithm for multivariate interpolation in arbitrary dimension via the delaunay triangulation. In: Proceedings of the ACMSE 2018 Conference, ACMSE '18, pp. 12:1–12:8. ACM, New York, NY, USA (2018). DOI 10.1145/3190645.3190680. URL http://doi.acm.org/10.1145/3190645.3190680
11. Cheney, E.W., Light, W.A.: A Course in Approximation Theory, vol. 101. American Mathematical Soc. (2009)

12. Chkifa, A., Cohen, A., Schwab, C.: High-dimensional adaptive sparse polynomial interpolation and applications to parametric pdes. Foundations of Computational Mathematics **14**(4), 601–633 (2014)
13. Chollet, F., et al.: Keras. https://keras.io (2015)
14. Clevert, D.A., Unterthiner, T., Hochreiter, S.: Fast and accurate deep network learning by exponential linear units (elus). arXiv preprint arXiv:1511.07289 (2015)
15. Cortez, P., Morais, A.d.J.R.: A data mining approach to predict forest fires using meteorological data. 13th Portuguese Conference on Artificial Intelligence (2007)
16. Cortez, P., Silva, A.M.G.: Using data mining to predict secondary school student performance. Proceedings of 5th Annual Future Business Technology Conference, Porto (2008)
17. Cover, T., Hart, P.: Nearest neighbor pattern classification. IEEE Transactions on Information Theory **13**(1), 21–27 (1967)
18. Dahl, G.E., Sainath, T.N., Hinton, G.E.: Improving deep neural networks for lvcsr using rectified linear units and dropout. In: IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2013, pp. 8609–8613. IEEE (2013)
19. De Vito, S., Massera, E., Piga, M., Martinotto, L., Di Francia, G.: On field calibration of an electronic nose for benzene estimation in an urban pollution monitoring scenario. Sensors and Actuators B: Chemical **129**(2), 750–757 (2008)
20. Dennis Jr, J.E., Schnabel, R.B.: Numerical Methods for Unconstrained Optimization and Nonlinear Equations, vol. 16. Siam (1996)
21. Dirichlet, G.L.: Über die reduction der positiven quadratischen formen mit drei unbestimmten ganzen zahlen. Journal für die Reine und Angewandte Mathematik **40**, 209–227 (1850)
22. Friedman, J.H.: Multivariate adaptive regression splines. The Annals of Statistics pp. 1–67 (1991)
23. Friedman, J.H., the Computational Statistics Laboritory of Stanford University: Fast mars. (1993)
24. Fritsch, F.N., Carlson, R.E.: Monotone piecewise cubic interpolation. SIAM Journal on Numerical Analysis **17**(2), 238–246 (1980)
25. Goh, G.: Why momentum really works. Distill (2017). DOI 10.23915/distill.00006. URL http://distill.pub/2017/momentum
26. Gordon, W.J., Wixom, J.A.: Shepard's method of "metric interpolation" to bivariate and multivariate interpolation. Mathematics of Computation **32**(141), 253–264 (1978)
27. Hornik, K., Stinchcombe, M., White, H.: Multilayer feedforward networks are universal approximators. Neural networks **2**(5), 359–366 (1989)
28. Kohavi, R., et al.: A study of cross-validation and bootstrap for accuracy estimation and model selection. In: IJCAI, vol. 14, pp. 1137–1145. Montreal, Canada (1995)
29. Kövari, T., Pommerenke, C.: On the distribution of fekete points. Mathematika **15**(1), 70–75 (1968)
30. Lazos, D., Sproul, A.B., Kay, M.: Optimisation of energy management in commercial buildings with weather forecasting inputs: A review. Renewable and Sustainable Energy Reviews **39**, 587–603 (2014)
31. Lee, D.T., Schachter, B.J.: Two algorithms for constructing a delaunay triangulation. International Journal of Computer & Information Sciences **9**(3), 219–242 (1980)
32. Lilliefors, H.W.: On the kolmogorov-smirnov test for normality with mean and variance unknown. Journal of the American Statistical Association **62**(318), 399–402 (1967)
33. Lux, T.C.H., Pittman, R., Shende, M., Shende, A.: Applications of supervised learning techniques on undergraduate admissions data. In: Proceedings of the ACM International Conference on Computing Frontiers, pp. 412–417. ACM (2016)
34. Lux, T.C.H., Watson, L.T., Chang, T.H., Bernard, J., Li, B., Yu, X., Xu, L., Back, G., Butt, A.R., Cameron, K.W., et al.: Nonparametric distribution models for predicting and managing computational performance variability. In: SoutheastCon 2018, pp. 1–7. IEEE (2018)
35. Lux, T.C.H., Watson, L.T., Chang, T.H., Bernard, J., Li, B., Yu, X., Xu, L., Back, G., Butt, A.R., Cameron, K.W., et al.: Novel meshes for multivariate interpolation and approximation. In: Proceedings of the ACMSE 2018 Conference, p. 13. ACM (2018)

36. Migliorati, G., Nobile, F., von Schwerin, E., Tempone, R.: Approximation of quantities of interest in stochastic pdes by the random discrete l^2 projection on polynomial spaces. SIAM Journal on Scientific Computing **35**(3), A1440–A1460 (2013)
37. Møller, M.F.: A scaled conjugate gradient algorithm for fast supervised learning. Neural networks **6**(4), 525–533 (1993)
38. Navidi, W.C.: Statistics for Engineers and Scientists, 4 edn. McGraw-Hill Education (2015)
39. Nobile, F., Tamellini, L., Tempone, R.: Convergence of quasi-optimal sparse-grid approximation of hilbert-space-valued functions: application to random elliptic pdes. Numerische Mathematik **134**(2), 343–388 (2016)
40. Norcott, W.D.: Iozone filesystem benchmark (2017). URL http://www.iozone.org. [Online; accessed 2017-11-12]
41. Park, J.S.: Optimal latin-hypercube designs for computer experiments. Journal of statistical planning and inference **39**(1), 95–111 (1994)
42. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research **12**, 2825–2830 (2011)
43. Pozzolo, A.D., Caelen, O., Johnson, R.A., Bontempi, G.: Calibrating probability with undersampling for unbalanced classification. In: IEEE Symposium Series on Computational Intelligence, pp. 159–166. IEEE (2015). DOI 10.1109/SSCI.2015.33. URL https://www.kaggle.com/mlg-ulb/creditcardfraud. [Online; accessed 2019-01-25]
44. Robbins, H., Monro, S.: A stochastic approximation method. The Annals of Mathematical Statistics pp. 400–407 (1951)
45. Rudy, J., Cherti, M.: Py-earth: A python implementation of multivariate adaptive regression splines (2017). URL https://github.com/scikit-learn-contrib/py-earth. [Online; accessed 2017-07-09]
46. Rumelhart, D.E., Hinton, G.E., Williams, R.J., et al.: Learning representations by back-propagating errors. Cognitive modeling **5**(3), 1 (1988)
47. Shepard, D.: A two-dimensional interpolation function for irregularly-spaced data. In: Proceedings of the 1968 23rd ACM national conference, pp. 517–524. ACM (1968)
48. Thacker, W.I., Zhang, J., Watson, L.T., Birch, J.B., Iyer, M.A., Berry, M.W.: Algorithm 905: Sheppack: Modified shepard algorithm for interpolation of scattered multivariate data. ACM Transactions on Mathematical Software (TOMS) **37**(3), 34 (2010)
49. Tsanas, A., Little, M.A., McSharry, P.E., Ramig, L.O.: Accurate telemonitoring of parkinson's disease progression by noninvasive speech tests. IEEE Transactions on Biomedical Engineering **57**(4), 884–893 (2010)
50. Unther Greiner, G., Hormann, K.: Interpolating and approximating scattered 3d-data with hierarchical tensor product b-splines. In: Proceedings of Chamonix, p. 1 (1996)
51. Williams, G.J.: Weather dataset rattle package. In: Rattle: A Data Mining GUI for R, vol. 1, pp. 45–55. The R Journal (2009). URL https://www.kaggle.com/jsphyg/weather-dataset-rattle-package. [Online; accessed 2019-01-25]

A Appendix

Statistical Terminology. A random variable X is precisely defined by its *cumulative distribution function* (CDF) F_X and the derivative of the CDF, the *probability density function* (PDF) f_X. For any possible value x of X, the *percentile* of x is 100 $F_X(x)$, the percentage of values drawn from X that would be less than or equal to x as the number of samples tends towards infinity. The *quartiles* of X are its 25-th, 50-th (median), and 75-th percentiles. The absolute difference between the median and an adjacent quartile is an *interquartile range*. Given an independent and identically distributed sample from X and presuming that X has finite mean and variance, a *confidence interval* can be drawn about any percentile estimated from the sample. A *confidence interval* describes the probability that a value lies within an interval. The *null hypothesis* is a statement (derived from some test statistic) that the expected value of the observed statistic is equal to an assumed population statistic. The p value is the probability of observing a given statistic if the *null hypothesis* is true. For a more detailed introduction to statistics and related terminology, see the work of Navidi [38].

Raw Numerical Results. The tables that follow show the precise experimental results for all data sets presented in Section 6. The tests were all run serially on an otherwise idle machine with a CentOS 6.10 operating system and an Intel i7-3770 CPU operating at 3.4 GHz. The detailed performance results in the tables that follow are very much dependent on the problem and the algorithm implementation (e.g., some codes are TOMS software, some industry distributions, and others are from conference paper venues). Different typeface is used to show best performers, however not much significance should be attached to ranking algorithms based on small time (millisecond) differences. The results serve as a demonstration of conceptual validity.

Algorithm	Min	25^{th}	50^{th}	75^{th}	Max
MARS	0.00984	3.11	7.01	*11.7*	1090.0
SVR	0.0118	**0.615**	**0.931**	**5.89**	1090.0
MLP	0.0426	2.63	5.27	14.0	1090.0
Delaunay	**0.0**	1.98	5.37	13.1	*1080.0*
ShepMod	**0.0**	1.93	6.27	16.0	1090.0
LSHEP	0.04	2.17	8.87	19.1	**1070.0**
Voronoi	*0.00982*	3.65	7.56	15.6	1090.0
BoxSpline	**0.0**	*1.27*	*4.61*	12.6	1090.0

Table 3 This numerical data accompanies the visual provided in Figure 10. The columns of absolute error percentiles correspond to the minimum, first quartile, median, third quartile, and maximum absolute errors respectively. The minimum of each column is boldface, while the second lowest value is italicized. All values are rounded to three significant digits.

Algorithm	% Best
MARS	7.3
SVR	**78.0**
MLP	0.0
Delaunay	0.2
ShepMod	2.0
LSHEP	5.1
Voronoi	0.0
BoxSpline	*9.7*

Fit/Prep. Time (s)	App. Time (s)	Total App. Time (s)
29.1	0.00137	0.0686
0.00584	*0.0000620*	*0.00310*
32.8	0.000871	0.0436
0.0151	0.0234	1.18
0.00634	0.0000644	0.00322
0.0275	**0.0000463**	**0.00231**
0.0152	0.000396	0.0198
0.00724	0.0000978	0.00489

Table 4 The left above shows how often each algorithm had the lowest absolute error approximating forest fire data in Table 3. On the right columns are median fit time of 454 points, median time for one approximation, and median time approximating 50 points.

Algorithm	Min	25^{th}	50^{th}	75^{th}	Max
MARS	0.00948	9.98	20.4	32.9	863.0
SVR	0.00233	3.15	7.21	11.3	**28.6**
MLP	0.0000239	*0.533*	*1.25*	**2.84**	39.3
Delaunay	3.72×10^{-12}	1.2	3.5	7.67	30.7
ShepMod	**0.0**	**0.255**	**0.908**	*3.43*	34.5
LSHEP	0.00254	2.93	7.16	13.1	*29.0*
Voronoi	**0.0**	1.29	3.52	6.87	30.1
BoxSpline	0.006	4.3	8.91	15.1	45.3

Table 5 This numerical data accompanies the visual provided in Figure 12. The columns of absolute error percentiles correspond to the minimum, first quartile, median, third quartile, and maximum absolute errors respectively. The minimum of each column is boldface, while the second lowest value is italicized. All values are rounded to three significant digits.

Algorithm	% Best
MARS	0.0
SVR	0.1
MLP	*32.0*
Delaunay	0.0
ShepMod	**66.4**
LSHEP	0.0
Voronoi	1.6
BoxSpline	0.0

Fit/Prep. Time (s)	App. Time (s)	Total App. Time (s)
37.9	0.00253	1.48
0.859	*0.000181*	*0.106*
348.0	0.00111	0.653
2.47	1.22	720.0
1.13	0.000182	0.107
2.39	**0.000144**	**0.0845**
2.77	0.0274	16.1
1.26	0.000643	0.377

Table 6 The left above shows how often each algorithm had the lowest absolute error approximating Parkinson's data in Table 5. On the right columns are median fit time of 5288 points, median time for one approximation, and median time approximating 587 points.

Algorithm	Min	25^{th}	50^{th}	75^{th}	Max
MARS	6.45×10^{-15}	$\mathbf{2.70 \times 10^{-14}}$	1.66	1.96	*53.3*
SVR	0.0000915	0.0833	0.263	*1.13*	109.0
MLP	0.0000689	0.0337	**0.0795**	**0.264**	**5.31**
Delaunay	**0.0**	0.187	0.919	2.72	56.3
ShepMod	**0.0**	0.0957	0.685	2.9	73.2
LSHEP	**0.0**	*0.0153*	*0.106*	1.17	113.0
Voronoi	**0.0**	0.43	1.28	2.94	83.8
BoxSpline	**0.0**	0.342	1.59	5.53	119.0

Table 7 This numerical data accompanies the visual provided in Figure 14. The columns of absolute error percentiles correspond to the minimum, first quartile, median, third quartile, and maximum absolute errors respectively. The minimum value of each column is boldface, while the second lowest is italicized. All values are rounded to three significant digits.

Algorithm	% Best
MARS	10.7
SVR	0.0
MLP	**60.9**
Delaunay	0.1
ShepMod	3.5
LSHEP	*28.4*
Voronoi	0.2
BoxSpline	4.1

Fit/Prep. Time (s)	App. Time (s)	Total App. Time (s)
0.151	0.000117	0.0304
0.133	**0.0000947**	**0.0246**
169.0	0.00137	0.356
0.664	0.886	230.0
0.265	0.000128	0.0333
0.874	*0.0000975*	*0.0254*
0.675	0.0270	7.01
0.330	0.000406	0.106

Table 8 Left table shows how often each algorithm had the lowest absolute error approximating Sydney rainfall data in Table 7. On the right columns are median fit time of 2349 points, median time for one approximation, and median time approximating 260 points.

Algorithm	Min	25^{th}	50^{th}	75^{th}	Max
MARS	5.36	3610.0	4580.0	5450.0	13400.0
SVR	0.00706	8.14	*13.0*	41.6	7690.0
MLP	0.00151	**1.6**	**3.86**	**8.34**	**604.0**
Delaunay	**0.0**	*2.69*	13.8	*35.0*	*4840.0*
ShepMod	**0.0**	4.21	17.3	51.6	6510.0
LSHEP	1.27	199.0	260.0	343.0	6530.0
Voronoi	2.89×10^{-10}	14.7	32.8	52.9	4860.0
BoxSpline	**0.0**	12.4	35.0	90.1	7690.0

Table 9 This numerical data accompanies the visual provided in Figure 16. The columns of absolute error percentiles correspond to the minimum, first quartile, median, third quartile, and maximum absolute errors respectively. The minimum value of each column is boldface, while the second lowest is italicized. All values are rounded to three significant digits.

Algorithm	% Best
MARS	0.0
SVR	0.0
MLP	**79.5**
Delaunay	*20.5*
ShepMod	0.1
LSHEP	0.0
Voronoi	0.0
BoxSpline	0.0

Fit/Prep. Time (s)	App. Time (s)	Total App. Time (s)
22.0	0.00148	0.820
1.01	*0.000210*	*0.117*
290.0	0.000714	0.397
3.12	3.71	2070.0
1.45	0.000220	0.122
3.47	**0.000176**	**0.0981**
3.32	0.0950	52.8
1.66	0.000956	0.532

Table 10 The left above shows how often each algorithm had the lowest absolute error approximating credit card transaction data in Table 9. On the right columns are median fit time of 5006 points, median time for one approximation, and median time approximating 556 points.

Algorithm	Min	25^{th}	50^{th}	75^{th}	Max
Delaunay	**0.0287**	**0.0914**	**0.158**	*0.308*	*0.897*
ShepMod	0.0307	*0.0983*	*0.164*	0.335	1.0
Voronoi	*0.0303*	0.1	0.165	**0.274**	**0.762**
BoxSpline	0.0679	0.429	0.571	0.752	1.0

Table 11 This numerical data accompanies the visual provided in Figure 18. The columns of absolute error percentiles correspond to the minimum, first quartile, median, third quartile, and maximum KS statistics respectively between truth and guess for models predicting the distribution of I/O throughput that will be observed at previously unseen system configurations. The minimum value of each column is boldface, while the second lowest is italicized. All values are rounded to three significant digits.

Algorithm	% Best
Delaunay	**62.0**
ShepMod	0.0
Voronoi	*38.0*
BoxSpline	0.0

Fit/Prep. Time (s)	App. Time (s)	Total App. Time (s)
0.344	0.00984	2.71
0.0884	**0.000145**	**0.0436**
0.360	0.00253	0.762
0.0972	*0.000210*	*0.0633*

Table 12 The left above shows how often each algorithm had the lowest KS statistic on the I/O throughput distribution data in Table 11. On the right columns are median fit time of 2715 points, median time for one approximation, and median time approximating 301 points.