Contents lists available at ScienceDirect

Journal of Computer and System Sciences

www.elsevier.com/locate/jcss

Weighted L_∞ isotonic regression

Quentin F. Stout

Computer Science and Engineering, University of Michigan, Ann Arbor, MI 48109-2121, United States

ARTICLE INFO

Article history:
Received 24 March 2012
Received in revised form 16 July 2013
Accepted 23 June 2017
Available online 14 September 2017

Keywords:
Isotonic regression
Monotonic
Unimodal
L_∞
Tree
Linear order
Dag

ABSTRACT

Algorithms are given for determining weighted L_∞ isotonic regressions satisfying order constraints given by a directed acyclic graph with n vertices and m edges. An $\Theta(m \log n)$ algorithm is given, but it uses parametric search, so a practical approach is introduced, based on calculating prefix solutions. For linear and tree orderings it yields isotonic and unimodal regressions in $\Theta(n \log n)$ time. Practical algorithms are given for when the values are constrained to a specified set, and when the number of different weights, or different values, is $\ll n$. We also give a simple randomized algorithm taking $\Theta(m \log n)$ expected time. L_∞ isotonic regressions are not unique, so we examine properties of the regressions an algorithm produces. In this regard the prefix approach is superior to algorithms, such as parametric search and the randomized algorithm, which are based on feasibility tests.

1. Introduction

A directed acyclic graph (dag) $G = (V, E)$ with n vertices and m edges defines a partial order over the vertices, where $u \prec v$ if and only if there is a path from u to v. A function g on V is *isotonic* iff it is a weakly order-preserving mapping into the real numbers, i.e., iff for all $u, v \in V$, if $u \prec v$ then $g(u) \le g(v)$. By *weighted data* on G we mean a pair of real-valued functions (f, w) on V, where w, the *weights*, is non-negative, and f, the *values*, is arbitrary. Given weighted data (f, w) on G, an L_p *isotonic regression* is an isotonic function g on G that minimizes

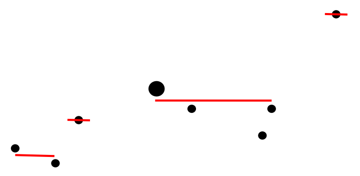
$$\begin{array}{ll} \left(\sum_{v \in V} w(v) \cdot |f(v) - g(v)|^p\right)^{1/p} & 1 \le p < \infty \\ \max_{v \in V} w(v) \cdot |f(v) - g(v)| & p = \infty \end{array}$$

among all isotonic functions. The L_p *regression error* is the value of this expression. For $1 < p < \infty$ the regression values are unique, but for L_1 and L_∞ they may not be. For example, on the vertices $\{1, 2, 3\}$, if $f = 3, 1, 2.5$ and $w = 2, 2, 1$: for $1 < p < \infty$ the L_p isotonic regression is $g(1) = g(2) = 2$, $g(3) = 2.5$, L_1 isotonic regressions are of the form $g(3) = 2.5$ and $g(1) = g(2) = x$ for $x \in [1, 2.5]$, and L_∞ isotonic regressions are of the form $g(1) = g(2) = 2$ and $g(3) \in [2, 4.5]$. In the example in Fig. 1 note that the regression values form level sets and that the regression is undefined in some regions.

Isotonic regression has long been applied to a wide range of problems in statistics [5,21,38,49] and classification [14,16,44], with numerous recent applications to machine learning and data mining [27,28,34,35,37,50]. Routines for it are available in several machine-learning systems [39,40]. It is also used for some optimization problems [6,29]. It is of increasing importance as researchers reduce their assumptions, replacing parametric requirements with weaker assumptions about an underlying direction. For example, to shrink tumors via radiation and chemotherapy, it might be assumed that at any

E-mail address: qstout@umich.edu.

http://dx.doi.org/10.1016/j.jcss.2017.09.001



Dots represent data, size represents their weight, lines are regression values

Fig. 1. An isotonic regression.

given radiation level the shrinkage increases with dose, and at any given dose the shrinkage increases with radiation. However, there may be no assumptions about a low dose and high radiation combination vs. a high dose and low radiation. Independent variables need not have a metric, merely an ordering, such as S $<$ M $<$ L $<$ XL sizes.

Isotonic functions are also known as monotonic, monotonic increasing, or order-preserving (as a function from G to $\Re$), and the L_∞ metric is also known as the supremum norm, Chebyshev distance, uniform metric, minimax optimization, or bottleneck criterion.

We develop algorithms for finding L_∞ isotonic regressions for weighted data. In Section 3 we give an algorithm for general dags taking $\Theta(m \log n)$ time. While conceptually simple, based on using parametric search as a black box procedure, it is of only theoretical interest because parametric search is very complex and infeasible. Therefore we give practical alternatives for several important special cases. E.g., when the number of different weights or different values is $\ll n$ a practical search can be used (Section 4.1).

L_∞ isotonic regressions are not unique, raising questions as to whether some are better than others. Previous algorithms have been compared only in terms of their running time, not in their mathematical properties when viewed as a mapping from data to isotonic functions. In Section 2 we examine properties such as minimizing the number of vertices with large regression errors, looking beyond just minimizing the worst error. By these measures the parametric search approach produces the worst regressions, lacking several properties that are shared by all L_p regressions for $1 < p < \infty$. In contrast, Strict, introduced in Section 2.1.3, has many of these properties. This topic is pursued further in [43].

Section 2.1.1 introduces Prefix, which is used in Section 6 to find isotonic regression for linear and tree orders and river isotonic regression on trees, where the regression value of a node is the same as the largest of its children. Unimodal regression on linear and tree orders is also examined, where the objective is to determine an optimal root and an isotonic regression towards the root. This arises in optimization problems such as competing failure modes [17,23,24] and multi-arm bandits [25,51].

Section 4.2 considers isotonic regression where the regression values are restricted to a specific set, such as the integers, using algorithms that are quite different from those in the other sections. An $\Theta(n \log n)$ algorithm is given for finding an integer-valued isotonic regression for a linear order, improving upon the $\Theta(n^2)$ algorithm of Liu and Ubhaya [33]. Section 7 contains concluding remarks.

Finally, Appendix A gives a very practical algorithm for arbitrary dags, replacing parametric search with a simple randomized procedure. It takes $\Theta(m \log n)$ expected time, not the worst-case time considered elsewhere in the paper, achieving this with high probability.

2. Background

Throughout we assume that G is a single connected component, and hence $m \geq n-1$. If it has more than one component then isotonic regressions can be found for each component separately.

The complexity of determining an isotonic regression depends on the regression metric and the partially ordered set. For example, for a linear order it is well-known that a simple left-right scanning approach using pair adjacent violators, PAV, can be used to determine the L_2 isotonic regression in $\Theta(n)$ time, L_1 in $\Theta(n \log n)$ time [3,42], and L_∞ on unweighted data in $\Theta(n)$ time. In Section 6 an algorithm taking $\Theta(n \log n)$ time is given for L_∞ isotonic regression on weighted data.

L_∞ isotonic regression on an arbitrary dag (poset) can easily be done in $\Theta(m)$ time for unweighted data (see Section 2.1.1), but for weighted data the fastest previously known algorithm is $\Theta(n \log^2 n + m \log n)$, due to Kaufman and Tamir [29]. Theorem 1 reduces this to $\Theta(m \log n)$.

Given weighted data (f, w) on dag $G = (V, E)$, for vertex $v \in V$, the *error of using r as the regression value at v*, $\mathrm{err}(v, r)$, is $w(v) \cdot |f(v) - r|$. Given vertices u and v, the weighted mean of their values, $\mathrm{mean}(f, w : u, v)$, is $[w(u) f(u) + w(v) f(v)] / [w(u) + w(v)]$. Note that $\mathrm{mean}(f, w : u, v)$ has equal error for the two vertices. Let $\mathrm{mean\_err}(f, w : u, v)$ denote this error.

There is a simple geometric interpretation of mean and mean_err: when regression values are plotted horizontally and the error is plotted vertically, the ray with x-intercept $f(u)$ and slope $-w(u)$ gives the error for using a regression value below $f(u)$ at u, and the ray with x-intercept $f(v)$ and slope $w(v)$ gives the error for using a regression value above $f(v)$ at v. The regression value where these lines intersect is $\mathrm{mean}(f, w : u, v)$, and the error of the intersection point is $\mathrm{mean\_err}(f, w : u, v)$. See Fig. 2.

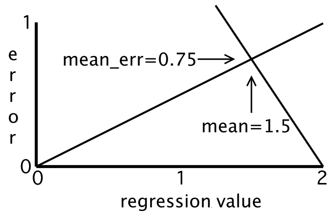


Fig. 2. The mean and mean_err of $(0, 0.5)$ and $(2, 1.5)$.

For $u, v \in V$, if $u \prec v$ and $f(u) > f(v)$ then u, v are a *violating pair* and $\text{mean\_err}(f, w : u, v)$ is a lower bound on the regression error. The only way the error at u can be less than $\text{mean\_err}(f, w : u, v)$ is if the regression value is larger than their mean. However, $u \prec v$ implies that the regression value at u is $\leq$ that at v, so the regression at v would have an error larger than the error for the mean. More generally, given a set $S \subset V$, the L_∞ mean of the weighted data on S is $\text{mean}(f, w : u', v')$, where

$$\text{mean\_err}(f, w : u', v') = \max\{\text{mean\_err}(f, w : u, v) : u, v \in S\}$$

2.1. Regression mappings

L_∞ isotonic regression is not always unique, so we examine properties of the regressions produced by various algorithms. An L_∞ *isotonic regression mapping*, informally called a regression mapping, takes a dag $G = (V, E)$ and weighted data (f, w) on G and maps them to an L_∞ isotonic regression of (f, w). Here we present several regression mappings. A more extensive discussion of their properties appears in [43].

2.1.1. Basic and Prefix

The most commonly used regression mapping, Basic, sets the regression value at vertex x to be $\text{mean}(f, w : u', v')$, where u', v' maximize $\{\text{mean\_err}(f, w : u, v) : u \preceq x \preceq v\}$ (if there are two such pairs then they have the same mean). The values are easily seen to be isotonic, and the regression error is optimal. Basic has been studied since the 1970's [49].

We introduce a closely related regression, Prefix, defined as follows: let

$$\text{pre}(v) = \max\{\text{mean}(f, w : u, v) : u \preceq v \text{ and } f(u) \geq f(v)\}$$

$$\text{Prefix}(u) = \min\{\text{pre}(v) : u \preceq v\}$$

For any $x \in V$, $\text{pre}(x) = \text{Basic}_x(f, w)(x)$, where Basic_x is the Basic isotonic regression on the subdag induced by the vertices $V_x = \{v \in V : v \preceq x\}$. The regression error on V_x is $\max\{\text{err}(v, \text{pre}(v)) : v \in V_x\}$. The construction of Prefix guarantees that it is isotonic, and optimality follows from that of Basic.

For unweighted data the calculation for Basic reduces to $(\max\{f(u) : u \preceq x\} + \min\{f(v) : x \preceq v\})/2$, which can be determined in $\Theta(m)$ time by using topological sorting and reverse topological sorting. Prefix can similarly be computed in $\Theta(m)$ time. For weighted data, suppose the transitive closure of G is given, and it has m' edges. It is straightforward to compute Prefix in $\Theta(m')$ time since the calculation of $\text{pre}(v)$ is linear in the number of v's predecessors. $\text{Basic}(v)$ involves pairs of predecessors and successors, but can be computed in linear time via linear programming [4,29]. Thus Basic too can be determined in $\Theta(m')$ total time but it is more complicated than Prefix.

Section 6 shows that Prefix can be efficiently calculated for linear and tree orderings without using the transitive closure.

2.1.2. Min, Max, and Avg

The Min L_∞ isotonic regression of (f, w) is the pointwise minimum of all L_∞ isotonic regressions of (f, w), and Max is the pointwise maximum. The pointwise minimum of isotonic functions is isotonic, and since at any vertex Min has an error the same as at least one L_∞ isotonic regression its maximum error is optimal. Hence it, and similarly Max, is an L_∞ isotonic regression. Avg is the pointwise average of Min and Max. The fastest L_∞ isotonic regression algorithm for general dags, Theorem 1, computes Min, Max, and Avg. Given the optimal error, all three can be computed in $\Theta(m)$ time (via the feasibility test in Section 3), and hence they can always be computed at least as fast as any other regression.

2.1.3. Strict

The *strict L_∞ isotonic regression* (Strict) is the limit, as $p \to \infty$, of L_p isotonic regression. It is easy to show that the limit exists for all weighted data and dags. It has been called the "best best" L_∞ isotonic regression [30]. Algorithms to compute Strict, and proofs of some of its properties, appear in [43]. Given the transitive closure, it can be computed in $\Theta(m' + n' \log n')$ time, where m' is the number of edges in the transitive closure and n' is the number of violating pairs.

2.2. Properties of regression mappings

There is a natural pointwise ordering on functions, denoted $\leq^p$, where $f \leq^p g$ iff $f(v) \leq g(v)$ for all $v \in V$. A regression mapping M is *monotonic* iff for any dag G and weights w on G, for functions f and g on G, if $f \leq^p g$ then $\mathsf{M}(f, w) \leq^p \mathsf{M}(g, w)$. For isotonic regression, monotonicity is a natural property to expect.

For $1 < p < \infty$, L_p isotonic regressions are monotonic mappings, and thus Strict is as well. Basic and Prefix are also monotonic, as can be seen from their construction. For unweighted data Avg = Basic, and hence it is monotonic, but for weighted data it isn't. E.g., for vertices $\{1, 2, 3\}$, let $w = 4, 4, 1$, $f = 2, 0, 2$, and $g = 3, 3, 3$. Then $f <^p g$, but $\mathsf{Avg}(f) = 1, 1, 3.5 \not\leq^p 3, 3, 3 = \mathsf{Avg}(g)$. Min and Max are not monotonic even for unweighted data. E.g., if $f = 1, 3, 1$ and $g = 1, 3, 3$ then $f \leq^p g$ but $\mathsf{Max}(f) = 2, 2, 2 \not\leq^p 1, 3, 3 = \mathsf{Max}(g)$. Similar examples hold for Min.

An important property of Strict is that it minimizes the number of large errors [43]. That is, if $g \neq \mathsf{Strict}(f, w)$ is an isotonic function on G, then there is a $C > 0$ such that g has more vertices with regression error $\geq C$ than does $\mathsf{Strict}(f, w)$, and for any $D > C$, g and $\mathsf{Strict}(f, w)$ have the same number of vertices with regression error $\geq D$ (it is not necessarily true that if $c < C$ then Strict has fewer vertices than g with regression error $\geq c$, but the concern is with the number of large errors). Min and Max are at the opposite end of the spectrum, in that for any vertex v, one or both of them has the largest regression error at v among all L_∞ isotonic regressions. For example, for the unweighted data 3, 1, 2, Strict is 2, 2, 2 while Basic and Prefix are 2, 2, 2.5, i.e., their last value has an unnecessary error. Max is even worse, being 2, 2, 3.

Min, Max and Min can result in values outside the data range, which might not be acceptable. E.g., for values 2, 3, 1, 2 with weights 1, 4, 4, 1, Min is −2, 2, 2, 2, Max is 2, 2, 2, 6, and Avg is 0, 2, 2, 4. While one could trim the values so that they lie within the data range, Prefix, Basic and Strict have this property automatically, as do all L_p isotonic regressions for $1 \leq p < \infty$.

3. Isotonic regression for arbitrary partial orders

As mentioned above, for arbitrary dags Prefix and Basic (and hence Min, Max, and Avg) can be calculated in $\Theta(TC(G) + m')$ time, where $TC(G)$ is the time to determine the transitive closure of G, and m' is the number of edges in the transitive closure. For arbitrary dags the transitive closure can be computed in $\Theta(\min\{nm, n^\omega\})$ time, where ω is such that matrix multiplication can be performed in $O(n^\omega)$ time. It is known that $\omega < 2.376$.

A generic approach for Min, Max and Min, not using transitive closure, is to couple a feasibility test with a search procedure. The feasibility test is given $\epsilon \geq 0$ and determines if there is an isotonic regression with L_∞ error $\leq \epsilon$, while the search procedure determines a sequence of ϵ values converging to the optimal value. Here the tests are constructive in that if there is a regression with error ϵ then the test produces one. This approach has also been used in L_∞ histogram construction and step function approximation [12,19].

Kaufman and Tamir [29] provided a feasibility test taking $\Theta(m \log n)$ time. A simpler test taking only $\Theta(m)$ time is as follows: given $\epsilon > 0$ and vertex v, v's *ϵ-window* is the real interval $[a_v, b_v]$, where $w(v)(f(v) - a_v) = w(v)(b_v - f(v)) = \epsilon$. An isotonic function g has regression error $\leq \epsilon$ iff $g(v) \in [a_v, b_v]$ for all $v \in V$. Thus (f, w) has an isotonic regression of error ϵ iff $s(v) = \max\{a_u : u \preceq v\}$ is one, which is iff $s(v) \leq b_v$ for all v. The isotonic constraint forces any isotonic regression of error $\leq \epsilon$ to be at least $s(v)$, and hence if $s(v) > b_v$ then no isotonic function can have error $\leq \epsilon$. s can be computed via topological sort in $\Theta(m)$ time. When ϵ is the optimal regression error then s is Min. Max can be found similarly.

Theorem 1. *Given weighted data (f, w) on a connected dag $G = (V, E)$, in $\Theta(m \log n)$ time the* Min, Max *and* Avg *L_∞ isotonic regressions can be determined.*

Proof. Since the optimal regression error is of the form $\mathsf{mean\_err}(f, w : u, v)$ for some $u, v \in V$, a straightforward approach is to do a binary search on these values, using a feasibility test to decide whether a larger or smaller value is needed. A naive implementation would explicitly compute all mean_err values, taking $\Theta(n^2 + m \log n)$ time. Kaufman and Tamir [29] noted that parametric search could be used instead, taking $\Theta(n \log n)$ time to produce $\Theta(\log n)$ error values which are evaluated via a feasibility search. Thus their total time is $\Theta(\log n(n + m \log n))$. Using our faster feasibility test reduces the time to $\Theta(m \log n)$. □

Linear and rooted tree orderings have $m = n - 1$, and d-dimensional grids have $\Theta(dn)$ edges, hence:

Corollary 2. *Given weighted data (f, w) on a linear, tree, or d-dimensional grid ordering, the* Min, Max *and* Avg *L_∞ isotonic regressions can be determined in $\Theta(n \log n)$ time, where for d-dimensional grids the implied constants are a function of d.* □

In Section 6 it is shown that Prefix can be computed in $\Theta(n \log n)$ time for linear and tree orders, but it is not known how to compute Prefix this quickly for d-dimensional grids, $d \geq 2$.

Unfortunately, parametric search is quite complex and completely impractical, making the theorem of only theoretical interest (see the survey article by Agarwal and Sharir [2]). The remaining sections of the paper give algorithms which avoid it.

4. Constraints on data or regression values

Here we give far more practical algorithms when the data has only a few different weights or values, or when the regression values are constrained to a given set.

4.1. Data with few different weights or few different values

In some applications the number of different weights is significantly less than n. For example, the author has encountered this scenario in computer simulations of complex experiments. A program of moderate accuracy was run for many points in the parameter space, while one of greater accuracy, but taking far longer, was run for only a few. Results of the latter were weighted more heavily than those of the former. In other situations there may be only a few different values but many weights.

While parametric search is impractical, for a modest number of different weights or different values a quite practical search is available.

Theorem 3. *Given weighted data (f, w) on a connected dag $G = (V, E)$, with t different weights, or t different values, the* Min, Max *and* Avg *L_∞ isotonic regressions can be found in $\Theta(tn + m\log n)$ time and the* Prefix *L_∞ isotonic regression can be found in $\Theta(tm)$ time.*

Proof. We give the search algorithm for weights, with the one for values being nearly identical. For Min, Max and Avg we first give a version taking $\Theta((tn + m)\log n)$ time, and then reduce this to the time claimed.

Suppose there are t weights $w_1 \ldots w_t$. For weight w_i let $Z_i = \{f(u) : w(u) = w_i,\ u \in V\}$, and let $z_i(1) \ldots z_i(k(i))$ be the values of Z_i in sorted order, where $k(i) = |Z_i|$. For vertex v, let $k(v, i)$ be the largest value such that $z_i(k(v, i)) \le f(v)$. A binary search over the range $1 \ldots k(v, i)$ will try to locate a j such that the mean error of $(z_i(j), w_i)$ and $(f(v), w(v))$ is the L_∞ isotonic regression error (of course, there may be no such value for a given v and i). Note that the error is decreasing in j.

The search occurs in parallel for all $v \in V$ and $1 \le i \le t$, using a linear time algorithm for determining order statistics. At any step let $e(v, i)$ denote the mean error of $(z_i(j), w_i)$ and $(f(v), w(v))$ where j is the middle of the remaining range in the search at v for weight w_i, i.e., initially $j = \lfloor(1 + k(v, i))/2\rfloor$. The weight of $e(v, i)$ is the size of the remaining range. A feasibility test is run on the weighted median of all tn of the $e(v, i)$ weighted values. At least $1/4$ of the sum of the sizes of the remaining ranges is eliminated at each step, so only $\Theta(\log n)$ iterations are needed.

To reduce the time further we use a more efficient approach for determining which mean_err values to calculate. For weights w_i and w_j, let M be the $k(i) \times k(j)$ matrix where $M(a, b)$ is the mean_err of $(z_i(a), w_i)$ and $(z_j(b), w_j)$ if $z_i(a) \ge z_j(b)$, and 0 otherwise. (M is only conceptual, it isn't actually created.) The rows and columns of M are sorted, so one can use searching on sorted matrices [20] as the search procedure. See Agarwal and Sharir [2] for a simple description: the procedure is easy to implement, it's the time analysis that's more complicated. Letting $K = k(i) + k(j)$, the search evaluates $\Theta(K)$ entries of M and uses $\Theta(\log K)$ feasibility tests, taking $\Theta(K)$ time beyond that of the feasibility tests. Thus the time is as claimed.

A simpler approach can be used for Prefix. For each weight w_i, using topological sorting, for all vertices v determine $\max\{f(u) : w(u) = w_i,\ u \preceq v\}$. From the t values at v one can determine pre(v) and hence Prefix can be computed in the time claimed. □

Note that Prefix can be computed faster than the time in Theorem 1 when $t = o(\log n)$. Min, Max and Avg can be computed in the same time bound by using Prefix to determine the optimal regression error.

4.2. Restricted regression values

Isotonic regressions with restricted values have been studied by several authors (see [3] and the references therein), including their use in various classification problems [18]. For integer-valued isotonic regression on a linear order, Goldstein and Kruskal [22] gave a $\Theta(n)$ time algorithm for the L_2 metric, and Liu and Ubhaya [33] gave a $\Theta(n^2)$ time L_∞ algorithm for unweighted data. Theorem 4 below, coupled with Theorem 1, shows that an integer-valued L_∞ regression can be found in $\Theta(m\log n)$ time for any dag. Theorem 6 shows that these ideas can be extended to find approximations with specified accuracy.

We use an approach applicable to arbitrary dags and regression values restricted to a set S of real numbers. An *S-valued L_p isotonic regression* is an S-valued isotonic function which minimizes the L_p regression error among all S-valued isotonic functions. S must be a closed discrete set, which implies that for any real number x, if $x \le \max\{s \in S\}$ then $\lceil x\rceil_S = \min\{s : s \ge x,\ s \in S\}$ is well-defined, and correspondingly if $x \ge \min\{s \in S\}$ then $\lfloor x\rfloor_S = \max\{s : s \le x,\ s \in S\}$ is well-defined. If $x < \min\{s \in S\}$ let $\lfloor x\rfloor_S = \min\{s \in S\}$, and if $x > \max\{s \in S\}$ then $\lceil x\rceil_S = \max\{s \in S\}$. Further, for $x \in S$ there is a unique successor if $x < \max\{s \in S\}$ and a unique predecessor if $x > \min\{s \in S\}$. We assume that predecessor, successor, $\lfloor\cdot\rfloor_S$, and $\lceil\cdot\rceil_S$ can be determined in constant time.

We will show that one can quickly convert an unrestricted isotonic regression into an S-valued one.

Theorem 4. *Given weighted data (f, w) on a connected dag $G = (V, E)$, a closed discrete set S of real numbers, and an unrestricted L_∞ isotonic regression of f, an S-valued L_∞ isotonic regression can be determined in $\Theta(m)$ time.*

We first establish a property connecting values of restricted and unrestricted isotonic regressions. For linear orders, Goldstein and Kruskal [22] showed that for L_2 integer isotonic regression, rounding the unrestricted isotonic regression gives the correct regression, and Liu and Ubhaya [33] showed that this is true for the L_∞ metric when the data is unweighted and the original regression is Basic. However, for weighted data, or when the original regression is not Basic, integer-valued L_∞ isotonic regression is a bit more complicated. For example, on vertices $\{0, 1\}$ with the values 0.6, 0 and weights 2, 1, the unique unrestricted L_∞ isotonic regression is 0.4, 0.4 yet the unique integer-valued one is 1, 1, not 0, 0. However, while rounding does not always give the optimal S-valued isotonic regression, it nearly does.

Lemma 5. *For $1 \le p \le \infty$, given weighted data (f, w) on a connected dag $G = (V, E)$, a closed discrete set S of real numbers, and an L_p isotonic regression g of (f, w), there is an S-valued L_p isotonic regression $\hat{g}$ such that $\hat{g}(v) \in \{\lfloor g(v) \rfloor_S, \lceil g(v) \rceil_S\}$ for all $v \in V$.*

Proof. Given p, let g be an unrestricted L_p isotonic regression, and let $\hat{g}$ be an S-valued L_p isotonic regression. If there are vertices v such that $\hat{g}(v) > \lceil g(v) \rceil_S$ then let u be one that maximizes the difference (a similar proof holds if there are vertices v such that $\hat{g}(v) < \lfloor g(v) \rfloor_S$). Let $A = \{v : g(v) = g(u) \text{ and } \hat{g}(v) = \hat{g}(u),\ v \in V\}$. For any vertex $v \notin A$ which is a predecessor of an element of A, $\hat{g}(v) < \hat{g}(u)$ because the fact that it isn't in A means either $\hat{g}(v) < \hat{g}(u)$ or $g(v) < g(u)$, but if only the latter holds then u does not optimize the difference. Similarly, if $v \notin A$ is a successor of a vertex in A then $g(v) > g(u)$.

Let $x \in S$ be the predecessor of $\hat{g}(u)$. Note that $x \ge \lceil g(u) \rceil_S$. Let $\hat{g}'$ be the S-valued isotonic function which is $\hat{g}$ on $V \setminus A$ and x on A. If the regression error of $\hat{g}'$ is less than that of $\hat{g}$ then $\hat{g}$ was not optimal. If the regression error of $\hat{g}'$ is larger then the weighted mean of A is greater than x. Let $y = \min\{x, \min\{g(v) : g(v) > g(u),\ v \in V\}\}$. Raising the value of g on A to y, and keeping it the same elsewhere, would decrease the L_p error and maintain the isotonic property, contradicting g's optimality.

Finally, if the L_p regression error of $\hat{g}'$ is the same as that of $\hat{g}$ then recursively consider $\hat{g}'$. Since S is discrete there are only finitely many values between x and $\lceil g(u) \rceil_S$, so the process can only recurse a finite number of times. □

Proof of Theorem 4. Let g be an unrestricted L_∞ isotonic regression and for $r \in S$ let $\mathcal{L}_r$ be the set $\{v : \lfloor g(v) \rfloor_S = r,\ v \in V\}$. Note that this is a union of level sets in g. No matter what choices are made for the regression values within $\mathcal{L}_r$, since these choices are either r or its successor r^+ in S, they do not impose any isotonic restrictions on the choices for other level sets since the values used on $\mathcal{L}_r$ are at least as large as the upper values in any predecessor and no larger than the lower values in any successor. Therefore we can consider each $\mathcal{L}_r$ separately.

For $v \in \mathcal{L}_r$, define functions down_err and up_err by

$$\text{down\_err}(v) = \max\{\text{err}(u, r) : u \preceq v,\ u \in \mathcal{L}_r\}$$

$$\text{up\_err}(v) = \max\{\text{err}(u, r^+) : u \succeq v,\ u \in \mathcal{L}_r\}$$

down_err(v) is a lower bound on the L_∞ isotonic regression error if the regression value at v is r since all predecessors in $\mathcal{L}_r$ must also have regression value r, and similarly up_err(v) is a lower bound if it is r^+. Let $\hat{g}$ be defined by

$$\hat{g}(v) = \begin{cases} r & \text{if down\_err}(v) \le \text{up\_err}(v) \\ r^+ & \text{otherwise} \end{cases}$$

Since down_err is monotonic increasing on $\mathcal{L}_r$ and up_err is monotonic decreasing, $\hat{g}$ is isotonic increasing. At each vertex v, changing the choice for $\hat{g}(v)$ can never decrease the overall regression error, and hence an optimal S-valued L_∞ isotonic regression has been found.

down_err and up_err can be computed by topological sorting, completing the proof of the theorem. □

Thus for unweighted data an S-valued isotonic regression can be found in $\Theta(m)$ time. For weighted data it could be done in $\Theta(m \log n)$ time via Theorem 4, but for small S one can do better.

Theorem 6. *Given weighted data (f, w) on a connected dag $G = (V, E)$, and given a finite set S of real numbers, in $\Theta(m \log |S|)$ time an S-valued L_∞ isotonic regression can be determined.*

Proof. The algorithm follows an approach in [44], which was used for L_p isotonic regression for $1 \le p < \infty$. Let $S = \{s_1, \ldots, s_d\}$, let $[s_a, s_b]$, $1 \le a \le b \le d$ denote the interval $s_a \ldots s_b$ of values in S, and for $v \in V$ let $\text{err}_S(v, a, b)$ denote $\min\{\text{err}(v, t) : t \in [s_a, s_b]\}$. The algorithm determines intervals of S in which the regression value for v will be assigned, where the intervals are progressively halved until they are a single value, which is the regression value of v. At each stage

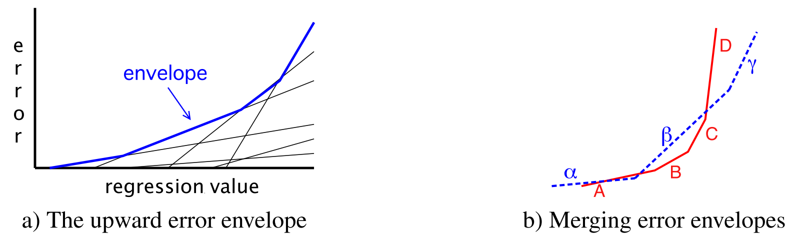


Fig. 3. Error envelopes.

the assignment is isotonic in that if $u \prec v$ then either u and v are assigned to the same interval or the intervals have no overlap and u's interval has smaller values than v's. Initially each vertex is assigned to all of S, i.e., to $[s_1, s_d]$.

Suppose x is currently assigned to $[s_a, s_b]$, and let $c = \lfloor (a+b)/2 \rfloor$. Define

$$\text{down\_err}(x) = \max\{\text{err}_S(u, a, c) : u \preceq x,\ u \text{ currently assigned to } [s_a, s_b]\}$$
$$\text{up\_err}(x) = \max\{\text{err}_S(v, c{+}1, b) : v \succeq x,\ v \text{ currently assigned to } [s_a, s_b]\}$$

$\text{down\_err}(x)$ is a lower bound on the L_∞ regression error for vertices currently assigned to $[s_a, s_b]$ if x is assigned to $[s_a, s_c]$ in the next stage, and $\text{up\_err}(x)$ is a lower bound if it is assigned to $[s_{c+1}, s_b]$. Assign x to $[s_a, s_c]$ if $\text{down\_err}(x) \le \text{up\_err}(x)$ and to $[s_{c+1}, s_b]$ otherwise. As before, this assignment is isotonic and can be determined in $\Theta(m)$ time.

The proof that the final assignments form an S-valued L_∞ isotonic regression is similar to that in [44] and will be omitted. □

5. Error envelopes

Using intersecting line segments to find maximum violating pairs is a well-known approach to L_∞ regression. Given data (f, w) on a set S, for $x \ge \min\{f(v) : v \in S\}$ let $g(x)$ be the L_∞ regression error of using x as the regression value for $\{v \in S : f(v) \le x\}$. This has a simple geometric interpretation: for each $v \in S$, plot the ray with x-intercept $f(v)$ and slope $w(v)$. g is the set of line segments with no points above them. See Fig. 3 a). We call this upper envelope the *upward error envelope*, and $\max\{\text{err}(v, x) : v \in S, f(v) \ge x\}$ is the *downward error envelope*. If the envelopes intersect at regression value r, then r is the L_∞ weighted mean of S.

It is straightforward to use balanced search trees so that the upward and downward error envelopes can be maintained and in $\Theta(\log |S|)$ time one can insert a weighted value, determine the value of the envelope at a given regression value, determine the regression value having a given error, and determine the point of intersection of a ray with an envelope. Given the error envelopes one can determine their intersection in $\Theta(\log |S|)$ time and thus one can determine the mean of S in $\Theta(|S| \log |S|)$ time. It is possible to do this in $\Theta(|S|)$ time by viewing it as a 2-dimensional linear programming problem, as noted in Section 2.1.1. This is needed for Basic to achieve its time bound, but is unnecessary for Prefix.

A slight variation will prove useful later. Given the downward error envelope E of a set S of vertices, and given a vertex v, an *error query of E* determines the intersection of the error envelope with the ray with x-intercept $f(v)$ and slope $w(v)$, i.e., the ray representing the error in using a regression value $\ge f(v)$ at v. The maximum of the error queries for vertices in S is the regression error of using the L_∞ mean on S.

Another operation involving envelopes is merger. The only special aspect is that after two envelopes are merged, using standard merging of balanced binary search trees, some of the segments may need to be removed. E.g., in Fig. 3 b), when the envelope with Roman labeling and the one Greek labeling are merged, segments B, C, and γ will be removed. The endpoints of some of the remaining segments also change. Using a relatively straightforward procedure, once the envelopes are merged the removal of segments and changing of endpoints can be performed in $\Theta(a + b \log n)$ time, where a is the size of the smaller envelope and b is the number of segments removed.

Given sets S_1, S_2 of vertices, a *bipartite maximum violators* operation determines a pair of vertices $u_1 \in S_1$, $u_2 \in S_2$ that maximize $\text{mean\_err}(f, w : v_1, v_2)$ among all pairs where $v_1 \in S_1$, $v_2 \in S_2$ and $f(v_1) \ge f(v_2)$. If there are no such pairs then it returns the empty set. Note that S_1 and S_2 might overlap.

Lemma 7 *(Repeated violators). Given sets S_1 and S_2 of weighted values, in $\Theta(N \log N)$ time one can do an arbitrary sequence of $O(N)$ deletion and bipartite maximum violators operations, where $N = |S_1| + |S_2|$.*

Proof. A downward error envelope for S_1, and an upward one for S_2, will be used to determine the violators. The envelopes will be threaded so that one can determine the segments adjacent to a given segment in constant time. They are "semi-dynamic" since only deletions change them once they are initially constructed. Hershberger and Suri [26] showed how to create the initial envelopes, and perform all the deletions, in $O(N \log N)$ time (they solved the semi-dynamic problem of maintaining the convex hull of a set of planar points which, by duality, can be used to solve the error envelope problem).

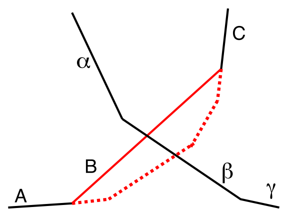


Fig. 4. Dynamic envelope intersections.

Thus the only time remaining to be analyzed is the time needed to find the maximum violators. Given the envelopes, the initial maximum violators can be found in $\Theta(\log N)$ time. For subsequent violators, the only case of interest is when the segment deleted was one of the ones in the intersection. For example, in Fig. 4, suppose initially B and β intersect, and then B is deleted, resulting in new segments in the upward error envelope represented by dashed lines. The new intersection can involve β or a successor, but not α. Further, on the upward envelope it might involve C since that segment may have lengthened, but cannot involve any successor of C. It might, however, involve some predecessor of C. The intersection is found by checking C, then its predecessor, etc., until it is found. During this process it may also be discovered that the intersection will involve a successor of α, etc.

While a single deletion may involve $\Theta(N)$ segments, if the new intersection does not involve C then no future intersection can involve C until its predecessor has been deleted. Therefore the total number of segments examined over all iterations is $O(N)$. □

For integer $k \geq 1$, a dag G is *k-transitive closed* if whenever $u \prec v$ there is a path of length $\leq k$ from u to v. Error envelopes can be used to extend approaches for transitive closures to 2-transitive closed dags.

Proposition 8. *Given weighted data (f, w) on a 2-transitive closed connected dag $G = (V, E)$, the* Prefix *isotonic regression can be determined in $\Theta(m \log m)$ time, and* Min, Max *and* Avg *in $\Theta(m + n \log n)$ time.*

Proof. For Prefix, to compute $\text{pre}(v)$ for vertex v, let $\text{pre}'(v) = \max\{\text{mean\_err}(f, w : u, v) : (u, v) \in E \text{ and } f(u) \geq f(v)\}$. All other predecessors of v are at distance 2, so for every vertex p, compute the downward envelope D_p of $\{(f(u), w(u)) : (u, p) \in E\}$. Let $\text{err}(p, v)$ denote the value of an error query for v on D_p. Then $\text{pre}(v) = \max\{\text{pre}'(v), \max\{\text{err}(p, v) : (p, v) \in E\}\}$. The time to create the envelopes and do the queries is $\Theta(m \log m)$.

For the other regressions, first compute the maximum error for adjacent violating pairs. Then for every vertex create the downward envelope of the data at its preceding neighbors and the upward envelope of its succeeding neighbors and find the error of their intersection. The optimal regression error is the maximum of all of these values. To create the envelopes efficiently, first sort the weights. Going through the vertices in this ordering, put their data in the appropriate envelopes. It is easy to show that by inserting in this order the envelopes can be created in linear time, as can finding their intersection. □

In [45] this approach was further extended by using a construction similar to Steiner 2-transitive-closure spanners [8]. Dag $G = (V, E)$ is embedded into dag $G' = (V', E')$ where $V \subset V'$ and for any pair $u, v \in V$, u precedes v in G iff u precedes v in G'. Further, if u precedes v in G then there is a path from u to v in G' of length ≤ 2. Data on G is extended to G', and an isotonic regression on G' induces one on G. See [45] for further details. Most of the material there appeared in a 2008 version of the current paper.

One can interpret L_∞ isotonic regression as minimizing the maximum "penalty" at the vertices, given the isotonic requirement. This can be applied to rather general penalty functions, such as a classification problem with ordered classes and penalties for misclassifying a vertex. The time of the algorithms depend on the complexity of determining the regression value minimizing the error over a set of weighted values. One extension where there is no increase in time is when the error is linear but the weight for using regressions values larger than the data value is different than the weight for using regression values less than the data. For example, overestimating the capacity of a bridge is more serious than underestimating it. None of the operations on envelopes require that the weights for errors above and below the data value be the same.

6. Linear and tree orderings

Linear orders are the most important dag by far, and isotonic regression on linear and tree orders has been used for decades (see the extensive references in [5,38]). Recent applications of isotonic regression on trees include analyzing web data [11,35]. For rooted trees the ordering is towards the root.

For linear orders it is widely known that the L_1 and L_2 isotonic regressions can easily be found in $\Theta(n \log n)$ and $\Theta(n)$ time, respectively. For trees the fastest isotonic regression algorithms use a bottom-up pair adjacent violators (PAV)

```
{DecErrEnv($v$) is the downward error envelope of $\{u : u \preceq v\}$ for $v \in V$}

Using a topological sort traversal,
    DecErrEnv($v$) = $\cup$\{DecErrEnv($u$) : $(u, v) \in E$\}
    Insert $(f(v), w(v))$ into DecErrEnv($v$)
    pre($v$) = result of error query of DecErrEnv($v$)

Using topological sort traversal in reverse order,
    Prefix($v$) = min\{pre($u$) : $v \preceq u$\}
```

Algorithm A. Computing Prefix using topological sorting on dag $G = (V, E)$.

approach [47], taking $\Theta(n \log n)$ time for the L_2 [36] and L_1 [44] metrics. Corollary 2 shows that Min, Max and Avg can be determined in $\Theta(n \log n)$ time via parametric search. By using a bottom-up approach Prefix can be computed in the same time without requiring parametric search, and hence Min, Max and Avg too can be computed in the same time without parametric search.

Theorem 9. *Given weighted data (f, w) on a rooted tree $T = (V, E)$, Algorithm A computes the* Prefix *L_∞ isotonic regression in $\Theta(n \log n)$ time.*

Proof. Envelope merging was described in Section 5. A balanced tree of size a can be merged with one of size b, $a \le b$, in $\Theta(a(1 + \log(b/a)))$ time, hence all of the initial union and insertion operations can be performed in $\Theta(n \log n)$ time [10]. The time of the segment removal during each merge is linear in the number of elements in the smaller envelope, plus the time needed to remove intervals. An interval is removed at most once throughout all of the mergers, so the total time for removals is $O(n \log n)$, and since the processes are initiated by segments in the smaller tree, the total time for initiations is $O(n \log n)$. This completes the proof of Theorem 9. □

6.1. *River isotonic regression for trees*

Isotonic regression on trees arises in a variety of hierarchical classification and taxonomy problems in data mining and machine learning. Recent variations of isotonic regression for such problems include requiring that the regression value at a node be greater than or equal to the sum of the values of its children [7], and requiring that for each non-leaf node the regression value is the largest value of a child. Punera and Ghosh [37] called the latter “strict classification monotonicity”, but that is the opposite of the usual meaning of “strictly monotonic”. Here it is called *river isotonic regression* due to its similarity to river naming conventions. They solved the L_1 problem in $\Theta(n^2 \log n)$ time. We use a Prefix-like approach to show:

Theorem 10. *Given weighted data (f, w) on a rooted tree $T = (V, E)$, in $\Theta(n \log n)$ time an L_∞ river isotonic regression can be determined. If the data is unweighted then this can be done in $\Theta(n)$ time.*

Proof. For the unweighted case, at vertex v let a be the largest data value in v's subtree. For a path p from v to a leaf node under it, let b be the smallest value on p. If the regression values along the path are all the same, then the overall regression error in v's subtree is at least $(a - b)/2$, and choosing $x = (a + b)/2$ as the regression value at v achieves this error along p. Given this choice, for any isotonic regression on the remainder of v's subtree, reducing any values larger than x to x cannot increase the overall error. At each node, keep track of the minimal value on the path beneath with maximal minimal value, the child that achieves this, and the maximum value at all vertices in the subtree. If v were the root than this path gives an optimal regression value for it. These can all be determined in $\Theta(n)$ time by a bottom-up process.

Just as for Prefix, a top-down second stage is used, but it is slightly more complicated since the value at the root may be larger or smaller than its children. The value at the root is that of the bottom-up phase. Recursively, if the value at a node v has been determined to be x in the top-down stage, let w be the child that was recorded on the bottom-up phase. w's value will be x. For all other children, their value is the minimum of x and their value on the bottom-up phase. It's relatively straightforward to show that this produces an optimal river isotonic regression.

For the weighted case, for a vertex v again let p be a path from a leaf to v. Given the downward envelope for all vertices in v's subtree, and the upward envelope for vertices in p, using their intersection as the regression value at v gives a lower bound on the total regression error in the subtree when all vertices in p must have the same value. Choosing p to minimize the error of the intersection gives the minimal possible error bound for v's subtree. A top-down stage provides the final regression value.

Determining the path to use in the weighted case is more complex than for the unweighted case. For example, suppose there is vertex v with two leaf children. The left child has data value 0 and weight 1, the right child has value 1 and weight 2, and v has value 2 and weight 2. Then on the upward stage the value at v will be 1.5, using the path to the right child. Suppose v's parent u has a descendent with value 11 and weight 1, and that the optimal path for u involves v. Then the optimal path is from u to v's left child, with a regression value of 5 and error 6. Had the path terminated at v's right child then the regression value would have been $4\ \frac{2}{3}$, with error $6\ \frac{1}{3}$. Thus, even though u's path passes through v, the continuation to a leaf is different than for v.

To keep track of the correct path information upward envelopes are used, but are slightly different. At vertex v, the envelopes for its children are merged together. However, rather than keeping the maximum error for each regression value, the smallest value is kept since there is a path achieving it. Once the children's envelopes are merged, then the segment corresponding to v is added, but it is inserted keeping track of the maximum error at a regression value since any path through v must include it. It's straightforward to show that the envelope merging operations can still be done in $\Theta(n \log n)$ total time. A final detail is that to determine the values during the top-down stage one needs to determine how to assign values to children. This requires the envelopes used when going up, so as they are being created they are maintained in persistent form so that intermediate stages can be recovered. □

6.2. Unimodal regression for linear orders

Given weighted data on a linear order $v_1 < \ldots < v_n$, a *unimodal regression* g is a regression that minimizes the regression error subject to the unimodal constraint

$$g(v_1) \le g(v_2) \ldots \le g(v_k) \ge g(v_{k+1}) \ldots \ge g(v_n)$$

for some k, $1 \le k \le n$. Unimodal orderings are also known as umbrella orderings. Optimal modes may not be unique, which can be seen by considering the unweighted data 1, 0, 1. This example shows there need not be a unimodal regression corresponding to Max, i.e., one which is pointwise as large as any other.

Several unimodal regression algorithms [13,42,48] have been applied to "competing failure modes" problems [17,23,24]. For example, in dose-response settings, as the dose increases the efficacy increases, but toxicity increases as well. The goal is to find a dose that maximizes the probability of being both efficacious and nontoxic [17,24]. A multi-arm bandit model is used, one bandit per dose, where the expected value of the bandits is assumed to be unimodal. The weights reflect the uncertainty in the expected value. An adaptive trial design is used, where after each observation, unimodal regression is used to decide which dose to use next. Unimodal bandit models have been examined since the 1980's [25].

A prefix approach was used in [42] to find the unimodal regression of a linear order, taking $\Theta(n)$ time for L_2, $\Theta(n \log n)$ for L_1, and $\Theta(n)$ for L_∞ with unweighted data. For this approach the information that needs to be computed at each vertex v_k is the regression error of an isotonic regression on $v_1 \ldots v_k$. This is the maximum of the error of using $\mathsf{pre}(i)$ at v_i, $1 \le i \le k$, which can be computed as pre is being computed. The corresponding information for decreasing isotonic regressions on $v_k \ldots v_n$ is also determined (computed in reverse order). An optimal mode vertex for L_∞ is one that minimizes the larger of these two errors. Using Algorithm A gives the following corollary which was also obtained by Lin et al. [31].

Corollary 11. *Given weighted data (f, w) on a linear order of n vertices, an L_∞ unimodal regression can be determined in $\Theta(n \log n)$ time.* □

6.3. Unimodal regression for trees

For an undirected tree the unimodal problem is to find an optimal root and the isotonic regression on the resulting rooted tree. Agarwal, Phillips and Sadri [1] were motivated to study this problem by an example concerning determining the surface topology in geographic information systems. They showed that for a tree with unweighted data the L_2 unimodal regression, with a Lipschitz condition imposed, can be found in $\Theta(n \log^3 n)$ time. In a machine learning context, Yu and Mannor [51] gave active learning algorithms for locating the root of a tree for a multi-arm bandit model with a bandit at each vertex, where the expected reward is unimodal. This is similar to the use of unimodal regression in response adaptive dose-response designs mentioned above. Bandit models using rooted trees appear in [15].

The algorithms below are quite different than those for unimodal regression on a linear order, in that they first locate the root without creating regressions, and then any algorithm for rooted trees can be applied. For unweighted data, locating a root is simple.

Proposition 12. *Given unweighted data f on an undirected tree $T = (V, E)$, in $\Theta(n)$ time an L_∞ unimodal regression can be found.*

Proof. Let x be a vertex where f achieves its maximum and let g be an optimal L_∞ unimodal regression on T. Let g' be the function on T given by $g'(v) = \min\{g(v), g(x)\}$ for all $v \in V$. If $\epsilon = |f(x) - g(x)|$, then $||f - g||_\infty \ge \epsilon$. Since $f(v) \le f(x)$ for all $v \in V$, $|g'(v) - f(v)| \le \max\{\epsilon, |g(v) - f(v)|\}$. Thus $||g' - f||_\infty \le ||g - f||_\infty$, and, since g was optimal, so is g'. Further, g' is isotonic when the tree is directed with x as its root.

Thus, to find a unimodal regression it suffices to locate a vertex with maximum data value, orient the tree with this as the root, and use Prefix to determine the isotonic regression. □

For weighted data it is not immediately obvious which vertices can be optimal roots, but there are edges for which one can quickly determine their correct orientation. The *path from u to v*, for $u, v \in V$, means the simple path from u to v. For weighted data (f, w) on the undirected tree $T = (V, E)$, let u, v be such that $\mathsf{mean\_err}(f, w: u, v) = \max\{\mathsf{mean\_err}(f, w: p, q) : p, q \in V\}$. Suppose $f(u) < f(v)$. If in the directed tree u is on the path from v to the root then they are violators and the regression error is at least $\mathsf{mean\_err}(f, w: u, v)$. This is the error of the regression where all values are $\mathsf{mean}(f, w: u, v)$, and hence every vertex is an optimal root. To obtain a smaller regression error it must be that u is not on the path from v to the root of the directed tree, in which case the edge incident to u on the simple path from u to v is directed towards the root. This will be used to show the following:

Theorem 13. *Given weighted data (f, w) on an undirected tree $T = (V, E)$, in $\Theta(n \log n)$ time an L_∞ unimodal regression can be found.*

Proof. The algorithm locates the root via recursion. Locate a pair of vertices u, v which maximize mean_err. If $f(u) < f(v)$ then let e be the edge incident to u on the path towards v. Remove e, which partitions T into trees $T_1 = (V_1, E_1)$, $T_2 = (V_2, E_2)$, where $u \in V_1$. Recursively find an optimum root r for a unimodal regression of T_2. Lemma 14 proves that r is an optimal root for unimodal regression on T. Once an optimal root is known the Prefix isotonic regression can be determined in $\Theta(n \log n)$ time (Theorem 9).

To find u, v, set $S_1 = S_2 = V$ and perform a bipartite maximum violator operation. Once T_1 is determined, all weighted values corresponding to it are removed from S_1 and S_2 and the resulting sets are used for the subproblem on T_2. The repeated violators lemma (Lemma 7) shows that one can find pairs of maximum violators during all recursive steps in $\Theta(n \log n)$ total time.

Knowing u and v does not immediately tell one which edge incident to u is e. Lemma 15 shows that one can find e in $O(|V_1|)$ time. Using this, T_1 can be removed from T, creating T_2, in $\Theta(|V_1|)$ time. Thus the total time for recursively shrinking the subtrees is $\Theta(n)$. $\square$

Lemma 14. *For an undirected tree $T = (V, E)$ with weighted data (f, w), let u_1, u_2 be a pair which maximizes* mean_err *and for which $f(u_1) < f(u_2)$. Let $\{u_1, x\}$ be an edge on the path from u_1 to u_2. Let $T_1 = (V_1, E_1)$, $T_2 = (V_2, E_2)$ be the subtrees of T induced by removing this edge, where $u_1 \in T_1$ and $x \in T_2$. Then an optimal root for L_∞ unimodal regression of (f, w) restricted to T_2 is an optimal root for L_∞ unimodal regression of (f, w) on T.*

Proof. If the minimal regression error of a unimodal regression on T is $\mathsf{mean\_err}(f, w: u_1, u_2)$ then all vertices are optimal roots. Otherwise the regression value at u_2 must be $> \mathsf{mean}(f, w: u_1, u_2)$ and at u_1 is $< \mathsf{mean}(f, w: u_1, u_2)$, and therefore any optimal root is in T_2. Let g be a unimodal regression on T_2 with root r, and let h be a unimodal regression on T. Define g' on T as follows:

$$g'(v) = \begin{cases} h(v) & \text{if } v \in T_1 \\ \max\{h(u_1), g(v)\} & \text{if } v \text{ is on the path from } r \text{ to } x \\ g(v) & \text{otherwise} \end{cases}$$

Since the root of h is in T_2, h restricted to T_1 has u_1 as its root, and the construction insures that g' is unimodal on T, with root r. Since g and h are optimal, to show that g' is optimal we only need to show that on the path from r to x its error is no worse than that of g or h.

For an optimal regression the value at u_2 is $> h(u_1)$, and hence on the path from r to u_2, $g > h(u_1)$. Similarly, on the path from x to u_2, $h \geq h(u_1)$. Since the path from r to x is a subset of the union of the paths from r to u_2 and x to u_2, the value of g' on the path from r to x is between (or equal to) the values of g and h, and hence its error is no more than the maximum of theirs. Thus it too is optimal. $\square$

Lemma 15. *Given an undirected tree $T = (V, E)$ and vertices $u, v \in V$, $u \neq v$, let $x \in V$ be such that the edge $\{u, x\}$ is on the path from u to v. Then x can be determined in $O(|T'|)$ time where T' is the maximal subtree of T containing u but not containing x.*

Proof. Let T' denote the maximal subtree of T containing u but not containing x. Let $p_1, \ldots, p_k$ be the neighbors of u. Viewing u as the root, let T_i denote the subtree with root p_i, $i = 1, k$. On each T_i a preorder traversal will be used. A global traversal is performed as follows: visit the first node in the traversal of T_1, then the first node in the traversal of $T_2, \ldots$, first node in T_k, then the 2nd node in the traversal of T_1, 2nd node in the traversal of $T_2, \ldots$. If v is reached while traversing T_j, or for every T_i, $i \neq j$, the final node in the traversal of T_i has been reached, then $x = p_j$ and $T' = \{u\} \cup \{T_i : i = 1 \ldots k,\ i \neq j\}$. In all cases, x is identified and the number of nodes examined in the traversal of T_j is no more than the number of nodes traversed in the other subtrees, i.e., in T'. $\square$

7. Final comments

For an arbitrary dag, Theorem 1 shows that the Min, Max and Avg L_∞ isotonic regressions can be found in $\Theta(m \log n)$ time. It is an open question as to whether this is optimal, and the optimal times for Prefix and Basic are also unknown.

Unfortunately the algorithm in Theorem 1 is based on parametric search, which is quite infeasible, so practical algorithms were given for various settings. While the emphasis is on worst-case time, Appendix A provides a practical randomized algorithm taking $\Theta(m \log n)$ time with high probability.

This paper introduced the Prefix regression. Basic has been extensively studied and is trivial to compute for unweighted data, but for weighted data Prefix is more useful. Prefix can be computed by reusing calculations at one vertex to reduce those at successors. For linear and tree orders, river isotonic regression, and unimodal regression, it can be determined in $\Theta(n \log n)$ time, obviating the need for the parametric search used by Chun et al. [13]. While there are now linear time algorithms for linear and tree orders [46], they use feasibility tests and yield the less desirable Min, Max or Avg regressions. That paper also shows that these regressions can be computed in $\Theta(n \log^{d-1} n)$ time for d-dimensional points with component-wise ordering, and Prefix can be computed in $\Theta(n \log^{d} n)$ time. Isotonic regression on points in d-dimensional space, which corresponds to having d independent variables, has been extensively studied [5,9,21,38,41,45].

Beyond their computational time, an important aspect of L_∞ isotonic regression algorithms is the properties of the regressions they produce. The algorithms in Theorem 1 and Appendix A use feasibility tests, which results in regressions (Min, Max, or Avg) which can have undesirable properties (see Section 2.2). Prefix and Basic are somewhat better, and Strict is the best but more difficult to compute. These issues, and algorithms for Strict, are pursued in [43].

Overall, among the current algorithms for L_∞ isotonic regression on arbitrary dags there is an inverse relationship between an algorithm's time and the desirability of the regressions it produces.

Acknowledgments

Research partially supported by NSF grant CDI-1027192, DOE grant DE-FC52-08NA28616, and King Abdulaziz University.

Appendix A. Randomized $\Theta(m \log n)$ expected time algorithm

As noted earlier, the algorithm in Theorem 1 determines L_∞ isotonic regression in $\Theta(m \log n)$ time, but its use of parametric search makes it impractical. Parametric search determines a sequence of potential regression errors that converge to the optimal value, with each error being checked via a feasibility test. This approach has also been applied to problems such as determining optimal L_∞ step functions with a specified number of steps [19]. Here we replace parametric search with a simple randomized search to determine the sequence of regression errors to be tested. It takes $\Theta(m \log n)$ expected time, not the worst-case time considered everywhere else in the paper. However, it achieves this time with high probability.

As before, we consider the problem of creating an L_∞ isotonic regression of the graph $G = (V, E)$ with weighted data (f, w), and we reduce to the connected case and hence $m \geq n - 1$. The approach is based on the observation that the optimal regression error ϵ^* is the maximum $\text{mean\_err}(f, w : u, v)$ among all violating pairs $u, v \in V$, i.e., where $u \prec v$ and $f(u) > f(v)$. Determining whether $u \prec v$ may take $\Theta(m)$ time, so we ignore the violating condition and consider the larger set $R = \{\text{mean\_err}(f, w : u, v) : u, v \in V, u \neq v\}$. It might be a multiset, but that does not affect the algorithm nor analysis. Algorithm B locates $\epsilon^* \in R$.

Step 3 requires explanation. For a vertex p let U_p be the "upper" interval of values $\geq f(p)$ where the weighted error at p is in $[a_1, a_2)$, and L_p the "lower" interval of values $\leq f(p)$ where the error is in $[a_1, a_2)$. For any vertices p, q, if $f(p) < f(q)$ then $\text{mean\_err}(f, w : p, q) \in [a_1, a_2)$ iff $U_p \cap L_q \neq \emptyset$. This gives an easy way to construct B. First sort the smallest endpoints of the upper intervals of all vertices. Let T denote this sorted array. Then, for all $q \in V$, if $L_q = [c, d)$, find the first value x in T where $x \leq c$, and then sequentially go through T until a value is encountered that is $> d$. For every value y encountered in this scan (except the last), if it is the smaller endpoint of U_p, then add $\text{mean\_err}(f, w : p, q)$ to B. A similar procedure is used where the smallest endpoints of the lower intervals are sorted and the scans are based on the upper intervals. After the two sort and scan procedures, all of the elements of B have been determined. The time is $\Theta(n \log n + |B|)$.

Step 1 can be violated a slight bit, in that the pairs can be chosen randomly with replacement. It may be that the same pair is chosen more than once, but the expected number of such occurrences is so small it doesn't affect the analysis in any meaningful way and it can simplify the implementation. For Step 2, utilizing a linear time median algorithm makes the time $\Theta(m \log n)$. Step 4's time is $\Theta(|B| + m \log n)$.

The final aspect is to determine $|B|$. To bound the probability that $|B| > n \ln n$, note that this is true iff there is an interval of $n \ln n$ consecutive elements of R which contains ϵ^* but no elements of A. For any set $C \subset R$ of size $n \ln n$, the probability that a specific element of A is not in C is $1 - \frac{2 \ln n}{n-1}$, and hence the probability that none are in it is $< (1 - \frac{2 \ln n}{n-1})^{n \ln n} \approx 1/n^{2 \ln n}$. There are $n \ln n$ intervals of R of length $n \ln n$ which contain ϵ^*, hence the probability that $|B| > n \ln n$ is $\approx \ln n / n^{2 \ln n - 1}$. Thus the total expected time is $\Theta(m \log n)$, and this is achieved with high probability. The worst-case time is $\Theta(n^2 + m \log n)$.

```
1. Select a random subset $A \subset R$ of size $n \ln(n)$ by randomly selecting $n \ln(n)$ pairs of vertices $u \neq v$ and adding $\text{mean\_err}(f, w : u, v)$ to $A$. Also add $\{0, \infty\}$ to A.
2. Use a binary search, incorporating feasibility tests, to determine consecutive elements $a_1, a_2 \in A$ such that $a_1 \leq \epsilon^* < a_2$.
3. Create the set $B = \{x : a_1 \leq x < a_2,\ x \in R\}$.
4. Use a binary search on B, incorporating feasibility tests, to locate $\epsilon^*$.
```

Algorithm B. Randomized algorithm to determine optimal regression error ϵ^*. $R = \{\text{mean\_err}(f, w : u, v) : u, v \in V\}$.

Once ϵ^* is known, Min, Max and Avg can be computed in $\Theta(m)$ worst-case time. Thus:

Theorem 16. *Given weighted data (f, w) on a connected dag $G = (V, E)$, by using Algorithm B the* Min, Max, *and* Avg *L_∞ isotonic regressions can be determined in $\Theta(m \log n)$ expected time with high probability.* □

Feasibility tests are used for additional L_∞ optimization problems such as histogram construction and step function approximation [12,19], isotonic regression with a specified number of steps, and 1-dimensional weighted k-centers. Algorithm B provides a simple, systematic approach to these problems, yielding algorithms that take $\Theta(n \log n)$ expected time with high probability. A $\Theta(n \log n)$ expected-time algorithm for step function approximation appears in [32], but it is much more complicated. Since feasibility tests are usually quite fast, solutions based on Algorithm B are quite practical.

References

[1] K. Agarwal, J.M. Phillips, B. Sadri, Lipschitz unimodal and isotonic regression on paths and trees, in: LATIN 2010, 2010, pp. 384–396.
[2] P.K. Agarwal, M. Sharir, Efficient algorithms for geometric optimization, ACM Comput. Surv. (1998) 412–458.
[3] R.K. Ahuja, J.B. Orlin, A fast scaling algorithm for minimizing separable convex functions subject to chain constraints, Oper. Res. 49 (2001) 784–789.
[4] S. Angelov, B. Harb, S. Kannan, L.-S. Wang, Weighted isotonic regression under the L_1 norm, in: Symp. Discrete Algorithms, 2006, pp. 783–791.
[5] R.E. Barlow, D.J. Bartholomew, J.M. Bremner, H.D. Brunk, Statistical Inference under Order Restrictions: The Theory and Application of Isotonic Regression, Wiley, 1972.
[6] R.E. Barlow, H.D. Brunk, The isotonic regression problem and its dual, J. Am. Stat. Soc. 67 (1972) 140–147.
[7] S. Benabbas, H.C. Lee, J. Oren, Y. Ye, Efficient sum-based hierarchical smoothing under ℓ_1 norm, arXiv:1108.1751, 2011.
[8] P. Berman, A. Bhattacharyya, E. Grigorescu, S. Raskhodnikova, D.P. Woodruff, G. Yaroslavtsev, Steiner transitive-closure spanners of low-dimensional posets, Combinatorica 34 (2014) 255–277.
[9] G. Bril, R. Dykstra, C. Pillars, T. Robertson, Algorithm AS 206: isotonic regression in two independent variables, J. R. Stat. Soc., Ser. C, Appl. Stat. 33 (1984) 352–357.
[10] M.R. Brown, R.E. Tarjan, A fast merging algorithm, J. Assoc. Comput. Mach. 26 (1979) 211–226.
[11] D. Chakrabarti, R. Kumar, K. Punera, Page-level template detection via isotonic smoothing, in: Proc. Int'l. World Wide Web Conf., 2007.
[12] D.Z. Chen, H. Wang, Approximating points by a piecewise linear function, Algorithmica 66 (2013) 682–713.
[13] J. Chun, K. Sadakane, T. Tokuyama, Linear time algorithm for approximating a curve by a single-peaked curve, Algorithmica 44 (2006) 103–115.
[14] R. Chandrasekaran, Y.U. Rhy, V.S. Jacob, S. Hong, Isotonic separation, INFORMS J. Comput. 17 (2005) 462–474.
[15] P.-A. Coquelin, R. Munos, Bandit algorithms for tree search, in: Proc. Conf. Uncertainty in AI, 2007, pp. 67–74.
[16] K. Dembczynski, S. Greco, W. Kotlowski, R. Slowinski, Statistical model for rough set approach to multicriteria classification, in: PKDD 2007: European Conf. Principles and Practice Knowledge Discovery in Databases, in: Lecture Notes in Computer Science, vol. 4702, Springer, 2007, pp. 164–175.
[17] S. Durham, N. Flournoy, W. Li, A sequential design for maximizing the probability of a favorable response, Can. J. Stat. 26 (1998) 479–495.
[18] R. Dykstra, J. Hewett, T. Robertson, Nonparametric, isotonic discriminant procedures, Biometrika 86 (1999) 429–438.
[19] H. Fournier, A. Vigneron, A deterministic algorithm for fitting a step function to a weighted point-set, Inf. Process. Lett. 113 (2013) 51–54.
[20] G.N. Frederickson, D.B. Johnson, Generalized selection and ranking: sorted matrices, SIAM J. Comput. 13 (1984) 14–30.
[21] F. Gebhardt, An algorithm for monotone regression with one or more independent variables, Biometrika 57 (1970) 263–271.
[22] A.J. Goldstein, J.B. Kruskal, Least-squares fitting by monotonic functions having integer values, J. Am. Stat. Assoc. 71 (1976) 370–373.
[23] N. Haiminen, A. Gionis, Unimodal segmentation of sequences, in: Proc. 4th IEEE Int'l. Conf. Data Mining, 2004, pp. 106–113.
[24] J. Hardwick, M. Meyer, Q.F. Stout, Directed walk designs for dose response problems with competing failure modes, Biometrics 59 (2003) 229–236.
[25] U. Herkenrath, The N-armed bandit with unimodal structure, Metrika 30 (1983) 195–210.
[26] J. Hershberger, S. Suri, Applications of a semi-dynamic convex hull algorithm, BIT 32 (1992) 249–267.
[27] S.M. Kakade, V. Kanade, O. Shamir, A. Kalai, Efficient learning of generalized linear and single index models with isotonic regression, in: NIPS 2011, 2011.
[28] A.T. Kalai, R. Sastry, The isotron algorithm: high-dimensional isotonic regression, in: Proc. Comp. Learning Theory (COLT), 2009.
[29] Y. Kaufman, A. Tamir, Locating service centers with precedence constraints, Discrete Appl. Math. 47 (1993) 251–261.
[30] D. Legg, D. Townsend, Best monotone approximation in $L_\infty[0, 1]$, J. Approx. Theory 42 (1984) 30–35.
[31] T.-C. Lin, C.-C. Kuo, Y.-H. Hsieh, B.-F. Wang, Efficient algorithms for the inverse sorting problem with bound constraints under the l_∞-norm and the Hamming distance, J. Comput. Syst. Sci. 75 (2009) 451–464.
[32] J.-Y. Liy, A randomized algorithm for weighted approximation of points by a step function, in: COCOA, 2010, pp. 300–308.
[33] M.-H. Liu, V.A. Ubhaya, Integer isotone optimization, SIAM J. Optim. 7 (1997) 1152–1159.
[34] A.K. Menon, X. Jiang, S. Vembu, C. Elkan, L. Ohno-Machado, Predicting accurate probabilities with a ranking loss, in: Proc. ICML, 2012.
[35] T. Moon, A. Smola, Y. Chang, Z. Zheng, IntervalRank — isotonic regression with listwise and pairwise constraints, in: Proc. Web Search and Data Mining, 2010, pp. 151–160.
[36] P.M. Pardalos, G. Xue, Algorithms for a class of isotonic regression problems, Algorithmica 23 (1999) 211–222.
[37] K. Punera, J. Ghosh, Enhanced hierarchical classification via isotonic smoothing, in: Proc. Int'l. Conf. World Wide Web, 2008, pp. 151–160.
[38] T. Robertson, F.T. Wright, R.L. Dykstra, Order Restricted Statistical Inference, Wiley, 1988.
[39] scikit-learn.org/stable/modules/isotonic.html.
[40] spark.apache.org/docs/latest/mllib-isotonic-regression.html.
[41] J. Spouge, H. Wan, W.J. Wilber, Least squares isotonic regression in two dimensions, J. Optim. Theory Appl. 117 (2003) 585–605.
[42] Q.F. Stout, Unimodal regression via prefix isotonic regression, Comput. Stat. Data Anal. 53 (2008) 289–297; A preliminary version appeared in: Optimal Algorithms for Unimodal Regression, in: Computing Science and Statistics, vol. 32, 2000.
[43] Q.F. Stout, Strict L_∞ isotonic regression, J. Optim. Theory Appl. 152 (2012) 121–135.
[44] Q.F. Stout, Isotonic regression via partitioning, Algorithmica 66 (2013) 93–112.
[45] Q.F. Stout, Isotonic regression for multiple independent variables, Algorithmica 71 (2015) 450–470.
[46] Q.F. Stout, L_∞ isotonic regression for linear, multidimensional, and tree orders, arXiv:1507.02226, 2015.
[47] W.A. Thompson Jr., The problem of negative estimates of variance components, Ann. Math. Stat. 33 (1962) 273–289.
[48] T.R. Turner, P.C. Wollan, Locating a maximum using isotonic regression, Comput. Stat. Data Anal. 25 (1997) 305–320.
[49] V.A. Ubhaya, Isotone optimization, I, II, J. Approx. Theory 12 (1974) 146–159, 315–331.
[50] M. Velikova, H. Daniels, Monotone Prediction Models in Data Mining, VDM Verlag, 2008.
[51] J.Y. Yu, S. Mannor, Unimodal bandits, in: Proc. Int'l. Conf. Machine Learn., 2011, pp. 41–48.