An Algorithm for L_∞ Approximation by Step Functions

Quentin F. Stout
Computer Science and Engineering
University of Michigan
Ann Arbor, MI 48109-2121
qstout@umich.edu +1 734.763.1518

Abstract

We give an algorithm for determining an optimal step function approximation of weighted data, where the error is measured with respect to the L_∞ norm. The algorithm takes $\Theta(n + \log n \cdot b(1 + \log n/b))$ time and $\Theta(n)$ space, where b is the number of steps. Thus the time is $\Theta(n \log n)$ in the worst case and $\Theta(n)$ when $b = O(n/\log n \log\log n)$. A minor change determines the optimal reduced isotonic regression in the same time and space bounds, and the algorithm also solves the k-center problem for 1-dimensional data.
Keywords: step function approximation; reduced isotonic regression; variable width histogram; k-center

1 Introduction

Step functions are a fundamental form of approximation, arising in variable width histograms, databases, segmentation, approximating sets of planar points, piecewise constant approximations, etc. Here we are interested in L_∞ stepwise approximation of weighted data. By weighted data $(\mathbf{y}, \mathbf{w})$ on $1 \ldots n$ we mean values (y_i, w_i), $1 \le i \le n$, where y_i is an arbitrary real number and w_i (the weight) is a nonnegative real number. For integers $i \le j$ let $[i : j]$ denote $i \ldots j$. A function f on $[1 : n]$ is a *b-step function* iff there are indices $j_1 = 1 < j_1 < \ldots < j_{b+1} = n + 1$ and real values C_k, $k \in [1 : b]$, such that $f_i = C_k$ for $i \in [j_k : j_{k+1} - 1]$. f is an *optimal L_∞ b-step approximation of* $(\mathbf{y}, \mathbf{w})$ iff it minimizes the weighted L_∞ error, $\max\{w_i \cdot |f_i - y_i| : i \in [1 : n]\}$, among all b-step functions. Since a step can be split into smaller ones, we do not differential between "b steps" and "no more than b steps".

Several authors have developed algorithms for L_∞ b-step regression [2, 3, 4, 5, 7, 8, 10, 11, 12, 13]. The fastest practical algorithm for weighted data takes $\Theta(\min\{n \log^2 n,\ n \log n + b^2 \log^4 n\})$ time and $\Theta(n \log n)$ space [2]. There is a $\Theta(n \log n)$ time algorithm [5, 10], but it is decidedly impractical. As Fournier and Vigneron noted [5], "it would be interesting to have a practical $O(n \log n)$-time deterministic algorithm". We present such an algorithm, and improve upon the time bound when $b = o(n)$. Further, it uses only $\Theta(n)$ space.

With a small change the algorithm also produces a "reduced isotonic" b-step function. f is an *isotonic* function iff $f_1 \le f_2 \le \ldots \le f_n$, and is an *optimal L_∞ b-step reduced isotonic regression of* $(\mathbf{y}, \mathbf{w})$ iff it minimizes the L_∞ error among all isotonic b-step functions. Isotonic regression is an important form of nonparametric regression that allows researchers to replace parametric assumptions with weaker shape constraints [1, 14]. Some researchers were concerned that it can overfit the data and/or be too complicated [9, 15, 16] and resorted to reduced isotonic regression. However, they used approximations because previous exact algorithms were too slow.

2 L_∞ b-Step Approximation

Algorithm A is similar to those in [2, 4] where a tree structure is used to determine the L_∞ error of an optimal step on a given interval. They also utilize a feasibility test, i.e., a decision procedure which is given ϵ and b

build an interval tree of bounded envelopes,
    utilizing feasibility tests during the construction
use search in a sorted matrix to determine minimum feasible error ϵ,
    exploiting the fact that evaluations at one stage are related to those in the previous stage
use ϵ to generate an optimal approximation

Algorithm A: Optimal L_∞ stepwise approximation or reduced isotonic regression

and decides if there is a b-step approximation with error no more than ϵ. If there is such an approximation then the test produces it. We incorporate important improvements to this basic approach. Feasibility tests are used to build the tree, not just during the search, and we exploit the fact that the values being determined at one stage of the search are related to those determined previously. We will show:

Theorem 1 *Given weighted data $(\mathbf{y}, \mathbf{w})$ and number of steps b, Algorithm A finds an optimal L_∞ b-step approximation, or an optimal L_∞ b-step reduced isotonic regression, in $\Theta(n + \log n \cdot b(1 + \log n/b))$ time and $\Theta(n)$ space.*

Given a set $(\mathbf{y}, \mathbf{w})$ of weighted values and $k \in [1:n]$, the *1-dimensional weighted k-center problem* is to find a set $S = \{s_1, \ldots, s_k\}$ of real numbers that minimizes $\max\{d(y_i, S) : i \in [1:n]\}$, where $d(\cdot, S)$ is the weighted distance to S, i.e., $d(y_i, S) = \min\{w_i \cdot |y_i - s_j| : j \in [1:k]\}$. This can be determined by finding an optimal k-step approximation of the values in sorted order, using the step values as the elements of S. Thus Algorithm A also solves the k-center problem for sorted weighted data in the time indicated.

From now on we generally omit mention of "L_∞" and "optimal" since they are implied. To simplify exposition we assume that n is an integral power of 2.

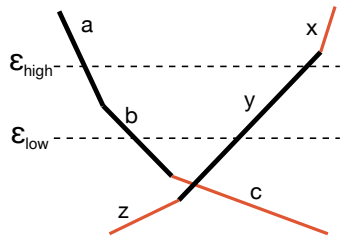
2.1 Tree of Bounded Envelopes

For a weighted value (y, w), the error of using $z \geq y$ as its regression value is given by the ray in the upper half-plane that starts at $(y, 0)$ with slope w. Given a set of weighted data $(\mathbf{y}, \mathbf{w})$, its *upward error envelope* is the topmost sequence of line segments corresponding to all such rays. For each z, it gives the maximum error of using z as the regression value for all points (y_i, w_i) where $z \geq y_i$. The *downward error envelope* uses rays in the upper half-plane starting at $(y_i, 0)$ with slope $-w_i$, representing the error of using a regression value $\leq y_i$. The intersection of the downward and upward error envelopes gives the regression value minimizing the error over the entire set, i.e., the weighted L_∞ mean, and its error.

An *interval tree* has a root which corresponds to the interval $[1:n]$, its children correspond to $[1:n/2]$ and $[n/2+1:n]$, their children represent intervals of length $n/4$, etc. The leaves are the intervals of length 1, i.e., 1, 2, ... n. The intervals represented in the tree will be called *basic intervals*.

Several authors [2, 4, 8] used an interval tree where each node contains the upward and downward error envelopes of the data in its interval, but most of the rays are unnecessary. Let ϵ^* be the (unknown) error of an optimal b-piece approximation, and let $\epsilon_{\text{low}} < \epsilon^* \leq \epsilon_{\text{high}}$. Then ϵ^* can be determined using only the rays representing errors in $(\epsilon_{\text{low}}, \epsilon_{\text{high}})$. All others are discarded, and the remaining ones are a *bounded envelope*. See Figure 1. In our interval tree each node contains its upward and downward bounded envelopes.

Given a set of intervals that have been merged into a single step, and given $\epsilon \in (\epsilon_{\text{low}}, \epsilon_{\text{high}}]$, one can quickly determine the error of the step relative to ϵ. Let U be the union of the segments in the intervals' upward bounded envelopes and D the union of the segments in their downward bounded envelopes. For ray



a, b, and y are in bounded envelopes; c, x and z are discarded
Since $\mathsf{b}^{-1}(\epsilon_{\text{low}}) < \mathsf{y}^{-1}(\epsilon_{\text{low}})$, interval error $< \epsilon_{\text{low}}$

Figure 1: Downward and upward bounded envelopes

r let $r^{-1}(y)$ denote the x such that r has error y at x. Let $u = \min\{r^{-1}(\epsilon) : r \in U\}$ and $d = \max\{r^{-1}(\epsilon) : r \in D\}$. u and d are the points with error ϵ on the upward and downward, respectively, envelopes of the union of the intervals. Then the step has error $<$, $=$, or $> \epsilon$ if $u > d$, $u = d$, or $u < d$, respectively. This *bounding test* can be decided in time linear in the total number of rays in the intervals' bounded envelopes.

The bounded envelopes are stored as a doubly-linked list in order of slope. Whenever a node is visited, rays with segments outside $(\epsilon_{\text{low}}, \epsilon_{\text{high}})$ are discarded. The time to perform this is charged to the rays, not the search, and the time is linear in the number of rays discarded. Whenever the number of remaining rays is counted, the count is only of those that would not be discarded given the current $(\epsilon_{\text{low}}, \epsilon_{\text{high}})$.

2.2 Feasibility Tests

An arbitrary interval $[i : j]$ can be decomposed into $\Theta(\log(j - i + 1))$ basic intervals where the sizes increase at each step and then decrease, with perhaps two intervals of the same size in the middle. E.g., $[2 : 13] = [2 : 2] \cup [3 : 4] \cup [5 : 8] \cup [9 : 12] \cup [13 : 13]$. These can be generated in $\Theta(\log n)$ time by a tree traversal starting at i, moving upward to the least common ancestor of i and j, and then downward to j. Suppose, given i and $\epsilon \in (\epsilon_{\text{low}}, \epsilon_{\text{high}})$, we want to determine the maximum j such that the error of making $[i : j]$ a single step is $\leq \epsilon$. We do this by locating $j + 1$. By incrementally updating u and d used in the bounding test, on the upward pass at each node p one can determine if $j + 1$ is less than or equal to the largest value in p's subtree by using the bounded envelopes in p's right subtree to decide if adding the right tree gives an error $> \epsilon$. On the downward pass one can decide if $j + 1$ is in p's right subtree by deciding if adding the left subtree gives an error $> \epsilon$. Not counting the queries of children's envelopes, the nodes visited are the same as those in going from i to $j + 1$ when $j + 1$ is known.

Given b and $\epsilon \in (\epsilon_{\text{low}}, \epsilon_{\text{high}})$, a *feasibility test* determines if there is a b-step function with regression error $\leq \epsilon$. This can be accomplished by starting at 1 and determining the largest j_1 for which the error of making $[i : j_1]$ a single step is $\leq \epsilon$, then starting at $j_1 + 1$ and determining the largest j_2 for which the error of making $[j_1 + 1 : j_2]$ a single step is $\leq \epsilon$, etc. If the b^{th} step is finished before n is reached then ϵ is infeasible and the test stops. Otherwise, ϵ is feasible and the steps have been identified.

A feasibility test for b-step reduced isotonic regression is essentially the same. The regression value used for the first step is the smallest possible with error $\leq \epsilon$, i.e., d. Denoting this as d_1, in the search for the second step its d value is initialized to d_1, not $-\infty$, and then the search continues as before, always using the final value of d_k to initialize d_{k+1}. For both tests the time is linear in the number of rays encountered.

Let W be the set of nodes visited in a feasibility test. The test visits each node of W at most twice, once on an upward phase and once on a downward phase. Thus at any level at most b nodes are visited. The top

$\lfloor \lg b \rfloor$ levels have a total of $\Theta(b)$ nodes. There are $\lceil \lg n \rceil - \lfloor \lg b \rfloor = \Theta(\log n/b)$ levels below this, so in total $\Theta(b(1 + \log n/b))$ nodes are visited. Feasibility tests during tree construction have a slight change in that when level k is being constructed, at level k the search goes sideways, not upwards, from one node to the next at the same level. Thus the number of nodes visited increases by at most $n/2^k$.

2.3 Constructing the Tree

A straightforward generation of bounded envelopes first constructs standard error envelopes with $\Theta(n \log n)$ total segments and then eliminates rays. To reduce the space and time we interleave constructing the interval tree of bounded envelopes with continually narrowing the gap between ϵ_{low} and ϵ_{high}. At the end of the construction, $(\epsilon_{\text{low}}, \epsilon_{\text{high}}]$ will be so small that each bounded envelope is a single ray.

First a feasibility test with $\epsilon = 0$ is performed using only the base level. If it passes then the algorithm is done. Otherwise, set $\epsilon_{\text{low}} = 0$, $\epsilon_{\text{high}} = \infty$. At level 0 each interval is a singleton, with single rays in its upward and downward envelopes for a total of $2n$ rays. At the next level envelopes from below are merged, forming $\leq 2n$ segments (some rays may be completely covered and hence immediately discarded), and there are $\leq n$ segment endpoints (e.g., the endpoint of a and b in Fig. 1). Put the errors of these endpoints in a multiset R and move up to level 2. Throughout, R is an unordered multiset of endpoint errors in $(\epsilon_{\text{low}}, \epsilon_{\text{high}})$, where $|R|$ is the total number of rays in envelopes in all of the envelopes created so far, minus the 1 per envelope required and minus those that will be discarded when their node is visited. For any feasibility test the time is at worst linear in the time it would take if there were only 1 ray per envelope (analyzed in Section 2.2) plus the size of R.

To describe the procedure for level k, let $m = n/2^k$. At the start $|R| \leq 4m$. There are $4m$ envelopes at level $k-1$, each requiring one ray, and all of the entries in R might correspond to additional rays at that level, so when the envelopes are merged to form the $2m$ envelopes at level k there may be $8m - 2m = 6m$ segment endpoints. Add those endpoint errors which are in $(\epsilon_{\text{low}}, \epsilon_{\text{high}})$ to R, which may now have size $10m$. Take the median error in R and do a feasibility test of it. Depending on the outcome, one of ϵ_{low} and ϵ_{high} is adjusted, and at least 1/2 the entries in R can be eliminated. Doing this 3 times results in $|R| < 2m$, completing the procedure for level k.

When the top is reached $|R| \leq 4$ and by using feasibility tests all the endpoint errors can be eliminated, i.e., $(\epsilon_{\text{low}}, \epsilon_{\text{high}})$ has been narrowed so that at every node of the interval tree the upward and downward bounded envelopes have only one ray. Go through the tree and remove all rays with segments outside $(\epsilon_{\text{low}}, \epsilon_{\text{high}})$, taking $\Theta(n)$ time. This completes the tree construction. There are $\Theta(\log n)$ feasibility tests, each involving time added by moving sideways at level k, rather than up and down in the tree. The total sideways time is $\Theta(n)$, so the time to construct the tree is $\Theta(n + \log n \cdot b(1 + \log n/b))$.

2.4 Search for Minimal Feasible Error

The L_∞ error of a stepwise approximation is the maximum of the L_∞ errors of its steps. Thus there is an interval $[i\!:\!j]$ such that the error of an optimal b-step approximation is the error of using the weighted L_∞ mean as the step value on $[i\!:\!j]$. A search on such errors, coupled with a feasibility test, can find the minimal feasible error. "Parametric search" was used in [5, 10] but this is only of theoretical interest since parametric search is completely impractical, involving very complex data structures and quite large constants.

Search in a sorted matrix provides a practical approach (see [6]). Let E be the $n \times n$ matrix where $E(i, j)$ is the error of using the L_∞ mean on $[i\!:\!j]$ if $i \leq j$, and is 0 if $i > j$. E is not actually created, but rather serves as a conceptual guide. Its rows are nondecreasing and the columns are nonincreasing, so for

any submatrix its largest entry is in the upper right and the smallest is in the lower left. The algorithm has stages $0 \ldots \lg n - 1$, where at the start of stage i there is a collection of disjoint square submatrices of size $n/2^i$. Note that the minimal feasible error is one of the entries of E.

Stage 0 starts with all of E. At each stage, divide all of the matrices into quadrants, and let ϵ_1 be a median of the lower left entries of the quadrants, i.e., a median of their smallest values, and let ϵ_2 be a median of the upper right entries. If ϵ_1 is feasible then any quadrant with smallest value $\geq \epsilon_1$ is eliminated and $\epsilon_{\text{high}} = \min\{\epsilon_{\text{high}}, \epsilon_1\}$, while if ϵ_1 is not feasible then any quadrant with largest value $\leq \epsilon_1$ is eliminated and $\epsilon_{\text{low}} = \max\{\epsilon_{\text{low}}, \epsilon_1\}$. Similar eliminations are done based on the feasibility of ϵ_2. The remaining quadrants are the matrices that start of the next stage. After the last stage, when the remaining matrices are 1×1, a standard binary search on these values is used to locate the minimal feasible error.

This search uses $\Theta(\log n)$ feasibility tests, and, as proven in [6], only $\Theta(n)$ entries of E are evaluated.

2.5 Evaluating E

For intervals $I, J \subseteq [1\!:\!n]$ let $E(I, J)$ denote the submatrix $\{E(i, j) : i \in I, j \in J\}$, i.e., the submatrix of all intervals starting at some $i \in I$ and ending at some $j \in J$. Search in a sorted matrix has the property that at the start of stage s there is a collection of submatrices of the form $E(I, J)$ for basic intervals I, J of size $n/2^s$. Either $I = J$, or I is to the left of J and there is a (perhaps empty) interval K between them with length an integral multiple of $n/2^s$. During stage s, I and J are cut in half into I_1, I_2 and J_1, J_2, respectively, and the smallest and largest values in $E(I_1, J_1)$, $E(I_1, J_2)$, $E(I_2, J_1)$, and $E(I_2, J_2)$ are determined. Let i'_1 denote the smallest index in I_1 and i''_1 its largest index, and let j'_1, j''_1 be the smallest and largest, respectively, indices in J_1. Then the smallest value in $E(I_1, J_1)$, i.e., $E(i''_1, j'_1)$, is the error of making the interval $i''_1 \cup I_2 K \cup j'_1$ a single step, and the largest value in $E(I_1, J_1)$, $E(i'_1, j''_1)$, is the error of the interval $I_1 I_2 K J_1$. Similarly the smallest and largest values in each of $E(I_1, J_2)$, $E(I_2, J_1)$ and $E(I_2, J_2)$ are the errors of intervals which are the union of consecutive elements of $\{I_1, I_2, K, J_1, J_2\}$, with perhaps additional singleton indices at the start or end. The upward and downward envelopes for $E(i''_1, j'_1)$ have single rays for I_2, i''_1, and j'_1, so the time to determine their intersection, or decide it is outside of $(e_{\text{low}}, e_{\text{high}})$, is linear in a constant plus the size of the bounded envelopes for K. Similar results hold for all of the other values needed at this stage.

Associate the bounded envelopes for K with $E(I, J)$, and if, say, $E(I_1, J_1)$ is kept for stage $s + 1$ then the envelopes for $I_2 K$ are associated with it, and similarly for each of the other children of $E(I, J)$. Just as for the upward construction of the tree, as the search in a sorted matrix is proceeding top-down, in addition to the feasibility tests for the basic search we reduce the number of rays by interleaving tests based on the endpoints of the segments. Each bounded envelope can be copied 4 times, and each child may add up to 2 rays, so by using 2 additional tests at each step the total number of rays in the bounded envelopes at stage s is $\Theta(2^s)$. Since the time to evaluate an entry of E is linear in the number of rays involved, the total time for the evaluation of entries of E over all steps of the algorithm is $\Theta(n)$, and the total time for the feasibility tests is $\Theta(\log n \cdot b(1 + \log n/b))$.

3 Final Comments

Step function approximation arises in a variety of guises and contexts, as does isotonic regression (see the myriad citations to [1, 14]). For weighted data, Algorithm A finds an L_∞ b-step approximation, an L_∞ b-step reduced isotonic regression, or solves the 1-dimensional k-center problem for sorted data, in $\Theta(n + \log n \cdot b(1 + \log n/b))$ time. Previous algorithms had slower worst-case time [2, 3, 4, 7, 8, 11, 12, 13]

or, as their authors noted, were highly impractical [5, 10] (and these too were slower when $b = o(n)$). Further, they all used $\Omega(n \log n)$ space, while Algorithm A uses only $\Theta(n)$. The algorithm even improves upon the previous fastest algorithm for unweighted data, which takes $\Theta(n + b^2 \log^3 n)$ time [8].

Acknowledgement: Research partially supported by NSF grant CDI-1027192

References

[1] Barlow, RE, Bartholomew, DJ, Bremner, JM, and Brunk, HD, *Statistical Inference Under Order Restrictions: The Theory and Application of Isotonic Regression*, John Wiley, 1972.

[2] Chen, DZ and Wang, H, "Approximating points by a piecewise linear function", *Algorithmica* 66 (2013), pp. 682–713.

[3] Díaz-Báñnez, JM and Mesa, JA, "Fitting rectilinear polygonal curves to a set of points in the plane", *Eur. J. Operational Res.* 130 (2001), pp. 214–222.

[4] Fournier, H and Vigneron, A, "Fitting a step function to a point set", *Algor.* 60 (2011), pp. 95–109.

[5] Fournier, H and Vigneron, A, "A deterministic algorithm for fitting a step function to a weighted point-set", *Info. Proc. Let.* 113 (2013), pp. 51–54.

[6] Frederickson, G and Johnson, D, "Generalized selection and ranking: Sorted matrices", *SIAM J. Comp.* 13 (1984), pp. 14–30.

[7] Fülöp, J and Prill, M, "On the minimax approximation in the class of the univariate piecewise constant functions", *Oper. Res. Let.* 12 (1992), pp. 307–312.

[8] Guha, S and Shim, K, "A note on linear time algorithms for maximum error histograms", *IEEE Trans. Knowledge and Data Engin.* 19 (2007), pp. 993–997.

[9] Haiminen, N, Gionis, A, and Laasonen, K, "Algorithms for unimodal segmentation with applications to unimodality detection", *Knowl. Info. Sys.* 14 (2008), pp. 39–57.

[10] Hardwick, J and Stout, QF, "Optimal reduced isotonic reduction", *Proc. Interface 2012*.

[11] Karras, P, Sacharidis, D., and Mamoulis, N, "Exploiting duality in summarization with deterministic guarantees", *Proc. Int'l. Conf. Knowledge Discovery and Data Mining* (KDD) (2007), pp. 380–389.

[12] Liu, J-Y, "A randomized algorithm for weighted approximation of points by a step function", *COCOA* 1 (2010), pp. 300–308.

[13] Mayster, Y and Lopez, MA, "Approximating a set of points by a step function", *J. Vis. Commun. Image R.* 17 (2006), pp. 1178–1189.

[14] Robertson, T, Wright, FT, and Dykstra, RL, *Order Restricted Statistical Inference*, Wiley, 1988.

[15] Salanti, G and Ulm, K, "A nonparametric changepoint model for stratifying continuous variables under order restrictions and binary outcome", *Stat. Methods Med. Res.* 12 (2003), pp. 351–367.

[16] Schell, MJ and Singh, B, "The reduced monotonic regression method", *J. Amer. Stat. Assoc.* 92 (1997), pp. 128–135.