75-02

On the Calculation of the Piecewise Linear Approximation to a Discrete Function

JOOS VANDEWALLE

Abstract—A repeated scanning algorithm for determining the minimax approximation to an arbitrary function by means of a piecewise linear function with a fixed number of variable knots is described, and it is compared to optimal and suboptimal methods.

Index Terms—Discrete data, feature generation, minimax approximation, piecewise linear function, waveform segmentation.

Manuscript received November 9, 1973; revised November 2, 1974.
The author is with the Department of Electrical Engineering, Catholic University of Louvain, Heverlee, Belgium.

I. INTRODUCTION

The problem of the approximation to functions by piecewise linear functions (discontinuous at the knots) and spline functions (continuous at the knots) appears in a number of fields, e.g., encoding of information (speech), function generation for hybrid computation, and feature generation for pattern recognition.

In this correspondence it is our purpose to approximate a discrete function of one variable by a piecewise linear function with n segments and with $n-1$ variable knots so that the maximum difference between the approxmation and the function is minimal (L_∞ or minimax norm).

After assembling some basic facts, we describe in Algorithm 1 an elementary scan, which is used repeatedly in Algorithm 2 to calculate the global minimax error.

Depending on the guess for the global minimax error, Algorithm 1 allows us to derive computationally interesting properties for the global minimax error (bounds or its exact value) which are then used repeatedly in Algorithm 2 to make better guesses. Some programming notes are included.

Finally the results of this optimal method are compared to those of descent methods (e.g., Pavlidis [6]) and of dynamic programming (Bellman [1]).

The method of Lawson [5, pp. 8–9] is not applicable since it assumes that the minimax error in one segment is proportional to the square of the length of the segment, which is no longer valid for functions with an inflection point. For the same reason and because of the discreteness of the data the methods of Phillips [7] and Cox [2] are not applicable.

We will use the following notations:

t_i, f_i $(i = 1,2,\cdots N)$	discrete data set with $t_1 < t_2 < t_3 \cdots < t_N$.
n	number of segments of the piecewise linear approximation.
$\bar{u}$	a vector with as component j $(j = 0,(1)n)$ the endpoint u_j of the segment j. These are called the knots or the breakpoints of the piecewise linear approximation.
U	the set of allowable values for $\bar{u} = \{\bar{u} \mid t_1 = u_0 < u_1 < u_2 \cdots u_n = t_N\}$.
e_j	the minimax error of the approximation of the data in the jth segment with a straight line. e_j is a function of u_{j-1} and u_j.
$f(\bar{u})$	maximal error of all the segments: max (e_j) with $\bar{u} \in U$, $j = 1(1)n$.
f^*	min $f(\bar{u})$, the global minimax error, $\bar{u} \in U$.
U^*	the set of optimal knot positions: $\{\bar{u} \mid \bar{u} \in U, f(\bar{u}) = f^*\}$.

II. PRELIMINARIES

In this section some facts are assembled upon which the algorithm will be based.

1) The best approximation to the data with a straight line in a segment is unique and can be easily determined by the exchange method [8] and the error norm is a nondecreasing (nonincreasing) function of the endpoint (beginpoint) of the segment [6].

2) As every u_j $(j = 1\ (1)n-1)$ is situated in a half-open interval of two consecutive t_i called $[t_{m_j}, t_{m_j+1})$ and the value of e_j is the same for all this u_j, we can assume that the u_j coincide with the $t_{m_j}: u_j = t_{m_j}$ with $m_0 = 0$ and $m_n = N$. Then t_{m_j+1} is the first data point of the $(j+1)$th segment.

3) A theorem about the set of optimal values of the knots U^*, (see Lawson [4], [5]): For any $\bar{u} \in U$ and $\bar{u}^* \in U^*$ there is a path P in U connecting $\bar{u}$ to $\bar{u}^*$ having the property that $f(\bar{u})$ is nonincreasing as $\bar{u}$ moves along P from u to $\bar{u}^*$. Thus all $\bar{u}^*$ are connected with a path belonging to U^*, and will thus have knot positions that are not very much different from each other, so that we are only interested in "an" approximation with error f^*.

III. ALGORITHM 1

This algorithm calculates an approximation of the data in $[t_1, t_N]$ with n' segments $(n' \leq n)$ and with knots $t_{m'_j}$. In every segment (except eventually the nth) the minimax error e_j has to be lower than or equal to a prescribed value: $e (e \geq 0)$ and in addition $t_{m'_{n'-1}}$ has to be maximal.

Step 1: Start with the first segment $(j = 1)$ and with the first data point $(t_1 = t_{m'_{0+1}})$.

Step 2: Search for the greatest t_{m_j} so that the minimax error in the jth segment does not exceed e. This point $t_{m'_j}$ is unique [this ensues from Section II-1)] and can be determined by repeatedly splitting between an upper bound with error $e_j^u (e_j^u > e)$ and a lower bound with error $e_j^l (e_j^l \leq e)$. There is always a lower bound for $t_{m'_{j-1}+1}$. If there are not enough data to obtain an upper bound, set $t_{m'_j} = t_N$ and go to Step 5. In other words we reduce the difference between these bounds by repeatedly splitting until we have obtained two consecutive points and then the lower bound is $t_{m'_j}$. The convergence can be accelerated by increasing the lower bound or decreasing the upper bound in every split-step with data points that cannot alter the respective minimax errors.

Step 3: If $t_{m'_j} \neq t_N$ and $j < n-1$, go to Step 2.

Step 4: If $j = n-1$, compute $d' = e(t_{m'_{n-1}+1}, t_N)$ and

$$e' = \max e_j(t_{m_{j-1}+1}, t_{m_j}), \qquad j = 1\ (1)n-1$$

and

$$e'' = \min e_j^u - \delta, \qquad j = 1\ (1)n-1$$

(with δ a small positive number compared to e'') and stop.

Step 5: Call $n' = j$ the number of segments and compute

$$e' = \max e_j(t_{m_{j-1}+1}, t_{m_j}), \qquad j = 1\ (1)n'-1$$

and stop.

If the algorithm stopped at Step 4, it obtained a value $e'' > e$, in so far that, if the Algorithm 1 is restarted with this value as prescribed maximum error, the approximation remains the same. In function of the value chosen for e the algorithm stopped at Steps 4 or 5 and obtained some e' and/or d', e''; all this will enable us to determine upper and lower bounds for f^*.

Properties:

a) if $n' < n$, then $e \geq e' > f^*$; (1)

b) if $n' = n$ and $e' \geq d'$, then $e'' > e \geq e' \geq f^* \geq d'$; (2)

c) if $n' = n$ and $e'' \geq d' \geq e'$, then $e'' \geq d' = f^* \geq e'$; (3)

d) if $n' = n$ and $d' \geq e'' > e \geq e'$,

then $d' \geq f^* \geq e'' > e \geq e'$. (4)

It is obvious that by increasing e, $t_{m'_{n-1}}$ (if it exists) increases and the algorithm evolves from situation d) to c), b), and a).

The proof to (1)–(4) is easy. Supposing it were not true, then it is in contradiction to the minimality of f^* or to the fact that $t_{m'_{n-1}}$ is a nondecreasing function of e.

IV. ALGORITHM 2

This algorithm calculates an approximation with n segments and with minimax error lower than $f^*(1+\epsilon)$, where ϵ is the desired

TABLE I

FLOW CHART OF THE ALGORITHM FOR MINIMAX APPROXIMATION WITH n SEGMENTS

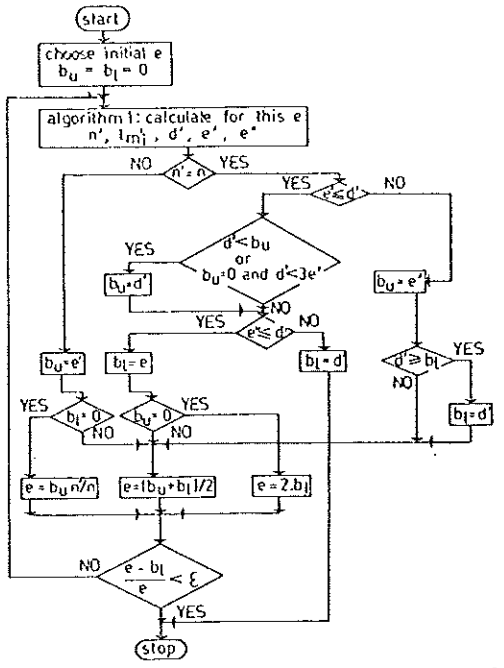


TABLE II

COMPARISON OF THE RESULTS OF 8 TESTCASES FOR METHODS 1)–4)

kind of data	N	n	range of the data	initial maximal errors		maximal errors			number of line approximations				execution time (s)		
				1,2	3	1	2	3	1	2	3	4	1	2	3
I	308	33	900	81.2	90.6	27.5	27.5	14.6	122	295	985	919,512	0.25	0.54	1.63
I	308	21	900	81.2	18.2	35.8	35.0	17.5	77	345	347	680,316	0.22	0.74	0.79
I	308	15	900	55.0	27.5	31.8	24.1	22.4	51	165	186	508,338	0.16	0.57	0.62
I	308	12	900	89.7	27.5	35.8	27.1	24.1	38	405	253	408,120	0.18	1.25	0.84
I	308	8	900	90.0	40.0	50.0	44.8	36.8	27	108	258	259,012	0.13	0.58	1.33
II	500	5	153.5	28.5	8.1	28.5	4.9	4.9	11	120	125	363,340	0.27	1.47	1.75
III	500	7	71.3	9.7	9.2	7.0	6.0	5.0	14	154	189	595,114	0.23	1.81	1.76
IV	500	40	97.7	8.5	5.0	6.9	5.7	5.0	292	369	419	3,376,420	0.74	0.87	2.24

relative precision on f^* (see flow chart on Table I). Using (1)–(4) we devised the flow chart so that Algorithm 1 is first used to obtain an upper bound (b_u) and a lower bound (b_l) for the unknown f^*. Then by taking iteratively $e = (b_u + b_l)/2$ in a new call to Algorithm 1, the interval for f^* is at least halved (in most of the cases we can reduce $b_u - b_l$ even more). Notice that we do not accept d' as upper bound if $d' \geq 3e'$ and $n' = n$ because this upper bound would be too great. To calculate the initial value of e we use the following procedure: split the set of t_i ($i = 1(1)N$) in n intervals of equal data points and calculate the minimax error in every interval; then the initial value of e is the mean minimax error of the intervals divided by 2. The algorithm always finds a finite upper and lower bound and reduces the interval between these bounds at least by a half, and converges thus at least linearly.

V. COMPARISON OF THE METHODS

We used four test functions: an electrocardiogram (Data I), a piecewise linear function with homogeneous noise (Data II), a hyperbolic function with noise (Data III), and $0.1 \times I \times \sin(I/4)$ $I = 1(1)500$ (Data IV).

The measured parameters in Table II are the range of the data,

the initial and final L_∞ error norms, the number of line approximations, and the execution times without input–output operations. We compared the results of four methods.

1) The approximation obtained by the algorithm of Pavlidis[6] while keeping track of which intervals changed at each iteration as described in [6].

2) Same as 1) but with the modification that in the flow chart of Table 1 in [6, p. 691] we add an equal sign in the condition $\max(d_j, d_{j+1}) < \max(e_j, e_{j+1})$ so that the algorithm does not necessarily stop when the error is not diminishing locally (care must be taken if cycling occurs by imposing a maximal number of iteration steps). In the first two methods the initial segmentation of the function is so that the integral squared error (ISE) is equal in every segment.

3) The results of Algorithm 2 with $\epsilon = 10^{-4}$.

4) The results of the dynamic programming method as described by Bellman [1] but adapted for minimax approximations.

All subroutines (except 4) are programmed on IBM 370/158 in Fortan 4 and are implemented through a Fortran H complier.

A complete listing of the program and an exposition of the meaning of the variables is given in [9] (method 3).

Table II allows us to draw some conclusions.

Comparing methods 1)–4), and using the same testcase we see that the maximal errors reduce while the number of line approximations and the execution times increase. The methods 1)–3) are thus complementary and allow to make still better approximations for more execution time. Methods 1) and 2) are suboptimal and can stop when the gradient is zero, without reaching the absolute minimum and it is difficult to get and idea how far it stops from the absolute minimum. Method 3, on the other hand, gives bounds on the error and will reach the minimum.

Most of the time in the execution of every algorithm is spent on line approximations. In Table II upper bounds for the number of line approximations with dynamic programming are given [3], and these can still be reduced [3]. Nevertheless a decrease of a factor thousand will be difficult to obtain so that not much attention is paid to this method. It is hard to express the execution time or the number of line approximations of method 3) in function of the number of data points, the number of line segments and the range of the data. The reason for this is that the increase in iterations (linear scans) has sometimes more influence on the execution time than the increase in data points or in line segments.

ACKNOWLEDGMENT

The author wishes to thank reviewer A of this correspondence for his valuable comments and suggestions during its preparation.

REFERENCES

[1] R. Bellman, "On the approximation of curves by line segments using dynamic programming," *Commun. Ass. Comput. Mach.*, vol. 4, p. 284, June 1961.
[2] M. G. Cox, "An algorithm for approximating convex functions by means of first degree splines," *Comput. J.*, vol. 14, pp. 272–275, Aug. 1971.
[3] ——, "Curve fitting with piecewise polynomials," *J. Inst. Math. Its Appl.*, vol. 8, pp. 36–52, 1971.
[4] C. L. Lawson, "Characteristic properties of the segmented rational minimax approximation problem," *Numer. Math.*, vol. 6, pp. 293–301, 1964.
[5] ——, "Segmented rational minmax approximation, characteristic properties and computional methods," Jet Propulsion Lab., Calif. Inst. of Tech., Pasadena, Tech. Rep. 32-579, Dec. 1963.
[6] T. Pavlidis, "Waveform segmentation through functional approximation," *IEEE Trans. Comput.*, vol. C-22, pp. 689–697, July 1973.
[7] G. M. Phillips, "Algorithms for piecewise straight line approximations," *Comput. J.*, vol. 11, pp. 211–212, 1968.
[8] M. J. D. Powell, "The exchange algorithm for calculating minimax linear approximations over a discrete point set," in *Methods of Numerical Approximation*, D. C. Handscomb, Ed. Oxford, England: Pergamon Press, 1966, pp. 73–81.
[9] J. Vandewalle, "Subroutine for the piecewise-linear minimax approximation to a discrete function by a scanning technique," Elektrotech. Inst., Kardinaal Mercierlaan 94, Heverlee, Belgium, Tech. Rep., Oct. 1974.