# An interactive polynomial approximation algorithm

Karl-Heinz Mohn* and Roger V. Roman†

**The Remez algorithm for Chebyshev approximation of real continuous functions is adapted to an interactive environment. With the aid of a graphic display terminal the user is directly involved in each iteration, contributing global human logic and controlling the extent of iterative refinement. The underlying mathematics, an implementation description, and several illustrative examples are presented.**
(Received May 1974)

The improvements in the economics of interactive access to large-scale computing systems and the advancements in the sophistication of interactive terminal devices in recent years have combined to provide a rich environment for numerical mathematical problem solving. Initially several general purpose conversational systems were developed which were oriented toward conventional typewriter-like terminals. Some of these systems have been extended to utilise the additional power of graphical terminal devices. For a survey of interactive graphical systems for mathematics see Smith (1970).

In this paper we describe the implementation of an interactive graphical algorithm for Chebyshev approximation of continuous functions by polynomials. The implementation of this algorithm addresses two main goals: first, the user should be able to formulate his problem in 'normal' mathematical notation and secondly, he should be included in the portions of the problem solution process where his human mental capabilities are more efficient than automated logic. In the first section of this paper we describe the mathematical background of the underlying procedure; in the second section the implementation environment and organisation are described; and finally we present some examples to illustrate the use of the resulting system.

**1. Mathematical formulation** (following Ehlich (1967))

Given a continuous function $f(x)$ over a closed real interval $[a, b]$ the objective of minimax polynomial approximation is to find a polynomial $P_n(x)$ of degree $n$ such that

$$E^* = \max |E(x)| = \max |f(x) - P_n(x)|, \; a \leqslant x \leqslant b$$

is minimal. It is shown in Meinardus (1967) that $P_n(x)$ is the best polynomial approximation to $f(x)$ on $[a, b]$ if and only if there exists a set of $n + 2$ points $\{x_i\}$ where

$$a \leqslant x_0 < x_1 < \ldots < x_{n+1} \leqslant b$$

such that

$$E(x_{i+1}) = -E(x_i)$$

and

$$\max |E(x)| = E^* = |E(x_i)|, \; i = 0, \ldots, n + 1$$

The objective of the Remez algorithm is to iteratively determine these $n + 2$ points at which the error function $E(x)$ takes its maximum value with alternating sign. In the following paragraphs we describe how this is accomplished by means of interpolation.

Initially we choose a set $\{x_i\}^\nu$ of any $n + 2$ points in the interval $[a, b]$ and determine the interpolation polynomial $Q^\nu_{n+1}(x)$ of degree $\leqslant n + 1$ such that:

$$Q^\nu_{n+1}(x_i) = f(x_i), \; i = 0, \ldots, n + 1$$

and a polynomial $R^\nu_{n+1}(x)$ of exact degree $n + 1$ such that

$$R^\nu_{n+1}(x_i) = (-1)^i, \; i = 0, \ldots, n + 1 \; .$$

Let $a^\nu_{n+1}$ and $b^\nu_{n+1}$ be the high order coefficients of $Q^\nu_{n+1}(x)$ and $R^\nu_{n+1}(x)$ respectively. If we then choose

$$L^\nu = \frac{a^\nu_{n+1}}{b^\nu_{n+1}} \tag{1.1}$$

the polynomial

$$P^\nu_n(x) = Q^\nu_{n+1}(x) - L^\nu R^\nu_{n+1}(x) \tag{1.2}$$

is of degree $\leqslant n$ and

$$f(x_i) - P^\nu_n(x_i) = \eta(-1)^i L^\nu, \; i = 0, \ldots, n + 1, \eta = \pm 1 \; .$$

This implies that we have already found a polynomial of degree $\leqslant n$ for which the error function $E(x)$ assumes a unique value $L$ on the $n + 2$ points $\{x_i\}^\nu$ with alternating sign. According to the theorem of de la Vallee Poussin,

$$E^* \geqslant |L^\nu|$$

holds. Consequently there are two possibilities:

(*a*) $$E^* = |L^\nu| \; ,$$

in which case we have found the best approximation polynomial which is $P^\nu_n(x)$; or

(*b*) $$E^* > |L^\nu|$$

in which case we exchange the set $\{x_i\}^\nu$ for a set $\{x_i\}^{\nu+1}$ such that

$$f(x_i^{\nu+1}) - P^\nu_n(x_i^{\nu+1}) = \eta(-1)^i \xi_i^{\nu+1}$$

with $\eta = \pm 1$ and $\xi_i^{\nu+1} \geqslant |L^\nu|$ and for at least one $x_j^{\nu+1}$

$$|f(x_j^{\nu+1}) - P^\nu_n(x_j^{\nu+1})| = \|f(x) - P^\nu_n(x)\| \; .$$

The exchange is done in the following manner. Let $y$ be a member of $\{x_i\}^{\nu+1}$ bit not a member of $\{x_i\}^\nu$.

Then if

(*a*) $y - x_0^\nu$ and

(i) $\operatorname{sign}(f - P^\nu_n)(x_0^\nu) = \operatorname{sign}(f - P^\nu_n)(y)$ then $x_0^{\nu+1} = y$ and $x_i^{\nu+1} = x_i^\nu$, for $i = 1, \ldots, n + 1$ or

(ii) $\operatorname{sign}(f - P^\nu_n)(x_0^\nu) = -\operatorname{sign}(f - P^\nu_n)(y)$ then $x_0^{\nu+1} = y$

This work was supported in part by NSF Contract GP-05850.
*Ruhr University Bochum, Rechenzentrum, 463 Bochum-Querenburg, Universitätsstrasse 150, West-Germany. While visiting Purdue University Computer Sciences Department, 1971-1972.
†Purdue University, Lafayette, Indiana, USA.

and $x_{i+1}^{\nu+1} = x_i^\nu$, $i = 0, \ldots, n$ or if

(*b*) $x_k^\nu < y < x_{k+1}^\nu$ for $0 \leqslant k \leqslant n$ and

(i) $\text{sign}\,(f - P_n^\nu)(x_k^\nu) = \text{sign}\,(f - P_n^\nu)(y)$ then $x_k^{\nu+1} = y$ and $x_i^{\nu+1} = x_i^\nu$, for $i \neq k$ or

(ii) $\text{sign}\,(f - P_n^\nu)(x_k^\nu) = -\text{sign}\,(f - P_n^\nu)(y)$ then $x_{k+1}^\nu = y$ and $x_i^{\nu+1} = x_i^\nu$, for $i \neq k + 1$ or if

(*c*) $x_{n+1}^\nu < y$ we proceed analogously to case (*a*).

Now we again determine polynomials $Q_{n+1}^{\nu+1}$, $R_{n+1}^{\nu+1}$, and $P_n^{\nu+1}$. In this way we obtain a sequence of $L^\nu$ which converges to $E^*$ (see Meinandus, 1967).

## 2. Program description and implementation

The previously described algorithm has been implemented within the Numerical Analysis Problem Solving System (Roman and Symes, 1968), NAPSS, which is itself implemented on Purdue University Computing Centre's CDC6500/IBM7094 remote terminal system, PROCSY. Ninety per cent of NAPSS and all of the approximation algorithm are written in FORTRAN IV. The NAPSS system creates an interactive mathematical problem statement and solution environment. One significant feature provided allows the user to define a wide variety of mathematical functions and manipulate them with conventional functional operators such as differentiation and integration. For example, the mathematical functions

$$g(x) = \frac{1}{1 + x}$$

and

$$f(x) = \begin{cases} 1 + x \text{ for } -1 \leqslant x < -0{\cdot}5 \\ -x \text{ for } -0{\cdot}5 \leqslant x < 0 \\ x \text{ otherwise} \end{cases}$$

can be represented as 'symbolic' functions *G* and *F* in NAPSS by

$$G(X) \leftarrow 1/(1 + X)$$

and

$$\begin{aligned} F(X) &\leftarrow 1 + X \text{ FOR } -1 \leqslant X < -0{\cdot}5 \\ &\leftarrow -X \text{ FOR } -0{\cdot}5 \leqslant X < 0\ , \\ &\leftarrow X\ . \end{aligned}$$

In a similar fashion functions may be defined in terms of two vectors representing selected independent and dependent variable values. For example, the statements

$$\begin{aligned} &A \leftarrow (1, 2, 3) \\ &B \leftarrow (4, 5, 6) \\ &F(X \leftarrow A) \leftarrow B \end{aligned}$$

define a 'point-valued' function *F* where $F(1) = 4$, $F(2) = 5$, and $F(3) = 6$. Such a function is useful for manipulating certain types of experimental data and for retaining the numeric solution of a differential equation. Requests for evaluation of a point-valued function at undefined points are processed by means of a cubic interpolation algorithm. In this sense point-valued functions may be considered to be continuous.

Once the user has defined a function *F* which he wants to approximate with a polynomial *P* of degree *N* on the interval [*A*, *B*], he enters the NAPSS statement

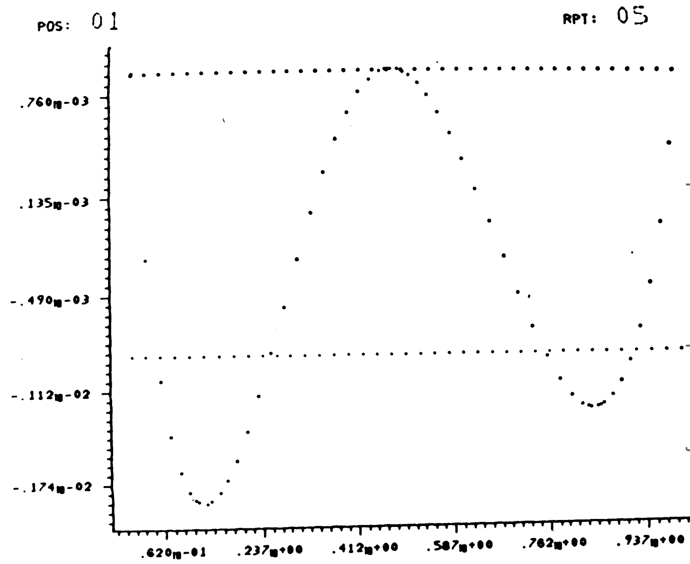
APPROXIMATE *F* BY *P* ON *A* TO *B* DEGREE *N* .

In response to this statement the NAPSS processor generates a six-step interactive approximation procedure.

*Step* 1:
Using *A*, *B* and *N*, generate $N + 2$ equally spaced interpolation nodes $X_0^1, \ldots, X_{N+1}^1$: in the interval [*A*, *B*].

```
POS: 38                                              RPT: 05

.... napss

  JOB ACN68

<> PURDUE  UNIVERSITY'S  NAPSS <>
<> 13:00:16               08/08/72 <>

  1.00  f(x) ← 1 / (1 + x)
  2.00  approximate f by p on 0 to 1 degree 3_
```

**Fig. 1**




```
X SCALE =  .0017500                         X = -0003600.0m-5
Y SCALE =   .625000m-05                     Y = -000203375.0m-8
.8929m-03 = ...    -.8929m-03 = ...   (EXPR)(X) = ...

<> PURDUE  UNIVERSITY'S  NAPSS <>
<> 13:21:51               08/08/72 <>

  1.00  f(x) ← 1 / (1 + x)
  2.00  approximate f by p on 0 to 1 degree 3
NEW DEGREE(RETURN OR VALUE): _
```

**Fig. 2**

*Step* 2:
Using *F*, *A*, *B*, *N*, and $X_0^\nu, \ldots, X_{N+1}^\nu$, generate $\pm L^\nu(1{\cdot}1)$ and the coefficients $C_0^\nu, \ldots, C_N^\nu$ of the approximation polynomial $P_n^\nu$ (1.2). (Here we used the fast algorithm described in (Gustafson, 1971)).

*Step* 3:
Using the coefficients from Step 2 define the polynomial $P_n^\nu$ as a NAPSS symbolic function in Horner's form.

*Step* 4:
Using *P* and *F* define the error function

$$E(X) = F(X) - P_n^\nu(X)\ .$$

*Step* 5:
Plot the function *E* and the two reference lines $+L^\nu$ and $-L^\nu$ on the interval [*A*, *B*]. Now the user has three possibilities for his response. In consequence.

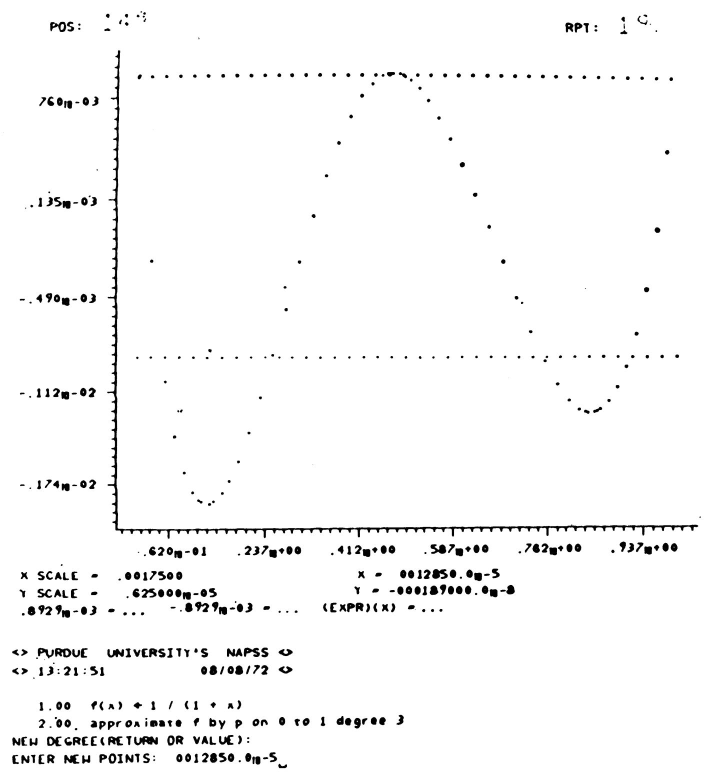



Fig. 3

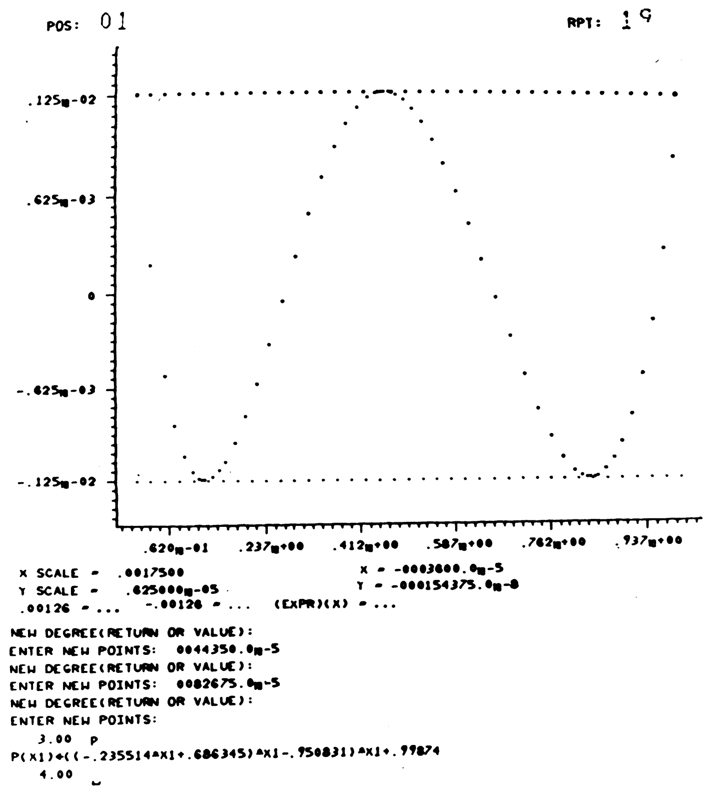



Fig. 5

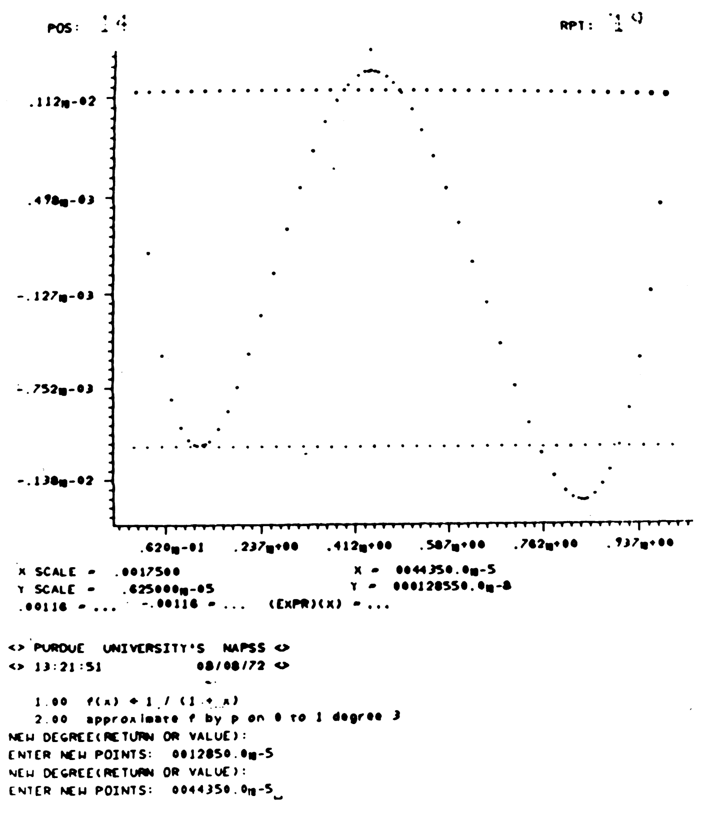



Fig. 4

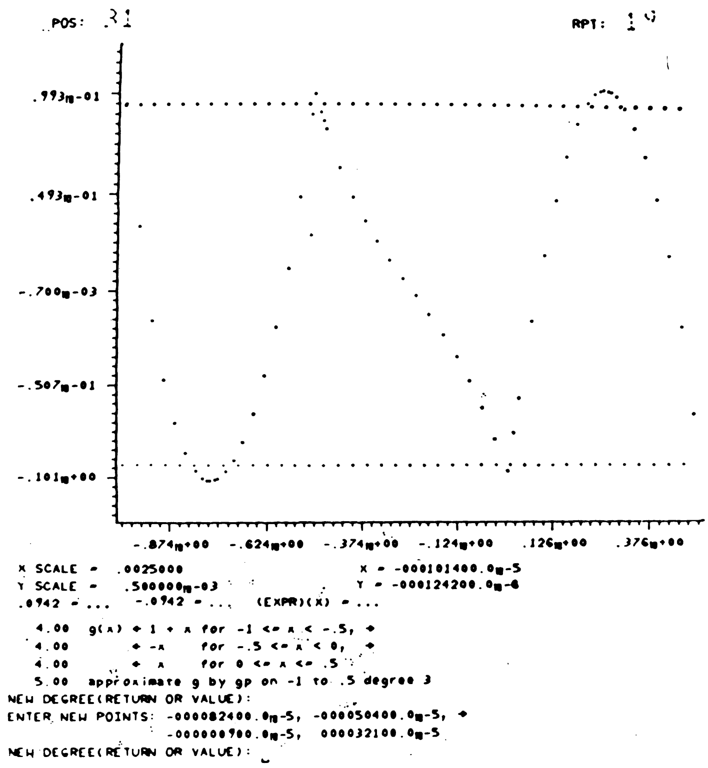



Fig. 6

*Step* 6:

Determine if the user

(*a*) is satisfied with the error curve

(*b*) wishes to define a new degree $N'$, in which case proceed to Step 1

(*c*) wishes to replace some of the $\{X_i\}^\nu$ by values extracted from the plot of $E(X)$. If so, generate a new set of $\{X_i\}^{\nu+1}$ incorporating the new values and an appropriate subset of $\{X_i\}^\nu$ as described in Section 1 and proceed to Step 2 with $\nu = \nu + 1$.

Steps 1 through 4 are invisible to the user. Step 5 requires no action from the user but the plot it produces is employed by the user in determining his response(s) in Step 6.

The facilities required for Steps 3 through 5 were present in NAPSS prior to the introduction of the APPROXIMATE statement. Steps 1, 2 and 6 are implemented by a single driving

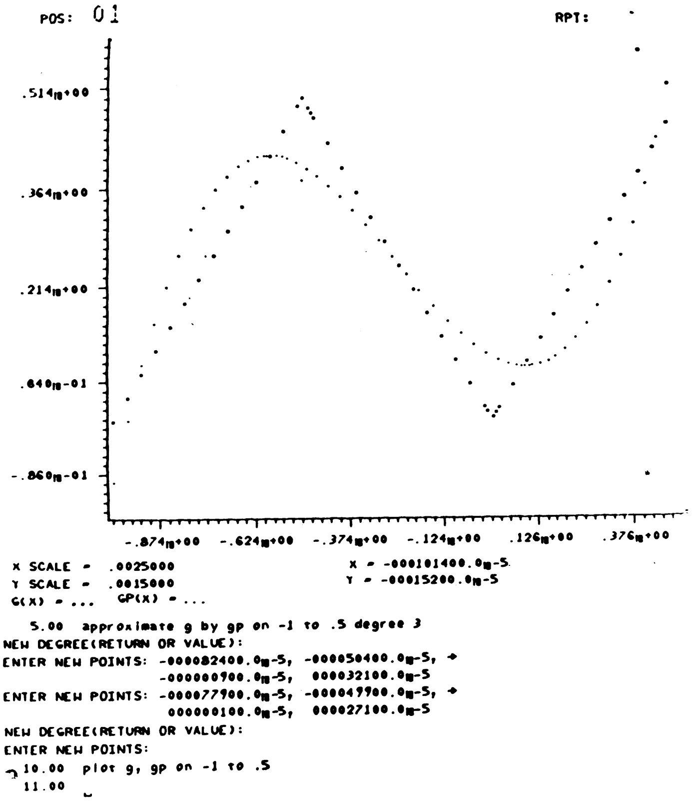



**Fig. 7**

subroutine with four service subroutines. The driving subroutine:

1. Distinguishes and supervises Steps 1, 2 and 6
2. Provides the interface between the NAPSS function and vector data structures and the service subroutines which are written in conventional argument-oriented FORTRAN
3. Handles error message processing.

The four service routines implement Step 1, Step 2, the interactive portion of Step 6, and the interpolation node replacement portion of Step 6c.

### 3. Description and illustrations of an actual terminal session

The graphics terminal employed in this implementation is the IMLAC PDS-1 Graphics Display Terminal produced by IMLAC Corporation, Waltham, Massachusetts. This terminal consists of a minicomputer with an 8K 16 bit word, 2 microsecond central storage, a keyboard for user input, and a display processor which refreshes the display screen under the direction of a program resident in central storage. In its present environment the PDS-1 communicates with the PROCSY terminal system via a standard terminal interface. Aside from a fast (600 baud) transmission rate this terminal receives no special services from PROCSY.

A software package was developed which allows the user to carry on conventional line-by-line interaction with PROCSY in the lower portion of the screen while observing and manipulating a two dimensional plot of one or more NAPSS functions in the top portion.

**Fig. 1** depicts the initial phase of the approximation procedure. The user has entered the NAPSS system, defined a function, $F(X) \leftarrow 1/(1 + X)$, and requested an approximation of $F$ by a third degree polynomial $P$ on the interval [0, 1].

**Fig. 2** shows the status of the screen at the beginning of Step 6 of the approximation algorithm. An initial $P_3^1(X)$ has been generated and the error function

$$E^1(X) = F(X) - P_3^1(X)$$

as well as the $+L^1$ (= 0·0008929) and $-L^1$ reference lines have been plotted. At this point the user may change the degree of the approximating polynomial or proceed to improve the third degree approximation.

In **Fig. 3** the user has indicated that a new degree is not desired and was prompted to select one or more new $X$ values to replace members of $\{X_i\}^1$ in the construction of $\{X_i\}^2$. The PDS-1 program permits the user to manipulate $X$-axis and $Y$-axis cursor lines within the body of the plot, independent of the remote terminal system. In the lower right corner of the plot area the current positions of the $X$-axis and $Y$-axis cursor lines are displayed numerically:

$$X = 0012850{\cdot}0_{10} - 5$$
$$Y = -000189000{\cdot}0_{10} - 8$$

The user has positioned the $X$-axis cursor line (faintly visible in the photograph) to select a value in $[A, B]$, $X^*$ (= 0·1285), such that

$$|F(X^*) - P_3^1(X^*)| > |F(x) - P_3^1(x)|, A \leqslant x \leqslant B \ .$$

By depressing a function key on the PDS-1 keyboard, the numeric value of $X^*$ is moved from the plot display to the user's current input line.

In **Fig. 4** the new $P_3^2(X)$ has been generated, a plot of

$$E^2(X) = F(X) - P_3^2(X)$$

appears, and the user has selected another $X^*$ (= 0·44350) for inclusion in $\{X_i\}^3$.

In **Fig. 5** we see the final plot of the error function after one additional iteration (not shown). After exiting the approximation algorithm (by not supplying an $X^*$) the user has wished to display the symbolic representation of $P_3^4(X)$,

$$P(X) = -0{\cdot}235514x^3 + 0{\cdot}686345x^2 - 0{\cdot}950831x + 0{\cdot}99874 .$$

The progress in the convergence of the approximation polynomial can be observed by examining the values of $L$, which according to de la Vallee Poussin increase until the error curve has attained its characteristic shape. In the initial plot (Fig. 3) $L^1$ is 0·0008929. In the second iteration (Fig. 4) $L^2$ is 0·00116. Finally, after the last iteration (Fig. 5) $L^4$ is 0·00126.

**Figs. 6** and **7** illustrate the approximation of a function of multiple domains. Fig. 6 shows the definition of the function $G(X)$, the plot of the error curve for $GP(X)$, and the user's response. In this instance the user selected four points for replacement during a single iteration. Technically this is a purer example of the Remez algorithm than the preceding one where single point replacement implements a one-for-one exchange algorithm. Fig. 7 is a simultaneous plot of $G(X)$ and $GP(X)$ after the final (rather unsatisfactory!) approximation.

### 4. Conclusions

As indicated in the mathematical formulation, this effort centered about a known algorithm. It appeared to the authors that the difficult portion of a fully automated Remez algorithm the detection of relative maximas and minimas in the error curve, might best be performed by human intelligence. This suggests that the development of new numerical analysis algorithms especially for a graphical interactive-environment may yield practical solutions in problem areas (multivariate integration, for example) which are presently too thorny for complete automation.


### References

Ehlich, H. (1967). Vorlesung über Approximationstheorie, Class notes, Bochum, W. G., unpublished.

Gustafson, S. (1971). Rapid computation of general interpolation formulas and mechanical quadrature rules, *CACM*, Vol. 14, No. 12 December 1971, pp. 797-801.

Meinardus, G. (1967). *Approximation of functions: theory and numerical methods,* (Schumaker, L. L., trans.), Springer-Verlag, Berlin, New York, N.Y.

Roman, R. V., and Symes, L. S. (1968). Implementation considerations in a numerical analysis problem solving system, *Interactive systems for experimental applied mathematics,* (Klerer, M., and Reinfelds, J., eds.), Academic Press, New York, N.Y. pp. 400-410.

Smith, L. B. (1970). A survey of interactive graphical systems for mathematics, *Computing surveys,* Vol. 2, No. 4 (December 1970), pp. 261-301.

Symes, L. S., and Roman, R. V. (1968). Structure of a language for a numerical analysis problem solving system, *Interactive systems for experimental applied mathematics,* (Klerer, M., and Reinfelds, J., eds.), Academic Press, New York, N.Y., pp. 67-78.

---

# Second International Conference on very large data bases

The second International Conference on VLDB will be held in 1976 and will focus on all aspects of very large data bases. The conference is scheduled to be held September 8-10, 1976 in Brussels, Belgium.

Solicited papers include (but are not limited to) the following areas:
DBMS gross architecture for very large data bases
Data base language specification for very large data bases
Security and access control
Sharing and simultaneous access
Structuring and restructuring
Integrity and recovery
User interface performance and evaluation
Technology transfer
Experience with construction and use of very large data base systems
Very large data base research centers
Distributed data base network
Toward very intelligent data base systems
Hardware for very large data bases
System performance measurement
Data base construction

The Conference will be chaired by: Professor Erich J. Neuhold, University of Stuttgart, W-Germany, Dr. Sakti P. Ghosh, IBM Research Laboratory, San José, California and G. M. Nijssen, Brussels, Belgium

The program co-chairpersons are: Dr. Vincent Y. Lum, IBM Research Laboratory, San José, California and Professor Peter Lockemann, University of Karlsruhe, W-Germany

Five copies of each full paper from America, Asia and Australia should be sent to Dr. V. Y. Lum, Computer Science Department, IBM Research Laboratory, 5600 Cottle Road, San José, CA. 95193, USA; and full papers from Europe and Africa to Professor Peter Lockemann, University of Karlsruhe, 7500 Karlsruhe, Postfach 6380, W-Germany. The deadline for submission of papers is 15 April, 1976. Notification will be mailed by 16 June, 1976.

A selected number of papers from the conference will be published in a special issue of *ACM Transactions on Database Systems* (TODS). All other conference papers will appear in the conference proceedings. Sponsors to support the conference are solicited, eg. IFIP, ACM, ECI, IEEE.