UNIFORM APPROXIMATION OF EMPIRICAL FUNCTIONS
BEING EITHER MONOTONE
OR HAVING A FIXED NUMBER OF EXTREMA

1. Procedure declarations.
(a) The procedure \textit{prodrevsimpl} solves the following problem:
For a given number sequence \(\{y_1, y_2, \ldots, y_p\}\) determine a sequence
\(\{z_1, z_2, \ldots, z_p\}\) with the following properties:
(i) the function
\[
f = \max_{1 \leq k \leq p} |y_k - z_k|
\]
reaches its minimum;
(ii) the sequence \(\{z_i\}\) is monotone.

Data:
\begin{itemize}
  \item \texttt{ext} — integer variable with value 0 if the sequence \(\{z_i\}\) is non-
      increasing, and with value 1 if the sequence \(\{z_i\}\) is non-
      decreasing;
  \item \texttt{p} — number of elements in the sequence \(\{y_i\}\);
  \item \texttt{d[1ː3×p]} — data array, where \(d[1] = y_1, d[2] = y_2, \ldots, d[p] = y_p\).
\end{itemize}

Results:
\(\texttt{d[1ː3×p]}\) — array of results, where \(z_1 = d[1], z_2 = d[2], \ldots, z_p = d[p]\),
and \(\texttt{d[2×p]}\) contains the approximation error.

(b) The procedure \textit{prodrevsimplextr} solves problem (a) with (ii)
changed to
(iii) the sequence \(\{z_i\}\) has a fixed number of extrema.

Data:
\begin{itemize}
  \item \texttt{lex} — number of extrema in the sequence \(\{z_i\}\);
  \item \texttt{ext} — integer variable with value 0 if the first extremum is a minimum,
      and with value 1 if it is a maximum.
\end{itemize}

All remaining data and result parameters are the same as in (a).
2. Method used. The algorithms used in both procedures are derived from classical linear programming methods. They make use of special features of the considered problem; hence

1° the procedures require a small amount of computer memory, thus long sequences can be hand'ed by them;

2° the computation time is rather small and the cost of computation is in reasonable limits.

The problem which is solved by the procedure prodrevsimpl can be formulated as follows:

Given \( y_1, y_2, \ldots, y_p \), minimize \( \max_{1 \leq k \leq p} \left| y_k - z_k \right| \)

to satisfy

\[ z_1 \leq z_2 \leq \ldots \leq z_p. \]

The problem can be written in the following equivalent form (see [4], p. 244 -245):

Minimize

\[ g = z_{p+1} \]

under the conditions

\[ y_k - z_k + z_{p+1} \geq 0, \quad -(y_k - z_k) + z_{p+1} \geq 0 \quad (k = 1, 2, \ldots, p), \]

\[ z_1 \leq z_2 \leq \ldots \leq z_p. \]

Problem (3)-(4) is a linear programming problem. Conditions (4) are enlarged by

\[ z_k \geq 0 \quad (k = 1, 2, \ldots, p). \]

Conditions (5) depend upon the sign of the data \( y_1, y_2, \ldots, y_p \). If there are negative numbers in the data or if the difference between the maximum and minimum values in the sequence \( y_1, y_2, \ldots, y_p \) is not less than the minimum value of this sequence, then, in order to fulfill (5), a sufficient-ly large positive number \( r \) should be added to all elements of the sequence \( y_1, y_2, \ldots, y_p \); it is subtracted after the calculations from all \( z_1, z_2, \ldots, z_p \).

To obtain the initial basic program we introduce artificial variables into (4) and (5) as well as the function \( g = z_{3p} \) (where \( z_{3p} = z_{p+1} \)) as the equation with index 3p, and also we assume that the quantities \( y_k \) for \( k = 1, 2, \ldots, p \) have been already, if necessary, increased by \( r \).
procedure prodrevesimpl(ext,p,d);
value p;
integer ext,p;
array d;
begin
integer f,g,h,i,j,k,k1,l,l1,l2,m,n;
Boolean fg,fl,fn;
f:=ext=1;
n:=3×p;
h:=n-1;
f:=p+p;
begin
real r,t,x,y,y1,y2;
integer array q[1:p+1],tb,tn,t1[1:p],s[1:n],sz[5:h];
array b[1:n],dd[1:p];
t:=r=d[f-1]=d[p];
d[f]:=t;
j:=g=0;
y:=if fg then -1.0 else 1.0;
for i:=f-3 step -2 until 1 do
begin
j:=j+1;
x=d[i]=d[p-j];
d[i+1]:=-x;
if t<x
then t:=x
else
if x<r
then r=x;
tb[j]=tn[j]=j
end i;

\text{tn}[p]=tb[p]=p;

\textbf{if } r>0

\textbf{then}

\textbf{begin}

\quad x=t-r;

\quad r=\textbf{if } x<r \textbf{ then } 0 \textbf{ else } x+1.0

\textbf{end } r>0

\textbf{else}

\textbf{begin}

\quad t=t-r;

\quad r=\text{abs}(r)+t+1.0

\textbf{end } r\leq 0;

\textbf{for } i=1 \textbf{ step } 2 \textbf{ until } f \textbf{ do }

\textbf{begin}

\quad d[i]=d[i]+r;

\quad d[i+1]=d[i+1]-r

\textbf{end } i;

\quad j=1;

\textbf{for } i=f+1 \textbf{ step } 1 \textbf{ until } h \textbf{ do }

\textbf{begin}

\quad sz[i]=\textbf{if } fg \textbf{ then } j+1 \textbf{ else } j;

\quad t1[j]=i;

\quad j=j+1

\textbf{end } i;

\quad t1[p]=n;

\quad j=p+1;

\textbf{for } i=1 \textbf{ step } 1 \textbf{ until } n \textbf{ do }

\textbf{begin}

\quad b[i]=\textbf{if } igf \textbf{ then } d[i] \textbf{ else } 0;
\[ a[i] = i + j \]

\[ \text{end } i; \]
\[ i1 = p; \]
\[ \text{iter:} \]
\[ \text{if } g \neq l1 \]
\[ \text{then} \]
\[ \text{begin} \]
\[ t := b[2]; \]
\[ m = 2; \]
\[ j = 4; \]
\[ \text{et20: for } i = j \text{ step } 2 \text{ until } f \text{ do} \]
\[ \text{begin} \]
\[ x := b[i]; \]
\[ \text{if } x < t \]
\[ \text{then} \]
\[ \text{begin} \]
\[ t := x; \]
\[ m := i \]
\[ \text{end } x < t \]
\[ \text{end } i \]
\[ \text{end } g \neq l1 \]
\[ \text{else} \]
\[ \text{begin} \]
\[ t := b[1]; \]
\[ m := 1; \]
\[ j := 3; \]
\[ \text{so to et20} \]
\[ \text{end } g = l1; \]
\[ \text{if } t < .0 \]
\[ \text{then} \]
begin
iter1:
if \( t \geq m \)
then
begin
\( j = m + 2; \)
\( k = \begin{cases} j & \text{if } j \leq 5 \times m \text{ then } j \\ 1 + 1 & \text{else} \end{cases} \)
end \( \geq m \)
else \( k = sz[m]; \)
\( s[m] = k; \)
\( g = g + 1; \)
\( q[g] = m; \)
\( y_1 = y_2 = 0; \)
\( t = \begin{cases} -t & \text{if } k \leq 11 \text{ then } \text{else} \\ -5 \times t & \end{cases} \)
if \( k = 11 + 1 \)
then
begin
\( b[h + 1] = -t; \)
\( b[m] = t; \)
\( j = m - 1; \)
for \( i = 1 \) step 1 until \( j, m + 1 \) step 1 until \( h \) do
\( b[i] = b[i] - t \times \begin{cases} -2.0 & \text{if } i < (i + 1) \times 5 & \text{then} \\ 1.0 & \text{else} \end{cases} \)
end go to et1
end \( k = 11 + 1 \)
else
begin
\( j = k + k; \)
\( b[j] = b[j] + t; \)
\( b[j - 1] = b[j - 1] - t; \)
\[ x = ty; \]

**if** \( k = 1 \)

**then**

**begin**

**if** \( t_n[1] = 0 \)

**then**

**begin**

\[ y_1 = b[f+1] = b[f+1] - x; \]

\[ y_2 = b[f+2] = b[f+2] + x; \]

\[ j = \text{if} \ y_1 < 0 \ \text{then} \ f+1 \ \text{else if} \ y_2 < 0 \ \text{then} \ f+2 \ \text{else} \ j \]

**end**

**else**

**begin**

\[ y_1 = b[f+1] = b[f+1] + x; \]

**if** \( y_1 < 0 \)

**then** \( j = f+1 \)

**end**

**end** \( k = 1 \)

**else**

**if** \( k = 11 \)

**then**

**begin**

**if** \( t_n[p] = 0 \)

**then**

**begin**

\[ y_1 = b[h-1] = b[h-1] - x; \]

\[ y_2 = b[h] = b[h] + x; \]

\[ j = \text{if} \ y_1 < 0 \ \text{then} \ h-1 \ \text{else if} \ y_2 < 0 \ \text{then} \ h \ \text{else} \ j \]

**end**

**else**
\textbf{begin}
\begin{align*}
y_1 &= b[h] = b[h] - x; \\
\text{if} \ y_1 < 0 \\
\text{then} \ j &= h \\
\text{end}
\end{align*}
\textbf{end}
\textbf{else}
\begin{align*}
\textbf{begin} \\
 j &= t1[k]; \\
y_1 &= b[j] = b[j] + x; \\
y_2 &= b[j-1] = b[j-1] - x; \\
j &= \text{if} \ y_1 < 0 \ \text{then} \ j \ \text{else if} \ y_2 < 0 \ \text{then} \ j-1 \ \text{else} \ j
\end{align*}
\textbf{end};
\begin{align*}
b[m] &= t \\
\text{end} \ k &= l1 + 1; \\
\textbf{if} \ y_1 < 0 \\
\text{then} \ \textbf{begin} \\
t &= y_1; \\
m &= j; \\
\text{go to iter1} \\
\textbf{end}
\end{align*}
\textbf{end \ y_1 < 0}
\textbf{else}
\begin{align*}
\textbf{if} \ y_2 < 0 \\
\text{then} \ \textbf{begin} \\
t &= y_2; \\
m &= j; \\
\text{go to iter1} \\
\textbf{end};
\end{align*}
go to iter
and t<.0
else
et1: if b[h+1]>.0
    then
    begin
    fn=true;
    h=.5*f;
    j=q[g];
    for i=g-1 step -1 until 1 do
    begin
    m=q[i];
    k=s[m];
    fl=true;
    l1=t1[k];
    k1=k+k-1;
    if ~fg
    then l1=l1-1;
    if k1+k+1
    then
    begin
    if j=k1\lor j=l1
    then
    begin
    end
    end
else
    if k=1\lor(j=k1\lor j=l1)
then go to et2
else
  if \( k = h \wedge (j = k 1 \vee j = 1 1) \)
    then go to et2;
if \( f 1 \)
  then
else
  if \( f n \)
    then
    begin
    \( l = l 2 = m; \)
    \( f n = false \)
    end
else \( l 2 = m \)
and \( 1; \)
x = b[1];
1 = s[1];
l2 = s[l2];
if \( l > l 2 \)
  then
  begin
  i = 1;
  1 = l 2;
  l 2 = i
  and \( l > l 2; \)
k 1 = .5 \times f;
for \( i = 1 2 + 1 \) step 1 until \( k 1 \) do
  begin
    j = t b[i];
    if \( t n[j] = 0 \)
then go to et6
else
  if fg
    then
      begin
        if d[j+j-1]<x
          then l2=1
          end
        and fg
      else
        if d[j+j-1]>x
          then l2=1
          end
        and i;
  et6: for i=1-1 step -1 until 1 do
      begin
        j:=tb[i];
        if tn[j]=0
          then go to et7
        else
          if fg
            then
              begin
                if d[j+j-1]>x
                  then l=1
                  end
                and fg
              else
                if d[j+j-1]<x
                  then l=1
                  end
                and i;
          et7: for i=1 step 1 until l2 do
              begin

\[ j = tb[1]; \]
\[ dd[j] = x; \]
\[ tn[j] = 0 \]

\textbf{end 1;}

\[ j = 1; \]

\[ k = 12; \]

\textbf{if } 1 \leq 2 \textbf{ then } 1 = 1 \textbf{ else}

\textbf{if } tb[1] \leq tb[1-1] + 1 \textbf{ then}

\textbf{else}

\textbf{if } tb[1] \leq tb[1-2] + 2 \textbf{ then } 1 = 1 - 1; \]

\textbf{if } 12 \geq k1 - 1 \textbf{ then } 12 = k1 \textbf{ else}

\textbf{if } tb[12] \leq tb[12+1] - 1 \textbf{ then}

\textbf{else}

\textbf{if } tb[12] \leq tb[12+2] - 2 \textbf{ then } 12 = 12 + 1; \]

\textbf{if } j \neq 1 \textbf{ then}

\textbf{begin}

\[ i = tb[1]; \]
\[ dd[i] = -d[i+1]; \]
\[ tn[i] = 0 \]

\textbf{end } j \neq 1; \]

\textbf{if } k \neq 12
then
begin
i=tb[12];
dd[i]=-d[i+1];
tn[i]=0
end k112;
h=f=f-2*(12-1+1);
l1=.5*f;
j=1;
if k112
then
for i=1 step 1 until l1 do
begin
  tb[i]=tb[12+j];
  j=j+1
end k112,i;
k=0;
for i=1 step 1 until p do
  if tn[i]>0
    then
    begin
      k=k+2;
      b[k-1]=d[i+i-1];
      b[k]=d[i+i-1]
    end tn[i]>0;
l=k=j=0;
w=tn[1];
l1=f+1;
fn=-fg\w=0\Vfg\w>0;
fl=fg\w>0;
for i=1 step 1 until p do
  if tn[i]=0
    then j=j+1
  else
    if j≠0
      then
        begin
          j=0;
          k=k+1;
          h=h+2;
          g=f+k;
          l=l+1;
          t1[l]=if fn=fg then g else g+1;
          if ¬fn
            then sz[g]=1
          else
            if fg
              then sz[g]=l+1
              else sz[g+1]=1;
          if m=0∧k=1
            then b[f+1]=y×dd[i-1]
          else
            begin
              x=y×dd[i-1];
              if m≠0
                then g=g-1;
              b[g]:=x;
              b[g-1]:=-x
            end;
          k=k+1
end
else
  if k=0
  then
  begin
    k=k+1;
    go to et8
  end
else
  begin
    et8:
    g=f+k;
    if m\not=0\land g\not=12
    then b[g-1]=-0
    else b[g]=-0;
    l=l+1;
    t1[l]=if fn=fg then g else g+1;
    if l+1>p
    then
      else
        if t[m][i+1]=0\land f1
        then g=g+1;
      if -fn
      then sz[g]=1
      else
        if fg
        then sz[g]=l+1
        else sz[g+1]=1;
    h=h+1;
    k=k+1
  end i;
if \( j=p \)
then go to et9;
if \( t[n][p]=0 \)
then \( b[h]=-y \cdot d[i-2] \)
else \( h=h-1 \);
\( b[h+1]=0 \);
\( j=j+1 \);
\( k=h+1 \);
for \( i=1 \) step 1 until \( k \) do
\( s[i]=i+j \);
\( g=0 \);
go to iter
end \( b[h+1]=0 \);
for \( i=1 \) step 1 until \( l[l] \) do
\( d[tb[i]]=b[i+i] \);
end
for \( i=1 \) step 1 until \( p \) do
\( d[i]=d[i]-r \);
\( d[p+p]=-b[n] \)
end block
end prodrevsimp

Problem (3)-(5) can be written as follows:

Minimize

\[
(6) \quad z_{zp}
\]

under the conditions

\[
\begin{align*}
z_k - z_{p+1} + z_{2k-1}^d &= y_k \quad (k = 1, 2, \ldots, p), \\
-z_k - z_{p+1} + z_{2k}^d &= -y_k \\
z_1 - z_2 + z_{2p+1}^d &= 0, \\
z_2 - z_3 + z_{2p+2}^d &= 0, \\
& \quad \ldots \ldots \ldots \ldots \\
z_{p-1} - z_p + z_{3p-1}^d &= 0, \\
-z_{p+1} + z_{3p} &= 0, \\
z_k & \geq 0 \quad (k = 1, 2, \ldots, p), \quad z_l^d \geq 0 \quad (l = 1, 2, \ldots, p).
\end{align*}
\]
This problem can be solved by using the dual simplex algorithm ([3], § 6.4) and the revised simplex algorithm with product form of the inverse matrix ([1], [3]). These algorithms do not, however, take advantage of the characteristic features of this problem, therefore computer time is large and there may be lack of memory for greater input sequences.

Let us take the coefficients from (7) and form the following matrix of dimension $3p \times (p + 1)$:

$$
\bar{A} = \begin{bmatrix}
1 & 0 & 0 & \ldots & 0 & -1 \\
-1 & 0 & 0 & \ldots & 0 & -1 \\
0 & 1 & 0 & \ldots & 0 & -1 \\
0 & -1 & 0 & \ldots & 0 & -1 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & \ldots & 1 & -1 \\
0 & 0 & 0 & \ldots & -1 & -1 \\
1 & -1 & 0 & \ldots & 0 & 0 \\
0 & 1 & -1 & \ldots & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & \ldots & 1 & -1 \\
0 & 0 & 0 & \ldots & 0 & 1 & -1 \\
0 & 0 & 0 & \ldots & 0 & 0 & -1 
\end{bmatrix}
$$

Denote by $U$ the unit matrix of dimension $3p \times 3p$. The first $3p - 1$ rows and columns of $U$ form the initial basis $B$. Notice that the basic solution

$$d = (d_1, d_2, \ldots, d_{3p - 1}, d_{3p}) = (y_1, -y_1, y_2, -y_2, \ldots, y_p, -y_p, 0, \ldots, 0),$$

where $y_k > 0$ ($k = 1, 2, \ldots, p$), has negative components, but is a dual feasible initial solution.

Using the revised simplex algorithm, we calculate

$$\gamma_j = \bar{U}_{3p} \bar{A}_j \quad (j = 1, 2, \ldots, p + 1),$$

where $\bar{U}_{3p}$ denotes the last row vector of $U$ and $\bar{A}_j$ is the $j$-th column vector of $\bar{A}$. Since $\gamma_j \leq 0$ ($j = 1, 2, \ldots, p + 1$), the optimality conditions are satisfied. Thus, using the dual and revised simplex algorithms, the problem can be solved.

Now, let us consider the structure of the matrix $\bar{A}$. Notice that the non-zero elements of $\bar{A}$ can be calculated from the following equations which follow directly from (7):

$$
d_{ik} = \begin{cases}
1 & (i = 2k - 1, 2p + k), \\
-1 & (i = 2k, 2p + k - 1), 
\end{cases}
$$
where \( k \) is not equal to \( 1, p \) and \( p + 1 \), and

\[
\begin{align*}
\alpha_{i1} &= \begin{cases} 
1 & (i = 1, \, 2p + 1), \\
-1 & (i = 2),
\end{cases} \\
\alpha_{ip} &= \begin{cases} 
1 & (i = 2p - 1), \\
-1 & (i = 2p, \, 3p - 1),
\end{cases} \\
\alpha_{i, p+1} &= -1 \quad (i = 1, 2, \ldots, 2p, 3p).
\end{align*}
\]

There exists thus a possibility of forming the appropriate column vector of \( \overline{A} \) during the calculations. This is the first group of simplifications. The second simplification group follows from an analysis of the location of the non-zero elements in \( \overline{A} \). Additional simplifications are possible by the use of the product form of the inverse \( B^{-1} \) of the newly formed basis \( B' \).

Using the dual exit criterion

\[
d_i = \min_{d_i < 0} \{d_i\},
\]

we make the following

**Agreement.** If the minimum is reached for more than one \( d_i \), choose that one whose index \( l \) is the greatest one.

It is known that the inverse matrix in the \( t \)-th iteration can be obtained from the formula

\[
J^t_0 J^t_{l-1} \cdots J^t_l = B_t^{-1},
\]

where \( J_0 = I = B^{-1} \) and \( J_l^t \) is the elementary matrix of the \( t \)-th iteration, i.e. the matrix which is obtained from the unit matrix by replacing its column \( l \) with the column \( V_i \); the elements of \( V_i \) are determined as

\[
v_{il} = -\frac{w_{lk}}{w_{ik}} \quad (i \neq l), \quad v_{il} = \frac{1}{w_{lk}},
\]

the vector \( W_k \) being calculated from

\[
W_k = J_l^t (J^t_{l-1} \cdots (J_l^t \overline{A}_k) \cdots),
\]

where the index \( l \) indicates the column of the previous basis which is now being replaced.

Multiplication from the right of \( \overline{A}_j = (a_{1j}, a_{2j}, \ldots, a_{3p_j}) \) by the elementary matrix \( J^t_i \) leads to the vector \( W_j \) with components

\[
w_{ij} = a_{ij} + v_{il} a_{lj} \quad (i \neq l), \quad w_{ij} = v_{il} a_{lj}.
\]

Let us perform now a detailed analysis of the behaviour of the vector \( W_k \) during the iteration process. As is easily seen, in the first iteration we have \( W_k = \overline{A}_k \). Using (16), let us calculate \( W'_k \) in the second iteration:
$W_k' = J_1^l \bar{A}_{k'}$. The vector $\bar{A}_{p+1}$ enters the basis in the last iteration (see [2], Theorem 3.4). From the first iteration we obtain the relation $w_{lk} = a_{lk} = -1$, thus $v_l = 1/w_{lk} = -1$, where $l \leq 2p$ ($l$ is even) and $k = l/2$. From (14) we know that the vector $V_l$ for the matrix $J_1^l$ is equal to the vector $\bar{A}_k$.

Now, calculating the vector $W_k'$, let us notice that $a_{lj'} = 0$ in the vector $\bar{A}_{k'}$. This is due to the fact that the vector which leaves the basis in the first iteration has an index $l \leq 2p$ and in the rows from 1 to $2p$, with the exception of the row $p+1$, there is only one non-zero element. Therefore, by (16) we have the identity

$$W_k' = \bar{A}_{k'}.$$

and since the pivotal element $a_{lk'}$ equals $-1$, due to (14) for the matrix $J_2^l$, we obtain

$$V_l = W_k' = \bar{A}_{k'}.$$

From (17) we get

$$J_1^l \bar{A}_j = \bar{A}_j \quad (j = 1, 2, \ldots, p).$$

Similarly, for the matrix $J_2^l$ we have

$$J_2^l \bar{A}_j = \bar{A}_j \quad (j = 1, 2, \ldots, p).$$

This follows from (16) in which

$$a_{lj} = 0 \quad (j = 1, 2, \ldots, p),$$

where the index $l$ corresponds to the matrix $J_2^l$.

To prove (21), consider two possibilities:

(a) $l \leq 2p$,

(b) $l > 2p$.

In case (a) the derivation of (21) is the same as for (17).

Consider now case (b). Analyzing the formulas for the calculation of the vector $d$ notice that a negative element of this vector can appear for an index $l > 2p$ if the vector entering the basis in the first iteration has 1 at the $l$-th place. In the row $l$ we have one more non-zero element equal to $-1$. The vector $W_k'$ which corresponds to this element is already in the matrix $J_2^l$.

In fact, due to (13) and the Agreement, in the second iteration the basis is leaving a vector with index $l > 2p$ if the vector which entered the basis in the first iteration has the $l$-th element equal to 1. It follows from (19) that in the second iteration the candidate for the basis has to be searched among the secondary vectors of $\bar{A}$.

As we know, in the row $l$ there is still one non-zero element equal to $-1$ and the vector $W_k' = \bar{A}_{k'}$ whose $l$-th element is equal to $-1$ will
enter the basis in this iteration; from (18) we conclude that this will be the vector \( V_i \) of the matrix \( J_i^t \). Thus relation (21) must hold, and then (20) follows from (16).

Hence we have the following general statement:

**Theorem 1.** In a given iteration \( t \) the product of the actual inverse basis \( B^{-1} \) and any (but not the last) vector of \( \overline{A} \) is equal to this vector, i.e. \( B^{-1} \overline{A}_j = \overline{A}_j \) (\( j = 1, 2, \ldots, p \)).

From (15), Theorem 1 and the fact that the pivotal element \( a_{ik} \) of the vector \( \overline{A}_k \) equals \(-1\) we obtain

**Corollary 1.** In any iteration (with the exception of the last one) we have \( W_k = \overline{A}_k \), where \( \overline{A}_k \) is the vector entering the basis in this iteration.

To calculate the index of the vector which has to enter the basis in a given iteration, we must know, in accordance with the dual simplex algorithm, the quantity \( \gamma_j \) which can be calculated from (8). The actual vector \( U_{3p} \) can be evaluated from

\[
(\ldots (e_{3p}J_i^1)J_{i-1} \ldots )J_1^t = U_{3p},
\]

where \( e_{3p} = (0, 0, \ldots, 0, 1) \) is a vector with \( 3p \) components. The elements of \( U_{3p} \) can be determined by the relation

\[
u_l = e_i \quad (i \neq l), \quad u_l = e_i v_i.
\]

**Theorem 2.** In any iteration the vector \( U_{3p} \) is a unit vector, i.e.

\[
U_{3p} = (0, 0, \ldots, 0, 1).
\]

**Proof.** In the matrix \( \overline{A} \) the row with index \( 3p \) has only one non-zero element equal to \(-1\) which lies at the place \( p-1 \). We know from Corollary 1 that for the matrix \( J_1^t \) we have \( V_i = \overline{A}_j \) (\( j = 1, 2, \ldots, p \)). The product \( e_{3p}J_i^1 \) gives the unit vector \( U_{3p} \) because \( v_{3p,i} = a_{3p,i} = 0 \), thus \( u_l = 0 \) in (22).

Therefore, from the identity \( \gamma_j = U_{3p} \overline{A}_j \) (\( j = 1, 2, \ldots, p+1 \)) we get

\[
\gamma_j = \begin{cases} 
0 & (j = 1, 2, \ldots, p) \\
-1 & (j = p+1)
\end{cases}
\]

Thus in the second iteration the basis is entered by one of the vectors \( \overline{A}_j \) (\( j = 1, 2, \ldots, p \)) which follows directly from the dual simplex algorithm.

The next product \( (e_{3p}J_2^2)J_1^t = U_{3p} \) is also a unit vector, since for the matrix \( J_2^t \) we have \( v_{3p,i} = a_{3p,i} = 0 \) (this holds also for the matrix \( J_1^t \)) and by (22) we know that the vector \( U_{3p} \) will have only one non-zero element \( u_{3p} = 1 \).

This reasoning can be repeated for all subsequent matrices \( J_i^t \).

By Theorem 2 and by (8) we come to the following
COROLLARY 2. In any iteration the quantities $\gamma_j$ can be calculated by the relations $\gamma_j = 0$ ($j = 1, 2, \ldots, p$) and $\gamma_{p+1} = -1$.

THEOREM 3. In any iteration the index $k$ of the vector entering the basis can be determined by the relation

$$k = \begin{cases} 
\frac{l}{2} & \text{for } l \leq 2p \text{ and } l \text{ even}, \\
p + 1 & \text{for } l < 2p \text{ and } l \text{ odd}, \\
l - 2p + 1 & \text{for } l > 2p,
\end{cases}$$

where $l$ is the index of the vector which leaves the basis.

Proof. In the case of $l$ even and $l \leq 2p$ the relation $k = l/2$ follows from (9)-(11) and so does the relation $k = l - 2p + 1$ for $l > 2p$. If $l$ is odd and $l < 2p$, the relation $k = p + 1$ follows from (12).

In the last iteration the vector $W_k$ can also be determined by simple relations. We have the following

THEOREM 4. The index $k$ of the vector $W_k$ from the last iteration ($k = p + 1$) can be determined as follows:

$$k = \begin{cases} 
-2 & \text{for } i = 1, 3, \ldots, 2p - 1, \\
1 & \text{for } i \in I, \text{ where } I \text{ is the index set of the vectors leaving the basis in the given iteration}, \\
-1 & \text{for } i = 3p, \\
0 & \text{for all remaining } i.
\end{cases}$$

The proof of this theorem can be found in [2] (Theorem 3.6).

The actual vector $d'$ can be calculated by the formulas

$$d'_i = d_i - \frac{d_i}{w_{ik}} w_{ik} \ (i \neq l), \quad d'_l = \frac{d_l}{w_{ik}},$$

where $w_{ik} = -1$ in all but the last iterations, and in the last iteration, $w_{ik} = -2$.

The non-zero elements $w_{ik}$ are equal to 1 or $-1$ with the exception of the last iteration in which an element equal to $-2$ appears.

Hence the values of the vector $d'$ in all iterations can be determined as

$$d'_i = d_i + \delta d_i \ (i \neq l), \quad d'_l = \delta d_l,$$

where

$$\delta = \begin{cases} 
0 & \text{for } w_{ik} = 0, \\
-1 & \text{for } w_{ik} = -1 \text{ not in the iteration } p + 1 \text{ and for } w_{ik} = -2, \\
1 & \text{for } w_{ik} = 1 \text{ in the iterations } t = 1, 2, \ldots, p, \\
.5 & \text{for } w_{ik} = 1 \text{ in the iteration } p + 1, \\
-.5 & \text{for } w_{ik} = -1 \text{ in the iteration } p + 1,
\end{cases}$$

$$\delta_1 = \begin{cases} 
-.5 & \text{in the iteration } p + 1, \\
-1 & \text{in the iterations } t = 1, 2, \ldots, p.
\end{cases}$$
As follows from Corollary 1, formulas (16) can be replaced by (9)-(11), and in the last iteration — by (23).

In the algorithm realized by the procedure *prodresimplexextr* the results of problem (1)-(2) are used, i.e. one has to remember the optimum solution of problem (1)-(2), the index set of the vectors leaving the basis in consecutive iterations and the index set of the vectors entering the basis in consecutive iterations.

The procedures take also care of the case where there is more than one optimum solution. For instance, consider the following problem:

Minimize

\[
\max_{1 \leq k \leq 5} |y_k - z_k|
\]

provided \(z_1 \leq z_2 \leq z_3 \leq z_4 \leq z_5\), where \(\{y_1, y_2, y_3, y_4, y_5\} = \{3, 5, 7, 6, 8\}\).

The error of the optimum approximation for this problem is equal to .5, and the optimum solution is of the form

\(\{z_1, z_2, z_3, z_4, z_5\} = \{2.5, 4.5, 6.5, 6.5, 7.5\}\).

It seems to be natural to assume \(\{3, 5, 6.5, 6.5, 8\}\) as the optimum solution of this problem. This follows from the fact that we change only those values which violate the monotonicity of the function, leaving the other values unchanged. The procedure gives such a type of solution.

3. **Certification.** Table 1 gives the calculation times for some examples, the calculations having been carried out on the Odra 1204 computer. As is seen, the calculation times depend not only on the number of data, but also on their values.

<table>
<thead>
<tr>
<th>Number of data points</th>
<th>Number of extrema</th>
<th>Calculation time (in secs.)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><em>prodresimpl</em></td>
</tr>
<tr>
<td>25</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>50</td>
<td>3</td>
<td>22</td>
</tr>
<tr>
<td>95</td>
<td>3</td>
<td>120</td>
</tr>
<tr>
<td>95</td>
<td>3</td>
<td>150</td>
</tr>
<tr>
<td>95</td>
<td>3</td>
<td>70</td>
</tr>
</tbody>
</table>

Table 2 shows the calculation results for an example and with the use of the procedure *prodrevsimplexextr*. As data, sine function values with arguments \(.1k\) (\(k = 0, 1, \ldots, 94\)) are used, perturbed by the pseudorandom number equidistributed in the interval \((- .05, .05)\). The optimum approximation error equals .025.
<table>
<thead>
<tr>
<th>$k$</th>
<th>Data</th>
<th>Results</th>
<th>$k$</th>
<th>Data</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>.030</td>
<td>.030</td>
<td>48</td>
<td>-1.039</td>
<td>-1.039</td>
</tr>
<tr>
<td>1</td>
<td>.120</td>
<td>.120</td>
<td>49</td>
<td>-1.008</td>
<td>-1.008</td>
</tr>
<tr>
<td>2</td>
<td>.218</td>
<td>.218</td>
<td>50</td>
<td>-.933</td>
<td>-.933</td>
</tr>
<tr>
<td>3</td>
<td>.251</td>
<td>.251</td>
<td>51</td>
<td>-.882</td>
<td>-.888</td>
</tr>
<tr>
<td>4</td>
<td>.412</td>
<td>.412</td>
<td>52</td>
<td>-.895</td>
<td>-.888</td>
</tr>
<tr>
<td>5</td>
<td>.495</td>
<td>.495</td>
<td>53</td>
<td>-.833</td>
<td>-.833</td>
</tr>
<tr>
<td>6</td>
<td>.565</td>
<td>.565</td>
<td>54</td>
<td>-.740</td>
<td>-.740</td>
</tr>
<tr>
<td>7</td>
<td>.655</td>
<td>.655</td>
<td>55</td>
<td>-.728</td>
<td>-.728</td>
</tr>
<tr>
<td>8</td>
<td>.710</td>
<td>.710</td>
<td>56</td>
<td>-.661</td>
<td>-.661</td>
</tr>
<tr>
<td>9</td>
<td>.773</td>
<td>.773</td>
<td>57</td>
<td>-.530</td>
<td>-.530</td>
</tr>
<tr>
<td>10</td>
<td>.812</td>
<td>.812</td>
<td>58</td>
<td>-.419</td>
<td>-.419</td>
</tr>
<tr>
<td>11</td>
<td>.909</td>
<td>.906</td>
<td>59</td>
<td>-.363</td>
<td>-.363</td>
</tr>
<tr>
<td>12</td>
<td>.902</td>
<td>.906</td>
<td>60</td>
<td>-.326</td>
<td>-.326</td>
</tr>
<tr>
<td>13</td>
<td>.933</td>
<td>.933</td>
<td>61</td>
<td>-.210</td>
<td>-.210</td>
</tr>
<tr>
<td>14</td>
<td>1.006</td>
<td>.989</td>
<td>62</td>
<td>-.090</td>
<td>-.090</td>
</tr>
<tr>
<td>15</td>
<td>.974</td>
<td>.989</td>
<td>63</td>
<td>-.011</td>
<td>-.011</td>
</tr>
<tr>
<td>16</td>
<td>1.034</td>
<td>1.034</td>
<td>64</td>
<td>.166</td>
<td>.166</td>
</tr>
<tr>
<td>17</td>
<td>.943</td>
<td>.968</td>
<td>65</td>
<td>.166</td>
<td>.166</td>
</tr>
<tr>
<td>18</td>
<td>.993</td>
<td>.968</td>
<td>66</td>
<td>.334</td>
<td>.334</td>
</tr>
<tr>
<td>19</td>
<td>.970</td>
<td>.968</td>
<td>67</td>
<td>.421</td>
<td>.421</td>
</tr>
<tr>
<td>20</td>
<td>.954</td>
<td>.964</td>
<td>68</td>
<td>.481</td>
<td>.481</td>
</tr>
<tr>
<td>21</td>
<td>.886</td>
<td>.885</td>
<td>69</td>
<td>.606</td>
<td>.606</td>
</tr>
<tr>
<td>22</td>
<td>.759</td>
<td>.765</td>
<td>70</td>
<td>.685</td>
<td>.685</td>
</tr>
<tr>
<td>23</td>
<td>.771</td>
<td>.765</td>
<td>71</td>
<td>.740</td>
<td>.740</td>
</tr>
<tr>
<td>24</td>
<td>.713</td>
<td>.713</td>
<td>72</td>
<td>.804</td>
<td>.804</td>
</tr>
<tr>
<td>25</td>
<td>.622</td>
<td>.622</td>
<td>73</td>
<td>.868</td>
<td>.868</td>
</tr>
<tr>
<td>26</td>
<td>.501</td>
<td>.501</td>
<td>74</td>
<td>.930</td>
<td>.925</td>
</tr>
<tr>
<td>27</td>
<td>.441</td>
<td>.441</td>
<td>75</td>
<td>.920</td>
<td>.925</td>
</tr>
<tr>
<td>28</td>
<td>.320</td>
<td>.320</td>
<td>76</td>
<td>1.002</td>
<td>1.002</td>
</tr>
<tr>
<td>29</td>
<td>.205</td>
<td>.205</td>
<td>77</td>
<td>.978</td>
<td>.990</td>
</tr>
<tr>
<td>30</td>
<td>.148</td>
<td>.148</td>
<td>78</td>
<td>1.004</td>
<td>1.002</td>
</tr>
<tr>
<td>31</td>
<td>.025</td>
<td>.025</td>
<td>79</td>
<td>1.000</td>
<td>1.002</td>
</tr>
<tr>
<td>32</td>
<td>-.065</td>
<td>-.065</td>
<td>80</td>
<td>1.038</td>
<td>1.038</td>
</tr>
<tr>
<td>33</td>
<td>-.161</td>
<td>-.161</td>
<td>81</td>
<td>1.011</td>
<td>1.011</td>
</tr>
<tr>
<td>34</td>
<td>-.258</td>
<td>-.258</td>
<td>82</td>
<td>.898</td>
<td>.913</td>
</tr>
<tr>
<td>35</td>
<td>-.326</td>
<td>-.326</td>
<td>83</td>
<td>.929</td>
<td>.913</td>
</tr>
<tr>
<td>36</td>
<td>-.481</td>
<td>-.481</td>
<td>84</td>
<td>.826</td>
<td>.826</td>
</tr>
<tr>
<td>37</td>
<td>-.514</td>
<td>-.514</td>
<td>85</td>
<td>.803</td>
<td>.803</td>
</tr>
<tr>
<td>38</td>
<td>-.653</td>
<td>-.653</td>
<td>86</td>
<td>.687</td>
<td>.687</td>
</tr>
<tr>
<td>39</td>
<td>-.664</td>
<td>-.664</td>
<td>87</td>
<td>.659</td>
<td>.669</td>
</tr>
<tr>
<td>40</td>
<td>-.777</td>
<td>-.777</td>
<td>88</td>
<td>.634</td>
<td>.634</td>
</tr>
<tr>
<td>41</td>
<td>-.782</td>
<td>-.782</td>
<td>89</td>
<td>.519</td>
<td>.519</td>
</tr>
<tr>
<td>42</td>
<td>-.876</td>
<td>-.876</td>
<td>90</td>
<td>.407</td>
<td>.407</td>
</tr>
<tr>
<td>43</td>
<td>-.924</td>
<td>-.924</td>
<td>91</td>
<td>.284</td>
<td>.284</td>
</tr>
<tr>
<td>44</td>
<td>-.909</td>
<td>-.963</td>
<td>92</td>
<td>.236</td>
<td>.236</td>
</tr>
<tr>
<td>45</td>
<td>-.970</td>
<td>-.963</td>
<td>93</td>
<td>.164</td>
<td>.164</td>
</tr>
<tr>
<td>46</td>
<td>-.956</td>
<td>-.963</td>
<td>94</td>
<td>-.019</td>
<td>-.019</td>
</tr>
<tr>
<td>47</td>
<td>-1.034</td>
<td>-1.034</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
procedure prodreversimpleextr(lex, ext, p, d);
  value lex, p;
  integer lex, ext, p;
  array d;
begin
  integer f, g, h, i, j, j1, j2, k, k1, k2, l, l1, l2, m, n;
  Boolean fg, fl, fk, fn, fj;
  n:=3*p;
  h:=n-1;
  f:=p+p;
begin
  real r, t, x, y, y1, y2;
  integer array q[1:p+1], q1, tb, tn, t1[1:p], s, sn[1:n], s1[0:f-1];
  array b, bb[1:n];
  t:=r=d[f-1]=d[p];
  d[f]:=-t;
  l:=lex+2;
  f1:=l+5*lex;
  lex:=if fl .then 1+1 else l;
  fk:=false;
  fj:=ext=1;
  l1:=p;
  k2:=j:=0;
  y:=if fj .then 1.0 else -1.0;
  for i:=1 step 1 until p do
    begin
      q1[i]:=0;
      t1[i]:=f+i;
      tn[i]:=tb[i]=1
    end i;
\( s[0]=s[f-1]=y; \)
\[ \text{for } i=f-3 \text{ step } -2 \text{ until } 1 \text{ do} \]
begin
\( s[i]=y; \)
\( s[i+1]=-y; \)
\( j=j+1; \)
\( r=d[i]=d[p-j]; \)
\( d[i+1]=-x; \)
\text{if } t<x
\text{then } \ t=x
\text{else}
\text{if } x<r
\text{then } r=x
\text{end i;}
\text{if } r>0
\text{then}
begin
\( x=t-r; \)
\( r=\text{if } x<r \text{ then } 0 \text{ else } x+1.0 \)
\text{end r>0}
\text{else}
begin
\( t=t-r; \)
\( r=\text{abs}(r)+t+1.0 \)
\text{end r<0;}
\text{for } i=1 \text{ step } 2 \text{ until } f \text{ do}
begin
\( d[i]=d[i]+r; \)
\( d[i+1]=d[i+1]-r \)
\text{end i;}
\]
extr:
  j:=p+1;
  for i=1 step 1 until n do
    begin
      b[i]:=if i<f then d[i] else .0;
      s[i]:=i+j
    end 1;
  g:=0;
iter:
  if g+1
    then
      begin
        t:=b[2];
        m:=2;
        j:=4;
      end 20: for i:=j step 2 until f do
        begin
          x:=b[1];
          if x<t
            then
              begin
                t:=x;
                m:=1
              end
          end 1
        end g+1
    else
      begin
        t:=b[1];
        m:=1;
j=3;
    go to et20
    end g=11;
    if t<0
    then
    begin
    iter1:
    if f≥m
    then
    begin
    j=m+2;
    k=if j>.5>m then j else 11+1
    end
    f≥m
    else
    if fk
    then
    begin
    k=a[q[g]];
    j=tb[k];
    k=if a1[j+j-1]=1 then k+1 else k-1
    end
    else
    begin
    k=m-f;
    if k=a[q[g]]
    then k=k+1
    end;
    s[m]=k;
    g=g+1;
    q[g]=m;
\[ y_1 = y_2 = 0; \]
\[ t = \text{if } k < 1 \text{ then } -t \text{ else } -0.5 \times t; \]
\[ \text{if } k = 1 \text{ then } \]
\[ \text{begin} \]
\[ b[h+1] = t; \]
\[ b[m] = t; \]
\[ j = m - 1; \]
\[ \text{for } i = 1 \text{ step } 1 \text{ until } j, m+1 \text{ step } 1 \text{ until } b \text{ do} \]
\[ b[i] = b[i] - (\text{if } i < f \wedge i + 2 \leq 5 \times i \text{ then } -2.0 \text{ else if } s[i] < 1 \]
\[ \text{then } 1.0 \text{ else } 0); \]
\[ \text{go to } \text{et1} \]
\[ \text{end } k = 1 \text{+1} \]
\[ \text{else} \]
\[ \text{begin} \]
\[ l = k + k; \]
\[ b[1] = b[1] + t; \]
\[ b[l-1] = b[l-1] - t; \]
\[ j = 2 \times tb[k]; \]
\[ x = t \times s1[j-1]; \]
\[ y = t \times s1[j-2]; \]
\[ \text{if } k = 1 \]
\[ \text{then} \]
\[ \text{begin} \]
\[ \text{if } tn[1] = 0 \]
\[ \text{then} \]
\[ \text{begin} \]
\[ y_1 = b[f+1] = b[f+1] - y; \]
\[ y_2 = b[f+2] = b[f+2] - x; \]
\[ j = \text{if } y_1 < 0 \text{ then } f + 1 \text{ else if } y_2 < 0 \text{ then } f + 2 \text{ else } j \]
\begin{Verbatim}
and \text{tn}[1]=0
else
begin
  \text{y1}=\text{b}[f+1] = \text{b}[f+1] - x;
  if \text{y1} < 0
  then \text{j}=f+1
end
end \text{k}=1
else
  if \text{k}=11
  then
  begin
    if \text{tn}[p]=0
    then
      begin
        \text{y1}=\text{b}[h-1] = \text{b}[h-1] - y;
        \text{y2}=\text{b}[h] = \text{b}[h] - x;
        j=if \text{y1} < 0 \text{ then } h-1 \text{ else if } \text{y2} < 0 \text{ then } h \text{ else } j
      end
    \text{tn}[p]=0
  else
  begin
    \text{y1}=\text{b}[h] = \text{b}[h] - y;
    if \text{y1} < 0
    then \text{j}=h
  end
end
else
begin
  k=t1[k];
  \text{y1}=\text{b}[k] = \text{b}[k] - x;
\end{Verbatim}
\[ y_2 = b[k-1] = b[k-1] - y; \]

\[ j = \text{if } y_1 < 0 \text{ then } k \text{ else if } y_2 < 0 \text{ then } k - 1 \text{ else } j \]

\[ \text{end;} \]

\[ b[m] = t; \]

\[ \text{end } k + 1 \text{+1;} \]

\[ \text{if } y_1 < 0 \]

\[ \text{then} \]

\[ \text{begin} \]

\[ t = y_1; \]

\[ w = j; \]

\[ \text{go to iter1} \]

\[ \text{end } y_1 < 0 \]

\[ \text{else} \]

\[ \text{if } y_2 < 0 \]

\[ \text{then} \]

\[ \text{begin} \]

\[ t = y_2; \]

\[ w = j; \]

\[ \text{go to iter1} \]

\[ \text{end;} \]

\[ \text{go to iter} \]

\[ \text{end } t < 0 \]

\[ \text{else} \]

et1: \[ \text{if } b[h+1] \iva 0 \iva (k2+1eVfrika) \]

\[ \text{then} \]

\[ \text{begin} \]

\[ fn = \text{true;} \]

\[ j_2 = 0.5 \times f; \]

\[ j = q[g]; \]

\[ \text{for } i = g - 1 \text{ step } -1 \text{ until } 1 \text{ do} \]

\[ \]
begin
m=q[i];
k=s[m];
fg=true;
g=t1[k];
k1=k+k-1;
j1=tb[k];
if j1<=1&&(s1[j1+j1-1]!=v1k=1)
  then g=g-1;
if k1<i&&k2<j2
  then
  begin
    if j=k1Vj=g
      then
      begin
        et5:j=m;
        fg=false
      end
    end
  else
    if k=1&&(j=k1Vj=g)
      then go to et5
    else
      if k=j2&&(j=k1Vj=g)
        then go to et5;
  if fg
    then
  else
    if fn
    then
begin
l:=m;
fn:=false
end
else l2:=m
end 1;
x:=b[1];
l:=s[1];
l2:=s[12];
if l>l2
then
begin
i:=1;
l:=l2;
l2:=1
and l>l2;
if f_k
then
begin
k1:=.5x_f;
j1:=b[1];
fn:=s1[j1+j1-1]=1;
fl:=false;
for i:=l2+1 step 1 until k1 do
begin
j:=b[1];
if tn[j]=0
then go to et10
else
if j=q1[j]
then
begin
fl=true;
et11:
if fn
then
begin
if d[j+j-1]<x
then l2=1
end
else
if d[j+j-1]>x
then l2=1
end
else go to et11
end i;
et10:
fl=false;
for i=1-1 step -1 until 1 do
begin
j=tb[i];
if tn[j]=0\&\&fl
then go to et12
else
if j=q1[j]
then
begin
fl=true;
et13:
if fn
then
begin
if d[j+j-1]>x
then \( l = 1 \)
end ...
else
if \( d[j+j-1] < x \)
then \( l = 1 \)
end ...
else go to et13
end i;

et12: for \( i = 1 \) step 1 until 12 do begin
\( j = tb[i] \);
\( bb[j] = x \);
\( tn[j] = 0 \)
end i;
\( h = f - 2 \times (12 - l + 1) \);
\( l1 = .5 \times f \);
\( j = 1 \);
if \( k1 \neq 12 \)
then
for \( i = 1 \) step 1 until \( l1 \) do begin
\( j = j + 1 \)
end \( k1 \neq 12, l \);
\( k = 0 \);
for \( i = 1 \) step 1 until \( p \) do if \( tn[1] \neq 0 \)
then
begin
\( k = k + 2 \);
\[ b[k-1] = d[i+i-1]; \]
\[ b[k] = d[i+i] \]

\textbf{end} \ tn[1] \downarrow 0; \]
\[ l=k=j=0; \]
\[ m=tn[1]; \]
\[ fn=m \downarrow 0; \]
\[ k1=f+1; \]

\textbf{for} \ i=1 \ \textbf{step} \ 1 \ \textbf{until} \ p \ \textbf{do} \]
\[ \textbf{if} \ tn[1]=0 \]
\[ \textbf{then} \ j=j+1 \]
\[ \textbf{else} \]
\[ \textbf{if} \ j \downarrow 0 \]
\[ \textbf{then} \]
\[ \textbf{begin} \]
\[ j=0; \]
\[ k=k+1; \]
\[ h=h+2; \]
\[ g=f+k; \]
\[ l=l+1; \]
\[ t1[1]=\textbf{if} \ fn \ \textbf{then} \ g \ \textbf{else} \ g+1; \]
\[ \textbf{if} \ m=0 \land k=1 \]
\[ \textbf{then} \ b[k1]=s1[i+i-2] \times bb[i-1] \]
\[ \textbf{else} \]
\[ \textbf{begin} \]
\[ \textbf{if} \ fn \]
\[ \textbf{then} \ g=g-1; \]
\[ k2=i+1; \]
\[ t=bb[i-1]; \]
\[ b[g]=s1[k2-4] \times t; \]
\[ b[g-1]=s1[k2-3] \times t \]
\[ \textbf{end} \]
\[ \textbf{end} \]
\[ \textbf{end} \]
\[ \textbf{end} \]
end;
k=k+1
der
else
if k=0
then
begin
k=k+1;
go to et8
der
else
begin
et8:
g=f+k;
l=l+1;
if fn\&g\&k1
then b[g\[-1]=.0
else b[g]=.0;
t1[l]=if fn then g else g+1;
b=h+1;
k=k+1
der
if j=p
then go to et9;
if tn[p]=0
then
begin
j=p-j;
b[h]=s1[j+j\[-1]\times
end tn[p]=0
else h=h-1;
b[h+1]=.0;
j=l1+1;
k=h+1;

for i=1 step 1 until k do
  s[i]=i+j;
  g=0;
  go to iter
end f1
k2=k2+1;
q1[1]=1;
q1[12]=12;
if k2<lex
  then
else
  if f1
  then
    begin
      for i=1 step 1 until n do
        begin
          bb[i]=b[i];
          sn[i]=s[i]
        end 1;
      for i=p step -1 until 1 do
        if q1[i]<>0
          then
            begin
              q1[i]=0;
              q1[p]=p;
            end q1[i]<>0
    end q1[i]<>0
end f1;

et6: \( x = \text{if } f_j \text{ then } -1.0 \text{ else } 1.0; \)
\( l = f - 2; \)
\( \text{if } q_1[1] = 1 \)
\( \text{then} \)
\( \text{begin} \)
\( s_1[1] = x; \)
\( fn = \text{true} \)
\( \text{end} \)
\( q_1[1] = 1 \)
\( \text{else} \)
\( s_1[1] = -x; \)
\( s_1[l] = \text{if } q_1[p] = p \text{ then } -x \text{ else } x; \)
\( \text{for } i = 3 \text{ step } 2 \text{ until } 1 \text{ do} \)
\( \text{begin} \)
\( j = i + 2; \)
\( \text{if } q_1[j] \neq j \wedge fn \)
\( \text{then} \)
\( \text{begin} \)
\( s_1[i] = x; \)
\( s_1[i - 1] = -x \)
\( \text{end} \)
\( q_1[j] \neq j \wedge fn \)
\( \text{else} \)
\( \text{if } q_1[j] \neq j \)
\( \text{then} \)
\( \text{begin} \)
\( s[i] = -x; \)
\( s_1[i - 1] = x \)
\( \text{end} \)
\( \text{else} \)
\( \text{if } q_1[j] = j \wedge fn \)
\( \text{then} \)
begin
\( s_1[i] = s_1[i-1] = -x; \)
\( fn = \text{false} \)
end
else
begin
\( s_1[i] = s_1[i-1] = x; \)
\( fn = \text{true} \)
end
end i;
go to extr
end b[n+1] \( \Rightarrow \) O \( \langle k2 \rangle \text{lex} Vf_k \);
if fk
then
begin
et14: for i = 1 step 1 until l1 do
begin
j = tb[i];
bb[j] = d[j+j-1].
end i;
et9: for i = 1 step 1 until p do
\( d[i] = bb[i] - x; \)
go to et7
end fk;
if b[n] < bb[n] \( \wedge f_l \)
then
for i = 1 step 1 until n do
begin
b[i] = bb[i];
s[i] = sn[i]
\[ \text{and } b[n] < b[n] \land f1,1; \]

\[ \text{if } b[n] = 0 \]

\[ \text{then go to et14 } \]

\[ \text{else} \]

\[ \text{begin} \]

\[ f_b = \text{true}; \]

\[ \text{go to et1} \]

\[ \text{and } b[n] \neq 0; \]

\[ \text{et7:} \]

\[ d[p+p] = -b[n] \]

\[ \text{and block} \]

\[ \text{and prodrevsimplextr} \]

References


INSTITUTE OF MATHEMATICS AND PHYSICS
ACADEMY OF TECHNOLOGY AND AGRICULTURE
BYDGOSZCZ, POLAND

Received on 14. 12. 1976

F. PANKOWSKI (Bydgoszcz)

APROKSYMACJA JEDNOSTAJNA FUNKCJI EMPIRYCZNYCH,
MONOTONICZNYCH I O DANEJ LICZBIE EKSTREMÓW

STRESZCZENIE

Procedura prodrevsimple rozwiązuje następujące zadanie:
Dla danego ciągu liczbowego \( \{y_1, y_2, \ldots, y_p\} \) wyznaczyć taki ciąg \( \{z_1, z_2, \ldots, z_p\} \), który

(i) realizuje minimum funkcji \( f = \max_{1 \leq k \leq p} |y_k - z_k| \),
(ii) jest monotoniczny.
Dane:

\( ext \) — liczba całkowita o wartości 0, gdy rozwiązujemy zadanie przy założeniu, że badana funkcja jest nierosnąca, oraz 1, gdy zakładamy, że badana funkcja jest niemalejąca;

\( p \) — liczba danych \( y_1, y_2, \ldots, y_p \);

\( d[1 : 3 \times p] \) — tablica danych \( y_1, y_2, \ldots, y_p \) (\( d[1] = y_1, d[2] = y_2, \ldots, d[p] = y_p \)).

Wyniki:

\( d[1 : 3 \times p] \) — tablica wyników \( z_1, z_2, \ldots, z_p \) (\( z_1 = d[1], z_2 = d[2], \ldots, z_p = d[p] \));

\( d[2 \times p] \) — błąd aproksymacji jednostajnej.

Procedura \( \text{prodrexi} \) rozwiązuje zadanie takie jak procedura \( \text{prodrexsimple} \), przy czym punkt (ii) zamieniony jest następującym:

(iii) ma daną liczbę ekstremów.

Dane:

\( lex \) — liczba ekstremów aproksymowanej funkcji;

\( ext \) — liczba całkowita o wartości 0, gdy pierwsze ekstremum jest minimum, oraz 1, gdy pierwsze ekstremum jest maksimum.

Pozostałe dane i wyniki są takie jak w procedurze \( \text{prodrexsimple} \).