ZASTOSOWANIA MATEMATYKI
APPLICATIONES MATHEMATICAE
XVI, 3 (1979)

ALGORITHM 65

R. SMARZEWSKI and H. MALINOWSKI (Lublin)

DETERMINATION OF THE SOLUTION OF AN ABEL INTEGRAL EQUATION

1. Procedure declaration. The procedure *inteqAbel* determines the approximation $f_\Delta(s)$ of the solution $f(s)$ of the integral equation

$$g(t) = 2\int_t^R \frac{sf(s)}{\sqrt{s^2-t^2}}\,ds, \qquad t\in[0,R]. \tag{1}$$

This approximation $f_\Delta(s)$ is defined by

$$f_\Delta(s) = -\frac{1}{\pi}\int_s^R \frac{g'_\Delta(t)}{\sqrt{t^2-s^2}}\,dt, \qquad s\in[0,R], \tag{2}$$

where g_Δ is a spline function of degree $m = 2l-1$ $(1 < l \leqslant n)$ interpolating the function g at the nodes t_i, $0 = t_1 < t_2 < \ldots < t_n = R$, and given by

$$g_\Delta(t) = \sum_{i=0}^{m} \alpha_i t^i + \sum_{j=1}^{n} \beta_j \theta(t,t_j)(t-t_j)^m \tag{3}$$

with

$$\theta(t,t_j) = \begin{cases} 0 & \text{if } t < t_j, \\ 1 & \text{if } t \geqslant t_j. \end{cases}$$

Data:

s — an arbitrary fixed point from the interval $[0, R]$;
m — degree of the spline function (3);
n — number of nodes of the spline function g_Δ;
R — upper limit of integrals in (1) and (2);
alfa[*0*:*m*], *beta*[*1*:*n*] — arrays of coefficients of g_Δ;
T[*1*:*n*] — array of nodes of g_Δ.

Result:

f — approximate value of $f(s)$ defined by (1).

Other parameters:

E — label (outside of the body of the procedure *inteqAbel*) to which a jump is made when either $s < 0$ or $s > R$ or $s = 0$ and $\alpha_1 \neq 0$.

Remark. Note that in the last case we have $f_\Delta(0) = \infty$ (see [2]).

```
procedure inteqAbel(s,m,n,R,alfa,beta,T,f,E);
 value s,m,n,R;
 integer m,n;
 real s,R,f;
 array alfa,beta,T;
 label E;
 begin
  integer i,i1,ip,j,m1,m2,mm,si,si1,sl,sm,spl,spm,v;
  real r,r1,r2,R2,Rp,Rsq,s1,s2,sp,sq,sqtj,sum,sum1,t2,tj,tp;
  Boolean B;
  if s<.0Vs>R
   then go to E;
  m1:=m-1;
  n:=n-1;
  s2:=s×s;
  R2:=R×R;
  sq:=sqrt(R2-s2);
  Rsq:=R+sq;
  if s=.0
   then
   begin
    if alfa[1]≠.0
     then go to E;
    sum:=.0;
    Rp:=1.0;
    for i:=2 step 1 until m do
     begin
      Rp:=Rp×R;
      sum:=sum+alfa[i]×i×Rp/(i-1)
     end i;
```

```
    f:=beta[1]×Rp/m1;
    j:=2;
    go to next
  end s=.0;
r:=ln(Rsq/s);
sum:=alfa[1]×r+2.0×alfa[2]×sq;
f:=.0;
if R=s
  then go to fin;
i1:=1;
for i:=4 step 2 until m1 do
  begin
    i1:=i1+1;
    sl:=sm:=1;
    sum1:=.0;
    sp:=1.0;
    mm:=i-1;
    Rp:=sq↑mm;
    for v:=1 step 1 until i1 do
     begin
       sum1:=sum1+sl×Rp×sp/(sm×mm);
       sp:=sp×s2;
       Rp:=Rp/sq;
       mm:=mm-2;
       sl:=sl×(i1-v);
       sm:=sm×v
     end v;
    sum:=sum+i×alfa[i]×sum1
  end i;
i1:=0;
```

```
  for i:=3 step 2 until m do
   begin
    i1:=i1+1;
    sp:=s2;
    sl:=1;
    sm:=i-1;
    ip:=i-2;
    Rp:=R↑ip;
    sum1:=Rp/sm;
    for v:=2 step 1 until i1 do
     begin
      Rp:=Rp/R2;
      sl:=sl×ip;
      sm:=sm×(ip-1);
      ip:=ip-2;
      sum1:=sum1+sl×sp×Rp/sm;
      sp:=sp×s2
     end v;
    sum:=sum+i×alfa[i]×(sq×sum1+sl×sp×r/sm)
   end i;
  j:=1;
next:
  m2:=m1-1;
  for j:=j step 1 until n do
   begin
    tj:=T[j];
    B:=s<tj;
    tp:=tj↑m1;
    t2:=tj×tj;
    if B
```

```
 then sqtj:=sqrt(t2-s2);
si:=si1:=sl:=sm:=1;
i1:=0;
sp:=1.0;
sum1:=.0;
for i:=0 step 2 until m1 do
 begin
  r2:=si×sp/si1;
  si:=si×(i+1);
  si1:=si1×(i+2);
  sp:=sp×s2;
  s1:=.0;
  Rp:=1.0;
  spl:=1;
  spm:=i;
  for v:=1 step 1 until i1 do
   begin
    ip:=i-2×v+1;
    r1:=spl×Rp/spm;
    Rp:=Rp×s2;
    spl:=spl×ip;
    spm:=sp×(ip-1);
    if B
     then r1:=r1×(sq×R↑ip-sqtj×tj↑ip);
    s1:=s1+r1
   end v;
  s1:=if B then s1+r2×ln(Rsq/(tj+sqtj)) else-sq×s1+r2×r;
  sum1:=sum1+sl/sm×tp×s1;
  sl:=sl×(m1-i)×(m2-i);
  sm:=sm×(i+1)×(i+2);
```

```
    i1:=i1+1;
    if tj≠.0
     then tp:=tp/t2
   end i;
  tj:=-tj;
  tp:=tj↑m2;
  sl:=m1;
  sm:=1;
  i1:=0;
  for i:=1 step 2 until m2 do
   begin
    s1:=.0;
    r2:=1.0;
    Rp:=sq↑i;
    spl:=spm:=1;
    mm:=i;
    i1:=i1+1;
    for v:=1 step 1 until i1 do
     begin
      r1:=spl/spm×r2/mm;
      s1:=s1+r1×(if B then Rp-sqtj↑(i-2×v+2) else Rp);
      spl:=spl×(i1-v);
      spm:=spm×v;
      mm:=mm-2;
      r2:=r2×s2;
      Rp:=Rp/sq
     end v;
    sum1:=sum1+sl×tp×s1/sm;
    sl:=sl×(m1-i)×(m2-i);
    sm:=sm×(i+1)×(i+2);
```

```
        if tj≠.0
          then tp:=tp/t2
        end i;
      f:=f+beta[j]×sum1
    end j;
  f:=-.3183098862×(sum+m×f);
fin:
end inteqAbel
```

2. Method used. The method presented in [2] has been used. Additionally, the numerically stable method from [1] is proposed to determine α_i and β_j in (3).

3. Certification. The procedure *inteqAbel* has been tested on the Odra 1204 computer. The obtained results and the time of calculations have been given in [2].

References

[1] H. Malinowski and R. Smarzewski, *A numerical method for solving the Abel integral equation*, Zastosow. Matem. 16 (1978), p. 275-281.

[2] R. Smarzewski and H. Malinowski, *On the numerical solution of an Abel integral equation*, this fascicle, p. 497-503.

DEPARTMENT OF NUMERICAL METHODS
M. CURIE-SKŁODOWSKA UNIVERSITY
20-031 LUBLIN

Received on 22. 10. 1976

ALGORYTM 65

R. SMARZEWSKI i H. MALINOWSKI (Lublin)

WYZNACZANIE ROZWIĄZANIA PEWNEGO CAŁKOWEGO RÓWNANIA ABELA

STRESZCZENIE

Procedura *inteqAbel* wyznacza aproksymację $f_\Delta(s)$ rozwiązania $f(s)$ równania całkowego (1). Aproksymacja $f_\Delta(s)$ określona jest przez (2), gdzie g_Δ jest funkcją sklejaną stopnia $m = 2l-1$ $(1 < l < n)$, interpolującą funkcję g w węzłach t_i, $0 = t_1 < t_2 < \ldots < t_n = R$, i daną wzorem (3).

Dane:

s — dowolny ustalony punkt z przedziału $[0, R]$;
m — stopień funkcji sklejanej (3);
n — liczba węzłów funkcji sklejanej g_Δ;
R — górna granica całek w (1) i (2);
alfa $[0:m]$, *beta* $[1:n]$ — tablice współczynników g_Δ;
$T[1:n]$ — tablica węzłów g_Δ.

Wynik:

f — przybliżona wartość $f_\Delta(s)$ dla $f(s)$ określonego przez (1).

Inne parametry:

E — etykieta (poza treścią procedury *inteqAbel*), do której następuje skok, gdy $s < 0$ albo $s > R$, albo $s = 0$ i $\alpha_1 \neq 0$.

Uwaga. W ostatnim przypadku mamy $f_\Delta(0) = \infty$ [2].

W procedurze *inteqAbel* zastosowano metodę szczegółowo opisaną w [2]. Do wyznaczania współczynników α_i oraz β_j w (3) autorzy proponują numerycznie stabilną metodę z pracy [1]. Wyniki obliczeń, wykonanych na maszynie Odra 1204, zamieszczone w pracy [2], wykazały poprawność algorytmu.