Ann Haegemans; Robert Piessens

Algorithms. 41. FOURIER. Computation of Fourier-transform integrals

Aplikace matematiky, Vol. 21 (1976), No. 3, 229–236

Persistent URL: http://dml.cz/dmlcz/103641

ALGORITMY

41. FOURIER

COMPUTATION OF FOURIER-TRANSFORM INTEGRALS

ANN HAEGEMANS* and ROBERT PIESSENS

Applied Mathematics and Programming Division University of Leuven,
Celestijnenlaan 200 A, B-3030 Heverlee, Belgium

This procedure computes Fourier-transform integrals

$$(1)\qquad C(\omega) = \int_0^\infty \cos \omega t\, f(t)\, dt$$

and

$$(2)\qquad S(\omega) = \int_0^\infty \sin \omega t\, f(t)\, dt$$

The integral $C(\omega)$ is written as

$$(3)\qquad C(\omega) = \frac{1}{\omega}\int_0^{\pi/2} \cos x\, f\left(\frac{x}{\omega}\right) dx + \frac{1}{\omega}\sum_{k=1}^{\infty}(-1)^k \int_{-\pi/2}^{\pi/2} \cos x\, f\left(\frac{x+k\pi}{\omega}\right) dx$$

The first integral on the right side of (3) is evaluated using Romberg integration and a number of the integrals in the summation are computed using Gaussian quadrature formulas with $\cos x$ as weight function. The sum is then estimated by the ε-algorithm of Wynn. The modification of this method for the computation of $S(\omega)$ is straightforward. The procedure determines automatically the required number of terms in (3) and the order N of the Gaussian formulas ($N \leqq 32$) in order to have a prescribed accuracy. The abscissas and weights of the quadrature formulas are stored in the program.

The user has only to insert

(a) the value of ω
(b) a procedure for computing $f(t)$ ($t \geqq 0$)
(c) a tolerance ε, indicating the requested absolute accuracy

*) Stagiair of the N. F. W. O. (Belgium).

(d) a *boolean* parameter which must be *true* if equation (1) is to be calculated and *false* if equation (2) is to be calculated.

```
real procedure fourierint (f, omega, eps, cosine, ier);
value omega, eps; integer ier; real omega, eps;
real procedure f; boolean cosine;
comment this function returns the value of the integral from zero to infinity of
        f(x) × cos (w × x). The epsilon algorithm is used to accelerate the con-
        vergence.
        inputparameters
          f      ... function
          omega ... pulsation
          eps    ... desired absolute accuracy
          cosine ... = true if cosine transform is to be computed
                     = false if sine transform is to be computed
        outputparameters
          ier ... = k when in k half periods the integral cannot be computed
                  with the desired accuracy
                  = 1000 + k when in k half periods the integral cannot be comput-
                    ed with the desired accuracy and when there was no convergence
                    after 25 steps in the epsilon algorithm;
begin real array s[0 : 25, 1 : 3], x2[1 : 83], w2[1 : 83];
  real pi, a, z; integer sgn, sgnw, ii, i, im1, im2, j, j1, k;
  real procedure oscin1;
    begin real s, p, h, a1, a2; array t[0 : 12];
      integer n, j, i, k;
      comment Romberg integration;
      n := 1;
      t[0] := 0.25 × pi × f(0)/z;
      a1 := t[0];
      for k := 1 step 1 until 12 do
        begin n := 2 × n; h := 0.5 × pi/(n × z);
            p := 4; s := 0;
          comment evaluation of trapezoidal rule;
          for i := 1 step 2 until n do
            s := s + f(i × h) × cos(z × i × h);
          t[k] := t[k − 1]/2 + s × h;
          comment Romberg extrapolation;
          for j := k − 1 step − 1 until 0 do
            begin t[j] := (p × t[j + 1] − t[j])/(p − 1);
                p := 4 × p
            end j;
```

```
            if k < 2 then a2 := t[0]
            else if abs(a1 - a2) < 100 × eps ∧ abs(a2 - t[0]) < eps
            then go to OUT
            else begin a1 := a2;
                       a2 := t[0]
          end;
      end k;
      ier := 1;
OUT: oscin1 := t[0]
end of oscin1;
real procedure oscin2(j);
value j; integer j;
comment Gaussian quadrature;
begin real sum1, sum2; integer l, n, i;
   l := 1; sum1 := 1₁₀50;
   for n := 1 step 1 until 10, 12 step 4
   until 16 do
      begin
         sum2 := 0;
         for i := 1 step 1 until n do
         begin
            if cosine then sum2 := sum2 + w2[l] ×
              (f((x2[l] + (j - 1) × pi)/z) +
              f((- x2[l] + (j - 1) × pi)/z))
            else sum2 := sum2 + w2[l] ×
            (f((x2[l] + (j - 0.5) × pi)/z) +
              f((- x2[l] + (j - 0.5) × pi)/z));
              l := l + 1
         end i;
         comment test on accuracy;
         if (abs(sum1 - sum2) < eps) then go to FIN
         else sum1 := sum2
      end n;
      ier := ier + 1;
FIN: oscin2 := sum2/z;
end oscin2;

x2[1]  := 1.0000000000000000; x2[2]  := 0.4392874668600151;
w2[1]  := 0.6836673900899030; w2[2]  := 0.7759293818723798;
x2[3]  := 1.1906765638948560; x2[4]  := 0.3238521142128551;
w2[3]  := 0.2240706181276202; w2[4]  := 0.6058137001229013;
```

$x2[5] := 0.9197906655171356$; $x2[6] := 1.3639113020774860$;

$w2[5] := 0.3247985513772208$; $w2[6] := 0.0693877484998778$;

$x2[7] := 0.2564965074162312$; $x2[8] := 0.7434686478754924$;

$w2[7] := 0.4919957966032009$; $w2[8] := 0.3362644778528046$;

$x2[9] := 1.1537256454567280$; $x2[10] := 1.4414905401823580$;

$w2[9] := 0.1442040920302275$; $w2[10] := 0.0275356335137670$;

$x2[11] := 0.2123428870690829$; $x2[12] := 0.6221414570511029$;

$w2[11] := 0.4127123718856410$; $w2[12] := 0.3189730757362244$;

$x2[13] := 0.9878852856245157$; $x2[14] := 1.2824218392614150$;

$w2[13] := 0.1840793456376885$; $w2[14] := 0.0713116318694723$;

$x2[15] := 1.4825294561888940$; $x2[16] := 0.1811604682921093$;

$w2[15] := 0.0129235748709738$; $w2[16] := 0.3548619551685681$;

$x2[17] := 0.5341801247535697$; $x2[18] := 0.8595998627218704$;

$w2[17] := 0.2945380284952109$; $w2[18] := 0.1995586475169654$;

$x2[19] := 1.1400861905028520$; $x2[20] := 1.3600784903817600$;

$w2[19] := 0.1054805682651703$; $w2[20] := 0.0387356701110729$;

$x2[21] := 1.5067752445913270$; $x2[22] := 0.1579643816827197$;

$w2[21] := 0.0068251304430124$; $w2[22] := 0.3109730008916469$;

$x2[23] := 0.4676982277627329$; $x2[24] := 0.7590124026467929$;

$w2[23] := 0.2700416599568707$; $w2[24] := 0.2017941600465789$;

$x2[25] := 1.0201945939238280$; $x2[26] := 1.2404404819041370$;

$w2[25] := 0.1268326827931822$; $w2[26] := 0.0637384718429930$;

$x2[27] := 1.4103152290051090$; $x2[28] := 1.5222642294265360$;

$w2[27] := 0.0226888332160192$; $w2[28] := 0.0039311912527090$;

$x2[29] := 0.1400344442469677$; $x2[30] := 0.4157719767341894$;

$w2[29] := 0.2766109576482605$; $w2[30] := 0.2476363209463552$;

$x2[31] := 0.6786110809756055$; $x2[32] := 0.9202778620663710$;

$w2[31] := 0.1974114887025346$; $w2[32] := 0.1383419552695127$;

$x2[33] := 1.1330068786005000$; $x2[34] := 1.3097818904452940$;

$w2[33] := 0.0830266475732177$; $w2[34] := 0.0404378939465037$;

$x2[35] := 1.4446014873666510$; $x2[36] := 1.5327507132362300$;

$w2[35] := 0.0141152681568540$; $w2[36] := 0.0024194677567616$;

$x2[37] := 0.1257600093435378;$ $x2[38] := 0.3741337804491100;$

$w2[37] := 0.2490124783166383;$ $w2[38] := 0.2277838163923600;$

$x2[39] := 0.6131272013222985;$ $x2[40] := 0.8366838494891686;$

$w2[39] := 0.1899572674803385;$ $w2[40] := 0.1432902231210859;$

$x2[41] := 1.0390514169347030;$ $x2[42] := 1.2149186859960250;$

$w2[41] := 0.0963162571178597;$ $w2[42] := 0.0560897522251605;$

$x2[43] := 1.3595638434785010;$ $x2[44] := 1.4690102441122380;$

$w2[43] := 0.0267613742135268;$ $w2[44] := 0.0092199875076379;$

$x2[45] := 1.5401756452752600;$ $x2[46] := 0.1141265351397092;$

$w2[45] := 0.0015688436253925;$ $w2[46] := 0.2263773256347921;$

$x2[47] := 0.3400227092105425;$ $x2[48] := 0.5588854287853609;$

$w2[47] := 0.2103759285736437;$ $w2[48] := 0.1812793354304047;$

$x2[49] := 0.7661509164888158;$ $x2[50] := 0.9574468180865112;$

$w2[49] := 0.1441093214522280;$ $w2[50] := 0.1047137538563544;$

$x2[51] := 1.1286754879087500;$ $x2[52] := 1.2761025302615130;$

$w2[51] := 0.0684209743603314;$ $w2[52] := 0.0390333646242024;$

$x2[53] := 1.3964496146801940;$ $x2[54] := 1.4869880241013160;$

$w2[53] := 0.0183607593751510;$ $w2[54] := 0.0062681947298298;$

$x2[55] := 1.5456233315613980;$ $x2[56] := 0.0963084478801410;$

$w2[55] := 0.0010610419630624;$ $w2[56] := 0.1914882094764121;$

$x2[57] := 0.2875039148019434;$ $x2[58] := 0.4744518972742523;$

$w2[57] := 0.1817599290152465;$ $w2[58] := 0.1635809306224327;$

$x2[59] := 0.6543769826862947;$ $x2[60] := 0.8245889260330872;$

$w2[59] := 0.1392528630385753;$ $w2[60] := 0.1116634500044801;$

$x2[61] := 0.9825182190263299;$ $x2[62] := 1.1257529026114140;$

$w2[61] := 0.0837771210970006;$ $w2[62] := 0.0581732191016194;$

$x2[63] := 1.2520765599390550;$ $x2[64] := 1.3595070401753290;$

$w2[63] := 0.0367203461329272;$ $w2[64] := 0.0204245886202449;$

$x2[65] := 1.4463349271495560;$ $x2[66] := 1.5111599536054150;$

$w2[65] := 0.0094386847436585;$ $w2[66] := 0.0031851139632458;$

$x2[67] := 1.5529206113728880;$ $x2[68] := 0.0733917083744216;$

$w2[67] := 0.0005355441841570;$ $w2[68] := 0.1462831701019649;$

```
x2[69] := 0.2195432169219730; x2[70] := 0.3638036135469709;
w2[69] := 0.1419245828750883; w2[70] := 0.1335454455634596;
x2[71] := 0.5049275494140530; x2[72] := 0.6416928214471680;
w2[71] := 0.1217818246542211; w2[72] := 0.1074943692922972;
x2[73] := 0.7729097379327027; x2[74] := 0.8974305746698114;
w2[73] := 0.0916731299075996; w2[74] := 0.0753350045262599;
x2[75] := 1.0141591218860860; x2[76] := 1.1220603001321730;
w2[75] := 0.0594279652953817; w2[76] := 0.0447534582942820;
x2[77] := 1.2201697939902740; x2[78] := 1.3076036151419680;
w2[77] := 0.0319139368609826; w2[78] := 0.0212875259455956;
x2[79] := 1.3835674612763600; x2[80] := 1.4473656842873490;
w2[79] := 0.0130274103587377; w2[80] := 0.0070804715378565;
x2[81] := 1.4984096108013310; x2[82] := 1.5362247792530200;
w2[81] := 0.0032183097393748; w2[82] := 0.0010740035337058;
x2[83] := 1.5604552434125810; w2[83] := 0.0001793915131927;
sgn := -1; ii := 1; ier := 0;
a := 0; sgnw := 1; z := omega;
s[0, 1] := 1₁₀50; pi = 3.141592653589793;
comment test for the sign of the pulsation;
if z < 0 then begin z := - z; sgnw := -1
            end;
comment test to see if the cosine transform or the sine transform is desired;
if cosine then begin a := oscin1;
                   sgnw := 1; ii := ii + 1;
                   s[1, 2] := - oscin2 (ii);
                   sgn := - sgn; ii := ii + 1
            end
        else begin s[1, 2] := oscin2 (ii);
                   ii := ii + 1
            end;
s[1, 3] := s[1, 2] + oscin2 (ii) × sgn;
comment epsilon algorithm;
for i := 2 step 1 until 25 do
  begin for k := 2 step 1 until 3 do
    begin s[1, 1] := s[1, 2]; s[1, 2] := s[1, 3];
      ii := ii + 1; sgn := - sgn;
      s[1, 3] := [1, 3] + oscin2 (ii) × sgn;
```

```
        for j := 1 step 1 until i - 2 do
        begin j1 := j + 1; s[j1, 1] := s[j1, 2];
          s[j1, 2] := s[j1, 3];
          s[j1, 3] := if abs(s[j, 2] - s[j, 1]) < 1₁₀ - 30
           ∨ abs(s[j, 2] - s[j, 3]) < 1₁₀ - 30
           ∨ abs(s[j, 2] - s[j - 1, 1]) < 1₁₀ - 30
          then s[j, 2]
          else s[j, 2] - 1/(1/(s[j, 2] - s[j, 1])
             + 1/(s[j, 2] - s[j, 3]) - 1/(s[j, 2] - s[j - 1, 1]))
        end;
        j1 := i - 1;
        s[i, k] := if abs(s[j1, 2] - s[j1, 1]) < 1₁₀ - 30
         ∨ abs(s[j1, 2] - s[j1, 3]) < 1₁₀ - 30
         ∨ abs(s[j1, 2] - s[j1 - 1, 1]) < 1₁₀ - 30
        then s[j1, 2]
        else s[j1, 2] - 1/(1/(s[j1, 2] - s[j1, 1])
           + 1/(s[j1, 2] - s[j1, 3]) - 1/(s[j1, 2] - s[j1 - 1, 1]))
      end;
    if abs(s[i, 2] - s[i, 3]) < eps ∧ abs(s[i - 1, 2] -
        s[i - 1, 3]) < eps then go to OUT
    end;
    ier := ier + 1000;
OUT: fourierint := (s[i, 2] + a) × sgnw
end of fourierint
```

Examples. We tested this procedure for a large number of functions $f(t)$ and various values of ω and ε. For 97% of the testcases reliable results were obtained.

Table 1

$$\int_0^\infty \frac{1}{x^2 + a^2} \cos \omega x dx$$

a	ω	$\varepsilon = 10^{-5}$		$\varepsilon = 10^{-10}$	
		ε_{act}	N	ε_{act}	N
0·125	0·5	$0{\cdot}27 \times 10^{-6}$	608	$0{\cdot}11 \times 10^{-10}$	4302
	8·0	$0{\cdot}15 \times 10^{-6}$	160	$0{\cdot}41 \times 10^{-11}$	520
	256·0	$0{\cdot}13 \times 10^{-6}$	128	$0{\cdot}41 \times 10^{-11}$	224
2·0	0·5	$0{\cdot}68 \times 10^{-6}$	112	$0{\cdot}87 \times 10^{-11}$	322
	8·0	$0{\cdot}40 \times 10^{-8}$	128	$0{\cdot}65 \times 10^{-11}$	200
	256·0	$0{\cdot}82 \times 10^{-8}$	76	$0{\cdot}16 \times 10^{-14}$	112

Moreover, the algorithm is found to be very efficient. The results of two test functions are listed in table 1 and 2, where we have used the notation N to denote the number of function evaluations, ε the requested absolute accuracy and ε_{act} the actual absolute error.

Table 2

$$\int_0^\infty \frac{x}{x^2 + a^2} \sin \omega x dx$$

a	ω	$\varepsilon = 10^{-5}$		$\varepsilon = 10^{-10}$	
		ε_{act}	N	ε_{act}	N
0·125	0·5	$0{\cdot}21 \times 10^{-6}$	274	$0{\cdot}45 \times 10^{-6}$ (ier = 1)	362
	8·0	$0{\cdot}20 \times 10^{-6}$	138	$0{\cdot}35 \times 10^{-11}$	304
	256·0	$0{\cdot}91 \times 10^{-7}$	96	$0{\cdot}54 \times 10^{-10}$	168
2·0	0·5	$0{\cdot}20 \times 10^{-6}$	138	$0{\cdot}22 \times 10^{-11}$	268
	8·0	$0{\cdot}58 \times 10^{-6}$	96	$0{\cdot}16 \times 10^{-11}$	168
	256·0	$0{\cdot}27 \times 10^{-12}$	72	$0{\cdot}56 \times 10^{-14}$	96

Acknowledgement. This research is supported by the F.K.F.O., Belgium under grant *n*° 10174.