

# Some properties of the corrugated elliptical waveguide. Part II

## **Citation for published version (APA):**

Thurlings, L. F. G. (1975). *Some properties of the corrugated elliptical waveguide. Part II*. Technische Hogeschool Eindhoven.

## **Document status and date:**

Published: 01/01/1975

## **Document Version:**

Publisher's PDF, also known as Version of Record (includes final page, issue and volume numbers)

## **Please check the document version of this publication:**

- A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
- The final author version and the galley proof are versions of the publication after peer review.
- The final published version features the final layout of the paper including the volume, issue and page numbers.

[Link to publication](#)

## **General rights**

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license above, please follow below link for the End User Agreement:

[www.tue.nl/taverne](http://www.tue.nl/taverne)

## **Take down policy**

If you believe that this document breaches copyright please contact us at:

[openaccess@tue.nl](mailto:openaccess@tue.nl)

providing details and we will investigate your claim.

EINDHOVEN UNIVERSITY OF TECHNOLOGY

DEPARTMENT OF ELECTRICAL ENGINEERING

Some properties of the corrugated  
elliptical waveguide

by

ir. L.F.G. Thurlings

Part II

ET-6-1975

August 1975

Contents

Chapter

1	Introduction	3
2	ZEROPPOINTDD2	4
3	EIGENVALUETEST	8
4	DEPTHODD1/DEPTHODD2	12
5	ABCD/ABEF/PLOTLAMBDAQ	18
6	RADABCD/RADABEF	34

## Chapter 1

### Introduction

The programs used in this report are:

ZEROPOINTODD ( see [2] )

ZEROPOINTODD2

EIGENVALUEODD ( see [2] )

DEPTHODD1

DEPTHODD2

ABCD

AEEF

PLOTLAMBDAQ ( constructed by I.C. Ongers )

RADABCD

RADABEF

With ZEROPOINTODD and ZEROPOINTODD2 we can determine the cut-off frequencies and with EIGENVALUEODD (and EIGENVALUETEST) the phase-factor for the anisotropic waveguide. The programs DEPTHODD1 and DEPTHODD2 give the depth of the grooves of the corrugated waveguide for the even and the odd modes respectively, the programs ABCD and AEEF the phasefactor. With PLOTLAMBDAQ we are able to plot the phase-factor for the corrugated as well as for the anisotropic waveguide. RADABCD and RADABEF compute the radiationpatterns in the E- and H-plane for the even and odd modes respectively.

To determine the properties of the corrugated waveguide numerically is rather expensive because most of the programs take a lot of processor-time. The calculation of one radiationpattern takes about eight minutes, the determination of the curve of the dominant-mode phase-factor about two hours. For the construction of antennas for experimen-

tal work we only need to ascertain the depth of the grooves, which may be done with the programs DEPTHODD1 and DEPTHODD2. If we want to be sure that the given frequency is above the cut-off frequency of the dominant mode we have to compute the cut-off frequency with ZEROPOINTODD2.

We shall now describe the programs in detail.

## Chapter 2

### ZEROPOINTODD2

(for ZEROPOINTODD see [2], p93)

This program calculates the cut-off frequency of the  $e^{EH_{nm}}$  and  $\circ^{EH_{nm}}$  modes, for  $n = \text{odd}$ .

These cut-off frequencies are determined by the zeros of the Mathieu-functions:

$$e^{EH_{nm}} \rightarrow S_{e_n}(\xi_2) = 0 \quad (1.1)$$

$$\circ^{EH_{nm}} \rightarrow C_{e_n}(\xi_2) = 0 \quad (1.2)$$

First of all, the program computes in a given interval of  $q$  the Mathieufunction and stores these data in an array. Then it searches for a change of sign and if it finds one it computes the zeros. Except for the Mathieufunctions, the program is in principle the same as ZEROPOINTODD [2].

MACHEPS

$1^{-36}$ , needed in the eigenvalue and fouriercoefficients of the Mathieufunctions

L

eigenvalue of the Mathieufunction

L1

- not in use -

L2

- not in use -

NMIN

lowest order of the Mathieufunction

NMAX

highest order of the Mathieufunction

IMIN	}	q-interval
IMAX		
ISTEP		
DELTAQ		
TELLER		number of eccentricities
KSI		$\xi$
Q		q
EXC		eccentricity
A [0:IMAX]		array for q
B [0:IMAX]		array for $C_{2n+1}(\xi, q)$ c.q. $Se_{2n+1}(\xi, q)$
N		order of the Mathieu function
ZEROCCE2N1(TWON,KSI,Q)		procedure to compute $C_{2n+1}(\xi, q)$
ZEROSSE2N1(TWON,KSI,Q)		procedure to compute $Se_{2n+1}(\xi, q)$
SQRT( 4Q )		kh
2SQRT( Q ) COSH( KSI )		ka

ZERO POINT ZERO TWO

```

'BEGIN'
'FILE' OUTPUTS

'REAL' MACHEPS, L1, L2 ;
'INTEGER' IMIN, IMAX, ISTEP, DELTAG, NMIND, NMAX, K
MACHEPS:= 2**(-36);
NMIND:= 1;
NMAX:= 5;
IMIN:= 0;
IMAX:=300;
ISTEP:= 5;
DELTAG:= 10;
'BEGIN'

```

## EXTERNAIS

```

'REAL''PROCEDURE'ZERUCCE2N1    (T0K0, KSI, G);
'VALUE'T0K0, KSI, G; 'INTEGER'T0K0; 'REAL' KSI; G;
'BEGIN''REAL'   NURM,M1,N1,SUM;

  'ARRAY'F[0:20], JLG[1420];
  L1= EIGENVALLE2N1(T0K0,0);
  FOURIERCOEFF2N1(T0K0, G, L1, 20, F, NURM);
  JAPLUSN(2*SQRT(G)*SINH(KSI), 0, 42, 10, J);
  ZERUCCE2N1    := CCE2N1(NURM,20,F,J)/SINH(KSI)*COSH(KSI);
'END'ZERUCCE2N1;

```

```

%.....%
%.....%
'REAL''PROCEDURE'ZEROSSE2N1 (THCR, KSI, Q);
'VALUE'THUN, KSI, Q; 'INTEGER' THUN; 'REAL' KSI, Q;
'BEGIN' 'REAL' NORM,M1,N1; %
%
'ARRAY'F[0:20], JLC:42];
L1# EIGENVALLESE2N1(THUN, Q);
FOURIERCOEFFSE2N1(THUN, Q, L, 20, F, NCRM);
JAPLSN(2+50RT(G)*SINP(KSI)*L, 42, 10, J);
ZEROSSE2N1 := SUSE2N1; (NCRM=20,F,J);
'END' ZEROSSE2N1;

```

```

'REAL' KSI; G>EXC;
'INTEGER' I, N>TELLER;
'REAL' 'ARRAY' A[0:IMAX] >B[0:IMAX];

'FOR' TELLER:=1 'STEP' 1 'UNTIL' 1 'DL'
'BEGIN'
  EXC= 'CASE' TELLER-1 'OF' ( 0.806 );
  ZERODINAB( EXC*COSH(KSI)=1,KSI,0,10, N=10,A=10 );

'FOR' N:= NMIN 'STEP' 2 'UNTIL' NMAX 'DL'
  'BEGIN' WRITE(CUTPUT,</>120("")); ;
    WRITE(CUTPUT,<>20X1);
    WRITE(CUTPUT,<>"EURULEAN_EDD_SOLUTION2");
    WRITE(CUTPUT,SPACE(5));
    WRITE(CUTPUT,<>"N="); WRITE(CUTPUT,<E18.11>, N);
    WRITE(CUTPUT,<>"---KSI="); WRITE(CUTPUT,<E18.11>, KSI);
    WRITE(CUTPUT,<>"---EXC="); ;
    WRITE(CUTPUT,<E18.11>, 1/COSH(KSI));
    WRITE(CUTPUT,SPACE(2));
  END;
END;

```

```

X+++++  

WHITE(OUTPUT,</x100("=")> );  

WHITE(OUTPUT,</x"1">x10>"u">x10>"CCE2N1-----"> );  

WHITE(OUTPUT,<x20>"L"> );  

'FOR' I := IMIN 'STEP' 1STEP 'UNTIL' IMAX 'DO'  

'BEGIN'  

    Q := I/DELTAG ;  

    'IF' I 'EGT' 0 'THEN' Q:=e-3;

```

```

A[1]:= 0;
B[1]:= ZEROCCE2N1      (N,KSI>0);
  WRITE(OUTPUT,</>,I3, X10,F8.4,X10,E18.10>,
        I,A[1],B[1] );
WRITE(OUTPUT,<X10,E18.11>,L);
*END' CCE2N1POINTCALCULATION
WHITE(OUTPUT,</>,100("=-")> );
*****+
**WRITE(OUTPUT, SPACE(5)) >

*****+
*WHITE(OUTPUT,< "I">,X10,"Q">,X10,"CCE2N1----(U)">
          X10,"HK",X10,"KA"> );
*FOR' I:= IMIN 'STEP' ISTEP 'UNTIL' IMAX-ISTEP 'DG'
*BEGIN'
  'IF' SIGNC B(I) > 'NEG' SIGNC B[1+ISTEP] )
  'THEN'
  'BEGIN'
    ZEROINABC ZEROCCEN1      (N,KSI>0)>G,A[1]>A[1+ISTEP],E=10,F=10);
    WRITE(OUTPUT,</>,13,4(X8,E18.11)>,I,G,
          ZEROCCEN1      (N,KSI>0)>
          SORT(4*G),
          E*SORT(Q)*COSH(KSI) );
  'END';
*END' ZEROCCE2N1      CALCULATION;
*****+
*END';

*FOR' N:= NMIN 'STEP' 2 'UNTIL' NMAX 'DG'
  'BEGIN'WHITE(OUTPUT,</>,120("=-")> );
  WRITE(OUTPUT,<>20X1>);
  WRITE(OUTPUT,<>"EUROBEAN_CDC_SOLUTION2">);
  WRITE(OUTPUT,SPACE(5));
  WRITE(OUTPUT,<>"N=">); WRITE(OUTPUT,<E18.11>, N);
  WRITE(OUTPUT,<>"---KSI=">); WRITE(OUTPUT,<E18.11>,KSI);
  WRITE(OUTPUT,<>"---E=">);
  WRITE(OUTPUT,<E18.11>,1/COSH(KSI));
  WRITE(OUTPUT,SPACE(2));
*****+
*WHITE(OUTPUT,<>100("=-")> );
*WHITE(OUTPUT,</>,100("=-")> );
WHITE(OUTPUT,<>,I1>,X10,"SSE2N1----"> );
WHITE(OUTPUT,<>X20,"L">);
*FOR' I:= IMIN 'STEP' ISTEP 'UNTIL' IMAX 'DG'
*BEGIN'
  'IF' I 'EGLE' 0 'THEN' W:=E=3;
  A[1]:= 0;
  B[1]:= ZEROSSE2N1      (N,KSI>0);
  WRITE(OUTPUT,</>,13, X10,F8.4,X10,E18.10>,
        I,A[1],B[1] );
WHITE(OUTPUT,<X10,E18.11>,L);
*END' SSE2N1      CALCULATION;
WHITE(OUTPUT,</>,100("=-")> );
*****+
**WRITE(OUTPUT, SPACE(5)) >

*****+
*WHITE(OUTPUT,< "I">,X10,"Q">,X10,"SSE2N1----(W)">
          X10,"HK",X10,"KA"> );
*FOR' I:= IMIN 'STEP' ISTEP 'UNTIL' IMAX-ISTEP 'DG'
*BEGIN'
  'IF' SIGNC B(I) > 'NEG' SIGNC B[1+ISTEP] )
  'THEN'
  'BEGIN'
    ZEROINABC ZEROSSE2N1      (N,KSI>0)>G,A[1]>A[1+ISTEP],E=10,F=10);
    WRITE(OUTPUT,</>,13,4(X8,E18.11)>,I,G,
          ZEROSSE2N1      (N,KSI>0)>
          SORT(4*G),
          E*SORT(Q)*COSH(KSI) );
  'END';
*END' ZEROSSE2N1      CALCULATION;
*****+
*END';
*END';
*END';

```

## Chapter 3

### EIGENVALUETEST

This program has been used in the numerical experiments of Part I, Chapter 2. It merely calculates the zeros of the determinant:

$$|AB - \lambda I| = 0 \quad (1.3)$$

at a given interval.

LAMBDA	$2^{-36}$
N	order of the matrix
M	$M = N + 10$ , max. order of the Mathieu function
DETERM( LAMBDA,A,AA,N)	procedure to compute the determinant for fixed value of $\lambda$ matrix AB
A,AA [0:N,0:N]	
FC,FS [1:N,0:M]	fouriercoefficients
RTR, RTI [0:N]	- not in use -
LU [1:N, 1:N]	- not in use in main program - in procedure DETERM : croutdecomposition position
KSI	$\xi$
Q	$q = q_c$
EXC	eccentricity
KA	$k_a$
BETAPPOINT	$\beta' = \beta / k_a$
DET	determinant
i, j, L	ancillary variables
LMAX	number of eccentricities

LAMBDA  
DELTALAMBDA  
LAMBDASTEP  
LAMBDAWIN  
LAMBDAWMAX  
ii  
LAMBDAARRAY  
DETARRAY

}

$\lambda$

array for  $\lambda$

array for determinant

\*BEGIN\*\*FILE\*OUTPUT\* 1st

```

*REAL' MACHEPS, LAMBDA
*INTEGER' N, K
MACHEPS:=2**(-10)
N:=100
K:=200
*BEGIN*
*+*****+
*PROCEDURE* FILLD01(CC, KSI, L, A, R, FC, FS);
*VALUE* KSI, N, K, *REAL, CC, KSI, *INTEGER, N;
*ARRAY* A[*,*], FC[*,*], FS[*,*];
*BEGIN*
    *INTEGER* L, K, N, R;
    *REAL* L, S, NORM, P;
    *ARRAY* FLU[*,*], VELA[*,*], CC1*CC2*, SSE1*N1*, J[0:2*N+2];
    N2:=2*N+1;
    JAPLUSN(2*SORT(G)*SINH(KSI), 0, 2*N+2, 10, 0);
    *FOR*I:=1*STEP*2*UNTIL*N2
    *DO* BEGIN
        L:= EIGENVALUEC2N1(I, G);
        FOURIERCOEFFC2N1(I, G, L, N, F, NORM);
        P:= NORM;
        *FOR*K:=0*STEP*1*UNTIL*N*DO*FC(I+1)'DIV'2, K]:=F[K]/NORM;
        CC1(I+1)'DIV'2]:=-(H1=CCE2N1(P, N, F, J))*COSH(KSI)/SINH(KSI);
        CC2(I+1)'DIV'2]:=CCER2N1POINT(P, N, F, J)*2*SQRT(G)*
            COSH(KSI)**2/SINH(KSI)**R/SINH(KSI)**2;
        S:=0;
        *FOR* K:=0 *STEP* 1 *UNTIL* N *DO*
        S:=S+F[K];
        S:=S/FLU[J];
        CC1(I+1)'DIV'2]:=CC1(I+1)'DIV'2*S;
        CC2(I+1)'DIV'2]:=CC2(I+1)'DIV'2*S;
    *END*;
    *FOR*I:=1*STEP*2*UNTIL*N2
    *DO* BEGIN
        L:= EIGENVALUEC2N1(I, G);
        FOURIERCOEFFC2N1(I, G, L, N, F, NORM);
        P:= NORM;
        *FOR*K:=0*STEP*1*UNTIL*N*DO*FS(I+1)'DIV'2, K]:=F[K]/NORM;
        SSE1(I+1)'DIV'2]:=SSE2N1(P, N, F, J)/SSE2N1POINT(P, N, F, J)/
            (2*COSH(KSI)*SQRT(G));
    *END*;
    *FOR*I:=1*STEP*1*UNTIL*N
    *DO* *FOR*K:=1*STEP*1*UNTIL*N
    *DO* BEGIN*S:=0*
        *FOR*R:=0*STEP*1*UNTIL*N
        *DO*S:=S+(2*R+1)*FS(I,R)*FC[K,R];
        VELA[K]:=S;
    *END*;
    *FOR*I:=1*STEP*1*UNTIL*N
    *DO* *FOR*K:=1*STEP*1*UNTIL*N
    *DO* BEGIN*S:=0*
        *FOR*R:=1*STEP*1*UNTIL*N
        *DO*S:=S+S*VELA[R]*V(R, IJ*V[R, K];
        A(I, K):=S*CC1(K)/CC2(I));
    *END*;
*END*FILLD01;

*+*****+
*REAL'*PROCEDURE* DETERM_LAMBDA(A, R, K);
*VALUE* N;
*REAL' LAMBDA;
*INTEGER' N;
*ARRAY* A[*,*], R[*,*];
*BEGIN*
    *ARRAY* LUL 1:N, 1:N, 1:N;
    *INTEGER* ARRAY* RL 1:N, 1:N;
    *INTEGER* I, J, L;
    *BOOLEAN* SINGULAR;

```



## Chapter 4

### DEPTHODD1 / DEPTHODD2

The program DEPTHODD1 computes the depth of the grooves for the even modes ( $e_{nm}^{EH}$ ), DEPTHODD2 for the odd modes ( $\sigma_{nm}^{EH}$ ). For this depth the waveguide is anisotropic.

The eccentricity  $e_1$  and the frequency  $ka_1$  must be given beforehand. Then the program computes the eccentricity  $e_2$  (and several other quantities such as  $e_1 - e_2$ ,  $b_2/a_2$ ,  $\xi_1$  tec.), as the zero of the function

$$Se_n'(\xi_1, q_0) \cdot Gey_n(\xi_2, q_0) - Se_n(\xi_2, q_0) \cdot Gey_n'(\xi_1, q_0) \quad (1.4)$$

for the even modes, and for the odd modes the zero of the function:

$$Ce_n'(\xi_1, q_0) \cdot Fey_n(\xi_2, q_0) - Ce_n(\xi_2, q_0) \cdot Fey_n'(\xi_1, q_0) \quad (1.5)$$

It also checks the result by substituting the obtained value in this function, which result should then be zero or a very small value.

KS11	$\xi_1$
KS12	$\xi_2$
NORM	normalisation factor of the Mathieu function
FX	function (1.4) or (1.5)
EXC1	eccentricity $e_1$
EXC2	eccentricity $e_2$
Q	$q = q_c$
KA1	$ka_1$
KA2	$ka_2$
PI	$\pi$
K	initial value of FX

L	eigenvalue of the Mathieufunction
A	$S_{e_{2n+1}}(\xi_2, q_0)$ ; $G_{e_{2n+1}}(\xi_2, q_0)$
B	$G_{ey'_{2n+1}}(\xi_1, q_0)$ ; $F_{ey'_{2n+1}}(\xi_1, q_0)$
C	$S_{e'_{2n+1}}(\xi_1, q_0)$ ; $G'_{e'_{2n+1}}(\xi_1, q_0)$
D	$G_{ey_{2n+1}}(\xi_2, q_0)$ ; $F_{ey_{2n+1}}(\xi_2, q_0)$
H	step distance of $\xi$ ( in searching optimum $\xi_2$ )
X1	- not in use -
X2	- not in use -
OG	$\xi_1$ = lower limit of $\xi$
BG	$\xi_2 - H$ = upper limit of $\xi$
TWONL	order N of the mode
i	- not in use -
M	order of the Mathieufunction
TELLER	number of eccentricities
KK	$k_a$ , interval
F [0:M]	fouriercoefficients of the Mathieufunction
J [0: 2.M+2]	Besselfunctions for the M.f.
Y [0: 2.M+2]	Neumannfunctions for the M.f.
DEPTH1( KSI1,KSI2,Q,A,B,C,D )	procedure which computes the value of the functions (1.4) or (1.5)

DEPTHODD1

```
      & 'SET' TRELASRANT
      'DEUIN'
      'FILE' OUT

      'REAL' MACHEPS
      'MACHEPS= 2**(-36)'

      'BEGIN'
      6
      'REAL' KSI1,KSI2,G,A,B,C,D
      ,LX1,LX2,G,A,B,C,D
      P1,K1
      A1,B1,C1,D1
      X1,X2
      GEY2N1
      'INTEGER' TWUN1,TWUN2,KK
      'I := 20'
      'DEUIN'
      'REAL' 'ARRAY' FLU:M1, J[0:2*M+2], Y[0:2*M+2]

      -----
      'REAL' 'PROCEDURE' DEPTH1(KSI1,KSI2,G,A,B,C,D)
      'VALUE' KSI1,KSI2,G,A,B,C,D 'REAL' KSI1,KSI2,G,A,B,C,D
      'BEGIN'

      JAPLUSN(2*SQR(G)*SINH(KSI2)*0.2*M+2,10*J);
      A1=SSE2N1(NURM,M,F,J);

      JAPLUSN( SQR(G)*EXP(-KSI2)*0.5*M+1,10*J);
      YHPLUSN(SQR(G)*EXP(KSI2)*M+1,10*J);
      D1= GEY2N1(NURM,M,F,J,Y);

      DEPTH1 := A*B-C*D
      END DEPTH1;
      -----
      WRITE(OUT,</>"EVEN" );
      M:= 0.08
      P1:= 4*ARCTAN(1)
      TWUN1:=1
      WRITE(OUT,</>"TWUN1=",x4,13>,TWUN1 );

      'BEGIN'
      'FOR' TELLER:=1 'STEP' 1 'UNTIL' I 'DU'
      'BEGIN'
      EXC1:= 0.9180123
      ZERUNAB( EXL1+COS(KSI1)=1,KSI1,x1,J=-10..H=10);
      WRITE(OUT,</>"EXC1=""x4,F10.7", "KSI1=""x4,F10.7", "P1/A1=""x4,F10.7",
      // "A1/B1=""x4,F10.7", EXL1,KSI1, SGR(1-EXC1**2),
      1/SGR(1-EXC1**2));
      WRITE(OUT,</>"KA1=""x5,""KA2""x5,"KA1-KA2", "X4,""EXC2", "X4,
      -"KA12", "X4,""EXC1-EXC2", "X4,""B2/A2", "X4,""A2/B2", "X4,""NULL" );
      'FOR' KK:= 10 'STEP' 1 'UNTIL' 250 'DU'
      'BEGIN'
      KA1:= KK/100
      G:= (KA1*EXC1)**2/45
      L:= EIGENVALUES2N1(TWUN1,G);
      FOUZIEREFFSE2N1(TWUN1,G,L,F,NORMY);
```

```
JAPLUSN(2*SQRT(Q)*SINH(KSI11)*C**2*X**2*10*X))  
U:= Z*SQRT(Q)*CUSH(KSI11)*SSEZKIFGINT(NURM,M,F,U));  
JAPLUSN(SQRT(U)*EXP(-KSI11),U**2-X10*X));  
YPLUSN(SQRT(U)*EXP(KSI11),U**2-X10*X));  
E:=GEYZNAPOINT(NURM,M,F,Y,SQRT(Q)*EXP(-KSI11),  
SQRT(U)*EXP(KSI11));  
  
KSI2:=KSI11;  
K:=FX:=DEPTH1(KSI11,KSI2,Q,A,B,C,D);  
  
'WHILE' SIGN(K) 'EWL' SIGN(FX) 'DO'  
'BEGIN' FX:= DEPTH1(KSI11,KSI2,Q,A,B,C,D);  
  
    KSI2:= KSI2+ H;  
'END' FUNCTIEVERLOOF;  
  
    UG:= KSI11;  
    BG:= KSI2-H;  
  
    'IF' ZEROINABC(DEPTH1(KSI11,KSI2,Q,A,B,C,D),  
                          KSI2,UG,Bge=10,R=10);  
    'THEN'  
    'BEGIN'  
        EXC2:= 1/CUSH(KSI2);  
        WRITET(OUT,</>,(F10.7,X1)), KA1,KA2:= KA1*EXC1/EXC2, KA2=KA1*  
                          EXC2, KSI2, EXC1=EXC2,  
                          SQRT(1-EXC2**2), 1/SQRT(1-EXC2**2),  
                          DEPTH1(KSI11,KSI2,Q,A,B,C,D));  
    'END' NULPUNT;  
  
    'END' TWON LOOP -> LOOPA;  
  
        WRITET(OUT,</>,100("=-")> );  
'END'  
'END'  
'END'  
'END'  
'END'
```

DEPTHODD2

```
CAPLUSN(Z*SGRT(G)*SINH(KSI1),0,Z*M+Z*10,Y))  
C:= -CCE2N1(NURM,M,F,J)/(SINH(KSI1)**Z)+  
2*SGRT(G)*SINH(KSI1)*( CUSH(KSI1)/SINH(KSI1) )**2*  
CCE2N1PGLT(NURM,M,F,J));  
CAPLUSN(SWHTLG)*EXP(-KSI1),0,M+2 ,10,Y));  
YPLUSH(SWHT(G)*EXP(KSI1),M+2 ,10,Y));  
BIEFELY2N1PGLT(NURM,M,F,J,Y*SGRT(G)*EXP(-KSI1),  
SURT(G)*EXP(KSI1) );  
  
KSI2:=KSI1;  
K:=FX:=DEPTH1(KSI1,KSI2,W,A,B,C,D);  
  
'WHILE' SIGN(K) 'EGNL' SIGN(FX) 'DO'  
'BEGIN' FX:= DEPTH1(KSI1,KSI2,W,A,B,C,D);  
  
KSI2:= KSI2+ H;  
'END' FUNCTIEVERLOOP;  
  
UG:= KSI1;  
BG:= KSI2-H;  
  
'IF' ZERDINABL DEPTH1(KSI1,KSI2,W,A,B,C,D),  
KSI2,UG,BG,H-10,e-10 )  
'THEN'  
'BEGIN'  
EXC2:= 1/CUSH(KSI2);  
WRITL(OUT,</>,(F10.7>X1)), KA1,KA2:= KA1*EXC1/EXC2, KA2=KA1,  
EXC2, KSI2, EXC1*EXC2,  
SGRT(1-EXC2**2), 1/SGRT(1-EXC2**2),  
DEPTH1(KSI1,KSI2,W,A,B,C,D) );  
'END' NULPUNT;  
  
'END' TWON LOOP & LOOPS;  
WRITL(OUT,</>,100("=-") );  
  
'END' TELLEHLOOP;  
'END'  
'END'  
'END'.
```

## Chapter 5

### ABCD / ABEF / PLOTLAMDAQ

The program ABCD computes the phasefactor for the corrugated waveguide for the even modes, ABEF for the odd modes. PLOTLAMDAQ produces the graphs.

We shall only describe the program ABCD for ABEF has the same structure.

The program ABCD computes the zeros of the determinant:

$$\left| (BA - \lambda I) - (1-\lambda) CD \right| = 0 \quad (1.6)$$

This happens in the procedure DET( LAMBDA, KSI1, KSI2, QC, AA, A, N ), while the procedure FILLODDA and FILCD calculates the matrix BA and CD respectively ( for the odd modes FILLODDA computes the matrix AB ). For a given value of  $q = q_c$  the program determines in a given  $\lambda$ -interval the determinant. After this it searches for a change of sign and computes the zero.

The matrix BA is a function of  $q_c$  only, the matrix CD of  $q_c$  and  $q_o$ . The evaluation of the matrix CD is more extensive than the matrix AB, so it seems to be rather convenient to compute matrix CD once and keep AB variable. However, in that case we must know  $q_c$  and  $q_o$  beforehand, which implies that  $q_o$  is constant. There is no severe objection against this, but we want to obtain results which are comparable with those of the anisotropic case which means that  $q_c$  is constant.

Then it is easier to observe the influence of the grooves, which infact is embodied in the matrix CD. Thus we keep  $q_c$  constant, and make  $\lambda$  (and thus  $q_o = q_c \cdot \frac{\lambda}{\lambda-1}$  ) variable. This implies that we compute the matrix AB once and vary the matrix CD with  $\lambda$ .

The program punches on card the data  $(\lambda, q_c)$ . The last card contains

the text "EINDE VAN EXC2". This card must be removed before using the data set for the program PLOTLAMBDAQ.

FILLODDA

Q	$q_c$
KSI	$\xi_1$
N	order of the matrix
A [1:N,1:N]	matrix BA ( or AB )
FS [1:N, 1:M]	fouriercoefficients B( $q_c$ )
SS1 [1:N]	$S_{e_{2n+1}} (\xi_1, q_c)$
SS2 [1:N]	$S'_{e_{2n+1}} (\xi_1, q_c)$
M = N + 10	max. order of the Mathieufunction
CC1 [1:N]	$C_{e_{2n+1}} (\xi_1, q_c)$
CC2 [1:N]	$C'_{e_{2n+1}} (\xi_1, q_c)$
F [0:M]	fouriercoefficients
V [1:N,1:N]	matrix <u>V</u> (see [2], Appendix D )

FILLCD ( or FILEF )

Q	$q_o$
KSI1	$\xi_1$
KSI2	$\xi_2$
N	order of the matrix
CD [1:N,1:N]	CD
EF [1:N,1:N]	EF
M	max. order of the Mathieufunction
FSC [1:N,0:M]	fouriercoefficients $B(q_c)$
FCC [1:N,0:M]	fouriercoefficients $A(q_c)$
SS1C [1:N]	$S_{e_{2n+1}} (\xi_1, q_c)$
SS2C [1:N]	$S'_{e_{2n+1}} (\xi_1, q_c)$
CC1C [1:N]	$C_{e_{2n+1}} (\xi_1, q_c)$

$CC2C[1:N]$	$\underline{C}_e'(\xi, q_c)$
$NN[1:N]$	$\underline{\underline{N}}^T \cdot \underline{\underline{N}}^{-1}$ (see 2 , p146)
$MM[1:N]$	$\underline{\underline{M}}^T \cdot \underline{\underline{M}}^{-1}$ (see 2 , p156)
$YT[1:N, 1:N]$	$\underline{\underline{Y}}^T$
$ZT[1:N, 1:N]$	$\underline{\underline{Z}}^T$
$ZTINVERS[1:N, 1:N]$	$(\underline{\underline{Z}}^T)^{-1}$
next: YT and YTINVERS (c.q. ZT, ZTINVERS) are destroyed:	
$YT[1:N, 1:N]$	$[\underline{\underline{S}_e}(\xi, q_c)]^{-1} \cdot \underline{\underline{Y}}^T \cdot \underline{\underline{N}}^T \cdot \underline{\underline{N}}^{-1}$
$YTINVERS[1:N, 1:N]$	$(\underline{\underline{Y}}^T)^{-1} \cdot \underline{\underline{S}_e}(\xi, q_c)$
$CD[1:N, 1:N]$	$\underline{\underline{C}_D} = [\underline{\underline{S}_e}(\xi, q_c)]^{-1} \cdot \underline{\underline{Y}}^T \cdot \underline{\underline{N}}^T \cdot \underline{\underline{N}}^{-1} \cdot (\underline{\underline{Y}}^T)^{-1} \cdot \underline{\underline{S}_e}(\xi, q_c)$
$ZT[1:N, 1:N]$	$[\underline{\underline{C}_e}(\xi, q_c)]^{-1} \cdot \underline{\underline{Z}}^T \cdot \underline{\underline{M}}^T \cdot \underline{\underline{M}}^{-1}$
$ZTINVERS[1:N, 1:N]$	$(\underline{\underline{Z}}^T)^{-1} \cdot \underline{\underline{C}_e}(\xi, q_c)$
$EF[1:N, 1:N]$	$\underline{\underline{E}_F} = [\underline{\underline{C}_e}(\xi, q_c)]^{-1} \cdot \underline{\underline{Z}}^T \cdot \underline{\underline{M}}^T \cdot \underline{\underline{M}}^{-1} \cdot (\underline{\underline{Z}}^T)^{-1} \cdot \underline{\underline{C}_e}(\xi, q_c)$

### DET

procedure which computes the determinant  
of  $|(BA-\lambda I)-(1-\lambda)CD|$  for the even modes  
and  $|(AB-\lambda I)-(1-\lambda)EF|$  for the odd modes

### main program:

IMAX	number of eccentricities
EXC1	eccentricity $e_1$
EXC2	eccentricity $e_2$
QMIN	
QMAX	
QSTEP	{ q -interval
DETAQ	
LMIN	
LMAX	
LSTEP	{ -interval
DELTAL	
L	

MAX

HULP

DETERM [ ... ]

LAMBDAARRAY [ ... ]

maximum of N eigenvalues for a N x N matrix

preceding value of determinant

array for the determinant

array for "eigenvalue"  $\lambda$  ( it is not necessary that the arrays are completely filled.

When we have found N eigenvalues-- i.e. N time change of sign--than at least all the

other eigenvalues are not valid (in this

statement it is assumed that we decrease the

value of  $\lambda$  when searching the changes of sign)

ABCD

```
'SET' THELIBRARY
'BEGIN'
'FILE' OUT,IN(KIND=READER);

'FILE' PUNCH;

'INTEGER' N, M;
'REAL' MACHEPS;
MACHEPS:= 2**(-36);
N:=4; M:=N+10;
'BEGIN'

'COMMENT' THIS PROGRAM CALCULATES THE PHASEFACTOR FOR THE CORRUGATED
      ELLIPTICAL WAVEGUIDE. IT COMPUTES THE ZEROS OF THE DETERMINANT
      OF ((AB-L*I)-(1-L)CDI),FOR FIXED QC. THE INTERVAL OF L IS GIVEN
      BY LMIN AND LMAX. INPUT ON CARD ARE IMAX, WHICH IS THE NUMBER
      OF QC, EXC1,EXC2,QC,LMIN,LMAX,LSTEP,DELTAL;

'REAL' QC,00,KSI1,KSI2,EXC1,EXC2,DELTAL,LAMBDA,DETHLUP,HULP;
'INTEGER' I2, MAX, LOWERLIMIT;
'INTEGER' I,IMAX,LMIN,LMAX,LSTEP,J,K,L;
'INTEGER' QMIN,QMAX,QSTEP,DELTAN;
'ARRAY' A,CD,AAT[1:N,1:N],FSC[1:N,0:M],SS1C,SS2C[1:N];

%-----+
Z+++++++
*PROCEDURE FILLODDA(Q, KSI, N, A,M,FS,SS1,SS2);
*VALUE* Q, KSI, N, M *REAL* Q, KSI; *INTEGER* N, M;
*ARRAY* A[*,*], FSE[*,*], SS1,SS2[*];
*BEGIN* INTEGER I, K, N2, R;

*REAL* L, S, NORM, H;
*ARRAY* F[0:M], V[1:N,1:N], CC1,CC2[1:M], J[0:2*M+2];
N2:=2*N-1;
JAPLUSN(2*SQRT(Q)*SINH(KSI), 0, 2*M+2, 10, J);
FOR I:=1 STEP 2 UNTIL N2
DO BEGIN L:=EIGENVALUE2N1(I, 0);
FOURIERCOFFCE2N1(I, Q, L, M, F, NORM);
FOR K:=0 STEP 1 UNTIL M DO FCI(I+1)'DIV'2, K]:=F[K]/NORM;
CC1[(I+1)'DIV'2]:=(H:=(CCE2N1(1,M,F,J))*COSH(KSI))/SINH(KSI);
CC2[(I+1)'DIV'2]:=CCE2N1POINT(1, M, F, J)*2*SQRT(Q)*
COSH(KSI)**2/SINH(KSI)-H/SINH(KSI)**2;
END;
FOR I:=1 STEP 2 UNTIL N2
DO BEGIN L:=EIGENVALUE2N1(I, 0);
FOURIERCOFFSE2N1(I, Q, L, M, F, NORM);
FOR K:=0 STEP 1 UNTIL M DO FSI(I+1)'DIV'2, K]:=F[K]/NORM;
SS1[(I+1)'DIV'2]:=SSE2N1(1,M,F,J);
SS2[(I+1)'DIV'2]:=SSE2N1POINT(1,M,F,J)*
(2*COSH(KSI)*SQRT(Q));
END;
FOR I:=1 STEP 1 UNTIL N
DO FOR K:=1 STEP 1 UNTIL N
DO BEGIN S:=0;
FOR R:=0 STEP 1 UNTIL M
DO S:=S+(2*R+1)*F(I,R)*FCI(K,R);

```

```

      VEL[K]:=S;
    END;
  FOR I:=1 STEP 1 UNTIL N
  DO FOR K:=1 STEP 1 UNTIL N
    DO REGIST[S]:=0;
    FOR R:=1 STEP 1 UNTIL N
    DO S:=S+CC1[R]*VEL[K]*VEL[R]/CC2[R];
    AL[I,K]:=S*SS1[K]/SS2[I];
  END;
ENDFILEDDA$;

Z+++++++
%+++++++
*PROCEDURE FILEDCD Q0, KSI1, KSI2, N, CD, M, FSC, SS1C, SS2C );
*VALUE Q0, KSI1, KSI2, N, M;
*REAL Q0, KSI1, KSI2 ;
*INTEGER N, M ;
*ARRAY CD[*,*], FSC[*,*], SS1C, SS2C[*] ;
*BEGIN
  BOOLEAN SINGULAR;
  REAL LAMBDA, NORM, SQRTQ, V1, V2, V3, V4, X1, X2, X3, X4, S,
  V6, V7 , V5,X5,X6;
  ARRAY F[0:M], FS0[1:N,0:M], Y2,Y5[0:M+2],NN[1:N],
  YT,YTINVERS, LU, ZI[1:N,1:N], J1,J2,J3,J5[0:2*M+2];
  INTEGER ARRAY P[1:N] ;
  INTEGER I, L, K, R ,J;
  SQRTQ:= SQRT(Q0);
  V1:= 2*SQRTQ*SINH(KSI2);
  V2:= 2*SQRTQ*SINH(KSI1);
  V3:= SQRTQ*EXP(-KSI2);
  V4:= Q0/V3 ;
  V5:= 2*SQRTQ*COSH(KSI1) ;
  V6:= SQRTQ*EXP(-KSI1) ;
  V7:= Q0/V6 ;
  JAPLUSNC V1, 0, 2*M+2, 10, J1);
  JAPLUSN (V3, 0, M+1, 10, J2);
  YPLUSNC V4, M+1, 10,Y2);
  JAPLUSNC V2, 0, 2*M+2, 10, J3);
  JAPLUSNC V6, 0, M+2, 10, J5);
  YPLUSNC V7, M+2, 10, Y5);
  FOR I:= 1 'STEP' 2 'UNTIL' 2*N-1 'DO'
  BEGIN
    LAMBDA:= EIGENVALUESE2N1( I,Q0 );
    FOURIERCOEFFSE2N1( I, Q0, LAMBDA, M, F, NORM );
    FOR L:=0 'STEP' 1 'UNTIL' M 'DO'
    FS0[ (I+1)'DIV' 2,L]:= F[L]/NORM;
    X1:= SSE2N1( 1, M, F, J1);
    X2:= GEY2N1( 1, M, F, J2,Y2);
    X3:= SSE2N1( 1, M, F, J3);
    X4:= V5+SSE2N1POINT( 1, M, F, J5);
    X5:= GEY2N1( 1, M, F, J5,Y5);
    X6:= GEY2N1POINT( 1, M, F, J5,Y5,V5, V7 );
    NN[ (I+1)'DIV' 2 ]:= ((4*X2-X1*X6)/( X3*X2-X1*X5));
  END 'STEP';
  FOR I:=1 'STEP' 1 'UNTIL' N 'DO'
  FOR K:=1 'STEP' 1 'UNTIL' N 'DO'
  BEGIN
    S:=0;
    FOR R:=0 'STEP' 1 'UNTIL' M 'DO'

```

```
S:= S+FSC(I,RI)*FSC(EK,RI);
YT(K,I) := S;
END;

ROUTINE DECOMPOSITION(1, N, YT, LU, P, MACHEPS, SINGULAR);
IF SINGULAR THEN
  WRITE(COUT, <>, "SINGULAR-PROCEDURE-CD");
ELSE CROUTINVERSE(1, N, LU, P, YTINVERS);
FOR I:=1 STEP 1 UNTIL N DO;
FOR J:=1 STEP 1 UNTIL N DO;
BEGIN;
YT[I,J]:=NNE(J)*YT[I,J]/SS2C(I);
YTINVERSE(I,J) := YTINVERSE(I,J)*SS1C(I);
END;

FOR I:=1 STEP 1 UNTIL N DO;
FOR J:=1 STEP 1 UNTIL N DO;
BEGIN;
S:=0;
FOR R:=1 STEP 1 UNTIL N DO;
S:= S+YT(I,R)*YTINVERSE(R,J);
CD(I,J):= S;
END;
END; PROCEDURE CD;
*****
REAL PROCEDURE DET(LAMBDA, KSI1, KSI2, QC, AA, A-N);
VALUE LAMBDA, QC, N, KSI1, KSI2;
REAL LAMBDA, QC, KSI1, KSI2;
INTEGER N;
ARRAY AA, AL, *, *;
BEGIN INTEGER ARRAY P[1:N];
REAL ARRAY CD, LU[1:N, 1:N];
INTEGER J, K; REAL QD; BCOLEAN SINGULAR;
QD:=QC*LAMBDA/(LAMBDA-1);
FILL(CD(QD, KSI1, KSI2, N, CD, M, FSC, SS1C, SS2C));
FOR J:=1 STEP 1 UNTIL N DO AA[J,J]:=A[J,J]-LAMBDA;
FOR J:=1 STEP 1 UNTIL N DO
FOR K:=1 STEP 1 UNTIL N DO
CD(J,K):=AA[J,K]-(1-LAMBDA)*CD(J,K);
CROUTINDECOMPOSITION(1, N, CD, LU, P, MACHEPS, SINGULAR);
IF SINGULAR THEN WRITE(COUT, <>, "SINGULAR_DETERMINANT");
DET:=CROUTINDETERMINANT(1, N, LU, P);
END; PROCEDURE DET;
*****

```

```
IMAX:=1;
WRITE(COUT, <>, "IMAX=", 14//>, IMAX);
FOR I:=1 STEP 1 UNTIL IMAX DO;
BEGIN;
EXC1:= 'CASE' I-1 'OF' ( 0.918012 );
EXC2:= 'CASE' I-1 'OF' ( 0.806 );
QMIN:= 'CASE' I-1 'OF' ( 25 );
QMAX:= 'CASE' I-1 'OF' ( 45 );
QSTEP:= 'CASE' I-1 'OF' ( 1 );
DELTAQ:= 'CASE' I-1 'OF' ( 10 );
LMIN:= 'CASE' I-1 'OF' ( 101 );
LMAX:= 'CASE' I-1 'OF' ( 201 );
LSTEP:= 'CASE' I-1 'OF' ( 2 );

```

```
DELTAL:= *CASE* I=1 *OF* ( 100 );
      ZERO IN AB(EXC1*COSH(KSI1)-1,KSI1,0,10,0-10,0-10);
      ZERO IN AB(EXC2*COSH(KSI2)-1,KSI2,0,10,0-10,0-10);
*FOR* I2:= QMIN *STEP* 0STEP *UNTIL* QMAX *DO*
*BEGIN*
QC:= -I2/DELTAL;
      WRITE(OUT,</, "EXC1=",E18.11,X5,"KSI1=",E18.11,
      //,"EXC2=",E18.11,X5,"KSI2=",E18.11,
      //,"QC_=",E18.11,
      //,"LOWERLIMIT=",E18.11,
      //,"UPPERLIMIT=",E18.11>;
      EXC1,KSI1,EXC2,KSI2,QC,LMIN/DELTAL,LMAX/DELTAL);
      WRITE(OUT,</, "LAMBDA",X25,"DETERMINANT">);

FILLDDA(QC,KSI1,N,A,A+FSC,S1C,S2C);
*FOR* J:=1 *STEP* 1 *UNTIL* N*DO*
*FOR* i:=1 *STEP* 1 *UNTIL* N*DO*
AA(J,K):=AA(J,K);
*BEGIN*
  ARRAY LAMBDAARRAY,DETERM0:(LMAX-LMIN)*DIV*LSTEP;
MAX:= 0;
HULP:= DETERM0*(LMAX-LMIN)*DIV*LSTEP; := DETHULP;
      DET(LMAX/DELTAL, KSI1,KSI2,0C,AA,A,N);
      LAMBDAARRAY((LMAX-LMIN)*DIV*LSTEP):= LMAX/DELTAL;
      WRITE(OUT,</, 2(E18.11,X10)>,LMAX/DELTAL,DETHULP );
L:= LMAX-LSTEP;
*WHILE* MAX <SS1 N AND* L >GTR* LMIN *DO*
*BEGIN* LAMRDA:=L/DELTAL;
      DETHULP:=DET(LAMBDA,KSI1,KSI2,0C,AA,A,N);
      WRITE(OUT,</, 2(E18.11,X10)>,LAMBDA,DETHULP);
      DETERM0(L-LMIN)*DIV*LSTEP):=DETHULP;
      LAMBDAARRAY((L-LMIN)*DIV*LSTEP):=LAMBDA;

      *IF* SIGN(HULP) *NEQ* SIGN(DETHULP) *THEN* MAX:=MAX+1 ;
      HULP:= DETHULP;
      LOWERLIMIT:= L;
      L:= L-LSTEP;
*END* LAMBDASTEP;

      WRITE(OUT,</, "MAX=",X4,I3>,MAX);
      WRITE(OUT,</, "LAMBDA" ,X25,"BETAPINT",X20,"KA1">);

*FOR* L:= ((LMAX-LMIN)*DIV*LSTEP-1) *STEP* -1 *UNTIL*
      (LOWERLIMIT-LMIN)*DIV*LSTEP *DO*
*BEGIN*
*IF* SIGN(DETERM0) *NEQ* SIGN(DETERM0+1)
*THEN* *BEGIN* ZERODINAB(DET(LAMBDA,KSI1,KSI2,0C,AA,A,N),LAMBDA,
      LAMBDAARRAY[L],LAMBDAARRAY[L+1],0-10,0-10);
      WRITE(OUT,</, 1(E18.11,X10)>, LAMBDA);
      WRITE(PUNCH,</, 2(E18.11,"")>,LAMBDA,0C);
      *IF* LAMBDA*GTR*1 *THEN*
      WRITE(OUT,<2(E18.11,X10)>,SQRT(1/LAMBDA),2+SQRT(0C*
      LAMBDA/(LAMBDA-1))/EXC1);
      *END*;
*END*LLLOOP;
*END*;

      WRITE(OUT,</, 100("-"))>;
*END*;
      WRITE(PUNCH,</, " EINDE-VAN-EXC2=",E18.11>,EXC2);
*END* ILGUP;
*END*;

*END*.
```

ABEF

```
3 'SET' THE LIBRARY
'BEGIN'
'FILE' OUT, IN(KIND=READER)

'FILE' PUNCH

'INTEGER N, M
'REAL' RACHEPS;

RACHEPS:= 2**(-36);
N:=4; M:= N+10;
'BEGIN'

'COMMENT' THIS PROGRAM CALCULATED THE PHASEFACTOR FOR THE CORRUGATED ELLIPTICAL WAVEGUIDE. IT COMPUTES THE ZEROS OF THE DETERMINANT OF EXAR(L1)-C1-EXP(I*P) FOR FIXED Q. THE INTERVAL OF L IS GIVE BY LMIN AND LMAX. INPUT IN CARD ARE IMAX, WHICH IS THE NUMBER OF Q, EXC1,EXC2,Q,LMIN,LMAX,LSTEP AND DELTAL;

'REAL Q, G, KSI1, KSI2, EXC1, EXC2, DELTAL, LAMDA, DEThLUP, HULP;
'INTEGER I, MAX, LOWERLIMIT;
'INTEGER J, IMAX, LMIN, LMAX, LSTEP, J, K, L;
'INTEGER QMIN, QMAX, QSTEP, DELTAQ;
'ARRAY A, EF, AA(1:N, 1:N), FC(1:N, 0:M), CC1C, CC2C(1:N);

X-----+
Z+++++++
'PROCEDURE FILLODDA(Q, KSI, N, M, FC, CC1, CC2);
'VALUE Q, KSI, N, M; 'REAL Q, KSI; 'INTEGER N, M;
'ARRAY A(*, *), FC(*, *), CC1, CC2(*)';
'BEGIN' 'INTEGER I, K, N2, R;

'REAL L, S, NORM, H;
'ARRAY F(0:M), V(1:N+1:N), SS1, SS2(1:N),
          JE0:2*M+2];
'ARRAY FSI(1:N, 0:M);
N2:=2*N-1;
JAPLUSN(2*SORT(Q)*SINH(KSI), 0, 2*M+2, 10, J);
'FOR I:=1 STEP 2 UNTIL N2
'DO BEGIN L:=EIGENVALUECE2N1(I, Q);
        FOURIERCOEFFCE2N1(I, 0, L, M, F, NORM);
        FOR K:=0 STEP 1 UNTIL M DO FCI(I+1)*DIV*2, KJ:=FC(K)/NORM;
        CC1((I+1)*DIV*2):=(H:=CCE2N1(1,M,F,J))*COSH(KSI)/SINH(KSI);
        CC2((I+1)*DIV*2):=CCE2N1POINT(1, M, F, J)*2*SORT(Q)*
                           COSH(KSI)**2/SINH(KSI)-H/SINH(KSI)**2;
    END;
'FOR I:=1 STEP 2 UNTIL N2
'DO BEGIN L:=EIGENVALUESE2N1(I, Q);
        FOURIERCOEFFSE2N1(I, 0, L, M, F, NORM);
        FOR K:=0 STEP 1 UNTIL M DO FSC(I+1)*DIV*2, KJ:=F(K)/NORM;
        SS1((I+1)*DIV*2):=SSE2N1(1,M,F,J);
        SS2((I+1)*DIV*2):=SSE2N1POINT(1, M, F, J)*
                           (2*COSH(KSI)*SQRT(Q));
    END;
'FOR I:=1 STEP 1 UNTIL N
'DO FOR K:=1 STEP 1 UNTIL N

'DO BEGIN S:=0;
        FOR R:=0 STEP 1 UNTIL M
            DO S:=S+(2*R+1)*FSI(R)*FC(R,K);
        VEL, KJ:=S;
    END;
'FOR I:=1 STEP 1 UNTIL N
'DO FOR K:=1 STEP 1 UNTIL N
'DO BEGIN S:=0;
        FOR R:=1 STEP 1 UNTIL N
            DO S:=S+SS1(R)*V(R,I)*VER(KJ/SS2(R));
        A(I, KJ):=S*CC1KJ/CC2(I);
    END;
END FILLODDA;
```

```
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
*PROCEDURE* FILEFF( Q0, KSI1, KSI2, N, FF, M, FCC, CC1C, CC2C )*
*VALU1* Q0, KSI1, KSI2, N, M
*REAL* Q0, KSI1, KSI2 ;
*INTEGER* N, M ;
*ARRAY* FF(1:N,1:M), FCC1:1:N,0:M), Y1, Y2, Y3, Y4, X1, X2, X3, X4, S ,
V5, V6, V7 , V8, X5, X6;
*ARRAY* FCC1:1:N,0:M), Y1, Y2, Y3, Y4, X1, X2, X3, X4, S ,
ZT, ZTINVERS, LU [1:N,1:N], J1, J2, J3, J5[0:2*M+2];
*INTEGER* FCC1:1:N) ;
*INTEGER* I, L, N, R , J;
SORT0:= SORT(00);
V1:= 2*SORT0*SINH(KSI2);
V2:= 2*SORT0*SINH(KSI1);
V3:= SORT0*EXP(-KSI2);
V4:= Q0/V3 ;
V5:= 2*SORT0*COSH(KSI1) ;
V6:= SORT0*EXP(-KSI1) ;
V7:= Q0/V6 ;
JAPLUSNC( V1, 0, 2*M+2, 10, J1);
JAPLUSN( V3, 0, M+1, 10, J2);
YPLUSNC( V4, M+1, 10, Y2);
JAPLUSNC( V2, 0, 2*M+2, 10, J3);
JAPLUSNC( V6, 0, M+2, 10, J5);
YPLUSNC( V7, M+2, 10, Y5);

*FOR* I:= 1 *STEP* 2 *UNTIL* 2*N-1 *DO*
*BEGIN*
  LAMBDA:= EIGENVALUECE2N1( I, 00 );
  FOURIERCOEFFCE2N1( I, Q0, LAMBDA, M, F, NORM );
  *FOR* L:=0 *STEP* 1 *UNTIL* M *DO*
    FCC1( I+1)*DIV* 2,L):= FCC1/NORM;
    X1:= CCE2N1( I, M, F, J1)*COSH(KSI2)/SINH(KSI2);

    X2:= FEY2N1( I, M, F, J2,Y2);

    X3:= CCE2N1( I, M, F, J3)*COSH(KSI1)/SINH(KSI1);
    X4:=-CCE2N1( I, M, F, J3)/(SINH(KSI1)**2)+V5*COSH(KSI1)/SINH(KSI1)*
      CCE2N1POINT( I, M, F, J3);

    X5:= FEY2N1( I, M, F, J5,Y5);
    X6:= FEY2N1POINT( I, M, F, J5,Y5,V6, V7 );

    MDC( I+1)*DIV*2 ):= (X4*(X2-X1*X6)/( X3*X2-X1*X5));

*END* ISTEP;

*FOR* I:=1 *STEP* 1 *UNTIL* N *DO*
*FOR* K:=1 *STEP* 1 *UNTIL* N *DO*
*BEGIN*
  S:=0;
  *FOR* R:=0 *STEP* 1 *UNTIL* M *DO*
    S:= S+FCC1(I,R)*FCC1K,R);
    ZTEK,I):= S;
*END*;

CROUTDECOMPOSITION( I, N, ZT, LU, P, MACHEPS, SINGULAR);
*IF* SINGULAR *THEN*
  WRITE(OUT , <>, "SINGULAR-PROCEDURE-CD");
*ELSE* CROUTINVERSE( I, N, LU, P, ZTINVERS );

*FOR* I:=1 *STEP* 1 *UNTIL* N *DO*
*FOR* J:=1 *STEP* 1 *UNTIL* N *DO*
*BEGIN*
  ZT(I,J):= RMEJ1*ZTEI,J1/CC2C1;
  ZTINVERSE(I,J1):= ZTINVERSE(J1*CC1C1,J1);
*END*;

*FOR* I:=1 *STEP* 1 *UNTIL* N *DO*
*FOR* J:=1 *STEP* 1 *UNTIL* N *DO*
*BEGIN*
  S:=0;
  *FOR* R:=1 *STEP* 1 *UNTIL* N *DO*
    S:= S+ZTEI,R)*ZTINVERSE(R,J1);
```

```
EF[I,J]:=0;
END;
END' PROCEDURE EFF;

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
REAL' PROCEDURE DET(LAMBDA,KSI1,KSI2,QC,AA,A,N);
VALUE LAMBDA,QC,N,KSI1,KSI2;
REAL' LAMBDA,QC,KSI1,KSI2;
INTEGER;
ARRAY AA,A[1:N];
BEGIN' INTEGER' ARRAY PE1:N];
REAL' ARRAY EF,L1,N,1:N];
INTEGER I,J; REAL' DOF' SINGULAR;
DO:=QC*LAMBDA/(LAMBDA-1);
FILLFE(N,O,KSI1,KSI2,N,F1,M,FCC,CC1C,CC2C);
FOR J:=1 STEP 1 UNTIL N DO;
FOR K:=1 STEP 1 UNTIL N DO;
EF[J,K]:=AA[J,K]-(1-LAMBDA)*AE[J,K];
CHOUTDEC(P,POSITION(1,N,EF,L1,P,MACHEPS,SINGULAR));
IF SINGULAR THEN WRITE(OUT,<"/>" "SINGULAR DETERMINANT">);
DET:=CHOUTDETERMINANT(1,N,L1,P);
END' PROCEDURE DET;
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

IMAX:=1;
FOR I:=1 STEP 1 UNTIL IMAX DO;
BEGIN' WRITE(OUT,<"/>" "IMAX="">I4,//>,IMAX);
EXC1:= 'CASE' I-1 'OF' ( 0.918012 );
EXC2:= 'CASE' I-1 'OF' ( 0.806 );
QMIN:= 'CASE' I-1 'OF' ( 395 );
QMAX:= 'CASE' I-1 'OF' ( 410 );
QSTEP:= 'CASE' I-1 'OF' ( 1 );
DELTAL:= 'CASE' I-1 'OF' ( 100 );
LMIN:= 'CASE' I-1 'OF' ( 101 );
LMAX:= 'CASE' I-1 'OF' ( 401 );
LSTEP:= 'CASE' I-1 'OF' ( 2 );
DELTAL:= 'CASE' I-1 'OF' ( 100 );
ZERO IN AB(EXC1*COSH(KSI1)=1,KSI1,O,10,E-10);
ZERO IN AB(EXC2*COSH(KSI2)=1,KSI2,O,10,E-10);
FOR 12:= QMIN STEP QSTEP UNTIL QMAX DO;
BEGIN';
QC:= 12/DELTAL;
WRITE(OUT,<"/>" "EXC1="">E18.11,X5,"KSI1="">E18.11,
/>" "EXC2="">E18.11,X5,"KSI2="">E18.11,
/>" "QC_="">E18.11,
/>" "LOWERLIMIT="">E18.11,
/>" "UPPERLIMIT="">E18.11>;
EXC1,KSI1,EXC2,KSI2,QC,LMIN/DELTAL,LMAX/DELTAL);
WRITE(OUT,<"/>" "LAMBDA",X25,"DETERMINANT">);

FILLODDA(QC,KSI1,A,M,FCC,CC1C,CC2C);
FOR J:=1 STEP 1 UNTIL N DO;
FOR K:=1 STEP 1 UNTIL N DO;
AA[J,K]:=AE[J,K];
BEGIN';
ARRAY LAMBDAARRAY,DETERM[0:(LMAX-LMIN)/DIV'LSTEP];
MAXI:= 0;
HULP:= DETERM[(LMAX-LMIN)/DIV'LSTEP]:= DETHULP:=
DETLMAX/DELTAL, KSI1,KSI2,QC,AA,A,N);
LAMBDAARRAY[LMAX-LMIN] 'DIV' LSTEP := LMAX/DELTAL;
WHITE(OUT,<"/>(E18.11,X10)>,LMAX/DELTAL,DETHULP );
L:= LMAX-LSTEP;
WHILE MAX 'LSS' A AND L 'GTR' LMIN 'DC'
BEGIN'LAMBDA:=L/DELTAL;
DETHULP:=DET(LAMBDA,KSI1,KSI2,QC,AA,A,N);
WHITE(OUT,<"/>(E18.11,X10)>,LAMBDA,DETHULP);
DETERM[L-LMIN] 'DIV' LSTEP:=DETHULP;
LAMBDAARRAY[L-LMIN] 'DIV' LSTEP:=LAMBDA;
```

```
'IF' SIGN(HULP) 'NEW' SIGN(DETHULP) 'THEN' MAX:=MAX+1 ;
HULP:=DETHULP;
LOWERLIMIT:= L;
L:= L-LSTEP;
'END'LAMBDASTEPI;

WHITE(DUT)<//>"MAX=",X4,X13>,MAX);
WHITE(DUT)<//>"LAMBDA=">X25,"BETAPoint",X20,"KA1">);

'FOR' L:= ((LMAX-LMIN) 'DIV' LSTEP-1) 'STEP' -1 'UNTIL'
(LOWERLIMIT-LMIN) 'DIV' LSTEP '00'
'BEGIN'
'IF!SIGN(DETERM[L])NEQ'SIGN(DETERM[L+1])
'THEN''BEGIN'ZERODIRK(DET(LAMHUA,KSI1,KSI2,GC,AA,A,A),LAMBDA,
LAMBDAARRAY[L],LAMBDAARRAY[L+1],E-10,E-10);
WHITE(LUT)</>1(E18.11,X10)>> LAMBDA);
WHITE(PUNCH,</>2(E18.11,"")>> LAMBDA,GC);
'IF'LAMBDA'GTR'1'THEN'
WHITE(DUT)<2(E18.11,X10)>>SQRT(1/LAMBDA)*2*SGRT(GC*
LAMBDA/(LAMBDA-1))/EXC1);
'END';
'END'LLLOOP;
'END';

WHITE(DUT)</>100("=");
'END';
WHITE(PUNCH,</> EINDE-VAN=EXC2=E18.11>,EXC2);
'END'ILOOP;
'END';

'END';
```

PLOTLAMBDAQ (constructed by I.C. Ongers)

This program plots for the anisotropic as well as for the corrugated waveguides the  $(\lambda, q_c)$  solution and the  $(\beta', ka)$  transformation. The data set is obtained from the programs ABCD and ABEF or from EIGENVALUEODD. Each data set contains  $(\lambda, q_c)$  values only. On each card one number-pair. The data set should be closed by the figures 3000, 3000, (do not forget to remove the card with the text "EINIE VAN EXC2"). Each data set should be preceded by "mode", which is:

- 0 for the anisotropic case
- 1 for the odd-modes of the corrugated waveguide
- 2 for the even modes.

Then follow figures which give the scales of the several axis. The whole data set should be preceded by a figure which give the number of the data sets involved.

example

Suppose we have two data sets, one for the anisotropic case and the other for the corrugated case.

2,	two data sets
0,	i.e. "mode": anisotropic case
0,10,1,0,5,0.5,	$q_c$ interval from 0 to 10, with step 1; $\lambda$ interval from 0 to 5 with step 0.5
0,20,1,0,1,0.1,	$ka$ interval from 0 to 20, with step 1, $\beta'$ interval from 0 to 1 with step 0.1
0.8768,	eccentricity $e = 0.8768$
$(\lambda, q_c)$	
$\lambda, q_c,$	
$\lambda, q_c,$	data set

$\lambda, q_c,$   
 $\lambda, q_c,$   
 $\lambda, q_c,$   
3000,3000,  
2,  
0,20,0.5,-6,6,1,  
0,40,2,0,1,0.1,  
0.9180, 0.806,  
 $\lambda, q_c,$   
 $\lambda, q_c,$   
 $\lambda, q_c,$   
 $\lambda, q_c,$   
3000,3000,

lock of data set  
even mode  
 $0 < q_c < 20 ; q_{step} = 0.5$   
 $-6 < \lambda < 6 ; \lambda_{step} = 1$   
 $0 < k_a < 40 ; k_a_{step} = 2$   
 $0 < \beta' < 1 ; \beta'_{step} = 0.1$   
 $e_r = 0.9180 ; e_z = 0.806$

lock of data set

## PLOTLAMBDAQ

'A LITTLE SPICE'

URLAUFTEILUNGSSYSTEM "LINDNER"

```

        DRAW AXIS(SPECIFYING X-INTERCEPTS) / STEP=1 / LAMMAX=LAMMIN
        'FALSE' / 'TRUE')*
        DRAW AXIS(SPECIFYING X-INTERCEPTS) / STEP=1 / LAMMAX=LAMMIN
        'FALSE' / 'TRUE')*
        IF MODE=0 THEN*
          BEGIN CHARACTERISTICS TYPE=1,1,1,"EXCL=");

```

DATA FOR THE SYSTEMS OF EQUATIONS 14-4, 14-5, "E" AND 14-6, EXC 13  
AND TECHNICAL 14-23, 14-24, "MANISOTROPIC" 13

وَالْمُؤْمِنُونَ إِذَا قُرِئُوا إِذَا قُرِئُوا قَالُوا هُنَّا مُؤْمِنُونَ

2015-08-01 12:14:00.000 "F" 0x44 EXC13  
2015-08-01 12:14:00.000 "F" 0x44 EXC23  
2015-08-01 12:14:00.000 "D" 0x12501000 "2000"  
2015-08-01 12:14:00.000 "D" 0x12501000 "EVEN")

卷之三

2010-01-01 10:00:00 2010-01-01 10:00:00  
2010-01-01 10:00:00 2010-01-01 10:00:00  
2010-01-01 10:00:00 2010-01-01 10:00:00

DEAR RAYMOND FREDERICK KAMINSKI,  
I AM PLEASED TO TELL YOU THAT I HAVE RECEIVED YOUR  
LETTER AND APPRECIATE YOUR CONCERN FOR THE  
WELL-BEING OF MY SON. I AM SORRY THAT HE IS  
NOT FEELING WELL, BUT HE IS RECEIVING MEDICAL  
ATTENTION AND SHOULD RECOVER SOON. I HOPE  
YOU ARE DOING WELL AND THANK YOU FOR YOUR  
CONCERN.

SELECT \* FROM V\$SESSION WHERE STATUS='INACTIVE' AND EXC\_J3  
NOT IN ('ALTER SYSTEM SET LOGICAL STANDBY'')

وَمِنْهُمْ مَنْ يَرْجُوا أَنَّ اللَّهَ يَعْلَمُ مَا لَا يَرَى إِنَّ اللَّهَ عَزِيزٌ عَلَى الْأَنْعَامِ

```
      REAL VALUE(100,100,21,4,1,0) *F=6.44*EXC2J;
      REAL VALUE(100,100,14,4,1,0) *F=6.4*EXC1J;
      IF (JGDE=1) THEN !GRANTEK (PLUT>230.1>250.10)=>"U000"
      IF (JGDE>1) THEN !GRANTEK (PLUT>230.1>250.10)=>"EVEN")
```

LUCY (PLATE) 24

## Chapter 6

### RADABCD/RADABEF

With the program RADABCD we can compute the radiationpattern of the even modes ( $e^{EH_{nm}}$ ), with RADABEF that of the odd modes. The ( $\beta$ ,  $ka$ ) solution and the corresponding ( $\lambda_1, q_c$ ) solution is known from the calculations with ABCD and ABEF. Now we have to find the corresponding eigenvector (see part I, Ch. 3) to compute the aperture fields. Then we are able to compute the radiationpattern. We shall discuss here the program RADABCD.

We use the procedures FILLBA and FILLCD. Procedure FILLBA is the same as FILLODDA of the program ABCD, except for the output arrays FC, BB, NORMCE and NORMSE, and the last statement of the procedure which is the construction of the array BB. The procedure FILLCD is the same as the one in program ABCD.

We construct in ABCD the matrix

$$(BA - CD) - \lambda_1 (I - CD) \quad (1.7)$$

and  $\lambda_1$  satisfies the equation

$$\left| (BA - CD) - \lambda_1 (I - CD) \right| = 0 \quad (1.8)$$

Actually, in ABCD we used

$$\left| (BA - \lambda I) - (I - \lambda) CD \right| = 0 \quad (1.9)$$

So we have to find the eigenvector from:

$$\{(BA - CD) - \lambda_1 (I - CD)\} b = 0 \quad (1.10)$$

or:

$$\{(BA - CD)(J - CD)^{-1} - \lambda_1 J\} \underline{b}^* = 0 \quad (1.11)$$

$$\underline{b}^* = (J - CD) \underline{b} \quad (1.12)$$

Now we construct in this program the matrix

$$(BA - CD)(J - CD)^{-1} \quad (1.13)$$

This is possible with the procedures FILLBA and FILLCD for both  $q_c$  and  $q_o = q_c \cdot \frac{\lambda_1}{\lambda_{-1}}$  are known. The matrix BA only depends on  $q_c$ , which is an independent variable. The matrix CD depends on both  $q_c$  and  $q_o$ . Therefore we need to transport the fouriercoefficients calculated in FILLBA to FILLCD, which happens in the array FSC. (Later, both kinds of fouriercoefficients are needed in the calculation of the aperturefields, as well as the normalisationfactors NORMCE and NORMSE). Now the procedure NONSYMEIGENVECTORS computes the eigenvector  $\underline{b}^*$ . This procedure first computes all the eigenvalues of the matrix  $(BA-CD)(I-CD)^{-1}$  and stores the real parts in the array RTR and the imaginary parts in the array RTI. Naturally, only one of these eigenvalues is equal to  $\lambda_1$ . This eigenvalue has to be found again first. We have to take the corresponding eigenvector out of the matrix U where all the eigenvectors are stored. (To the  $k^{th}$  eigenvalue belongs the  $k^{th}$  column). Now we must determine the eigenvector  $\underline{b}$  from (1.12):

$$\underline{b} = (J - CD)^{-1} \cdot \underline{b}^* \quad (1.12)$$

We also need the eigenvector  $\underline{a}$  (see [2]), which may be found by

computing:

$$\underline{Q} = \beta' \underline{\underline{A}} \underline{b} \quad (1.14)$$

and the matrix  $\underline{\underline{A}}$  is the array BB which has been delivered by the procedure FILLBA. Then for completeness we check the zero-vector of eq. (1.10).

The next step is to compute a matrix of fieldpoints of the aperture. Thus we need the formulas of the fieldcomponents ([2], p13), where again Mathieufunctions have to be computed. (The abbreviations  $s_1$  to  $s_4$  are the same as those of [2], p13). The rest is the same as for the anisotropic case ([2], p115).

For the odd modes in RADABEF the fieldcomponents have to be changed, see [2], p 14, and for the radiationpattern we now have (see also [2], p39):

$$E_\theta(\theta, \eta) = 0 \quad (1.15)$$

$$E_\phi(\theta, \phi) = 0 \quad (1.16)$$

N	order of the matrices
M = N + 10	length of the Mathieufunctions
NKSI	number of samples in $\xi$ -direction in the apertureplane
NETA	number of samples in $\eta$ -direction in the apertureplane
NTHETA	number of samples over the angle $\theta$ of the radiationpattern

TELLER	number of eccentricities
BA	matrix BA
BB	matrix A
A1	ancillary arrays
A2	
CD	matrix CD
IU	matrix for Croutdecomposition
FC	fouriercoefficients $A_n^m(q_c)$
FSC	fouriercoefficients $B_n^m(q_c)$
AAA	ancillary array
U	eigenvectors
EETA	$E_\eta$
EKSI	$E_\xi$
HETA	$H_\eta$
HKSI	$H_\xi$
RTR	array for $\text{Re} \{ \lambda \}$
RTI	array for $\text{Im} \{ \lambda \}$
B	original eigenvector $b^*$
X	vector $a$
Y	vector $b = (I-CD)^{-1} b^*$
NORMCE	norm. coeff. for the Mathieufunctions
NORMSE	norm. coeff. for the Mathieufunctions
SS1C	$S_e(\xi_1, q_c)$
SS2C	$S_e'(\xi_1, q_c)$
F1	$A_n^m(q_c)$
F2	$B_n^m(q_c)$
JJ	Besselfunctions for the Mathieufunctions
SI	sine functions for the Mathieufunctions
CO	cosine functions for the Mathieufunctions
EXC1	eccentricity $e_1$
EXC2	eccentricity $e_2$

QC	$q_c$
QO	$q_o$
SQRTQC	$\sqrt{q_c}$
EIGENV	eigenvalue $\lambda_1$ (input)
BETAPPOINT	$\beta' = \beta/h_0$
KH	kh
KAL	$k_a$
KSI1	$\xi_1$
KSI2	$\xi_2$
KSI	$\xi$
ETA	$\eta$
DEIKSI	step in $\xi$ direction in aperture plane
DELETA	step in $\eta$ direction in aperture plane
THETA	beamangle $\theta$
DELTHETA	step distance of $\theta$
THETAMAX	max. value of $\theta$
NORM1	ancillary variables
NORM2	
NORMS	norm. factors of the M.f.
NORMC	
FIELD1/FIELD4	fieldcomponents
s1 / s4	ancillary variables
s	unnormalised radiationpattern
CT	$\cos(\theta)$
ST	$\sin(\theta)$
DELTA	ancillary variables in searching the correct eigenvalue
HULP	
H	ancillary variable
PI2	$2\pi$

RADABCD

```
$ 'SET' SEG
$ 'SET' THELIBRARY
'BEGIN'
'FILE' OUTPUT,PUNCH

'REAL' MACHEPS
'INTEGER' NMA, NETAI,NKSI,NTHETA, TELLER
MACHEPS=2**(-36)
N1=4;
M1=N+10;
NKSI= 10;
NETAI= 160;
NTHETA= 45;

'BEGIN'

-----
'PROCEDURE' FILLBAC Q, KSI, N, BA, M, FC, FS,BB, NORMCE, NORMSE,SS1,SS2}
'VALUE' Q, KSI, N, M; 'REAL' Q, KSI; 'INTEGER' N, M
'ARRAY' BA, FC, FS, BB[*,*], NORMCE, NORMSE, SS1, SS2[*]
'BEGIN' 'INTEGER' I, K, N2, R;
'ARRAY' F[0:M], V[1:N]*1[N], CC1,CC2[1:N], J[0:2*M+2];
'REAL' L, S, NORM, H;
N2=2*N-1;
JAPLUSN(2*SQRT(Q)*SINH(KSI), 0, 2*M+2, 10, J);
'FOR' I:=1'STEP'2'UNTIL'N2
'DO' 'BEGIN' L:=EIGENVALUEC2N1(I, Q);
    FUURIERCOEFFCE2N1(I, Q, L, M, F, NORM);
    'FOR' K:=0'STEP'1'UNTIL'N'DO' FCE((I+1)'DIV'2, K):=F[K];
    NORMCE((I+1)'DIV' 2) := NORM;
    CC1((I+1)'DIV'2):=(H1=CCE2N1(I,M,F,J))*COSH(KSI)/SINH(KSI);
    CC2((I+1)'DIV'2):=CCE2N1POINT(I, M, F, J)*2*SQRT(Q)*
        COSH(KSI)**2/SINH(KSI)*H/SINH(KSI)**2;
    'END';
    'FOR' I:=1'STEP'2'UNTIL'N2
    'DO' 'BEGIN' L:=EIGENVALUESE2N1(I, Q);
        FUURIERCOEFFSE2N1(I, Q, L, M, F, NORM);
        'FOR' K:=0'STEP'1'UNTIL'N'DO' FS((I+1)'DIV'2, K):=F[K];
        NORMSE((I+1)'DIV' 2) := NORM;
        SS1((I+1)'DIV'2):=SSE2N1(I,M,F,J);
        SS2((I+1)'DIV'2):=SSE2N1POINT(I,M,F,J)*
            (2*COSH(KSI)*SQRT(Q));
    'END';
    'FOR' I:=1'STEP'1'UNTIL'N
    'DO' 'FOR' K:=1'STEP'1'UNTIL'N
    'DO' 'BEGIN' S:=0;
        'FOR' R:=0'STEP'1'UNTIL'M
        'DO' S:=S+(2*R+1)*FS[I,R]*FC[K,R];
        VCI, K):=S/( NORMSE[I]*NORMCE[K] );
    'END';
    'FOR' I:=1'STEP'1'UNTIL'N
    'DO' 'FOR' K:=1'STEP'1'UNTIL'N
    'DO' 'BEGIN' S:=0;
        'FOR' R:=1'STEP'1'UNTIL'N
        'DO' S:=S+CC1[R]*V[I,R]+V[K,R]/CC2[R];
        BA[I,K]:= S*SS1[K]/SS2[I];
    'END';
    'FOR' I:=1'STEP'1'UNTIL'N
    'DO' 'FOR' K:=1'STEP'1'UNTIL'N'DO' BB[I,K]:= V[K,I]*SS1[K]/CC2[I];
'END' FILLBAC

-----
'PROCEDURE' FILLCDC Q0, KSI1, KSI2, N, CD, M, FSC, SS1C, SS2C }
'VALUE' Q0, KSI1, KSI2, N, M
'REAL' Q0, KSI1, KSI2;
'INTEGER' N, M;
'ARRAY' CD[*,*], FSC[*,*], SS1C,SS2C[*];
'BEGIN'
'BOOLEAN' SINGULAR;
'REAL' LAMBDA, NORM, SQRTQ, V1, V2, V3, V4, X1, X2, X3, X4, S,
    V6, V7, V5, X5, X6;
```

```
'ARRAY' F[0:N], FS0[1:N,0:N], Y2, Y5L0:M+2], NN[1:N],
      YTAYTINVERS, LU, ZI[1:N,1:N], J1,J2,J3,J5[0:N+2])
'INTEGER' 'ARRAY' P[1:N]
'INTEGER' I, L, K, R, J

SORTQ:= SORT(GU);
V1:= 2*SORTQ*SINH(KSI2);
V2:= 2*SORTQ*SINH(KSI1);
V3:= SORTQ*EXP(-KSI2);
V4:= Q0/V3;
V5:= 2*SORTQ*COSH(KSI1);
V6:= SORTQ*EXP(-KSI1);
V7:= Q0/V6;
JAPLUSNC( V1, 0, 2*M+2, 10, J1);
JAPLUSNC( V3, 0, M+1, 10, J2);
YPLUSNC( V4, M+1, 10, Y2);
JAPLUSNC( V2, 0, 2*M+2, 10, J3);
JAPLUSNC( V6, 0, M+2, 10, J5);
YPLUSNC( V7, M+2, 10, Y5);

'FOR' I:= 1 'STEP' 2 'UNTIL' 2*N-1 'DO'
'BEGIN'
  LAMBDA:= EIGENVALUESE2N1( I,Q0 );
  FOURIERCOEFFSE2N1( I, Q0, LAMBDA, M, F, NORM );
  'FOR' L:=0 'STEP' 1 'UNTIL' M 'DO'
    FS0[ (I+1)'DIV' 2,L]:= F[L]/NORM;
  X1:= SSE2N1( 1, M, F, J1);

  X2:= GEY2N1( 1, M, F, J2,Y2);

  X3:= SSE2N1( 1, M, F, J3);
  X4:= V5*SSE2N1POINT( 1, M, F, J3);

  X5:= GEY2N1( 1, M, F, J5,Y5);
  X6:= GEY2N1POINT( 1, M, F, J5,Y5,V6, V7 );

  NN[ (I+1)'DIV' 2 ]:= (X4*X2-X1*X6)/( X3*X2-X1*X5);
'END' 'ISTEP';

'FOR' I:=1 'STEP' 1 'UNTIL' N 'DO'
'FOR' K:=1 'STEP' 1 'UNTIL' N 'DO'
'BEGIN'
  S:=0;
  'FOR' R:=0 'STEP' 1 'UNTIL' M 'DO'
    S:= S+FS0[I,R]*FSC[K,R];
    YT[K,I]:= S;
'END';

CROUTDECOMPOSITION( 1, N, YT, LU, P, MACHINEPS, SINGULAR);
'IF' SINGULAR 'THEN'
  WRITE(OUTPUT, //> "SINGULAR=PROCEDURE-C0");
'ELSE' CROUTINVERSE(1, N, LU, P, YTINVERS );

'FOR' I:=1 'STEP' 1 'UNTIL' N 'DO'
'FOR' J:=1 'STEP' 1 'UNTIL' N 'DO'
'BEGIN'
  YT[I,J]:=NN[J]*YT[I,J]/SS2C[I];
  YTINVERS[I,J]:= YTINVERS[I,J]*SS1C[J];
'END';

'FOR' I:=1 'STEP' 1 'UNTIL' N 'DO'
'FOR' J:=1 'STEP' 1 'UNTIL' N 'DO'
'BEGIN'
  S:=0;
  'FOR' K:=1 'STEP' 1 'UNTIL' N 'DO'
    S:= S+YT[I,R]*YTINVERS[R,J];
    C0[I,J]:= S;
  'END';
'END' PROCEDURE C0;

=====
REAL' 'PROCEDURE'SIMPSON(I,J,X,Y,FXY,A,B,N,C,D,M)
'VALUE'A,B,N,C,D,M'REAL'X,Y,FXY,A,B,C,D'INTEGER'I,J,N,M
'BEGIN''INTEGER'F1,F2'REAL'H,X,Y,S,SUM

HXI=(B-A)/N;HYI=(D-C)/M;F1:=4;SUM:=0;
'FOR' J:=0 'STEP' 1 'UNTIL' M
'DO' 'BEGIN' F1:='IF' F1='EQL' 4 'THEN' ('IF' J='EQL' 0 'OR' J='EQL' M 'THEN' 1 'ELSE'
2)'ELSE' 4;
```

```
Y1=C+J*HY;I:=0;X1=A$S1=FXY3;I=N;X1=B$S1=S+FXY3;F21=2;
'FUR' I:=N-1'STEP'=1'UNTIL'1
'DO' BEGIN X:=A+I*HX;F21='IF' F2'EQL'2'THEN'4'ELSE'2'
      S1=S+F2*FXY3
    END'
    SUM1=SUM+F1*S1
  END'
  SIMPSON1=SUM*HX*HY/9;
END'SIMPSON

=====
'FUR' TELLER1=1 'STEP' 1 'UNTIL' 2 'DO'
'BEGIN'
  'BEGIN'
    'ARRAY' BA, BB, A1, A2, C0, LU [1:N,1:N]
      FC, FSC [1:N,0:M],
      AA, U [0:N,0:N],
      EETA, EKSI, HETAA, HKSI [0:N,KSI>0:NETA],
      RTR, RTI [0:N],
      B, X, Y, NORMCE, NORMSE, SS1C, SS2C [1:N],
      F1, F2 [0:M],
      JJ [0:2*M+2],
      SI, CO [0:2*M+2]

    'REAL' EXC1, KSI1, NORM1, S1,
           EXC2, KSI2, NORM2, S2,
           QC, KSI, NORMS, S3,
           QO, ETAA, NORMC, S4,
           SQRTQC, DELKSI, FIELD1, S,
           EIGENV, DELETA, FIELD2, CT,
           BETAPUNCT, THETA, FIELD3, ST,
           KH, DELTHETA, FIELD4, DELTA,
           KA1, THETAMAX, HULP,
           M, PI2;

    'INTEGER' 'ARRAY' P[0:N]
    'BOOLEAN' SINGULAR;
    'INTEGER' I, J, R, K21;
    'LABEL' EXIT;

EXC1= 'CASE' TELLER=1 'OF' ( 0.876847, 0.876847 );
EXC2= 'CASE' TELLER=1 'OF' ( 0.8, 0.8 );
ZEROINAB(EXC1*COSH(KSI1)-1,KSI1=0,10,E-10);
ZEROINAB(EXC2*COSH(KSI2)-1,KSI2=0,10,E-10);

PI2= 8*ARCTAN(1);
DELKSI= KSI1/NKSI;
DELETA= 8*ARCTAN(1)/NETA;

THETAMAX= ARCTAN(1)*4;
DELTHETA= THETAMAX/NTHETA;

QC= 'CASE' TELLER=1 'OF' ( 2.9, 2.9 );
SQRTQC= SQRT(QC);
EIGENV= 'CASE' TELLER=1 'OF' (
  1.214044868, 1.19205267642 );
QO= QL*EIGENV/(EIGENV-1);
KA1= 2*SQRT(QO)/EXC1;
KH= KA1*EXC1;

WHITE</>, "EXC1=", E18.11, "X5", "KSI1=", E18.11, /
  "EXC2=", E18.11, /
  "QC_=", E18.11, /
  "LAMBDA=", E18.11, /
  "QO____=", E18.11, /
  "BETAPUNCT=", E18.11, /
  "KA1_____=", E18.11, /
  "KA2_____=", E18.11, /, EXC1, KSI1, EXC2, QC,
  EIGENV, QO,
  1/SQRT(EIGENV)*2*SQRT(QO)/EXC1, 2*SQRT(QO)/EXC2);

=====
FILLBA (QC,KSI1,N,BA,M,FC,FSC,BB,NORMCE,NORMSE,SS1C,SS2C);
FILLCO(QO,KSI1,KSI2,N,C0,M,FSC,SS1C,SS2C);
```

```

FOR I=1 STEP 1 UNTIL N DO
FOR J=1 STEP 1 UNTIL N DO
BEGIN
    A1[I,J]:= -CD[I,J]
    IF I = EQL J THEN A1[I,J]:= A1[I,J]+1
END;
CROUTDECOMPOSITION(1,N,A1,LU,P,MACHEPS,SINGULAR);
IF SINGULAR THEN BEGIN
    WRITE(OUTPUT,</"SINGULAR=IN= (I-CD)">); 'GOTO' EXIT;
END;
CROUTINVERSE(1,N,LU,P,A2);

FOR I=1 STEP 1 UNTIL N DO
FOR J=1 STEP 1 UNTIL N DO
A1[I,J]:=BA[I,J]-CD[I,J];

FOR I=1 STEP 1 UNTIL N DO
FOR J=1 STEP 1 UNTIL N DO
BEGIN
    S:=0;
    FOR R=1 STEP 1 UNTIL N DO
    S:= S+A1[R,I]*A2[R,J];
    A1[I,J]:=BA[I,J]+S;
END;

xx-----xx
xx DE OORSPRONELIJKE MATRIX BA IS NU VERNIETIGD
xx IN A1 STAAT (BA-CD)
xx IN A2 STAAT (I-CD)***1
xx-----xx

UNSMEIGENVECTORS(N,AA,RTR,RTIAU);

DELTAI=ABS(EIGENV=RTR[1]);

II:=1;
FOR I=2 STEP 1 UNTIL N DO
BEGIN
    MULP:= ABS(EIGENV=RTR[1]);
    IF MULP <= DELTA 'THEN' BEGIN
        DELTAI= MULP;
        II= I;
    END;
END;

IF RTI[II] = NEQ 0
'THEN' BEGIN
    WRITE(OUTPUT,</"RTR=",X10,"RTI=",>,
          2(E18.11*X10),RTI[II],RTI[II]);
    'GOTO' EXIT;
END;
ELSE BEGIN
    BETAPONTI= SQRT(1/RTH[II]);
    WRITE(OUTPUT,</"BETAPONT=", E18.11,>,
          BETAPONT);
END;

FOR II=1 STEP 1 UNTIL N DO B[II]:= U(I,II);
xx-----xx
xx DE EIGENVECTOR IS NU BEKEND
xx-----xx

WRITE(OUTPUT,</"X5,"RTR",X10,"RTI",X10,"EIGENVECTOR">);
FOR I=1 STEP 1 UNTIL N DO
BEGIN
    WRITE(OUTPUT,</"E18.11>,RTI[I]);
    IF I = EQL II 'THEN' WRITE(OUTPUT,<"<<<">)
    ELSE WRITE(OUTPUT,<"X3">);
    WRITE(OUTPUT,<E18.11*X3/E18.11>,RTI[I],B[II]);
END;

FOR I=1 STEP 1 UNTIL N DO
BEGIN
    S:=0;
    FOR J=1 STEP 1 UNTIL N DO
    S:= S+A2[I,J]*B[J];
    Y[I]:= S;
END;
B VECTORS;

```

```

'FOR'II=1'STEP'1'UNTIL'N'DO'
'BEGIN'
  S:=0;
  'FOR' JJ=1 'STEP' 1 'UNTIL' N 'DO'
    S:= S+BA[I,J]*Y[J];
    X[I]:= BETAPINT*S;
    BA[I,I]:= BA[I,I]-RT[R][I];
  'END' A VECTOR;
  WRITE(OUTPUT,<///"REL.ERROR">E10.3>x1>"x">, DELTA/
        EIGENV*100 );

  WRITE(OUTPUT,<//"/1",X10, "NULLVECTOR">);
  'FOR'II=1'STEP'1'UNTIL'N'DO'
  'BEGIN'
    S:=0;
    'FOR' JJ=1 'STEP' 1 'UNTIL' N 'DO'
      S:= S+BA[I,J]*Y[J];
    WRITE(OUTPUT,</>I3>X8,E18.11>, I> S);
  'END';
  XX-----;

  WRITE(OUTPUT[SKIP(1)]);
  'FOR' I:=0'STEP'1'UNTIL'NKSI
  'DO' 'BEGIN' KSI:=I*DELKSI;'IF' I'EQL'0 'THEN' KSI:=#-6;
    JAPLUSNC(2+SQRTGC *SINH(KSI))>0,2*M+2*10,JJ);
    WRITE(OUTPUT[SPACE(3)]);
  WRITE(OUTPUT,<"KSI,ETA,X,Y,EETA,EKSI,HETA,HKSI=">);
  'FOR' JJ=0'STEP'1'UNTIL' (NETA 'DIV' 4 )
  'DO' 'BEGIN' ETA:=J*WELETA;
    SINCOS(ETA,M,S1,CD);
    S1:=S2:=S3:=S4:=0;
    'FOR' R:=1'STEP'1'UNTIL'N
    'DO' 'BEGIN' 'FOR' K:=0'STEP'1'UNTIL'M
      'DO' 'BEGIN' F1[K]:=FC[R,K];
        F2[K]:=FSC[R,K];
      'END';
      NORMS:=NORMSE[R];
      NORM:=NORMCE[R];
      S3:=S3+X[R]*(M=CCE2N1(1,M,F1,JJ))*-
        COSH(KSI)/SINH(KSI)*
        CE2N1FCINT(NORMC,M,F1,S1));
      S1:=S1+X[R]*(CCE2N1FCINT(1,M,F1,JJ)*2*
        SQRTGC *COSH(KSI)**2/SINH(KSI)-H*
        SINH(KSI)**2)*CE2N1(NORMC,M,F1,CD));
      S2:=S2+Y[R]*SSE2N1(1,M,F2,JJ)*
        SE2N1PCINT(NORMS,M,F2,CD));
      S4:=S4+Y[R]*SSE2N1POINT(1,M,F2,JJ)*2*
        COSH(KSI)*SQRTGC *SE2N1(NORMS,M,F2,
        SI));
    'END';
    FIELD1:= EETA[I,J]:= S1-BETAPINT*S2;
    FIELD2:= EKSI[I,J]:= S3-BETAPINT*S4;
    FIELD3:= HETA[I,J]:= -BETAPINT*S3-S4;
    FIELD4:= HKSI[I,J]:= -BETAPINT*S1+S2;

    EETA[I, NETA 'DIV' 2-J]:= -FIELD1;
    EKSI[I, NETA 'DIV' 2-J]:= FIELD2;
    HETA[I, NETA 'DIV' 2-J]:= FIELD3;
    HKSI[I, NETA 'DIV' 2-J]:= -FIELD4;

    EETA[I, NETA 'DIV' 2+J]:= -FIELD1;
    EKSI[I, NETA 'DIV' 2+J]:= -FIELD2;
    HETA[I, NETA 'DIV' 2+J]:= -FIELD3;
    HKSI[I, NETA 'DIV' 2+J]:= -FIELD4;

    EETA[I,NETA-J]:= FIELD1;
    EKSI[I,NETA-J]:= -FIELD2;
    HETA[I,NETA-J]:= -FIELD3;
    HKSI[I,NETA-J]:= FIELD4;

    WRITE(OUTPUT,</>2F8.4>x3>2F8.4>x4(x2,E18.11)>+
      KSI,ETA,
      COSH(KSI)*COS(ETA),SINH(KSI)*SIN(ETA),
      EETA[I,J],EKSI[I,J],HETA[I,J],HKSI[I,J]);
  'END'ETA;
  'END'KSI;

```

```
      WRITE(OUTPUT,SKIP(1)); WRITE(OUTPUT,<"STRALINGSDIAGRAM">)
      WRITE(OUTPUT,</>, "THETA",X10, "FI=0-PLANE",X10,
            "FI=90-DEGREES-PLANE"> )
      'FOR'KI=0'STEP'2'UNTIL'NTHETA
      'DO''BEGIN'THETA:=R*DELTHTETA;
          CT:=COS(THETA); ST:=SIN(THETA);
          S3:=SIMPSUN(I,J,KSI,ETA,(SINH(KSI)*COS(ETA)*(HKSI[I,J]
              -CT*EETA[I,J]))+COSH(KSI)*SIN(ETA)*(+HETA[I,J]-
              CT*EKSI[I,J]))*COS(KH*COSH(KSI)*COS(ETA)*ST),
          0,KSI1,NKSI,0,PI2,NETA);
          S4:=SIMPSUN(I,J,KSI,ETA,(SINH(KSI)*COS(ETA)*(HKSI[I,J]
              -CT*EETA[I,J]) +COSH(KSI)*SIN(ETA)*(+HETA[I,J]-
              CT*EKSI[I,J]))*SIN(KH*COSH(KSI)*COS(ETA)*ST),
          0,KSI1,NKSI,0,PI2,NETA));
          S1:=S3**2+S4**2;
          'IF'R'EQL'0'THEN'NORM1:=S; S1:=10*LOG(S/NORM1);
          WRITE(OUTPUT,[SPACE(1)]);
          WRITE(OUTPUT,</>,I3,X3,F12,4>,4*R,THETA );
          WRITE(OUTPUT,<X5,E18,11>,S);
          S1:=SIMPSUN(I,J,KSI,ETA,(SINH(KSI)*COS(ETA)*(EETA[I,J]
              -CT*HKSI[I,J]) +CUSH(KSI)*SIN(ETA)*(+EKSI[I,J]+
              CT*HETA[I,J]))*COS(KH*SINH(KSI)*SIN(ETA)*ST),
          0,KSI1,NKSI,0,PI2,NETA);
          S2:=SIMPSUN(I,J,KSI,ETA,(SINH(KSI)*COS(ETA)*(EETA[I,J]
              -CT*HKSI[I,J]) +COSH(KSI)*SIN(ETA)*(+EKSI[I,J]+
              CT*HETA[I,J]))*SIN(KH*SINH(KSI)*SIN(ETA)*ST),
          0,KSI1,NKSI,0,PI2,NETA);
          S1:=S1**2+S2**2;
          'IF'R'EQL'0'THEN'NORM2:=S; S1:=10*LOG(S/NORM2);
          WRITE(OUTPUT,<X5,E18,11>,S);
          'END';
          EXIT;
      'END';
      'END' TELLER;
      'END';
      'END'.
```

RADABEF

```
& 'SET' SEQ
& 'SET' THELIBRARY
'BEGIN'
'FILE' OUTPUT, PUNCH

'REAL' RADABEF;
'INTEGER' NMAX, NETAKNSI, THEFA, TELLER;

-----
'PROCEDURE' KURVEIGENVECTORS (NPAR, RTR, RTI);;
'VALUE' N 'INTEGER' N;
'ARRAY' APLEX(0:N), RTR(0:N) ;;
'EXTERNAL';

-----
RADABEF:=**(-36);
N:=4;
M:= N+10;
NKS1:= 10;
NETA:= 100;
NTHETA:= 45;

-BEGIN
6

'FOR' TELLER:=1 'STEP' 1 'UNTIL' 6 'DO'
'BEGIN'
'BEGIN'
'ARRAY' A1, A2, E1, LU [1:N,1:N];
ECC,FSC [1:N,0:M];
A1, A2, E1, LU [0:N,0:M];
EKS1, EKSI, NETAY, RKS1 [0:N];
RTR, RTI [0:N];
B, X, Y, NORM1, NORM2, CC1C, CC2C [1:N];
F1, F2 [0:N];
UJ [0:2*M];
S1, C1 [0:2*M+2];
S2, C2 [0:2*M+2];
S3, C3 [0:2*M+2];
S4, C4 [0:2*M+2];
FIELD1, FIELD2, FIELD3, FIELD4 [0:N];
DELTS1, DELTS2, DELTS3, DELTS4 [0:N];
DELETAS, DELTAY, DELTHETA [0:N];
KAL, THETANAXX [0:N];
HULP, H [0:N];
P12 [0:N];

'REAL' EXC1, KSI1, KSI2, NORM1, S1, S2, S3, S4, CT, ST, DELTA;
EXC2, KSI1, KSI2, NORM2, S2, S3, S4, CT, ST, DELTA;
OC, KSI1, KSI2, NORM1, NORM2, S1, S2, S3, S4, CT, ST, DELTA;
OC1, ETAY, DELKSI1, FIELD1, S1, S2, S3, S4, CT, ST, DELTA;
OC2, DELKSI2, FIELD2, S2, S3, S4, CT, ST, DELTA;
SNHTNU, DELTS1, FIELD3, S3, S4, CT, ST, DELTA;
EIGENV, DELETAS, FIELD4, S4, CT, ST, DELTA;
BETAPUNCT, THETAY, DELTHETA, HULP, H, P12;
KAL, THETANAXX, HULP, H, P12;

'INTEGER' IARRAY1, P10:N);

'BOOLEAN' SINGULAR;
'INTEGER' I, J, K, L;
'LABEL' EXITA;

EXC1:= 'CASE' TELLER=1 'OF'
0.676847, 0.876847, 0.876847, 0.576847,
0.676847, 0.876847 'OF';
EXC2:= 'CASE' TELLER=1 'OF'
0.8*0.8*0.8*0.8,
0.8*0.8 'OF';
ZERGINAR(XC1*COSH(KS11)*1, KSI1, 0, 10, e-10, e-10);
ZERGINAR(XC2*COSH(KS12)*1, KSI2, 0, 10, e-10, e-10);

P12:= 8*ARCTAN(1);
DELKS1:= KSI1/NKS1;
DELETAS:= 8*ARCTAN(1)/NETA;

THEFAXX:= ARCTAN(1)*4;
DELTAY:= THEFAXX/RTHETA;
```

```
SC:= 'CASE' TELLER-1 'OF'  
2.904, 2.97, 2.383, 2.89, 2.698, 2.902 )  
EIGENV:= 'CASE' TELLER-1 'OF' 1  
1.15413352201, 1.06191259956, 1.09530709883,  
1.09857160911, 1.13523237224, 1.14828418141 )  
SQRT(GC):= SQRT(GC)  
W0:=GC*EIGENV/(EIGENV-1)  
KA1:= 2*SQRT(W0)/EXC1  
KHC:= KA1*EXC1  
  
WRITE(Output, <>, 'EXC1=', E18.11, 'X5, "KSII=", E18.11, /,  
'EXC2=', E18.11, /,  
'EXC3=', E18.11, /,  
'LAYBDA=', E18.11, /,  
'CC_____=', E18.11, /,  
'BETAPINT=', E18.11, /,  
'RAI_____=', E18.11, /,  
'RAZ_____=', E18.11, />, EXC1, KSII, EXC2, GC,  
EIGENV, /, 0,  
1/SQRT(EIGENV), 2*SQRT(GC)/EXC1, 2*SQRT(GC)/EXC2)  
  
*****  
FILLAU(GC, KSII, N, 48, FCL, FSC, 22, AUGHYCE, NORMSE, CC1C, CC2C);  
FILLER(GC, KSII, KSIZ, N, EF, FCC, CC1C, CC2C);  
  
*FOR 'I:=1' STEP '1' UNTIL 'N' DO  
*FOR 'J:=1' STEP '1' UNTIL 'N' DO  
*BEGIN  
A11,I,J:= -EF(I,J);  
*IF 'I' EQUAL 'J' THEN A11,I,J:= A11,I,J+1;  
*END;  
  
CRDUTDECOMPOSITION(1..N, A11, LU, P, NACHERP, SINGULAR);  
*IF 'SINGULAR' THEN *BEGIN  
WRITE(Output, <>, "SINGULAR-IN-",  
'(I=EF)">); GOTO EXIT;  
*END;  
CRDUTINVERSE(1..N, LU, P, A2);  
  
*FOR 'I:=1' STEP '1' UNTIL 'N' DO  
*FOR 'J:=1' STEP '1' UNTIL 'N' DO  
A11,I,J:= ABL1,I,J-EF(I,J);  
  
*FOR 'I:=1' STEP '1' UNTIL 'N' DO  
*FOR 'J:=1' STEP '1' UNTIL 'N' DO  
*BEGIN  
S:=0;  
*FOR 'R:=1' STEP '1' UNTIL 'N' DO  
S:= S+A11,I,R*A2(R,J);  
A11,I,J:= ABL1,I,J:= S;  
*END;  
  
*****  
* DE OORSPRUNELIJKE MATRIX AB IS NU VERNIETIGD  
* IN A1 STAAT (AE-EF)  
* IN A2 STAAT (I-EF)***  
*  
*****  
  
KORSYMEIGENVECTORS(N, A11, RTR, RTI, L)  
  
DELTA:=ABS(EIGENV-RTR(I))  
II:=1;  
*FOR 'I:=2' STEP '1' UNTIL 'N' DO  
*BEGIN  
HULP:= ABS(EIGENV-RTR(I))  
*IF 'HULP' LESS 'DELTA' THEN *BEGIN  
DELTA:= HULP;  
II:= I;  
*END;  
*END;  
  
*IF 'RTI' LESS '0'  
*THEN *BEGIN  
WRITE(Output, <>, 'RTR=', X10, "RTI=", X10, /,  
2(E18.11*X10), RTR(II), RTI(II))  
GOTO EXIT;  
*END;  
*ELSE *BEGIN
```

```

      RETAPINT:= SIN(1/RTA[1]) );
      WRITE(OUTPUT, < />, "RETAPCIN=" , E18.11, /> );
      RETAPINT );
      END;

      FOR I:=1 'STEP' 1 'UNTIL' N DO
      BEGIN
        % DE EIGENVECTOR IS NU BEKEND
        %
        WRITE(OUTPUT, < />, AS, "RT", X10, "RT1", X10, "EIGENVECTOR">> );
        'FOR' I:=1 'STEP' 1 'UNTIL' N DO
        'BEGIN'
          WRITE(OUTPUT, < />, E18.11, RT[1] );
          'IF' I 'EQL' 1 'THEN' WRITE(OUTPUT, <<<> )
            'ELSE' WRITE(OUTPUT, <<3> );
          WRITE(OUTPUT, < E18.11, X3, E18.11, RT[1], B11 ) ;
        'END';

        'FOR' I:=1 'STEP' 1 'UNTIL' N-1;
        'BEGIN'
          S:=0;
          'FOR' J:=1 'STEP' 1 'UNTIL' N 'DO'
            S:= S+RT[I,J]*X[J];
          X[I]:= S;
        'END' A VECTOR;
      'END';

      'FOR' I:=1 'STEP' 1 'UNTIL' N DO
      'BEGIN'
        S:=0;
        'FOR' J:=1 'STEP' 1 'UNTIL' N 'DO'
          S:= S+RT[I,J]*X[J];
        Y[I]:= RETAPINT*S;
        A[1,I]:= A[1,I]+RT[1,I];
      'END' B VECTOR;

      WRITE(OUTPUT, < />, "REL*ERROR", X10, "X", "X" );
      DELTA/
      EIGENV*100 );

      WRITE(OUTPUT, < />, "1", X10, "NULLVECTOR">> );
      'FOR' I:=1 'STEP' 1 'UNTIL' N DO
      'BEGIN'
        S:=0;
        'FOR' J:=1 'STEP' 1 'UNTIL' N 'DO'
          S:= S+RT[I,J]*X[J];
        WRITE(OUTPUT, < />, X3, E18.11, 1, S );
      'END';
      %

      WRITE(OUTPUT[SKIP(I)]);
      'FOR' I:=1 'STEP' 1 'UNTIL' NKS1
      'DO' 'BEGIN' PS1:=I*DELKSI; 'IF' I 'EGL' 0 'THEN' KS1:=#-6;
      S1:=PLUSK2*SQRTOC *SINH(KS1),0,2*N+2,10,JJ);
      WRITE(OUTPUT, < />, "KS1", ETAX, Y, EETA, EKSI, RETA, FKSI="">> );
      'FOR' J:=1 'STEP' 1 'UNTIL' (RETA 'DIV' 4 )
      'DO' 'BEGIN' RETA:=J*DELTAS;
      SINCOS(ETA,M,S1,C0);
      S1:=S2:=S3:=S4:=C0;
      'FOR' R:=1 'STEP' 1 'UNTIL' N
      'DO' 'BEGIN' 'FOR' K:=0 'STEP' 1 'UNTIL' N
      'DO' 'BEGIN' F1[K]:=FUC(R,K);
      F2[K]:=FSCL(R,K);
      'END';
      NURMS:=NURMS+E[R];
      NURMC:=NURMC+E[R];
      S3:=S3+X[R]*(H:=(CE2N1(1,M,F1,JJ))*_
      CUSH(KS1)/SINH(KS1)*_
      CE2N1PCINT(NURMC,M,F1,SI));
      S1:=S1+X[R]*(CC2N1PCINT(1,M,F1,JJ)*2*
      SWHTG *CUSH(KS1)*2/SINH(KS1)*H*
      SINH(KS1)*2)*CE2N1(NURMC,M,F1,C0);
      S2:=S2+Y[R]*SSE2N1(1,M,F2,JJ)*_
      SE2N1PCINT(NURMS,M,F2,C0);
      S4:=S4+Y[R]*SSE2N1PCINT(1,M,F2,JJ)*2*
      CUSH(KS1)*SWHTG *SE2N1(NURMS,M,F2,JJ);
      'END';
      S:=S1+S2+S3+S4;
      END;
    
```

```

FIELD01:= EKSI[I,J]:= BETAPoint*S1+S2;
FIELD02:= HETA[I,J]:= S1-BETAPoint*S2;
FIELD03:= HKSII[I,J]:= S3-BETAPoint*S4;

    EТАL1, NETA 'DIV' 2-JJ:= FIELD01;
    EКSII, NETA 'DIV' 2-JJ:= -FIELD02;
    EТАL1, NETA 'DIV' 2-JJ:= -FIELD03;
    HKSII, NETA 'DIV' 2-JJ:= FIELD04;

    EТАL1, NETA 'DIV' 2+JJ:= -FIELD01;
    EКSII, NETA 'DIV' 2+JJ:= -FIELD02;
    EТАL1, NETA 'DIV' 2+JJ:= -FIELD03;
    HKSII, NETA 'DIV' 2+JJ:= -FIELD04;

    EТАL1, NETA -JJ:= -FIELD01;
    EКSII, NETA -JJ:= FIELD02;
    EТАL1, NETA -JJ:= FIELD03;
    HKSII, NETA -JJ:= -FIELD04;

    WRITE(OUTPUT,</>2F8.4>x3>2F8.4>4(X2,E18.11)>;
          KSI,ETA,
          CUSH(KSI)*COS(ETA),SINH(KSI)*SIN(ETA),
          EETA[I,J],EKSI[I,J],HETA[I,J],HKSII[I,J]);
      *END'ETA';
  *END'KSI';

WRITE(OUTPUT[SKIP(1)])>WRITE(OUTPUT,<"STRALINGSDIAGRAM">);
WRITE(OUTPUT,</>, "THETA",X10, "FI=0-PLAYE",X10,
      "FI=90-DGREES-FLANE">;)

*FOR'KSI'STEP'2'UNTIL'NTHETA'
*00'BEGIN'THETA:=R*DELTTHETA;
    CT:=COS(THETA); ST:=SIN(THETA);
    S1:= S1*PSIN(I,J,KSI,ETA,(SINH(KSI)*COS(ETA)*(EKSI[I,J]
      +CT*EТАL1,I,J))+CUSH(KSI)*S1*(ETA)*(-EETA[I,J]+
      CT*HKSII[I,J]))*COS(KH*COSH(KSI)*COS(ETA)*ST),
      0*KSI1*AKSI1,0,PI2,NETA);
    S2:=S1*PSIN(I,J,KSI,ETA,(SINH(KSI)*COS(ETA)*(EKSI[I,J]
      +CT*EТАL1,I,J))+CUSH(KSI)*S1*(ETA)*(-EETA[I,J]+
      CT*HKSII[I,J]))*SIN(KH*COSH(KSI)*COS(ETA)*ST),
      0*KSI1*AKSI1,0,PI2,NETA);
    S:=S1**2+S2**2;
    'IF'R'EQL'0'THEN'NCHM1:=S;S1=10*LOG(S/NORM1);
    WRITE(OUTPUT[SPACE(1)])>;
    WRITE(OUTPUT,</>13,x3,F12.4>4*h,THETA );
    WRITE(OUTPUT,<x5,E18.11>S);
    S3:=S1*PSIN(I,J,KSI,ETA,(SINH(KSI)*COS(ETA)*(HETA[I,J]
      +CT*EКSII[I,J])+CUSH(KSI)*S1*(ETA)*( HKSII[I,J]-
      CT*EETA[I,J]))*COS(KH*SINH(KSI)*SIN(ETA)*ST),
      0*KSI1*AKSI1,0,PI2,NETA);
    S4:=S1*PSIN(I,J,KSI,ETA,(SINH(KSI)*COS(ETA)*(HETA[I,J]
      +CT*EKSI[I,J])+CUSH(KSI)*S1*(ETA)*( HKSII[I,J]-
      CT*EETA[I,J]))*SIN(KH*SINH(KSI)*SIN(ETA)*ST),
      0*KSI1*AKSI1,0,PI2,NETA);
    S:=S3**2+S4**2;
    'IF'R'EQL'0'THEN'NCHM2:=S;S1=10*LOG(S/NORM2);
    WRITE(OUTPUT,<x5,E18.11>S);
  *END';
  EXIT;
*END';
*END'ITLELLER;
*END';
*END';

```

- [1] J.K.M. Jansen and M.E.J. Jeuken  
Propagation and radiation properties of elliptical waveguide  
with anisotropic boundary  
ESTEC contract No 1657/72HP
- [2] L.F.G. Thurlings  
Elliptical waveguide as a feed for circularly polarised waves  
Afstudeerverslag ET-2-1975
- [3] Burroughs information BU3/1430/1431  
Library, Dent. of Mathematics, group Numerical Analysis  
University of Technology
- [4] Al-Harriri  
Low attenuation microwave waveguides  
Q.M.C. London 1974
- [5] J.H. Wilkinson  
The algebraic eigenvalue problem  
Clarendon Press, Oxford 1968
- [6] M.E.J. Jeuken and L.F.G. Thurlings  
The corrugated elliptical horn antenna  
IEEE Symposium 1975, Illinois
- [7] M.E.J. Jeuken  
Frequency-independence and symmetry properties of corrugated  
conical horn antennas with small flares  
Thesis 1970, Eindhoven University of Technology
- [8] M.J. Al-Hakkak and Y.T. Lo  
Circular waveguides and horns with anisotropic and corrugated  
boundaries  
Antenna Laboratory, University of Illinois