APPENDIX H. COMPUTER PROGRAMMES for arXiv.org:1109.4732v3 [hep-th] 25 Jun 2012

We list below the different computer programmes used to calculate our results.

1. The iterative integration in Comal. Sturm-Liouville theory for the iterative integration
2. Collocation in Matlab
3. Parametric function generation in Maple
4. Eigenvalue determination in Mathcad

References

[H1] E. Hansen, *Sædvanlige differentialligninger fra fysikken*, Polyteknisk Forlag, (Lyngby 1971) pp. 86.
[H2] E. Kreyszig, *Advanced Engineering Mathematics*, 9th ed., (John Wiley and Sons Inc., Singapore 2006), pp.203.
[H3] K. F. Riley, M. P. Hobson and S. J. Bence, *Mathematical Methods for Physics and Engineering*, (Cambridge University Press, Cambridge 1998), pp. 485.

The Programmes

1. The iterative integration in Comal. Sturm-Liouville theory for the iterative integration

In order to find the eigenvalues of the parametric one-dimensional Schrödinger equation (32) we first exploited Sturm-Liouville theory [H1, H2, H3]. The Sturm-Liouville problem is defined in the following way:

Definition: Sturm-Liouville's problem is an eigenvalue problem defined on an interval [a,b] and given by a differential equation of the form

$$(py')' + (\lambda r - q)y = 0 \qquad \text{(H1)}$$

and two boundary conditions which can be

$$\left.\begin{array}{r} y(a)=0 \\ \text{or}\;\; y'(a)-\alpha y(a)=0 \end{array}\right\} \text{ and } \left\{\begin{array}{r} y(b)=0 \\ \text{or}\;\; y'(b)+\beta y(b)=0 \end{array}\right.$$
$$\text{or} \qquad \text{(H2, H3)}$$
$$y(a)=y(b) \qquad \text{and} \qquad p(a)y'(a)=p(b)y'(b)$$

The latter requirement (H3) is called periodic boundary conditions. In equation (H1) the coefficient functions *p*, *r* and *q* are real, $p \in C^1$, r and $q \in C^0$ and $p(x)$ and $r(x) > 0$ for all $x \in [a,b]$. In the boundary conditions the constants α and β are real and independent of λ.

One readily verifies that (32) is a Sturm-Liouville problem. Thus we may exploit for iteration the following

Theorem: There exists a countable, infinite set of eigenvalues $\lambda_1 < \lambda_2 < ...$ The spectrum of eigenvalues is bounded from below and without any points of convergence, wherefore

$\lambda_n \to \infty$ for $n \to \infty$. The zeropoints of an eigenfunction φ_n, belonging to the eigenvalue λ_n, devides the interval of definition in exactly *n* partial intervals in which φ_n has a constant sign.

We use this theorem to squeeze our guesses for the eigenvalues in the iterations, carry out a numerical integration for that particular guess while keeping track on the number of zeros. Then we adjust the eigenvalue up or down accordingly for a new iteration to come closer to the required number of partial intervals for the eigenvalue in search. Once the boundary conditions are met within a prescribed precision the iteration stops, the eigenvalue is stored and a guess is constructed for the next eigenvalue in line. This next eigenvalue will be bounded from below by its predecessor and bounded from above by the eigenvalue corresponding to a constant potential equal to the maximum value $\frac{1}{2}\pi^2$ of the actual potential. The Comal program doing the iteration is called "backfit" and is exposed below. Note a technicality: Starting out the integration far from an actual eigenvalue may lead to divergencies. The program tackles this problem using three ideas. 1) We calculate the classical turning point in the actual potential and use the fact that the eigenstate is only oscillatory between the turning points and therefore will show all its zeros here. 2) Whenever the state crosses through a zero the product of two successive functional values will be negative. This simple fact is used to count the number of zeros during the integration which is a 4th order Runge-Kutta method. 3) Once the correct number of zeros has been reached the eigenvalue is fine tuned by integrating "backwards" from $\theta = \pi$ "under" the potential. If the function does not meet the boundary conditions at the origo $\theta = 0$, the eigenvalue is adjusted accordingly. For odd-label states with periodicity 2π this is done by keeping track on the derivative which should be zero at the origo whereas for even-label states the function itself should have a zero at origo.

The file "ComalIterationSturmLiouville.tif" shown below contains a scan of a print of the programme itself and a scan of an output of the comal programme "BACKFIT.cml". The comal programme dates back to the end of the eighties, beginning of nineties.

The Danish text says "Kind regards Ole Trinhammer...Save the print for me..." as the output was written to a public printer. In line 0085 to 0086 of the programme it says: "The machine is calculating, dear collegue...Ready by 'programme terminated' - please save the print for OT"

The outputs are parametric eigenvalues, i.e. eigenvalues for the one-dimensional Schrödinger equation with approximate potential. The eigenvalues are used in the programme "MapleParametricFunctionGeneator.mw" to generate the parameter functions used in the programme "MathcadParametricBasis.mcd" to calculate eigenvalues for the exact 3-dimensional problem from the allospatial Hamiltonian.

The iteration is started off by a guess from the general trend in eigenvalues known from Sturm-Liouville theory and the theory of the harmonic oscillator, see e.g. [59]. The individual integrations are started off with suitable boundary conditions according to the symmetry of the level sought. During the integration we keep track on the number of oscillations specific to the level sought for. If the number of oscillations is too large, the eigenvalue is adjusted downwards. Once the correct number of zeros has been reached the eigenvalue is fine-tuned by integrating backwards through the integration interval, i.e. from Pi back to 0, where we check that the boundary condition is correct. If not, the eigenvalue is adjusted accordingly.

Here comes the scan of Backfit.cml. The last page shows two printouts.

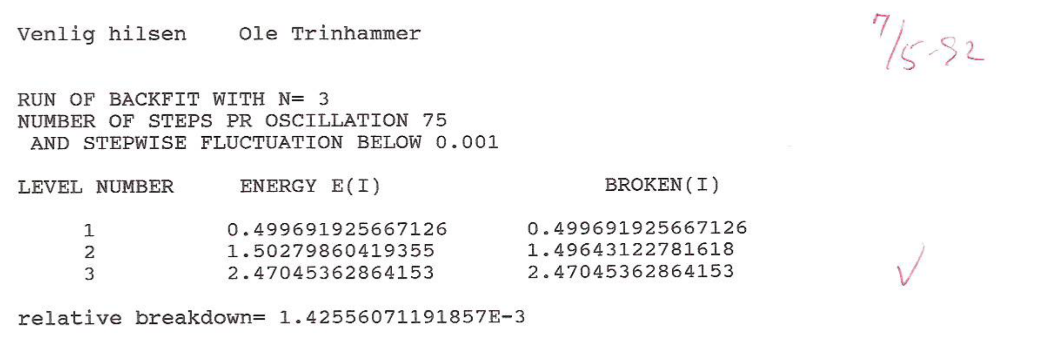
```
0010 //
0020 // BACKFIT
0030 //
0040 PRINT CHR$(12)
0050 INPUT "N= ": n
0055 INPUT "NUMBER OF LEVELS TO CALCULATE= ": m
0060 INPUT "ITERATION STOPS WHEN FLUCTUATION IS BELOW EPSILON= ": epsilon
0070 INPUT "INTEGRATIONSTEP PR OSCILLATION= ": skridt
0080 PRINT "N= ";n;"epsilon= ";epsilon;"SKRIDT/OSC (N*SK)= ";n*skridt
0084 PRINT
0085 PRINT "MASKINEN REGNER KÆRE KOLLEGA"
0086 PRINT "Færdig ved  'program afsluttet' - gem venligst udskrift til OT"
0087 PRINT
0088 PRINT
0090
0100 DIM e(m)
0110 DIM broken(m)
0120
0130
0140
0150 // BARE LINES CARRY FILING STRUCTURE FOR INTERRUPTED CALCULATIONS
0160
0170
0180 l:=n //                              l=g^2*N is a lattice reminiscence
0190 lc:=PI^2/(2*SQR(2))
0200 //
0209 // EIGENVALUE GUESS FROM A SUITABLY WEITHTED AVERAGE OF HARMONIC OSCILLATOR
0210 // AND STURM-LOUIVILLE THEORY FOR ZERO POTENTIAL
0220 //
0250 deltae:=2*SQR(2)*lc*lc/(lc*lc+l*l)+4*4*PI*PI*l*l/(lc*lc+l*l)
0260 //          EHARMON          VÆGT           ESTURM          VÆGT
0320 FOR i:=1 TO m DO
0330
0340
0350
0360   //
0370   // EIGENVALUE LIMITED ABOVE FROM STURM-THEORY FOR NEXT FOLLOWING LEVEL
0380   //
0390   eimax:=2*(i+1)*(i+1)*4*PI*PI+PI*PI/2
0400   //
0410   //       ESTURM(NEXT)      + POTMAX
0420   //
0430   IF i>1 THEN
0440     eimin:=e(i-1)
0450   ELSE
0460     eimin:=0
0470   ENDIF
0480   ei:=(eimax+eimin)/2
0500   //
0510   //
0520   //
0530   //
0540   //
0550   //
0590   // Iteration until correct number of nodes
0600   //
0601   zeros:=0
0620   IF i MOD 2=1 THEN
0622     node(ei,eimax,eimin,zeros)
0625     REPEAT
0630       IF 2*zeros>i-1 THEN
0640         eimax:=ei
0650         ei:=(eimax+eimin)/2
0670       ENDIF
0680       IF 2*zeros<i-1 THEN
0690         eimin:=ei
0700         ei:=(eimax+eimin)/2
0720       ENDIF
0722       node(ei,eimax,eimin,zeros)
0725     UNTIL 2*zeros=i-1
```

```
0730     //
0740     // Finetuning via backward integration
0750     //
0770     backfit(ei,eimax,eimin)
0790     e(i):=ei
0800   ENDIF
0802   //
0804   // Iteration of even states
0806   //
0810   IF i MOD 2=0 THEN
0815     node(ei,eimax,eimin,zeros)
0820     REPEAT
0840       IF 2*zeros>i-2 THEN
0850         eimax:=ei
0860         ei:=(eimax+eimin)/2
0870       ENDIF
0880       IF 2*zeros<i-2 THEN
0890         eimin:=ei
0900         ei:=(eimax+eimin)
0910       ENDIF
0912       node(ei,eimax,eimin,zeros)
0915     UNTIL 2*zeros=i-2
0920     //
0922     // Finetuning of even states
0924     //
0930     backfit(ei,eimax,eimin)
0950     e(i):=ei
0960   ENDIF
0970
1030
1040 ENDFOR i
1050
1060 //
1070 //
1080 // THIS PART OF THE PROGRAMME FINDS EIGENVALUES FOR BROKEN LEVELS
1090 //
1100 //
1110 FOR i:=1 TO m DO
1120   IF i MOD 2=1 THEN
1130     broken(i):=e(i)
1140   ELSE
1150     bimax:=e(i)
1160     bimin:=e(i-1)
1170     bi:=(bimax+bimin)/2
1180     zeros:=0
1200
1210
1220
1240
1250
1260
1270     node(bi,bimax,bimin,zeros)
1275     REPEAT
1280       IF 2*zeros>i-2 THEN
1290         bimax:=bi
1300         bi:=(bimax+bimin)/2
1310       ENDIF
1320       IF 2*zeros<i-2 THEN
1330         bimin:=bi
1340         bi:=(bimax+bimin)/2
1350       ENDIF
1450
1460
1470       node(bi,bimax,bimin,zeros)
1480     UNTIL 2*zeros=i-2
1485     backbrok(bi,bimax,bimin)
1490     broken(i):=bi
1500

1510
1520
```

```
1530   ENDIF
1540 ENDFOR i
1545 SELECT OUTPUT "prn"
1550 PRINT
1555 PRINT "Gem udskriften til mig"
1556 PRINT
1557 PRINT "Venlig hilsen    Ole Trinhammer"
1558 PRINT
1560 PRINT
1570 PRINT "RUN OF BACKFIT WITH N= ",n
1571 PRINT "NUMBER OF STEPS PR OSCILLATION ",n*skridt
1575 PRINT " AND STEPWISE FLUCTUATION BELOW ",epsilon
1580 PRINT
1590 PRINT "LEVEL NUMBER      ENERGY E(I)                 BROKEN(I)"
1600 PRINT
1610
1620 FOR i:=1 TO m DO
1630
1640
1650   PRINT TAB(5);i;TAB(17),e(i);TAB(40),broken(i)
1660 ENDFOR i
1670
1672 IF m=3 THEN
1673   break:=(e(2)-broken(2))/(broken(1)+broken(2)+broken(3))
1674   PRINT
1675   PRINT "relative breakdown= ",break
1676 ENDIF
1678 SELECT OUTPUT "con"
1680
1690
1700 //
1710 //
1720 // RUNGE KUTTA 4.ORDEN
1730 //
1740 PROC node(REF ei,REF eimax,REF eimin,REF zeros)
1750   x:=0
1760   IF i MOD 2=1 THEN
1770     y:=1
1780     v:=0
1790   ELSE
1800     y:=0
1810     v:=4*i*skridt
1820   ENDIF
1830   zeros:=0
1850   max:=n*i*skridt
1860   xturn:=SQR(2*ei)
1870   h:=xturn/max
1880   FOR j:=1 TO max+1 DO
1890     k1:=v
1900     l1:=y*(x*x-ei*2)
1910     k2:=v+h*l1/2
1920     l2:=(y+h*k1/2)*((x+h/2)*(x+h/2)-ei*2)
1930     k3:=v+h*l2/2
1940     l3:=(y+h*k2/2)*((x+h/2)*(x+h/2)-ei*2)
1950     k4:=v+h*l3
1960     l4:=(y+h*k3)*((x+h)*(x+h)-ei*2)
1970     yny:=y+h*(k1+2*k2+2*k3+k4)/6
1980     vny:=v+h*(l1+2*l2+2*l3+l4)/6
1985     IF j<=max THEN
1990       IF y*yny<0 THEN
2000         zeros:=zeros+1
2010       ENDIF
2020       IF y=0 AND x<>0 THEN
2030         zeros:=zeros+1
2040       ENDIF
2160       //
2170       // Interrupted return in case of overestimated eigenvalue
2180       //

2190       IF i MOD 2=1 AND 2*zeros>i-1 OR i MOD 2=0 AND 2*zeros>i-2 THEN
2200         RETURN
```

```
2210        ENDIF
2260      ENDIF
2270      x:=x+h
2320      y:=yny
2330      v:=vny
2340    ENDFOR j
2360  ENDPROC node
3000  PROC backfit(REF ei,REF eimax,REF eimin)
3010    max:=n*i*skridt
3015  again:
3020    x:=PI
3030    IF i MOD 2=1 THEN
3040      y:=1
3045      IF i=3 THEN y:=-y
3050      v:=0
3060    ELSE
3070      y:=0
3080      v:=1
3090    ENDIF
3100    xturn:=SQR(2*ei)
3110    h:=-PI/max
3120    FOR j:=1 TO max DO
3130      k1:=v
3140      l1:=y*(x*x-ei*2)
3150      k2:=v+h*l1/2
3160      l2:=(y+h*k1/2)*((x+h/2)*(x+h/2)-ei*2)
3170      k3:=v+h*l2/2
3180      l3:=(y+h*k2/2)*((x+h/2)*(x+h/2)-ei*2)
3200      k4:=v+h*l3
3210      l4:=(y+h*k3)*((x+h)*(x+h)-ei*2)
3220      yny:=y+h*(k1+2*k2+2*k3+k4)/6
3230      vny:=v+h*(l1+2*l2+2*l3+l4)/6
3240      //
3250      // Fitting boundary conditions at 0
3260      //
3270      IF j=max THEN
3274        WHILE eimax-eimin>epsilon*ei DO
3275
3280          IF i MOD 2=1 THEN
3290            IF vny>0 THEN
3300              eimax:=ei
3310              ei:=(eimax+eimin)/2
3320            ELSE
3330              eimin:=ei
3340              ei:=(eimax+eimin)/2
3350            ENDIF
3355          ENDIF
3360          IF i MOD 2=0 THEN
3370            IF yny>0 THEN
3380              eimax:=ei
3390              ei:=(eimax+eimin)/2
3400            ELSE
3410              eimin:=ei
3420              ei:=(eimax+eimin)/2
3430            ENDIF
3440          ENDIF
3443          GOTO again
3445        ENDWHILE
3450      ENDIF
3452      x:=x+h
3453      y:=yny
3454      v:=vny
3455    ENDFOR j
3460  ENDPROC backfit
4000  PROC backbrok(REF bi,REF bimax,REF bimin)
4010    max:=n*i*skridt
4015  again:
4020    x:=PI
```

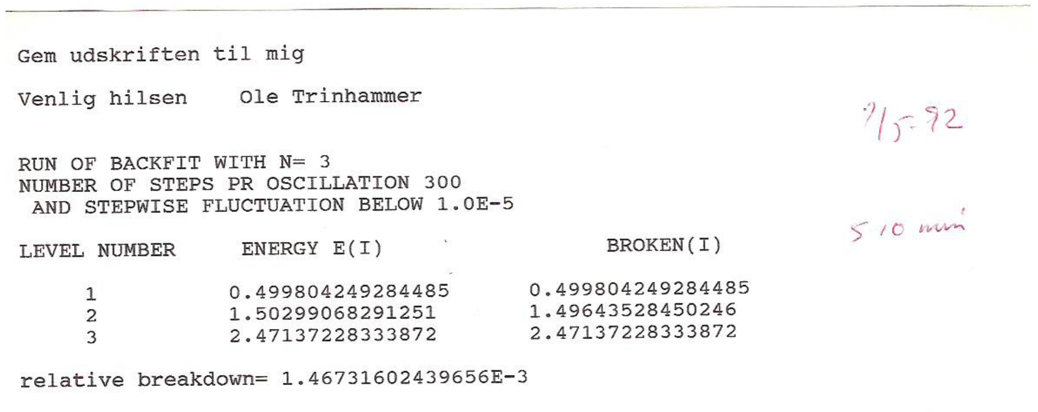
```
4030   y:=1
4040   v:=0
----------------------------------------------------------------------
4110   h:=-PI/max
4120   FOR j:=1 TO max DO
4130     k1:=v
4140     l1:=y*(x*x-bi*2)
4150     k2:=v+h*l1/2
4160     l2:=(y+h*k1/2)*((x+h/2)*(x+h/2)-bi*2)
4170     k3:=v+h*l2/2
4180     l3:=(y+h*k2/2)*((x+h/2)*(x+h/2)-bi*2)
4200     k4:=v+h*l3
4210     l4:=(y+h*k3)*((x+h)*(x+h)-bi*2)
4220     yny:=y+h*(k1+2*k2+2*k3+k4)/6
4230     vny:=v+h*(l1+2*l2+2*l3+l4)/6
4240     //
4250     // Fitting boundary conditions at 0
4260     //
4261     WHILE bimax-bimin>epsilon*bi DO  < IF j=max THEN
4262       IF yny<0 THEN
4263         bimax:=bi
4264         bi:=(bimax+bimin)/2
4265       ELSE
4266         bimin:=bi
4267         bi:=(bimax+bimin)/2
4270       ENDIF
4370       GOTO again
4380
4390
4400
4410
4420
4430
4435
4440
4445     ENDWHILE  ≤ ENDIF
4452     x:=x+h
4453     y:=yny
4454     v:=vny
4455   ENDFOR j
4460 ENDPROC backbrok
4470 END
```



```
Venlig hilsen     Ole Trinhammer

RUN OF BACKFIT WITH N= 3
NUMBER OF STEPS PR OSCILLATION 75
 AND STEPWISE FLUCTUATION BELOW 0.001

LEVEL NUMBER      ENERGY E(I)                  BROKEN(I)

     1           0.499691925667126       0.499691925667126
     2           1.50279860419355        1.49643122781618
     3           2.47045362864153        2.47045362864153

relative breakdown= 1.42556071191857E-3
```

7/5-92

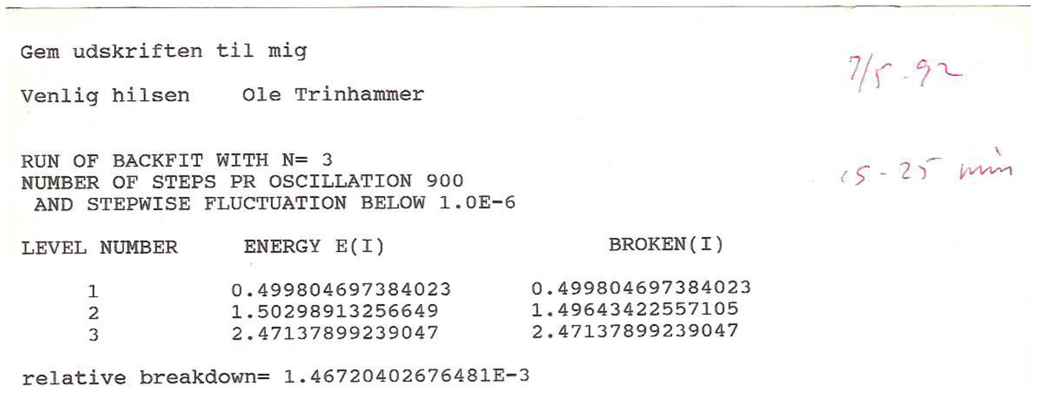
✓



```
Gem udskriften til mig

Venlig hilsen     Ole Trinhammer

RUN OF BACKFIT WITH N= 3
NUMBER OF STEPS PR OSCILLATION 300
 AND STEPWISE FLUCTUATION BELOW 1.0E-5

LEVEL NUMBER      ENERGY E(I)                  BROKEN(I)

     1           0.499804249284485       0.499804249284485
     2           1.50299068291251        1.49643528450246
     3           2.47137228333872        2.47137228333872

relative breakdown= 1.46731602439656E-3
```

7/5-92

5-10 min



```
Gem udskriften til mig

Venlig hilsen     Ole Trinhammer

RUN OF BACKFIT WITH N= 3
NUMBER OF STEPS PR OSCILLATION 900
 AND STEPWISE FLUCTUATION BELOW 1.0E-6

LEVEL NUMBER      ENERGY E(I)                  BROKEN(I)

     1           0.499804697384023       0.499804697384023
     2           1.50298913256649        1.49643422557105
     3           2.47137899239047        2.47137899239047

relative breakdown= 1.46720402676481E-3
```

7/5-92

15-25 min

2. Collocation in Matlab

Collocation programmes for odd- and even-labelled states and for broken even-labelled states date back to the middle of the nineties and were written in Matlab for DOS. The programmes still run under Matlab R2010b with minor error messages. Here is an edition with the new idea to couple perion doublings in even-label states with period doubling in an underlying odd-labelled state. The period doubling for “broken” odd-labelled states is described by changing from expansions on integer orders of even functions $\cos px$ to expansions on half integer orders $\cos(p-\frac{1}{2})x$.

BROKodd2012.m:

```
('Broken, odd states in 1 dimension')
format long e
n=input ('Number of collocation points is n, input n   ')
clear k
clear b
for i=1:n
xi=i*pi/(n+1);
for p=1:n
k(i,p)=cos((p-0.5)*xi);
b(i,p)=((p-0.5)*(p-0.5)+xi*xi)*k(i,p);
end
end
eig(k\b)/2;
clear e
e=sort(eig(k\b)/2);
('Eigenvalues for broken odd states (1 dimension). Number of collocations  '),n
e(1:10)
clear brokenodd
for j=1:10
brokenodd(j)=e(j);
end
clear k
clear b
clear e
save c:eigbrokodd.mat
```

3. Parametric function generation in Maple

The programme "MapleParametricFunctionTableGenerator.mw" shown below integrates the one-dimensional Schrödinger equation and tabulates the results as two files containing the parametric functions and their second derivatives evaluated in a certain set of points.

The files are to be read into the programme "MathcadParametricBasis.mcd", where eigenvalues for the 3-dimensional problem are calculated.

The eigenvalues on which the integrations are based are results from iterative integration with iterations guided by Sturm-Liouville theory, see "ComalIterationSturmLiouville.pdf".

PS: 'ligninger' means 'equations' and 'startbetingelser' means 'boundary conditions'

(1)

$\mathit{restart};\ \mathit{en} := 0.49980470:$
$\mathit{with}(\mathit{LinearAlgebra}):$
$v := x \rightarrow \left(x - \mathrm{round}\left(\frac{x}{2\cdot\pi}\right)\cdot 2\cdot\pi\right)^2$
$\mathit{ligninger} := \mathit{diff}(f(x), x) = \mathit{fx}(x), \mathit{diff}(\mathit{fx}(x), x) = (v(x) - 2\cdot\mathit{en})\cdot f(x):$
$\mathit{startbetingelser} := f(0) = 1, \mathit{fx}(0) = 0:$
$\mathit{solution} := \mathit{dsolve}([\mathit{ligninger}, \mathit{startbetingelser}], \mathit{type} = \mathit{numeric}, \mathit{method} = \mathit{dverk78}, \mathit{output}$
$\quad = \mathit{listprocedure}):$
$\mathit{fsol} := \mathit{rhs}(\mathit{solution}[2]):$

$\mathit{plot}(\mathit{fsol}, -3\cdot\pi..3\cdot\pi, \mathit{gridlines} = \mathit{false}, \mathit{labels} = [\theta, \varphi 1]):\ n := 11:\ m := 29:$
$s := i \rightarrow \mathit{fsol}\left(\frac{i\cdot 2\,\pi}{m}\right):\mathit{f1} := \mathit{Matrix}(m, n, s):$
$g := i \rightarrow \left(v\left(\frac{i\cdot 2\,\pi}{m}\right) - 2\cdot\mathit{en}\right)\cdot\mathit{fsol}\left(\frac{i\cdot 2\,\pi}{m}\right):\mathit{fxx1} := \mathit{Matrix}(m, n, g):$
$\mathit{ff} := \mathit{f1} : \mathit{ffxx} := \mathit{fxx1}:$
$\mathit{evalf}(\mathit{ffxx}):$

$\mathit{en} := 1.50298897:$
$\mathit{ligninger} := \mathit{diff}(f(x), x) = \mathit{fx}(x), \mathit{diff}(\mathit{fx}(x), x) = (v(x) - 2\cdot\mathit{en})\cdot f(x):$
$\mathit{startbetingelserU} := f(0) = 0, \mathit{fx}(0) = 1:$
$\mathit{solution} := \mathit{dsolve}([\mathit{ligninger}, \mathit{startbetingelserU}], \mathit{type} = \mathit{numeric}, \mathit{method} = \mathit{dverk78}, \mathit{output}$
$\quad = \mathit{listprocedure}):$
$\mathit{fsol} := \mathit{rhs}(\mathit{solution}[2]):$
$s := i \rightarrow \mathit{fsol}\left(\frac{i\cdot 2\,\pi}{m}\right):\mathit{f2} := \mathit{Matrix}(m, 1, s):$
$g := i \rightarrow \left(v\left(\frac{i\cdot 2\,\pi}{m}\right) - 2\cdot\mathit{en}\right)\cdot\mathit{fsol}\left(\frac{i\cdot 2\,\pi}{m}\right):\mathit{fxx2} := \mathit{Matrix}(m, 1, g):$

for i **from** 1 **to** m **do** $\mathit{ff}(i, 2) := \mathit{f2}(i)$ **od:**
for i **from** 1 **to** m **do** $\mathit{ffxx}(i, 2) := \mathit{fxx2}(i)$ **od:**

$\mathit{en} := 2.47137783:$
$\mathit{ligninger} := \mathit{diff}(f(x), x) = \mathit{fx}(x), \mathit{diff}(\mathit{fx}(x), x) = (v(x) - 2\cdot\mathit{en})\cdot f(x):$
$\mathit{startbetingelser} := f(0) = 1, \mathit{fx}(0) = 0:$
$\mathit{solution} := \mathit{dsolve}([\mathit{ligninger}, \mathit{startbetingelser}], \mathit{type} = \mathit{numeric}, \mathit{method} = \mathit{dverk78}, \mathit{output}$
$\quad = \mathit{listprocedure}):$
$\mathit{fsol} := \mathit{rhs}(\mathit{solution}[2]):$
$s := i \rightarrow \mathit{fsol}\left(\frac{i\cdot 2\,\pi}{m}\right):\mathit{f3} := \mathit{Matrix}(m, 1, s):$
$g := i \rightarrow \left(v\left(\frac{i\cdot 2\,\pi}{m}\right) - 2\cdot\mathit{en}\right)\cdot\mathit{fsol}\left(\frac{i\cdot 2\,\pi}{m}\right):\mathit{fxx3} := \mathit{Matrix}(m, 1, g):$
for i **from** 1 **to** m **do** $\mathit{ff}(i, 3) := \mathit{f3}(i)$ **od:**
for i **from** 1 **to** m **do** $\mathit{ffxx}(i, 3) := \mathit{fxx3}(i)$ **od:**

$\mathit{en} := 3.60050900:$
$\mathit{ligninger} := \mathit{diff}(f(x), x) = \mathit{fx}(x), \mathit{diff}(\mathit{fx}(x), x) = (v(x) - 2\cdot\mathit{en})\cdot f(x):$
$\mathit{startbetingelserU} := f(0) = 0, \mathit{fx}(0) = 1:$

$solution := dsolve([ligninger, startbetingelserU], type = numeric, method = dverk78, output = listprocedure)$:

$fsol := rhs(solution[2])$:

$s := i \rightarrow fsol\left(\frac{i \cdot 2\,\pi}{m}\right)$: $f4 := Matrix(m, 1, s)$:

$g := i \rightarrow \left(v\left(\frac{i \cdot 2\,\pi}{m}\right) - 2 \cdot en\right) \cdot fsol\left(\frac{i \cdot 2\,\pi}{m}\right)$: $fxx4 := Matrix(m, 1, g)$:

for i **from** 1 **to** m **do** $ff(i, 4) := f4(i)$ **od**:

for i **from** 1 **to** m **do** $ffxx(i, 4) := fxx4(i)$ **od**:

$en := 4.21850471$:

$ligninger := diff(f(x), x) = fx(x), diff(fx(x), x) = (v(x) - 2 \cdot en) \cdot f(x)$:

$startbetingelser := f(0) = 1, fx(0) = 0$:

$solution := dsolve([ligninger, startbetingelser], type = numeric, method = dverk78, output = listprocedure)$:

$fsol := rhs(solution[2])$:

$s := i \rightarrow fsol\left(\frac{i \cdot 2\,\pi}{m}\right)$: $f5 := Matrix(m, 1, s)$:

$g := i \rightarrow \left(v\left(\frac{i \cdot 2\,\pi}{m}\right) - 2 \cdot en\right) \cdot fsol\left(\frac{i \cdot 2\,\pi}{m}\right)$: $fxx5 := Matrix(m, 1, g)$:

for i **from** 1 **to** m **do** $ff(i, 5) := f5(i)$ **od**:

for i **from** 1 **to** m **do** $ffxx(i, 5) := fxx5(i)$ **od**:

$en := 6.19762900$:

$ligninger := diff(f(x), x) = fx(x), diff(fx(x), x) = (v(x) - 2 \cdot en) \cdot f(x)$:

$startbetingelserU := f(0) = 0, fx(0) = 1$:

$solution := dsolve([ligninger, startbetingelserU], type = numeric, method = dverk78, output = listprocedure)$:

$fsol := rhs(solution[2])$:

$s := i \rightarrow fsol\left(\frac{i \cdot 2\,\pi}{m}\right)$: $f6 := Matrix(m, 1, s)$:

$g := i \rightarrow \left(v\left(\frac{i \cdot 2\,\pi}{m}\right) - 2 \cdot en\right) \cdot fsol\left(\frac{i \cdot 2\,\pi}{m}\right)$: $fxx6 := Matrix(m, 1, g)$:

for i **from** 1 **to** m **do** $ff(i, 6) := f6(i)$ **od**:

for i **from** 1 **to** m **do** $ffxx(i, 6) := fxx6(i)$ **od**:

$en := 6.38310080$:

$ligninger := diff(f(x), x) = fx(x), diff(fx(x), x) = (v(x) - 2 \cdot en) \cdot f(x)$:

$startbetingelser := f(0) = 1, fx(0) = 0$:

$solution := dsolve([ligninger, startbetingelser], type = numeric, method = dverk78, output = listprocedure)$:

$fsol := rhs(solution[2])$:

$s := i \rightarrow fsol\left(\frac{i \cdot 2\,\pi}{m}\right)$: $f7 := Matrix(m, 1, s)$:

$g := i \rightarrow \left(v\left(\frac{i \cdot 2\,\pi}{m}\right) - 2 \cdot en\right) \cdot fsol\left(\frac{i \cdot 2\,\pi}{m}\right)$: $fxx7 := Matrix(m, 1, g)$:

for i **from** 1 **to** m **do** $ff(i, 7) := f7(i)$ **od**:

for i **from** 1 **to** m **do** $ffxx(i, 7) := fxx7(i)$ **od**:

$en := 9.68846629:$
$ligninger := diff(f(x), x) = fx(x), diff(fx(x), x) = (v(x) - 2 \cdot en) \cdot f(x):$
$startbetingelserU := f(0) = 0, fx(0) = 1:$
$solution := dsolve([ligninger, startbetingelserU], type = numeric, method = dverk78, output = listprocedure):$
$fsol := rhs(solution[2]):$
$s := i \rightarrow fsol\left(\frac{i \cdot 2\pi}{m}\right) : f8 := Matrix(m, 1, s):$
$g := i \rightarrow \left(v\left(\frac{i \cdot 2\pi}{m}\right) - 2 \cdot en\right) \cdot fsol\left(\frac{i \cdot 2\pi}{m}\right) : fxx8 := Matrix(m, 1, g):$
for i **from** 1 **to** m **do** $ff(i, 8) := f8(i)$ **od:**
for i **from** 1 **to** m **do** $ffxx(i, 8) := fxx8(i)$ **od:**

$en := 9.75132100:$
$ligninger := diff(f(x), x) = fx(x), diff(fx(x), x) = (v(x) - 2 \cdot en) \cdot f(x):$
$startbetingelser := f(0) = 1, fx(0) = 0:$
$solution := dsolve([ligninger, startbetingelser], type = numeric, method = dverk78, output = listprocedure):$
$fsol := rhs(solution[2]):$
$s := i \rightarrow fsol\left(\frac{i \cdot 2\pi}{m}\right) : f9 := Matrix(m, 1, s):$
$g := i \rightarrow \left(v\left(\frac{i \cdot 2\pi}{m}\right) - 2 \cdot en\right) \cdot fsol\left(\frac{i \cdot 2\pi}{m}\right) : fxx9 := Matrix(m, 1, g):$
for i **from** 1 **to** m **do** $ff(i, 9) := f9(i)$ **od:**
for i **from** 1 **to** m **do** $ffxx(i, 9) := fxx9(i)$ **od:**

$en := 14.1755275:$
$ligninger := diff(f(x), x) = fx(x), diff(fx(x), x) = (v(x) - 2 \cdot en) \cdot f(x):$
$startbetingelserU := f(0) = 0, fx(0) = 1:$
$solution := dsolve([ligninger, startbetingelserU], type = numeric, method = dverk78, output = listprocedure):$
$fsol := rhs(solution[2]):$
$s := i \rightarrow fsol\left(\frac{i \cdot 2\pi}{m}\right) : f10 := Matrix(m, 1, s):$
$g := i \rightarrow \left(v\left(\frac{i \cdot 2\pi}{m}\right) - 2 \cdot en\right) \cdot fsol\left(\frac{i \cdot 2\pi}{m}\right) : fxx10 := Matrix(m, 1, g):$
for i **from** 1 **to** m **do** $ff(i, 10) := f10(i)$ **od:**
for i **from** 1 **to** m **do** $ffxx(i, 10) := fxx10(i)$ **od:**

$en := 14.2063574:$
$ligninger := diff(f(x), x) = fx(x), diff(fx(x), x) = (v(x) - 2 \cdot en) \cdot f(x):$
$startbetingelser := f(0) = 1, fx(0) = 0:$
$solution := dsolve([ligninger, startbetingelser], type = numeric, method = dverk78, output = listprocedure):$
$fsol := rhs(solution[2]):$
$s := i \rightarrow fsol\left(\frac{i \cdot 2\pi}{m}\right) : f11 := Matrix(m, 1, s):$
$g := i \rightarrow \left(v\left(\frac{i \cdot 2\pi}{m}\right) - 2 \cdot en\right) \cdot fsol\left(\frac{i \cdot 2\pi}{m}\right) : fxx11 := Matrix(m, 1, g):$

```
for i from 1 to m do ff(i, 11) := f11(i) od:
for i from 1 to m do ffxx(i, 11) := fxx11(i) od:
ff:
evalf(ffxx) :
ExportMatrix(ffFile, ff, target=Matlab) :
ExportMatrix(ffxxFile, evalf(ffxx), target=Matlab);
                                        3988,00                                  (2)
```

4. Eigenvalue determination in Mathcad

The programme "MathcadParametricBasisPDF.mcd" shown below calculates eigenvalues for neutral charge N and Δ-states depending on results from the Maple programme "MapleParametricFunctionTableGenerator.mw" shown above. We start by reading files containing tabulated values of a set of parametric basis functions and their second derivatives generated by the Maple programme. Contrary to the exact calculations in appendix C the integrations are now simple sums of point values of the integrand multiplied by the step length to the cube, i.e. $1/M^3$, where M is the number of points tabulated for each function. We call it "Number of base points..." though this may not be the proper expression in English. The particular calculation shown below takes half a week on a ThinkPad T61 whereas – shown further below - a similar calculation based on the exact matrix elements for the trigonometric basis in appendix C takes less than five minutes.

MathcadParametricBasisPDF.mcd:

MathcadTrigonimeticBasisPDF.mcd:

The matrices for analytically determined matrix elements are labelled with a, like "Ia", "Iv2a" and so on. The corresponding eigenvalues are called "e" and "ec" respectively. "e" refers to approximate solutions where the global curvature and centrifugal potentials are disregarded. "ec" refers the full, exact solution. "eee" at the end of the programme is based on numerically calculated matrix elements. This is a much more time consuming process. It is encouraging though to have the numerical results for specific matrix elements as a check on the rather complicated algebraic expressions for analytically derived results.

MathcadTrigonometri-
cBasis
Analytical Integrals
Compare Numerical

$$f(p,q,r,x,y,z) := \left|\begin{pmatrix} \cos(p\cdot x) & \cos(p\cdot y) & \cos(p\cdot z) \\ \sin(q\cdot x) & \sin(q\cdot y) & \sin(q\cdot z) \\ \cos(r\cdot x) & \cos(r\cdot y) & \cos(r\cdot z) \end{pmatrix}\right|$$

$$fx(p,q,r,x,y,z) := \left|\begin{pmatrix} -p\cdot\sin(p\cdot x) & \cos(p\cdot y) & \cos(p\cdot z) \\ q\cdot\cos(q\cdot x) & \sin(q\cdot y) & \sin(q\cdot z) \\ -r\cdot\sin(r\cdot x) & \cos(r\cdot y) & \cos(r\cdot z) \end{pmatrix}\right|$$

$$fy(p,q,r,x,y,z) := \left|\begin{pmatrix} \cos(p\cdot x) & -p\cdot\sin(p\cdot y) & \cos(p\cdot z) \\ \sin(q\cdot x) & q\cdot\cos(q\cdot y) & \sin(q\cdot z) \\ \cos(r\cdot x) & -r\cdot\sin(r\cdot y) & \cos(r\cdot z) \end{pmatrix}\right|$$

$$fz(p,q,r,x,y,z) := \left|\begin{pmatrix} \cos(p\cdot x) & \cos(p\cdot y) & -p\cdot\sin(p\cdot z) \\ \sin(q\cdot x) & \sin(q\cdot y) & q\cdot\cos(q\cdot z) \\ \cos(r\cdot x) & \cos(r\cdot y) & -r\cdot\sin(r\cdot z) \end{pmatrix}\right|$$

$$fxx(p,q,r,x,y,z) := \left|\begin{pmatrix} -p\cdot p\cdot\cos(p\cdot x) & \cos(p\cdot y) & \cos(p\cdot z) \\ -q\cdot q\cdot\sin(q\cdot x) & \sin(q\cdot y) & \sin(q\cdot z) \\ -r\cdot r\cdot\cos(r\cdot x) & \cos(r\cdot y) & \cos(r\cdot z) \end{pmatrix}\right|$$

$$fyy(p,q,r,x,y,z) := \left|\begin{pmatrix} \cos(p\cdot x) & -p\cdot p\cdot\cos(p\cdot y) & \cos(p\cdot z) \\ \sin(q\cdot x) & -q\cdot q\cdot\sin(q\cdot y) & \sin(q\cdot z) \\ \cos(r\cdot x) & -r\cdot r\cdot\cos(r\cdot y) & \cos(r\cdot z) \end{pmatrix}\right|$$

$$fzz(p,q,r,x,y,z) := \left|\begin{pmatrix} \cos(p\cdot x) & \cos(p\cdot y) & -p\cdot p\cdot\cos(p\cdot z) \\ \sin(q\cdot x) & \sin(q\cdot y) & -q\cdot q\cdot\sin(q\cdot z) \\ \cos(r\cdot x) & \cos(r\cdot y) & -r\cdot r\cdot\cos(r\cdot z) \end{pmatrix}\right|$$

$$I(p,q,r,s,t,u) := \frac{\int_{-\pi}^{\pi}\int_{-\pi}^{\pi}\int_{-\pi}^{\pi} f(p,q,r,x,y,z)\cdot f(s,t,u,x,y,z)\,dx\,dy\,dz}{\pi^3}$$

$$dIxx(p,q,r,s,t,u) := \frac{\int_{-\pi}^{\pi}\int_{-\pi}^{\pi}\int_{-\pi}^{\pi} fxx(p,q,r,x,y,z)\cdot f(s,t,u,x,y,z)\,dx\,dy\,dz}{\pi^3}$$

$$dIyy(p,q,r,s,t,u) := \frac{\int_{-\pi}^{\pi}\int_{-\pi}^{\pi}\int_{-\pi}^{\pi} fyy(p,q,r,x,y,z)\cdot f(s,t,u,x,y,z)\,dx\,dy\,dz}{\pi^3}$$

$$\mathrm{dIzz}(p,q,r,s,t,u) := \frac{\int_{-\pi}^{\pi}\int_{-\pi}^{\pi}\int_{-\pi}^{\pi} \mathrm{fzz}(p,q,r,x,y,z)\cdot f(s,t,u,x,y,z)\,dx\,dy\,dz}{\pi^3}$$

$$\mathrm{dI}(p,q,r,s,t,u) := \mathrm{dIxx}(p,q,r,s,t,u) + \mathrm{dIyy}(p,q,r,s,t,u) + \mathrm{dIzz}(p,q,r,s,t,u)$$

$$\mathrm{Iv2}(p,q,r,s,t,u) := \frac{\left[\int_{-\pi}^{\pi}\int_{-\pi}^{\pi}\int_{-\pi}^{\pi} \left(x^2+y^2+z^2\right)\cdot f(p,q,r,x,y,z)\cdot f(s,t,u,x,y,z)\,dx\,dy\,dz\right]}{\pi^3}$$

```
v(N) :=  m ← 0
         for h ∈ 1..N
             p ← h − 1
             for q ∈ 1..N
                 for r ∈ p + 1..N
                     m ← m + 1
                     v_{1,m} ← p
                     v_{2,m} ← q
                     v_{3,m} ← r
         v
```

cols(v(4)) = 40

```
dim(N) :=  m ← 0
           for h ∈ 1..N
               p ← h − 1
               for q ∈ 1..N
                   for r ∈ p + 1..N
                       m ← m + 1
           m
```

```
d(x,y) :=  1  if |x| = |y| ∧ x ≠ 0
           2  if |x| = |y| ∧ x = 0
           0  otherwise
```

```
dd(x,y) :=  1  if x = y ∧ x ≠ 0
            (−1)  if x = −y ∧ x ≠ 0
            0  otherwise
```

```
nn(x,y) :=  |x + y| − |x − y|  if mod(x + y, 2) = 0
            0  otherwise
```

All integrals are dividered by pi^3, which faktorizes out in the eigenvaluedetermination

$Ia(p,q,r,s,t,u) := 6\,d(p,s)\cdot d(q,t)\cdot d(r,u)$

```
n(N) :=  d ← dim(N)
         v ← v(N)
         for  i ∈ 1 .. d
             p ← $v_{1,i}$
             q ← $v_{2,i}$
             r ← $v_{3,i}$
             for  j ∈ i .. d
                 s ← $v_{1,j}$
                 t ← $v_{2,j}$
                 u ← $v_{3,j}$
                 $n_{i,j}$ ← Ia(p,q,r,s,t,u)
                 $n_{j,i}$ ← $n_{i,j}$
         n
```

```
nnn(N) :=  d ← dim(N)
           v ← v(N)
           for  i ∈ 1 .. d
               p ← $v_{1,i}$
               q ← $v_{2,i}$
               r ← $v_{3,i}$
               for  j ∈ i .. d
                   s ← $v_{1,j}$
                   t ← $v_{2,j}$
                   u ← $v_{3,j}$
                   $nnn_{i,j}$ ← I(p,q,r,s,t,u)
                   $nnn_{j,i}$ ← $nnn_{i,j}$
           nnn
```

$dIa(p,q,r,s,t,u) := 6\cdot d(p,s)\cdot d(q,t)\cdot d(r,u)\cdot\left(0 - p^2 - q^2 - r^2\right)$ TJEK p og s = 0

$$
\mathrm{Iv2a}(p,q,r,s,t,u) := \begin{cases}
6\cdot\left(\dfrac{1}{2\cdot p^2}+\dfrac{1}{2\cdot r^2}+\pi^2-\dfrac{1}{2\cdot q^2}\right) & \text{if } p\cdot s \neq 0 \land p = s \land q = t \land r = u \\
6\cdot 4\cdot(-1)^{p+s}\cdot\dfrac{\left(p^2+s^2\right)}{\left(p^2-s^2\right)^2} & \text{if } p\cdot s \neq 0 \land p \neq s \land q = t \land r = u \\
6\cdot 4\cdot(-1)^{q+t}\cdot\dfrac{2q\cdot t}{\left(q^2-t^2\right)^2} & \text{if } p\cdot s \neq 0 \land q \neq t \land p = s \land r = u \\
6\cdot 4\cdot(-1)^{r+u}\cdot\dfrac{\left(r^2+u^2\right)}{\left(r^2-u^2\right)^2} & \text{if } p\cdot s \neq 0 \land r \neq u \land p = s \land q = t \\
0 - 6\cdot 4\cdot(-1)^{p+u}\cdot\dfrac{\left(p^2+u^2\right)}{\left(p^2-u^2\right)^2} & \text{if } p\cdot s \neq 0 \land p \neq u \land q = t \land r = s \\
0 - 6\cdot 4\cdot(-1)^{r+s}\cdot\dfrac{\left(r^2+s^2\right)}{\left(r^2-s^2\right)^2} & \text{if } p\cdot s \neq 0 \land r \neq s \land p = u \land q = t \\
24\cdot\left[\dfrac{(-1)^s}{s^2}\right] & \text{if } p = 0 \land s \neq 0 \land q = t \land r = u \\
24\cdot\left[\dfrac{-(-1)^u}{u^2}\right] & \text{if } p = 0 \land s \neq 0 \land u \neq 0 \land q = t \land r = s \\
0 & \text{if } p = 0 \land s \neq 0 \land q \neq t \\
6\cdot\left[\dfrac{1}{r^2}+2\pi^2+\dfrac{(-1)}{q^2}\right] & \text{if } p = 0 \land s = 0 \land q = t \land r = u \\
48\cdot\left[(-1)^{q+t}\dfrac{(2q\cdot t)}{\left(q^2-t^2\right)^2}\right] & \text{if } p = 0 \land s = 0 \land q \neq t \land r = u \\
48\cdot\left[(-1)^{r+u}\cdot\dfrac{\left(r^2+u^2\right)}{\left(r^2-u^2\right)^2}\right] & \text{if } p = 0 \land s = 0 \land r \neq u \land q = t \\
0 & \text{if } p = 0 \land s = 0 \land r \neq u \land q \neq t \\
0 & \text{otherwise}
\end{cases}
$$

```
m(N) :=  d ← dim(N)
         v ← v(N)
         for i ∈ 1 .. d
             p ← $v_{1,i}$
             q ← $v_{2,i}$
             r ← $v_{3,i}$
             for j ∈ i .. d
                 s ← $v_{1,j}$
                 t ← $v_{2,j}$
                 u ← $v_{3,j}$
                 $m_{i,j}$ ← Iv2a(p,q,r,s,t,u) − dIa(p,q,r,s,t,u)
                 $m_{j,i}$ ← $m_{i,j}$
         m
```

```
mmm(N) :=  d ← dim(N)
           v ← v(N)
           for i ∈ 1 .. d
               p ← $v_{1,i}$
               q ← $v_{2,i}$
               r ← $v_{3,i}$
               for j ∈ i .. d
                   s ← $v_{1,j}$
                   t ← $v_{2,j}$
                   u ← $v_{3,j}$
                   $mmm_{i,j}$ ← Iv2(p,q,r,s,t,u) − dI(p,q,r,s,t,u)
                   $mmm_{j,i}$ ← $mmm_{i,j}$
           mmm
```

$eps := 10^{-50}$ $\qquad\qquad L := \frac{1}{2} \qquad M2 := \frac{13}{4}$

$$s(x,y) := \begin{cases} 4\cdot 10^{100} & \text{if } |x-y| \le eps \vee \big||x-y| - 2\pi\big| \le eps \\ \dfrac{1}{[\sin[.5\cdot(x-y)]]^2} & \text{otherwise} \end{cases}$$

$$c(x,y,z) := -2 + (s(x,y) + s(y,z) + s(z,x))\frac{[L\cdot(L+1) + M2]}{3\cdot 8}$$

$$Ic(p,q,r,s,t,u) := \frac{\int_{-\pi}^{\pi}\int_{-\pi}^{\pi}\int_{-\pi}^{\pi} c(x,y,z)\cdot f(p,q,r,x,y,z)\cdot f(s,t,u,x,y,z)\,dx\,dy\,dz}{\pi^3}$$

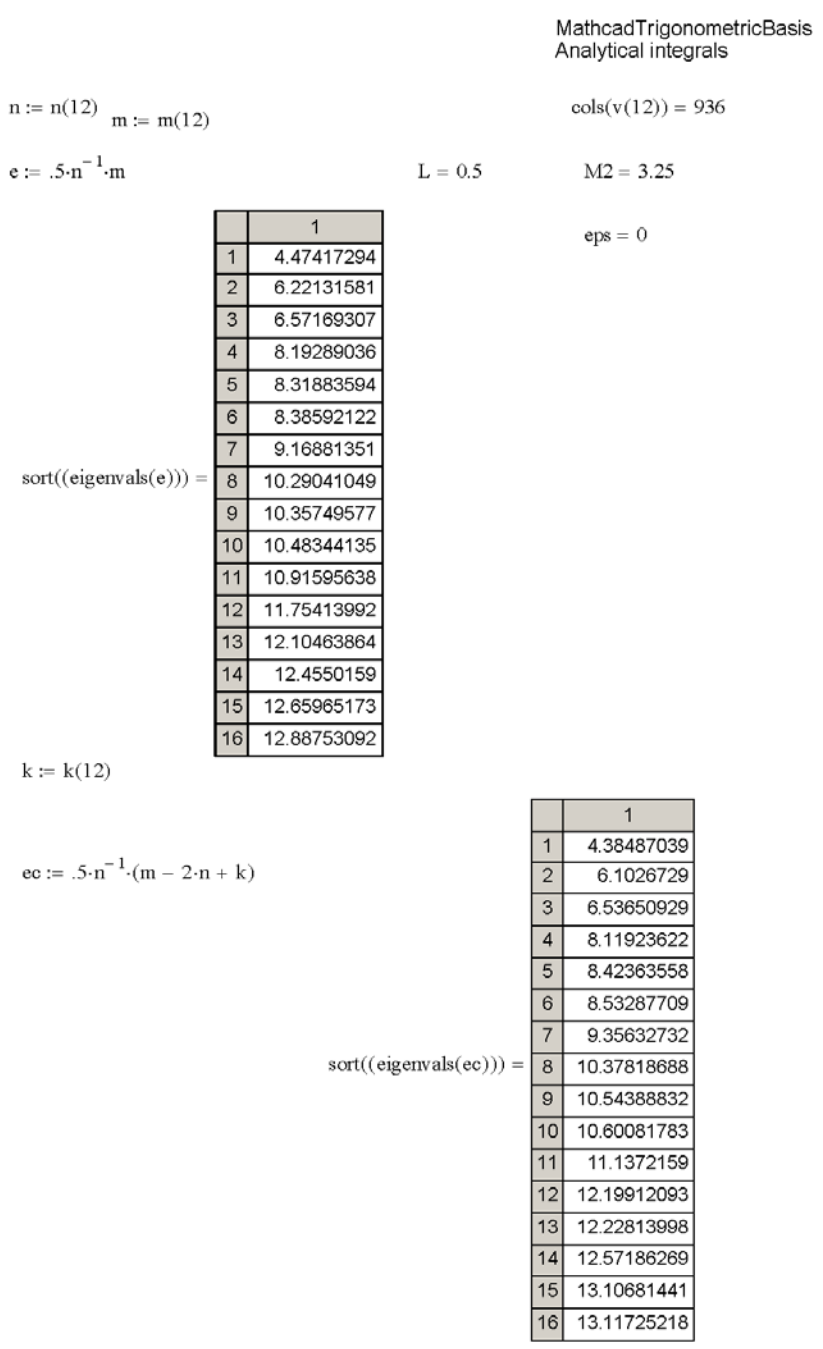
Icc should actually be multiplied by 3
Therefore a factor 1/3 is omitted in the
construction of
k below

```
Icc(p,q,r,s,t,u) := | I1 ← d(p,s)·(d(r − q,u − t)·nn(r + q,u + t) − d(r − q,u + t)·nn(r + q,u − t))
                    | I2 ← I1 + d(p,s)·(−d(r + q,u − t)·nn(r − q,u + t) + d(r + q,u + t)·nn(r − q,u ·     − t))
                    | I3 ← I2 + d(p,u)·(d(r − q,s + t)·nn(r + q,s − t) − d(r − q,s − t)·nn(r + q,s + t     t))
                    | I4 ← I3 + d(p,u)·(−d(r + q,s + t)·nn(r − q,s − t) + d(r + q,s − t)·nn(r − q,s +      t))
                    | I5 ← I4 + d(q,t)·(dd(p + r,s + u)·nn(r − p,u − s) + dd(p + r,u − s)·nn(r − p,u     ι + s))
                    | I6 ← I5 + d(q,t)·(dd(r − p,s + u)·nn(p + r,u − s) + dd(r − p,u − s)·nn(p + r,u     ι + s))
                    | I7 ← I6 + d(r,s)·(d(p + q,u − t)·nn(p − q,u + t) − d(p + q,u + t)·nn(p − q,u      − t))
                    | I8 ← I7 + d(r,s)·(−d(p − q,u − t)·nn(p + q,u + t) + d(p − q,u + t)·nn(p + q,ι     ı − t))
                    | I9 ← I8 + d(r,u)·(d(p + q,s + t)·nn(p − q,s − t) − d(p + q,s − t)·nn(p − q,s +     · t))
                    | I10 ← I9 + d(r,u)·(−d(p − q,s + t)·nn(p + q,s − t) + d(p − q,s − t)·nn(p + q,:     s + t))
                    | Icc ← I10
```

$c(x,y,z) := s(x,y)$

$$Iccc(p,q,r,s,t,u) := \frac{\int_{-\pi}^{\pi}\int_{-\pi}^{\pi}\int_{-\pi}^{\pi} c(x,y,z)\cdot f(p,q,r,x,y,z)\cdot f(s,t,u,x,y,z)\,dx\,dy\,dz}{\pi^3}$$

```
k(N) := | d ← dim(N)
        | v ← v(N)
        | for i ∈ 1 .. d
        |   | p ← v_{1,i}
        |   | q ← v_{2,i}
        |   | r ← v_{3,i}
        |   | for j ∈ i .. d
        |   |   | s ← v_{1,j}
        |   |   | t ← v_{2,j}
        |   |   | u ← v_{3,j}
        |   |   | k_{i,j} ← Icc(p,q,r,s,t,u)·$\frac{[L\cdot(L+1)+M2]}{8}$
        |   |   | k_{j,i} ← k_{i,j}
        | k
```



MathcadTrigonometricBasis
Analytical integrals

$n := n(12)$ $m := m(12)$

$cols(v(12)) = 936$

$e := .5 \cdot n^{-1} \cdot m$

$L = 0.5$

$M2 = 3.25$

$eps = 0$

sort((eigenvals(e))) =

	1
1	4.47417294
2	6.22131581
3	6.57169307
4	8.19289036
5	8.31883594
6	8.38592122
7	9.16881351
8	10.29041049
9	10.35749577
10	10.48344135
11	10.91595638
12	11.75413992
13	12.10463864
14	12.4550159
15	12.65965173
16	12.88753092

$k := k(12)$

$ec := .5 \cdot n^{-1} \cdot (m - 2 \cdot n + k)$

sort((eigenvals(ec))) =

	1
1	4.38487039
2	6.1026729
3	6.53650929
4	8.11923622
5	8.42363558
6	8.53287709
7	9.35632732
8	10.37818688
9	10.54388832
10	10.60081783
11	11.1372159
12	12.19912093
13	12.22813998
14	12.57186269
15	13.10681441
16	13.11725218

$nnn := nnn(2)$

$mmm := mmm(2)$

$eee := .5 \cdot nnn^{-1} \cdot mmm$

MathcalTrigonometricBasis
Numerical Integrals

Results shown here only for 6 base functions. A calculation with 936 base functions as above with the Analytical integrals is extraordinarily time consuming

$$\mathrm{sort}((\mathrm{eigenvals}(eee))) = \begin{pmatrix} 4.54217481 \\ 6.73188553 \\ 6.99332762 \\ 8.75986705 \\ 9.18303834 \\ 11.21101986 \end{pmatrix}$$