

Práctica Computacional: La Ecuación de Schrödinger Estados ligados por diagonalización [§]

A. Parte computacional

1. Utilizar el *notebook* DIAG.NB para obtener los estados ligados del oscilador armónico.
2. Comparar los resultados obtenidos numéricamente con los teóricos. Variar los parámetros y analizar la convergencia.
3. Comparar las funciones de onda con las teóricas.

Diagonalizando con Mathematica

Rango y Numero de Puntos

```
xmin = 0 ;
xmax = 5 ;
npts = 500 ;
Δ = (xmax - xmin) / npts ;
```

Creacion del Hamiltoniano

```
m = Table[ If [i = j - 1, -0.5 / (Δ^2),
              If [i = j, 1.0 / (Δ^2) + 0.5 (i + Δ)^2,
                  If [i = j + 1, -0.5 / (Δ^2), 0. ] ] ],
           {i, npts}, {j, npts}];
```

Autovalores (Energias)

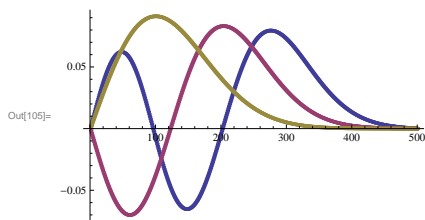
```
In[103]= Eigenvalues[m, -5]
```

```
Out[103]= {9.53424, 7.5024, 5.4999, 3.49992, 1.49998}
```

Primeros Autovectores

```
In[104]= Vvarios = Eigenvectors[m, -3];
```

```
In[105]= ListPlot[Vvarios]
```



```
In[106]= Vg = Eigenvectors[m, -3][[-1]];
```

[§]<http://www.df.uba.ar/users/dmitnik/computation/diagonal/diagonal.html>

B. Implementación en Fortran

El método de diagonalización directa será explicado brevemente en clase, y el alumno recibirá un programa básico (WELL.FOR) con el cual podrá realizar los ejercicios. El programa no está optimizado, su objetivo es pedagógico. Para hacer el programa más simple, la subrutina de diagonalización (TQLI.FOR) se compila en forma separada.

Se puede obtener una introducción sobre cómo compilar y cómo correr este programa, en

<http://www.df.uba.ar/users/dmitnik/computation/oscillator/intro.html>

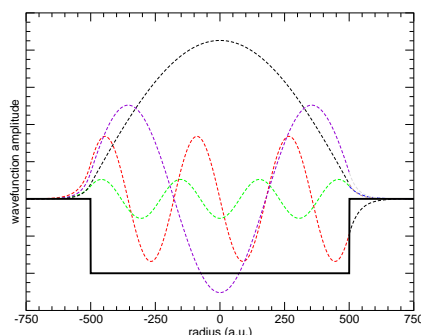


FIG. 1. Algunas de las funciones halladas utilizando el programa WELL.FOR. Los parámetros iniciales se dan como datos dentro del programa.

1. Aumente progresivamente la profundidad del pozo de potencial
 - (a) ¿Qué ocurre con el número de funciones halladas?
 - (b) ¿Qué ocurre con el número de nodos de ellas?
 - (c) ¿Qué ocurre con las energías?
2. Explore cómo cambian las energías variando el radio a
3. ¿Se puede determinar la profundidad del pozo conociendo la energía del estado básico?
4. Explore la paridad de las funciones
5. Modifique el programa y trate de obtener soluciones para otros potenciales. En particular, verifique sus respuestas a las preguntas II-B-7 y 8.

I. APÉNDICE: PROGRAMA WELL.FOR

Printed by Dario Mitnik

```

May 30, 05 15:14          well.for          Page 3/4
program potential_well          !! 30 May 2005
c-----sorting the eigenvectors/values into ascending energies
c-----Generates the complete set of mxpts orbitals by diagonalization
c.....by Dario Mitnik
c-----store all bound state wavefunctions
c-----open('orbital.dat', status='unknown')
c-----open('potentialwell.dat', status='unknown')
c-----ener(nq) = d(nq)
c.....data input if (ener(nq).lt.0) print45, nq, ener(nq)
45 call readmat(5x, 'n=', i3, 10x, 'energy=', f15.4, 'a.u.')

c-----construct the Hamiltonian
call constrH
do 100 irow=1, nrmesh
do 100 ircol=1, nrmesh
diagonalize sumsum+v(irow, nq)**2
call diagH
if (v(1, nq).lt.zero) sign=-one
c.....output do 200 irow=1, nrmesh
200 call printH chi(nq, irow)=sign*v(irow, nq)/sqrt(ptot)
500 continue
stop
endrun
end

c
c*****
c
subroutine printH
c-----Reads input energies and selected bound wavefunctions.
c.....
parameter (mxpts=500)
common/bkmesh/rmin, rmax, hr, nrmesh
parameter (eigen=500, mxpts), chi(mxpts, mxpts)
common/bkhamilt/d(mxpts), e(mxpts)
common/bkpot/apot, V0
write(10, 5)
format(105x) 'Results: '
write(10, 6)
format(105x) 'Potential Well: Energies and Wavefunctions: '
6 format(105x) 'lt.10) write(10, 45) nq, ener(nq)
20 continue
45 format(5x, 'n=', i5, 10x, 'energy=', f15.4, 'a.u.')
c-----wavefunction parameters
inptfile = 11
c-----output file integration parameters
50 ..... format(105x) 'Integration interval
60 ..... printH, 'give the number of points (0 to 1000)
c..... readH, nrmesh points
c..... if (nq.le.0) return
c..... printH, 'give the name of the output file (extension only)
read(*, 50) outputfile(6:30)
c..... default parameters
open(@unit=inptfile, file=outputfile, status='unknown')
nrmesh=100
do 75 irow=1, nrmesh
75 print*, 'give the space integration interval hr (a.u.):'
readwrite(inptfile, *) r, ' ', chi(nq, irow)+ener(nq)
print*, 'give the number of total steps (nrmesh) (max=', mxpts, ')'
reade(@mpesh=1e, status='keep')
write(100220), hr, nrmesh
format(5x, 'hr=', f10.4, 2x, 'nrmesh=', i5)
if (nrmesh.gt.mxpts) pause 'increase mxpts parameter'
end
rmin=0.0
rmax=nrmesh*hr
c***** write(10, 330) 'rmin', rmax*****
630 format(5x, 'rmin=', f10.4, 5x, 'rmax=', f10.4)
function Vpot(x)

```

Tuesday May 31, 2005

well.for

```

May 30, 05 15:14          well.for          Page 4/4
c-----potential V parameters
c-----Generates the Potential.
c.....V0--deep of the potential
c.....parameter (mxpts=500)
common/bkmesh/rmin, rmax, hr, nrmesh
common/bkhamilt/d(mxpts), e(mxpts)
common/bkpot/apot, V0
400 print*, 'give the potential range a (a.u.):'
data half, one, two, five/0.5, 1.0, 2.0, 5.0/
write(10, 440), V0, apot
c-----if (bndgtor) then on the grid
x0 = printx, 'this potential is too broad for your mesh'
go to 400
endiflow = x0-apot/two
440 format(105x) 'V0=', f10.4, 2x, 'a=', f10.4)
Vpot = 0.0
return if (x.gt.rlow.and.x.le.rhigh) Vpot=-apot
end
write(40, *) x, ' ', Vpot
c*****
c
return
subroutine constrH
c
c*****Generates the Hamiltonian:*****
c
parameter (mxpts=500)
common/bkmesh/rmin, rmax, hr, nrmesh
common/bkhamilt/d(mxpts), e(mxpts)
data half, one, two/0.5, 1.0, 2.0/
write(10, 50)
format(/5x, 'Construction of complete set of orbitals')
write(10, *) ' '
c-----building hamiltonian on the grid
x0 = (rmax + rmin)/two
do 40 i=1, nrmesh
r = i*hr
c.....diagonal terms
d(i) = one/hr**2 + Vpot(r)
c.....non-diagonal terms
e(i) = -half/hr**2
40 continue
return
end
c
c*****
c
subroutine diagH
c----- Matrix diagonalization.
c----- Also stores the Eigenvalues and Eigenvectors.
parameter (mxpts=500)
common/bkmesh/rmin, rmax, hr, nrmesh
common/bkhamilt/d(mxpts), e(mxpts)
c
d(mxpts) ! diagonal part of the Hamiltonian
e(mxpts) ! non-diagonal part of H
dimension v(mxpts, mxpts) ! eigenvectors
dimension wksp(mxpts), iwksp(mxpts) ! working arrays for sorting
common/bkeigen/ener(mxpts), chi(mxpts, mxpts)
c
chi(n, mxpts) ! normalized eigenvectors
data zero, one/0.0, 1.0/
c-----matrix diagonalization
call tqld(d, e, nrmesh, mxpts, v)

```

2/2

FIG. 2. Programa WELL.FOR para la solución de la ecuación de Schrödinger por diagonalización.