

## Apéndice A

# El formalismo de superoperadores

### A.1. El Espacio de Hilbert

Podemos pensar que el espacio de Hilbert es un espacio vectorial expandido por una base de autoestados ortonormales, que representamos  $|j\rangle$ .

Luego, un estado en la base de Hilbert queda representado por alguno de estos autoestados, o por combinaciones de ellos.

Dentro del espacio de Hilbert definimos la clase de los operadores, que actúan sobre los autoestados cambiándolos o multiplicándolos por un factor.

Ejemplos de operadores son los operadores de espín,  $I^+$ ;  $I^-$ ;  $I^z$ ;  $I^y$ ; etc. Ya sabemos de la mecánica cuántica como actúan estos operadores sobre un estado del espacio de Hilbert.

El espacio de Hilbert es en el que naturalmente trabajamos y definimos operaciones como producto escalar de autofunciones,  $\langle m|j\rangle$ ; trazas de operadores,  $\text{Tr}[I^z] = \sum_m \langle m|I^z|m\rangle$ ; definimos conmutadores,  $[I^k, I^l] = I^k I^l - I^l I^k$ ; y anti-conmutadores; etc.

Definimos también en este espacio el operador densidad; la ecuación de Liouville; los operadores de evolución y vimos como representar los pulsos de rf. Si trabajamos con un sistema de  $N$  espines  $I$ , la dimensión del espacio de Hilbert viene dada por  $(2I + 1)^N$ .

Si ahora avanzamos un paso en la jerarquía de los espacios lineales en mecánica cuántica, llegamos al espacio de Liouville.[15, 14]

### A.2. El Espacio de Liouville

Es un superespacio del espacio de Hilbert y es expandido por una base de  $(2I + 1)^N \times (2I + 1)^N$  estados. A estos estados base del espacio de Liouville y sus adjuntos los denotamos  $|\hat{Q}_j\rangle$  y  $\langle\hat{Q}_j|$ .

En analogía con los estados del espacio de Hilbert, definimos el producto de los estados del espacio de Liouville como:

$$(\hat{Q}_j|\hat{Q}_k) = \text{Tr} [\hat{Q}_j^\dagger \hat{Q}_k] \quad (\text{A.1})$$

$\widehat{Q}_j$  denota un operador de espín en el espacio de Hilbert (en el texto omitimos el *sombrero* en la notación, pero en esta sección resulta aclaratorio) y  $|\widehat{Q}_j\rangle$  un estado en el espacio de Liouville.

### A.2.1. Superoperadores de Espín

El rol de los operadores  $\widehat{Q}$  en el espacio de Hilbert queda reemplazado por los superoperadores  $\widehat{\widehat{Q}}$  en el espacio de Liouville. Quedando los superoperadores definidos con la siguiente expresión:

$$\begin{aligned}\widehat{\widehat{Q}}|\widehat{Q}_j\rangle &= |[\widehat{Q}, \widehat{Q}_j]\rangle \\ &= |\widehat{Q}\widehat{Q}_j - \widehat{Q}_j\widehat{Q}\rangle\end{aligned}\tag{A.2}$$

Es decir un operador del espacio de Hilbert es cambiado a un nuevo operador cuando le aplicamos un superoperador. La acción de un superoperador es realizar una operación de conmutación sobre el espacio de Hilbert.

De acuerdo a esta regla, los elementos de matriz de un superoperador son función del superoperador, y en su forma normalizada pueden expresarse como:

$$\begin{aligned}M_{jk}(\widehat{\widehat{Q}}) &= \frac{(\widehat{Q}_j|\widehat{\widehat{Q}}|\widehat{Q}_j)}{(\widehat{Q}_j|\widehat{Q}_j)} \\ &= \frac{(\widehat{Q}_j|[\widehat{Q}, \widehat{Q}_j])}{(\widehat{Q}_j|\widehat{Q}_j)} \\ &= \frac{([\widehat{Q}^\dagger, \widehat{Q}_j]|\widehat{Q}_k)}{(\widehat{Q}_j|\widehat{Q}_j)}\end{aligned}\tag{A.3}$$

### A.2.2. Superoperadores de Proyección

En el espacio de Hilbert un operador de proyección  $\widehat{P}_j$  que proyecta sobre un estado arbitrario  $|\psi\rangle = \sum_i c_i |i\rangle$  en la autofunción  $|j\rangle$  quedaba representado por

$$\widehat{P}_j = \frac{|j\rangle\langle j|}{\langle j|j\rangle}$$

y obteníamos la proyección:

$$\begin{aligned}\widehat{P}_j &= \sum_i c_i |j\rangle \frac{\langle j|i\rangle}{\langle j|j\rangle} \\ &= c_j |j\rangle\end{aligned}\tag{A.4}$$

que representa la cantidad de  $|j\rangle$  contenida en  $|\psi\rangle$ .

Los operadores de proyección pueden ser utilizados para la resolución espectral de un operador  $\hat{A}$ . El espectro de un operador queda bien definido como el conjunto completo de autovalores  $\{a_j, j = 1, \dots, n\}$ . Si  $|j\rangle$  es la correspondiente autofunción y  $\hat{P}_j$  el proyector asociado, es posible representar a  $\hat{A}$  como  $\hat{A} = \sum_i a_i \hat{P}_i$ .

Este conjunto particular de proyectores es llamado el conjunto espectral del operador  $\hat{A}$  y tiene la propiedad de que

$$\sum_j \hat{P}_j = \mathbb{I} \quad (\text{A.5})$$

En analogía con estos operadores  $\hat{P}_j$  podemos definir los superoperadores de proyección en el espacio de Liouville, los cuales proyectan un operador arbitrario  $\hat{A}$  en otro operador  $\hat{B}$ .

$$\hat{\hat{P}}_B = \frac{|\hat{B}\rangle \langle \hat{B}|}{(\hat{B}|\hat{B})} \quad (\text{A.6})$$

### A.2.3. Los Superpropagadores

Nuevamente, en analogía con los propagadores que definimos en el espacio de Hilbert, definimos los superpropagadores como superoperadores que determinan la evolución de un estado de Liouville. Con la siguiente expresión quedan definidos los superpropagadores:

$$\begin{aligned} \hat{\hat{U}}(\phi) |\hat{Q}_j\rangle &= \exp\left(-i\phi\hat{\hat{Q}}\right) |\hat{Q}_j\rangle \\ &= \left| \exp\left(-i\phi\hat{\hat{Q}}\right) \hat{Q}_j \exp\left(i\phi\hat{\hat{Q}}\right) \right\rangle \end{aligned} \quad (\text{A.7})$$

Luego, de la misma forma que lo hicimos en el espacio de Hilbert, definimos los superoperadores de evolución

$$\hat{\hat{L}}(t) = \exp\left(-\frac{i}{\hbar}\mathcal{H}t\right) \quad (\text{A.8})$$

La ventaja de trabajar en el espacio de Liouville resulta obvia si procesos disipativos, como por ejemplo relajación, son incluidos desde un tratamiento mecano cuaántico.

Estos procesos los podemos introducir en el formalismo de Liouville si introducimos una matriz real en la expresión para el superpropagador:

$$\begin{aligned} \hat{\hat{L}}(t) &= \exp\left(-i\hat{\hat{\mathcal{L}}}t\right) \\ &= \exp\left[-it\left(\frac{\hat{\hat{\mathcal{H}}}}{\hbar} - i\hat{\hat{\Gamma}}\right)\right] \end{aligned} \quad (\text{A.9})$$

$\hat{\hat{\mathcal{L}}}$  es el superoperador del sistema, el término  $\hat{\hat{\mathcal{H}}}$  representa el Liouvilliano y el término  $\hat{\hat{\Gamma}}$  denota el superoperador de relajación.

#### A.2.4. Superpropagadores para representar pulsos de rf

Análogamente a como definimos los operadores que representan un pulso en el espacio de Hilbert, definimos los superoperadores que representan pulsos de rf en el espacio de Liouville.

En forma general

$$\widehat{\widehat{P}}(\phi, \beta) = \exp\left(-i\phi\widehat{\widehat{I}}_z\right) \exp\left(-i\beta\widehat{\widehat{I}}_y\right) \exp\left(i\phi\widehat{\widehat{I}}_z\right) \quad (\text{A.10})$$

Donde  $\phi$  es el ángulo entre el eje  $\mathbf{y}$  en la terna rotante y el eje de rotación también en esa terna.  $\beta$  es el ángulo de rotación del pulso. Y vamos a estar interesados en los casos particulares:

$$\widehat{\widehat{P}}_x(\beta) = \exp\left(-i\beta\widehat{\widehat{I}}_x\right) \quad (\text{A.11})$$

$$\widehat{\widehat{P}}_y(\beta) = \exp\left(-i\beta\widehat{\widehat{I}}_y\right) \quad (\text{A.12})$$

Si queremos ver, por ejemplo el efecto de un pulso  $\widehat{\widehat{P}}_y(\beta)$  sobre el estado de Liouville  $|\widehat{\widehat{I}}_z\rangle$

$$\begin{aligned} \widehat{\widehat{P}}_y(\beta) |\widehat{\widehat{I}}_z\rangle &= \exp\left(-i\beta\widehat{\widehat{I}}_y\right) |\widehat{\widehat{I}}_z\rangle \\ &= |\widehat{\widehat{I}}_z\rangle \cos \beta + |\widehat{\widehat{I}}_x\rangle \sin \beta \end{aligned} \quad (\text{A.13})$$

#### A.2.5. Operadores para representar secuencias de pulsos

Vamos a estar interesados muchas veces en aplicar secuencias de pulsos al sistema de forma tal de lograr algún tipo de evolución.

Veamos como trabajar con este tipo de secuencias desde el formalismo del espacio de Liouville.

Supongamos una secuencia

$$P_a - L_a - P_b - L_b - P_c - L_c$$

$P_j$  representa las dinámicas de espines durante los pulsos y  $L_j$  la dinámica entre pulsos.

Luego la correspondiente evolución temporal de la matriz densidad puede ser formalmente representada por como:

$$\begin{aligned} |\widehat{\rho}(t)\rangle &= \widehat{\widehat{\mathcal{L}}} |\widehat{\rho}_0\rangle \\ &= \dots \widehat{\widehat{L}}_b \widehat{\widehat{P}}_b \widehat{\widehat{L}}_a \widehat{\widehat{P}}_a |\widehat{\rho}_0\rangle \end{aligned} \quad (\text{A.14})$$

El superpropagador  $\widehat{\widehat{\mathcal{L}}}$  y su correspondiente propagador  $\widehat{\mathcal{L}}$  representan la secuencia completa de pulsos, incluyendo los períodos de precesión libre. Los llamaremos respectivamente superpropagador y propagador de secuencias de pulsos.

### A.2.6. Función Respuesta:

Una descripción general de la evolución temporal de un sistema de espines nos la brinda la función respuesta. veremos una forma de llegar a una expresión para la función respuesta, también desde el formalismo de superoperadores.

Los espectrómetros estándar de resonancia magnética producen una señal proporcional al valor de expectación de los operadores de magnetización  $I_a$ ,  $a = x, y, z$

$$\begin{aligned}\langle I_a \rangle &= \text{Tr} [\hat{I}_a \hat{\rho}] \\ &= (\hat{I}_a^\dagger | \hat{\rho})\end{aligned}\tag{A.15}$$

Una señal es generada al aplicar una secuencia de pulsos de rf, resultando una respuesta transiente, la cual se puede describir con la siguiente función de respuesta lineal:

$$G_a(t) = \frac{\langle \hat{I}_a \rangle}{\text{Tr} [\hat{I}_a^2]} = \frac{\text{Tr} [\hat{I}_a \hat{\rho}(t)]}{\text{Tr} [\hat{I}_a^2]} = \frac{(\hat{I}_a^\dagger | \hat{\rho}(t))}{(\hat{I}_a | \hat{I}_a)}\tag{A.16}$$

Luego, las funciones respuesta  $G_x(t)$  y  $G_y(t)$  pueden ser combinadas mediante la relación

$$\hat{I}_\pm = \hat{I}_x \pm i\hat{I}_y$$

Con lo cual obtenemos la función respuesta compleja

$$G(t) = \frac{\langle \hat{I}_\pm \rangle}{\text{Tr} [\hat{I}_z^2]} = \frac{\text{Tr} [\hat{I}_\pm \hat{\rho}(t)]}{\text{Tr} [\hat{I}_z^2]} = \frac{(\hat{I}_\mp | \hat{\rho}(t))}{(\hat{I}_z | \hat{I}_z)}\tag{A.17}$$

Si estamos interesados en describir la función respuesta en el caso de resonancia magnética pulsada, utilizando la relación (A.14), obtenemos:

$$G_a(t) = \frac{(\hat{I}_a | \hat{\mathcal{L}} | \hat{\rho}_0)}{(\hat{I}_z | \hat{I}_z)} = \frac{\text{Tr} [\hat{I}_a^\dagger \hat{\mathcal{L}} \hat{\rho}_0]}{\text{Tr} [\hat{I}_z^2]}\tag{A.18}$$

$$G(t) = \frac{(\hat{I}_a | \hat{\mathcal{L}} | \hat{\rho}_0)}{(\hat{I}_z | \hat{I}_z)} = \frac{\text{Tr} [\hat{I}_\pm^\dagger \hat{\mathcal{L}} \hat{\rho}_0]}{\text{Tr} [\hat{I}_z^2]}\tag{A.19}$$

## Apéndice B

# Simulaciones numéricas

Estos programas fueron realizados en FORTRAN 90

### B.1. Simulaciones utilizadas en el capítulo 4

#### B.1.1. Segundos momentos locales $C_{60}$

```
program momentosloc
use portlib
implicit none
real a
integer semilla,p,sitios,r2,v,s
integer(1), allocatable :: coord(:,:,:),spins(:,:)
real(8) sum,sum11,sum12,sum22,Pi
integer(1) i,j,k,N,z,l
integer sitios11,sitios12,sitios22,x(3),y(3),centro(3)

print*, 'semilla,N?'
read*,semilla,N
call random_seed()

open(1,file='suma.dat')
open(2,file='dist_vecinos_mag.dat')

Pi= dacos(-1d0)

do z=1,5
!do N = 10,20
allocate(coord(0:2*N-2,0:2*N-2,0:2*N-2))

do i=0,2*N-2
do j=0,2*N-2
```

```

        do k=0,2*N-2
coord(i,j,k)=0
enddo
        enddo
enddo

!sorteo de sitios magneticos en la red

sitios = 0
do i = 0,2*N-2
    do j = 0,2*N-2
        do k= 0,2*N-2
            if ( mod(i+j+k,2)==0 ) then

                call random_number(a)
                if (a < 0.33) then

                    coord(i,j,k)=1
                    sitios = sitios + 1
                else if (a < 0.43) then

                    coord(i,j,k)=2
                    sitios = sitios + 1
                else
                    coord(i,j,k)=0
                endif
            endif
        enddo
    enddo
enddo

!seteo de array con los sitios magneticos

allocate(spins(1:sitios,1:4))
centro(1)=N-1
centro(2)=N
centro(3)=N

p=0
s=0
do i = 0,2*N-2
    do j = 0,2*N-2
        do k = 0,2*N-2
            if (coord(i,j,k)>0) then
p=p+1

```

```

        spins(p,1)=i
        spins(p,2)=j
        spins(p,3)=k
x(1)=i
x(2)=j
x(3)=k
if (dot_product(x-centro,x-centro) >((N-5)*(N-5))) then
spins(p,4)=0
else
spins(p,4)=1
s=s+1
endif

        endif
    enddo
enddo
    enddo
endif

print*,s

sum11=0
sum22=0
sum12=0
sitios11=0
sitios12=0
sitios22=0
do v = 1,sitios
if (spins(v,4)==1) then
call random_number(a)
if (a < 500.0/float(s)) then
sum = 0

        do j = 1,3
x(j) = spins(v,j)
        enddo
        do p = 1,sitios
            do j = 1,3
y(j) = spins(p,j)
            enddo
r2 =dot_product(y-x,y-x)

if(p==v .or. r2 > 10)then
sum=sum
else
sum = sum + coord(x(1),x(2),x(3)) * coord(y(1),y(2),y(3)) * ((1- 3d0 * dfloat(( y(
! write(2,*) v,r2

```



```

endif

    enddo
    write(1,*) sum*3961614.583/4.0/(Pi)**2
    endif
endif
enddo
deallocate(spins)
deallocate(coord)
!rewind(2)
!enddo
enddo
end program

```

### B.1.2. Aproximación de espines distintos

```

program Espines_distintos
use portlib
implicit none
real a
integer semilla,p,sitios,r2,v,s
integer(1), allocatable :: coord(:,:,:),spins(:,:)
real(8) sum,Pi,aij,tau,m,r,cosij,theta,phi
integer(1) i,j,k,N,z
integer x(3),y(3),dr(3),h(3),centro(3),q

print*, 'semilla?'
read*,semilla
call random_seed()
N=15

open(1,file='Iy3.dat')

tau=1d-3
Pi= dacos(-1d0)
allocate(coord(0:2*N-2,0:2*N-2,0:2*N-2))

do q=1,50

sum=0
theta=0
phi=Pi/40
do z=0,40
theta=theta+Pi/10
h(1)=sin(theta)*cos(phi)
h(2)=sin(theta)*sin(phi)

```

```
h(3)=cos(theta)
print*,q,z
```

```
do i=0,2*N-2
  do j=0,2*N-2
    do k=0,2*N-2
      coord(i,j,k)=0
    enddo
  enddo
enddo
```

```
!sorteo de sitios magneticos en la red
```

```
sitios = 0
do i = 0,2*N-2
  do j = 0,2*N-2
    do k= 0,2*N-2
      if ( mod(i+j+k,2)==0 ) then

        call random_number(a)
        if (a < 0.33) then

          coord(i,j,k)=1
          sitios = sitios + 1
        else if (a < 0.43) then

          coord(i,j,k)=2
          sitios = sitios + 1
        else
          coord(i,j,k)=0
        endif
      endif
    enddo
  enddo
enddo
```

```
!seteo de array con los sitios magneticos
```

```
allocate(spins(1:sitios,1:4))
centro(1)=N-1
centro(2)=N
centro(3)=N
```

```
p=0
s=0
```

```

do i = 0,2*N-2
  do j = 0,2*N-2
    do k = 0,2*N-2
      if (coord(i,j,k)>0) then
p=p+1

        spins(p,1)=i
        spins(p,2)=j
        spins(p,3)=k
x(1)=i
x(2)=j
x(3)=k
if (dot_product(x-centro,x-centro) >((N-5)*(N-5))) then
spins(p,4)=0
else
spins(p,4)=1
s=s+1
endif

      endif
    enddo
  enddo
enddo

!print*,s

do v = 1,sitios
  if (spins(v,4)==1) then
    do j = 1,3
      x(j) = spins(v,j)
    enddo
    m=1
    do p = 1,sitios
      do j = 1,3
        y(j) = spins(p,j)
      enddo

      dr=y-x
      r2=dot_product(dr,dr)
      r =dsqrt(dfloat(r2))
      if(p==v .or. r2 > 10)then
        m=m
      else
cosij=(dot_product(dr,h))/r
aij=135.02d0*(1- 3d0*cosij*cosij)/(r**3)
m=m*cos(aij*q*tau)
      endif
    enddo
  enddo
enddo

```

```

        enddo
        sum=sum+m
    endif
enddo
deallocate(spins)
enddo
write(1,*) q,sum
enddo
deallocate(coord)
end program

```

## B.2. Simulación utilizada en el capítulo 6

```

program error_y_frec_promediados
use portlib
implicit none
complex(8) e(4,4)
complex(8) Id(4,4),Ix(4,4),Iz(4,4),Iy(4,4)
complex(8) Xmas(4,4),Xmenos(4,4),Ymas(4,4),Ymenos(4,4),rhoCP(4,4),rhoMG(4,4),rhoFid
complex(8) UiCP(4,4),UpCP(4,4),UpMG(4,4),UiMG(4,4),TT1(4,4)
complex(8) TRcarr,TrMeiboom,TRfreed,TRcarrx,TrMeiboomx,TRfreedx,TRcarrz,TrMeiboomz,
complex(8) Carr(4,4),Meiboom(4,4),Freed(4,4),Carrx(4,4),Meiboomx(4,4),Freedx(4,4),C
real(8), allocatable:yc(:),ym(:),yf(:),ycx(:),ymx(:),yfx(:),ycz(:),ymz(:),y fz(:)
real(8) Pi,d1,d2,d,tau,delta,Hds(4,4),Hz(4,4)
real ss,r,aa,bb

integer n,pulsos,i,l

open(1,file='CPy_y_MGy.dat')
open(2,file='fid.dat')
Pi=dacos(-1d0)
call random_seed()

!constantes, ngulos y frecuencia

d =88.8d0
tau =1d-3

!defino matrices identidad y de Spin
    Id(1,1) = 1
    Id(1,2) = 0
    Id(1,3) = 0
    Id(1,4) = 0
    Id(2,1) = 0
    Id(2,2) = 1

```

```

Id(2,3) = 0
Id(2,4) = 0
Id(3,1) = 0
Id(3,2) = 0
Id(3,3) = 1
Id(3,4) = 0
Id(4,1) = 0
Id(4,2) = 0
Id(4,3) = 0
Id(4,4) = 1

```

```

Ix(1,1) = 0.D0
Ix(1,2) = 0.5D0
Ix(1,3) = 0.5D0
Ix(1,4) = 0.D0
Ix(2,1) = 0.5D0
Ix(2,2) = 0.D0
Ix(2,3) = 0.D0
Ix(2,4) = 0.5D0
Ix(3,1) = 0.5D0
Ix(3,2) = 0.D0
Ix(3,3) = 0.D0
Ix(3,4) = 0.5D0
Ix(4,1) = 0.D0
Ix(4,2) = 0.5D0
Ix(4,3) = 0.5D0
Ix(4,4) = 0.D0

```

```

Iy(1,1) = 0.D0
Iy(1,2) = -(0,1)*0.5D0
Iy(1,3) = -(0,1)*0.5D0
Iy(1,4) = 0.D0
Iy(2,1) = (0,1)*0.5D0
Iy(2,2) = 0.D0
Iy(2,3) = 0.D0
Iy(2,4) = -(0,1)*0.5D0
Iy(3,1) = (0,1)*0.5D0
Iy(3,2) = 0.D0
Iy(3,3) = 0.D0
Iy(3,4) = -(0,1)*0.5D0
Iy(4,1) = 0.D0
Iy(4,2) = (0,1)*0.5D0
Iy(4,3) = (0,1)*0.5D0
Iy(4,4) = 0.D0

```

```

Iz(1,1) = 1.D0
Iz(1,2) = 0.D0

```

```

Iz(1,3) = 0.D0
Iz(1,4) = 0.D0
Iz(2,1) = 0.D0
Iz(2,2) = 0.D0
Iz(2,3) = 0.D0
Iz(2,4) = 0.D0
Iz(3,1) = 0.D0
Iz(3,2) = 0.D0
Iz(3,3) = 0.D0
Iz(3,4) = 0.D0
Iz(4,1) = 0.D0
Iz(4,2) = 0.D0
Iz(4,3) = 0.D0
Iz(4,4) = -1.D0

```

```
!defino el Hamiltoniano del sistema
```

```

Hds(1,1) = 0.5d0
Hds(1,2) = 0
Hds(1,3) = 0
Hds(1,4) = 0
Hds(2,1) = 0
Hds(2,2) = -0.5d0
Hds(2,3) = -0.5d0
Hds(2,4) = 0
Hds(3,1) = 0
Hds(3,2) = -0.5d0
Hds(3,3) = -0.5d0
Hds(3,4) = 0
Hds(4,1) = 0
Hds(4,2) = 0
Hds(4,3) = 0
Hds(4,4) = 0.5d0

```

```

Hds=d*Hds
pulsos=500
allocate(yc(pulsos))
allocate(ym(pulsos))
allocate(yf(pulsos))
allocate(ycx(pulsos))
allocate(ymx(pulsos))
allocate(yfx(pulsos))
allocate(ycz(pulsos))
allocate(ymz(pulsos))
allocate(yfz(pulsos))

```

```

do n=1,pulsos
yc(n)=0
ym(n)=0
yf(n)=0
ycx(n)=0
ymx(n)=0
yfx(n)=0
ycz(n)=0
ymz(n)=0
yFz(n)=0
enddo
!pulsos de Pi con error delta
r=2*100      !con el primer factor indico el porcentaje que quiero para esa frecuencia
do l=1,r
print*,l
delta=0
call random_number(ss)
!delta=-0.1+0.2d0*ss
call random_number(aa)
d2=800-1600*aa
call random_number(bb)
d1=800-1600*bb
      Hz(1,1) = d1+d2
      Hz(1,2) = 0.D0
      Hz(1,3) = 0.D0
      Hz(1,4) = 0.D0
      Hz(2,1) = 0.D0
      Hz(2,2) = d1-d2
      Hz(2,3) = 0.D0
      Hz(2,4) = 0.D0
      Hz(3,1) = 0.D0
      Hz(3,2) = 0.D0
      Hz(3,3) = d2-d1
      Hz(3,4) = 0.D0
      Hz(4,1) = 0.D0
      Hz(4,2) = 0.D0
      Hz(4,3) = 0.D0
      Hz(4,4) = -d1-d2
Hz=0.5d0*Hz
call exponencial(e,(Hds+Hz)*tau*(0,1),30)
TT1=e
call exponencial(e,Ix*2*Pi*(1+delta)*(0,1d0),30)
Xmas = e
call exponencial(e,Iy*(0,1d0)*2*Pi*(1+delta),30)
Ymas =e
call exponencial(e,-Ix*(0,1d0)*2*Pi*(1+delta),30)
Xmenos =e

```

```
call exponencial(e,-Iy*(0,1d0)*2*Pi*(1+delta),30)
Ymenos =e
```

```
!Operadores evolucion, Evolucion de la matriz densidad
```

```
UiCP =matmul(matmul(TT1,Xmas),TT1)
UpCP =matmul(matmul(TT1,Xmenos),TT1)
UiMG=matmul(matmul(TT1,Ymas),TT1)
UpMG=matmul(matmul(TT1,Ymenos),TT1)
```

```
rhoCP=Iy
rhoMG=Iy
rhoFid=Iy
```

```
do n=1,pulsos
!if (mod(n,2)==1) then                                !lo usamos para +X-X+X-X
    !if (mod(n,4)==1.or. mod(n,4)==0) then !Lo usamos para +X-X-X+X
        rhoCP=matmul(matmul(transpose(dconjg(UiCP)),rhoCP),UiCP)
        rhoMG=matmul(matmul(transpose(dconjg(UiMG)),rhoMG),UiMG)
    !else
!rhoCP=matmul(matmul(transpose(dconjg(UpCP)),rhoCP),UpCP)
!rhoMG=matmul(matmul(transpose(dconjg(UpMG)),rhoMG),UpMG)
!end if
rhoFid=matmul(matmul(transpose(dconjg(TT1)),rhoFid),TT1)

Carr=matmul(rhoCP,Iy)
Meiboom=matmul(rhoMG,Iy)
freed=matmul(rhoFid,Iy)
```

```
TRcarr=0
TRmeiboom=0
TRfreed=0
```

```
do i=1,4
TRcarr=TRcarr + Carr(i,i)
TRmeiboom=TRmeiboom + Meiboom(i,i)
TRfreed=TRfreed + freed(i,i)
enddo
```

```
yc(n)=yc(n)+dReal(TRcarr)
```



```

ym(n)=ym(n)+dReal(TRmeiboom)
yf(n)=yf(n)+dReal(TRfreed)

enddo

enddo
do n=1,pulsos
write (1,*)n,((-1)**n)*yc(n),ym(n)
write (2,*)n,yf(n)
enddo
end program

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

subroutine exponencial(e,A,p) !Hace Taylor hasta orden p
complex(8) AA(4,4),A(4,4),e(4,4),Id(4,4)
integer p
real(8) fact
Id(1,1) = 1d0
Id(1,2) = 0
Id(1,3) = 0
Id(1,4) = 0
Id(2,1) = 0
Id(2,2) = 1d0
Id(2,3) = 0
Id(2,4) = 0
Id(3,1) = 0
Id(3,2) = 0
Id(3,3) = 1d0
Id(3,4) = 0
Id(4,1) = 0
Id(4,2) = 0
Id(4,3) = 0
Id(4,4) = 1d0

fact=1d0
e=Id
AA=A
do n=1,p
fact=fact*dfloat(n)
e=e+AA/fact
AA=matmul(AA,A)
enddo
endsubroutine

```

## Apéndice C

# Programas de medición

Se muestra en detalle el programa utilizado para realizar la medición en la que estábamos interesados en ver comportamiento par-impar. Los demás programas utilizados son los usuales para cada secuencia de pulsos.

```
; parimpar.PC

; CARR-PURCELL/GILL-MEIBOHM T2 SEQUENCE
; RECORDING DIRECTLY THE T2 CURVE

TAU1=VD-D7-D7-D7-D7-D7-D7
TAU2=VD-D7-D7-D7-D7-D7-D7

PROT F1 F2 XT
START, D1      [F1 @PLS1 STA RGATE]      ; 90 DEGREE PULSE AND TRIGGER
      VD
;
;
      LOOP C1 TIMES                        ; DO TD ECHOES
;
      D2      [F1 @PLS2 RGATE]            ; 180 DEGREE REFOCUSING PULSE
      TAU2

      D7                        ; SAMPLE 1 POINT
      2U [XAD]
      D7                        ; SAMPLE 1 POINT
      2U [XAD]
      D7                        ; SAMPLE 1 POINT
      2U [XAD]
      D7                        ; SAMPLE 1 POINT
      2U [XAD]
      D7                        ; SAMPLE 1 POINT
      2U [XAD]
      D7                        ; SAMPLE 1 POINT
      2U [XAD]
```

```

        D7                ; SAMPLE 1 POINT
        2U  [XAD]
        D7                ; SAMPLE 1 POINT
        2U  [XAD]
        D7                ; SAMPLE 1 POINT
        2U  [XAD]
        D7                ; SAMPLE 1 POINT
        2U  [XAD]
        D7                ; SAMPLE 1 POINT
        2U  [XAD]
        D7                ; SAMPLE 1 POINT
        2U  [XAD]
        D7                ; SAMPLE 1 POINT
        2U  [XAD]
        TAU1
        END LOOP
        DO
        DO
        ++PLS1
GOTO START

BEGIN LISTS
PLS1,   +X +X -X -X
PLS2,   +X -X +X -X           ;SECUENCIA TIPO CPAF2
RLS,    +X +X -X -X
END LISTS
; DEFINE VD LIST, START WITH TONEAQ.AUM
; USE MULTIPLE OF 2 AS TD
; USE QF XA RPN

```

# Bibliografia

- [1] A.E. Dementyev, D. Li, K. MacLean, and S. E. Barret. Anomalies in the nmr of silicon: unexpected spin echoes in a dilute dipolar solid. *Physical Review B*, 68(153302), 2003.
- [2] W.Harneit, C. Meyer, A. Weidinger, D. Suter, and J. Twamley. Architectures for a spin quantum computer based on endohedral fullerenes. *Phys. stat. sol (b)*, 233(3), 2002.
- [3] W. Harneit. A fullerene based electron spin quantum computer. *Physical Review A*, 65, 2002.
- [4] P.W. Anderson. Local moments and localized states. *Reviews of Modern Physics*, 50(2), 1978.
- [5] P.W. Anderson. Absense of diffusion in certain random lattices. *Physical Review*, 109(5), 1958.
- [6] C.P. Slichter. *Principles of magnetic resonance*. Springer-Verlag, 1992.
- [7] G. Usaj. *Tesis Doctoral*. 1999.
- [8] E. Fukushima and S.B.W Roeder. *Experimental pulse NMR, A nuts and bolts approach*. Addison Wesley, 1981.
- [9] L. E. Reichl. *A modern course in statistical physics*. John Wiley and Sons, Inc., 1998.
- [10] E. L. Hahn. Spin echoes. *Physical Review*, 80(4), 1950.
- [11] H. Y. Carr and E. M. Purcell. Effects of diffusion on free precession in nuclear magnetic resonance experiments. *Physical Review*, 94(3), 1954.
- [12] R. Tycko, G. Dabbagh, R.M. Flemming, R.C.Haddon, A.V. Makhija, and S.M. Zahurak. Molecular dynamics and the phase transition in solid C<sub>60</sub>. *Physical Review Letters*, 67(14), 1991.
- [13] A. Abragam. *The principles of nuclear magnetism*. Oxford University Press, 1961.
- [14] M. Mehring and V. A. Weberruß. *Object-Oriented Magnetic Resonance*. Academic Press, 2001.

- [15] R. R. Ernst, G. Bodenhausen, and A. Wokaun. *Principles of Nuclear Magnetic Resonance in One and Two Dimensions*. Oxford Science Publications, 1988.