Máster en Física y Tecnologías Físicas

Trabajo de fin de Máster

Optimización de software cuántico para qubits moleculares

Dorye L. Esteras Córdoba

Directores:
David Zueco Láinez
Fernando Luis Vitalla

Curso 2018-2019

Índice

Introducción 1

Contexto 2

Objetivos 4

I Computación cuántica 4

1. El algoritmo de Shor: la gran carrera 4

2. Decoherencia 5

3. El qubit 7

4. Operar sobre qubits: oscilaciones de Rabi 9

5. Qubits moleculares 11

II Corrección de errores en un qudit molecular 12

6. Modelo Efectivo: GdEr 12

7. EPR: Electron Paramagnetic Resonance 15

8. Simulación del Hamiltoniano 20

9. Corrección de errores 24

10. Conclusiones 29

Agradecimientos Quiero comenzar este trabajo agradeciendo a mis directores David Zueco Láinez y Fernando Luis Vitalla, los dos años que he podido pasar con ellos. También quiero agradecer, el tan agradable trato y la valiosa ayuda que he recibido del profesor Dr. Pablo Alonso. Por último quiero agradecer a mis padres, el apoyo y la paciencia que han tenido durante estos años de estudio.

Introducción

La computación cuántica es uno de los campos de estudio más en boga, que manifiesta la clara necesidad de manipular átomos en la tecnología actual, hecho que fue predicho por R. Feynman en su perspectiva de las tecnologías futuras [5]. A pesar de originarse en los años 80, gracias al pensamiento de grandes mentes como Richard Feynman y Paul Benioff, y los avances en algoritmos de otros grandes físicos como David Deutsch, Charles Benett, Lov Grover o Peter Shor, la computación cuántica no se convierte en una teoría aplicable hasta el comienzo del nuevo milenio, donde su implementación se vuelve real con la aparición de los primeros qubits y la implementación de algoritmos en los primeros ordenadores cuánticos. La relevancia de la computación cuántica crece de forma abrumadora, siendo la llave que puede abrir muchas puertas que hasta la fecha se encuentran cerradas. Su aplicación en simulación cuántica podría permitir obtener resultados nunca alcanzados en problemas que hasta la fecha se resisten, como algunos problemas de muchos cuerpos, los problemas con fermiones, y las interacciones a larga distancia. Grandes sistemas de espines podrían ser analizados mediante computación cuántica y también complejas moléculas, lo que está resultando en grandes inversiones por parte de la industria farmacéutica para el desarrollo de nuevos medicamentos. La información y la computación cuántica también pueden ser un relevante aliado en el campo de la inteligencia artificial, donde una de las principales metas es lograr paralelismo, un elemento muy presente en la tecnología cuántica. Por otro lado, el famoso problema de descomposición en factores primos que resuelve el algorítmo de Shor puede romper los sistemas de encriptación y seguridad de nuestras sociedades, de forma que también se ha generado una enorme inversión por parte de los gobiernos. En concreto, la Unión Europea ha creado una misión para desarrollar tecnologías cuántica, abriéndose una carrera por lograr obtener el primer ordenador cuántico en funcionamiento, una carrera que es vista por algunos grandes investigadores como un segundo proyecto Manhattan, necesario para combatir la amenaza que dichas maquinas van a suponer, en lo que quizás se conozca más adelante como la segunda revolución cuántica.

Para poder alcanzar el gran objetivo de esta carrera, es necesario comprender una serie de características que debe tener cualquier ordenador para poder ser funcional. Un ordenador debe estar basado en una tecnología que sea manipulable, escalable y corregible. Además de esto debe proporcionar tiempos de coherencia que permitan operar con un error mínimo. En estas fechas, ya existen ordenadores que han alcanzado los 50 qubits, un tamaño que si bien es deseable aumentar, es una figura de mérito para la que está predicha la supremacía cuántica. Además, con esta cifra es posible trabajar en simulación cuántica de forma satisfactoria, lo que es sin duda una de las grandes aplicaciones de un ordenador cuántico. Respecto a la escalabilidad, existen distintas tecnologías que aparentan permitir escalar dichos sistemas manteniendo su manipulabilidad. Sin embargo, a pesar de que se logren grandes metas en su manipulación y escalabilidad, ningún ordenador puede desempeñar una función si no es corregible, esto es, que pueda someterse a algoritmos de corrección de errores. Esto viene claramente impuesto por dos motivos principales, el primero de ellos es que dado que no existen los sistemas ideales, un ordenador siempre va a tener un error asociado. No llevar a cabo procesos de corrección de errores convierte a un

ordenador en una maquina realmente peligrosa, como puede esperarse de una máquina que se encuentra detrás de todos los procesos de nuestra sociedad. Por otro lado, la existencia de los procesos de decoherencia característicos de la computación cuántica suponen una fuente de error que estará siempre presente, siendo una de las grandes restricciones que nos impone la mecánica cuántica a la hora de hacer computación : decoherencia, no clonación y colapso de la función de onda. La correción de errores es entonces, el pilar que sostiene la posibilidad de crear ordenadores cuánticos, siendo por tanto la principal necesidad en el paradigma actual de la computación cuántica.

Contexto

El concepto de corrección de errores en cuanto a información y comunicaciones se refiere, es algo antiguo en nuestra especie como puede imaginarse, no obstante, su estudio tiene lugar de forma seria durante los años 30 y 40, en una época de guerra donde el surgimiento de la computación comienza a verse, como un elemento fundamental para obtener la victoria en una de las épocas más oscuras de nuestra especie. En el año 1936 comienza a gestarse la idea de una máquina capaz de computar, presentada por Alan Turing en la revista Proceedings of the London Mathematical Society, quién más tarde desarrollaría la primera máquina de computación. En 1946 se desarrolla ENIAC, una de las primeras computadoras junto con Z1 y las máquinas Colossus, que jugaron un papel fundamental en la segunda guerra mundial. Dentro de este marco tan oscuro, en 1948, Shannon publica la primera formulación matemática completa de la información y de las comunicaciones, conocida como la teoría matemática de la comunicación, publicada en el Bell System Technical Journal. En dicha teoría, Shannon da lugar a grandes avances, en primer lugar da una definición formal de la información, asociándola a la entropía de una distribución. Por otro lado, crea y demuestra dos teoremas con importantes implicaciones en teoría de la información, the noiseless coding theorem y the noisy-channel coding theorem. El primero de ellos permite cuantificar la redundancia que existe en un determinado mensaje. La redundancia permite proteger un mensaje frente a errores, su fundamento consiste en enviar en un mensaje información repetida, de forma que los posibles errores no impidan entender el mismo. Esto se puede comprobar de forma visual cuando cambiamos letras en un mensaje, hasta cierto punto, el mensaje puede seguir siendo comprendido debido a que la información contenida en él es redundante al estar basada en un sistema de 27 elementos que no son linealmente independientes. Cuando aumentamos el número de letras que variamos del mensaje, llegamos a un punto donde el mensaje no se puede reproducir. El primero de estos teoremas permite precisamente cuantificar cuanto puede acortarse un mensaje para poder ser decodificado posteriormente sin error. El segundo teorema cuantifica la cantidad de información repetida que debemos incorporar a un mensaje para poder reconstruirlo satisfactoriamente a partir de una señal ruidosa. Es por tanto, una de las primeras bases de los métodos de corrección de errores en teoría de la información.

Como puede verse, la teoría de Shannon es sin duda una de las fuentes principales a la hora de tratar con el error que se produce en la información que porta un sistema, como por ejemplo puede ser un sistema de qubits, de un ordenador cuántico. En concreto el segundo teorema de Shannon aquí mencionado nos dice como combatir el error introduciendo información redundante. No obstante, a pesar de contar con tan selecta teoría, en el campo de la información cuántica surgen una serie de dificultades propias de otra gran teoría, la teoría de la mecánica cuántica. En los sistemas cuánticos, los errores suponen un problema mucho mayor que en sus análogos clásicos. En primer lugar, los sistemas cuánticos tienen un tamaño mucho más reducido, y por ende, son mucho más susceptibles a las influencias y fluctuaciones del medio. Por otro lado, un bit clásico posee unicamente dos estados, $|0\rangle$ y $|1\rangle$, de forma que solo existe un tipo de error, el cambio de uno de dichos estados por el otro. Sin embargo, un bit cuántico, es decir un qubit, está formado por cualquier superposición de ambos estados, incluyendo también cualquier fase, de forma que existe una infinidad de estados posibles que un error puede generar, existiendo una gran cantidad de tipos de error que se deben controlar. Además de esta gran complejidad del error, debemos añadir otras grandes limitaciones de la mecánica cuántica que he mencionado en la introducción. En principio solo podemos detectar un error realizando una medida del sistema, lo que produce un colapso de la función de onda como dictan los postulados de la mecánica cuántica. Parece ser entonces que detectar un error de nuestro sistema implica destruirlo, cosa que no es en absoluto deseable. La solución a este problema podría ser, como sucede en computación clásica, la copia de estados. Obtener varios clones de un estado, nos permitiría operar en cada uno de ellos sin perder el estado original, pero precisamente el proceso de clonación se encuentra prohibido por el teorema de no clonación de la mecánica cuántica, que es la otra gran limitación mencionada anteriormente. Ante las dificultades que la teoría cuántica produce en computación, muchos años después, en 1996, Shor y Steane propusieron una serie de protocolos de corrección de errores en base al código de Hamming para bits clásicos. Dichos protocolos requieren de 9,7 o como mínimo 5 qubits para proteger un solo qubit de cualquier error aleatorio. En concreto se puede implementar una versión simplificada de estos códigos, que protege a un qubit de ciertos errores, tanto en fase como en módulo y que aunque no protege al qubit de cualquier tipo de error, supone un primer paso bastante importante en el marco actual de la computación cuántica para implementar correción de errores en sistemas reales. Siguiendo una propuesta moderna de 2018 [7], donde se consigue implementar un método de corrección de errores en una molécula de $^{173}Yb(trensal)$, **este trabajo busca** implementar un método de corrección de errores en un qubit real, obtenido con una molécula de GdEr. En ella, el gadolinio posee ocho estados de espín que forman un sistema de tres qubits que albergan al qubit lógico, el ión de erbio contribuye con un estado fundamental formado por sólo dos niveles bien aislados del resto, actuando como un qubit ancilla, que detecta los errores que se hayan producido y ayuda a corregirlos. Dicho procedimiento no ha sido realizado nunca en la molécula de GdEr y busca ser un paso hacia delante en la implementación de qubits mediante la tecnología de espines moleculares y en la implementación de corrección de errores en el marco de la computación cuántica y de la espintrónica, dos campos que pueden revolucionar la tecnología de nuestras sociedades, en un futuro no muy lejano.

Objetivos

- Modelización de la molécula de GdEr.
- Determinación experimental de los parámetros del Hamiltoniano correspondiente.
- Simulación y análisis de la molécula de GdEr para estudiar el sistema qubit-ancilla.
- Preparación teórica de un método de corrección de errores mediante la ancilla.

Parte I
Computación cuántica

Antes de comenzar con el núcleo principal del trabajo, resulta necesario hacer una introducción a todos los conceptos previos que voy a utilizar en él. Dichos conceptos son ideas fundamentales de la computación cuántica que encuentran en este trabajo, una aplicación teórica y experimental.

1. El algoritmo de Shor: la gran carrera

A pesar de que he dedicado una introducción a motivar la relevancia de la computación cuántica, deseo brevemente mostrar una gráfica que de fuerza a todos mis argumentos. A pesar de que sus aplicaciones son diversas y que poseen funciones diferentes en el desarrollo de la sociedad y de la ciencia, la aplicación que realmente ha revolucionado la situación de la computación cuántica, llevándola desde su situación en los años 80, hasta su posición actual como misión de la Unión Europea y como un objetivo a desarrollar a lo largo del planeta, es su papel en criptografía. Nuestras sociedades están basadas en gran medida en la informática, toda nuestra información, nuestras comunicaciones e incluso el dinero que guardamos en los bancos, se encuentra digitalizado y es guardado en servidores informáticos. En concreto las grandes empresas, los bancos y los gobiernos almacenan la información en grandes grupos de servidores que son conocidos comúnmente como granjas de servidores. En este aspecto digital de la sociedad, la seguridad informática es crucial, siendo la que hace posible sostener dichos sistemas de almacenamiento de datos, que hasta hace poco se guardaban físicamente en lugares ocultos y protegidos. La seguridad informática esta basada en sistemas de encriptación de clave pública, siendo el caso más conocido el protocolo RSA. En dichos sistemas, las claves que permiten encriptar son públicas y con ellas se puede obtener la clave de desencriptación, que en principio es conocida solo por una persona. La seguridad reside entonces en que para poder generar dicha clave de desencriptación, es necesario resolver un problema de descomposición en números primos, un problema de dependencia exponencial con el número de digitos, que para claves largas puede requerir un tiempo de computo del orden de la edad del universo. La seguridad informática es entonces absoluta.

En 1994 Peter Shor crea un algoritmo cuántico que permite resolver el problema de números primos, convirtiendo en polinómica la relación exponencial. Dicho comportamiento puede verse en la figura 1.

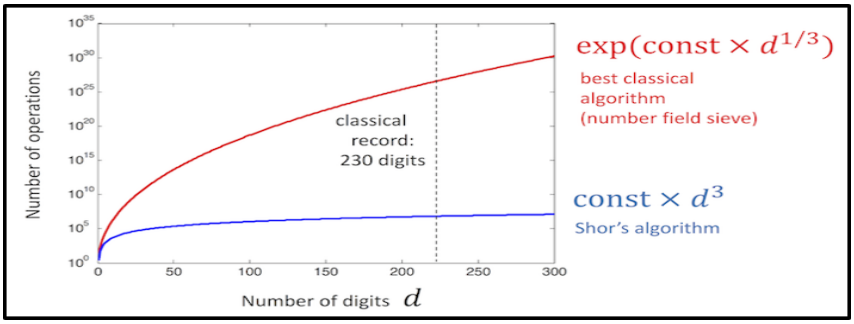


Figura 1: Análisis del problema de descomposición de números primos. Comparativa entre los mejores resultados clásicos y los resultados del algorítimo de Shor. Puede apreciarse el orden exponencial frente al orden polinómico con el que crece el problema en función de la longitud de las claves. También resulta sorprendente el número de operaciones necesarias para una clave de longitud cercana a 200 digitos. Imagen obtenida de la referencia [8].

Como puede imaginarse, el principal problema de dicho algoritmo era la necesidad de un ordenador cuántico, pero en 2001 dicho algorítmo fue implementado por IBM, con la aparición de los primeros ordenadores cuánticos. En ese momento, el problema de la criptografía se hizo real y comenzó una carrera por desarrollar el primer ordenador cuántico funcional que ha desembocado en una ley de Moore para las unidades fundamentales de la computación cuántica, los qubits.

2. Decoherencia

En un trabajo cuyo objetivo principal es implementar un método de corrección de errores, resulta natural comenzar introduciendo el fenómeno de decoherencia, la principal fuente de error de las tecnologías cuánticas. Los procesos de decoherencia son aquellos procesos en los que los estados cuánticos de nuestro sistema, por ejemplo un ordenador cuántico, interaccionan con el exterior transformándose en estados erróneos. Dicha transformación rompe las superposiciones cuánticas, dando al sistema un comportamiento clásico, destruyendo de esta forma el objetivo de crear ordenadores cuánticos. Además de eliminar el comportamiento cuántico, la decoherencia es una fuente de error que hace imposible la computación, se trata de un error de origen aleatorio al provenir de interacciones con multiples fuentes que no controlamos. De esta forma, tenemos decoherencia generada por las impurezas de un material, por los fonones, por núcleos adyacentes y también por los fotones que llegan a nuestro sistema, que resultan en la mayoría de casos incontrolables, siendo capaces de atravesar cualquier aislamiento para ciertas frecuencias. La decoherencia solo puede combatirse mejorando el aislamiento de los sistemas, lo que genera dos problemas fundamentales. El primero es que no existe el aislamiento ideal y perfecto, por tanto siempre tendremos decoherencia. En segundo lugar, aunque pudieramos lograr este aislamiento

perfecto, esto supondría crear sistemas no manipulables, lo que haría imposible la computación. Es necesario asumir entonces, que siempre habrá decoherencia dado que es una consecuencia directa del comportamiento cuántico de la materia, siendo necesario crear un equilibrio entre capacidad de manipulación y de aislamiento. La decoherencia impone una restricción fundamental sobre la computación cuántica, no solo debemos poder hacer operaciones (sistemas manipulables), a gran escala (sistemas escalables) y con gran fidelidad en los resultados (sistemas corregibles), además de esto, en computación cuántica, es necesario operar rápido, en ese breve lapso de tiempo en el que la decoherencia no ha transformado todavía los estados cuánticos de forma importante. Ese tiempo del que disponemos para operar, es el tiempo de coherencia, llamado así porque es el tiempo en el que se preserva la coherencia cuántica. Podemos hablar de dos tiempos de coherencia, coherencia en módulo (T_1) y en fase (T_2), el factor limitante suele ser T_2, siendo la fase lo primero que se pierde y pudiendo tener el módulo, un tiempo de coherencia de horas.

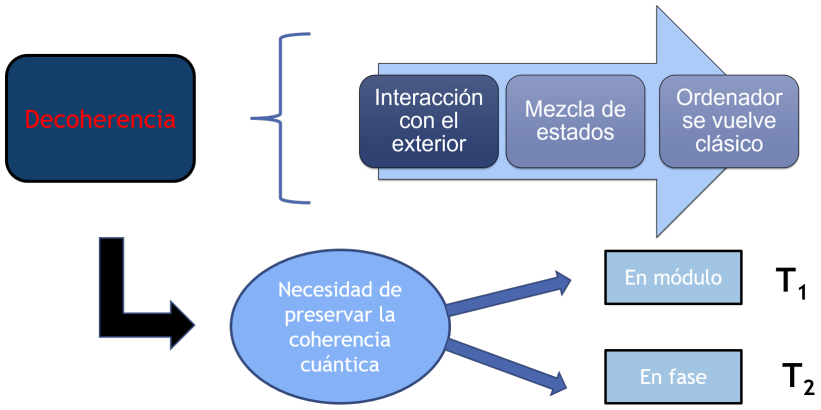


Figura 2: Esquema básico de la decoherencia donde está esquematizada la destrucción del comportamiento cuántico de los sistemas, y la necesidad que surge de obtener coherencia en ordenadores cuánticos logrando valores altos de T_1 y T_2

Una operación en computación cuántica puede llegar a realizarse en tiempos del orden de ns (como sucede con espines, la tecnología usada en este trabajo). Es por esto que un tiempo de coherencia de dicho orden es un mal indicativo, un orden de μs es un buen indicativo y finalmente, los tiempos más largos obtenidos son de segundos (usando espines electrónicos en semiconductores por ejemplo). Como puede apreciarse, la decoherencia tiene muchas implicaciones en el campo de la computación cuántica. Siendo la coherencia una de las cuatro grandes características que debe cumplir un ordenador cuántico.

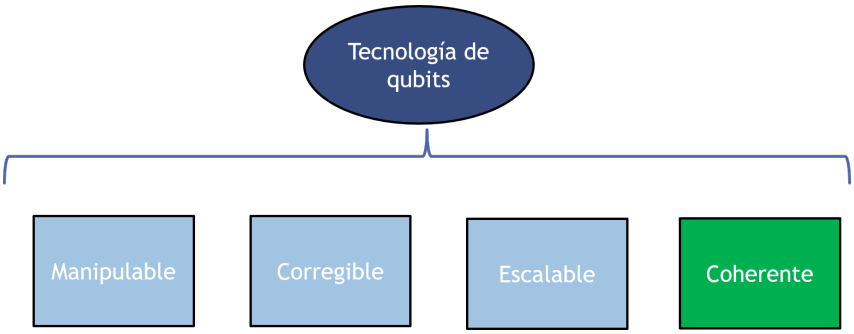


Figura 3: Características que debe tener una tecnología a la hora de implementar qubits para lograr obtener un ordenador cuántico. Se introduce la coherencia como cuarta característica fundamental debida a la necesidad de combatir la decoherencia.

Queda una última implicación, que esta realmente relacionada con este trabajo. Asumir la existencia de los procesos de decoherencia implica asumir que un ordenador cuántico es un sistema erróneo por naturaleza. Muchos son los ejemplos a lo largo de la historia que pueden escenificar la gravedad que puede llegar a tener un error de computo. ¿Puede acaso asumirse el error, como una característica inevitable de una máquina que aspira a crear un nuevo paradigma en la computación, y situarse detrás de los grandes procesos de nuestra sociedad? La respuesta evidentemente es negativa, pero el asilamiento perfecto no existe ni es deseable. Mi respuesta es que la corrección de errores es precisamente el proceso de cuya existencia depende la computación cuántica, es el único factor que puede sostener la posibilidad de hacer computación, con elementos que sufren de error por su propia naturaleza cuántica.

3. El qubit

Un qubit es la unidad básica de la computación cuántica, es una generalización del bit clásico, estando formado no solo por los estados $|0\rangle$ y $|1\rangle$, sino también por cualquier superposición entre ambos. Dicha definición toma la expresión de la superposición cuántica más general, como puede verse en la ecuación 1. Además el qubit puede representarse de forma muy visual en la esfera de Bloch (figura 4) donde el qubit puede expresarse como la combinación de cualquier superposición en módulo de la función de ondas y cualquier fase, como aparece en la ecuación 2.

$$|\Psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \tag{1}$$

Es en este aspecto donde residen las capacidades de paralelismo de un ordenador cuántico, la unidad básica de computación es un sistema de dos niveles cuántico.

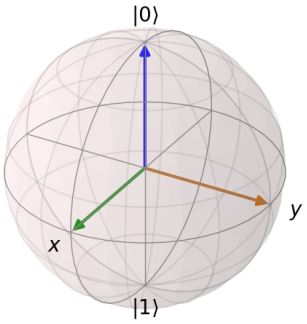


Figura 4: Representación del qubit en la esfera de Bloch. En ella están contenidos cualquier superposición en módulo y cualquier fase.

$$|\Psi\rangle = \cos\frac{\theta}{2}|0\rangle + e^{i\phi}\cos\frac{\theta}{2}|1\rangle \tag{2}$$

Un ejemplo del potencial que tiene esta definición puede verse en simulación cuántica. Para representar un espín con bits clásicos se necesitan del orden de 2^n bits, mientras que se necesitan unicamente n qubits. Eliminar la dependencia potencial con n, que representa el tamaño del sistema, suele significar en computación permitir que un problema irresoluble, pueda tener solución con un tiempo de computo finito.

Desde un punto de vista experimental, un qubit es todo soporte físico que permita implementar un sistema de dos niveles cuánticos. Existe una gran variedad de posibilidades, las más comunes son chips superconductores, iones atrapados y espines. En concreto en este trabajo utilizaré como qubit, un espín electrónico obtenido de una molécula de Gadolinio, esto es, un qubit molecular.

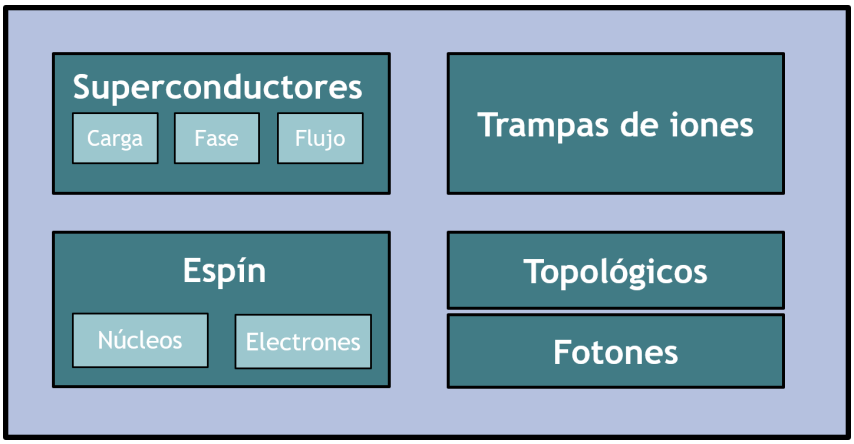


Figura 5: Esquema de las principales tecnologías de implementación de qubits. Las más desarrolladas son las basadas en trampas de iones (I. Cirac, P. Zoller [2]) y las basadas en superconductores. Por otro lado se encuentran las tecnologías basadas en espines, una propuesta muy prometedora. Por último se presentan algunas tecnologías que se encuentran todavía en desarrollo.

4. Operar sobre qubits: oscilaciones de Rabi

En computación cuántica, operar supone realizar una serie de transformaciones de los qubits. En el caso intuitivo de usar un espín como qubit, podemos imaginar estas operaciones como rotaciones del espín, que son capaces de obtener cualquier superposición de estados. Fuera de este ejemplo tan visual, las operaciones no tienen porque ser rotaciones, pero el fundamento es el mismo, operar consiste en realizar transformaciones de los estados del qubit. Dichas trasformaciones deben seguir las leyes de la física para poder ser implementadas, siendo operadores unitarios que cumplen el postulado de evolución (ecuación 3) y vienen restringidas por el teorema de no clonado de la mecánica cuántica, son llamadas puertas lógicas cuánticas.

$$|\Psi_f\rangle = U\,|\Psi_0\rangle \tag{3}$$

El aspecto que resulta transcendental en este trabajo es cómo se implementan estas puertas lógicas cuánticas desde el punto de vista práctico, lo que tiene lugar estimulando con un campo magnetíco dependiente del tiempo. Dicha estimulación, cuando se lleva acabo de forma resonante con una determinada transición, hace oscilar los estados cuánticos generando oscilaciones de Rabi.

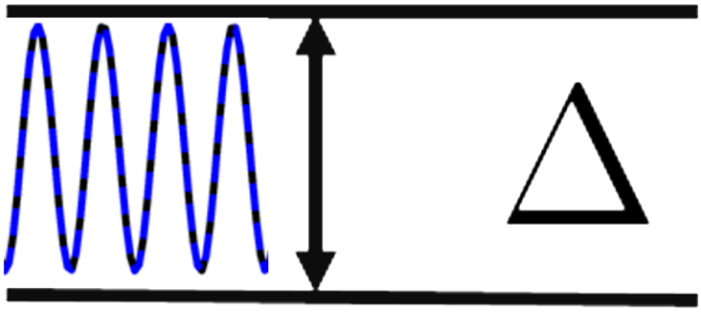


Figura 6: Oscilaciones de Rabi en un qubit (sistema de dos niveles) producidas por una estimulación resonante con un campo magnético dependiente del tiempo. Puede verse como el sistema evoluciona tomando la forma de cualquier superposición para distintos valores de tiempo.

Para el caso de un qubit (sistema de dos niveles) la estimulación de dicho sistema genera la evolución que viene representada en la figura 6. Dicho sistema viene representado de forma general por un Hamiltoniano de la forma de la ecuación 4, donde un determinado Hamiltoniano H_0 representa el sistema de dos niveles y H_1 es el término de Rabi, que representa las oscilaciones. Como puede verse, en resonancia, la evolución de los estados toma la forma de un seno o coseno al cuadrado.

$$H = H_0 + H_1 = \frac{\Delta\sigma_z}{2} + 2\lambda\cos(\omega t)\sigma_x \tag{4}$$

Las oscilaciones de Rabi son reales y para implementar cualquier puerta es necesario estimular con el resto de generadores de SU(2), es decir, con las tres sigmas de Pauli.

Para realizar operaciones sobre más de un qubit se precisa de un Hamiltoniano que describa a los distintos qubits y además, la interacción entre ellos. Para el caso de dos qubit el Hamiltoniano viene representado en la ecuación 5.

$$H_{2qubit} = \sigma_z^{(1)} + \epsilon\sigma_z^{(2)} + J\sigma_z^{(1)}\sigma_z^{(2)} \tag{5}$$

Esto da lugar a un sistema de niveles más complejo que el señalado para un qubit, con distintas transiciones posibles que pueden verse en la figura 7.

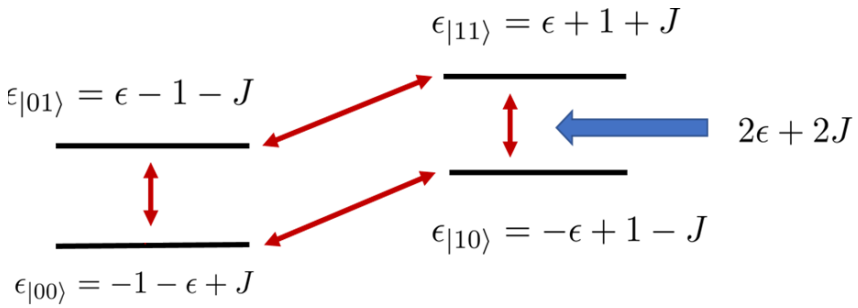
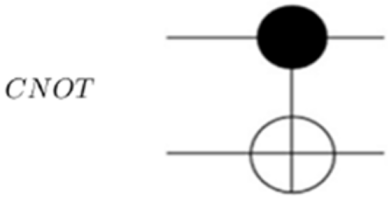


Figura 7: Niveles de energía correspondientes a un sistema de dos qubits. Pueden observarse distintas transiciones. La transición señalada en azul corresponde a la que permite implementar una puerta CNOT de dos qubits.

Seleccionando la transición a estimular, se puede implementar cualquier operación entre dos qubits. En la figura 7 ha sido señalada en azul la transición que permite implementar una puerta CNOT, dicha puerta viene caracterizada en la figura 8.



inicial		final	
$\|0\rangle$	$\|0\rangle$	$\|0\rangle$	$\|0\rangle$
$\|0\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$
$\|1\rangle$	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$
$\|1\rangle$	$\|1\rangle$	$\|1\rangle$	$\|0\rangle$

Figura 8: Puerta CNOT, simbolo y actuación. Puede verse enmarcado en azul, los únicos estados que evolucionan ante su actuación. Es debido a esto que la transición que implementa dicha puerta es la señalada en azul en la figura anterior.

Con el conjunto de operaciones sobre un qubit y una puerta de dos qubits como puede ser la puerta CNOT, se tiene un conjunto universal, con él se puede generar cualquier operación sobre el qubit.

5. Qubits moleculares

Como he comentado, de todas las formas que existen de implementar un qubit, la utilizada en este trabajo es la de usar espines electrónicos alojados en moléculas. A la hora de escoger una tecnología para desarrollar qubits, debemos tener en cuenta una serie de caracteríticas que determinaran nuestra capacidad de obtener un ordenador cuántico, dichas características han sido resumidas en la figura 3. En dichos aspectos los espines electrónicos, especialmente aquellos proporcionados por moléculas, sobresalen a la hora de implementar qubits. Los espines correspondientes a moléculas tienen una serie de niveles de energía discretos y su **manipulación** puede llevarse acabo de una forma muy sencilla mediante campos magnéticos. Además de esto, los espines proporcionados por moléculas tienen una gran **escalabilidad** dado que la molécula es la unidad más escalable que existe, siendo una unidad de tamaños muy diversos y que resulta fácilmente replicada mediante procedimientos químicos para generarlas en grandes números . A esto es necesario sumarle que los espines, son insensibles al campo eléctrico, siendo inmunes a la decoherencia producida por el mismo. Al usar espines electrónicos de moléculas, también se tienen sistemas muy insensibles a la decoherencia producida por campos magnéticos, ya que el momento magnético del electrón es muy pequeño. Todo esto hace que los qubits implementados con espínes moléculares ofrezcan unos tiempos de **coherencia** de los más largos conseguidos hasta la fecha. Las principales fuentes de decoherencia son debidas a las vibraciones (térmicas y de fonones), a los espines nucleares, especialmente del hidrógeno y del nitrógeno y a espines elecťronicos externos al sistema. El problema de las vibraciones puede solventarse con bajas temperaturas y buscando técnicas ingeniosas para bloquear las transciones generadas por los fonones. La influencia de los espines nucleares es el punto más complicado, dado que para eliminar la influencia del hidrógeno por ejemplo, sería necesario renunciar a toda la química orgánica. Por otro lado el problema de los espines electrónicos externos es también complicado, porque como se ha comentado anteriormente, aislar significa perder capacidad de manipulación. Es necesario por tanto, seguir desarrollando técnicas para obtener tiempos de coherencia largos y que a la vez permitan tener interacción fuerte entre qubits. Otro aspecto muy interesante de usar moléculas para hacer qubits que podría dar solución a estas fuentes de ruido, es la capacidad de diseñar las moléculas y su entorno para optimizarlas en distintas funciones, por ejemplo se podría diseñar sus niveles energéticos para hacer al qubit insensible al ruido magnético o bloquear determinadas transiciones, como las debidas a fonones. El bloqueo de transiciones podría entre otras cosas, funcionar como un interruptor que active o desactive dichas transiciones para generar operaciones y posteriormente almacenar la información de forma protegida. Otro de los grandes objetivos es conseguir moléculas con varios estados internos accesibles (qudits) que puedan codificar varios qubits, de forma que una sola molécula pudiera ser una unidad capaz de generar operaciones entre varios de ellos, punto en el cuál podría compararse con un procesador actual. En concreto el gadolinio, es uno de los candidatos favoritos para implementar qudits, debido a que es el ión de la tabla periódica con el valor de espín más alto y la anisotropía magnética es suficientemente baja para que todos ellos sean accesibles, por ejemplo mediante técnicas de resonancia con microondas. En la figura 9 se presentan distintos avances en la implementación de qubits moleculares,

puede verse además, su gran variedad.

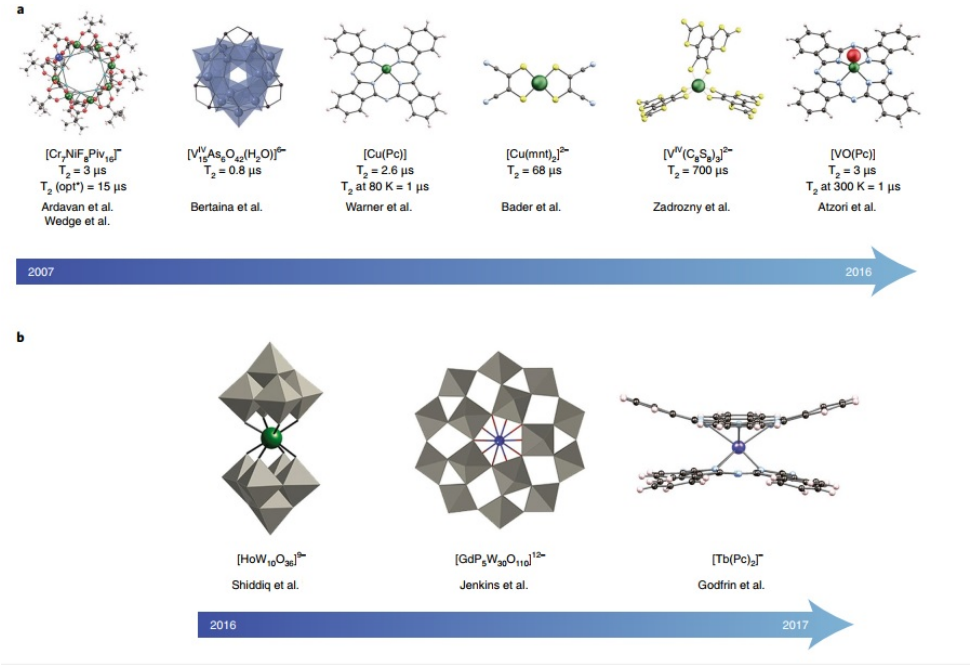


Figura 9: Resumen de los principales logros conseguidos en moléculas en cuanto a tiempos de coherencia y formación de estructuras. Imagen obtenida de [6]

Parte II

Corrección de errores en un qudit molecular

6. Modelo Efectivo: GdEr

En esta sección busco presentar la molécula de GdEr, así como presentar un modelo efectivo que la describa. Los datos experimentales obtenidos por el Prof. Dr. Pablo Alonso, serán posteriormente simulados en Matlab mediante el programa EasySpin para obtener los parámetros fundamentales de dicho modelo. Además la simulación del Hamiltoniano completo en python, permitirá entender el comportamiento del sistema para implementar un método de corrección de errores. La estructura de dicha molécula puede verse en la figura 10, la cuál ha sido obtenida a partir del programa Mercury mediante la representación de un fichero de datos experimental.

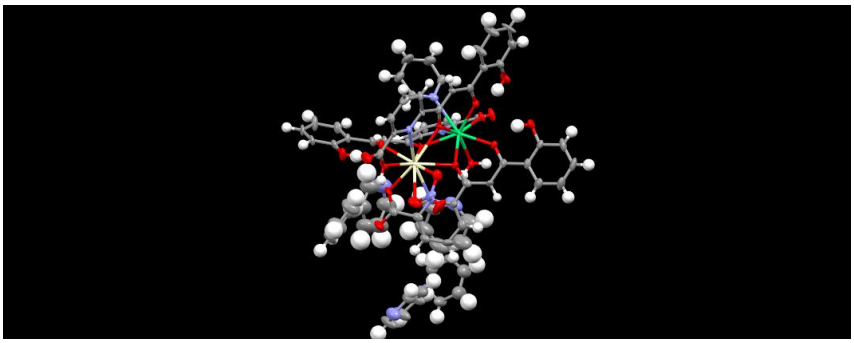


Figura 10: Estructura de la molécula de GdEr. Puede verse una estructura de carbonos en gris, seguida de los oxígenos en rojo y de hidrogenos en blanco. En el centro se encuentran en verde, el átomo de erbio y en color morado el átomo de gadolinio

En el sistema de GdEr, el átomo de gadolinio va a ejercer el papel de qubit, un espín de computación cuántica. El erbio sin embargo, va a ejercer el papel de una ancilla, un espín para corrección de errores. El átomo de gadolinio presenta un espín 7/2, siendo en realidad un qudit, un sistema capaz de implementar d qubits (un espín 7/2 implica la generación de ocho niveles energéticos, de forma que el gadolinio da lugar a un sistema de tres qubits). El átomo de erbio por otro lado, tiene un espín muy complejo que sin embargo, va a funcionar como un espín efectivo 1/2. Esto es debido a que los lantánidos poseen una propiedad muy especial, presentan un doblete fundamental muy aislado del resto de niveles, de forma que podemos trabajar con ellos ignorando el resto de niveles.
Para manipular y comprender dicho sistema planteo un modelo efectivo que describa el comportamiento del sistema. Este modelo está basado en un Hamiltoniano en el que figuran tres tipos de contribuciones: la contribución de la anisotropía magnética, que introduce un desdoblamiento de los niveles de los átomos incluso a campo cero, la contribución del efecto Zeeman, y finalmente una interacción entre ambos átomos en términos del Hamiltoniano de Heisenberg. En este modelo no contribuyen los términos del campo cristalino, al ser despreciable la interacción del mismo con las tierras raras, y tampoco entra en juego el efecto del término espín órbita al poseer el gadolinio un momento angular nulo y debido al comportamiento del erbio como espín efectivo, que solo contribuye con la presencia del campo magnético. Dicho Hamiltoniano viene descrito en función del gadolinio y del erbio en la ecuación 6.

$$H = H_{Gd} \otimes I_2 + I_8 \otimes H_{Er} + J\vec{S} \cdot \hat{g_n} \cdot \vec{\sigma} \tag{6}$$

Como puede verse, el Hamiltoniano queda resumido en el término del Gd, del Er, y de la interacción entre ambos. Dicha interacción incorpora un tensor $\hat{g_n}$ cuya elección permite aplicar un modelo de Heisenberg o de Ising. La elección final ha sido usar un modelo de Heisenberg, por tanto $\hat{g_n}$ es el operador identidad. Las contribuciones correspondientes a dichos átomos vienen descritas en las ecuaciones 7 y 8.

El Hamiltoniano del gadolinio está formado como puede verse en 7 por tres términos. Sus dos primeros términos representan la anisotropía magnética y su último término es el correspondiente al efecto Zeeman.

$$H_{Gd} = D(S_z^2 - \frac{s(s+1)}{3}) + E(S_x^2 - S_y^2) + g_s \mu_B \vec{H} \cdot \vec{S} \tag{7}$$

El Hamiltoniano del erbio no describe todas las contribuciones del átomo como he comentado anteriormente, describe únicamente un doblete muy aislado de los demás niveles. El Hamiltoniano que representa este comportamiento está formado únicamente como puede verse en 8 por el término Zeeman que genera el desdoblamiento de dichos niveles, dado que en ausencia de campo dicho doblete no existe y no hay contribución alguna.

$$H_{Er} = \mu_B \vec{H} \hat{g} \vec{\sigma} \tag{8}$$

Aunque pueda parecer redundante, resulta muy práctico reescribir el Hamiltoniano completo con todos los términos introducidos para poder consultarlo más adelante (ecuación9).

$$H = D(S_z^2 - \frac{s(s+1)}{3}) + E(S_x^2 - S_y^2) + g_s \mu_B \vec{H} \cdot \vec{S} + \mu_B \vec{H} \hat{g} \vec{\sigma} + J \vec{S} \cdot \vec{\sigma} \tag{9}$$

Es importante notar que $\vec{S}$ corresponde al espín del gadolinio, mientras que $\vec{\sigma}$ corresponde al vector de matrices de Pauli, procedentes del espín efectivo del erbio. Además puede observarse que el erbio posee un factor g anisótropo, de forma que viene descrito por un tensor $\hat{g}$, mientras que el gadolinio tiene un factor g isótropo que he expresado como una constante g_s. El sistema requiere entonces de siete parámetros fundamentales que deben ser determinados mediante experimento, las constantes del Hamiltoniano en ausencia de campo, D y E, las tres componentes de $\hat{g}$ correspondientes al erbio, el factor g_s del gadolinio y la constante que rige la interacción entre ambos átomos, J. Para su determinación, es necesario hacer uso de la técnica de EPR. A pesar de que todos los parámetros nombrados dependen de la molécula, D, E, $\hat{g}$, g_s son parámetros que pueden intuirse aproximadamente (orden de magnitud) de la bibliografía del gadolinio y del erbio. No obstante el parámetro J depende completamente de la molécula y no existe bibliografía sobre dicho parámetro. Esto es un problema dado que los parámetros son obtenidos de simulaciónes a partir de los experimentos de EPR, y las posibles combinaciones de parámetros a simular son muy numerosas. Afortunadamente hay una forma de estimar previamente el orden de magnitud de J, la aproximación dipolar magnética. En las tierras raras, es asumible que la interacción de canje tenga aproximadamente la misma magnitud que la interacción dipolar magnética, realizando este cálculo, se obtiene una aproximación muy valiosa al valor de J.

La interacción entre dos dipolos magnéticos viene dada en primer orden por la expresión (10). Para tener en cuenta únicamente este término se supone que los espines y la distancia entre átomos r_{12} son ortogonales. Los resultados de la ecuación (10) implican que en estas circunstancias la constante de canje viene dada por la ecuación (11).

$$E = \frac{\mu_0 \mu_1 \mu_2}{4\pi r_{12}^3} = \frac{\mu_0 g_1 g_2 \mu_B^2}{4\pi r_{12}^3} S_1 S_2 \tag{10}$$

$$J = \frac{\mu_0 g_1 g_2 \mu_B^2}{4\pi r_{12}^3} \tag{11}$$

Para obtener J en MHz analizo las unidades de μ_B y μ_0, dado que los factores de Landé son adimensionales.

$$[\mu_B]^2 = [J/T]^2 = [\frac{Kg \cdot m^2 \cdot s^{-2}}{Kg \cdot s^{-2} \cdot A^{-1}}]^2 = [m^4 \cdot A^2] \tag{12}$$

Con las ecuaciones (12) y (16), J está en las unidades de la ecuación (13):

$$[J] = [N \cdot m] \tag{13}$$

para tener unidades de frecuencia divido por h. Por tanto J se obtiene mediante la ecuación (14)

$$J = \frac{\mu_0 g_1 g_2 \mu_B^2}{4\pi_0 r_{12}^3 h} \approx 215,53 \cdot g_1 \cdot g_2 MHz \tag{14}$$

$$\mu_{\mathrm{B}} = 9,274 \times 10^{-24} J/T \tag{15}$$

$$\mu_0 = 4\pi \cdot 10^{-7} N A^{-2} \tag{16}$$

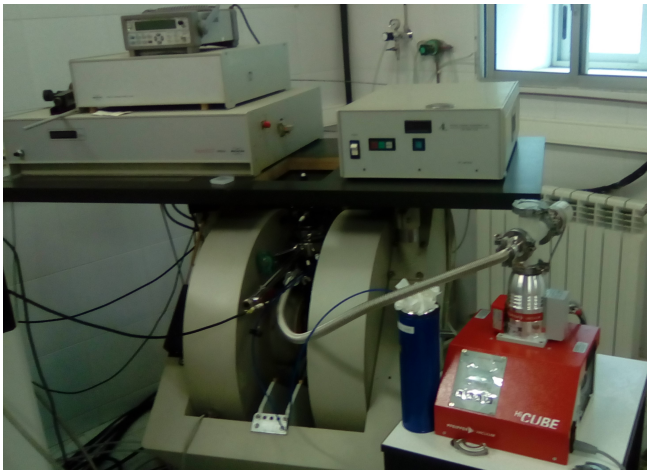
$$h = 6,626 \cdot 10^{-34} J \cdot s \tag{17}$$

Donde se ha tenido en cuenta que la distancia entre gadolinio y erbio es aproximadamente $3,851 \cdot 10^{-10} m$, dato que ha sido obtenido, analizando la estructura molecular (figura 10) con el programa Mercury. Como se puede ver en la ecuación 14, J es proporcional a g_1 (expresado como g_s en el Hamiltoniano del sistema) y g_2 (el factor del erbio, que probablemente corresponda a la dirección de máxima anisotropía), los factores de Landé de ambos átomos. La obtención de ambos parámetros con la técnica de EPR permitirá obtener un valor aproximado de J.

7. EPR: Electron Paramagnetic Resonance

EPR es una técnica experimental de espectroscopía que permite reconocer elementos que tienen sus electrones desapareados mediante la aplicación de un campo magnético. En concreto las tierras raras, como las usadas en este trabajo, son elementos 4f, con electrones desapareados en ellas, por lo que dicha técnica es realmente útil. Se trata de una técnica muy parecida al NMR (resonancia nuclear magnética) dado que ambas trabajan con la interacción entre radiación electromagnética y momentos magnéticos. La principal diferencia es que en el caso de EPR, los momentos son los correspondientes a los electrones y no a los espines nucleares como en NMR. Cada electrón posee un momento magnético intrínseco que proviene de su espín, en muchos casos, los electrones se encuentran apareados, siendo el momento global nulo. Es por esto que solo las especies con electrones desapareados pueden interaccionar con el campo magnético y dar información sobre el espectro de EPR. A continuación pueden verse unas imágenes del sistema de medición de EPR de la Universidad de Zaragoza. A la derecha se encuentra el contenedor donde se sumergen las muestras en helio líquido y a la izquierda el sistema formado por fuente, detector y electrónica.

(a) Contenedor de muestras



(b) Equipo de medida

El proceso experimental consiste entonces en introducir una muestra en un contenedor con helio líquido (el experimento de EPR en este caso es a bajas temperaturas). La muestra se encuentra en una cavidad donde se aplica un campo magnético variable. A continuación se inducen transiciones de electrones entre niveles, emitiendo fotones de una determinada energía en el rango de las microondas. Aplicando un campo magnético variable en la cavidad, se hace un barrido y cuando las energías de transición coinciden con la de los fotones, la muestra absorbe dichos fotones si la transición es permitida. Con un detector pueden medirse estos procesos de absorción y con ello las transiciones permitidas. En concreto en los datos obtenidos se ha medido la primera derivada de la respuesta de la muestra al capo magnético. El sistema experimental puede verse con más detalle en la figura 11.

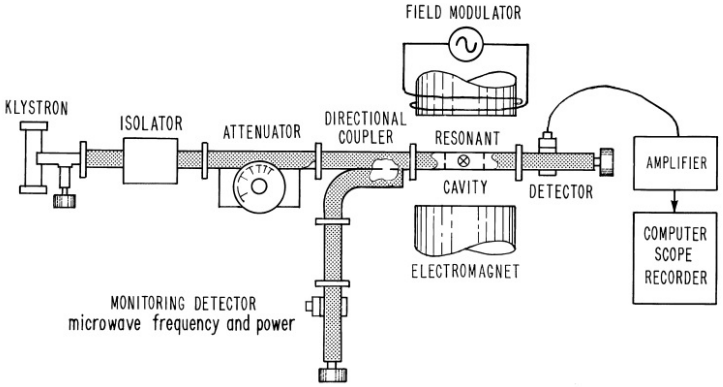


Figura 11: Esquema de un espectrómetro de EPR. Puede verse como fuente de microondas un klystron, un tubo de vacio conocido por sus características de bajo ruido. Por otro lado existe una cavidad donde se aplica el campo magnético, que esta conectada a un detector y también puede estarlo a un sistema de monitorización. La figura ha sido obtenida de la referencia [10]

Las medidas de EPR proporcionan un espectro en energías característico de los elementos o moléculas, que permite no solamente identificarlos, sino también deducir importantes parámetros que definen dicho espectro de energías, en este caso, los parámetros D, E, $\hat{g}$, g_s y J. Para poder obtener dichos parámetros es necesario hacer simulaciones mediante el programa EasySpin en lenguaje Matlab. Dicho programa incorpora la estructura del Hamiltoniano de espín, introduciendo parámetros podemos seleccionar los términos del Hamiltoniano que queremos utilizar para simular posteriormente con la herramienta Pepper. La estructura de dicho Hamiltoniano y de los parámetros a introducir viene explicitamente en la referencia [4]. Probando diversas combinaciones de parámetros y analizando las curvas obtenidas respecto a las curvas experimentales, se puede ir mejorando estos parámetros hasta encontrar los que mejor se ajustan a la curva experimental. Dado que el sistema de GdEr es muy complejo, su cálculo computacional no es trivial, y es conveniente obtener previamente, parámetros aproximados del Gd y del Er aislados. Por este motivo, las medidas de EPR se realizan también para ambos átomos haciendo uso de disoluciones congeladas de GdLu y LaEr, donde el lutecio y lantano no participan al no tener comportamiento magnético. Presento a continuación el código en matlab que permite obtener las curvas de experimento y simulación

Listing 1: Código general de las simulaciones

```
clc; clear; clf
% Lectura, normalización y representación de los datos experimentales
[X,Y,paraE]=eprload('C:\Users\Dorye\Documents\MATLAB\datosepr\dGE_001.DTA');
Be=X/10;
Ymax=max(Y);
Ymin=min(Y);
rango=Ymax-Ymin;
Y=Y/rango;
Ye=Y;
% Representación de la curva experimental
plot(Be,Ye,'r');hold on
% Pasamos los parámetros de la medida del fichero a las variables de la simulación
Exp.mwFreq=paraE.MWFQ/1e9;
Exp.Range=[paraE.XMIN/10 paraE.XMIN/10+paraE.XWID/10];
Exp.nPoints=paraE.XPTS;
Exp.Harmonic=paraE.Harmonic;

% Definimos los parámetros del Hamiltoniano (se introducen para cada caso a analizar)
Sys.S=
% Espín simulado
Sys.g=
% Factores de Landé
Sys.D=
% Vector de parámetros D y E
Sys.J=
% Constante de interacción
Sys.DStrain=
% Anchura general de los picos
```

```
Sys.gStrain=
%Anchura individual de los picos
Sys.HStrain = %Campo residual
Sys.lwpp=
%Anchura del ajuste y tipo (Gaussiano o Lorentziano)

%Realizamos la simulación del espectro y normalizamos la intensidad
simulada como se ha realizado con los datos experimentales
Opt.Method='matrix';
[Bc,Yc]=pepper(Sys,Exp,Opt);
Ycmax=max(Yc);
Ycmin=min(Yc);
rangc=Ycmax-Ycmin;
Yc=Yc/rangc;

%Representamos la simulación junto con los datos experimentales
plot(Bc,Yc,'b')
```

Las simulaciones con EasySpin para el erbio y gadolinio pueden verse en las figuras 12 y 13. Previamente a cada figura se encuentran los datos usados en la simulación, que si se insertan en el código anterior, en el apartado de parámetros del Hamiltoniano permiten reproducir las simulaciones.

Listing 2: Parámetros de simulación del LaEr

```
Sys.S=1/2;
Sys.g=[1.8,3.4,12];
Sys.gStrain=[1 2 3];
Sys.HStrain = [0 0 0];
Sys.lwpp=[0 11];
```

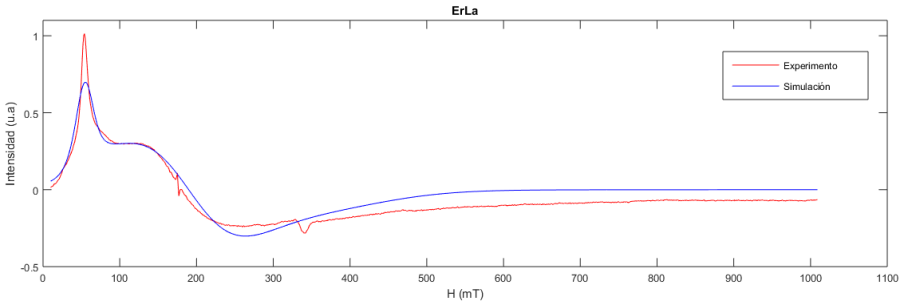


Figura 12: Espectro de EPR del LaEr, puede verse intensidad en unidades arbitrarias frente a intensidad de campo magnético. El LaEr presenta un pico cuyo ajuste es crucial para determinar con precisión $\hat{g}$.

Listing 3: Parámetros de simulación del GdLu

```
Sys.S=7/2;
Sys.g=2;
Sys.D=[2080,600];
Sys.HStrain=[0  0  0];
Sys.DStrain=0.8*[2080,  600];
Sys.lwpp=[15  0];
```

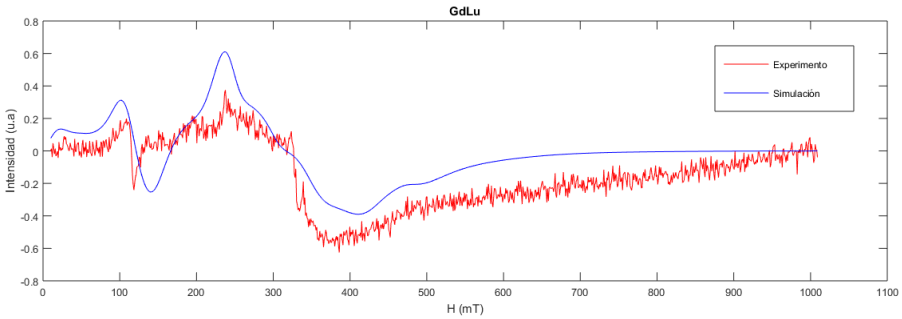


Figura 13: Espectro de EPR del GdLu, de nuevo la intensidad viene dada en unidades arbitrarias. El GdLu posee un espectro de EPR especialmente ruidoso, su ajuste es complicado, siendo posible reproducir su estructura mediante las simulaciones, pero con una precisión muy limitada.

Con el ajuste del LaEr se pueden obtener los valores del factor $\hat{g}$ anisótropo del erbio, con el ajuste del GdLu se puede obtener el valor g_s isótropo del gadolinio, además de los parámetros D y E del Hamiltoniano. Los resultados obtenidos se presentan en las ecuaciones 18, 19, 20 y 21.

$$\hat{g} = (1{,}8, 3{,}4, 12) \tag{18}$$

$$g_s = 2{,}0 \tag{19}$$

$$D = 2080MHz \tag{20}$$

$$E = 600MHz \tag{21}$$

Con los datos obtenidos solo queda obtener J. Como se comentó al final del apartado anterior, la ecuación 14 dada por la aproximación dipolar magnética permite encontrar un valor aproximado de J. Si se realiza el cálculo, sustituyendo $g_1 = 2$ como el factor g del gadolinio y $g_2 = 12$ como el valor del erbio correspondiente a la dirección de máxima anistropía (el erbio es anisótropo y tiene tres valores de g) se obtiene una aproximación del valor de J recogido en la ecuación 22

$$J_{dipolar} \approx 5172MHz \tag{22}$$

Obtenidos estos parámetros y la estimación del orden de J se realiza la simulación para la molécula de GdEr, que puede consultarse en la figura 14 junto con los parámetros de simulación.

Listing 4: Parámetros de simulación del GdEr

```
Sys.S=[7/2,1/2];
```

```
Sys.g=[1.99  1.99  1.99;1.8  3.4  12];
Sys.D=[2080  600;0  0];
Sys.J=4000;
Sys.DStrain=0.3*[2080  600;0  0];
Sys.gStrain=[1  3  0;1  3  1];
Sys.HStrain = [100  100  0];
Sys.lwpp=[25  0];
```

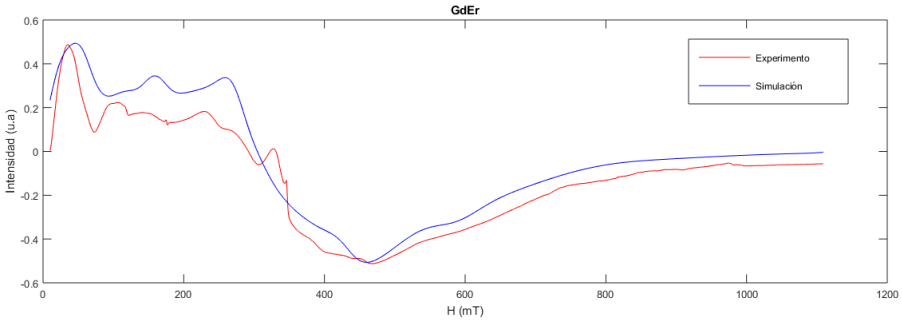


Figura 14: Espectro de EPR del GdEr, aunque el espectro experimental es más sencillo que el del gadolinio, su simulación es la más compleja, además al depender de los parámetros del gadolinio, la precisión es limitada. A pesar de dichas limitaciones la estructura del espectro se reproduce correctamente.

La simulación permite obtener el parámetro J que rige la interacción entre gadolinio y erbio, cuyo valor puede consultarse en la ecuación 23. Tal y como se había dicho, la aproximación dipolar magnética ha sido una estimación correcta y muy necesaria del orden de magnitud, pero no suficiente para conocer el valor de J con precisión. Para conocer el valor de J ha sido necesario acudir a la técnica de EPR y a simulaciones que ajusten dicho parámetro.

$$J = 4000MHz \tag{23}$$

Como puede verse, las simulaciones ajustan el comportamiento y la estructura de las moléculas sin aportar un ajuste ideal de la curva experimental, lo que resulta especialmente evidente en el caso del gadolinio y de la molécula de GdEr. Es importante en este aspecto, tener en cuenta que en las simulaciones de EPR el objetivo es obtener una predicción correcta de la estructura y con ello del comportamiento del sistema y no obtener un ajuste preciso de la magnitud de los picos y zonas secundarias de los espectros. Con dichos valores, se tiene pleno conocimiento de los parámetros del Hamiltoniano 9 y con ello se puede pasar a la simulación del Hamiltoniano, para lo que he hecho uso del lenguaje Python y de la librería Qutip como he comentado anteriormente.

8. Simulación del Hamiltoniano

Con los parámetros obtenidos anteriormente, tengo pleno conocimiento del Hamiltoniano de espín. Esto me permite simular dicho Hamiltoniano (ecuación 9) para obtener una descripción

completa del sistema y proceder a implementar el método de corrección de errores. Con este fin, comienzo simulando el Hamiltoniano del gadolinio, su comportamiento puede verse en la figura 15.

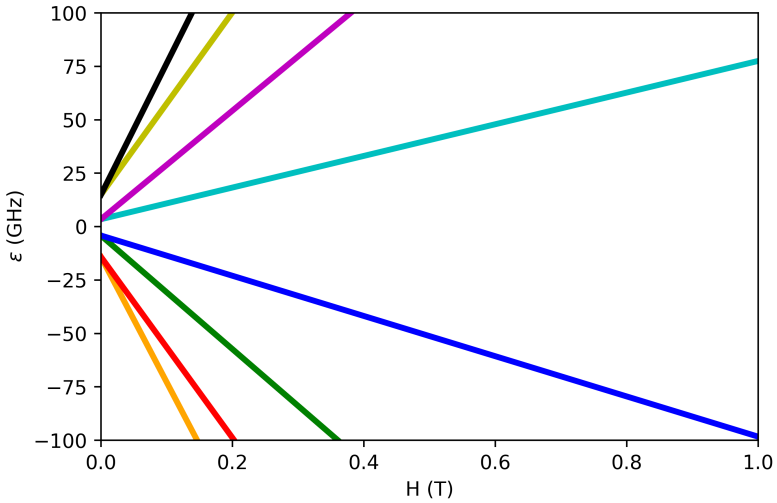


Figura 15: Simulación de los niveles de energía magnética de la molécula de GdLu obtenidos a partir de la diagonalización numérica de su Hamiltoniano de espín . Puede verse una degeneración por pares inicial que se rompe con el efecto Zeeman, dando lugar a ocho niveles de energía correspondientes a las terceras componentes de espín. El orden de los niveles se corresponde con el de las terceras componentes, siendo la mayor componente la asociada a la mayor energía.

Los dos primeros términos de su Hamiltoniano (ecuación 7) dan lugar a ocho niveles de energía degenerados dos a dos. Esto es debido a que el primer término es una matríz S_z con una constante y el segundo término puede reescribirse como suma de los operadores escalera al cuadrado, como se muestra en la relación 24 que mantiene una degeneración en las terceras componentes del espín de igual módulo.

$$D\left(\hat{S}_z^2 - \frac{S(S+1)}{3}\right) + E\left(\hat{S}_x^2 - \hat{S}_y^2\right) = D\left(\hat{S}_z^2 - \frac{S(S+1)}{3}\right) + E\frac{\left(\hat{S}_+^2 + \hat{S}_-^2\right)}{2} \tag{24}$$

El tércer término, es el término de Zeeman que da lugar a una ruptura de la degeneración generando ocho niveles diferenciados.
Al añadir el término del erbio y el acoplo expresado por el Hamiltoniano de Heisenberg, se obtiene el Hamiltoniano completo. Con el objetivo de estudiar el comportamiento del sistema en distintas situaciones, muestro en primer lugar un análisis previo a la obtención de los resultados experimentales presentados. En dicho análisis, estudio el sistema bajo unas condiciones de $\hat{g} = 2$ en la dirección del campo aplicado, en las cuales la contribución del erbio es similar a la del gadolinio. En ausencia de acoplo (término de Heisenberg), se tiene como es esperable, un comportamiento que se rige por el desdoblamiento Zeeman en las terceras componentes del

espín, no obstante, se mantiene cierta degeneración debido a la ausencia de este término. Al añadir el acoplo entre gadolinio y erbio, la repulsión entre electrones es tenida en cuenta, y se produce un desdoblamiento completo en los 16 niveles. Dicho comportamiento viene presentado en la figura 16

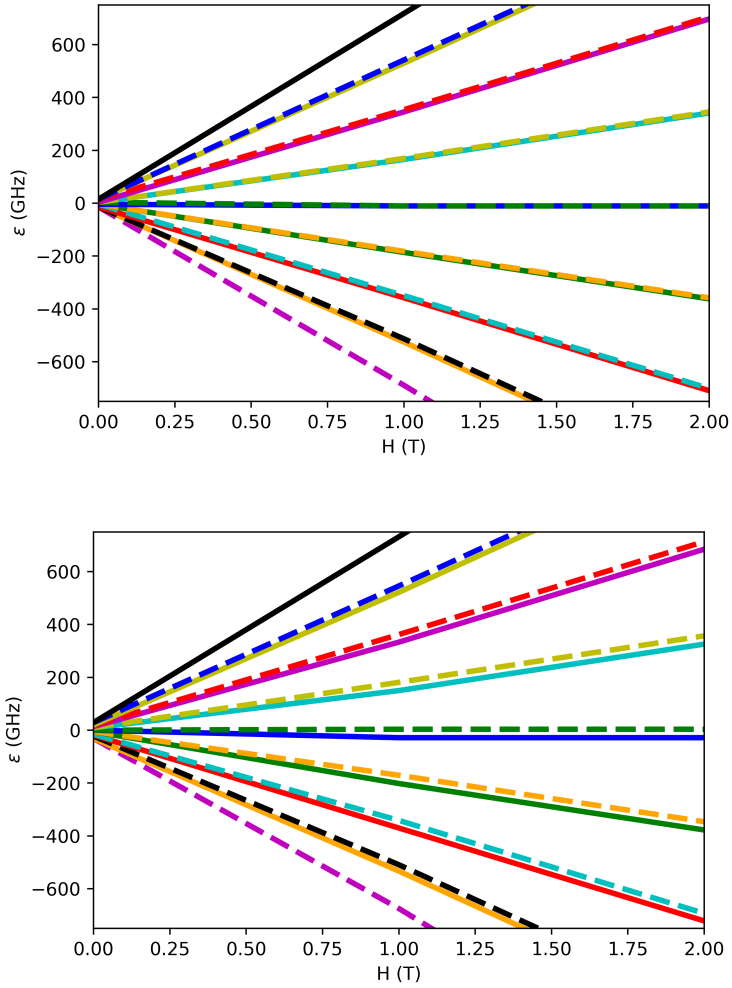


Figura 16: En primer lugar puede verse el comportamiento simulado del Hamiltoniano sin interacción. En él se produce un desdoblamiento de las terceras componentes, pero persiste degeneración debida a la ausencia de dicha interacción, apareciendo únicamente nueve niveles. Posteriormente se muestra el comportamiento del Hamiltoniano con interacción, donde la repulsión entre electrones genera un desdoblamiento completo en los 16 niveles del sistema, rompiéndose toda degeneración. Las lineas discontinuas se refieren a los estados con erbio $|\downarrow\rangle$. El análisis se realiza para $\hat{g} = 2$ en la dirección de campo aplicado. En el código de colores los niveles representados con lineas continuas se refieren a los niveles con el erbio $|\uparrow\rangle$, las lineas discontinuas representan a los estados con el erbio $|\downarrow\rangle$.

Para valores de campo muy altos, el efecto Zeeman predomina, de forma que los niveles se distancian conforme el campo aumenta. Esta separación se produce en función de la tercera componente total de espín. En la figura 17 se ilustra este fenómeno para valores más altos de campo.

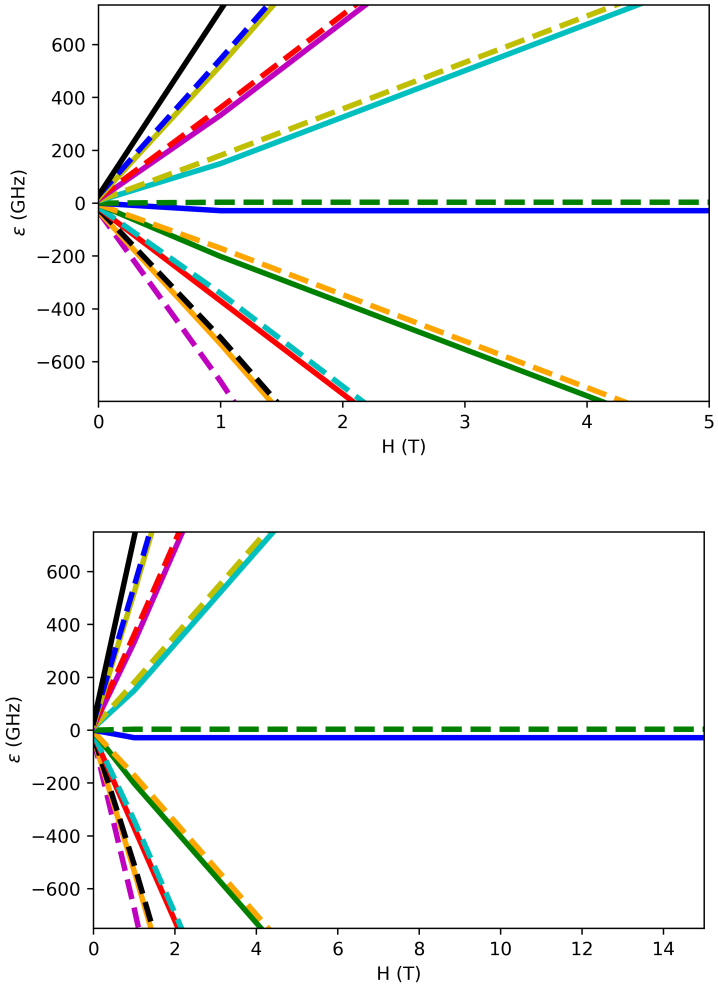


Figura 17: Comportamiento simulado del Hamiltoniano con interacción para campos altos, puede verse como los niveles se separan con el campo, polarizándose el sistema en valores muy positivos o muy negativos. De nuevo las líneas discontinuas marcan los estados con el erbio $|\downarrow\rangle$.

Con el conocimiento de los resultados de EPR, el sistema puede ser descrito con $\hat{g} = 12$ el valor de $\hat{g}$ correspondiente a la dirección de máxima anisotropía. La simulación rectificando este valor da lugar a una contribución mucho mayor del erbio, de forma que el espectro se divide en dos bandas, correspondientes a erbio $|\uparrow\rangle$ o $|\downarrow\rangle$, dicho comportamiento puede verse en la figura 18, que muestra los resultados finales del sistema estudiado.

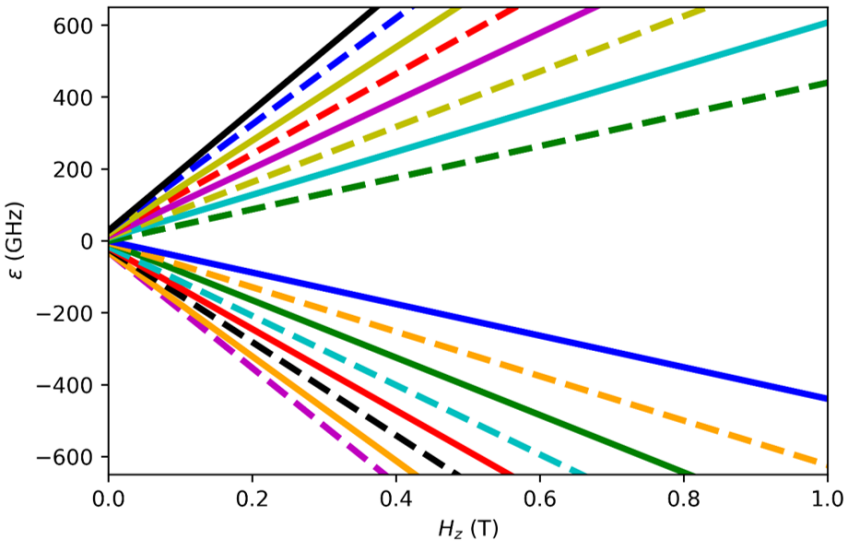


Figura 18: Espectro del sistema basado en los valores obtenidos en laboratorio. Puede verse en la franja inferior los estados con tercera componente del erbio $|\uparrow\rangle$ y en la franja superior los correspondientes a valores del erbio $|\downarrow\rangle$. La tercera componente del gadolinio se ordena de forma ascendente, desde su mayor valor hasta el inferior en cada franja.

Una vez simulado el sistema y comprendido el comportamiento del Hamiltoniano y de sus autovalores, se puede hacer uso de este conocimiento para implementar el método de corrección de errores.

9. Corrección de errores

El fundamento de la corrección de errores en este trabajo, es como he comentado anteriormente, evitar el colapso de la función de onda. El gadolinio forma un qubit, transformando sus estados mediante oscilaciones de Rabi se generan operaciones, pero en caso de que exista un error, en principio solo puede detectarse midiendo sobre el qubit, destruyendo de esta forma el estado del mismo. El sistema obtenido en la molécula de GdEr da lugar a 16 niveles de energía, para poder etiquetar los estados correspondientes a estos niveles de energía, es muy cómodo y funcional usar la base de S^2,S_z. Esto permite referirse a los estados como un ket que hace referencia a las terceras componentes de espín de los átomos. Por ejemplo, el ket $\left|\frac{7}{2}\right\rangle\left|\frac{1}{2}\right\rangle$ implica que el gadolinio tiene una tercera componente de espín $\frac{7}{2}$ y el erbio $\frac{1}{2}$.
El funcionamiento de la corrección de errores está basado en el erbio, la ancilla. Se comienza con la elección de dos estados para implementar un qubit, en las ecuaciones 25 y 26 se muestra un ejemplo de elección (la determinacón de los estados óptimos de trabajo para el sistema molecular

se expondrá en los resultados finales).

$$|0\rangle = \left|\frac{+7}{2}\right\rangle \left|\frac{1}{2}\right\rangle \tag{25}$$

$$|1\rangle = \left|\frac{-7}{2}\right\rangle \left|\frac{1}{2}\right\rangle \tag{26}$$

El estado del qubit vendrá definido por una superposición de ambos estados como se ve en la ecuación 27.

$$|\Psi\rangle = \alpha \left|\frac{+7}{2}\right\rangle \left|\frac{1}{2}\right\rangle + \beta \left|\frac{-7}{2}\right\rangle \left|\frac{1}{2}\right\rangle \tag{27}$$

En dichas condiciones puede ocurrir un error que transforme los estados de forma no deseada como por ejemplo se muestra en la ecuación 28.

$$|\Psi\rangle = \alpha \left|\frac{+5}{2}\right\rangle \left|\frac{1}{2}\right\rangle + \beta \left|\frac{-5}{2}\right\rangle \left|\frac{1}{2}\right\rangle \tag{28}$$

Para poder detectar si existe un error, se aplican dos pulsos π que inviertan el espín del erbio como se muestra en la figura 19, con el objetivo de llevar al sistema del estado de la ecuación 27 al estado de la ecuación 29.

$$|\Psi\rangle = \alpha \left|\frac{+7}{2}\right\rangle \left|\frac{-1}{2}\right\rangle + \beta \left|\frac{-7}{2}\right\rangle \left|\frac{-1}{2}\right\rangle \tag{29}$$

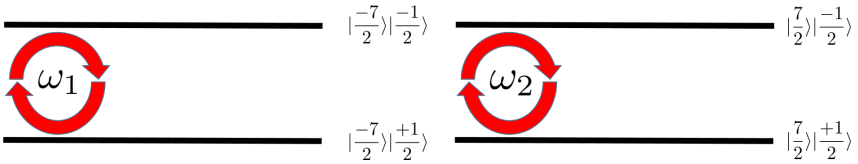
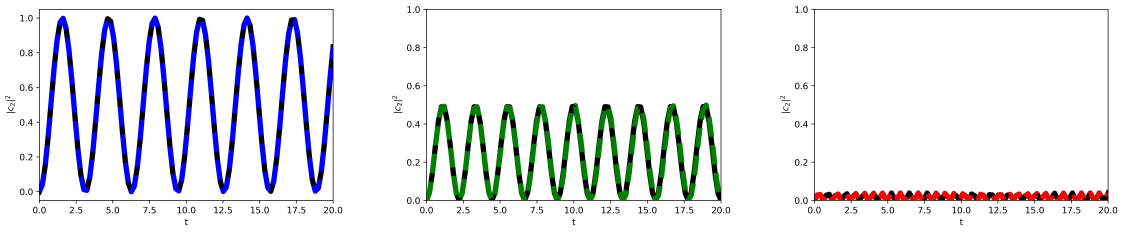


Figura 19: Creación del estado señalado en la ecuación 29. Los dos pulsos π invierten el espín de la ancilla en los estados $|0\rangle$ y $|1\rangle$. Dichos pulsos no tienen efecto en el estado erróneo de la ecuación 28 dado que la frecuencia de resonancia es distinta, evitándose la propagación del error.

El estado expresado en la ecuación 29 se conoce como chivato, dado que solo si no existe error, el sistema puede evolucionar a dicho estado. Para ilustrar este punto he realizado una simulación de las oscilaciones de Rabi producidas dentro y fuera de resonancia, los resultados se muestran en la figura 9. En el eje vertical está representado $|c_2(t)|^2$ que es la probabilidad en la función de ondas de la ecuación 30 que representa la evolución de la ancilla en dicho proceso.

$$|\Psi_{Er}\rangle = c_1(t) \left|\frac{1}{2}\right\rangle + c_2(t) \left|\frac{-1}{2}\right\rangle \tag{30}$$



(a) Resonancia (b) Fuera de Resonancia (c) Muy lejos de Resonancia

En dicha imagen se observa, como las oscilaciones de Rabi decaen fuertemente fuera de resonancia. De esta forma, ningun pulso cuya frecuencia no coincida con la resonancia de una transición hará evolucionar los estados. Esto tiene como consecuencia que cuando se genere un error, este no evolucione y quede aislado. Explicado el aislamiento del error queda explicar el funcionamiento de la detección, dicho proceso viene representado en la figura 20.

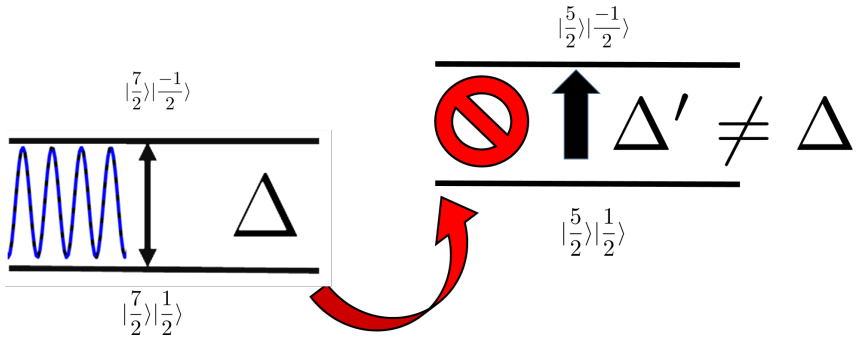


Figura 20: Proceso de aislamiento y detección de errores para el caso partícular del estado $|0\rangle$ señalado en la ecuación 25. El estado inicial, señalado a la izquierda puede sufrir un error que le lleve al estado de la derecha, señalado por la flecha roja. Para detectar el error se genera un pulso π de amplitud Δ que genera las oscilaciones de Rabi dibujadas en azul. En caso de no existir error el sistema evolucionaría y la ancilla quedaría con un estado $|\downarrow\rangle$ pero en caso de error, el estado erróneo de la derecha, no evoluciona, porque su transición requiere de una amplitud Δ' que no es igual a la aplicada. Posteriormente se mide la ancilla y se comprueba que esta no ha invertido su espín, de forma que se detecta el error. Dicho error debe ser identificado mediante la aplicación de los pulsos correspondientes a las transiciones adyacentes, lo que permite identificar el error concreto que se ha producido y posteriormente corregirlo con los pulsos adecuados.

Gracias a este sistema además de evitar la propagación del error, podemos detectarlo midiendo la ancilla sin colapsar la función de onda del gadolinio. Simplemente midiendo la ancilla, se detecta que esta no ha cambiado su estado invirtiendo el espín, como debía ocurrir en el estado final que deseabamos. La ancilla no ha cambiado de estado, de forma que se detecta un error que debe ser corregido mediante pulsos.

De este proceso quiero resaltar una serie de detalles. El primero es que el uso de una molécula para la implementación de qubits, como en este caso el GdEr, permite diseñar la estructura de niveles de los qubits. Dicha característica ha permitido obtener un sistema de energías donde, jugando con las transiciones, se ha conseguido implementar un algoritmo de corrección de errores. Por otro lado uno de los detalles fundamentales que ha permitido el proceso, es que la transición no deseada tiene una amplitud distinta a Δ. Esto no es una suposición, es uno de los dos factores que hay que buscar a la hora de seleccionar la transición donde implementar la corrección de errores. Si la amplitud de las transiciones adyacentes fueran iguales o muy parecidas, los errores se propagarían y seríamos incapaces de detectarlos. Es por esto que en la figura 20, la transición no deseada está marcada por una señal de prohibición. Por último, he dejado para este punto la explicación de un factor muy relevante, con la intención de facilitar la comprensión del proceso. El Hamiltoniano del sistema no es diagonal en la base S^2, S_z y los autoestados correspondientes a los niveles de energía no pueden etiquetarse de una forma tan sencilla (notesé que no hice uso de la palabra autoestado hasta este punto). Para obtener los autovalores dibujados en las distintas figuras, es necesario diagonalizar la matriz y por tanto llevarla a una base que no es la citada anteriormente. Los autoestados son realmente combinaciones lineales de terceras componentes del espín, que dificultan en gran medida el proceso de corrección de errores. Es por esto que el segundo factor que es necesario tener en cuenta, para elegir la transición correcta donde implementar este método, es escoger aquellos niveles cuyos estados tengan una fidelidad lo más alta posible, esto es, que el producto escalar señalado en la ecuación 31 sea lo más cercano a 1 posible (donde Ψ representa los autoestados correspondientes a los sistemas señalados).

$$Fidelidad = || \langle \Psi_{GdEr} | \Psi_{Gd} \Psi_{Er} \rangle || \tag{31}$$

Si la fidelidad es 1, los autoestados son exactamente los mismos que los estados de la base no diagonal. La fidelidad depende del campo magnético aplicado, dado que cuanto mayor es este, más despreciable es el término de interacción del Hamiltoniano. Es por esto que escoger la transición correcta implica también, seleccionar la región de campo aplicado para trabajar.
Puedo concluir el trabajo mediante un análisis de la anisotropía y de la fidelidad del sistema proporcionado por la molécula. Dicho análisis, presentado en las imágenes 21 y 22 me permite obtener como resultados finales, los estados óptimos de trabajo para implementar una corrección de errores en laboratorio, además de las condiciones de trabajo necesarias.

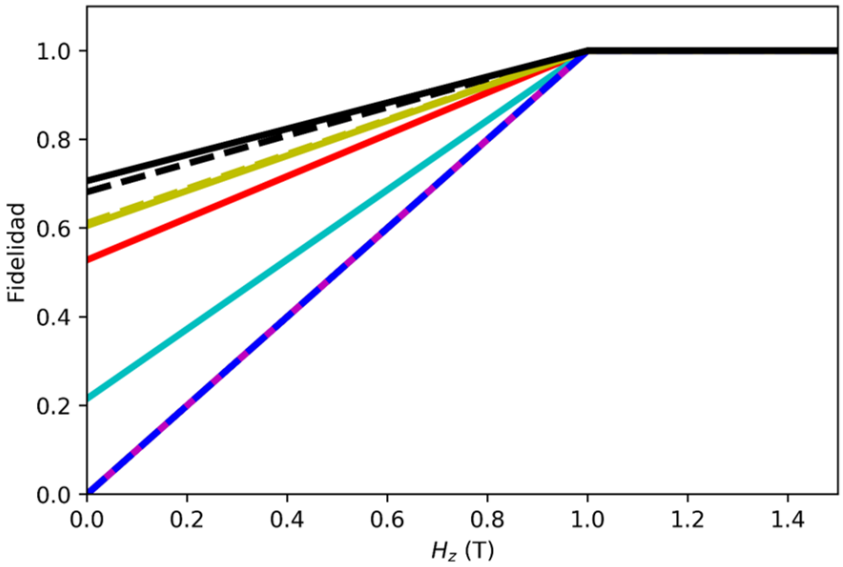


Figura 21: Fidelidades de los estados correspondientes a los 16 niveles de energía. Puede observarse como conforme aumenta el campo el acoplo se vuelve despreciable, téndiendo los distintos estados a una fidelidad plena, siendo en este punto completamente factorizables.

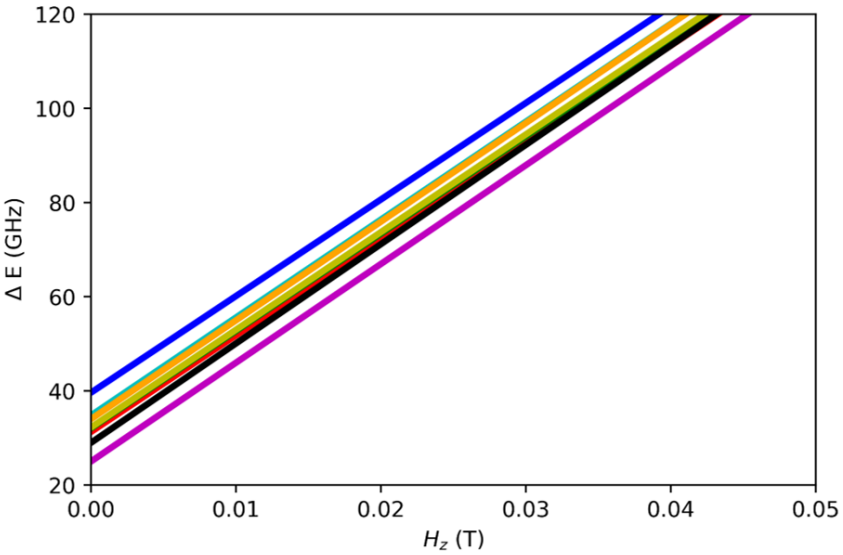


Figura 22: Diferencias de energía de las transiciones que invierten la ancilla. Tienden a ser muy parecidas, salvo las lineas morada y azul que se convierten en candidatas para la implementación del método de correción de errores. Las transiciones correspondientes vienen indicadas en las siguientes figuras. Puede observarse que como era de esperar, es necesario trabajar en rango de campo bajo para mantener la anisotropía del sistema evitando que el efecto Zeeman domine y haga todas las diferencias de energías iguales.

Los resultados obtenidos pueden verse en las figuras 23 y 24, donde pueden verse los estados y las transiciones seleccionadas además de las condiciones adecuadas de campo magnético.

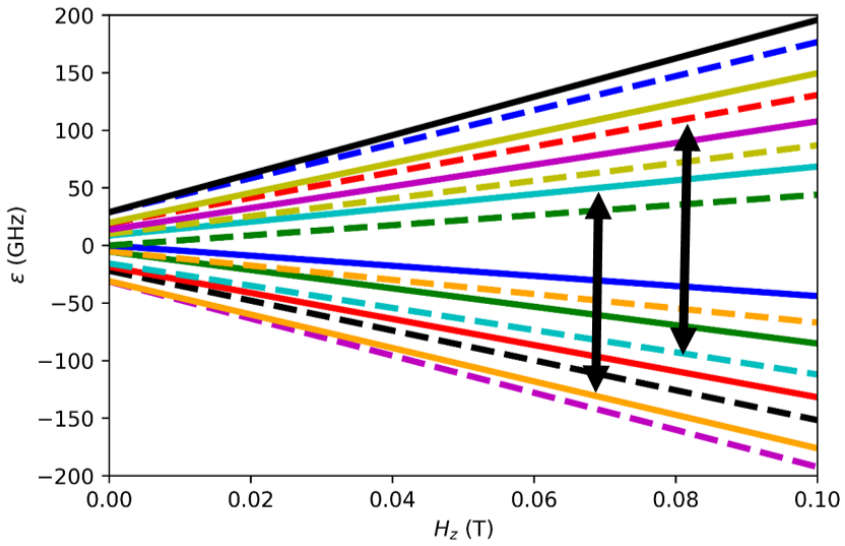


Figura 23: Transiciones candidatas para la implementación del método de corrección de errores. Su selección es debida a que las diferencias de energía de estas transiciones son realmente distintas de las del resto de transiciones.

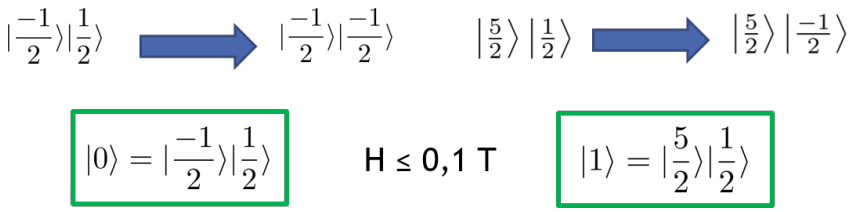


Figura 24: Selección del qubit de trabajo en base a las transiciones elegidas. El qubit formado por estos estados es el candidato óptimo para la implementación en laboratorio del método de corrección de errores. También se indica el rango de campo aplicado que permite la correcta implementación.

En esta sección, se ha explicado el método de corrección de errores aplicado a un qubit molecular perteneciente a una molécula de GdEr. Además, se han podido seleccionar adecuadamente las transiciones donde implementar el método, y las condiciones de trabajo. Llegados a este punto puedo hacer una conclusión final del trabajo.

10. Conclusiones

Este trabajo es un estudio téorico necesario para poder llevar acabo una implementación del método de corrección de errores, en un sistema experimental. La teoría y el trabajo de simula-

ción realizado en este trabajo, permiten tener una comprensión completa del comportamiento de la molécula, lo que resulta necesario para poder implementar el método aquí presentado. Se ha realizado una modelización del sistema, se han encontrado los parámetros que describen al Hamiltoniano del modelo mediante el análisis de datos experimentales. Se ha conseguido una descripción energética del sistema mediante simulaciones del Hamiltoniano, obteniéndose un mapa imprescindible para trabajar con dicha molécula. Por último, los argumentos proporcionados al final, han permitido obtener resultados definitivos, que permiten implementar el método de corrección de errores en un laboratorio.
También es importante recalcar que el método aquí presentado, a pesar de poder aplicarse en errores de fase o de amplitud indistintamente, no permite detectar cualquier error. Un ejemplo claro surge al plantearse que sucede si el error se produce en la ancilla y esta cambia de estado (al medir la ancilla su estado habría cambiado). Este método se limita a los errores que solo cambian los estados en una unidad la tercera componente del espín, y aunque cualquier error, puede describirse como una combinación de errores de este tipo, algorítmos de corrección de errores más complejos son necesarios para poder proteger completamente un qubit.

Referencias

[1] Stephen Barnett. *Quantum information*. Vol. 16. Oxford University Press, 2009.

[2] Juan I Cirac y Peter Zoller. "Quantum computations with cold trapped ions". En: *Physical review letters* 74.20 (1995), pág. 4091.

[3] Thomas Dittrich y col. *Quantum transport and dissipation*. Vol. 3. Wiley-Vch Weinheim, 1998.

[4] *EasySpin, Spin Hamiltonian.* `http://easyspin.org/easyspin/documentation/hamiltonian.html`.

[5] Richard P Feynman. "Simulating physics with computers". En: *International journal of theoretical physics* 21.6-7 (1982), págs. 467-488.

[6] A Gaita-Ariño y col. "Molecular spins for quantum computation". En: *Nature chemistry* 11.4 (2019), pág. 301.

[7] Riaz Hussain y col. "Coherent manipulation of a molecular Ln-based nuclear qudit coupled to an electron qubit". En: *Journal of the American Chemical Society* 140.31 (2018), págs. 9814-9818.

[8] *IBM,Shor algorithm.* `https://quantumexperience.ng.bluemix.net/proxy/tutorial/full-user-guide/004-Quantum_Algorithms/110-Shor's_algorithm.html`,

[9] MD Jenkins y col. "Coherent manipulation of three-qubit states in a molecular single-ion magnet". En: *Physical Review B* 95.6 (2017), pág. 064423.

[10] John Ashley Weil, James R Bolton y John E Wertz. *Electron paramagnetic resonance: elementary theory and practical applications*. 543.42 WEI. 1994.

Anexo

A continuación presento los códigos que he desarrollado que permiten obtener las gráficas. No se encuentran todos los códigos que he usado para los análisis, solo los que al final han dado material que aparece en el trabajo. Además los códigos se encuentran condensados de forma que de cada uno se pueden obtener distintas figuras variando los parámetros.

Análisis del gadolinio

Listing 5: Parámetros de simulación del GdEr

```
#Librerias y definición de parámetros
        import numpy as np
        import pylab as plt
        import cmath
        import math
        from scipy import *
        from qutip import *
        Sx=jmat(7/2,'x')
        Sy=jmat(7/2,'y')
        Sz=jmat(7/2,'z')
        s_x_2=jmat(7/2,'x')*jmat(7/2,'x')
        s_y_2=jmat(7/2,'y')*jmat(7/2,'y')
        s_z_2=jmat(7/2,'z')*jmat(7/2,'z')
        D=2080
        E=600
# C'alculo del Hamiltoniano para distintos valores de campo y obtenci'on de autovalores

for Hz in range(0, 10):
H_Gd=(D*(s_z_2-(7/2)*(1/3)*((7/2)+1))+E*(s_x_2-s_y_2)-175824*(Hz*Sz))/1000
eisystem=H_Gd.eigenstates()
if Hz==0:
X=[Hz];
autoval1=[eisystem[0][0]];
autoval2=[eisystem[0][1]];
autoval3=[eisystem[0][2]];
autoval4=[eisystem[0][3]];
autoval5=[eisystem[0][4]];
autoval6=[eisystem[0][5]];
autoval7=[eisystem[0][6]];
autoval8=[eisystem[0][7]];
else:
X.append(Hz);
autoval1.append(eisystem[0][0]);
autoval2.append(eisystem[0][1]);
autoval3.append(eisystem[0][2]);
autoval4.append(eisystem[0][3]);
autoval5.append(eisystem[0][4]);
```

```
autoval6.append(eisystem[0][5]);
autoval7.append(eisystem[0][6]);
autoval8.append(eisystem[0][7]);

Imagen=1;
fig, ax = plt.subplots()
ax.plot(X,autoval1, 'orange', label='Autovalor 1',linewidth=3);
ax.plot(X,autoval2, 'red', label='Autovalor 2',linewidth=3);
ax.plot(X,autoval3, 'g', label='Autovalor 3',linewidth=3);
ax.plot(X,autoval4, 'b', label='Autovalor 4',linewidth=3);
ax.plot(X,autoval5, 'c', label='Autovalor 5',linewidth=3);
ax.plot(X,autoval6, 'm', label='Autovalor 6',linewidth=3);
ax.plot(X,autoval7, 'y', label='Autovalor 7',linewidth=3);
ax.plot(X,autoval8, 'k', label='Autovalor 8',linewidth=3);

ax.set_xlim(0,1);
ax.set_ylim(-100,100);



ax.set_xlabel(r'H (T)');
ax.set_ylabel(r'$\epsilon$ (GHz)');
strn='Gd_Levels.png'
plt.savefig(strn, format='png', dpi=500)
plt.show(fig)
```

Análisis del GdEr

Listing 6: Parámetros de simulación del GdEr

```
# Librerias y definición de parámetros
        import numpy as np
        import pylab as plt
        import cmath
        import math
        import matplotlib.pyplot as pltm
        from scipy import *
        from qutip import *
        Sx=jmat(7/2,'x')
        Sy=jmat(7/2,'y')
        Sz=jmat(7/2,'z')
        s_z_2=jmat(7/2,'z')*jmat(7/2,'z')
        s_x_2=jmat(7/2,'x')*jmat(7/2,'x')
        s_y_2=jmat(7/2,'y')*jmat(7/2,'y')
        D=2080
        E=600
        J=4000
        gx=1.8
```

```
        gy=3.4
        gz=12
# Calculo del Hamiltoniano completo y representación para distintos valores de campo
for Hz in range(0, 30):
H_Gd=(D*(s_z_2-(7/2)*(1/3)*((7/2)+1))+E*(s_x_2-s_y_2)-175824*(Hz*Sz))
Her=-175824*0.5*12*sigmaz()*Hz
ExtendedEr=tensor(qeye(8),Her)
Hac=J*(tensor(Sx,sigmax())+tensor(Sy,sigmay())+tensor(Sz,sigmaz()))
ExtendedH_Gd=tensor(H_Gd,qeye(2))
ExtendedH_Gd.dims=ExtendedEr.dims
Hac.dims=ExtendedEr.dims
H=(ExtendedH_Gd+ExtendedEr)/1000
eisystemT=H.eigenstates()

if Hz==0:
X=[Hz];
autoval1=[eisystemT[0][0]];
autoval2=[eisystemT[0][1]];
autoval3=[eisystemT[0][2]];
autoval4=[eisystemT[0][3]];
autoval5=[eisystemT[0][4]];
autoval6=[eisystemT[0][5]];
autoval7=[eisystemT[0][6]];
autoval8=[eisystemT[0][7]];
autoval9=[eisystemT[0][8]];
autoval10=[eisystemT[0][9]];
autoval11=[eisystemT[0][10]];
autoval12=[eisystemT[0][11]];
autoval13=[eisystemT[0][12]];
autoval14=[eisystemT[0][13]];
autoval15=[eisystemT[0][14]];
autoval16=[eisystemT[0][15]];
else:
X.append(Hz);
autoval1.append(eisystemT[0][0]);
autoval2.append(eisystemT[0][1]);
autoval3.append(eisystemT[0][2]);
autoval4.append(eisystemT[0][3]);
autoval5.append(eisystemT[0][4]);
autoval6.append(eisystemT[0][5]);
autoval7.append(eisystemT[0][6]);
autoval8.append(eisystemT[0][7]);
autoval9.append(eisystemT[0][8]);
autoval10.append(eisystemT[0][9]);
autoval11.append(eisystemT[0][10]);
autoval12.append(eisystemT[0][11]);
autoval13.append(eisystemT[0][12]);
autoval14.append(eisystemT[0][13]);
autoval15.append(eisystemT[0][14]);
autoval16.append(eisystemT[0][15]);
```

```
Imagen=2;
fig , ax = plt.subplots()
ax.plot(X,autoval1 , 'm--', label='Autovalor 1',linewidth=3);
ax.plot(X,autoval2 , 'orange', label='Autovalor 2',linewidth=3);
ax.plot(X,autoval3 , 'k--', label='Autovalor 3',linewidth=3);
ax.plot(X,autoval4 , 'r', label='Autovalor 4',linewidth=3);
ax.plot(X,autoval5 , 'c--', label='Autovalor 5',linewidth=3);
ax.plot(X,autoval6 , 'g', label='Autovalor 6',linewidth=3);
ax.plot(X,autoval7 , color='orange',linestyle='dashed', label='Autovalor 7',linewidth=3);
ax.plot(X,autoval8 , 'b', label='Autovalor 8',linewidth=3);
ax.plot(X,autoval9 , 'g--', label='Autovalor 9',linewidth=3);
ax.plot(X,autoval10 , 'c', label='Autovalor 10',linewidth=3);
ax.plot(X,autoval11 , 'y--', label='Autovalor 11',linewidth=3);
ax.plot(X,autoval12 , 'm', label='Autovalor 12',linewidth=3);
ax.plot(X,autoval13 , 'r--', label='Autovalor 13',linewidth=3);
ax.plot(X,autoval14 , 'y', label='Autovalor 14',linewidth=3);
ax.plot(X,autoval15 , 'b--', label='Autovalor 15',linewidth=3);
ax.plot(X,autoval16 , 'k', label='Autovalor 16',linewidth=3);

ax.set_xlabel(r'H (T)');
ax.set_ylabel(r'$\epsilon$ (GHz)');
ax.set_xlim(0,2);
ax.set_ylim(-750,750);
strn='C9.png'
plt.savefig(strn, format='png', dpi=500)
plt.show(fig)
```

Listing 7: Oscilaciones de Rabi en la ancilla

```
        import numpy as np
        import pylab as plt
        import cmath
        from scipy import *
        from qutip import *
        Sx=jmat(7/2,'x')
        Sy=jmat(7/2,'y')
        Sz=jmat(7/2,'z')
        Ss=jmat(7/2,'+')
        s_z_2=jmat(7/2,'z')*jmat(7/2,'z')
        s_x_2=jmat(7/2,'x')*jmat(7/2,'x')
        s_y_2=jmat(7/2,'y')*jmat(7/2,'y')
        D=2080
        E=600
        J=4000
        gx=1.8
        gy=3.4
        gz=12

#inicializo variables
```

```
Psi0=basis(2,0)
tlist = linspace(0, 20, 100)
Lam=1                                  #Amplitud Lambda
delta=10
Omega=sqrt(delta*delta+4*Lam*Lam) # Frecuencia de Rabi
om=300                                 # Frecuencia
Delta=om-delta
Omegares=2*Lam                         # Valor de la frecuencia de Rabi en resonancia
Omegap=sqrt(2*2+4*Lam*Lam)             # Valor de la frecuencia de Rabi para delta=2


H0=Delta*sigmaz()/2; #Defino el Hamiltoniano
 (la dependencia con el tiempo tiene una sintaxis especial)
H1 =sigmax()
def H1_coeff(t,arg):
return 2*Lam*cos(om*t)
h_t = [H0,[H1, H1_coeff]]

modQuadC2=sigmax()*sigmap()    # Defino una matriz tal que su valor esperado sea |c2|^2

#Realizo la evolución temporal y calculo el valor esperado de modQuadC2 (delta=10)
medata=mesolve(h_t, Psi0, tlist, [],modQuadC2)

H0=om*sigmaz()/2; #defino el mismo Hamiltoniano pero bajo condición de resonancia
H1 =sigmax()
def H1_coeff(t,arg):
return 2*Lam*cos(om*t)
h_res = [H0,[H1, H1_coeff]]
meres=mesolve(h_res, Psi0, tlist, [],modQuadC2)  #calculo el valor de modQuadC2 en resonancia

H0=98*sigmaz()/2;  #defino el mismo Hamiltoniano pero con delta=2
H1 =sigmax()
def H1_coeff(t,arg):
return 2*Lam*cos(100*t)
h_res = [H0,[H1, H1_coeff]]
minimo=mesolve(h_res, Psi0, tlist, [],modQuadC2) #calculo el valor de modQuadC2 con delta=2

# Genero las imágenes
fig, ax = plt.subplots()
# Represento las soluciones de Rabi para resonancia, delta=2 y delta=10
#ax.plot(tlist,(2*Lam/Omegares)*(2*Lam/Omegares)*sin(Omegares*tlist/2)
*sin(Omegares*tlist/2),'-k',linewidth=6);
ax.plot(tlist,(2*Lam/Omegap)*(2*Lam/Omegap)*sin(Omegap*tlist/2)
*sin(Omegap*tlist/2),'-k',linewidth=6);
#ax.plot(tlist,(2*Lam/Omega)*(2*Lam/Omega)*sin(Omega*tlist/2)
*sin(Omega*tlist/2),'-k', linewidth=6);
# Represento las evoluciones temporales para resonancia, delta=2 y delta=10
#ax.plot(tlist,meres.expect[0],'--b',linewidth=6)
ax.plot(tlist,minimo.expect[0],'--g',linewidth=6)
#ax.plot(tlist,medata.expect[0],'--r',linewidth=6)
```

```
ax.set_xlabel(r't');
ax.set_ylabel(r'$|c_2|^2$');
ax.set_xlim(0,20);
ax.set_ylim(0,1);
ax.legend(loc=1);
plt.savefig('figura.png', format='png', dpi=500)
plt.show(fig)
```

Listing 8: Fidelidades

```
#Librerias y definición de parámetros
import numpy as np
import pylab as plt
import cmath
import math
from scipy import *
from qutip import *
Sx=jmat(7/2,'x')
Sy=jmat(7/2,'y')
Sz=jmat(7/2,'z')
s_x_2=jmat(7/2,'x')*jmat(7/2,'x')
s_y_2=jmat(7/2,'y')*jmat(7/2,'y')
s_z_2=jmat(7/2,'z')*jmat(7/2,'z')
D=2080
E=600
J=4000
V=[9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9]

#Simulación del Hamiltoniano y cálculo de autovectores
for Hz in range(0, 30):
H_Gd=(D*(s_z_2-(7/2)*(1/3)*((7/2)+1))+E*(s_x_2-s_y_2)-175824*(Hz*Sz))
Her=-175824*0.5*12*sigmaz()*Hz
ExtendedEr=tensor(qeye(8),Her)
Hac=J*(tensor(Sx,sigmax())+tensor(Sy,sigmay())+tensor(Sz,sigmaz()))
ExtendedH_Gd=tensor(H_Gd,qeye(2))
ExtendedH_Gd.dims=ExtendedEr.dims
Hac.dims=ExtendedEr.dims
H=(ExtendedH_Gd+ExtendedEr+Hac)/1000
eisystem=H.eigenstates()
autovecH=eisystem[1]
eisystem=H_Gd.eigenstates()
autovecH_Gd=eisystem[1]
eisystem=Her.eigenstates()
autovecHer=eisystem[1]

#Cálculo y representación de fidelidades
V[0]=(autovecH[0].trans()*tensor(autovecH_Gd[0],autovecHer[0])).norm()
V[1]=(autovecH[1].trans()*tensor(autovecH_Gd[1],autovecHer[0])).norm()
```

```
V[2]=(autovecH[2].trans()*tensor(autovecH_Gd[2],autovecHer[0])).norm()
V[3]=(autovecH[3].trans()*tensor(autovecH_Gd[3],autovecHer[0])).norm()
V[4]=(autovecH[4].trans()*tensor(autovecH_Gd[4],autovecHer[0])).norm()
V[5]=(autovecH[5].trans()*tensor(autovecH_Gd[5],autovecHer[0])).norm()
V[6]=(autovecH[6].trans()*tensor(autovecH_Gd[6],autovecHer[0])).norm()
V[7]=(autovecH[7].trans()*tensor(autovecH_Gd[7],autovecHer[0])).norm()
V[8]=(autovecH[8].trans()*tensor(autovecH_Gd[0],autovecHer[1])).norm()
V[9]=(autovecH[9].trans()*tensor(autovecH_Gd[1],autovecHer[1])).norm()
V[10]=(autovecH[10].trans()*tensor(autovecH_Gd[2],autovecHer[1])).norm()
V[11]=(autovecH[11].trans()*tensor(autovecH_Gd[3],autovecHer[1])).norm()
V[12]=(autovecH[12].trans()*tensor(autovecH_Gd[4],autovecHer[1])).norm()
V[13]=(autovecH[13].trans()*tensor(autovecH_Gd[5],autovecHer[1])).norm()
V[14]=(autovecH[14].trans()*tensor(autovecH_Gd[6],autovecHer[1])).norm()
V[15]=(autovecH[15].trans()*tensor(autovecH_Gd[7],autovecHer[1])).norm()
if Hz==0:
X=[Hz];
autoval1=[V[0]];
autoval2=[V[1]];
autoval3=[V[2]];
autoval4=[V[3]];
autoval5=[V[4]];
autoval6=[V[5]];
autoval7=[V[6]];
autoval8=[V[7]];
autoval9=[V[8]];
autoval10=[V[9]];
autoval11=[V[10]];
autoval12=[V[11]];
autoval13=[V[12]];
autoval14=[V[13]];
autoval15=[V[14]];
autoval16=[V[15]];
else:
X.append(Hz);
autoval1.append(V[0]);
autoval2.append(V[1]);
autoval3.append(V[2]);
autoval4.append(V[3]);
autoval5.append(V[4]);
autoval6.append(V[5]);
autoval7.append(V[6]);
autoval8.append(V[7]);
autoval9.append(V[8]);
autoval10.append(V[9]);
autoval11.append(V[10]);
autoval12.append(V[11]);
autoval13.append(V[12]);
autoval14.append(V[13]);
autoval15.append(V[14]);
autoval16.append(V[15]);
```

```
Imagen=0;
fig , ax = plt.subplots()
ax.plot(X,autoval1 , 'm--', label='Autovalor 1',linewidth=3);
ax.plot(X,autoval2 , 'orange', label='Autovalor 2',linewidth=3);
ax.plot(X,autoval3 , 'k--', label='Autovalor 3',linewidth=3);
ax.plot(X,autoval4 , 'r', label='Autovalor 4',linewidth=3);
ax.plot(X,autoval5 , 'c--', label='Autovalor 5',linewidth=3);
ax.plot(X,autoval6 , 'g', label='Autovalor 6',linewidth=3);
ax.plot(X,autoval7 , color='orange',linestyle='dashed', label='Autovalor 7',linewidth=3);
ax.plot(X,autoval8 , 'b', label='Autovalor 8',linewidth=3);
ax.plot(X,autoval9 , 'g--', label='Autovalor 9',linewidth=3);
ax.plot(X,autoval10 , 'c', label='Autovalor 10',linewidth=3);
ax.plot(X,autoval11 , 'y--', label='Autovalor 11',linewidth=3);
ax.plot(X,autoval12 , 'm', label='Autovalor 12',linewidth=3);
ax.plot(X,autoval13 , 'r--', label='Autovalor 13',linewidth=3);
ax.plot(X,autoval14 , 'y', label='Autovalor 14',linewidth=3);
ax.plot(X,autoval15 , 'b--', label='Autovalor 15',linewidth=3);
ax.plot(X,autoval16 , 'k', label='Autovalor 16',linewidth=3);

strn='C4.png'
ax.set_xlim(0,1.5);
ax.set_ylim(0,1.1);
ax.set_xlabel(r'$H_z$ (T)');
ax.set_ylabel(r'Fidelidad');
plt.savefig(strn, format='png',dpi=500)
```

Listing 9: Anisotropía

```
#Librerias y definición de parámetros
import numpy as np
import pylab as plt
import cmath
import math
from scipy import *
from qutip import *
Sx=jmat(7/2,'x')
Sy=jmat(7/2,'y')
Sz=jmat(7/2,'z')
s_x_2=jmat(7/2,'x')*jmat(7/2,'x')
s_y_2=jmat(7/2,'y')*jmat(7/2,'y')
s_z_2=jmat(7/2,'z')*jmat(7/2,'z')
D=2080
E=600
J=4000

#Simulación del Hamiltoniano y obtención de autovalores
for Hz in range(0, 100):
H_Gd=(D*(s_z_2-(7/2)*(1/3)*((7/2)+1))+E*(s_x_2-s_y_2)-175824*(Hz*Sz))
Her=-175824*0.5*12*sigmaz()*Hz
```

```
ExtendedEr=tensor(qeye(8),Her)
Hac=J*(tensor(Sx,sigmax())+tensor(Sy,sigmay())+tensor(Sz,sigmaz()))
ExtendedH_Gd=tensor(H_Gd,qeye(2))
ExtendedH_Gd.dims=ExtendedEr.dims
Hac.dims=ExtendedEr.dims
H=(ExtendedH_Gd+ExtendedEr+Hac)/1000
eisystem=(H).eigenstates()
autovalH=eisystem[0]
#Cálculo de las diferencias de energías y representación
DeltaE=[abs(autovalH[0]-autovalH[8]),abs(autovalH[1]-autovalH[9]),
abs(autovalH[2]-autovalH[10]),abs(autovalH[3]-autovalH[11]),
abs(autovalH[4]-autovalH[12]),abs(autovalH[5]-autovalH[13]),
abs(autovalH[6]-autovalH[14]),abs(autovalH[7]-autovalH[15])]
if Hz==0:
X=[Hz];
autoval1=[DeltaE[0]];
autoval2=[DeltaE[1]];
autoval3=[DeltaE[2]];
autoval4=[DeltaE[3]];
autoval5=[DeltaE[4]];
autoval6=[DeltaE[5]];
autoval7=[DeltaE[6]];
autoval8=[DeltaE[7]];

else:
X.append(Hz);
autoval1.append(DeltaE[0]);
autoval2.append(DeltaE[1]);
autoval3.append(DeltaE[2]);
autoval4.append(DeltaE[3]);
autoval5.append(DeltaE[4]);
autoval6.append(DeltaE[5]);
autoval7.append(DeltaE[6]);
autoval8.append(DeltaE[7]);


Imagen=0;
fig, ax = plt.subplots()
ax.plot(X,autoval1, 'r', label='Autovalor 1',linewidth=3);
ax.plot(X,autoval2, 'b', label='Autovalor 2',linewidth=3);
ax.plot(X,autoval3, 'g', label='Autovalor 3',linewidth=3);
ax.plot(X,autoval4, 'y', label='Autovalor 4',linewidth=3);
ax.plot(X,autoval5, 'c', label='Autovalor 5',linewidth=3);
ax.plot(X,autoval6, 'm', label='Autovalor 6',linewidth=3);
ax.plot(X,autoval7, 'orange', label='Autovalor 7',linewidth=3);
ax.plot(X,autoval8, 'k', label='Autovalor 8',linewidth=3);

strn='H_Levels.png'
ax.set_xlim(0,0.05);
ax.set_ylim(20,120);
```

```
ax.set_xlabel(r'$H_z$ (T)');
ax.set_ylabel(r'$\Delta$ E (GHz)');

plt.savefig(strn, format='png',dpi=500)
```