Computadoras cuánticas

Miquel, César

2002

Tesis presentada para obtener el grado de Doctor en Ciencias Físicas de la Universidad de Buenos Aires

UNIVERSIDAD DE BUENOS AIRES

Facultad de Ciencias Exactas y Naturales

Departamento de Física

Computadoras Cuánticas

por César Miquel

Director de Tesis: Juan Pablo Paz
Lugar de trabajo: Departamento de Física, FCEyN, UBA.
Trabajo de Tesis para optaro por el título de Doctor en Ciencias Física

10 de septiembre del 2002

Computadoras Cuánticas

César Miquel

Tésis de Doctorado
Facultad de Ciencias Exactas y Naturales, UBA

Resumen

En esta tesis se estudian tres aspectos importantes de la computación cuántica. En una primera parte se examina la estabilidad de las computadoras cuánticas frente al ruido. Esta es una rama de vital importancia para la computación cuántica ya que su éxito o fracaso depende de nuestro conocimiento del efecto del ruido sobre la computadora y de nuestra habilidad e ingenio para superar sus efectos desvastadores. Aqui estudiamos el efecto de errores en algoritmos cuánticos y mostramos qué estrategias podemos usar para combatirlos. Presentamos resultados numéricos que permiten predecir el impacto de errores unitarios en computación cuántica. Para combatirlos discutimos las distintas alternativas, presentamos varios códigos de corrección de errores, y analizamos el proceso de corrección continua de errores. En segundo lugar se examina y presenta un método novedoso que permite representar el estado de una cmoputadora en el espacio de fases. Este método no solamente permite visualizar el estado de una computadora cuántica aislada sino que se adapta muy bien al caso en que esta evolucione de manera ruidosa. Por otra parte, esta representación permite comprender mejor la naturaleza de ciertos algoritmos cuánticos. En tercer, y último lugar presentamos una revisión de las técnicas experimentales que permiten realiazr computación cuántica en espectrómetros de RMN. Como resultado original en este campo, presentamos un método (y los primeros resultados experimentales) para medir la función de Wigner.

palabras claves: Computadoras cuánticas, códigos de corrección de errores, decoherencia, resonancia magnética en líquidos, distribuciones en espacio de fase.

Quantum Computers

César Miquel

PhD. Thesis
Facultad de Ciencias Exactas y Naturales, UBA

Abstract

In this thesis we investigate three important issues in quantum computation. In the first part we examine the stability of a quantum computer when subjected to noise. Our ability to create a working quantum information processor depends on our understanding of these effects. In this context we numerically study unitary errors in Shors factoring algorithm and characterize its impact. To overcome them we present several quantum error correcting codes and analyze the process of continuous error correction. In the second part we study and present a novel method to represent the state of a quantum computer in phase space. This method not only allows us to have a pictoric representation of the state of an isolated computer but also adapts well when describing its evolution in a noisy environment. In addition it gives us new insight on how certain algorithms work. In the last part we present a revision of the experimental techniques that allow us to perform quantum computation on NMR spectrometers. As an original result in this area we present a new method (and the first experimental results) to measure the Wigner distribution of a finite dimensional system.

keywords: Quantum computers, quantum information processing, quantum error correction, decoherence, nuclear megnetic resonance in liquids, phase space distributions.

Índice General

1. Introducción

1.1. Computación y física

A fines del año 1999 la sociedad estaba conmocionada con el llamado "problema del año 2000". Se especulaba que al acabarse el siglo muchos programas que almacenaban los últimos dos dígitos del año comenzarían a hacer cálculos erróneos y los sistemas de todo el mundo terminarían colapsando. Por suerte el año nuevo pasó sin mayores problemas pero algunos investigadores sostienen que el problema del año 2000 no es nada comparado con el que tendremos dentro de 30-40 años. Si observamos el avance de la industria electrónica vemos que la densidad de transistores en un procesador se duplica cada aproximadamente 18 meses. Esta tendencia, conocida como ley de Moore, nos dice que, de sostenerse, dentro de unos años el tamaño de los transistores llegará a ser el de un átomo. En ése punto deberá haber un cambio radical para poder avanzar en la velocidad de procesamiento de las computadoras. Claramente para ese entonces tendremos que entender y aprender a utilizar en otra escala los efectos cuánticos en los transistores. Pero quizás el cambio de paradigma necesario para que la industria informática siga avanzando en velocidad llegue de la computación cuántica. Esta tesis trata de estas nuevas computadoras que recientemente mostraron que son capaces de resolver ciertos problemas en forma exponencialmente más veloz que sus parientes clásicas.

Si bien la teoría de la computación es una disciplina caracterizada por un alto grado de abstracción en algún punto debemos reconocer que, para que la teoría tenga alguna aplicación, debemos ser capaces de construir una computadora con elementos del mundo real. En otras palabras, las computadoras son sistemas físicos que almacenan y procesan información regidas por las mismas leyes que rigen el comportamiento de la naturaleza. Dichas leyes (las leyes de la física) limitan la capacidad de dichas computadoras. Partiendo de reconocer que "la información y su procesamiento" son procesos físicos, desde la década del 70 algunos físicos empezaron a estudiar cuales eran las limitaciones fundamentales impuestas por las leyes de la física a la computación. Una primer pregunta que surgió en este contexto es si es necesario consumir energía para calcular o si, por el contrario, esto puede hacerse 'gratis'. Landauer, Bennett, Fredkin [Bennet73, Fredkin82, Landauer82] y otros estudiaron esta pregunta y concluyeron que es posible computar reversiblemente (sin disipar energía) siempre y cuando el cálculo que se haga sea lógicamente reversible (o sea, que la salida permita recuperar la entrada). La existencia de una conexión directa entre reversibilidad lógica y reversibilidad física (termodinámica) es un resultado importante (obtenido por Landauer y desarrollado por Bennett) que abrió una nueva rama de la física: la física de la información.

Toda computadora puede ser diseñada a partir de ciertas componentes elementales (llamadas compuertas). Una ingeniosa implementación de una

compuerta lógica reversible que funciona sin pérdida de energía fue propuesta por Fredkin. Su modelo utiliza como portadores de la información discos que interactúan mediante choques perfectamente elásticos. Para que esta computadora (que es en realidad un complicado billar) funcione, los choques deben ser perfectamente elásticos y las velocidades iniciales deben ser conocidas con precisión arbitrariamente grande. Los trabajos de Bennett y Landauer muestran que la disipación de energía (o el aumento de entropía) puede ser evitado durante la computación pero está inevitablemente asociado al borrado de la información. La llamada "conjetura de Landauer" establece que por cada bit borrado se produce inevitablemente un aumento de entropía que es $\Delta S = k \log 2$. En consecuencia, la irreversibilidad física sólo será posible si se cuenta con una cantidad infinita de espacio para almacenar información sin borrarla.

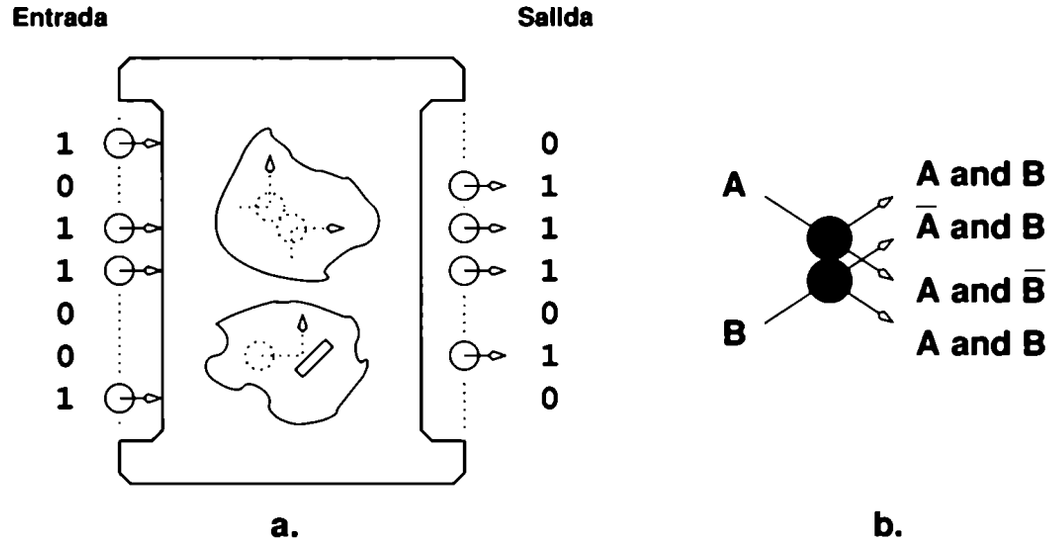


Figura 1.1: Computadora de billar de Fredkin. (a) Los bits son representados por la presencia (1) o ausencia (0) de los discos a tiempos determinados t_i. (b) Las operaciones lógicas se producen al colisionar los discos. En el ejemplo vemos como es posible implementar un AND y negaciones (denotadas por una barra sobre la letra de entrada, i.e., $\bar{A}$)

1.2. Computación cuántica

Pero además de estudiar las limitaciones que sus leyes imponen sobre la computación, la física ha sugerido recientemente nuevos caminos para esta disciplina. En efecto, para un físico es interesante notar que la definición abstracta de computadora (basada en lo que se conoce como máquina de Turing [Turing37]) implícitamente supone que la misma es un sistema clásico (o sea, su estado y evolución están regidos por leyes "clásicas", i.e. no cuánticas). Siempre se supone que la computadora evoluciona pasando por una secuencia de estados computacionales bien definidos. Sin embargo la física ha descubierto que una descripción completa de la naturaleza requiere el abandono de los conceptos clásicos basados en nuestro sentido común, por ejemplo el de trayectoria. Las leyes de la mecánica cuántica nos permiten imaginar sistemas

(computadoras) cuya dinámica y sus posibles estados sean cuánticos.

Richard Feynman fue uno de los primeros en analizar este tipo de "computadoras cuánticas" [Feynman82] preguntándose si las mismas no serán imprescindibles para simular eficientemente a la naturaleza en escalas microscópicas. Su argumento es el siguiente: supongamos que existe una correspondencia uno a uno entre los estados de una computadora y un sistema físico. Decimos que la computadora simula al sistema si podemos hacer que recorra sus estados en el mismo orden en que lo haría el sistema. ¿Que sistemas seremos capaces de simular entonces con una computadora clásica? Para contestar ésta pregunta Feynman define primero cuándo un sistema es simulable: cuando el tiempo (y el espacio) que requiere nuestra computadora no crece exponencialmente con el tamaño del sistema. Todos los problemas de la mecánica clásica son, en este sentido, simulables. Sin embargo, para simular problemas de mecánica cuántica en una computadora clásica se requiere un tiempo que es exponencial en el tamaño del sistema a simular. Un ejemplo sencillo es la simulación de un sistema de N espines. La dimensión del espacio de estados del sistema es exponencial en N (2^N). Para almacenar la información necesaria para reconstruir el estado del sistema se necesita una cantidad exponencialmente grande de memoria por lo que calcular su evolución también requiere un tiempo exponencial. En consecuencia esto sugiere que sólo usando una computadora que evoluciona según las leyes de la mecánica cuántica sería posible simular la evolución del estado cuántico de un sistema sin necesidad de usar un tiempo exponencial en el número partículas.

A partir de los trabajos de Feynman muchos investigadores comenzaron a estudiar las posibles implicancias de la existencia de computadoras cuánticas. En rigor, toda la teoría de la computación debería ajustarse a esta idea, un proceso que se ha desarrollado en sucesivos pasos y que aún no ha concluido. Un pionero en el campo, David Deutch, introdujo en sucesivos trabajos, la noción de máquina de Turing cúantica [Deutch85] y definió y estudió las compuertas lógicas que serían necesarias para construir computadoras cuánticas *universales* [Deutch88]. Así, dio los ingredientes elementales para poder escribir cualquier "programa" cuántico. Durante muchos años no estuvo claro si las computadoras cuánticas eran mas poderosas, desde el punto de vista de la complejidad algorítmica, que las computadoras clásicas. David Deutch presentó una seria de problemas de limitada utilidad, que son resueltos más eficientemente por una computadora cuántica [Deutch92]. Sin embargo, la situación cambió radicalmente en 1994 con el trabajo de Peter Shor quien demostró que el problema de la factorización de números enteros puede resolverse eficientemente en una computadora cuántica. El problema consiste en encontrar los factores primos de un número entero N. Todos los algoritmos clásicos conocidos son exponencialmente grandes en el tamaño de la entrada (el número de bits para representar N). Sin embargo, Shor presentó un algoritmo que factoriza enteros en tiempo polinomial utilizando una computadora cuántica [Shor94]. Este problema es importante no sólo desde el punto de

vista académico sino también práctico ya que la seguridad de uno de los algoritmos de encripción de clave pública más utilizados se basa en la dificultad de factorizar enteros grandes. Por primera vez existía un problema relevante que podía ser resuelto más eficientemente con una computadora cuántica.

A partir de ese momento hubo una explosión de trabajos y un gran interés en el área de computación cuántica. Además, la idea no sólo resulta interesante para los físicos sino que numerosos matemáticos y científicos de la computación comenzaron a estudiar éste nuevo tipo de computadoras. En 1996, L. Grover [Grover96] presentó el que probablemente es hoy el segundo algoritmo de computación cuántica más importante: la búsqueda en una base de datos. El problema a resolver puede describirse matemáticamente de la siguiente manera: sea $f(x)$ una función con $x \in \{1, \ldots, N\}$ que vale 1 para algún valor w y 0 para todos los demás. La tarea a resolver es encontrar el valor de w evaluando la función $f(x)$. Es claro que si no sabemos nada sobre la función $f(x)$ (es decir: no conocemos ninguna propiedad que pueda ayudar en la búsqueda) la única manera de encontrar w es simplemente probar evaluando $f(x)$ en distintos valores de x hasta encontrar aquel para el cual $f(x) = 1$. Es claro que este algoritmo es exponencial en el tamaño de la base de datos: si $N = 2^L$, el tiempo promedio que tardamos en encontrarlo es $t \sim O(N) = O(2^L)$. Sin embargo, Grover encuentra que con una computadora cuántica es posible encontrar el ítem marcado (w) en un tiempo que es proporcional a $\sqrt{N}$. Aunque el algoritmo sigue siendo exponencial es sorprendente que cuánticamente podamos hacerlo de una manera mucho más eficientemente. Hoy en día el campo ha crecido al punto que hay varios libros sobre el tema y las revistas tienen ya una sección dedicada a lo que se conoce como 'procesamiento de información cuántica' que no sólo incluye computación cuántica sino temas tan diversos como métodos criptográficos utilizando mecánica cuántica, teoría de juegos, comunicación a través de canales cuánticos, etc.

1.3. Contenido de la tesis

En esta tesis presentaremos diversos resultados que contribuyeron al desarrollo de esta nueva área. Uno de los principales problemas de las computadoras cuánticas es su inherente inestabilidad. La computadora cuántica funciona como un gran interferómetro en el que la señal proveniente de la interferencia entre distintas trayectorias computacionales puede ser fácilmente borrada por la acción de cualquier fuente de ruido que introduzca decoherencia. Debido a la sensibilidad de las computadoras a las perturbaciones es importante entender el origen de dicha inestabilidad y buscar métodos para corregir errores cuando se procesa información con una computadora cuántica. Después de una breve introducción a la computación cuántica en el próximo capítulo, dedicaremos el capítulo 3 a estudiar la sensibilidad de las computadoras cuánticas y los métodos que permiten que, en teoría, se puedan implementar algoritmos en computadoras que no son perfectas. Allí presentaremos los resultados de las primeras simulaciones completas del algoritmo de factorización de enteros de

Shor. Se estudió la evolución de una computadora cuántica construida siguiendo una propuesta originalmente presentada por I. Cirac y P. Zoller en la cual la información está almacenada en un conjunto de iones atrapados que son manipulados mediante pulsos láser convenientemente elegidos. Después de diseñar la secuencia de pulsos necesaria para factorizar el número 15 se examinó la evolución de dicha computadora, formada por 18 iones, sometida a cerca de 15,000 pulsos láser imperfectos. Mostramos que el impacto de las imperfecciones es tan importante que para poder implementar cualquier algoritmo más o menos realista es fundamental utilizar algún método de corrección de errores. Además, encontramos una fórmula que permite predecir la dependencia de la fidelidad de la computadora como función del número de pulsos y de la dispersión en el error de los mismos. La segunda mitad del capítulo está dedicada a introducir la teoría de los códigos de corrección de errores y presentar nuestra contribución a esta área: el código perfecto de 5 bits y los estudios de los llamados "códigos de corrección continua". Estos modelos intentan estudiar cuales son los límites del proceso de corrección de errores.

El capítulo 4 está dedicado al estudio de métodos para representar el estado de la computadora en el espacio de las fases. La idea consiste en introducir una representación con propiedades similares a las que definen las funciones de Wigner de una partícula pero para sistemas con espacio de Hilbert de dimensión discreta como son las computadoras cuánticas). El capítulo está dedicado a introducir el formalismo que desarrollamos para funciones de Wigner discretas, presentar el diseño de un circuito cuántico que permite medir dicha función de Wigner y finalmente discutir cómo pueden ser utilizadas para analizar la evolución de una computadora cuántica.

Finalmente, el capítulo 5 combina varios de los elementos presentados en capítulos anteriores e incluye algunos resultados experimentales originales. Comienza con una introducción a lo que hoy en día es el área más desarrollada experimentalmente en computación cuántica: resonancia magnética nuclear (RMN). Utilizando RMN en líquidos es posible utilizar los espines nucleares de moléculas simples como qubits de una computadora cuántica. La primer mitad del capítulo introduce las nociones necesarias para poder entender cómo es posible implementar una computadora cuántica utilizando RMN. La segunda mitad presenta una serie de resultados experimentales obtenidos en el espectrómetro del LANAIS en la FCEyN y en el Los Alamos National Laboratorio. Los experimentos incluyen resultados sobre la generación de estados pseudo-puros y la implementación de compuertas lógicas (por ejemplo, CNOT) (resultados ya obtenidos por varios grupos en otros laboratorios del mundo). Por último presentamos los resultados correspondientes a la utilización del método desarrollado en el capítulo 3 para medir la función de Wigner discreta de un sistema de dos qubits. Este último resultado es totalmente inédito y muestra una nueva forma de hacer tomografía con una computadora cuántica.

El contenido de esta tesis ha sido publicada parcialmente de acuerdo al

siguiente detalle: los estudios de estabilidad de computadoras cuánticas del capítulo 3 están publicadas en la referencia [Miquel97] (la referencia [Miquel96], que corresponde al contenido de la tesis de Licenciatura del autor sirvió de base para ese trabajo). El descubrimiento del código perfecto de 5 qubits fue presentado en [Laflamme95]. La representación de los estados y la evolución de una computadora cuántica mediante la función de Wigner del capítulo 4 fue descrita por primera vez en las referencias [Miquel02a, Bianucci01]. La medición de la función de Wigner mediante técnicas de RMN como fue presentada en el capítulo 5 fue publicada en [Miquel02b].

algoritmo sino una tesis propuesta independientemente por Church y Turing en 1930 [1]. Para Turing, cualquier algoritmo puede ser implementado por una máquina de Turing Universal y por lo tanto los dos conceptos son equivalentes.

Cualquier máquina de Turing se puede pensar entonces como una función que relaciona un estado inicial de la cinta con un estado final. Vemos entonces que podemos pensar a un programa como una función que relaciona estado inicial con estado final.

Una vez que tenemos una máquina de Turing que implementa un algoritmo es importante saber la dependencia del número de pasos necesarios para ejecutar el algoritmo, como función del "tamaño" de los datos de entrada. Si el tiempo que tarda una máquina de Turing en resolver un problema es polinomial en el tamaño del problema decimos que el algoritmo es de tipo **P**. Estos problemas son considerados como "resolubles en la práctica". Existe, sin embargo, una gran variedad de problemas para los cuales no se ha encontrado un algoritmo polinomial que sea capaz de resolverlos con una máquina de Turing. Dentro de esta clase de problemas existe un subgrupo que tiene la siguiente propiedad: es posible verificar eficientemente si un candidato es o no solución de una instancia particular del problema. Sin embargo, el número de candidatos posibles crece de manera exponencial con el tamaño del problema. Esto implica que si bien, dado una posible solución es fácil verificar si es la correcta el problema principal radica en encontrarla. [2] A esta clase de complejidad se la conoce como **NP** que proviene de 'non-deterministic polynomial', es decir, que se resuelven en tiempo polinomial con una máquina de Turing no-determinista (la resolución es 'no-determinista' ya que requiere de un candidato obtenido de ese modo). Los problemas **NP** surgen reiteradamente en la práctica y es de gran importancia obtener métodos para resolverlos eficientemente. Entre los ejemplos célebres de problemas **NP** podemos citar (además de todos aquellos que pertenecen a la clase **P**, ya que $\mathbf{P} \subseteq \mathbf{NP}$) el del viajante de comercio, la división de un circuito minimizando el número de conexiones, etc. Otro ejemplo, al que nos referiremos más adelante, es la factorización de números enteros. En ese caso, los mejores algoritmos dependen exponencialmente del tamaño del número (identificamos al tamaño del número como la cantidad de bits necesarios para almacenarlo, $L \sim \log N$). Muchos de los problemas duros y la factorización en particular, pueden ser ex-

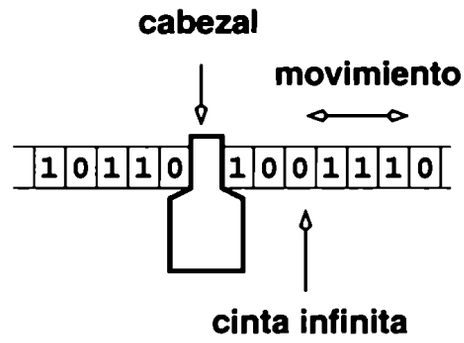


Figura 2.1: Esquema de una máquina de Turing y sus partes: el cabezal de lectura/escritura y la cinta infinita marcada con los dos tipos de símbolos (0 y 1).

[1] Church llegó a las mismas conclusiones que Turing pero por un camino totalmente distinto. Más tarde se mostró que ambos métodos eran equivalentes.

[2] Esto puede parecer obvio pero existen problemas para los cuales, simplemente *verificar* si un candidato es solución o no requiere un tiempo exponencial en una máquina de Turing.

plotados para diseñar algoritmos de encripción en los que se utiliza una clave para codificar los mensajes.

Es importante destacar que la eficiencia de un algoritmo no sólo debe tener en cuenta la dependencia del tiempo con el tamaño del problema sino también debe considerar los requerimientos de espacio. En efecto, es fácil demostrar que todo algoritmo exponencial en t puede volverse polinomial si uno utiliza una cantidad exponencial de "espacio". En consecuencia la clasificación adecuada de eficiencia tiene en cuenta la dependencia de "los recursos utilizados" (espacio y tiempo) con el tamaño del problema. Desde un punto de vista físico la eficiencia debe incluir todos los recursos físicos (incluyendo, por ejemplo, la energía).

2.1.2. Compuertas y circuitos lógicos

Del análisis anterior vimos que podíamos pensar a una máquina de Turing (ejecutando un cierto algoritmo) como una función que relaciona estados computacionales iniciales con estados finales. Otra forma de ver un programa entonces es pensarlo como una caja negra con una entrada y una salida (ver la figura 2.2). Si ponemos un estado s_{inicial} en la entrada obtenemos después de un tiempo finito t el estado s_{final} en la salida. Una forma conveniente de codificar los estados inicial y final (y todos los otros estados que la computadora adquiere en tiempos intermedios del cómputo) es representándolo en forma binaria. Cada estado está entonces etiquetado por una secuencia de l "bits" que pueden asumir dos valores: 0 o 1.

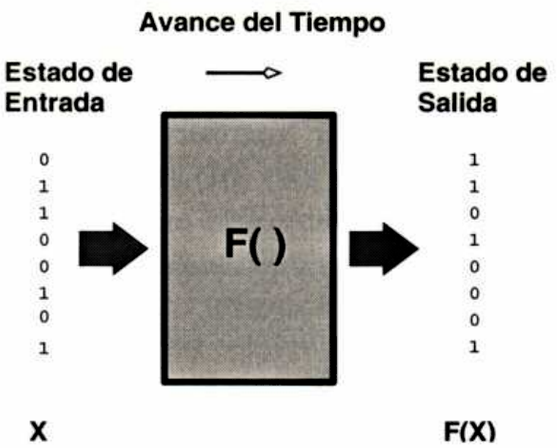
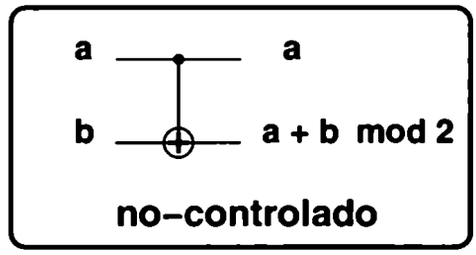


Figura 2.2: Cualquier programa puede ser visto como una función que relaciona estados de entrada con estados de salida. Aquí los estados están representados por número binarios.

Así como es común dividir un programa grande en subrutinas mas simples podemos dividir nuestra caja en cajas más chicas que realizan operaciones más elementales. La descripción más elemental a la que podemos llegar es una en la cual realizamos operaciones muy simples en un conjunto reducido de bits de nuestra máquina. A estas cajas se las conoce como *compuertas*. Operan sobre unos pocos bits y tardan un tiempo τ que es fijo e independiente del estado de entrada. Por ejemplo una operación elemental puede simplemente permutar dos bits de la memoria o ponerlos en cero. En la figura 2.3 vemos otro ejemplo de compuerta elemental.

Como vemos la compuerta opera sobre dos bits a y b. El sistema evoluciona de izquierda a derecha a medida que transcurre el tiempo. En la figura 2.3



ab	cd
00	00
01	01
10	11
11	10

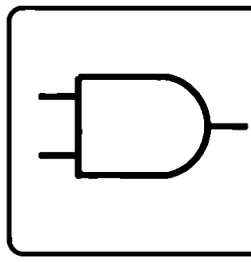
Figura 2.3: Esquema de una compuerta no-controlado. La compuerta de tiene dos bits de entrada y dos de salida. A la derecha de la figura vemos la tabla de verdad de dicha compuerta.

vemos que la computadora pasa del estado ab al cd. El valor de cd depende de ab y está resumido en la tabla que aparece en la figura. Otra forma de representar esta tabla (que usaremos más adelante) es utilizando notación matricial. A cada uno de los cuatro estados posibles los podemos identificar con vectores (o kets en la notación de mecánica cuántica que usaremos después) de un espacio vectorial. La acción de la compuerta se puede pensar como un operador (matriz) que cambia el estado de los kets (vectores) de entrada. Si bien ahora trataremos sólo con estados clásicos la notación nos prepara para poder pasar a definir compuertas cuánticas que operaran en un espacio de Hilbert. Si llamamos $|00\rangle, |01\rangle, |10\rangle$ y $|11\rangle$ a los estados posibles en la entrada, podemos definir la compuerta anterior diciendo cómo actúa sobre cada uno de los estados $|00\rangle \ldots |11\rangle$. En esa base, la matriz de esta compuerta es:

$$m_{i,j} = \langle i|U_{cnot}|j\rangle = \begin{pmatrix} 1 & & & \\ & 1 & & \\ & & 0 & 1 \\ & & 1 & 0 \end{pmatrix} \tag{2.1}$$

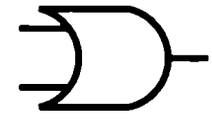
Vemos que la función de esta compuerta es permutar los estados $|10\rangle$ y $|11\rangle$. Se la conoce con el nombre de no-controlado ya que su función es negar el bit b si el bit a es 1 (o sea, la negación del segundo bit está controlado por el primero).

AND



a	b	Salida
0	0	0
0	1	0
1	0	0
1	1	1

XOR



a	b	Salida
0	0	0
0	1	1
1	0	1
1	1	0

Figura 2.4: Esquema de dos compuertas muy comunes usadas en lógica digital: al AND y el XOR. Las compuertas tienen dos bits de entrada y uno de salida. En la figura vemos además su tabla de verdad.

Es importante notar una diferencia sustancial entre las compuertas lógicas usadas comúnmente en circuitos digitales (AND y XOR) y la compuerta no-controlada descripta en la figura 2.3. En efecto, las compuertas AND y XOR son lógicamente irreversibles mientras que el no-controlado es lógicamente reversible ya que la entrada puede encontrarse a partir de la salida (el circuito es uno a uno). La importancia de la reversibilidad para nosotros es que, en mecánica cuántica la evolución está dada por un operador que siempre es reversible y por lo tanto resultará imprescindible utilizar lógica reversible para diseñar los circuitos.

La lógica irreversible trae como consecuencia inevitable la pérdida de información y la disipación de energía. Landauer mostró que la disipación necesariamente tiene lugar durante el borrado de información (y puede evitarse en toda otra instancia del cálculo). Como consecuencia de esto es imposible hacer una compuerta irreversible que no pierda energía. Este hecho hace que la lógica reversible parezca una solución más atractiva. Sin embargo casi todas las computadoras usadas hoy en día operan con lógica irreversible. El motivo principal es que las computadoras actuales disipan mucho más energía que la asociada al borrado (que disipa $kT \log 2$ por bit borrado). En consecuencia, el descubrimiento de Landauer es, hasta hoy, irrelevante en la práctica (aunque algunos especulan que puede tener importancia con nuevas tecnologías en las próximas décadas). Por otra parte, el uso de lógica reversible no es muy atrayente ya que la misma tiene una serie de desventajas que la hacen menos práctica que la irreversible.

Es natural preguntarse qué tipo de compuertas necesitamos si queremos estudiar un circuito arbitrariamente complejo. Esta pregunta nos conduce al concepto de compuertas universales. Diremos que un conjunto de compuertas C es universal (con respecto a las funciones de $\mathcal{Z} \to \mathcal{Z}$) si es posible, utilizando sólo compuertas en C, implementar cualquier función de enteros en enteros. Un ejemplo de compuerta universal con respecto a las funciones de $\mathcal{Z} \to \mathcal{Z}$ es la compuerta NAND. La tabla de verdad para esta compuerta irreversible es:

a	b	NAND
0	0	1
0	1	1
1	0	1
1	1	0

Las compuertas de Toffoli, que aparecen en la figura 2.5, son también universales. Dicha compuerta, que tiene tres bits de entrada y tres de salida, es una simple generalización del no-controlado. Vimos que el no-controlado niega el segundo bit si el primero es 1. La compuerta de Toffoli niega el tercer bit si los dos primeros bits son ambos iguales a 1. Si llamamos $\{|000\rangle, |001\rangle, \ldots |111\rangle\}$ a los 8 posibles estados de entrada, la acción de la compuerta de Toffoli sobre cada estado de entrada está descripta por la siguiente matriz:

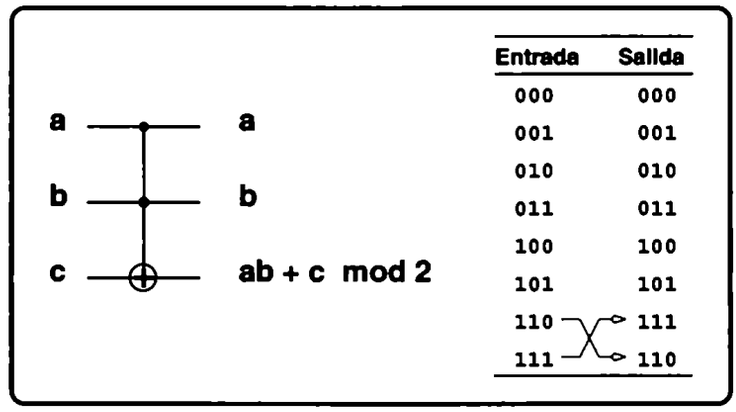


Figura 2.5: En la figura vemos un esquema de la compuerta de Toffoli. Dicha compuerta tiene tres bits y tiene como función negar el tercer bit cuando los dos controles (a y b) son 1.

$$\begin{pmatrix} 1 & & & & & & & \\ & 1 & & & & & & \\ & & 1 & & & & & \\ & & & 1 & & & & \\ & & & & 1 & & & \\ & & & & & 1 & & \\ & & & & & & 0 & 1 \\ & & & & & & 1 & 0 \end{pmatrix}$$

Es fácil demostrar que la compuerta de Toffoli es universal; es decir, con ella podemos implementar cualquier función booleana. Para la demostración sólo hay que darse cuenta que es posible hacer utilizando una compuerta NAND (que se sabe es universal) utilizando una compuerta de Toffoli seteando el tercer bit en 1. Este resultado es importante porque nos limita el número de compuertas necesarias para hacer computación clásica reversible, y como veremos después, también computación cuántica.

2.2. Computadoras cuánticas

Hasta ahora sólo hemos discutido computadoras y circuitos clásicas. ¿Qué es una computadora cuántica? Uno de las tantas suposiciones que hemos hecho implícitamente es que cada bit debe estar siempre en un estado bien definido 1 o 0. Supongamos ahora que el bit es un sistema cúantico de dos niveles. Por ejemplo si pensamos en un espín, los estados computacionales (clásicos) pueden ser elegidos como $|0\rangle \equiv |+\hbar/2\rangle$ y $|1\rangle \equiv |-\hbar/2\rangle$ donde $|\pm\hbar/2\rangle$ son los auto-estados de la componente z del operador de spin con autovalor $\pm\hbar/2$. Sin embargo, la mecánica cuántica, basada en el principio de superposición, permite que bits cuánticos existan en superposiciones coherentes de estados computacionales. En efecto, el estado $|\phi\rangle = \frac{1}{\sqrt{2}}(|1\rangle + |0\rangle)$ es uno de los estados posibles de cualquier bit cuántico (que en el futuro llamaremos qubit). Esto es algo que clásicamente es imposible pero perfectamente aceptable desde el

punto de vista de la mecánica cuántica.

Si tenemos un conjunto de m bits el estado más general de ese conjunto es un estado del espacio de de Hilbert de dimensión 2^m. Dicho estado puede escribirse como una combinación lineal de la forma:

$$|\phi\rangle = \sum_{i=0}^{2^m-1} c_i|i\rangle \tag{2.2}$$

donde los elementos de la base que denotamos por $|i\rangle$ son de la forma $|0\rangle \equiv |00\ldots0\rangle, |1\rangle \equiv |00\ldots01\rangle, |2\rangle \equiv |00\ldots10\rangle, |3\rangle \equiv |00\ldots11\rangle, \ldots$, etc. corresponden a un estado de espines $|a\rangle = |a_{m-1}a_{m-2}\ldots a_0\rangle$ donde $a = \sum_{i=0}^{m-1} a_i 2^i$. Los coeficientes c_i son números complejos que satisfacen la condición de normalización:

$$\sum_{i=0}^{n} |\, c_i \,|^2 = 1$$

Los estados $|a\rangle$ corresponden a los "estados computacionales", es decir que una computadora cuántica es un sistema que puede existir en un estado como (2.2) y, por lo tanto, estar en "superposición de estados computacionales".

La evolución de la computadora está regida por cierto Hamiltoniano $H(t)$, que tiene asociado un operador de evolución temporal $\hat{U}(t)$. En las próximas secciones describiremos como diseñar este operador en términos de operaciones elementales que constituirán un circuito cuántico. Si pensamos entonces que un operador $\hat{U}(t)$ corresponde a un cierto programa concluimos que al aplicar este operador a un estado del tipo (2.2) cada componente del estado $|\Phi(t)\rangle$ corresponderá en algún sentido a una computadora clásica haciendo un cálculo distinto (que corresponde al mismo programa con una condición inicial distinta). Ésta propiedad se la conoce como "paralelismo cuántico". El estado computacional final es la superposición coherente de aquellos estados que se obtienen siguiendo todas las trayectorias computacionales posibles. Sin embargo, no es evidente que el paralelismo cuántico tenga alguna utilidad práctica ya que al efectuar una medición al final del cómputo, el estado de la computadora colapsará sobre un único estado computacional. Recién cuando Shor presentó su algoritmo de factorización polinomial es que la mayoría de los investigadores entendieron el poder de éstas nuevas computadoras, que reside en un uso inteligente del paralelismo cuántico.

En general cualquier operador unitario $\hat{U}$ perteneciente a $\mathrm{SU}(2^n)$ es un circuito cuántico de n bits en n bits. El único requisito que impone la mecánica cuántica es que la evolución sea unitaria y por lo tanto reversible. El ejemplo más simple es el de una compuerta que opera sobre un único bit. Sólo hay dos compuertas clásicas de este tipo: la identidad y la negación. Sin embargo existen infinitas compuertas cuánticas que operan sobre un bit: cualquier operador perteneciente a el grupo de transformaciones de $\mathrm{SU}(2)$.

Así como en la teoría de compuertas lógicas clásicas teníamos compuertas universales también las hay en la generalización cuántica. Se puede demostrar que existe una compuerta mediante la cual se puede construir cualquier transformación unitaria de n bits en n bits [Deutch88]. Sorprendentemente, Barenco et al [Barenco95] demostraron que el no-controlado junto con una rotación del estado de un bit son capaces de generar cualquier transformación unitaria de n bits en n bits. Es decir, basta tener una compuerta de dos qubits (el no-controlado) y poder hacer cualquier compuerta de 1 qubit para poder generar cualquier operador unitario. Esto debe contrastarse con el caso clásico para el cual la mínima compuerta que cumpla con el análogo clásico de esa propiedad tiene 3 entradas. Ya aquí podemos ver que vivir en un espacio más amplio (como es el de Hilbert) tiene sus ventajas.

2.3. Algoritmo de Grover

Como ya dijimos, no es obvio que el paralelismo cuántico sea suficiente para ver que una computadora cuántica sea mas poderosa que una clásica. La forma más ilustrativa de ver esto es mediante un ejemplo: el algoritmo de Grover que implementa una búsqueda en una base de datos. El problema a resolver es el siguiente: dada una base de datos con $N \gg 1$ elementos sin ningún orden aparente tenemos que encontrar en ella un elemento que satisface un cierto criterio de búsqueda. Es importante pensar que la base de datos no tiene ningún orden particular. Encontrar un número de teléfono en una guía es fácil porque está ordenada alfabéticamente. Sin embargo, el problema inverso de encontrar un nombre dado el número de teléfono es mucho más complicado porque no hay ningún orden. La única manera de encontrarlo es buscar uno por uno y eso requiere, en promedio, mirar $O(N)$ registros. La novedad es que con una computadora cuántica sólo debemos mirar $O(\sqrt{N})$. Esto es extremadamente sorprendente: mirando menos registros de la base podemos encontrar uno sin saber nada más! En realidad para poder lograrlo utilizamos el paralelismo cuántico para mirar *todos* los registros simultáneamente. Claramente, desde el punto de vista clásico esto es imposible pero la mecánica cuántica abre nuevas posibilidades a la computación.

Para poder presentar el algoritmo de Grover, pensamos ahora el problema en términos matemáticos: la base de datos es un conjunto de registros $\{x = 0, 1, \ldots, N-1\}$ y el criterio de búsqueda está representado como una función $f_w(x)$ con $x \in \{0, \ldots, N-1\}$ que tiene la propiedad de que: $f_w(x) = 0$ si $x \neq w$ pero $f_w(w) = 1$. Dada la función $f_w(x)$ queremos encontrar el valor de x tal que $f_w(x) = 1$ (es decir, el elemento $x = w$). La función $f_w(x)$ puede interpretarse como una caja negra u oráculo al cual se lo alimenta con un valor de x y responde si es o no el ítem buscado (devolviendo un 1 o un 0). En términos de operadores cuánticos vamos a pensar que el oráculo es un operador $\hat{U}_w$ que actúa sobre los estados computacionales de la siguiente manera:

$$\hat{U}_w|x\rangle = (-1)^{f_w(x)}|x\rangle \tag{2.3}$$

es decir, si le aplico $\hat{U}_w$ a mi estado me devuelve el mismo estado si $x \neq w$ o $-|x\rangle$ si $x = w$. Es decir, pone la información de si es el elemento buscado en la fase del estado. Otra forma de escribir el mismo operador es:

$$\hat{U}_w = 1 - 2|w\rangle\langle w|. \tag{2.4}$$

Dicho operador tiene la siguiente interpretación geométrica: dado cualquier vector en el espacio de Hilbert del sistema, lo refleja con el hiper-plano ortogonal al vector $|w\rangle$. Es decir: invierte la componente paralela a $|w\rangle$ manteniendo la proyección sobre $|w^{\perp}\rangle$ invariante. El algoritmo de Grover para encontrar quien es $|w\rangle$ parte inicializando la computadora cuántica en el estado

$$|s\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N} |x\rangle \tag{2.5}$$

es decir, una superposición de todos los estados posibles (registros de la base de datos). Si se hace una medición del estado de la computadora es claro que la probabilidad de obtener el estado $|w\rangle$ es $1/N$ pues $|\langle w|s\rangle|^2 = 1/N$. El algoritmo de Grover hace aumentar la probabilidad de que el estado de la computadora cuántica sea $|w\rangle$. Para ello hace falta definir un operador más:

$$\hat{U}_s = 2|s\rangle\langle s| - 1. \tag{2.6}$$

Al igual que $\hat{U}_w$, este operador refleja alrededor del hiper-plano ortogonal al estado $|s\rangle$. Finalmente podemos definir una iteración del algoritmo de Grover como el siguiente operador:

$$\hat{U}_{\text{Grover}} = \hat{U}_s \hat{U}_w. \tag{2.7}$$

En la figura 2.6 se muestra la primera iteración del algoritmo de Grover. El estado de la computadora es $|s\rangle$ y definamos $|\langle w|s\rangle| = 1/\sqrt{N} \equiv \sin\theta$. Al aplicar $\hat{U}_w$ al estado, la componente paralela al eje $|w\rangle$ se invierte mientras que la paralela a $|w^{\perp}\rangle$ se mantiene igual. Ahora el estado de la computadora forma un ángulo de $-\theta$ con respecto al eje $|w^{\perp}\rangle$ (o 2θ entre $|s\rangle$ y $\hat{U}_w|s\rangle$). Aplicando $\hat{U}_s$ a este estado, la componente perpendicular a s cambia de signo mientras que la paralela se mantiene igual. El estado en este punto forma un ángulo de 3θ con respecto a $w^{\perp}$. Aquí termina la primer iteración del algoritmo de Grover. Si repetimos el procedimiento una vez más veremos el estado estado resultante forma un ángulo de 5θ con respecto a $w^{\perp}$. Es decir, una iteración rotó el vector un ángulo 2θ. Es fácil convencerse que después de n iteraciones el vector habrá rotado un ángulo $2n\theta$ hacia la dirección $|w\rangle$ formando un ángulo $(2n+1)\theta$ con $|w^{\perp}\rangle$. Es claro que para que la probabilidad de medir el estado $|w\rangle$ sea lo más grande posible debe satisfacerse que $2n\theta \sim \pi/2$ de donde deducimos que el número de iteraciones necesarias para que $p \sim 1$ es $n \sim \pi/(4\theta)$. Como $\theta \sim 1/\sqrt{N}$ el número de iteraciones es $n \sim \sqrt{N}$.

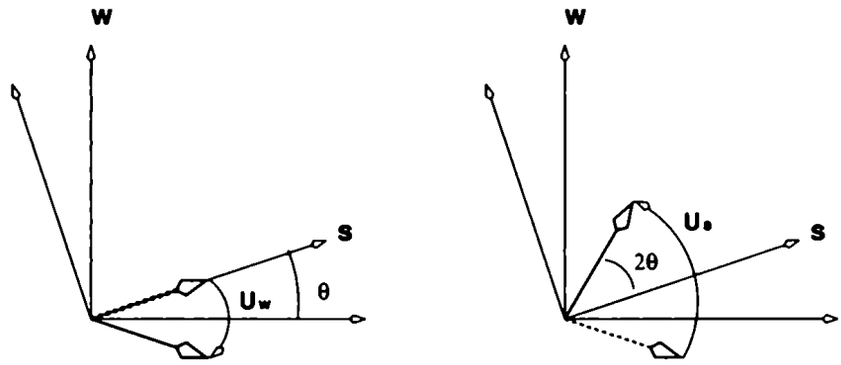


Figura 2.6: Iteración del algoritmo de Grover. A la izquierda se ve la acción del operador $\hat{U}_w$ sobre el estado $|s\rangle$. A la derecha vemos la acción de $\hat{U}_s$ sobre el estado $\hat{U}_w|s\rangle$. El estado final es $\hat{U}_s\hat{U}_w|s\rangle = \hat{U}_{\text{grover}}|s\rangle$.

El algoritmo de Grover es de gran utilidad y el truco de amplificación de amplitudes es usado en varias aplicaciones de computación cuántica. Es interesante destacar que es posible obtener una mejora cuadrática en cualquier algoritmo **NP** utilizando el método de Grover. Justamente todos estos problemas tienen la particularidad de que es fácil verificar si una instancia del problema es la solución (así como era fácil mirar un registro en la base y ver si el número de teléfono corresponde al que estamos buscando). El problema radica en el gran número de candidatos a ser la solución. Es claro que usando el truco de amplificación de amplitud podemos mejorar la probabilidad de toparnos con la solución correcta. Desafortunadamente esto no achica demasiado la brecha exponencial entre los algoritmos **P** y los **NP** pero sigue siendo un resultado útil y muy general.

Una vez que nos convencimos que sería útil tener una computadora cuántica podemos pasar a preguntarnos cuales son los requisitos para poder hacer una. Rápidamente uno se encuentra con dificultades debido a que los sistemas cuánticos son increhiblemente sensibles a los efectos del entorno (razón por la cual no vemos estos efectos en el mundo macroscópico de la vida cotidiana). En el próximo capítulo veremos que problemas hay y qué soluciones existen para combatir el efecto del entorno sobre una computadora cuántica.

3. Estabilidad de las computadoras cuánticas

Dado que poca gente discute los beneficios de disponer de una computadora cuántica y que se han invertido considerables sumas de dinero en investigación para construir una de estas máquinas, surge naturalmente la pregunta ¿porqué no tenemos una aún? El principal obstáculo para la computación cuántica se origina, sin duda alguna, en los efectos del ruido sobre la dinámica de estas máquinas. El problema radica en la sensibilidad de los sistemas cuánticos a interacciones con su entorno. Ninguna computadora está totalmente aislada de su entorno lo que hace que no podamos pensar a la computadora como un sistema aislado. Para poder observar efectos cuánticos como la interferencia de estados o entrelazamiento debemos poder manipular sistemas microscópicos como átomos, fotones en cavidades, espines nucleares, etc. En la mayoría de los casos las energías asociadas a cambios de niveles son pequeñas y por lo tanto debemos tomar en cuenta energías de interacción con el entorno. La interacción de la computadora con grados de libertad externos va a introducir errores en la ejecución de algoritmos por lo que es claro que encontrar métodos para prevenir y detectar errores es de suma importancia.

Existe otra fuente de errores en computación cuántica además de la interacción con el entorno: la imperfección de las operaciones necesarias para implementar compuertas cuánticas. En computación clásica, los elementos de un circuito pueden estar en dos estados: 0 y 1. Sin embargo, en computación cuántica el espacio de estados de un qubit es inifinitamente más grande que el de un bit clásico por lo que hay muchos tipos de errores que pueden aparecer al querer hacer una operación lógica. Es cierto que en las computadoras digitales los estados computacionales son transformados en voltajes y en realidad hay mas de dos estados posibles. Típicamente en un circuito digital el 0 está representado por voltaje menor que un cierto umbral mientras que el 1 está representado por voltaje superior a otro umbral. Las compuertas digitales están diseñadas de manera que tengan cierta tolerancia y de compuerta en compuerta se corrigen los voltajes para mantenerlos por arriba (o por debajo) del correspondiente umbral. Esta corrección hace que 'microscópicamente' la dinámica sea irreversible: distintos estados de entrada a la compuerta son mapeados al *mismo* estado a la salida. Es decir: la 'tolerancia' necesaria para que la computación clásica no se destruya hace que la computadora tenga que disipar. Desafortunadamente esta misma técnica no sirve en una computadora cuántica porque su dinámica es reversible y por lo tanto incompatible con estas ideas. Lo mismo ocurre con la computadora de billar de la sección 1.1. Zurek [Zurek84] mostró que las computadora de Fredkin es sensible a las perturbaciones iniciales y al posicionamiento de los espejos. Pequeñas perturbaciones hacen que la computadora pierda sincronización y se arruina rápidamente el cálculo. Es decir: la reversibilidad de la computadora (junto con la complejidad) es un obstáculo para que la dinámica sea estable.

En principio el panorama parece desalentador: cualquier perturbación minúscula o interacción con un entorno destrozaría las correlaciones cuánticas

tan importantes en los algoritmos cuánticos o descarrilaría el cómputo. Sin embargo, recientemente se demostró que, aún en presencia de errores, es posible implementar algoritmos cuánticos eficientemente. La idea es usar códigos de corrección cuántica de errores que utilizan redundancia para proteger la información almacenada en la computadora. En este capítulo nos dedicaremos a dos tareas: 1) estudiar los efectos de errores y su impacto en distintos algoritmos cuánticos y 2) discutir cuales son las posibles soluciones y mostrar nuestra contribución a esta área. En la primer parte del capítulo discutiremos los distintos tipos de errores e introduciremos algunas nociones de cómo medir la inestabilidad de un cómputo. Luego presentaremos resultados numéricos del estudio de inestabilidad para el caso particular de una computadora cuántica basada en una trampa de iones fríos que implementa el algoritmo de factorización de Shor. Estos resultados muestran que, para poder implementar cualquier algoritmo en una computadora cuántica va a ser necesario usar algún tipo de corrección de errores. En la segunda parte del capítulo presentamos el formalismo de estabilizadores para los códigos de corrección, introducimos el código perfecto de 5 bits y presentamos el formalismo de la corrección continua. Finalmente presentamos algunas ideas propuestas por otros investigadores de métodos de corrección pasiva.

3.1. Estabilidad de las computadoras clásicas

Antes de poder hablar de inestabilidad en computadoras cuánticas es útil mirar que pasa en sus familiares las computadoras clásicas. Desde los comienzos hubo numerosos obstáculos para lograr que las computadoras fueran lo suficientemente confiables. Las primeras máquinas estaban hechas con válvulas y era común que las mismas se quemaran o dejasen de funcionar. Sin embargo, con la llegada del transistor los problemas de 'hardware' se hicieron menos importantes aunque no desaparecieron en su totalidad. Si bien las computadoras electrónicas sufren pequeños desniveles en las tensiones asignadas al cero o uno lógico, cada compuerta compensa, a su salida, las fluctuaciones de tensión de manera que el error no se acumula. De esta manera los elementos mismos con los que está hecha la computadora están diseñados de manera de compensar errores y corregirlos [Keyes89].

Desafortunadamente esta misma idea no se puede usar cuando la computación es físicamente reversible. Una buena forma de ver esto es pensar en la computadora de Fredkin. Los pequeños errores en las velocidades iniciales o en la posición de los espejos usados para reflejar las bolas hacen que, el billar rápidamente se vuelva 'caótico'. Cualquier mecanismo usado para re-establecer el curso de la computación implicaría gastar energía de alguna manera u otra.

¿Y que hay de la interacción de la computadora con el entorno? A primera vista parecería que esto no importa en las computadoras actuales pero dejar una PC por unas horas sin la refrigeración adecuada pronto muestra lo contrario. La gran diferencia es que, para cambiar el estado de un bit en una computadora clásica se requiere energía que es grande comparada con la energía

térmica provista por el entorno mientras que las computadoras cuánticas son mucho mas sensibles.

Como vemos, las computadoras clásicas también sufren los efectos de errores pero su correcto funcionamiento es obtenido entregando energía al sistema, construyendo las mismas con dispositivos que son tolerante a fallas y usando códigos de corrección de errores cuando todo falla. Un ejemplo de estos últimos se usa en los discos compactos de música y computación actual, en los discos rígidos y las memorias. La idea es usar redundancia para almacenar la información en varios lugares de manera que si el ruido afecta algún bit, se puede reconstruir la información a partir de los demás. Esta idea de redundancia es llevada al extremo en el transbordador espacial de la NASA donde se utilizan 3 computadoras que todas hacen lo mismo y en presencia de algún error se decide por 'voto de la mayoría'.

3.2. Tipos de perturbaciones

3.2.1. Errores unitarios de evolución

Como dijimos, en computación cuántica podemos distinguir dos tipos de errores: unitarios y no-unitarios. Aquí discutiremos los primeros. La evolución de un algoritmo cuántico se describe utilizando un operador de evolución unitario $\hat{U}_{\text{se}}(t)$ dependiente del tiempo. En presencia de errores, este operador se ve modificado y esto se representa por un operador $\tilde{U}(t)$ que describe su comportamiento anómalo. La computadora recorre el espacio de Hilbert siguiendo alguna trayectoria que se separa de la trayectoria correcta dada por $\hat{U}(t)$. La característica de este tipo de errores es que conservan la probabilidad, es decir, la evolución sigue siendo unitaria.

Un ejemplo simple de error unitario puede ser una 'sobre-rotación' de algún bit. Supongamos que queremos aplicar una operación simple sobre un bit de un registro: invertir el estado del mismo. Esto se logra aplicando un operador de rotación que rota el estado de ese bit en π alrededor del eje x (o cualquier otro eje en el plano x-y). Sin embargo, un error que podemos esperar es el de rotar en un ángulo que no es π sino $\pi + \epsilon$. El operador unitario que representa a esta operación se $\hat{U} = \exp(-i(\pi/2 + \epsilon)\sigma_x)$ y su efecto en un estado es:

$$\begin{aligned} \tilde{U}|\psi\rangle &= \exp(-i(\pi/2+\epsilon)\hat{\sigma}_x)|\psi\rangle \\ &= [\cos(\pi/2+\epsilon) \; - i\sin(\pi/2+\epsilon)\;\hat{\sigma}_x]|\psi\rangle \\ &\sim [-\epsilon \; - i\hat{\sigma}_x]|\psi\rangle = \hat{U}_{\text{se}}|\psi\rangle + \epsilon|\tilde{\psi}\rangle. \end{aligned} \tag{3.1}$$

Este tipo de errores está presente en prácticamente todas las implementaciones conocidas hoy en día. Durante un algoritmo cuántico se producirán una serie de errores de distinta magnitud ϵ_i. El impacto de este tipo de errores, claramente dependerá de la distribución que describa los sucesivos errores ϵ_i. Podemos distinguir dos tipos de errores unitarios: aleatorios y sistemáticos.

Los aleatorios son cuando los valores ϵ_i toman distintos valores de manera azarosa: puede variar el signo, módulo o las dos cosas simultáneamente. Una forma natural de modelar estos errores es suponer que son gausianos con una dada media y un ancho σ. Los errores sistemáticos son producidos por, por ejemplo, siempre sobre-rotar en la misma dirección. También puede haber sistemáticos donde el error no es siempre constante pero tiene media distinta de cero.

Ante una propuesta experimental concreta es interesante estudiar el impacto de dichos errores en algoritmos cuánticos para poder estimar la eficiencia de nuestra computadora. Más adelante mostraremos un estudio concreto del estudio de dicho tipo de errores presentado en [Miquel97].

3.2.2. Errores producidos por el entorno

Los errores producidos por el entorno son un poco mas complicados de describir. En la mayoría de los casos la computadora evolucionará a estados que dejan de ser puros por lo que es necesaria una descripción en términos del operador densidad $\hat{\rho}$ del sistema. El punto de partida para describir la dinámica de la computadora es suponer que, inicialmente el sistema `computadora + entorno` está en un estado no correlacionado:

$$\hat{\rho}_0 = \hat{\rho}_{\text{in}} \otimes \hat{\rho}_{\text{e}}$$

El sistema compuesto evolucionará unitariamente con algún Hamiltoniano que incluye tanto la evolución de cada uno de sus componentes como de la interacción sistema-entorno. Como en realidad nos interesa qué le pasa al sistema (y no al entorno), debemos ignorar los grados de libertad externos al mismo. Esto se logra trazando sobre el entorno para obtener la matriz densidad del sistema:

$$\hat{\rho}_{\text{out}} = \text{Tr}_{\text{ent}} \left[\hat{U}\, \hat{\rho}_0\, \hat{U}^\dagger \right] \tag{3.2}$$

Resulta incómodo utilizar la expresión anterior para hacer cálculos y existe una manera mejor de describir la evolución. Presentaremos ahora un formalismo más cómodo. Para simplificar el cálculo haremos una suposición adicional [1]: que el entorno se encuentra en un estado puro al principio, es decir, $\hat{\rho}_{\text{e}} = |e_0\rangle\langle e_0|$. Si $\{|e_k\rangle\}$ es una base del espacio de Hilbert del entorno y escribimos explícitamente la traza de la ecuación (3.2) obtenemos:

[1] Dicha suposición no es necesaria y si no se hace se obtiene una expresión ligeramente más complicada para los operadores $\hat{E}_k$.

$$\begin{aligned}\hat{\rho}_{\text{out}} &= \text{Tr}_{\text{ent}}\left[\hat{U}\left(\hat{\rho}_{\text{in}}\otimes\hat{\rho}_{\text{e}}\right)\hat{U}^{\dagger}\right] \\ &= \sum_{k}\langle e_k|\hat{U}\left(\hat{\rho}_{\text{in}}\otimes|e_0\rangle\langle e_0|\right)\hat{U}^{\dagger}|e_k\rangle \\ &= \sum_{k}\hat{E}_k\,\hat{\rho}_{\text{in}}\,\hat{E}_k^{\dagger}\end{aligned} \tag{3.3}$$

donde $\hat{E}_k \equiv \langle e_k|\hat{U}|e_0\rangle$ son operadores que actúan en el espacio de estados del sistema. El conjunto de operadores $\{\hat{E}_k\}$ caracteriza la interacción entre el entorno y el sistema y satisfacen que $\sum_k \hat{E}_k^{\dagger}\hat{E}_k \leq I$. La ecuación (3.3) muestra que la evolución del sistema está descripta por un mapa: el estado inicial del sistema $\hat{\rho}_{\text{in}}$ es mapeado en el estado:

$$\hat{\rho}_{\text{in}} \rightarrow \hat{\rho}_{\text{out}} = \mathcal{E}(\hat{\rho}_{\text{in}}) = \sum_{k}\hat{E}_k\hat{\rho}_{\text{in}}\hat{E}_k^{\dagger}. \tag{3.4}$$

Esta expresión tiene una interpretación sencilla: el mapa (3.4) puede pensarse como un proceso en el cual el estado inicial $\hat{\rho}_{\text{in}}$ pasa a ser:

$$\hat{E}_k\hat{\rho}_{\text{in}}E_k^{\dagger}/\text{Tr}(\hat{E}_k\hat{\rho}_{\text{in}}E_k^{\dagger})$$

con probabilidad $\text{Tr}(\hat{E}_k\hat{\rho}_{\text{in}}E_k^{\dagger})$. Esto es similar a lo que, en la literatura de teoría clásica de la comunicación, se conoce con el nombre de canal ruidoso. En ellos la información es sometida a diversos errores con alguna distribución de probabilidad. Por ejemplo, supongamos una memoria defectuosa que, después de almacenar la información en ella por un tiempo Δt, con probabilidad p cambia el estado del bit almacenado en ella. Para describir el efecto en la información almacenada en ella supongamos que $\vec{p} = (p_0, p_1)$ describe las probabilidades que, a $t = 0$ el bit esté en el estado 0 (1) con probabilidad p_0 (p_1). Después de un tiempo Δt, debido al efecto del ruido en el canal, dichas amplitudes se ven modificadas y resultan:

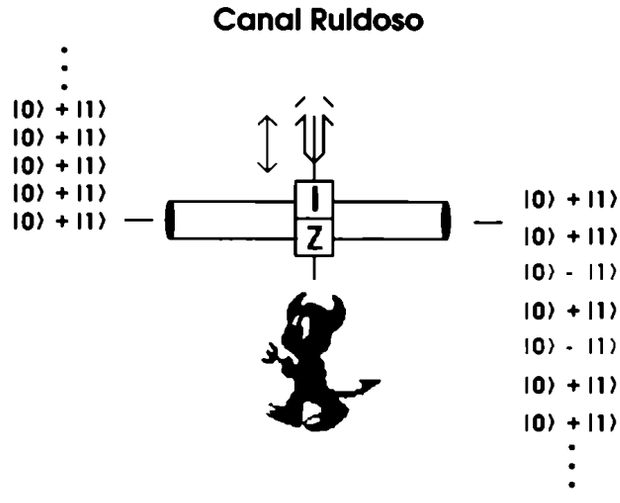
$$\begin{pmatrix} p_0' \\ p_1' \end{pmatrix} = \begin{bmatrix} 1-p & p \\ p & 1-p \end{bmatrix}\begin{pmatrix} p_0 \\ p_1 \end{pmatrix} \tag{3.5}$$

Vemos que la relación que existe entre el estado antes y después de entrar al canal ruidoso es lineal. Algo análogo ocurre en el caso cuántico donde las matrices densidad de entrada y salida esta relacionadas de manera lineal. La gran diferencia consiste en el menú de posibilidades cuando tenemos que elegir un tipo de error. En el caso clásico solo hay dos tipos de errores posibles: el caso que mostramos arriba de intercambiar el estado o un caso en el que, independientemente del estado inicial siempre pasamos al mismo estado final (ya sea 0 o 1). En el caso cúantico tenemos una gran variedad de operadores $\hat{E}_k$ posibles que describen los posibles errores. Para ver un caso interesante pero a la vez simple consideremos el llamado canal depolarizante. Este es

un ejemplo puramente cuántico (sin análogo clásico) que luego usaremos. En el canal depolarizante los operadores $\hat{E}_k$ son dos: $E_0 = I$ (la identidad: no hacerle nada al bit) y $E_1 = \hat{\sigma}_z$ (aplicarle $\hat{\sigma}_z$) al bit. Como muchas veces aparecerán $\sigma_{x,y,z}$ por comodidad las denotaremos por $\hat{X}, \hat{Y}, \hat{Z}$. El efecto de este canal sobre un estado $\hat{\rho}_{\text{in}}$ es entonces:

$$\hat{\rho}_{\text{in}} \rightarrow \hat{\rho}_{\text{out}} = (1-p)\hat{\rho}_{\text{in}} + p\hat{Z}\hat{\rho}_{\text{in}}\hat{Z} \tag{3.6}$$

Esta acción puede ser pensada como que existe un agente externo (un "demonio") que, aleatoriamente modifica el estado del qubit aplicando con probabilidad p el operador $\hat{Z}$ o dejándolo intacto con probabilidad $(1-p)$ (ver figura 3.1). Al principio puede parecer que no hace mucho pues, si $\hat{\rho}_{\text{in}}$ es cualquiera de los estados computacionales ($|0\rangle\langle 0|$ o $|1\rangle\langle 1|$) éstos no se modifican. Sin embargo, es fácil verificar que si el estado inicial es una superposición de dos estados computacionales (digamos por ejemplo: $\hat{\rho}_{\text{in}} = |\psi_+\rangle\langle\psi_+|$ donde $|\psi_\pm\rangle = (|0\rangle \pm |1\rangle)/2)$ entonces el efecto de $\hat{Z}$ es intercambiar $|\psi_+\rangle$ con $|\psi_-\rangle$ y ahora el canal arruinará la relación de fase existente entre estados como se puede ver usando (3.6) sobre nuestro estado inicial. Esto se ve reflejado si miramos los elementos no diagonales de $\hat{\rho}$ en la base de auto-estados de σ_z. Es fácil verificar que, después de n iteraciones, los elemento diagonales no son modificados mientras que los elementos fuera de la diagonal aparecen multiplicados por un factor $(1-2p)^n$. Asintóticamente $\hat{\rho}$ tiende a la matriz identidad, es decir: hemos perdido toda la información de fase. Este efecto es netamente cuántico y no tiene análogo clásico.



- $Z\,|0\rangle + |1\rangle = |0\rangle - |1\rangle$

- $|1\rangle$ y $|0\rangle$ son autoestados de Z con autovalor -1 y +1 respectivamente.

Figura 3.1: Nos podemos imaginar que un agente externo (el "demonio") elige aleatoriamente si aplica el operador $\hat{\sigma}_z$ o $\hat{\mathbb{I}}$ con probabilidad p y $(1-p)$.

El formalismo presentado aquí fue introducido por Kraus [Kraus83] y resulta de suma utilidad. Al conjunto de operadores $\{\hat{E}_k\}$ se los conoce como

operadores de Kraus. La transformación $\mathcal{E}()$ es un mapa que debe satisfacer ciertas propiedades: debe preservar o disminuir la traza (nunca aumentarla), ser lineal y conservar la positivadad de $\hat{\rho}$. Estos requisitos son fundamentales para que el sistema evolucione de una manera físicamente aceptable. Finalmente el formalismo es también aplicable cuando los errores son unitarios: basta reemplazar los operadores de Kraus por un único operador $\hat{E}_0 = \hat{U}$ para re-obtener la evolución unitaria.

3.3. Medidas de la sensibilidad de una computadora cuántica

Para poder estudiar los efectos de errores en la computadora necesitamos de alguna forma de cuantificar el error introducido durante el cómputo. A medida que la computadora evoluciona nos gustaría ver a qué velocidad se separan las 'trayectorias computacionales' dependiendo del error. Para ello tendremos que definir algún tipo de distancia en el espacio de Hilbert: dados dos estados del espacio nos interesa tener un número que nos dice cuan semejantes/distintos son. Hay varias medidas de distancia que pueden usarse. En esta sección hablaremos de una de una de ellas: la fidelidad.

Cuando evolucionamos la computadora por un determinado tiempo obtendremos un estado que difiere del estado correcto debido a los errores que se produjeron durante el algoritmo. Como al repetir el algoritmo (partiendo del mismo estado inicial) obtenemos un estado distinto del anterior, debemos recurrir a algún método para poder obtener una medida del error promedio. Esto se puede lograr promediando las fidelidades de cada corrida. Otra medida de la que hablaremos es la de *dimensión del espacio de Hilbert explorada.* Dicha mediad da un indicio de cuan grande es el espacio de Hilbert explorado, debido a los errores en las distintas evoluciones.

3.3.1. Fidelidad

El concepto más simple es el de la fidelidad: dados dos estados $|\psi_1\rangle$ y $|\psi_2\rangle$ nos gustaría poder ver que tan separados están en el espacio de Hilbert. Una medida natural es tomar el producto escalar de los dos vectores. Como el producto escalar de dos vectores del espacio de Hilbert no es un número real conviene definir la fidelidad como:

$$\mathcal{F}(|\psi_1\rangle, |\psi_2\rangle) = \sqrt{|\langle\psi_1|\psi_2\rangle|^2}. \tag{3.7}$$

Definida de esta manera $\mathcal{F}$ toma valores entre 0 y 1: cuando es 0 los estados son exactamente ortogonales mientras que cuando vale 1 $|\psi_1\rangle$ y $|\psi_2\rangle$ son idénticos.

3.3.2. Dimensión del espacio explorado

Otra medida útil en la estabilidad de las computadoras cuánticas es la dimensión del espacio explorada por las distintas trayectorias computacionales

debido a errores. La idea es relativamente simple e intuitiva: si evolucionamos un estado inicial $|\psi_0\rangle$ después de cada realización obtendremos un estado final distinto $|\psi_i\rangle$ debido a errores en la evolución. Podemos entonces generar una lista $\mathcal{L} = (|\psi_1\rangle, \ldots, |\psi_N\rangle)$ de N vectores en el espacio de Hilbert de dimensión D. Cada vector en la lista tiene una probabilidad de ocurrencia p_i de ocurrir que dependerá del ruido. Promediando sobre la perturbación obtenemos una matriz densidad promedio que describe el estado perturbado:

$$\hat{\rho} = \sum_{j=1}^{N} p_j \, |\psi_j\rangle\langle\psi_j|. \tag{3.8}$$

La entropía de este estado se define como:

$$H(\hat{\rho}) = -\mathrm{Tr}\left[\hat{\rho} \log_2 \hat{\rho}\right] \tag{3.9}$$

y es un número real que va entre 0 y $\log_2 D$.

Analicemos un poco qué información nos aporta la entropía para lo cual conviene discutir dos casos extremos. Si no perturbamos el sistema, después de evolucionarlo obtendremos siempre un único estado. En este caso todas las p_i son 0 salvo una y es obvio que la entropía es entonces 0. En el peor de los casos, el ruido sera tal que todos los estados finales $|\psi_i\rangle$ son ortogonales y con igual probabilidad $p_i = 1/D$ (es obvio que a lo sumo podemos obtener D estados distintos por lo que la suma en (3.8) tiene D términos distintos con igual probabilidad). Es fácil de calcular entonces la entropía y obtenemos $H(\hat{\rho}) = \log_2 D$. De estos ejemplos vemos que $H(\hat{\rho})$ definida así está de alguna manera, midiendo el espacio que ocupan nuestros vectores finales en el espacio de Hilbert. Si todos están relativamente juntos obtendremos un valor chico mientras que si las trayectorias son muy distintas ocuparan una dimensión mayor en el espacio de Hilbert. Es natural entonces definir la dimensión explorada por los vectores como:

$$D_{\text{expl.}} = 2^{H(\hat{\rho})} \tag{3.10}$$

Desafortunadamente para sistemas relativamente grandes (mas de 15-20 qubits) resulta difícil medir (3.9). El problema radica en que para poder calcular $H(\hat{\rho})$ debemos diagonalizar $\hat{\rho}$ (una matriz de $D \times D$ donde $D = 2^L$ con L el número de qubits). La dimensión de dicha matriz crece exponencialmente con L por lo cual sólo se puede utilizar dicha definición para sistemas chicos. Afortunadamente existe otra magnitud llamada la entropía lineal que es una cota superior para la entropía. Su definición es:

$$H_{\text{lineal}} = -\log_2 \left(\mathrm{Tr}\, \hat{\rho}^2\right). \tag{3.11}$$

Esta expresión tiene dos ventajas en comparación con la entropía: la primera es que en algunas circunstancias es más fácil de encontrar expresiones

analíticamente cerradas. La otra es que, cuando no se puede calcular exactamente, se la puede acotar inferiormente por el cuadrado del autovalor más grande de $\hat{\rho}$. Dicho autovalor puede ser encontrado usando técnicas iterativas que no necesitan diagonalizar al operador densidad.

3.4. Un estudio de inestabilidad: factorizando 15

Para poder tener una idea de los errores que podemos esperar en una computadora cuántica real decidimos estudiar el algoritmo de factorización de Shor [Shor94] implementado en una trampa de iones como la propuesta en [Cirac95]. En el apéndice A se da una descripción detallada de la implementación de Cirac y Zoller de la trampa. Aquí presentaremos los resultados de simulaciones numéricas de una computadora cuántica que ejecuta el algoritmo de factorización de Shor para encontrar los factores primos de un número pequeño ($N = 15$). En las simulaciones seguimos la evolución cuántica de $n_i = 18$ iones atrapados sometidos a $n_p \sim 15,000$ pulsos láser. Estos estudios fueron los primeros en los cuales se investigaba la evolución de una computadora cuántica basada en una trampa de iones ejecutando un algoritmo considerablemente complejo (debido a la gran cantidad de operaciones).

El algoritmo de factorización de Shor es, a esta altura muy conocido. El problema a resolver es, dado un número $N \in \mathbb{N}$ que es el producto de dos enteros desconocidos p y q hallar dichos factores. Aunque el problema parece trivial, hasta el día de hoy el mejor algoritmo de factorización está basado en el 'Sieve Numérico' (ver [Knuth81], [Silverman02]) y es exponencial en N. Según [RSA02], para poder factorizar un número de 310 dígitos decimales con este algoritmo necesitaríamos aproximadamente el equivalente a 342,000,000 computadoras Pentium de 500 MHZ con 170 Gb de memoria (cada una!) trabajando durante un año *sin parar*. Esta cifra es bastante ilustrativa para mostrar la dificultad de factorizar números grandes. Aunque factorizar enteros parece una tarea bastante inútil la realidad es que en la actualidad es de gran importancia: la seguridad del algoritmo más utilizado hoy en día en todo el mundo para sistemas de encripción con clave pública está basada en la imposibilidad de factorizar enteros grandes. Todas las transacciones seguras que se utilizan en la web hoy en día y las comunicaciones vía 'secure shell' (`ssh`) utilizan el algoritmo de encripción desarrollado por Rivest, Shamir y Adleman (RSA) de encripción con clave pública.

3.4.1. Factorización cuántica

El algoritmo que encontró Shor utiliza una computadora cuántica para factorizar el número N resolviendo un problema equivalente: encontrar el orden r de un número y. Es decir, el menor entero r tal que $y^r = 1 \bmod N$. Para resolver este problema se utiliza una computadora cuántica que encuentra la periodicidad (de una manera eficiente, es decir, polinomial en el número de bits de N) de una función $F(a) : \mathbb{Z} \rightarrow \mathbb{Z}$. En el algoritmo de factorización:

$$F(a) = y^a \bmod N \tag{3.12}$$

donde $y, N \in \mathbb{Z}$ son dados. Para encontrar los factores de N uno elije un y al azar e inicia la computadora en el estado $|\Psi_0\rangle = \frac{1}{\sqrt{q}} \sum_{j=0}^{q-1} |j\rangle_1 |0\rangle_2$. Aquí, $|j\rangle_{1,2}$ representan dos registros de la computadora cuyos estados están definidos por la representación binaria de j (q que debe estar entre N^2 y $2N^2$). Si a este estado le aplicamos la transformación unitaria que mapea el estado $|j\rangle_1 |0\rangle_2$ en $|j\rangle_1 |y^j \bmod N\rangle_2$, el estado de la computadora resulta: $|\Psi_1\rangle = \frac{1}{\sqrt{q}} \sum_{j=0}^{q-1} |j\rangle_1 |y^j \bmod N\rangle_2$. Si ahora hacemos la transformada de Fourier cuántica al primer registro y lo medimos obtenemos una distribución $P(c)$ que da la probabilidad de que dicho registro tome el valor c Esta distribución puede ser calculada analíticamente utilizando mecánica cuántica y resulta una función con picos separados por un múltiplo de $1/r$. Midiendo la distancia entre los picos uno puede determinar eficientemente r (ver figura 3.2).

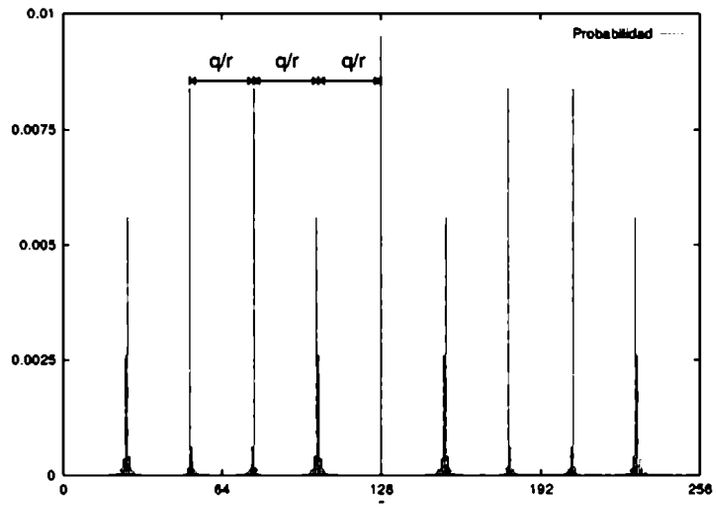


Figura 3.2: Transformada de Fourier del primer registro. El gráfico muestra la probabilidad de medir c en el primer registro. Vemos que es una función con picos separados por una distancia q/r. Conociendo q y la distancia entre picos podemos determinar r.

3.4.2. Una implementación utilizando iones fríos

Para poder estudiar los efectos de errores unitarios en dicho algoritmo necesitamos una implementación física para nuestra computadora cuántica. Aquí asumiremos que la computadora utiliza la propuesta de Cirac y Zoller [Cirac95] debido a que varios grupos experimentales están el proceso de construirlas. En esta implementación cada qubit es almacenado en los niveles internos de un ion. Los mismos están atrapados en una trampa lineal y enfriados con láser al nivel fundamental traslacional. Dos estados de larga vida media $|g\rangle$ y $|e\rangle$ de cada ion juegan el papel de estados computacionales. Cada ion es accedido utilizando un láser y es posible generar oscilaciones de Rabi entre los estados computacionales sintonizando la frecuencia del láser a la diferencia $\hbar\omega$ entre el estado fundamental $|g\rangle$ y el excitado $|e\rangle$. En este caso el estado cuántico de cada qubit evoluciona como $|\Psi(t)\rangle = U(t)|\Psi(0)\rangle$, donde la matriz $U(t)$ en la

base $(|g\rangle, |e\rangle)$ es:

$$U(t) = \begin{pmatrix} \cos\Omega t & -ie^{-i\Phi}\sin\Omega t \\ -ie^{i\Phi}\sin\Omega t & \cos\Omega t \end{pmatrix} \tag{3.13}$$

Controlando la frecuencia de Rabi Ω, la fase Φ y la duración t del pulso, se pueden implementar operaciones arbitrarias de un qubit. En el apéndice A se muestra con mas detalle como hacer operaciones de dos qubits pero esencialmente se ve que dichas operaciones se pueden hacer con una serie de evoluciones similares a la mostrada en (3.13). Es claro que si la intensidad o la fase del láser fluctúan, dichas variaciones modificarán la frecuencia Ω o la fase Ψ introduciendo errores en la evolución. Nos interesa estudiar el efecto de dichos errores unitarios en la evolución de nuestro algoritmo.

3.4.3. El circuito para factorización

Podemos dividir en tres etapas fundamentales al programa cuántico necesario para factorizar un número N: i) la preparación del estado inicial, ii) la exponenciación modular y iii) la transformada de Fourier cuántica. La generación del estado inicial es trivial: basta con generar el estado $|0\rangle = |0\rangle \otimes \ldots \otimes |0\rangle$ y aplicarle una Hadamard a los primeros $2L$ qubits (al primer registro). El resultado de esta operación es el estado:

$$|\Psi_0\rangle = \frac{1}{\sqrt{q}} \sum_{j=0}^{q-1} |j\rangle_1 |0\rangle_2 \tag{3.14}$$

con $q = 2^{2L}$.

La transformada de Fourier cuántica (el paso iii.) es ligeramente más complicado. Coppersmith [Coppersmith94] encontró un circuito simple que implementa la operación utilizando $O[\text{poly}(\log q)]$ compuertas. Para el caso de factorizar $N = 15$ éste número es mucho mas chico que el número de compuertas necesarias para la exponenciación modular por lo que no simularemos errores en ella. Lo mismo haremos con las operaciones necesarias para generar el estado inicial.

Sin duda la parte más complicada del algoritmo de Shor (en términos de circuitos) es la exponenciación modular. En [Miquel96] se describe con todo detalle la implementación de dicho circuito. Aquí presentaremos la idea general y algunas modificaciones que hicimos para utilizar menos qubits.

En la figura 3.3 vemos un diagrama esquemático del circuito y cómo es el resultado de aplicarlo al estado $|a, 0\rangle$. Para calcular $y^a \bmod N$ conviene expresar al número a en base 2. En esta notación $a = \sum_{i=0}^{2L+1} a_i 2^i$ donde a_i son 0 o 1. Si escribimos $y^a \bmod N$ usando la expansión de a en notación binaria vemos que:

$$
\begin{aligned}
y^a \bmod N &= y^{\left[\sum_{i=0}^{2L+1} a_i 2^i\right]} \bmod N \\
&= (y^{a_0}) \cdot (y^2)^{a_1} \cdot (y^{2^2})^{a_2} \cdot \ldots \cdot (y^{2^{2L+1}})^{a_{2L+1}} \bmod N \quad (3.15)
\end{aligned}
$$

Como los a_i son cero o uno, ésta última expresión sugiere que para poder hacer exponenciación modular, basta con ser capaz de hacer un circuito que multiplica su entrada por $y^{2^i} \pmod N$ controlada por cada uno de los bits del registro a. Cada uno de estos multiplicadores toma como entrada un ket en el estado $|ctl, y\rangle$ y tiene como salida el ket $|ctl, (C^{ctl} \cdot y)\rangle$ (es decir: y multiplicado por C) si el bit de control es 1 o $|ctl, y\rangle$ si el bit de control es 0. En la figura 3.3 se puede ver el circuito de exponenciación formado por los distintos circuitos multiplicadores. Cada uno de ellos tiene como control un bit del primer registro de la computadora y como constante de multiplicación a $y \bmod N$, $y^2 \bmod N$, $y^4 \bmod N$, etc.

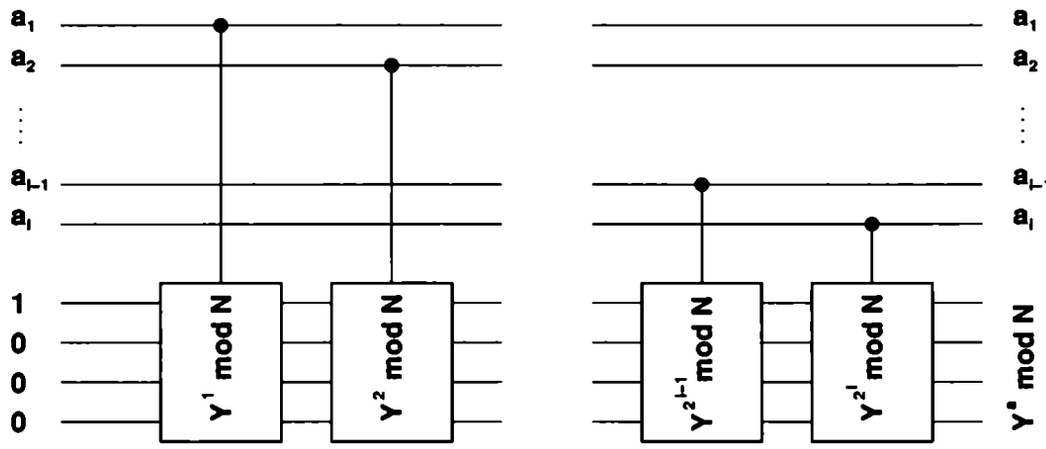


Figura 3.3: La exponenciación modular se puede reducir a una secuencia de multiplicaciones controladas (mod N). Cada circuito multiplica, controladamente por el bit a_i la entrada por el número $y^{2^i} \bmod N$.

Para ver cómo se implementa el circuito de multiplicación se puede seguir un procedimiento análogo al usado para derivar (3.15) pero para la multiplicación. Si escribimos la entrada al circuito de multiplicación como I en binario el resultado es:

$$
\begin{aligned}
C \cdot I \bmod N &= C \cdot \sum_{i=0}^{L} I_i 2^i \bmod N \\
&= \sum_{i=0}^{L} I_i \cdot (2^i \cdot C \bmod N) \bmod N \quad (3.16)
\end{aligned}
$$

Nuevamente como los I_i sólo pueden ser 0 ó 1 vemos que la multiplicación se puede escribir como una sumatoria de L términos. Cada término aparece

multiplicado por I_i y por lo tanto se puede re-interpretar como que vamos haciendo sumas controladas por los I_i (módulo N). En otras palabras: podemos descomponer la multiplicación en una serie de cajas, cada una de las cuales toma como entrada un número y le suma $(2^i \cdot C \bmod N) \bmod N$ si el control es 1 o lo deja igual si el control es 0.

Es importante recordar que la operación de multiplicación es módulo N. Esto implica que, cada una de las sumas también debe ser módulo N. Sin embargo resulta que sumar módulo N es ligeramente más complicado que simplemente sumar. Para ello lo que hay que hacer es primero sumar los números, luego comparar si el resultado es mayor (o menor) que N y finalmente restar N si hace falta. En [Miquel96] se muestra todo el proceso para la suma mod N.

Para poder simular la factorización de un número de L bits hace falta, además de los $2L + 1$ qubits del primer registro y los L bits para el segundo registro del algoritmo de Shor, qubits de trabajo. Dichos qubits se utilizan como almacenamiento temporario y tienen que partir de un estado de referencia y terminar en el mismo estado para poder factorizarlos del estado de la computadora. Debido a que la simulación de un sistema cuántico crece exponencialmente con el número de bits es útil minimizar el número de qubits temporarios para poder hacer la simulación. Para ello utilizamos un circuito modificado de suma que no utiliza bits de trabajo (como lo hace el presentado en [Miquel96]). Resulta que es fácil hacer un circuito que suma uno al registro de entrada y que no necesita bits de trabajo. En la figura 3.4 se ve dicho circuito. Es fácil ver que si escribimos la entrada en notación binario obtenemos, a la salida, el mismo número más 1. La desventaja de este circuito es que necesita compuertas controladas por mucho qubits. Sin embargo, dichas compuertas son muy sencillas de implementar utilizando la computadora de Cirac y Zoller por lo que no representa una limitación. Para sumar un número que no sea uno basta con observar que también se puede hacer circuitos que suman 2, 4, etc. Por ejemplo, para sumar 2 a la entrada basta con obviar todas las operaciones en el bit menos significativo. Para sumar 4 obviamos las operaciones en los 2 bits menos significativos y así siguiendo.

Para realizar la exponenciación modular necesitamos $4L + 2$ qubits: L bits almacenan el resultado mientras que $L + 2$ bits son usados como bits temporarios. Si le sumamos los $2L$ bits necesarios como primer registro de la computadora queda un total de $4L + 2$ qubits. Para factorizar $N = 15$ necesitamos 18 iones pero además hay un qubit adicional (sistema de dos niveles) que representa los grados de libertad traslacionales del centro de masa. Aquí estamos truncando el oscilador armónico quedándonos con los primeros dos estados: el fundamental y primer excitado. Claramente es fundamental que el sistema esta muy bien aislado térmicamente para no excitar al centro de masa a estados con más de 1 fonón. La operación de exponenciación modular requiere del orden de $O(10)L^5$ operaciones elementales (compuertas de 1 y 2 qubits). En definitiva: se simularon 18 iones de 2 niveles sometidos a 15,000 pulsos láser ($\sim 10^4$ pulsos resonantes y 4×10^3 pulsos fuera de resonancia). Los primeros ocho iones almacenaban a, los próximos 4: $y^a \bmod N$ y los últimos

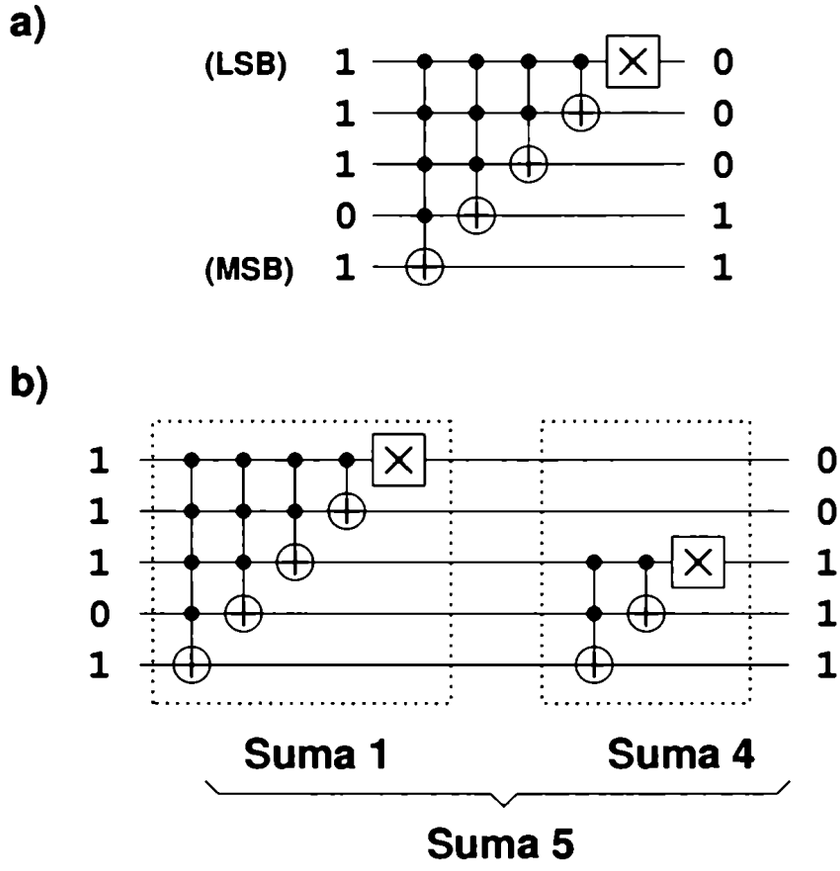


Figura 3.4: a) Circuito que suma 1 a la entrada. En la figura vemos el circuito aplicado al número 23 (10111 en base 2). Para sumar numeros más grande se combina con circuitos que suman potencias de 2. Por ejemplo, en la figura b) se muestra un circuito que suma 5. Estos modelos de sumadores tienen la ventaja de no usar bits extras de trabajo aunque requieren de compuertas con muchos controles.

6 se usaban como bits temporales.

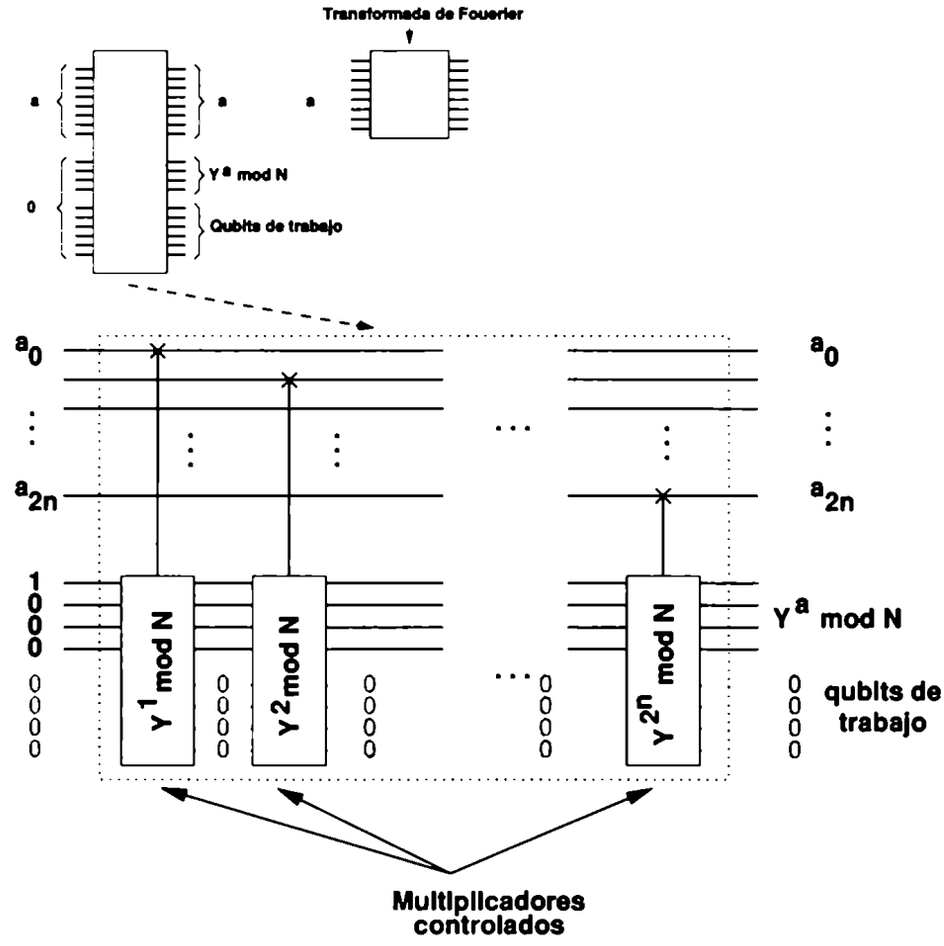


Figura 3.5: Esquema general del circuito usado para simular el algoritmo de Shor. La computadora tiene tres registros: en el primero se almacena a, en el segundo y^a mod N (después de aplicar el circuito de exponenciación modular) y el tercero se usa para almacenar bits de trabajo.

3.4.4. Resultados de las simulaciones

Para estudiar los efectos de errores unitarios simulamos el algoritmo de factorización asumiendo que la duración de los pulsos y su fase sufrían fluctuaciones. Asumimos que Ωt y Φ son variables Gaussianas aleatorias distribuidas con media $\bar{\epsilon}$ y dispersión σ. En la figura 3.6 a) se ve la probabilidad conjunta de que los dos registros de la computadora tomen el valor (a, x) antes de implementar la transformada de Fourier (la figura 3.6 b) muestra $P(c)$ después de la transformar Fourier). Para el caso en que no hay errores sistemáticos ($\bar{\epsilon} = 0$) es claro que una dispersión de tan sólo el 5% es suficiente para borrar cualquier periodicidad. En [Cirac95] se muestó que con el 5% de error en los pulsos todavía se obtienen buenos resultados. Desgraciadamente nuestros resultados muestran que no ocurre en el caso de la exponenciación modular debido a la gran cantidad de pulsos necesarios para su implementación.

Como mencionábamos hacia el principio del capítulo, la fidelidad es una buena medida para cuantificar los efectos de los errores en la ejecución del algo-

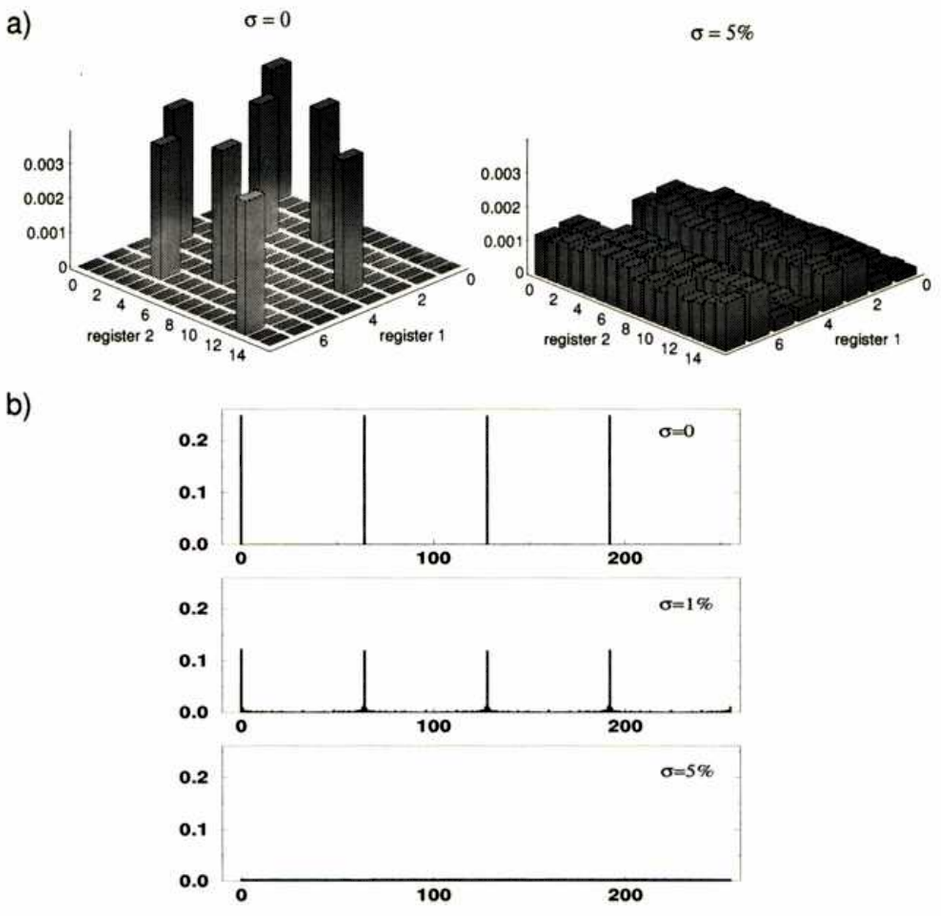


Figura 3.6: a) Probabilidad para la medición de los dos registros de la computadora. Después de la exponenciación modular, si no hubo errores ($\sigma = 0$), la computadora debería estar en una superposición de la forma: $|j\rangle|y \bmod N\rangle$. Como en la figura $y = 7$, los únicos estados presentes deberían ser: $|0\rangle|1\rangle$, $|1\rangle|7\rangle$, $|2\rangle|4\rangle$, etc. Si observamos el caso con errores ($\sigma = 5\%$) vemos que no tienen ninguna similitud con la correcta. b) Probabilidad de medir un valor en el primer registro después de la transformada de Fourier para $\sigma = 0, 1$ y 5%. Para el último caso toda la periodicidad se perdió y se obtiene una distribución uniforme.

ritmo. En la figura 3.7 se muestra la fidelidad del estado al final del circuito de exponenciación modular en función de la dispersión. Los resultados numéricos son bien aproximados utilizando un modelo simple en el cual asumimos l qubits *independientes* cada uno sometido a n/l pulsos con error. Tratando al centro de masa de manera diferente pues recibe mayor cantidad de pulsos que el resto de los iones y promediando sobre una distribución Gaussiana de dispersión σ se obtiene la siguiente expresión para la fidelidad promedio:

$$\bar{F} = \left[\frac{1}{2}\left(1 + e^{-2n_t\sigma^2/l}\right)\right]^l \left[\frac{1}{2}\left(1 + e^{-2\sigma^2 n_{\rm cm}}\right)\right], \tag{3.17}$$

donde n_t es el numero total de pulsos y $n_{\rm cm}$ el numero de pulsos que afectan el centro de masa. Esta misma fórmula, ahora vista como función de n (y parametrizada para distintos valores de la dispersión) concuerda con la observación de la fidelidad como función del tiempo.

La fórmula (3.17) se puede usar para hacer una estimación de qué precisión

necesitamos para lograr hacer un numero de operaciones n. Si usamos que para el algoritmo de Shor el número de qubits necesarios es proporcional al numero de bits L del número a factorizar ($l = aL$) y el número de operaciones proporcional a $n = bL^3$ podemos usar estos datos en (3.17) y pedimos tener una fidelidad $\bar{F}$ al final del cómputo obtenemos:

$$\sigma^2 \sim -\frac{\ln \bar{F}}{n} \tag{3.18}$$

donde hemos supuesto que $|\frac{\ln \bar{F}}{aL}| \ll 1$. Es decir: el error permitido σ escalea como $1/\sqrt{n} \sim 1/L^{3/2}$, es decir, es polinomial con el tamaño del número a factorizar (y con el número de operaciones a realizar). Esto es importante pues si la dispersión fuese exponencial estaríamos en serias dificultades.

También, como habíamos mencionado es interesante estudiar el espacio explorado por el algoritmo debido al efecto de los errores. Para ello calculamos la entropía lineal (ecuación (3.11)) promediando con la distribución de errores y obtenemos:

$$S_{\text{lin}} = l + 1 - \log_2 \left[(1 + e^{-4n_t\sigma^2/l})^l (1 + e^{-4n_{\text{cm}}\sigma^2}) \right]. \tag{3.19}$$

Esta estimación utilizando el modelo simple propuesta anteriormente muestra que, para dispersiones de poco mas del 3-4% la computadora explora todo el espacio de Hilbert disponible. Desafortunadamente no es viable comprobar esta formula experimentalmente pues requiere de demasiadas corridas ya que, con cada evolución obtendríamos un estado ortogonal (si nuestra estimación es correcta) a los anteriores y debido al enorme tamaño del espacio de Hilbert, necesitamos hacer del orden de 2^{18} corridas para tener suficientes datos numéricos.

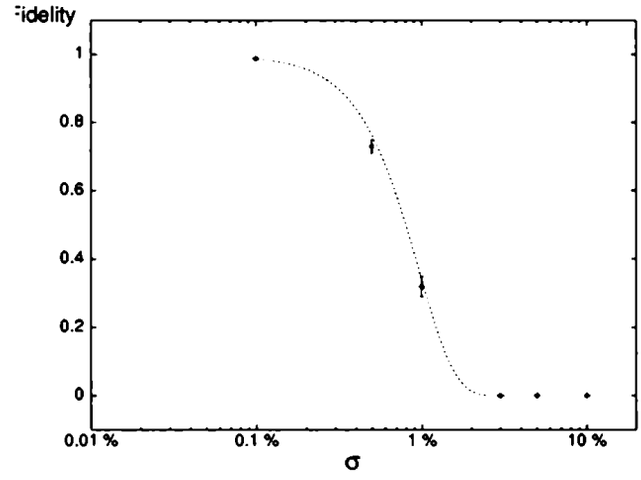


Figura 3.7: Fidelidad como función de la dispersión σ del ruido (para $\bar{\epsilon} = 0$) al final del circuito de exponenciación modular usado para factorizar $N = 15$ (con $q = 256$ e $y = 7$). La linea de puntos es la estimación que se obtiene usando el modelo sencillo descripto en el texto con $n_t = 1.5 \times 10$ y $n_{\text{cm}} = 10$ en la ecuación (3.17).

3.5. Códigos de corrección

Los estudio presentados en la sección anterior claramente muestran que va ser muy difícil hacer cualquier cálculo complicado (y útil) con una computadora cuántica a menos que podamos encontrar alguna manera de corregir los errores que se generan a medida que la computadora evoluciona. Si a los errores unitarios le sumamos los efectos del entorno es claro que debemos encontrar una estrategia para combatir los errores. Históricamente, el primer paso fue atacar el problema de errores generados por la interacción entorno-computadora. Para ello se desarrollo la teoría de códigos de corrección cuántica. Aquí haremos un resumen de dicha teoría para señalar nuestra contribución a ese campo.

3.5.1. Códigos de corrección de errores clásicos

La mejor manera de entender los códigos de corrección cuántico es empezar por introducir sus parientes clásicos. Supongamos que tenemos un bit clásico b que queremos proteger del entorno que, debido al ruido, con probabilidad p invierte su estado después de un tiempo t. Es claro que no podemos simplemente almacenar información en un conjunto de estos bits pues, después de un tiempo t varios de ellos pueden cambiar de estado sin nosotros darnos cuenta. ¿Habrá alguna manera de proteger la información almacenada allí? La respuesta es afirmativa pero para ello debemos utilizar códigos de corrección. El ejemplo más sencillo de un código de corrección es usar redundancia de la manera mas directa. Por cada bit b de información que queremos almacenar en la memoria defectuosa copiamos el estado en varios bits: (i.e. $b \rightarrow (b, b, \ldots, b)$). Después de algún tiempo t algunos de estos bits habrán sido invertidos debido al efecto del entorno. La manera más burda de recuperar la información es hacer un "voto por la mayoría", es decir, vemos cuántos de los bits están en 1 y cuantos en 0 y decidimos que el bit será 0 (o uno) dependiendo del resultado de la votación. Claramente este procedimiento será exitoso si la probabilidad p de inversión es suficientemente chica. Para ser un poco más cuantitativos supongamos que hacemos tres copias del bit b. La probabilidad de que ninguno de los bits cambie es $P(\text{todo igual}) = (1-p)^3$, la de un bit invertido es $P(1 \text{ bit}) = 3p(1-p)^2$, la de dos bits $P(2 \text{ bits}) = 3p^2(1-p)$ mientras que la de que los tres bits cambien es $P(3 \text{ bits}) = p^3$. Si no usamos redundancia, la probabilidad de que el bit este en su estado correcto después de un tiempo t es $1-p$ mientras que si usamos este procedimiento, la probabilidad de mantener la información intacta pasa a ser $1 - 3p^2 - 2p^3 = 1 - O(p^2)$, que es mas cercana a 1 para p suficientemente chica. Es decir, hemos mejorado de esta manera la probabilidad de obtener el resultado original almacenado en la memoria. Hay una gran variedad de métodos mas eficientes de codificar d bits en n portadores y existen libros enteros que estudian los códigos de corrección [MacWilliams77]. Sin embargo, este ejemplo muestra como utilizando redundancia podemos mejorar la probabilidad de que nuestra información permanezca intacta.

3.5.2. Códigos de corrección cuánticos

La próxima pregunta a hacerse es si será posible utilizar la misma idea del código presentado antes para el caso cuántico. A primera vista parece imposible pues es bien sabido [Wooters82] que es imposible clonar un bit cuántico. Además para poder hacer la votación debemos hacer una medición sobre nuestros bits que haría colapsar la función de onda destruyendo la información almacenada allí. Sorprendentemente para muchos físicos, Peter Shor mostró que era posible proteger la información cuántica lo que disparó un gran interés en encontrar códigos cuánticos y estudiar sus propiedades. Para muchos más detalles de lo presentado aquí ver [Gottesman00, Gottesman96, Preskill98] aquí nos limitaremos a presentar las ideas mas sencillas.

3.5.3. Ejemplo sencillo: 3 qubits contra errores tipo Z

Como vimos antes, el menú de errores posibles es mucho mas amplio en el caso cuántico. Supongamos ahora que tenemos un qubit q que queremos almacenar en un registro y la interacción con el entorno es modelada correctamente por el canal depolarizante presentado en la sección 3.2.2. El estado inicial $\hat{\rho}_{\text{in}} = |\psi_0\rangle\langle\psi_0|$ (donde $|\psi_0\rangle \equiv \alpha|0\rangle + \beta|1\rangle$) evoluciona, después de un tiempo t de estar en contacto con el entorno a:

$$\hat{\rho}_{\text{in}} \rightarrow \hat{\rho}_{\text{out}} = (1-p)\hat{\rho}_{\text{in}} + p\hat{Z}\hat{\rho}_{\text{in}}\hat{Z}. \tag{3.20}$$

Es fácil calcular la fidelidad del proceso y demostrar que se cumple que: $\mathcal{F}(\hat{\rho}_{\text{in}}, \hat{\rho}_{\text{out}}) = 1 - 4p|\alpha\beta|^2$. Vemos que la fidelidad depende linealmente de la probabilidad p en forma similar a lo que pasaba en el caso clásico. Ahora mostraremos que utilizando un código de corrección cuántica es posible obtener una fidelidad que difiere de 1 cuadráticamente con p.

Como en el caso clásico usaremos tres qubits para codificar la información de nuestro bit q de una manera muy sutil. La idea principal consiste en codificar los estados lógicos 0 y 1 en estados entrelazados de tres qubits de manera que si alguno de los portadores sufre un error el estado lógico es mapeado en sub-espacios ortogonales (uno por cada tipo de error). Una vez que se produjo el error es posible medir en qué sub-espacio bidimensional quedó el sistema y saber qué error se produjo. Concretamente, definamos los siguientes estados en el espacio de Hilbert del sistema de 3 qubits:

$$\begin{aligned} |0\rangle_L &\equiv \frac{1}{2}(|000\rangle + |011\rangle + |101\rangle + |110\rangle) \\ |1\rangle_L &\equiv \frac{1}{2}(|111\rangle + |100\rangle + |010\rangle + |001\rangle) \end{aligned} \tag{3.21}$$

que generan el sub-espacio lógico H_L de dimensión 2. El primer paso en cualquier código de corrección consiste en mapear el estado del qubit a un sub-espacio en un espacio de dimensión mayor. Esto se logra mediante un circuito que mapea el estado $|\psi_0\rangle = \alpha|0\rangle + \beta|1\rangle \rightarrow \alpha|0\rangle_L + \beta|1\rangle_L$. Los estados $|0\rangle_L$

y $|1\rangle_L$ están elegidos cuidadosamente para que tengan la siguiente propiedad: cuando le aplicamos un operador error (es decir, le aplicamos un operador $\hat{Z}$ a cualquiera de los portadores) obtenemos estados que son perpendiculares entre sí. Es fácil ver que los siguientes estados son todos perpendiculares: $|0\rangle_L \perp \hat{Z}_i|0\rangle_L \perp |1\rangle_L \perp \hat{Z}_i|1\rangle_L$ para $i = 1, 2, 3$. En otras palabras: los dos vectores lógicos $\{|0\rangle_L, |1\rangle_L\}$ y sus "descendientes erróneos" forman un conjunto de 8 vectores ortogonales que forman una base del espacio de Hilbert de 3 qubits. El espacio de Hilbert es descompuesto de esta manera en cuatro sub-espacios bidimensionales ortogonales (ver figura 3.8). Como consecuencia, existe un observable que podemos medir para determinar en cual de los 4 sub-espacios está el vector. Midiéndolo podemos ver qué error se produjo (ninguno, $\hat{Z}_1$, $\hat{Z}_2$ o $\hat{Z}_3$) y podemos recuperar el estado original.

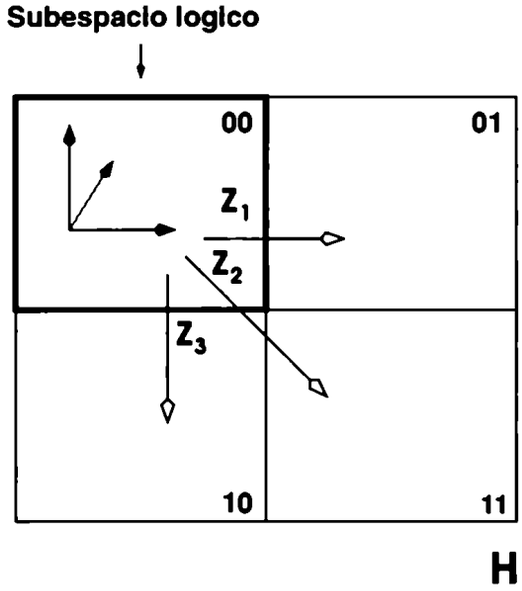


Figura 3.8: El espacio de Hilbert es dividido en cuatro sub-espacios ortogonales de dos dimensiones cada uno. Un sub-espacio está generado por los estados $|0\rangle_L$ y $|1\rangle_L$. Los otros se obtienen aplicando un operador $\hat{Z}_i$ sobre cada uno de los 3 portadores.

Para completar la descripción basta mostrar los observables que debemos medir para determinar el error. Para ello debemos mirar las simetrías que exhiben los estados lógicos de (3.21). Es claro que son auto-vectores de los siguientes operadores: $\hat{M}_1 \equiv \hat{X}_1\hat{X}_2$ y $\hat{M}_2 \equiv \hat{X}_2\hat{X}_3$ con auto-valor $+1$ (el estado $|0\rangle_L$ es superposición homogénea de estados que tienen un número par de unos mientras que $|1\rangle_L$ tiene sólo estados con un número impar de unos; por lo que son invariantes si invertimos dos bits simultáneamanete que es lo que hacen $\hat{M}_1$ y $\hat{M}_2$). También es fácil verificar que $\hat{M}_1$ y $\hat{M}_2$ son hermíticos, con auto-valores ± 1 (pues $\hat{M}_i^2 = 1$) y conmutan entre sí. También se puede ver que los descendientes erróneos del sub-espacio lógico son auto-vectores de $\hat{M}_i$. Por ejemplo, el sub-espacio $Z_1 H_L$ está formado por una combinación lineal de los vectores $\{\hat{Z}_1|0\rangle_L, \hat{Z}_1|1\rangle_L\}$ y son auto-estados de $\hat{M}_1$ y $\hat{M}_2$ con auto-valor -1. Esto es consecuencia de que el operador error Z_1 anticonmuta con $\hat{M}_1$ y $\hat{M}_2$ por lo que transforma auto-estados de $\hat{M}_i$ en otros auto-estados con distinto auto-valor (si $\hat{M}_i|\phi\rangle = |\phi\rangle$ entonces $\hat{M}_i\hat{Z}_i|\phi\rangle = -\hat{Z}_1|\phi\rangle$). Por lo tanto, si nuestro objetivo es saber a cual de los cuatro sub-espacios un estado pertenece tenemos

que medir $\hat{M}_1$ y $\hat{M}_2$. El resultado de la medición es un par de números que son ± 1 (los auto-valores de $\hat{M}_1$ y $\hat{M}_2$) y cada uno de los cuatro pares posibles (conocidos como síndromes) que identifican unívocamente cada uno de los cuatro sub-espacios: H_L corresponde al par $(+1,+1)$, $Z_1 H_L$ corresponde a $(-1,-1)$, $Z_2 H_L$ a $(-1,+1)$ y finalmente a $Z_3 H_L$ le corresponde $(+1,-1)$.

Es interesante pensar qué tipo de procedimiento físico debemos seguir para medir los observables $\hat{M}_i$. Como vimos, los operadores son productos de operadores de Pauli. Sin embargo, es muy importante medir estos observables **sin** medir individualmente los factores que aparecen en estos productos! En nuestro caso debemos medir el producto $\hat{M}_1 = \hat{X}_1 \hat{X}_2$ y $\hat{M}_2 = \hat{X}_2 \hat{X}_3$ pero no lo podemos hacer simplemente midiendo cada $\hat{X}_i$ por separado. Si hiciésemos eso estaríamos midiendo un conjunto completo de observables y haríamos que el sistema colapse en un estado. Para poder hacer corrección de errores debemos ser capaces de obtener suficiente información para determinar el sub-espacio pero no romper la coherencia que existe dentro del sub-espacio (¡que tiene nuestra información tan valiosa sobre q!). En otras palabras, queremos que la medición proyecte el estado en alguno de los sub-espacios bidimensionales y no que la colapse en un rayo del espacio.

Aunque pueda parecer complicado medir los operadores $\hat{M}_i$ en [Pringe97] Pringe muestra como diseñar sistemáticamente circuitos para medir conjuntos de operadores formados por productos de operadores de Pauli. Para mostrar que es posible hacerlo basta estudiar un poco el circuito de la figura 3.9. Si llamamos $\hat{D}$ al operador que representa ese circuito se puede verificar que $\hat{Z}_2 \hat{D} = \hat{D} \hat{M}_1$ (ver [Pringe97] para una técnica eficiente de verificar dicha igualdad). Esta identidad quiere decir que si $|\phi\rangle$ es un auto-vector de $\hat{M}_1$ con auto-valor m_1 entonces $\hat{D}|\phi\rangle$ es un auto-vector de $\hat{Z}_2$ con auto-valor m_1. Por lo tanto, si le aplicamos $\hat{D}$ a $|\phi\rangle$ y medimos el valor del segundo qubit (es decir, medimos $\hat{Z}_2$) entonces estamos en efecto midiendo $\hat{M}_1$. Análogamente, se puede verificar que $\hat{Z}_3 \hat{D} = \hat{D} \hat{M}_2$ y por lo tanto, midiendo el estado del tercer bit estamos midiendo $\hat{M}_2$.

Para recuperarse de un error debemos hacer una operación simple al qubit restante (el primero en nuestro caso). Este qubit contiene el estado cuántico a almacenar a menos de una transformación unitaria. Para averiguar como recuperarse del error debemos ver cual es el efecto del circuito sobre los errores mismos. Usando las técnicas de [Pringe97] es fácil verificar que: $\hat{Z}_1 \hat{X}_2 \hat{X}_3 \hat{D} = \hat{D} \hat{Z}_1$, $\hat{X}_2 \hat{D} = \hat{D} \hat{Z}_2$, $\hat{X}_3 \hat{D} = \hat{D} \hat{Z}_3$. Estas identidades tienen la siguiente interpretación. Veamos a modo de ejemplo la primera: si le aplicamos $\hat{Z}_1$ al estado codificado y luego lo pasamos por el circuito decodificador esto es equivalente a agarra el estado codificado, pasarlo por el decodificador (lo que dejaría los qubits 2 y 3 en el estado $|0\rangle$), luego invertir los dos qubits ($\hat{X}_1$, $\hat{X}_2$) y aplicarle $\hat{Z}$ al primero. Al hacer una medición sobre los qubits 2 y 3 obtendríamos 1 y en el qubit 1 el estado almacenado a menos de una operación $\hat{Z}$. Es decir, si nuestro sistema sufrió un $\hat{Z}_1$, al pasar por el circuito decodificador veríamos los bits 2 y 3 invertidos y sabemos que hubo un $\hat{Z}_1$ por lo que le aplicamos $\hat{Z}_1$ al primer qubit y re-obtenemos el estado original (sin saber cual es!). Análogamente, si ocurrió un error $\hat{Z}_2$ o un $\hat{Z}_3$ las identida-

des nos dicen que cambiaron los bits 2 y 3 pero no debemos hacer nada para recuperarlo pues el primero qubit no fue modificado.

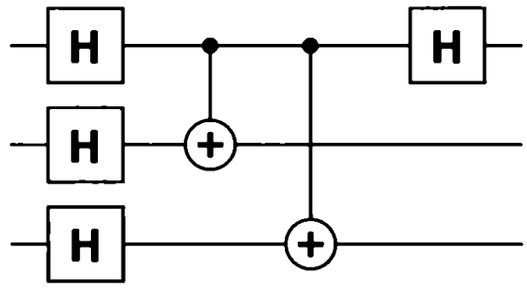


Figura 3.9: Circuito usado para medir los operadores $\hat{M}_1$ y $\hat{M}_2$. Midiendo el valor de $\hat{Z}_2$ y $\hat{Z}_3$ obtenemos los auto-valores de $\hat{M}_{1,2}$.

En resumen, el procedimiento de corrección de errores es el siguiente: (1) ponemos el primero qubit en el estado $\alpha|0\rangle + \beta|1\rangle$ y los bits 2 y 3 en $|0\rangle$. (2) Pasamos este estado por el circuito inverso a $\hat{D}$ para obtener el estado codificado $\alpha|0\rangle_L + \beta|1\rangle_L$. (3) dejamos que el sistema interactué por un tiempo en el cual puede ocurrir uno de los 4 posibles errores: $\mathbb{I}$ o $\hat{Z}_i$. (4) pasamos el estado por el circuito decodificador $\hat{D}$. (5) medimos los bits 2 y 3. Si obtenemos el autovalor -1 en ambos bits le aplicamos $\hat{Z}$ al primer bit. Para cualquier otro valor no hacemos nada. Después de este procedimiento hemos recuperado el estado original (sin nunca enterarnos cual era!). Si queremos almacenarlo por mas tiempo debemos re-inicializar los bits 2 y 3 y repetir el procedimiento. Este efecto de borrado es responsable de deshacerse de la entropía generada por los errores. En [Nielsen98] se discuten aspectos muy interesantes sobre la relación que hay entre la segunda ley de la termodinámica y el proceso de corrección de errores.

En el caso discutido hasta este momento consideramos lo que puede pasar si ocurre un único error. Una situación más realista debe incluir el efecto de más de un error. Cuando ocurren más errores el código ya no los puede corregir y por lo tanto el estado se degradará. Para calcular este efecto supongamos que un error $\hat{Z}$ ocurre con probabilidad p. Es fácil convencerse que si esto ocurre, la matriz densidad a la salida del canal es:

$$\begin{aligned} \hat{\rho}_{\text{out}} &= (1-p)^3\hat{\rho}_{\text{in}} + p(1-p)^2\sum_i \hat{Z}_i\hat{\rho}_{\text{in}}\hat{Z}_i \\ &+ p^2(1-p)\sum_{i\neq j}\hat{Z}_i\hat{Z}_j\hat{\rho}_{\text{in}}\hat{Z}_i\hat{Z}_j + p^3\hat{Z}_1\hat{Z}_2\hat{Z}_3\hat{\rho}_{\text{in}}\hat{Z}_1\hat{Z}_2\hat{Z}_3 \end{aligned} \tag{3.22}$$

Si ahora aplicamos corrección de errores a esta matriz densidad es claro que los primeros dos términos de dicha expresión serán proporcionales a $\hat{\rho}_{\text{in}}$. Este procedimiento, al igual que en el caso de códigos clásicos elimina el término proporcional a p. Si ahora calculamos la fidelidad del estado después de ser sometida al proceso de corrección de errores vemos que ahora es cuadrática en p. Aplicando la técnica de corrección de errores hemos disminuido la probabilidad de tener un error.

3.5.4. Un paréntesis histórico

En nuestro modelo simple hemos sólo permitido errores tipo $\hat{Z}$. Es claro que en la realidad aparecerán otros tipos de errores asociados a las matrices de Pauli $\hat{X}$ (inversiones) e $\hat{Y}$ (inversiones + cambio de fase). Afortunadamente es posible combatir estos tipos de errores (y también errores unitarios que se escriben como combinaciones lineales de la identidad y los tres operadores de Pauli) utilizando técnicas similares a la mostrada como ejemplo.

Quizás convenga hacer un paréntesis parar hablar en este punto un poco acerca de la historia de los códigos cuánticos de corrección. Como ya dijimos, Shor [Shor95] fue el primero en proponer un código de 9 qubits que protegía un bit lógico de todos los errores posibles ($\hat{X}$, $\hat{Y}$ y $\hat{Z}$). Para mas o menos la misma época A. Steane [Steane96a], [Steane96b] introdujo un código que protegía un qubit en 7 qubits. Sin embargo, se sospechaba que debía existir un código aun más chico que proteja un qubit contra cualquier tipo de errores. Un argumento para entender esto es el siguiente: debido a que los sub-espacios destinados a cada tipo de error deben ser ortogonales (si suponemos que el código es no-degenerado: es decir, dos errores **no** mandan a nuestro vector al mismo sub-espacio) el espacio de Hilbert tiene que ser lo suficientemente grande para contener a todos estos sub-espacios. Si tenemos 3 tipos de error por cada qubit, necesitamos en total $3n + 1$ sub-espacios bidimensionales, ortogonales (donde n es número de qubits del código y el 1 aparece para el sub-espacio de ningún error). Para que estos sub-espacios entren en un espacio de dimensión 2^n debe satisfacerse que:

$$2(3n + 1) \leq 2^n. \tag{3.23}$$

El mínimo n que satisface la desigualdad (de hecho la satura) es $n = 5$. Este concepto se puede generalizar: si queremos codificar k bits en n portadores y protegerlos contra errores en, como máximo t qubits simultáneamente, se debe satisfacer la siguiente desigualdad de la cual (3.23) es un caso particular:

$$2^k \sum_{l=0}^{t} 3^l \binom{n}{l} \leq 2^n \tag{3.24}$$

Ese mismo año nuestro grupo en colaboración con investigadores de Los Alamos encontró el código de 5 qubits [Laflamme95]. Más o menos de manera simultánea Bennett y DiVincezo, usando técnicas totalmente distintas encontraron un código también equivalente de 5 qubits. Para encontrar los estados lógicos del código de 5 bits propusimos como estados lógicos combinaciones lineales de 8 kets que obtuvimos del código de Steane de 7 bits omitiendo dos qubits. Los coeficientes de la expansión serían ± 1 lo que limita a 2^8 al número de estados lógicos posibles. Para determinar cual es la combinación adecuada escribimos un programa que recorría cada una de los posibles estados lógicos y verificaba si satisfacía las condiciones de ortogonalidad. Es decir, verificaba si los estados lógicos definidos con una dada combinación de ± 1, al aplicarle

cada uno de los posibles 15 errores era mapeado en un sub-espacio ortogonal a todos los demás. Sorprendentemente el programa rápidamente encontró numerosos estados que satisfacían estos criterios, uno de los cuales era:

$$\begin{aligned} |0\rangle_L \equiv & -|00000\rangle + |01111\rangle - |10011\rangle + |11100\rangle \\ & +|00110\rangle + |01001\rangle + |10101\rangle + |11010\rangle \\ |1\rangle_L \equiv & -|11111\rangle + |10000\rangle + |01100\rangle - |00011\rangle \\ & +|11001\rangle + |10110\rangle - |01010\rangle - |00101\rangle \end{aligned} \tag{3.25}$$

Es un poco aburrido pero trivial verificar que, si aplicamos los operadores $\hat{X}_i$, $\hat{Y}_i$ o $\hat{Z}_i$ a cada uno de los cinco qubits i obtendremos estados que son todos ortogonales. También en [Laflamme95] mostramos que es posible escribir un circuito para obtener éstos estados. El mismo tenía una propiedad que ninguno de los métodos de corrección de errores introducidos hasta el momento tenía: el circuito codificador y el circuito usado para detectar los errores eran uno el inverso del otro. Ahora sabemos que siempre podemos diseñar el circuito codificador de manera que su inverso sea el que reconoce los errores. En la figura 3.10 vemos el circuito codificador/decodificador.

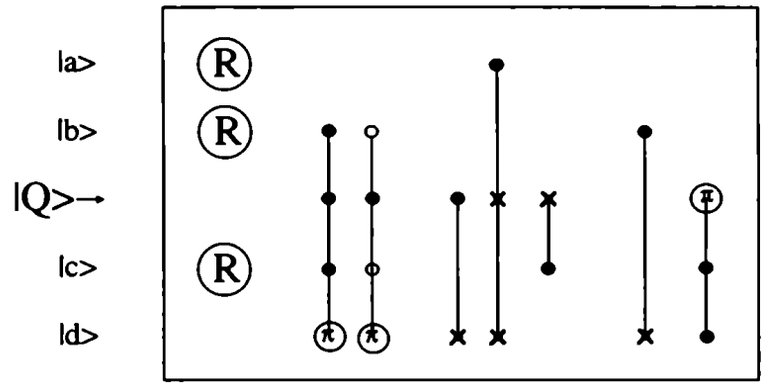


Figura 3.10: Circuito usados para generar los estados lógicos $|0\rangle_L$ y $|1\rangle_L$ del código de 5 bits. R es la operación que mapea $|0\rangle \rightarrow (|0\rangle + |1\rangle)/\sqrt{2}$ y $|1\rangle \rightarrow (|0\rangle - |1\rangle)/\sqrt{2}$. Los elementos con una $\times$ corresponden a un no controlado (donde el control es un círculo lleno); si el control es 1 el bit que tiene una $\times$ es invertido. Los elementos con π son operaciones que le agregan una fase -1 si los controles son 1 (o 0 si el control tiene un círculo hueco).

El circuito funciona de la siguiente manera: se ponen todos los qubits de entrada en el estado $|0\rangle$ salvo el $|Q\rangle$ donde almacenamos el estado a proteger (por ejemplo, $\alpha|0\rangle + \beta|1\rangle$). Al aplicarle dicho circuito obtenemos el estado $\alpha|0\rangle_L + \beta|0\rangle_L$. Si ahora le aplicamos cualquier operador de error a alguno de los 5 qubits (es decir, aplicamos $\hat{X}_i$, $\hat{Y}_i$ o $\hat{Z}_i$ a algún qubit) obtenemos el estado erróneo. Dicho estado se puede pasar ahora por el circuito inverso (que no es otra cosa que aplicar las mismas compuertas del circuito codificador en orden inverso debido a la reversibilidad de cada una de ellas). A la salida del circuito decodificador obtendremos un estado que tiene cuatro qubits con un

patrón de 1s y 0s que son el síndrome (es decir, me dicen cual de los 15 errores posibles se se le aplicó) y el quinto qubit $|Q'\rangle$ (en el medio) tiene un estado que es $\hat{U}_{\text{err}}(\alpha|0\rangle + \beta|1\rangle)$ donde $\hat{U}_{\text{err}}$ es un operador unitario conocido que depende del error. Para restaurar la información almacenada en el qubit lógico hay que medir el síndrome y aplicar $\hat{U}_{\text{err}}^{\dagger}$ a $|Q'\rangle$. La lista completa de síndromes y y operaciones de restauración están resumidas en la tabla 3.1.

Como vemos, la manera en la que llegamos al código de 5 qubits no es muy sistemática. Hasta este punto todos los códigos se obtenían de manera más bien artesanal: se entendía la idea de usar sub-espacios ortogonales y se utilizaban muchas ideas de códigos de corrección clásicos. Recién al año siguiente, Gottesman [Gottesman96] y Calderbank et. al. [Calderbank96] introdujeron el formalismo de estabilizadores que presentaremos a continuación y que hoy en día se usa continuamente. La idea principal era la de reconocer que lo que caracterizaba al código no eran los estados lógicos sino los operadores cuyos auto-estados eran los estados lógicos. Dichos operadores son lo que se conoce como el estabilizador del código.

Error	Síndrome $\lvert a'b'c'd'\rangle$	Estado erróneo $\lvert Q'\rangle$
$\mathbb{I}$	0000	$\alpha\lvert 0\rangle + \beta\lvert 1\rangle$
$\hat{Y}_3$	1101	$-\alpha\lvert 1\rangle + \beta\lvert 0\rangle$
$\hat{Y}_5$	1111	$-\alpha\lvert 0\rangle + \beta\lvert 1\rangle$
$\hat{X}_2$	0001	
$\hat{Z}_3$	1010	$\alpha\lvert 0\rangle - \beta\lvert 1\rangle$
$\hat{Z}_5$	1100	
$\hat{Y}_2$	0101	
$\hat{X}_5$	0011	
$\hat{Z}_1$	1000	$-\alpha\lvert 0\rangle - \beta\lvert 1\rangle$
$\hat{Z}_2$	0100	
$\hat{Z}_4$	0010	
$\hat{X}_1$	0110	
$\hat{X}_3$	0111	
$\hat{X}_4$	1011	$-\alpha\lvert 1\rangle - \beta\lvert 0\rangle$
$\hat{Y}_1$	1110	
$\hat{Y}_4$	1001	

Tabla 3.1: Tabla de sindromes y las correspondientes operaciones que hay que hacer para restaurar la información almacenada para el código perfecto de 5 qubits.

3.5.5. Formalismo de estabilizadores

Presentaremos aquí el formalismo de estabilizadores que describe una amplia gama de familias de códigos de corrección como el presentado en el capítulo anterior. Seguiremos a [Paz98] en la presentación. Vamos a considerar códigos que protegen k qubits codificándolos en n portadores. El espacio lógico $\mathcal{H}_k$ es un sub-espacio de dimensión 2^k del espacio total de Hilbert de dimensión 2^n. $\mathcal{H}_n$ es un producto tensorial de n espacios bidimensionales y tiene como base natural a productos tensoriales de cada uno de los n portadores de información. Esta es la llamada "base física" que se puede formar por los auto-estados de los operadores $\{\hat{Z}_1, \ldots, \hat{Z}_n\}$ (por conveniencia vamos a etiquetar a la base no por los auto-valores de los operadores correspondientes que son ± 1 sino por los auto-valores de los proyectores sobre el sub-espacio -1 que son 0 o 1: es decir, la etiqueta $z_j = 0$ ($z_j = 1$) corresponde al auto-valor $+1$ (-1) del operador $\hat{Z}_j$). Por convención ordenaremos los n portadores de manera que los últimos k qubits tienen el estado que queremos codificar y los primeros $n - k$ son utilizados para los síndromes. Con esta convención, los estados de la base física son de la forma $|s, z\rangle_P = |s\rangle_P \otimes |z\rangle_P$ (donde las cadenas $s = (s_1, \ldots, s_{n-k})$, $z = (z_1, \ldots, z_k)$ guardan el correspondiente auto-valor y el subíndice P identifica la base física).

Un código de corrección es un mapeo del estado $|0\rangle_P \otimes |\psi\rangle_P$ sobre el espacio $\mathcal{H}_k$ que está formado por estados entrelazados de los n portadores. Una clase general de códigos es descripto en términos del grupo estabilizador. El estabilizador de un código es un grupo Abeliano formado por todos los operadores que son productos tensoriales de matrices de Pauli que tienen a $\mathcal{H}_k$ como espacio invariante con auto-valor $+1$. Cada elemento del estabilizador, que es un grupo finito con 2^{n-l} elementos, se puede obtener multiplicando apropiadamente $n - k$ generadores que denotaremos por $\hat{M}_1, \ldots, \hat{M}_{n-k}$. Los elementos del estabilizador están completamente degenerados el es espacio $\mathcal{H}_k$ (ya que todos los estados pertenecientes a $\mathcal{H}_k$ son auto-estados con auto-valor $+1$ de todos los $\hat{M}_j$). Para definir una base en el espacio del código elegimos k operadores mas $\hat{L}_1, \ldots, \hat{L}_k$, formados por productos tensoriales de matrices de Pauli que conmutan con todos los elementos del estabilizador. Estos operadores, $\hat{L}_i$, $i = 1, \ldots, k$ son los "punteros lógicos" porque definen las direcciones dentro de $\mathcal{H}_k$ asociadas con cada uno de los estados $|0\rangle_L, \ldots, |2^k - 1\rangle_L$ (los punteros lógicos perteneces al grupo de operadores que conmutan con el estabilizador conocidos como el "normalizador").

Los $n - k$ generadores del estabilizador junto con los k punteros lógicos forman un Conjunto Completo de Operadores que Conmutadores (CCOC) cuyos auto-estados forman una base completa del espacio de Hilbert $\mathcal{H}_n$. Los elementos de la "base lógica", están etiquetados por n números cuánticos y los denotaremos con $|m, l\rangle_L$ donde las cadenas de bits $m = (m_1, \ldots, m_{n-k})$ y $l = (l_1, \ldots, l_k)$ identifican los auto-valores correspondientes y el subíndice L denota la base lógica. El CCOC formado por los generadores del estabilizador y los punteros lógicos define una prescripción para descomponer el espacio de Hilbert original de los n portadores en un producto tensoriales de un sub-espacio lógico $\mathcal{L}$ de dimensión 2^k y un sub-espacio del síndrome $\mathcal{Y}$ de dimensión

2^{n-k}. Los elementos de la base lógica (que son estados entrelazados de los n portadores) son productos tensoriales de estados pertenecientes a $\mathcal{L}$ y $\mathcal{Y}$: $|m,l\rangle_L = |m\rangle_L \otimes |l\rangle_L$. Los estados codificados, que pertenecen a $\mathcal{H}_k$ son también estados productos de la forma $|\psi\rangle = |0\rangle_L \otimes \sum_l c_l |l\rangle_L$.

El código protege estados cuánticos frente a errores $\hat{E}_a$ cuya acción sobre los estados lógicos es cambiar el síndrome y, eventualmente, rotar el estado lógico en $\mathcal{L}$ de una manera que depende del síndrome:

$$\hat{E}_a |m\rangle_L \otimes |l\rangle_L = e^{i\Phi_{ma}} |m + c_a\rangle_L \otimes \hat{U}_a |l\rangle_L. \tag{3.26}$$

Aquí, $\hat{U}_a$ es un operador unitario que actúa en el espacio lógico $\mathcal{L}$, y Φ_{ma} son fases que pueden depender del síndrome y del error. El error $\hat{E}_a$ cambia el síndrome de m a $m + c_a$ donde c_a es una cadena de bits que almacena el patrón de conmutación entre el error y los generadores del estabilizador (el bit i-ésimo de la cadena es 1 si el error anticonmuta con $\hat{M}_i$ o de otra manera 0). La razón de esto es que cuando el error $\hat{E}_a$ actúa sobre un estado lógico cambia el auto-valor del operador $\hat{M}_i$ sólo si $\{\hat{M}_i, \hat{E}_a\} = 0$. La etiqueta a se usa para identificar el error, es arbitraria y para el caso de códigos no-degenerados (que son los que discutiremos aquí) siempre es posible etiquetar los errores por el patrón de conmutación c_a (es decir, podemos siempre elegir $a = c_a$).

Para corregir la acción de cualquier error $\hat{E}_a$ (o cualquier combinación lineal de ellos) uno puede primero detectarlo midiendo el síndrome (es decir, medir los observables $\hat{M}_j$, $j = 1, \ldots, n-k$) y luego recuperar el estado original aplicando $\hat{U}_a^\dagger$. Este proceso de detección-corrección puede ser convenientemente descripto con el formalismo de operaciones cuánticas (el mismo usado para describir la interacción entorno-computadora en la sección 3.2.2) de manera de mapear el estado erróneo $\hat{\rho}_{\text{in}}$ al estado corregido $\hat{\rho}_{\text{out}}$:

$$\hat{\rho}_{\text{out}} = \sum_{m=0}^{N} \hat{R}_m \, \hat{\rho}_{\text{in}} \, \hat{R}_m, \tag{3.27}$$

donde la suma es sobre todos los síndromes posibles ($N = 2^{n-k} - 1$) y los operadores de recuperación para cada síndrome son:

$$\hat{R}_m = |0\rangle_L \langle m|_L \otimes \hat{U}_m^\dagger. \tag{3.28}$$

Por construcción los operadores de recuperación satisfacen la identidad $\sum_{m=0}^{N} \hat{R}_m^\dagger \hat{R}_m = \mathbb{I}$.

Debido a que en nuestra descripción el proceso de detección-recuperación está formulado enteramente en la base lógica, todavía no hace referencia alguna a las operaciones de codificación y decodificación que pueden ser simplemente definidas como un cambio de base. El operador de codificación $\hat{C}$ es un operador unitario que mapea la base física de n portadores a la base lógica formada

por estados entrelazados. La transformación $\hat{C}$ transforma operadores $\hat{Z}_i$ (cuyos auto-valores definen estados de la base lógica) en operadores $\hat{M}_j$, $\hat{L}_{j'}$ (que etiqueta estados de la base lógica). Es decir, el operador de codificación $\hat{C}$ es tal que $\hat{Z}_j = \hat{C}^\dagger \hat{M}_j \hat{C}$, $j = 1, \ldots, n-k$ y $\hat{Z}_{n-k+j'} = \hat{C}^\dagger \hat{L}_{j'} \hat{C}$, $j' = 1, \ldots, k$. Tomando esto en cuenta, la acción del operador $\hat{R}_m$ se puede describir en la base física con la siguiente secuencia de operaciones: (i) decodificar el estado, (ii) medir el síndrome en la base física midiendo $\hat{Z}_j$ en los primeros $n-k$ portadores, (iii) si el resultado de la medición es la cadena s, aplicar el operador de recuperación $\hat{U}_s^\dagger$ y re-inicializar el síndrome nuevamente a cero y (iv) codificar nuevamente el estado resultante.

3.5.6. Códigos de corrección continua

El proceso de corrección de errores es, por naturaleza discreto: codifico mi estado, lo dejo en contacto con el entorno por un tiempo, hago una corrección para recuperar el estado, codifico, etc. Una pregunta que se puede hacer es: ¿Que pasa en el límite en el cual las operaciones de corrección se hacen instantáneamente? Aquí presentaremos algunos resultados mostrando lo que sucede en el límite continuo del proceso de corrección de errores. Para ello, Paz y Zurek [Paz98] proponen la siguiente ecuación que describe el proceso de corrección de errores en el límite continuo:

$$\dot{\hat{\rho}} = \underbrace{-i[\hat{H}_0, \hat{\rho}]}_{\text{evolución libre}} + \underbrace{\sum_{a=1}^{N} \gamma'_a \left(\hat{E}_a^\dagger \hat{\rho} \hat{E}_a - \frac{1}{2} \hat{E}_a^\dagger \hat{E}_a \hat{\rho} - \frac{1}{2} \hat{\rho} \hat{E}_a^\dagger \hat{E}_a \right)}_{\text{efecto de errores}} + \underbrace{\sum_{m=0}^{N} \gamma_m \left(\hat{R}_m^\dagger \hat{\rho} \hat{R}_m - \frac{1}{2} \hat{R}_m^\dagger \hat{R}_m \hat{\rho} - \frac{1}{2} \hat{\rho} \hat{R}_m^\dagger \hat{R}_m \right)}_{\text{proceso de corrección}} \tag{3.29}$$

La evolución de $\hat{\rho}$ está dada por 3 términos: el primero describe alguna evolución de nuestro qubit. El qubit está evolucionando porque externamente estoy manipulando mi estado. Además de la dinámica externa debo incluir el efecto de los errores que están continuamente intentando "arruinar" mi estado. Los coeficiente γ_m son las 'constante de decaimiento' de cada tipo de error. Dan una idea de la velocidad con la cual se están generando errores de tipo m. Finalmente el efecto de la corrección continua está dado por el último término de la ecuación. Nuevamente tenemos coeficientes γ'_a que dan idea de que tan rápido estamos arreglando nuestro sistema.

Si usamos el formalismo de estabilizadores podemos escribir la matriz densidad en la base lógica:

$$\hat{\rho} = \sum_{m,m'} |m\rangle\langle m'| \otimes \hat{\rho}_{mm'}$$

Como vamos a estar interesados en estudiar que pasa en cada uno de los sub-espacios correspondientes a cada síndrome vamos a intentar encontrar una ecuación para $\hat{\rho}_m \equiv \hat{\rho}_{mm}$. Para poder analizar (y resolver) las ecuaciones resulta útil expresar la ecuación de evolución usando la representación de Bloch para la matriz densidad (ver caja aparte):

$$\hat{\rho}_m = \frac{1}{2}\left(p_m \hat{\mathbb{I}} + \vec{r}_m \cdot \hat{\vec{\sigma}}\right)$$

Si suponemos que la evolución del sistema es de la forma: $\hat{H}_0 = \sum_{m=1}^{N} |m\rangle\langle m| \otimes \hat{h}_m$ con $\hat{h}_m = \Omega\hat{\sigma}_k/2$ (que representa una rotación de los estados lógicos entorno al eje k) obtenemos la siguiente ecuación para $\hat{r}_m$:

$$\dot{\vec{r}}_m = \vec{\Omega} \times \vec{r}_m - \left[\gamma(1-\delta_{m0}) + \sum_{a=1}^{N} \gamma'_a\right] \vec{r}_m + \sum_{a=1}^{N} \mathbf{\Lambda}_a(\gamma\delta_{m0}\vec{r}_a + \gamma'_a \vec{r}_{m\oplus a}) \quad (3.30)$$

Con $(\mathbf{\Lambda}_a)_{i,j} \equiv \frac{1}{2}\delta_{i,j}\mathrm{Tr}\left(\hat{\sigma}_j \hat{U}_a\right)^2$.

Dicha ecuación describe la evolución del sistema y es muy general: es aplicable a todos los códigos de corrección que admiten descripción en términos de estabilizadores. Una vez que conocemos el código (su estabilizador, patrones de conmutación, operadores de recuperación, etc) es posible resolverla y analizar el límite continuo.

Para estudiar las características de la ecuación (3.30) la resolvimos para el código de 3 bits usado a lo largo del capítulo y nuestro código de 5 bits. Para poder resolver (3.30) necesitamos el estabilizador del código que queremos resolver. El estabilizador del código de 3 bits ya fue presentado. El de 5 bits es:

$$\begin{aligned}
\hat{M}_1 &= \hat{Z}_1\hat{Z}_2\hat{Z}_4\hat{Z}_5 \\
\hat{M}_2 &= \hat{X}_1\hat{X}_2\hat{X}_3\hat{Z}_5 \\
\hat{M}_3 &= \hat{Z}_1\hat{X}_2\hat{Z}_3\hat{X}_4 \\
\hat{M}_4 &= \hat{Y}_1\hat{Z}_2\hat{Z}_3\hat{Y}_5 \\
\hat{L} &= \hat{Z}_1\hat{Z}_3\hat{Z}_5
\end{aligned}$$

SOLUCIÓN FORMAL DE LA ECUACIÓN MAESTRA

La ecuación maestra (3.30) tiene la siguiente forma general:

$$\dot{\rho} = -i\,(H_{\text{eff}}\rho - \rho H_{\text{eff}}) + \gamma \sum_m a_m \rho\, a_m$$

con $H_{\text{eff}} \equiv H_0 - i\frac{1}{2}\gamma \sum_m a_m^\dagger a_m$. Si pasamos a la representación interacción: $\rho_{\text{int}} = U_0^\dagger \rho U_0$ con $U_0(t) \equiv \exp(-iH_{\text{eff}}t)$, podemos reescribir la ecuación anterior de la siguiente manera:

$$\dot{\rho}_{\text{int}} = \gamma \sum_m a_m^\dagger(t)\rho_{\text{int}} a_m(t) \qquad \text{donde} \quad a_m(t) = U_0^\dagger(t)\, a_m\, U_0(t)$$

cuya solución formal se obtiene expandiendo ρ en una serie de potencias de γ:

$$\rho_{\text{int}}(t) = \rho_{\text{int}}(0) + \sum_{n=1}^{\infty} \gamma^n \rho_{\text{int}}^{(n)} \tag{3.31}$$

y resulta que:

$$\rho_{\text{int}}^{(n)} = \sum_{m_1 \ldots m_n} \int_0^t dt_n \int_0^{t_n} dt_{n-1} \ldots \int_0^{t_2} dt_1 a_{m_n}^\dagger(t_n) \ldots a_{m_1}^\dagger(t_1) \rho_0^{\text{int}} a_{m_1}(t_1) \ldots a_{m_n}(t_n).$$

Si ahora volvemos de la representación interacción usando que $\rho(t) = U_0(t)\rho_{\text{int}}U_0^\dagger(t)$ obtenemos la siguiente solución formal si el estado inicial es $\rho_0 = |\psi_0\rangle\langle\psi_0|$:

$$\begin{aligned} \rho(t) &= |\psi_0\rangle\langle\psi_0| + \\ &+ \sum_{n=1}^{\infty} \gamma^n \sum_{m_1 \ldots m_n} \int_0^t dt_n \int_0^{t_n} dt_{n-1} \ldots \int_0^{t_2} dt_1 |\psi(t, t_n, \ldots, t_1)\rangle\langle(t, t_n, \ldots, t_1)| \end{aligned} \tag{3.32}$$

donde $|\psi(t, t_n \ldots t_0)\rangle \equiv U_0(t - t_n) a_{m_n}^\dagger U_0(t_n - t_{n-1})\, a_{m_{n-1}}^\dagger \ldots a_{m_1}^\dagger\, U_0(t_1)|\psi(0)\rangle$ En el texto de discute la interpretación de dicha solución formal.

Dado el estabilizador podemos calcular los patrones de conmutación con los distintos errores $\hat{E}_a$ (para poder saber quienes son $m \oplus a$ en (3.30)) y los operadores de recuperación $\hat{U}_a$.

En la parte superior del gráfico de la figura 3.11 vemos la fidelidad como función del tiempo para el proceso de corrección continua para distintos cocientes de γ/γ'. Como es de esperar la fidelidad decae (por mas corrección de errores que hagamos) al valor asintótico 1/2. Abajo vemos las componente y y z del vector $\vec{r}(t)$. Si no hubiese errores el vector rotaría en el plano y-z. Debido al efecto de los errores se puede observar que la rotación está amortiguada.

La comparación de los dos códigos muestra que en el de 5 bits la fidelidad decae más rápidamente para los mismos cocientes γ/γ'. Intuitivamente podemos esperar eso porque hay muchos mas tipos de errores que pueden aparecer. Esta impresión se puede corroborar si sacamos errores de la ecuación maestra y comparamos. Vimos que, cuanto menos errores posibles afectan a nuestro estado mas lentamente cae la fidelidad.

Uno puede preguntarse porque, si estamos corrigiendo continuamente la fidelidad, inevitablemente cae a 1/2. La razón fundamental es que la corrección de errores no garantiza que podamos almacenar la información por un tiempo indeterminado. Los códigos que presentamos aquí están diseñados suponien-

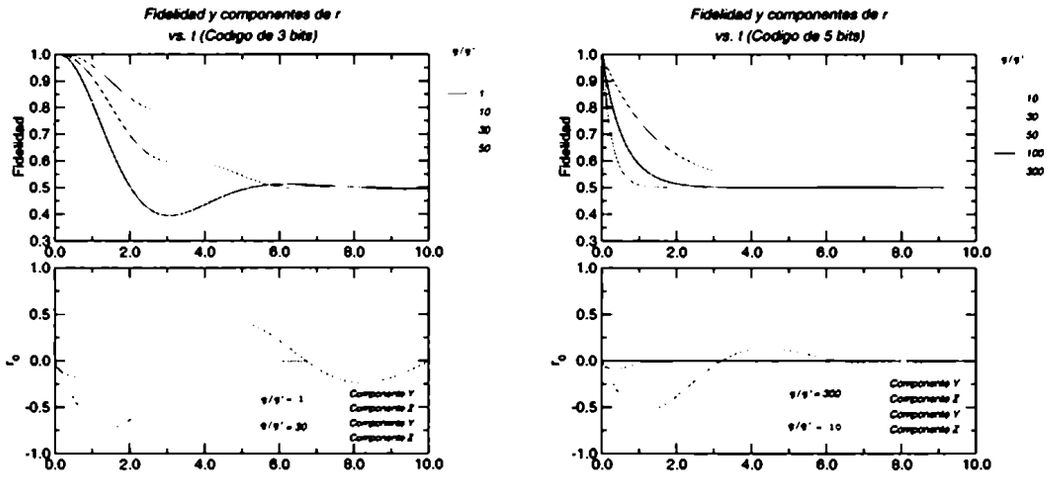
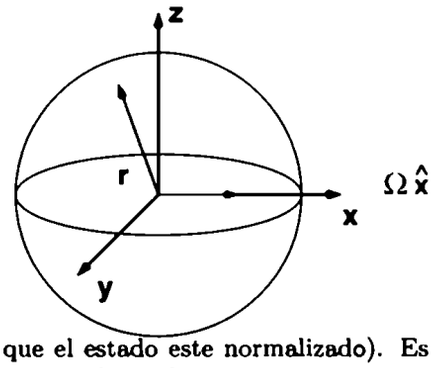


Figura 3.11: Ejemplo de corrección continua para los códigos de 3 y 5 bits. El gráfico superior muestra la fidelidad para distintos parámetros de γ/γ'. El inferior muestra como el efecto del error amortigua las rotaciones de $\vec{r}(t)$

do que sólo hay un error antes de cada proceso de corrección. Sin embargo esto no ocurre en el caso que estamos estudiando. La forma de verlo es analizando la solución formal de la ecuación maestra 3.30. En la caja se muestra la solución para una ecuación maestra que tiene la misma estructura que la 3.30. Allí se tomaron todos los γ_i y γ'_j iguales a γ y tanto los operadores de error como de recuperación se los denota como $\hat{a}_m$. El Hamiltoniano efectivo $H_{\text{eff}} = H_0 - i\frac{1}{2}c\gamma$ para alguna constante c debido a la propiedad de completitud tanto de los errores como de los operadores de recuperación. La solución formal a la ecuación maestra es una suerte de promedio de varias matrices densidades $|\psi(t_n \ldots t_1)\rangle\langle\psi(t_n \ldots t_1)|$ multiplicada por potencias de γ. Los vectores $|\psi(t_n \ldots t_1)\rangle$ se obtienen propagando con el Hamiltoniano efectivo H_{eff} y aplicando operadores a_m que representan tanto errores como operaciones de recuperación. Es decir: partiendo del estado $|\psi(0)\rangle$ lo propagamos por un tiempo $t_1 - t_0$ con el Hamiltoniano **no unitario** H_{eff}. Luego le aplicamos alguno de los operadores $\hat{a}_m$ que puede ser tanto un error como una operación de recuperación. Luego lo propagamos por un tiempo $t_2 - t_1$ con H_{eff}, le aplicamos nuevamente alguno de los $\hat{a}_m$, y así siguiendo. Al final de este proceso tendremos uno de los vectores $|\psi(t_n \ldots t_1)\rangle$. La matriz densidad final es un promedio sobre todos las posibles realizaciones de ese proceso. Es claro, que en la solución formal siempre aparecerán términos que tienen más de un error entre operaciones de corrección por lo que el proceso de corrección de errores no es siempre perfecto. Por este motivo, por mas corrección inevitablemente terminamos en el estado con fidelidad 1/2. Sin embargo no hay que desesperanzase: podremos hacer computación cuántica si logramos hacer nuestras operaciones en tiempo cortos comparados con los tiempos de decaimiento del estado.

REPRESENTACIÓN DE BLOCH



Muchas veces para describir el estado de un qubit conviene mirar lo que se conoce como "representación de Bloch". Cualquier matriz densidad de un sistema de dos estados puede ser escrita como combinación de la identidad y las matrices de Pauli:

$$\hat{\rho} = \frac{1}{2}\left(p\mathbb{I} + \vec{r}\cdot\vec{\hat{\sigma}}\right).$$

Como las matrices de Pauli tienen traza nula, debe ser que $p = \mathrm{Tr}\hat{\rho} = 1$ (siempre que el estado este normalizado). Es fácil demostrar que el módulo del vector $\vec{r}$ es 1 si el estado es puro y menor que 1 si es mixto. Es decir: todos los estados físicamente válidos están descriptos por un vector $\vec{r}$ que está dentro de la llamada "esfera de Bloch". Los estados puros forman la cascara de dicha esfera mientras que el interior esta poblado por estados mixtos.
El polo norte (sur) de la esfera de Bloch corresponde a los auto-vectores de la matriz de Pauli $\hat{Z}$ con auto-valor $+1$ (-1). Cuando $\hat{r}$ esta sobre el eje x el estados es un auto-vector de $\hat{X}$ y lo mismo cuando está sobre el eje y (además debe ser que $|\vec{r}| = 1$).

3.6. Computación cuántica tolerante a fallas

En todo lo discutido hasta el momento hemos hecho numerosas suposiciones que, en la práctica, rara vez se cumplen: (i) las operaciones de codificar mi estado y decodificarlo se hacen perfectamente. (ii) podemos preparar estados $|0\rangle$ con perfecta precisión. Es claro que en la vida real estas suposiciones son exageradas por lo que pueden ser un obstáculo importante. Afortunadamente el problema fue planteado y resuelto por numerosos investigadores. Shor nuevamente [Shor96a] (e independientemente Kitaev y Knill et. al [Knill98a]) mostraron que es posible hacer computación cuántica sobre estados codificados. Es decir: no hace falta codificar-decodificar para hacer operaciones lógicas. Shor además logró mostrar que, si el efecto del ruido de la memoria cuántica y de las operaciones es estocástica con probabilidad de error del orden de $O(\log^{-\alpha} n)$ (donde n es el número total de operaciones con ruido necesaria para implementar el algoritmo y α una constante positiva), se podía implementar dicho algoritmo con probabilidad **arbitrariamente** cerca a 1 (el error se puede achicar todo lo que uno quiere). Este resultado era alentador: acotando el error podemos lograr la precisión que queramos. Sin embargo habían algunos problemas por resolver: a medida que n crece necesitamos cada vez más precisión en nuestras compuertas fundamentales. Por suerte este problema también fue resuelto independientemente por Knill y Laflamme [Knill98a], Aharonov y Ben-Or [Aharanov96] y Kitaev [Kitaev96] que mostraron que era posible implementar cualquier algoritmo cuántico si el error por compuerta cuántica era mas chico que una determinado umbral. Luego otros investigadores mejoraron los umbrales [Gottesman98]. En [Preskill98] hay una excelente descripción de los métodos usados.

3.7. Métodos alternativos

Antes de terminar el capítulo es interesante comentar algunas ideas sobre posibles implementaciones de sistemas que se auto-corrigen. Las propuestas que discutiremos son dos métodos en los cuales el proceso de corrección es pasivo en contraposición con los códigos de corrección que pueden ser considerados 'activos'. La idea de estos métodos es que el sistema sea tolerante a errores automáticamente y no que haya que controlarlo y corregirlo constantemente. Las dos propuestas que discutiremos se conocen como espacios libres de decoherencia y códigos tóricos.

En el caso más general uno no tiene ninguna información acerca de los errores que se generan ni el tipo de interacción que hay entre la computadora y el entorno. Afortunadamente en muchos casos esta suposición es demasiado estricta y la información adicional puede resultar útil para proteger la información almacenada en la computadora. Esta es la filosofía de los espacio libres de decoherencia [Zanardi97, Duan98, Lidar98]. En la descripción subsiguiente nos basaremos en la teoría propuesta en [Lidar98]. Supongamos que la interacción computadora-entorno es de la forma:

$$\hat{H}_{\mathrm{CE}} = \sum_{\lambda} \hat{F}_{\lambda} \otimes \hat{B}_{\lambda} \tag{3.33}$$

donde $\hat{F}_{\lambda}$ son operadores que actúan en el espacio de la estados de la computadora y $\hat{B}_{\lambda}$ operan en el entorno. En algunos casos es posible hallar un conjunto de vectores $\{|\Psi\rangle\}$ que satisfacen:

$$\hat{F}_{\lambda}|\Psi\rangle = c_{\lambda}|\Psi\rangle \tag{3.34}$$

es decir, son auto-estados degenerados de todos los operadores que actúan sobre la computadora (los $\hat{F}_{\lambda}$). El sub-espacio generado por los $\{|\Psi\rangle\}$ se lo conoce como *espacio libre de decoherencia* y tiene la siguiente propiedad: si empezamos con un estado de la computadora que está dentro de este sub-espacio, la interacción con el entorno no lo sacará de allí. Si utilizamos ese sub-espacio para almacenar información nos garantizamos que el entorno no la afectará. Aunque pueda parecer un poco ingenuo pensar que vamos a conocer la interacción con el entorno, existe al menos un caso bastante plausible en el cual esta teoría puede resultar útil. Supongamos que el entorno actúa de manera indistinta en cualquiera de un grupo de K qubits. Es decir: la interacción computadora-entorno es simétrica en los qubits. Si además suponemos que los operadores $\hat{F}_{\lambda}$ son los operadores de Pauli $\hat{\sigma}_i$ entonces la interacción computadora-entorno puede ser descripta por el siguiente Hamiltoniano:

$$\hat{H}'_{\mathrm{CE}} = \sum_{i=x,y,z} \hat{S}_i \otimes \hat{B}_i \tag{3.35}$$

donde $\hat{S}_i \equiv \sum_{\lambda}^{K} \sigma_i^{\lambda}$ son los operadores de espín total en la dirección $i = \{x, y, z\}$. Es decir: en el espacio del sistema la interacción es proporcional al

espín total en cada una de las tres direcciones. De nuestro conocimiento de la estructura del grupo SU(2) conocemos un sub-espacio que cumple con la propiedad de ser invariante ante la acción de $\hat{S}_i$: el sub-espacio de espín total $S = 0$. Es sabido que los auto-estados del espín total son:

$$\hat{S}^2|S,m\rangle = S(S+1)|S,m\rangle \qquad \hat{S}_z|S,m\rangle = m|S,m\rangle$$

con $m = -S, -S+1, \ldots, 0, \ldots, S$ y $S = 0, 1, \ldots K/2$ para K par y $S = 1/2, 3/2, \ldots, K/2$ para K impar. La degeneración de los estados de espín total S es

$$\frac{K!(2S+1)}{(K/2+S+1)!(K/2-S)!}.$$

Como los operadores $\hat{S}_x$ y $\hat{S}_y$ se puede escribir como combinación lineal de $\hat{S}_\pm \equiv \hat{S}_x \pm \hat{S}_y$ y además $\hat{S}_\pm|0,0\rangle = \hat{S}_z|0,0\rangle = 0$ es claro que los vectores que pertenecen al sub-espacio de $S = 0$ (que tiene dimensión $K!/[(K/2+1)!(K/2)!]$) cumplen con la propiedad 3.34 (con coeficientes $c_\lambda = 0$) por lo que generan un espacio libre de decoherencia. Para $K = 2$ (es decir: dos qubits) el sub-espacio corresponde al estado singlete y tiene dimensión 1 por lo no resulta demasiado útil. Pero para $K = 4$ ya el sub-espacio tiene dimensión 2 y puede ser utilizado como qubit. Finalmente es interesante ver que en limite de K grande la dimensión del espacio con $S = 0$ va como $2^{K-\frac{3}{2}\log_2 K}$ por lo que la eficiencia de este método tiende a 1 (donde por eficiencia entendemos número de qubits codificados sobre número de qubits utilizados para codificar).

La técnica de espacios libres de decoherencia puede ser combinada con los métodos convencionales de corrección de errores como se muestra en [Lidar99] y también hay propuestas para hacer computación cuántica tolerante a fallas [Bacon00] utilizando éste concepto.

La otra idea interesante es la llamada 'computación cuántica topológica' y fue propuesta por Kitaev en [Kitaev97] (para una explicación más superficial se puede leer [Preskill97]). Aquí nos limitaremos a dar una breve descripción de algunas las ideas presentadas en ese trabajo.

El punto de partida es una construcción geométrica del estabilizador de un código de corrección de errores. Consideremos una grilla cuadrada de $k \times k$ sitios con condiciones periódicas de contorno. Si pensamos que en cada arista de la grilla hay un qubit podemos definir los siguientes operadores para cada vértice s y cada cara p (ver figura 3.12):

$$\hat{A}_s = \prod_{j \in cruz(s)} \hat{\sigma}_x^j \qquad \hat{B}_p = \prod_{j \in borde(p)} \hat{\sigma}_z^j \tag{3.36}$$

Cada uno de estos operadores conmuta entre sí pues la $cruz(s)$ y el $borde(s)$ tienen 0 o 2 aristas en común (y por ende cero o dos operadores de Pauli que anticonmutan). Los operadores $\hat{A}_s$ y $\hat{B}_p$ son hermíticos y tienen auto-valores

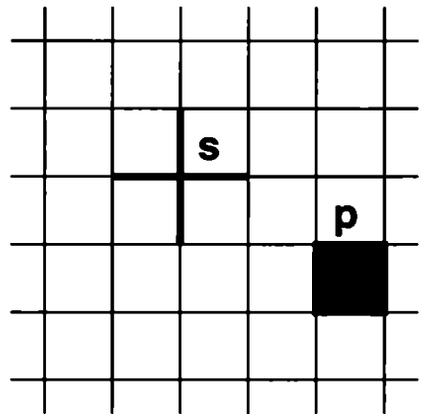


Figura 3.12: En cada arista de la grilla hay un espín. En la figura se ven las cuatro aristas que forman la $cruz(s)$ y el $borde(p)$. La grilla tiene condiciones periódicas de contorno.

± 1. Llamaremos a $\mathcal{N}$ al espacio de Hilbert de todos los $n = 2k^2$ qubits y definiremos $\mathcal{L} \subseteq \mathcal{N}$ como el siguiente sub-espacio:

$$\mathcal{L} = \left\{ |\xi\rangle \in \mathcal{N} : \hat{A}_s|\xi\rangle = |\xi\rangle, \hat{B}_p|\xi\rangle = |\xi\rangle \text{ para todo } s, p \right\}$$

Para el caso de una grilla cuadrada (un toro) la dimensión del sub-espacio es 4 pero es posible considerar otras geometrías que tendrán distinta dimensión. La propiedad interesante de dicho sub-espacio es que es posible corregir hasta $\lfloor (k-1)/2 \rfloor$ errores del tipo

$$\hat{E}(\alpha_1, \ldots, \alpha_n; \beta_1, \ldots, \beta_n) = \prod_j (\sigma_x^j)^{\alpha_j} \prod_k (\sigma_z^k)^{\beta_k} \qquad (\alpha_j, \beta_j = 0, 1)$$

Es decir, los operadores $\hat{A}_s$ y $\hat{B}_p$ son el estabilizador de un código que corrige hasta $\lfloor (k-1)/2 \rfloor$ errores. Esta construcción tiene varias ventajas: 1) cada estabilizador está formado por un número acotado de qubits (4 como máximo), 2) Cada qubit está involucrado en un número acotado de estabilizadores (4 como máximo), y 3) no hay límite en el número de errores que se pueden corregir (agrandando la grilla). Pero probablemente la idea más interesante de la propuesta de Kitaev es la siguiente: supongamos que de alguna manera logramos construirnos el siguiente Hamiltoniano:

$$\hat{H}_0 = -\sum_s \hat{A}_s - \sum_p \hat{B}_p.$$

Es fácil diagonalizar ese operador y rápidamente llegamos a la conclusión que el estado fundamental de dicho sistema no es otro que el sub-espacio protegido! Si la computadora está en contacto con un baño térmico a bajas temperaturas el sistema tenderá al estado de más baja energía. De esta manera, la computadora naturalmente tiende al estado protegido. Errores en los

qubits excitan al sistema del estado fundamental a otros niveles pero el mismo proceso de relajación se encarga de corregir los errores de manera automática!

Si bien esta técnica de protección sería muy útil no es fácil encontrar un sistema descripto por el Hamiltoniano H_0 debido a que involucra interacciones de 4 cuerpos. No obstante la idea de buscar sistemas cuyo estado fundamental sea un sub-espacio protegido es interesante. En sus trabajos, Kitaev también muestra que no sólo es posible almacenar información en este sistema sino que también procesarla de una manera tolerante a fallas.

3.8. Resumen

En este capítulo hemos estudiado los efectos del ruido (unitario o no-unitario) sobre la computación cuántica. Nuestro trabajo [Miquel97] fue pionero en este área. Allí presentamos la primera simulación completa de la factorización para una computadora cuántica relativamente grande. Aunque un procesador cuántico con 18 iones puede parecer algo muy limitado es importante destacar que en su momento nuestro estudio fue la simulación mas completa realizada hasta entonces. Asimismo, el hecho de haber obtenido expresiones analíticas para la fidelidad (que concuerdan con los resultados numéricos) es un aporte valioso que fue reconocido por varios autores. Posteriormente Obenland y Despain [Obenlan98] también simularon el algoritmo de factorización así como el de Grover en una trampa de iones. En esos trabajos, con un punto de vista centrado más en aspectos computacionales que físicos, se presentan resultados con mayor cantidad de iones (hasta 27) utilizando algoritmos más sofisticados en computadoras paralelas. Sin embargo las conclusiones no son muy distintas a las obtenidas en nuestro trabajo: para poder hacer cualquier clase de computación cuántica es fundamental utilizar métodos de corrección de errores.

Una vez que reconocemos que los errores que se generan a lo largo de la trayectoria computacional degradan la información almacenada y procesada, es importante estudiar métodos para proteger dicha información. Nuestra contribución principal al desarrollo de los códigos de corrección de errores fue el descubrimiento del código perfecto de 5 qubits y el estudio del proceso de corrección continua de errores. La literatura sobre este tema es extensa y nuestra contribución es reconocida por su importancia histórica al ser uno de los primeros códigos encontrados cuando todavía no se había desarrollado el método de los estabilizadores.

4. Representación en espacio de fase de una computadora cuántica

Una de las áreas más prometedoras para la aplicación de las computadoras cuánticas es la simulación de sistemas físicos. Así como muchas veces usamos simulaciones numéricas para resolver problemas de la física que somos incapaces de resolver analíticamente es posible hacer lo mismo con una computadora cuántica. Ya Feynman [Feynman82] había sugerido que una computadora cuántica sería útil para simular sistemas cuánticos. La dificultad de simular este tipo de problemas en computadoras clásicas proviene del crecimiento exponencial del espacio de Hilbert con el tamaño del sistema. Lloyd mostró que la conjetura de Feynman que una computadora cuántica puede simular eficientemente cualquier sistema cuántico local es verdadera [Lloyd93]. Varios autores mostraron que las computadoras cuánticas son mucho más eficientes para estos problemas: Zalka [Zalka98] mostró como podemos simular una partícula en una dimensión sometida a un potencial arbitrario $V(q)$. Somaroo et. al. [Somaroo99] utilizan un sistema de espines (mediante resonancia magnética nuclear en líquidos) para simular un oscilador armónico truncado. También analizan el efecto de un término cúbico en el potencial de oscilador armónico. Una de las áreas de más actividad desde hace muchos años es el estudio del caos en sistemas cuánticas. Schack et al [Schak98] mostraron como es posible utilizar una computadora cuántica para simular el caballito de batalla en el estudio de sistema caóticos: el mapa del panadero. Este mapa es ampliamente estudiado en la literatura y su límite semi-clásico ayuda a tomar contacto con toda la teoría clásica del caos. En todos estos estudios es provechoso tener una representación visual de la evolución para poder comparar los mapas cuánticos con sus parientes clásicos. En mecánica clásica se utiliza el espacio de fase para tener una representación gráfica del estado del sistema. Afortunadamente es posible encontrar en mecánica cuántica una representación en espacio de fase para representar estados y su evolución temporal que, en el límite semi-clásico coincide con el espacio de fase clásico. Esta representación fue introducida por Wigner [Wigner84] y se conoce como función de Wigner. La dificultad con la que uno se topa al intentar utilizar éstas técnicas con una computadora cuántica es que dicha representación está definida para sistemas con coordenadas continuas. Tanto en el estudio de sistemas caóticos como en computación cuántica trabajamos con espacios de dimensión finita. En éste capítulo introducimos una representación en el espacio de fase para sistemas cuánticos con espacio de Hilbert de dimensión finita. Haremos esto siguiendo un enfoque similar al de Leonhardt [Leonhardt96] quien definió funciones de Wigner para variables de espín (entero o semi-entero). Existen, además otros enfoques propuestos por Bouzouina y Bievre [Bouzouina96] y por Takami et al [Takami01].

Una pregunta interesante es si es posible utilizar esta representación para analizar los algoritmos cuánticos conocidos hasta el momento o incluso, encontrar algoritmos nuevos. Si bien hasta el momento la respuesta no es conocida aquí veremos algunos ejemplos de aplicaciones de estas ideas a distintos algo-

ritmos cuánticos como la transformada de Fourier cuántica, y el algoritmo de Grover.

4.1. Funciones de Wigner para sistemas continuos

Antes de introducir la representación para sistemas discretos es útil hablar de las funciones de Wigner para sistemas continuos. Para una partícula en una dimensión, la función de Wigner se puede calcular conocida la matriz densidad $\hat{\rho}$ del sistema utilizando la siguiente expresión:

$$W(q,p) = \int \frac{d\lambda}{2\pi\hbar} e^{i\lambda p/\hbar} \langle q - \lambda/2|\hat{\rho}|q + \lambda/2\rangle. \tag{4.1}$$

Ésta función es la más parecida a una distribución clásica $f(q,p)$ para un sistema cuántico. Las tres propiedades que la definen son: (P1) $W(q,p)$ es real, (P2) el producto escalar entre dos estados $\hat{\rho}_1$, $\hat{\rho}_2$ se puede computar a partir de la función de Wigner mediante: $\mathrm{Tr}[\hat{\rho}_1\hat{\rho}_2] = 2\pi\hbar \int dqdp W_1(q,p)W_2(q,p)$ y (P3) la integral a lo largo de cualquier recta en el espacio de fase definida por la ecuación $a_1 q + a_2 p = a_3$ es la densidad de probabilidad de que el observable $a_1\hat{Q} + a_2\hat{P}$ tome el valor a_3. Esta última propiedad (el hecho que $W(q,p)$ da las distribuciones marginales correctas para cualquier cuadratura) es la más restrictiva y, como Bertrand y Bertrand muestran [Bertrand87], junto con P1-P2 determinan la función de Wigner de manera unívoca.

Es conveniente escribir $W(q,p)$ como el valor de expectación de un operador $\hat{A}(q,p)$ conocido como "operador de punto en el espacio de fase" [Wooters87] (u operadores de Fano [Fano57]):

$$W(q,p) = \mathrm{Tr}[\hat{\rho}\,\hat{A}(q,p)\,]. \tag{4.2}$$

El operador $\hat{A}(q,p)$ depende paramétricamente de q,p y puede escribirse en términos de productos simetrizados de funciones delta:

$$\begin{aligned} \hat{A}(q,p) &= :\delta(\hat{P} - p)\,\delta(\hat{Q} - q): \\ &= \int \frac{d\lambda d\lambda'}{(2\pi\hbar)^2} e^{-i\frac{\lambda}{\hbar}(\hat{P}-p)+i\frac{\lambda'}{\hbar}(\hat{Q}-q)} \\ &= \int \frac{d\lambda d\lambda'}{(2\pi\hbar)^2} \hat{D}(\lambda,\lambda')\, e^{-\frac{i}{\hbar}(\lambda' q - \lambda p)}, \end{aligned} \tag{4.3}$$

donde hemos identificado el operador de traslaciones continuo $\hat{D}(\lambda,\lambda') = \exp[-\frac{i}{\hbar}(\lambda\hat{P} - \lambda'\hat{Q})]$. La expresión (4.3) se puede interpretar como una doble transformada de Fourier del operador desplazamiento (y, por lo tanto, $W(q,p)$ es la doble transformada de Fourier del valor de expectación de $D(\lambda,\lambda')$). Resulta aún más conveniente reescribir esta ecuación de la siguiente manera:

$$\hat{A}(q,p) = \frac{1}{\pi\hbar}\hat{D}(q,p)\hat{R}\hat{D}^{\dagger}(q,p), \tag{4.4}$$

donde $\hat{R}$ es el operador de reflexión (paridad) que actúa de la siguiente manera $\hat{R}|x\rangle = |-x\rangle$ sobre los auto-estados de posición. Es decir: la función de Wigner es el valor de expectación del operador de reflexión desplazado.

Para demostrar que la definición de la función de Wigner mostrada aquí cumple con P1-P3 basta estudiar algunas de las propiedades de los $\hat{A}(q,p)$. Que $W(q,p)$ es real (P1) se deduce de la hermiticidad de los operadores $\hat{A}(q,p)$. La propiedad (P2) es consecuencia de la completitud de $\hat{A}(q,p)$ pues es fácil demostrar que:

$$\mathrm{Tr}[\hat{A}(q,p)\hat{A}(q',p')] = \frac{1}{2\pi\hbar}\delta(q-q')\delta(p-p'). \tag{4.5}$$

Como consecuencia es posible invertir (4.2) y escribir la matriz densidad como combinación lineal de los operadores de espacio de fase. La función de Wigner determina los coeficientes de dicha expansión: $\hat{\rho} = 2\pi\hbar \int dqdpW(q,p)\hat{A}(q,p)$ de donde podemos verificar P2. La última propiedad (P3) se demuestra observando que si integramos $\hat{A}(q,p)$ sobre una linea en el espacio de fase uno siempre obtiene un proyector. Es decir:

$$\int dqdp\,\delta(a_1q + a_2p - a_3)\hat{A}(q,p) = |a_3\rangle\langle a_3|, \tag{4.6}$$

donde $|a_3\rangle$ es un auto-estado del operador $a_1\hat{Q} + a_2\hat{P}$ con auto-valor a_3. Esta identidad se puede demostrar si escribimos la función delta como una integral de una exponencial y resolviendo la integral sobre una linea del espacio de fase. Omitimos la prueba ya que más adelante incluiremos la versión discreta de la misma.

4.2. Funciones de Wigner para sistemas discretos

4.2.1. Nociones preliminares: espacio de fase discreto

Ahora consideremos un sistema cuántico con un espacio de Hilbert de dimensión N (en general vamos a pensar en una computadora cuántica pero el formalismo es aplicable a cualquier sistema discreto). Introduzcamos la siguiente base del espacio $B_x = \{|n\rangle, n = 0,\ldots,N-1\}$ que interpretaremos como la base de posición (con condiciones periódicas de contorno: $|n+N\rangle = |n\rangle$). En el caso de una computadora cuántica esta base será la 'base computacional'. Dada la base B_x, hay una manera natural de introducir una base conjugada de momentos $B_p = \{|k\rangle, k = 0,\ldots,N-1\}$ utilizando la transforma de Fourier discreta. Los estados de B_p se obtienen de los estados de B_x mediante:

$$|k\rangle = \frac{1}{\sqrt{N}}\sum_{n=0}^{N-1}\exp(i2\pi nk/N)|n\rangle. \tag{4.7}$$

Por lo tanto, al igual que en el caso continuo, posición y momento están

relacionados por medio de una transformada de Fourier. También es importante destacar que la dimensionalidad del espacio de Hilbert del sistema está relacionado con la constante de Plank. Si suponemos que el espacio de fase tiene un área finita (que supondremos igual a 1 en unidades apropiadas), en esta área deben ser capaces de entrar N estados ortogonales. Si cada estado ocupa una área en el espacio de fase de $2\pi\hbar$ entonces:

$$N = 1/2\pi\hbar. \tag{4.8}$$

En otras palabras, N juega el lugar de la inversa de la constante de Plank (y por lo tanto, el límite semi-clásico corresponde a N grande).

Una vez que tenemos las bases de posición y momento, debemos introducir operadores de desplazamiento. Al trabajar en un espacio discreto no tiene sentido hablar de desplazamientos infinitesimales. De todas maneras podemos introducir operadores de traslación finita $\hat{U}$ y $\hat{V}$, que generan traslaciones en posición y momento respectivamente. El operador de traslación $\hat{U}$ genera shifts cíclicos en la base de posición y es diagonal en la base de momentos:

$$\hat{U}^m|n\rangle = |n+m\rangle,\ \hat{U}^m|k\rangle = \exp(-2\pi imk/N)|k\rangle, \tag{4.9}$$

(todas las sumas dentro de los kets son mod N). Análogamente, el operador $\hat{V}$ es un shift en momento y es diagonal en la base de posición:

$$\hat{V}^m|k\rangle = |k+m\rangle,\ \hat{V}^m|n\rangle = \exp(i2\pi mn/N)|n\rangle \tag{4.10}$$

Los operadores $\hat{U}$ y $\hat{V}$ satisfacen relaciones de conmutación que generalizan las del caso continuo. De hecho es fácil verificar la siguiente identidad:

$$\hat{V}^p\hat{U}^q = \hat{U}^q\hat{V}^p \exp(i2\pi pq/N). \tag{4.11}$$

Con estas definiciones ahora podemos introducir el análogo del operador de desplazamiento continuo $\hat{D}(q,p) = \exp[-\frac{i}{\hbar}(q\hat{P} - p\hat{Q})]$. Para ello lo podemos re-escribir como: $\hat{D}(q,p) = \exp(-iq\hat{P}/\hbar)\exp(ip\hat{Q}/\hbar)\exp(ipq/2\hbar)$ (utilizando la famosa fórmula $e^{\hat{A}+\hat{B}} = e^{-[\hat{A},\hat{B}]/2}e^{\hat{A}}e^{\hat{B}}$ y las relaciones de conmutación canónicas entre $\hat{Q}$ y $\hat{P}$). Si ahora identificamos los correspondientes operadores de desplazamiento, el análogo para el caso discreto resulta:

$$\hat{T}(q,p) \equiv \hat{U}^q\hat{V}^p \exp(i\pi qp/N). \tag{4.12}$$

Estos operadores obedecen la siguiente regla de composición:

$$\hat{T}(q_1,p_1)\hat{T}(q_2,p_2) = \hat{T}(q_1+q_2, p_1+p_2)e^{i\pi(p_1q_2 - q_1p_2)/N},$$

donde la fase que aparece en el término de la mano derecha tiene una interpretación geométrica como el área de un triángulo (ver más adelante). De la definición anterior es fácil ver que el operador de traslación es tal que:

$$\hat{T}^{\dagger}(q,p) = \hat{T}(2N-q,2N-p) \tag{4.14}$$
$$= \hat{T}(N-q,N-p)\,(-1)^{N+p+q}. \tag{4.15}$$

Por último, utilizando la regla de composición debe ser que:

$$\hat{T}(\lambda q,\lambda p) = \hat{T}^{\lambda}(q,p) \tag{4.16}$$

para cualquier entero λ. Uno puede estar tentado en definir operadores de posición y momento en el caso discreto. Podríamos intentarlo escribiendo $\hat{U}$ y $\hat{V}$ como exponenciales de dos operadores $\hat{Q}$ y $\hat{P}$ definidos de manera que sean diagonales en las bases B_x y B_p respectivamente: $\hat{U} = \exp(-i2\pi\hat{P}/N)$ y $\hat{V} = \exp(i2\pi\hat{Q}/N)$, con $\hat{Q} \equiv \sum_n n|n\rangle\langle n|$ y $\hat{P} \equiv \sum_k k|k\rangle\langle k|$. Sin embargo resulta que estos operadores no resultan canónicamente conjugados ya que su conmutador no es proporcional a la identidad como en el caso continuo (para espacios de dimensión finita es imposible definir dos operadores cuyo conmutador es proporcional a la identidad). Por lo tanto, aun cuando los operadores de desplazamiento satisfacen las misma propiedades que sus versiones continuas esto no ocurre con los operadores de posición y momento así definidos.

También resultará útil definir el operador reflexión (o de paridad) $\hat{R}$ que actuando sobre auto-estados de posición (y momento) satisface $\hat{R}|n\rangle \equiv |-n\rangle$ (como siempre, las sumas deben interpretarse mod N). Es fácil verificar la siguiente relación entre el operador de paridad y los operadores de traslación $\hat{U}$ y $\hat{V}$:

$$\hat{U}\hat{R} = \hat{R}\hat{U}^{-1},\ \hat{V}\hat{R} = \hat{R}\hat{V}^{-1}. \tag{4.17}$$

Más adelante será importante darse cuenta que el operador de reflexión esta estrechamente ligado con la transformada de Fourier discreta. De hecho, denotamos al operador de la transformada de Fourier discreta con $\hat{U}_{UF}$ (i.e., el operador cuyo elementos de matriz en la base B_x son $\langle n'|U_{FT}|n\rangle = \exp(i2\pi nn'/N)$), entonces el operador de reflexión es simplemente el cuadrado de la transformada de Fourier:

$$\hat{R} = U_{FT}^2. \tag{4.18}$$

4.2.2. Función de Wigner discreta

Para definir una función de Wigner discreta es más conveniente buscar una generalización para el caso discreto de los operadores $\hat{A}(q,p)$. Para ello

queremos encontrar una base del espacio de operadores hermíticos (como la dimensión del espacio de Hilbert es N dicha base está compuesta por N^2 operadores). Para encontrar dicha base es fundamental que al integrar la función de Wigner resultante sobre lineas en el espacio de las fases obtengamos las densidades de probabilidad marginales. La generalización al caso discreto no es directa y conviene ver un par de intentos para ver cuales son las dificultades. Si intentamos generalizar (4.3) al caso discreto obtenemos:

$$\hat{A}(q,p) = \frac{1}{N^2} \sum_{m,k=0}^{N-1} \hat{T}(m,k) \exp\{-i2\pi \frac{(kq - mp)}{N}\}. \tag{4.19}$$

Desafortunadamente, la expresión para $\hat{A}(q,p)$ resulta ser no-hermítica. Esto se puede verificar si tomamos el hermítico conjugado de la expresión anterior y usamos el hecho que el hermítico conjugado del operador de traslación en el espacio de fase, como se ve en (4.15), es tal que $\hat{T}^\dagger(q,p) \neq \hat{T}(N-q, N-p)$.

Otro intento para definir los operadores $\hat{A}(q,p)$ es generalizar la ecuación (4.4). En este caso sí obtenemos un operador que es hermítico por construcción:

$$\hat{A}(q,p) = \frac{1}{\pi\hbar} \hat{D}(q,p) \hat{R} \hat{D}^\dagger(q,p) \rightarrow \frac{1}{N/2} \hat{T}(q,p) \hat{R} \hat{T}^\dagger(q,p). \tag{4.20}$$

El problema ahora es que el conjunto así definido no es completo. Si usamos la definición de $\hat{T}(q,p)$ dado (4.12) y las relaciones de conmutación entre $\hat{U}$, $\hat{V}$ y $\hat{R}$ obtenemos:

$$\hat{A}(q,p) = \frac{1}{N} \hat{U}^{2q} \hat{R} \hat{V}^{-2p} \exp(4\pi i\, pq/N). \tag{4.21}$$

Como $\hat{U}$ y $\hat{V}$ son operadores cíclicos con período N, es fácil ver que si q y p toman valores entre 0 y N, la expresión anterior da sólo $N^2/4$ operadores independientes. La ecuación (4.21) implica que $\hat{A}(q + N/2, p) = \hat{A}(q,p)$ y lo mismo con p por lo que sólo tenemos $N/2 \times N/2$ operadores independientes.

Curiosamente, la solución a los problemas que surgen en las dos propuestas hechas es la misma: como han hecho otros autores [Hannay80, Bouzouina96], el problema es resuelto expandiendo el espacio de fase a una grilla de $2N \times 2N$ puntos. Esto se logra reemplazando $N \rightarrow 2N$ en la ecuación (4.19) o permitiendo que q y p tomen valores semi-enteros en (4.21). La definición correcta resulta entonces:

$$\hat{A}(\alpha) = \frac{1}{(2N)^2} \sum_{\lambda,\lambda'=0}^{2N-1} \hat{T}(\lambda,\lambda') \exp\{-i2\pi \frac{(\lambda' q - \lambda p)}{2N}\} \tag{4.22}$$

$$= \frac{1}{2N} \hat{U}^q \hat{R} \hat{V}^{-p} \exp(i\pi pq/N) \tag{4.23}$$

donde $\alpha = (q,p)$ denota un punto en el espacio de fase y q y p toman valores entre 0 y $2N-1$.

Hay que recordar que al definir los operadores de punto del espacio de fase en una grilla de $2N \times 2N$ puntos, tenemos un total de $4N^2$ operadores que no puede ser todos independientes. Es fácil verificar que sólo N^2 de ellos son independientes pues:

$$\hat{A}(q+\sigma_q N, p+\sigma_p N) = \hat{A}(q,p)\,(-1)^{\sigma_p q+\sigma_q p+\sigma_q\sigma_p N}, \tag{4.24}$$

para $\sigma_q, \sigma_p = 0, 1$. Es claro entonces que los N^2 operadores definidos en la primera sub-grilla de $N \times N$ determinan al resto. Por comodidad denotaremos por G_N a ésta sub-grilla (i.e., $G_N = \{\alpha = (q,p); 0 \leq q,p \leq N-1\}$). El conjunto G_{2N} denotará al conjunto completo de $2N \times 2N$ puntos del espacio de fase.

Antes de mostrar que la función de Wigner definida usando los $\hat{A}(q,p)$ satisface las propiedades P1-P3 conviene estudiar algunas de las propiedades de los operadores de punto. Estos operadores están estrechamente relacionados con las traslaciones. En particular, el producto de dos $\hat{A}(q,p)$ nos genera una traslación:

$$\hat{A}(\alpha)\hat{A}(\alpha') = \hat{T}(q-q', p-p')\,\frac{(-1)^{qp+q'p'+qp'+pq'}}{4N^2} \tag{4.25}$$

También resultará útil invertir la relación (4.22) para poder escribir al operadores de traslación como función de los $\hat{A}(\alpha)$:

$$\tilde{T}(n,k) = 2N \sum_{q,p=0}^{2N-1} \hat{A}(q,p)\ \exp(-i\frac{2\pi}{2N}(np-kq)), \tag{4.26}$$

De las propiedades anteriores es posible mostrar que $\hat{A}(\alpha)$ forman un conjunto completo cuando α toma valores en la primer sub-grilla G_N de $N \times N$ puntos. Si tomamos la traza de (4.25) obtenemos:

$$\mathrm{Tr}[\hat{A}(\alpha)\hat{A}(\alpha')] = \frac{1}{4N}\,\delta_N(q'-q)\delta_N(p'-p) \tag{4.27}$$

donde tanto α como α' están en la grilla G_N y $\delta_N(q) \equiv \frac{1}{N}\sum_{n=0}^{N-1} \mathrm{e}^{2\pi iq/N}$ es la función delta periódica (que es cero a menos que $q = 0 \bmod N$).

Por lo discutido anteriormente, la función de Wigner discreta queda definida entonces como:

$$W(\alpha) = \mathrm{Tr}(\hat{A}(\alpha)\hat{\rho}) \tag{4.28}$$

done $\alpha \in G_{2N}$. Claramente los $4N^2$ puntos no son todos independientes pues la función de Wigner así definida satisface:

$$\hat{W}(q+\sigma_q N, p+\sigma_p N) = \hat{W}(q,p)\,(-1)^{\sigma_p q+\sigma_q p+\sigma_q\sigma_p N} \tag{4.29}$$

Como los operadores $\hat{A}(\alpha)$ son un conjunto completo es posible expandir cualquier matriz densidad como combinación lineal de dichos operadores. Es claro entonces que $W(\alpha)$ no son otra cosa que los coeficientes de expansión de dicha combinación lineal:

$$\hat{\rho} = 4N \sum_{\alpha\in G_N} W(\alpha)\hat{A}(\alpha) \tag{4.30}$$

$$= N \sum_{\tilde{\alpha}\in G_{2N}} W(\tilde{\alpha})\hat{A}(\tilde{\alpha}). \tag{4.31}$$

La última expresión se obtiene de (4.30) si observamos que la contribución de cada una de las cuatro grillas de $N \times N$ son idénticas.

Ahora sí podemos mostrar que la función de Wigner que hemos definido satisface las propiedades P1-P3. La primer propiedad es simplemente consecuencia de el hecho que los operadores $\hat{A}(\alpha)$ son hermíticos por construcción. (P2) es consecuencia de la completitud del conjunto $\hat{A}(\alpha)$ que permite mostrar que:

$$\mathrm{Tr}[\rho_1\rho_2] = N \sum_{\alpha\in G_{2N}} W_1(\alpha)W_2(\alpha) \tag{4.32}$$

La propiedad P3 es bastante más sutil y requiere un poco de cuidado. La clave está en mostrar que los operadores $\hat{A}(q,p)$ sumados sobre todos los puntos del espacio de fase que pertenecen a una linea L generan un operador de proyección (esto garantiza que cuando sumamos los valores de la función de Wigner sobre una recta obtenemos un número positivo que puede interpretarse como una probabilidad). Antes de demostrarlo conviene definir con mas exactitud que queremos decir por una linea $L(n_1,n_2,n_3)$ en nuestro espacio de fase que es una grilla. Introduzcamos la siguiente definición: una linea L es un conjunto de puntos de la grilla definidos por $L = L(n_1,n_2,n_3) = \{(q,p) \in G_{2N},$ tal que $n_1 p - n_2 q = n_3$, con $0 \leq n_i \leq 2N-1\}$. Mas adelante discutiremos algunas propiedades de las lineas sobre una grilla G_{2N}. Aquí solo nos interesa destacar que es posible introducir una noción de paralelismo: dos lineas son paralelas si están parametrizadas por los mismos enteros n_1 y n_2 (en la figura (4.1) mostramos algunos ejemplos en una grilla).

Mostremos ahora que si sumamos operadores de punto sobre una linea obtenemos un proyector. Nos interesa entonces analizar el siguiente operador:

$$\hat{A}_L \equiv \sum_{(q,p)\in L} \hat{A}(q,p). \tag{4.33}$$

Claramente lo podemos reescribir de la siguiente manera:

$$
\begin{aligned}
A_L &= \sum_{q,p=0}^{2N-1} \hat{A}(q,p)\ \delta_{2N}(n_1 p - n_2 q - n_3) \\
&= \frac{1}{2N} \sum_{\lambda=0}^{2N-1} \sum_{q,p=0}^{2N-1} \hat{A}(q,p)\ \mathrm{e}^{-i\frac{2\pi}{2N}\lambda(n_1 p - n_2 q - n_3)} \\
&= \sum_{\lambda=0}^{2N-1} \hat{T}^{\lambda}(n_1,n_2)\ \mathrm{e}^{-i\frac{2\pi}{2N}n_3\lambda} \qquad (4.34)
\end{aligned}
$$

donde para hacer la suma sobre la grilla en el espacio de fase usamos el hecho que la transformada de Fourier de los $\hat{A}(\alpha)$ está dada por (4.26). Como $\hat{T}(n_1,n_2)$ es un operador unitario tiene N auto-vectores $|\phi_j\rangle$ con auto-valor $\exp(i2\pi\phi_j/N)$. Si tenemos en cuenta que el operador es cíclico, debe satisfacerse que $\hat{T}^N = \mathbb{I}$ por lo que sus auto-valores son las raíces N-ésimas de la unidad y los ϕ_j deben ser enteros. Ahora podemos reescribir (4.34) como:

$$
\begin{aligned}
\hat{A}_L &= \sum_{\lambda=0}^{2N-1} \sum_{j=0}^{N} \mathrm{e}^{i\frac{2\pi}{2N}(2\phi_j - n_3)\lambda}\ |\phi_j\rangle\langle\phi_j| \\
&= \sum_{j=0}^{N} \delta_{2N}(2\phi_j - n_3)\ |\phi_j\rangle\langle\phi_j|. \qquad (4.35)
\end{aligned}
$$

Aquí es claro que $\hat{A}_L$ es un operador de proyección sobre el sub-espacio generado por los auto-vectores del operador de traslación $\hat{T}(n_1,n_2)$. La dimensionalidad d del sub-espacio es igual a la traza de $\hat{A}_L$. Para calcularla debemos observar que, en general,

$$
\mathrm{Tr}[\hat{A}(q,p)] = \frac{1}{2N} \sum_{q'=0}^{N-1} \delta_N(q - 2q')\ \mathrm{e}^{i\pi(q-2q')p/N} \qquad (4.36)
$$

La última ecuación es fácil de evaluar pero el resultado depende de la paridad de N. Si N es par, $\mathrm{Tr}(\hat{A}(q,p)) = 1/N$ sólo si tanto q como p son pares y cero en todos los otros casos. Para valores impares de N, $\mathrm{Tr}(\hat{A}(q,p)) = 1/2N$ para todos los q y p excepto cuando son los dos impares en cuyo caso toma el mismo valor con el signo cambiado (es decir $-1/2N$). Como a nosotros nos interesan las computadoras cuánticas que utilizan espacios de Hilbert de dimensión par de aquí en adelante asumiremos N par. Usando el resultado anterior para la traza de $\hat{A}(\alpha)$ es fácil ver que la dimensionalidad de $\hat{A}_L$ es $1/N$ multiplicado por el número de puntos que pertenecen a L que tienen coordenadas con q y p pares. Como consecuencia inmediata es claro que si n_3 es impar, $d = 0$ (pues la suma de dos números pares nunca puede ser impar). Finalmente es posible escribir una expresión explícita para d (cuando N es par):

$$d = \frac{1}{2} \sum_{\lambda=0}^{N-1} \delta_N(\lambda n_1)\delta_N(\lambda n_2) e^{i\frac{2\pi}{2N}m\lambda} (1 + (-1)^{n_3}) \tag{4.37}$$

Veamos una aplicación del resultado anterior en un caso concreto: sea L_q la linea $a = n_3$ (es decir: $n_1 = 1$, $n_2 = 0$), la función de Wigner sumada sobre todos los puntos de L_q es $\sum_{(q,p)\in L_q} W(q,p) = \sum_p W(n_3,p) = \langle n_3/2|\hat{\rho}|n_3/2\rangle$ si n_3 es par y cero en el caso contrario. Análogamente, si consideramos lineas horizontales (L_p definida como $p = n_3'$) obtenemos $\sum_{(q,p)\in L_p} W(q,p) = \sum_q W(q,n_3') = \langle n_3'/2|\hat{\rho}|n_3'/2\rangle$ si n_3' es par y cero en el caso contrario. Estos son sólo dos ejemplos del resultado general: la función de Wigner siempre genera la distribuciones marginales correctas (que es la propiedad fundamental de la función de Wigner en el caso continuo).

Antes de pasar a ver algunos ejemplos vamos a mencionar algunas propiedades conocidas (y otras nuevas) de las lineas sobre grillas. La mayoría de las propiedades fueron introducidas por Wooters [Wooters87]. Como habíamos dicho era posible definir lineas $L(n_1,n_2,n_3)$ y una noción de paralelismo en la grilla G_{2N}. Una 'foliación' de la grilla es una familia de lineas paralelas obtenidas si fijamos n_1 y n_2 pero variamos n_3 (en general, dos lineas serán paralelas si el cociente n_1/n_2 es el mismo). Si N es un número primo entonces la grilla G_N (que tiene $N \times N$ puntos) hay exactamente $N(N+1)$ lineas distintas que se pueden agrupar en $N+1$ conjuntos de lineas paralelas (hay exactamente $N+1$ foliaciones de la grilla). Si N no es primo o, como es el caso que nos interesa, la grilla es G_{2N}, este resultado deja de ser válido. Por ejemplo, es claro que la ecuación $n_1 q - n_2 p = n_3 \pmod{2N}$ no tiene solución para valores impares de n_3 y valores pares de n_1 y n_2. Por lo que no es en general cierto que las lineas tengan todas $2N$ puntos: como mostramos recién hay casos en los que las lineas pueden no tener ningún punto, y a veces es posible construir lineas con un número de puntos que son múltiplos de $2N$. Este es el caso si n_1 y n_2 tienen factores comunes primos con N. Por ejemplo en el caso $2N = 8$, cada linea $n_1 = n_2 = 1$ tiene 8 puntos, pero las lineas con $n_1 = n_2 = 2$ no tienen puntos si n_3 es impar o tiene 16 puntos si n_3 es par. De todas formas, la manera más sencilla de representar las lineas en el espacio de fase es observando que la topología de la grilla es un toro (debido a las propiedades periódicas de contorno) por lo que las lineas se envuelven alrededor de la superficie del toro. El número de puntos en la linea está relacionado al número de vueltas que da la linea antes de cerrarse con sigo mismo. La figura 4.1 muestra un ejemplo de dos conjuntos de lineas en la grilla G_{2N} para el caso $N = 4$.

Hagamos un pequeño resumen de lo mostrado en esta sección: definimos una función de Wigner para sistemas de dimensión finita (de dimensión arbitraria N). La función se define como el valor de expectación de los operadores de punto $\hat{A}(\alpha)$ dado por (4.23). Esta definición es tal que $W(\alpha)$ es real, se puede usar para calcular productos escalares entre estados y da las distribuciones marginales correctas cuando se suman sobre lineas en el espacio de fase que es una grilla G_{2N} con $2N \times 2N$ puntos. El tamaño de la grilla para el espacio de fase es importante para obtener una función de Wigner que cumpla

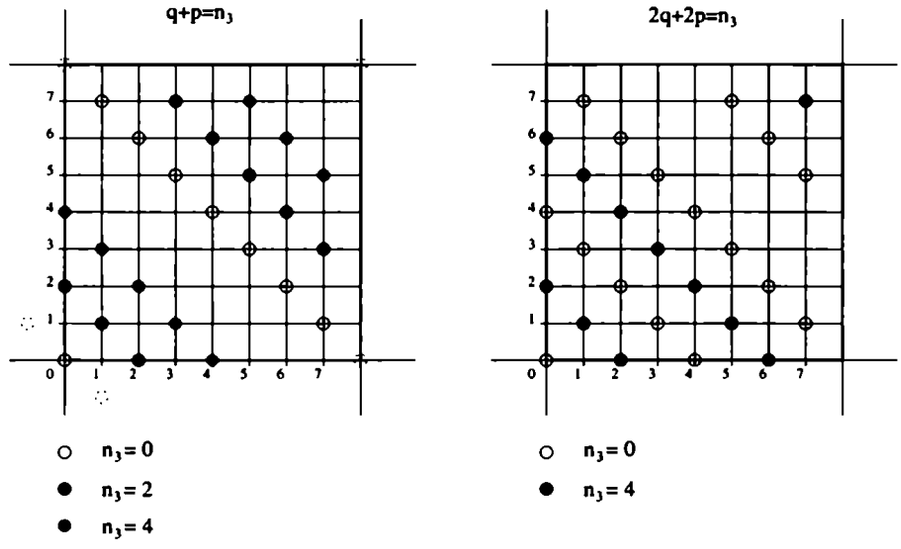


Figura 4.1: En la parte izquierda se muestran las rectas correspondientes a $q + p = 0, 2$. Ambas rectas tienen un total de 8 puntos y la ecuación $q + p = n_3$ tiene ocho rectas distintas. La parte derecha muestra la ecuación $2q + 2p = n_3$. Allí sólo hay 4 rectas distintas, cada una con 16 puntos. Estos casos muestran que, si bien las dos rectas tienen la misma pendiente, las rectas pueden tener distinta cantidad de puntos dependiendo de n_3.

con las propiedades P1-P3. Los valores de $W(\alpha)$ sobre la sub-grilla G_N son suficientes para reconstruir el resto del espacio de fase (pues el conjunto $\hat{A}(\alpha)$ es completo cuando α perteneces a la grilla G_N. Sin embargo, la redundancia introducida al duplicar el número de sitios tanto en q como en p es esencial cuando queremos imponer la propiedad P3.

4.3. Funciones de Wigner para estados cuánticos

Para calcular la función de Wigner de cualquier estado cuántico es conveniente usar la ecuación (4.23) para $\hat{A}(q,p)$ y escribir:

$$W(q,p) \equiv \frac{1}{2N} \sum_{n=0}^{N-1} \langle q - n|\hat{\rho}|n\rangle e^{i\frac{2\pi}{N}n(p-q/2)}. \tag{4.38}$$

Es importante recordar que basta calcular los N^2 valores independientes en la primer sub-grilla (restringiéndonos a $\alpha = (q,p) \in G_N$) y usar (4.29) para calcular en los restantes $3N^2$ puntos.

Antes de mostrar casos específicos de representaciones de Wigner para estados vale la pena mencionar algunas características generales de la representación de estados puros. Si el estado cuántico es puro entonces $\hat{\rho}$ es un operador de proyección. al expandir $\hat{\rho}$ en operadores de punto usando (4.31) e imponiendo la condición $\hat{\rho}^2 = \hat{\rho}$ obtenemos:

$$W(\alpha) = 4N^2 \sum_{\beta\gamma \in G_N} \Gamma(\alpha,\beta,\gamma) W(\beta) W(\gamma). \tag{4.39}$$

donde la función $\Gamma(\alpha,\beta,\gamma)$, depende de tres puntos del espacio de fase (i.e. un triángulo) es

$$\begin{aligned} \Gamma(\alpha,\beta,\gamma) &= \mathrm{Tr}(\hat{A}(\alpha)\hat{A}(\beta)\hat{A}(\gamma)) \\ &= \frac{1}{4N^3} e^{i\frac{2\pi}{N}S(\alpha,\beta\gamma)} \end{aligned} \tag{4.40}$$

si dos o tres de los puntos (α,β,γ) tiene coordenadas q par y p par. De otra manera $\Gamma(\alpha,\beta,\gamma) = 0$. En la expresión anterior, que es válida para valores pares de N, $S(\alpha,\beta,\gamma)$ es el área del triángulo formado por los tres puntos del espacio de fase (medidos en unidades donde el área de un triángulo en el que los tres puntos estén separados por un sitio es igual a uno). Una expresión similar fue obtenida por Wooters [Wooters87]. La función de tres puntos $\Gamma(\alpha,\beta,\gamma)$ tiene un significado geométrico y juega un rol interesante cuando se estudian las propiedades de la evolución temporal.

4.3.1. Auto-estados de posición y momentos, y sus superposiciones

Como primer ejemplo podemos calcular la función de Wigner de un auto-estado de posición (un estado computacional de la computadora cuántica) $\hat{\rho}_{q_0} = |q_0\rangle\langle q_0|$. Es simple obtener un expresión analítica para $W(q,p)$:

$$\begin{aligned} W_{q_0}(q,p) &= \frac{1}{2N} \langle q_0 | \hat{U}^q \hat{R} \hat{V}^{-p} | q_0 \rangle e^{i\pi pq/N} \\ &= \frac{1}{2N} \delta_N(q - 2q_0)\ (-1)^{p(q-2q_0)_N} \end{aligned} \tag{4.41}$$

donde z_N denota z modulo N. La función es cero en todos lados salvo en dos franjas localizadas en $q = 2q_0$ modulo N. Cuando $q = 2q_0$, $W(q,p)$ toma el valor constante $1/2N$ mientras que para $q = 2q_0 \pm N$ vale $1/2N$ para valores pares de p o $-1/2N$ para valores impares. Estas oscilaciones son características de fenómenos de interferencia y se puede interpretar que aparecen debido a la interferencia entre la franja que tiene $q = 2q_0$ y su imagen especular a una distancia $2N$ de $2q_0$, que aparece debido a las condiciones periódicas de contorno. El hecho que la función de Wigner tome valores negativos es fundamental para reobtener las distribuciones marginales correctas. Sumando los valores de $W(q,p)$ sobre una linea vertical da la probabilidad de medir $q/2$ que es 1 para $q = 2q_0$ y cero para todos los otros valores. Se puede hacer un cálculo muy similar par auto-estados de momento $\hat{\rho} = |k_0\rangle\langle k_0|$. El resultado es muy similar pero ahora las franjas son horizontales.

Es interesante analizar también la función de Wigner de un estado que es superposición de dos estados computacionales: $|\psi\rangle = \left(|q_0\rangle + e^{-i\phi}|q_1\rangle\right)/\sqrt{2}$.

En este caso también es posible obtener una expresión cerrada para $W(q,p)$ que resulta:

$$W(q,p) = \frac{1}{2}\left[W_{q_0}(q,p) + W_{q_1}(q,p) + \Delta W_{q_0,q_1}(q,p)\right]$$

donde el término de interferencia es

$$\Delta W_{q_0,q_1}(q,p) \equiv \frac{1}{N}\,\delta_N(\tilde{q})\,(-1)^{\tilde{q}p}\,\cos\left(\frac{2\pi}{\lambda}p + \phi\right),$$

con $\tilde{q} = q_0 + q_1 - q$ y $\lambda \equiv 2N/(q_0 - q_1)$. Vemos entonces que la función de Wigner tiene dos términos que corresponden a los dos estados computacionales y un término de interferencia que tiene un pico en la franja vertical ubicada en el punto medio $q = q_0 + q_1 \mod N$. En esta franja la función de Wigner oscila con una longitud de onda que es inversamente proporcional a la separación entre las dos franjas principales. Este patrón oscilatorio tiene su imagen correspondiente y la interferencia genera a su vez otras oscilaciones. La figura 4.5 muestra la función de Wigner para un estado superposición de dos estados computacionales. En el gráfico es fácil reconocer las características recién discutidas.

4.3.2. Otros estados cuánticos

Es posible calcular la función de Wigner en forma cerrada para algunos otros casos interesantes. En la figura 4.6 se ve la función de Wigner para un estado totalmente mixto (que, como dijimos es cero en todos los puntos salvo para valores de α con q y p pares). También mostramos en la figura 4.6 la función de Wigner para un estado Gaussiano que es una superposición de estados computacionales (con condiciones periódicas de contorno, es decir, la suma de un estado Gaussiano centrado sobre un punto del espacio de fase y su infinitas imágenes Gaussianas especulares). El gráfico muestra, además de la Gaussiana principal a tres imágenes especulares moduladas por franjas de interferencia. Estas corresponden a la interferencia entre el pico principal y sus imágenes especulares (obsérvese la orientación de las franjas).

4.4. Evolución en el espacio de fase

Además de representar estados es posible también estudiar la evolución de los mismos en el espacio de fase. Si $\hat{U}$ es el operador unitario de evolución que toma el estado del sistema al tiempo t y lo evoluciona al tiempo $t+1$, la matriz densidad evoluciona de acuerdo a:

$$\rho(t+1) = U\rho(t)U^{\dagger}.$$

Usando esto, es simple ver que la función de Wigner evoluciona de la siguiente manera:

$$W(\alpha, t+1) = \sum_{\beta \in G_{2N}} Z_{\alpha\beta} W(\beta, t). \tag{4.43}$$

donde la matriz $Z_{\alpha\beta}$ se define como:

$$Z_{\alpha\beta} = N \quad Tr(\hat{A}(\alpha) U \hat{A}(\beta) U^{\dagger}). \tag{4.44}$$

Es decir, la evolución temporal en el espacio de fase es representada como una transformación lineal (que claramente es una consecuencia de la linealidad de la ecuación de Schrödinger). La unitariedad de la transformación impone algunos restricciones sobre la matriz $Z_{\alpha\beta}$: para preservar la pureza de los estados, la evolución temporal debe preservar la estructura de la ecuación de vínculo (4.39). Por lo tanto, la matriz deja invariante la función de tres puntos $\Gamma(\alpha, \beta, \gamma)$, i.e.

$$\Gamma(\alpha', \beta', \gamma') = \sum_{\alpha\beta\gamma} Z_{\alpha'\alpha} Z_{\beta'\beta} Z_{\gamma'\gamma} \Gamma(\alpha, \beta, \gamma).$$

La matriz real $Z_{\alpha\beta}$ contiene toda la información de la evolución temporal. En general, esta matriz conecta un punto α con muchos puntos β por lo que, en general, la evolución será no-local en el espacio de fase. Esta es una característica de la mecánica cuántica: los sistemas clásico, por el contrario, evolucionan siguiendo un flujo de trayectorias. En ese caso, el valor de la distribución clásica $W(\alpha, t+1)$ es igual al valor $W(\beta, t)$ para algún punto β que está bien definido por el valor de α y t. Es interesante preguntarse qué clase de operadores unitarios generaran dinámicas locales en el espacio de fase. A continuación veremos algunos ejemplos.

4.4.1. Traslaciones en el espacio de fases

Es fácil demostrar que si consideramos la evolución unitaria que corresponde a una traslación en el espacio de fase, es decir, el operador $\hat{T}(\sigma) = \hat{T}(q,p)$ entonces la evolución temporal es local en el espacio de fase. Mas aun, la evolución cuántica y clásica son idénticas pues el valor de la función de Wigner es trasladada rígidamente en el espacio de fase:

$$U = \hat{T}(\sigma) \iff W(\alpha, t+1) = W(\alpha - 2\sigma, t). \tag{4.46}$$

Notar el factor 2 en la expresión que es una simple consecuencia de trabajar en un espacio de fase de $2N \times 2N$. Otro ejemplo de evolución local es la asociada al operador $\hat{A}(\alpha)$. En tal caso la evolución resultante no es sólo una traslación sino que además está combinada con una reflexión:

$$U = \hat{A}(\sigma) \iff W(\alpha, t+1) = W(2\sigma - \alpha, t). \tag{4.47}$$

Los dos ejemplos presentados aquí son ejemplos donde hay una correspondencia entre evolución clásica y cuántica. Ahora veremos que hay otras familias de operadores un poco mas interesantes que también tienen correspondencia clásica-cuántica.

4.4.2. La transformada de Fourier

La transformada de Fourier discreta es un operador unitario muy usado en algoritmos cuánticos. Debido a que transforma las dos bases (la de posición y momento) es de esperar que tenga una representación simple en el espacio de fases. De hecho es fácil demostrar que la transformada de Fourier no es otra cosa que una rotación de 90 grados:

$$U = U_{FT} \iff W(q,p,t+1) = W(-p,q,t). \tag{4.48}$$

donde $-p$ es la inversa aditiva mod (N) de p. Nuevamente vemos que la transformada de Fourier está representada por una operación local en el espacio de fases (y, en este sentido es una operación completamente clásica en cada una de las sub-grilla de $N \times N$). Por ejemplo, si aplicamos la transformada de Fourier al estado de la figura 4.5 (un estado computacional y una superposición de dos estados computacionales) uno obtiene la función de Wigner resultante rotando la figura 90 grados (i.e., se obtiene un estado de momento y una superposición de estados de momento donde el patrón vertical es mapeado horizontalmente). También es claro que si aplicamos la transformada de Fourier dos veces obtenemos una rotación de 180 grados. Esto no es otra cosa que el operador reflexión que es equivalente a lo que nos dice la ecuación (4.18).

4.4.3. Los mapas cuadráticos (gatos) se propagan clásicamente

Las operaciones que mostramos antes (traslaciones rígidas, reflexiones y transformadas de Fourier) son algunos de los ejemplos de operadores clásicos de evolución en el espacio de fases. Una familia mas general de tales operadores se discutirá aquí. Esta familia está compuesta por las cuantizaciones de sistemas dinámicos clásicos con Hamiltonianos que son cuadráticos y que tienen un espacio de fases con condiciones periódicas de contorno (como el espacio de fases es un toro, las transformaciones clásicas son los automorfismos lineales del toro [Arnold68]). Ahora presentaremos rápidamente estos operadores unitarios y mostraremos que la evolución clásica y cuántica coinciden (algo que ya había sido demostrado por Hannay y Berry [Hannay80] utilizando técnicas totalmente distintas). Consideremos la siguiente familia de operadores con dos parámetros

$$U_{cat} = \mathcal{V}_b \mathcal{T} \mathcal{V}_a. \tag{4.49}$$

Los operadores $\mathcal{V}_a$ y $\mathcal{T}$ son diagonales en la base de posición y momento (respectivamente) y satisfacen:

$$\begin{aligned} \mathcal{V}_j|n\rangle &= \exp(-i2\pi n^2(1-j)/2N)|n\rangle \\ \mathcal{T}|k\rangle &= \exp(-i2\pi k^2/2N)|k\rangle \end{aligned} \tag{4.50}$$

donde j es un entero. El operador U_{cat} puede ser interpretado como el operador de evolución de un sistema con un Hamiltoniano en el cual el término cinético y potencial son prendidos y apagados de manera alternada. Los parámetros a y b son enteros que miden la intensidad de cada patada potencial. Es fácil encontrar el sistema clásico que corresponde a dicho operador: una forma de hacerlo tomando elementos de matriz del operador (4.49) en la base computacional y mostrar que

$$\langle n'|U_{cat}|n\rangle = K \exp(i2\pi(an^2 + bn'^2 - 2nn')/2N) \tag{4.51}$$

donde K es una constante de normalización. El exponente en ésta ecuación puede ser interpretado como la acción clásica del sistema donde n y n' son los valores finales e iniciales de la coordenadas. Por lo tanto, la ecuación clásica de movimiento correspondiente al sistema es:

$$n = bn' + p' \qquad p = (ab-1)n' + ap'. \tag{4.52}$$

El sistema clásico fue extensamente estudiado [Arnold68]: para valores enteres de a y b, es un miembro de la famosa familia de mapas "gatos" [Arnold68], i.e. todos los autmorfismos lineales del toro. El sistema es caótico cuando los auto-valores de la transformación lineal $\mathcal{M}$ que mapea $\alpha' = (n', p')$ en $\alpha = (n, p)$ como en la ecuación (4.52) son ambos reales (sino el mapa es integrable). Este es el caso si $\mathrm{Tr}\mathcal{M} = a + b > 2$ (notar que $\mathcal{M}$ tiene determinante 1). En particular, cuando $a = 2$ y $b = 1$ el mapa es el llamado mapa del gato

de Arnold ($n = n' + p'$, $p = n' + 2p$). El mapa así definido, si miramos (4.49), es simplemente una patada cinética seguida por una patada potencial donde el potencial es un oscilador de potencial invertido.

Usando lo resultados anteriores se puede ver la razón por la cual la evolución clásica y cuántica son idénticas. Para ello computemos la matriz $Z_{\alpha\beta}$ que evoluciona la función de Wigner. Para hacer el cálculo es útil primero observar que, si el operador $\hat{U}$ es el dado por (4.49), entonces la siguiente identidad debe cumplirse:

$$U_{cat}\ \hat{A}(\alpha) = \hat{A}(\mathcal{M}\alpha)\ U_{cat}, \tag{4.53}$$

donde, como antes, la transformación lineal es la dada en (4.52). Esto implica que el operador unitario mapea al operador de punto de fase en la misma manera que la dinámica clásica mapea a los puntos en el espacio de fase. Usando esto, no es difícil ver que la función de Wigner evoluciona clásicamente:

$$U = U_{cat} \iff W(\alpha, t+1) = W(\mathcal{M}^{-1}\alpha, t). \tag{4.54}$$

Nuestro resultado no es sorprendente cuando se lo ve desde la perspectiva de la mecánica cuántica ordinaria de los sistemas continuos. Es bien sabido que la evolución clásica y cuántica son idénticas en el espacio de fase si y sólo si el sistema clásico tiene un Hamiltoniano que es cuadrático en q y p. Aquí mostramos que el mismo resultado es válido en espacios de fase discretos.

4.4.4. Compuertas booleanas en el espacio de fases

La familia de mapas del gato presentada en la sección anterior, aunque amplia, no incluye operadores que son más naturales en el ámbito de la computación cuántica. Aquí analizaremos la dinámica generada por operadores que implementan operaciones booleanas y veremos cuando son clásicas y cuando cuánticas. Estas operaciones son fundamentales a la hora de implementar cualquier algoritmo cuántico. Analicemos entonces una función biyectiva: $f : Z_N \to Z_N$ (f es una permutación de N enteros). Dada f podemos definir un operador unitario $\hat{U}_f$ cuya acción en la base computacional es permutar los vectores de la misma manera que lo hace f:

$$U_f|n\rangle = |f(n)\rangle \tag{4.55}$$

La función booleana f corresponderá a algún circuito general (reversible) clásico. Mas aún, podemos asociarle a f un mapa clásico en el espacio de fase. Dicho mapa permuta las columnas en el espacio de fase de la misma manera que lo hace f, es decir, manda la franja vertical etiquetada por n a la etiquetada por $f(n)$. Ahora intentaremos ver bajo que condiciones la evolución en el espacio de fase asociada a $\hat{U}_f$ es idéntica a la clásica. La respuesta es sorprendentemente simple: el mapa cuántico es idéntico al clásico sólo si $f(n) = n + a \pmod{(N)}$, para cualquier entero a. Es decir, las permutaciones

clásica y cuántica son las mismas sólo en el caso que f sea un desplazamiento cíclico.

La demostración es simple: consideremos la acción de $\hat{U}_f$ sobre dos tipos de estados. Primero miremos cómo transforma $\hat{U}_f$ a la función de Wigner de un estado computacional (es decir, a la matriz densidad de la forma $\rho = |n_0\rangle\langle n_0|$, mostrada en la figura 4.5) Luego analizaremos como $\hat{U}_f$ modifica el término de interferencia que aparece cuando tenemos una superposición de dos estados computacionales (por ejemplo: $\rho = |n_0\rangle\langle n_1| \pm |n_1\rangle\langle n_0|$). Estos operadores forman una base completa del espacio de matrices Hermíticas por lo que mirando como evolucionan al aplicarles el operador de evolución podemos entender que pasará con estados más generales. Como el operador $\hat{U}_f$ permuta estados computacionales la función de Wigner actuando sobre cualquier estado de este tipo coincide con el caso clásico (es decir, traslada las columnas verticales como el mapa clásico $2n \rightarrow f(2n)$). Sin embargo, cuando aplicamos el operador de evolución a estados superposición el resultado es bastante distinto. Como habíamos visto, la franja de interferencia está siempre en el punto medio entre los dos estados superposición (el punto $n = n_0 + n_1$). Por lo tanto, para que el estado se transforme clásicamente debe ocurrir que el punto medio se mapee al punto medio de los estados transformados: $2f(n_1 + n_2) = f(2n_1) + f(2n_2)$. Mas aún, el estado original tiene oscilaciones con una longitud de onda que depende de la separación entre los estados ($\lambda = 2N/(n_0 - n_1)$. Por lo tanto, para que la longitud de onda permanezca invariante debe satisfacerse que: $2(n_0 - n_1) = f(2n_0) - f(2n_1)$. No es difícil convencerse que la única manera de satisfacer todas estas condidicion es que $f(n) = n + a$, es decir, la función corresponde a desplazamientos rígidos de la base computacional.

El resultado anterior se puede generalizar de la siguiente manera: definamos un operador más general $\hat{U}_{f,g}$ que tiene asociado dos funciones uno a uno de enteros a enteros (f y g) definido de la siguiente manera: $U_{f,g}|n\rangle = \exp(i2\pi g(n)/N)|f(n)\rangle$. Haciendo el mismo tipo de análisis de la sección anterior es posible mostrar que este operador actuará de la manera clásica sólo si $f(n) = n + a$, $g(n) = n + b$. En estos casos el operador no es otro que el operador traslación en el espacio de fase $\hat{T}(a, b)$. En resumen: si definimos un análogo a las compuertas clásicas los únicos operadores unitarios que generan la misma dinámica en el espacio de fase que sus análogos clásicos son los operadores de traslación.

Finalmente es interesante notar que la misma idea de la demostración que usamos para probar dichas propiedades se puede extender a sistema continuos y nos proporciona una nueva forma de entender por qué sólo los sistemas lineales evolucionan igual que los clásicos para todo tiempo.

4.4.5. Evolución no-local (cuántica) en el espacio de fase

En lineas generales, los casos presentados hasta aquí son la excepción: la mayoría de los operadores unitarios generan una evolución complicada y no-local en el espacio de fase. Como ejemplo tomemos el caso de invertir el qubit menos significativo (el operador asociado es $\sigma_x^{(0)}$). En este caso es fácil obtener:

$$U = \sigma_x^{(0)} \iff Z_{\alpha 0} = Z_{0\alpha} = 2/N \quad \text{si } \alpha \text{ es par} \tag{4.56}$$

y es cero en todos los otros casos ($\alpha = (q, p)$ es par cuando tanto q como p son pares). Los otros elementos de matriz son fáciles de calcular pero con este sólo es posible ver la naturaleza no-local de la evolución. La ecuación anterior implica que cuando $\hat{U} = \sigma_x^{(0)}$ el próximo valor de $W(0)$ es proporcional a la suma de $W(\alpha)$ sobre todos los valores pares de α lo cual es un mapa claramente no-local.

Otro ejemplo sencillo de evolución no-local es la dada por el siguiente operador $U_{FT}^{(N/2)}$ que consiste de no hacer nada con el qubit más significativo y aplicarle una transformada de Fourier al resto (es decir, una transformada sobre el espacio de dimensión $N/2$). En tal caso la evolución se parece a una rotación en 90 grados (como el caso de la transformada de Fourier completa) pero es bastante no-local. Este tipo de transformación es esencial en la construcción del mapa del panadero. Aquí sólo nos concentraremos en la descripción en espacio de fase de la misma. Clásicamente, cuando aplicamos esta operación a una franja vertical (un estado computacional), el circuito traslada la franja según $n' \to 2n(modN)$, lo contrae a la mitad de su altura y lo rota 90 grados. En la figura 4.7 se pueden ver remanentes de este comportamiento. Sin embargo, en la figura también se pueden apreciar efectos de difracción marcados que hacen que $Z_{\alpha\beta}$ sea no-local. No es difícil entender de donde vienen dichos efectos: el bit más significativo divide el espacio de fase en dos regiones y el mapa actúa de manera discontinua sobre cada una de ellas generando difracción de la misma manera que una pantalla la generaría en un sistema óptico. Esto se puede contrastar con lo que ocurre cuando se aplica el mapa del gato de Arnold al mismo estado. En ese caso, la franja vertical es transportada por el flujo clásico lineal.

4.4.6. Algoritmos cuánticos en espacio de fase

Todos los algoritmos cuánticos no son otra cosa que operadores unitarios que se diseñaron de manera tal que tras aplicarlos varias veces se genera un estado en el cual está codificada la respuesta a algún problema computacional. Midiendo dicho estado obtenemos información sobre la respuesta de dicho problema. Los algoritmos cuánticos están diseñados de manera que se comportan como una función regular con N. En muchos casos, estas regularidades deben ser manifestadas en el espacio de fases. De hecho, en el espacio de fase dicha representación es más útil en el límite de N grandes (una suerte de límite semi-clásico ya que $1/N$ se comporta como una constante de Plank efectiva). El estudio de la representación de los algoritmos en espacio de fases está recién empezando a estudiarse pero ya se pueden presentar algunos resultados interesantes. Como ejemplo discutiremos el algoritmo de búsqueda de Grover cuya representación en espacio de fases se muestra en la figura 4.8.

En la figura 4.8 se muestra la función de Wigner del estado cuántico de la computadora después de cada iteración del algoritmo de Grover. El sistema

tiene un espacio de Hilbert de dimensión $N = 32$ (es decir, la computadora tiene 5 qubits) y el algoritmo está diseñado para buscar el elemento marcado que aquí hemos tomado como $q = 10$. El estado inicial es una superposición de todos los posibles estados computacionales con igual peso (es decir, un estado de momento que hemos elegido con con momento cero). De la figura se observa inmediatamente que el algoritmos funciona de manera muy simple. El estado inicial tiene una función de Wigner que es una franja horizontal (con su franja de interferencia asociada). Después de cada iteración, $W(q,p)$ muestra un patrón de difracción para convertirse en un estado computacional puro al final de la búsqueda (en nuestro caso, después de $T \approx \pi\sqrt{N}/4 \approx 5$ iteraciones que es el óptimo para nuestro caso). Esta representación muestra que, como en el caso de un mapa, el algoritmo de Grover tiene un punto fijo en el espacio de fase con coordenada igual a la del elemento marcado ($q = 10$ en nuestro caso) y momento igual al del estado inicial.

4.5. Como medir la función de Wigner

Uno de los hallazgos mas interesantes que encontramos estudiando ésta representación en el espacio de las fases es que es simple construir un circuito cuántico para medir $W(q,p)$. No sólo es fácil medirla sino que el circuito encontrado tiene varias aplicaciones interesantes. En efecto, dicho circuito (que describimos más abajo) es similar al usado para medir auto-valores de un operador unitario $\hat{U}$ [Cleve98] y varios algoritmos pueden ser interpretados como alguna reencarnación del mismo. Como primer paso vamos a describir una estrategia general para medir la función de Wigner de un sistema cuántico (tanto discreto como continuo) propuesta en [Miquel02b]. El método es una aplicación de un método muy general para hacer tomografía y generaliza una propuesta para medir la función de Wigner en el contexto de cavidades QED

La idea básica se puede entender fácilmente a partir de un algoritmo cuántico sintetizado en un circuito cuántico simple como el de la figura 4.2. Analicemos sus propiedades: un sistema inicialmente en el estado $\hat{\rho}$ se pone en contacto con un qubit auxiliar preparado en el estado $|0\rangle$. El sistema auxiliar juega el rol de "partícula de prueba" en un experimento similar a un scattering. El algoritmo representado por el circuito es: i) Aplicar una transformación de Hadamard al qubit auxiliar, ii) Aplicar una operación "control-$\hat{U}$" (si el qubit auxiliar es $|1\rangle$ le aplica $\hat{U}$ a $\hat{\rho}$ mientras que si es $|0\rangle$ no hace nada), iii) aplicar otra Hadamard al qubit auxiliar y finalmente hace una medición *débil* del qubit auxiliar midiendo la polarización (es decir, medir el valor de expectación de σ_z y σ_y). Es fácil comprobar que el circuito tiene la siguiente propiedad:

$$\langle\sigma_z\rangle = \mathrm{Re}[\,\mathrm{Tr}\,(\hat{U}\hat{\rho})\,], \quad \langle\sigma_y\rangle = -\mathrm{Im}[\,\mathrm{Tr}\,(\hat{U}\hat{\rho})\,]. \tag{4.57}$$

De estas dos ecuaciones vemos que la medición final revela propiedades tanto del estado $\hat{\rho}$ como del operador unitario $\hat{U}$.

Es claro entonces que el circuito se puede usar para dos tareas: por un

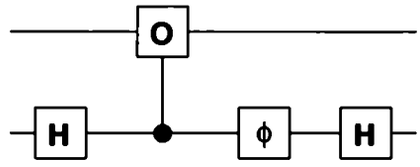


Figura 4.2: Circuito para medir Re[Tr $(U\rho)$], para un dado operador unitario $\hat{U}$.

lado se lo puede usar para extraer información sobre $\hat{U}$ si conocemos el estado inicial $\hat{\rho}$. Por el otro el mismo circuito se puede usar para conocer propiedades del estado $\hat{\rho}$ usando distintos operadores $\hat{U}$. Por ejemplo: tomando como $\hat{U}$ los distintos operadores producto podemos, mediante varios experimentos, calcular los coeficientes de expansión de $\hat{\rho}$ en la base de operadores producto. Lo mismo se puede hacer tomando el conjunto de operadores $\hat{A}(q,p)$ introducidos en el capítulo 3. En este sentido, el circuito puede ser adaptado para usarlo como un tomógrafo o como un espectrómetro.

Como dijimos, tomando $\hat{U} = 2N\hat{A}(\alpha)$ se puede medir, directamente los coeficiente de expansión de $\hat{\rho}$ en la base de operadores de espacio de fase. Los coeficientes $W(\alpha)$, no son otra cosa que el valor de la función de Wigner en un punto del espacio de fase $\alpha = (q,p)$.

Debería ser claro que este método es aplicable, en principio, tanto para sistemas continuos como para discretos. Para medir $W(\alpha)$ del estado $\hat{\rho}$ para una partícula cuántica uno tiene que correr el algoritmo con el sistema en el estado $D(\alpha)\hat{\rho}D^{\dagger}(\alpha)$ (obtenido desplazando a $\hat{\rho}$) y usando $\hat{U} = \hat{R}$ (el operador reflexión). Para medir la función de Wigner en el caso discreto es mas conveniente usar $\hat{U} = 2N\hat{A}(\alpha)$. En ambos casos el operador aplicado es unitario y hermítico y la medición se puede hacer midiendo sólo la componente z de la polarización.

Recientemente se han llevado acabo diversos experimentos para medir la función de Wigner en varias áreas de la física para sistemas continuos ([Dunn94]). Es interesante notar que el esquema presentado describe el experimento llevado acabo para medir la función de Wigner del estado del campo electromagnético en una cavidad superconductora [Lutterbacj97, Nagues00]. En ese caso el sistema a medir es un modo del campo electromagnético almacenado en una cavidad de alto Q y el bit auxiliar es un qubit formado por un átomo de dos niveles $|g\rangle$ y $|e\rangle$. Curiosamente, éste experimento se puede describir fácilmente con el circuito de la figura 4.2: i) el átomo pasa por lo zona Ramsey que tiene el efecto de implementar una transformada Hadamard. Una fuente de radiofrecuencia es conectada a la cavidad desplazando el campo en el espacio de fase (donde el desplazamiento está dado por α), ii) el átomo pasa por la cavidad interactuando dispersivamente con el campo. La interacción está diseñada de manera que si el átomo se encuentra en el estado $|g\rangle$ nada pasa mientras que si está en el estado $|e\rangle$ el estado adquiere una fase π por fotón de la cavidad. Esta interacción es simplemente un control-$exp(-i\pi\hat{N})$ (donde $\hat{N}$ es el operador número de fotones) que no es otra cosa que la "reflexión controlada". iii) El átomo deja la cavidad pasando por otra zona de Ramsey que

aplica otra compuerta Hadamard. Finalmente el átomo es detectado por un contador para determinar si está en el estado fundamental $|g\rangle$ o en el excitado $|e\rangle$. El experimento se repite varias veces y la función de Wigner se obtiene de la diferencia de la probabilidad de obtener el átomo en el estado fundamental o excitado: $W(q,p) = 2(P(e) - P(g))/\hbar$. Como vemos, el experimento con las cavidades QED es un ejemplo tomográfico del circuito de la figura 4.2. El punto importante es que el método es generalizable a sistemas arbitrarios (discretos o continuos).

Para el caso discreto uno puede demostrar que el circuito 4.2 puede ser eficientemente descompuesto como una secuencia de operaciones con compuertas elementales. Para ello sólo debemos implementar las operaciones: control-$\tilde{U}$, $\tilde{V}$ y $\hat{R}$. Todas ellas se pueden hacer utilizando compuertas como las usadas en [Miquel96] que requieren un numero polinomial función de $\log(N)$ de compuertas elementales. Para descomponer $\hat{V}$ como una secuencia de compuertas de uno y dos qubits debemos recordar su acción sobre un ket: $|n\rangle$ que es:

$$\begin{aligned} \hat{V}^p|n\rangle &= \exp(i2\pi pn/N)|n\rangle \\ &= \prod_{i=0}^{L-1} \left[\exp(i2\pi p 2^i/N)\right]^{n_i} |n\rangle \end{aligned}$$

donde n_i es la expansión binaria de n. De esta expresión vemos que le tenemos que aplicar la fase $\exp(i2\pi p 2^i/N)$ sólo si $n_i = 1$. Por consiguiente, $\hat{V}$ es simplemente una secuencia de compuertas que aplican una fase al estado dependiendo del estado de cada bit de n como se ve en el siguiente circuito:

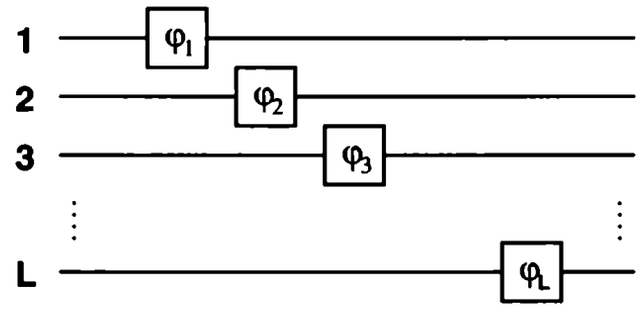


Figura 4.3: Circuito para implementar el operador $\hat{V}^{-p}$. Cada caja es una operación sobre un qubit tal que: $\phi_i|0\rangle = |0\rangle$ mientras que $\phi_i|1\rangle = \exp(i2\pi p 2^i/N)|1\rangle$.

Como el operador $\hat{U}$ no es otra cosa que un desplazamiento en la base de posición se puede usar el circuito sumador módulo N presentado en el capítulo 3 o simplemente observar que $\hat{U}$ se obtiene del circuito de $\hat{V}$ usando que $\hat{U} = U_{FT}^{-1}\hat{V}U_{FT}$. Finalmente, el operador de reflexión se puede implementar si observamos que $\hat{R} = \hat{U}_{\mathrm{FT}}^2$ (dos transformaciones de Fourier son equivalentes a una reflexión). El circuito completo se puede ver en la figura 4.4.

El procedimiento entonces para medir la función de Wigner del estado $\hat{\rho}_o$ es el siguiente: i) preparamos al sistema en estado $|0\rangle\langle 0| \otimes \hat{\rho}_o$, ii) le aplicamos

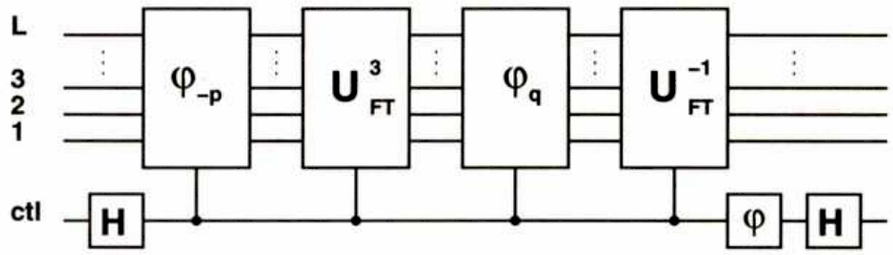


Figura 4.4: Circuito para medir la función de un estado almacenado en L qubits de una computadora cuántica. El operador ϕ_{-p} es un circuito como el de la figura 4.3 mientras que $\hat{U}_{\mathsf{FT}}$ implementa la transformada de Fourier discreta. la transformada Hadamard mientras que es una rotación alrededor de z en un ángulo ϕ.

el circuito de la figura 4.4, iii) medimos la componente z del qubit de control, iv) repetimos el procedimiento hasta tener $\langle \sigma_z \rangle = 2N\mathrm{Re}[\,\mathrm{Tr}\,(\hat{A}(q,p)\hat{\rho})\,], = W(q,p)$. Cambiando el circuito para cada uno de los puntos q, p medimos un punto de la función de Wigner. Es importante aclarar que la medición de la función de Wigner en un punto del espacio de fases es eficiente pero que la determinación completa de la función de Wigner es una tarea que es exponencial en el número de qubits (lo cual es natural ya que es equivalente a la determinación completa del estado cuántico del sistema).

El circuito introducido para medir la función de Wigner resulta muy útil y tiene numerosas aplicaciones. Como ya dijimos se puede interpretar tanto como un espectrómetro para averiguar propiedades de $\hat{U}$ si conozco el estado del sistema o como tomógrafo para medir $\hat{\rho}$ utilizando operadores $\hat{U}$ apropiados. Por ejemplo, en el estudio de sistemas caóticos cuánticos nos interesa calcular $\mathrm{Tr}\,\hat{U}^n$, es decir, trazas de potencias del operador evolución. Estas trazas guardan información acerca de la estructura de órbitas periódicas del sistema. Para ver esto basta escribir:

$$\mathrm{Tr}\,U^n = \sum_i \langle \psi_i | \hat{U}^n | \psi_i \rangle$$

donde $|\psi_i\rangle$ es alguna base del espacio de Hilbert. En palabras, esta ecuación nos está midiendo el solapamiento promedio entre algún estado $|\psi_i\rangle$ y el estado $\hat{U}^n|\psi_i\rangle$. Nos dice cuánto coinciden $|\psi_i\rangle$ con el mismo estado propagado n veces. Si hay alguna periodicidad ese número serás mas grande que si no la hay por lo tanto las trazas tienen información de las órbitas de período n.

Con el circuito general que hemos mostrado es trivial hacer esto. Es sabido como construir circuitos correspondientes a mapas caóticos. Por ejemplo, Schack mostró como implementar un circuito del mapa del panadero utilizando compuertas elementales. Si agregamos un control adicional e iniciamos al sistema en el estado $\hat{\rho} = \mathbb{I}/N$ podemos usar nuestro circuito para estudiar las trazas para el operador evolución del mapa del panadero. Esta es una magnitud que es directamente observable. Sin embargo, hacer el cálculo análogo en una computadora clásica resulta prácticamente imposible (con la tecnología de hoy) para sistemas con mas de 30 qubits. También es trivial implementar los mapas del gato introducidos en este capítulo y medir sus trazas.

4.6. Resumen

En este capítulo presentamos una representación en el espacio de fases para sistemas cuánticos con espacio de Hilbert discreto. Dicha representación resulta útil para estudiar la evolución de una computadora cuántica y sistemas caóticos cuánticos y su límite semi-clásico. Además mostramos que es posible medir dicha distribución con un circuito simple que también resulta útil para obtener tanto propiedades del espectro de un operador desconocido o hacer tomografía de estados. Aunque el circuito era conocido, su interpretación como espectrógrafo o tomógrafo nos abre numerosas puertas para investigar, por ejemplo, sistemas cuánticos caóticos con una computadora cuántica. En el próximo capítulo veremos resultados experimentales de la medición de la distribución en espacio de fase para un sistema con un espacio de Hilbert de baja dimensión.

Láminas color

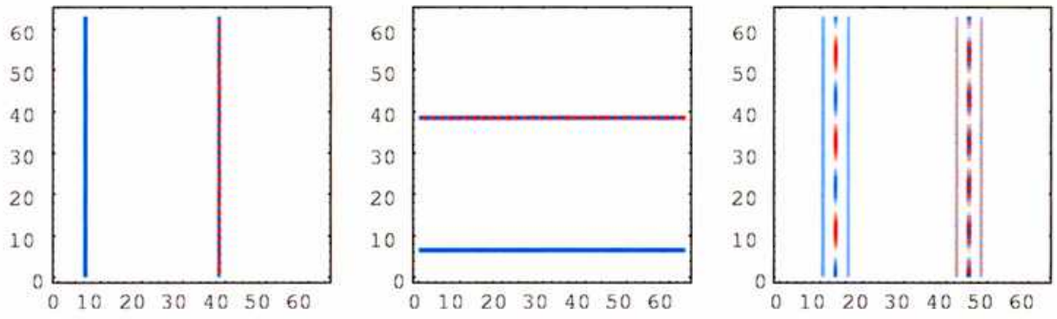


Figura 4.5: Función de Wigner de un auto-estado de posición, un estado de momento y de una superposición de auto-estados de posición. Observar en todos los casos las franjas de interferencia entre el estado y la imágen resultante de imponer condiciones periódicas de contorno. En todos los gráficos el color azul representa valores positivos de la función de Wigner mientras que rojo valores negativos.

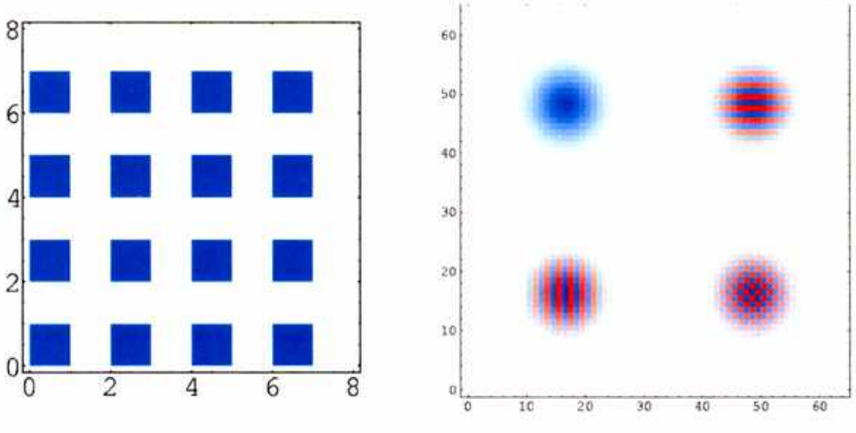


Figura 4.6: A la izquierda se puede ver la función de Wigner del estado mixto. Es distinta de cero sólo para valores de q y p pares. A la derecha se observa un estado Gaussiano. Observar las imágenes generadas por la interferencia debido a las condiciones periódicas de contorno.

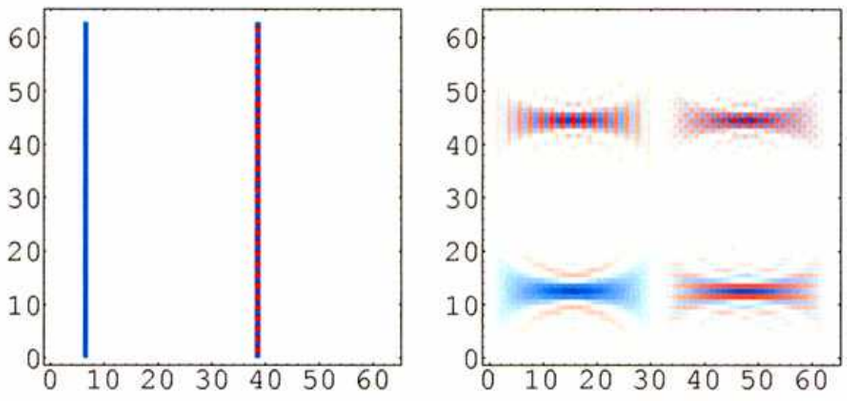


Figura 4.7: En la figura de la derecha vemos el efecto de aplicarle la transformada de Fourier $U_{FT}^{(N/2)}$ a un estado computacional. Claramente se ven los efectos de difracción que muestran que la evolución es no-local.

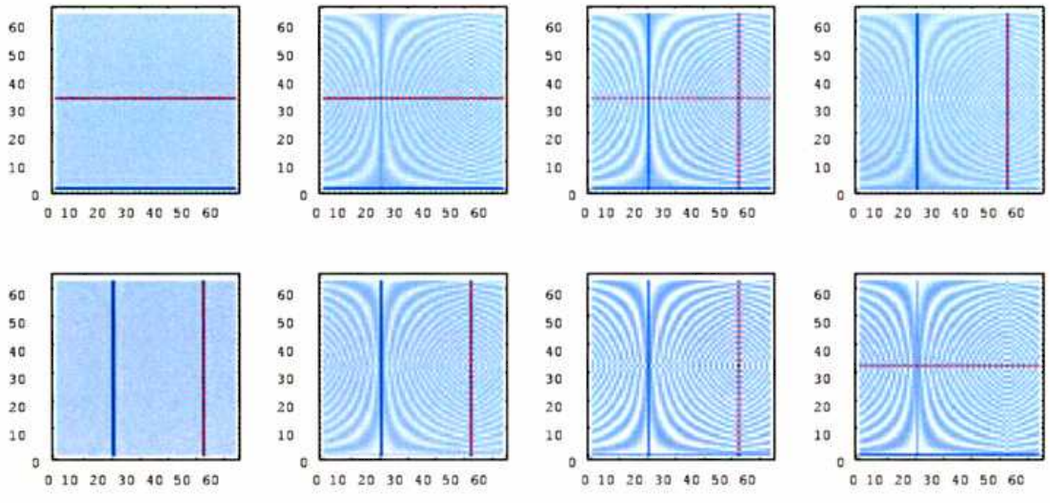


Figura 4.8: Algoritmo de Grover en el espacio de fase. Se observa como el estado inicial de momento $p = 0$ se va convirtiendo, iteración tras iteración en el estado $q = 10$ que corresponde al ítem buscado.

5. Computación Cuántica con Resonancia Magnética Nuclear

Sin duda toda los algoritmos y herramientas de procesamiento de información cuántica desarrollada en los últimos 8 años serían inútiles sino fuera posible construir una computadora cuántica. En paralelo a todos los avances teóricos, grupos experimentales alrededor del mundo han estudiado diversos sistemas para poder desarrollar el "hardware" de una computadora cuántica. DiVincenzo [DiVincenzo00] en listó de manera sistemática los requisitos para que un sistema físico sea utilizable como computadora cuántica. Estos son: i) un sistema físico para almacenar los qubits ii) la posibilidad de generar un estado inicial $|00\ldots0\rangle$. iii) tiempo de decoherencia largos comparado con el tiempo necesario para hacer una operación, iv) un mecanismo para implementar compuertas universales, y v) una forma de medir el estado de los qubits. Los sistemas estudiados que verifican los requisitos mínimos pasan de átomos/iones en trampas magneto-ópticas, arreglos de junturas Josephson en superconductores, cavidades ópticas a quantum dots y resonancia magnética nuclear (RMN) en sólidos y líquidos (Ver [Nielsen00] para una lista completa de referencias). Sin embargo, prácticamente los únicos resultados interesantes que se han obtenido han sido utilizando RMN en líquidos. Los espines de una molécula en solución están casi totalmente aislados de sus vecinos por cuestiones geométricas mientras que, debido a su difusión en el líquido, el acoplamiento dipolar entre moléculas promedia a cero. Esto los convierte en un buen lugar para almacenar y procesar información cuántica. Mediante pulsos de radiofrecuencia sintonizados a las frecuencias de resonancia de los distintos núcleos se puede manipular el estado individual de cada qubit. Para lograr hacer lógica condicional entre qubits se usa la interacción natural entre espines nucleares. La dificultad principal para usar RMN en líquidos proviene del hecho que el estado inicial de las moléculas es mixto. Afortunadamente es posible superar este obstáculo como veremos más adelante.

En este capítulo discutiremos a fondo esta implementación y presentaremos resultados experimentales llevados a cabo en el laboratorio LANAIS del Departamento de Física de la FCEyN y en Los Alamos, EEUU. Comenzaremos dando un vistazo general al área de resonancia magnética nuclear en líquidos para luego discutir las herramientas básicas para poder hacer computación cuántica con RMN. Finalmente mostramos los resultados de varios experimentos incluyendo la medición de la función de Wigner para un sistema de dos espines.

5.1. RMN en líquidos

En la naturaleza los núcleos presentan un momento dipolar $\vec{\mu}$ que es proporcional al momento angular $\vec{I}$. La constante de proporcionalidad es el factor giro-magnético (aquí γ) y depende del isótopo del núcleo. Para algunos isótopos dicha constante puede ser cero como es el caso del ^{12}C. Al introducir un núcleo en un campo magnético el espín del mismo (y por lo tanto el

momento dipolar) precederá alrededor del campo magnético B_0 con una frecuencia $\omega_0 = \gamma B_0$. Si tenemos una muestra con muchos núcleos es posible, mediante una bobina medir la corriente inducida en ella debido a la precesión de los momentos dipolares de todos los núcleos. Midiendo la frecuencia de esta corriente oscilatoria se puede determinar la frecuencia ω_0 que nos da información del tipo de núcleos presentes en la muestra. Debido a la interacción entre espines de una misma molécula es posible, midiendo la señal sobre la bobina extraer también información acerca de los acoplamientos de los espines en la molécula. Este es el principio usado en espectroscopía RMN y comprende una rama enorme tanto de la física como de la química moderna. Sin embargo, nosotros usaremos RMN en una manera muy distinta a la que están acostumbrados los espectroscopistas. En computación cuántica no nos interesa encontrar los acoplamientos entre espines sino, una vez que los conocemos, usarlos para procesar información almacenada en ellos. Para ello se utilizan moléculas simples y bien estudiadas para tener el máximo control sobre su dinámica.

A modo de ejemplo discutiremos todos los principios básicos para una molécula hipotética simple de dos núcleos: un hidrógeno ^{1}H y un carbono ^{13}C. Un ejemplo de dicha molécula es el cloroformo ($CHCl_3$) donde el carbono es reemplazado por su isótopo ^{13}C. Cada núcleo se comporta esencialmente como una partícula de espín 1/2. Como el protón y el carbono tiene factores giro-magnéticos muy distintos (el del carbono es 4 veces más chico que el del ^{1}H), sus frecuencias de resonancia serán muy distintas. En el espectrómetro que usamos, la frecuencia de resonancia del ^{1}H es aproximadamente 500 MHz mientras que la del carbono es de $\sim$ 125 MHz. Si por un momento olvidamos cualquier interacción que pueda existir entre los momentos magnéticos de los espines (y entre moléculas) y medimos la corriente que se induce en la bobina de observación aparecerían dos frecuencias características de los núcleos. En RMN es costumbre mirar la transformada de Fourier de la corriente que presentará picos en las frecuencias de resonancia del hidrógeno y carbono.

En este punto es conveniente hablar un poco acerca de las interacciones entre espines. Debemos distinguir dos tipos: las interacciones con espines de la misma molécula y con espines de otras moléculas. Empecemos con las internas. Si bien en la mayoría de las moléculas la distancia entre núcleos es grande como para que los espines interactúen de manera directa, sí lo hacen de manera indirecta mediante su interacción con los electrones que orbitan en la molécula. A esta interacción se la conoce como acoplamiento J. El origen del acoplamiento es fácil de entender: el campo generado por el momento magnético nuclear interactúa con el espín de los electrones que orbitan en la molécula. Estos electrones, a su vez, están compartidos por varios núcleos e interactúan con otros núcleos vecinos también a través del campo magnético. La otra interacción relevante es la que existe entre moléculas. En los sólidos dicha interacción es de tipo dipolar y es muy importante. Sin embargo, en los líquidos a temperatura ambiente, dicha interacción promedia a cero debido a la rápido rotación de las moléculas dentro del fluido de manera que, cada molécula está virtualmente aislada de las demás. De esta manera cuando

consideramos RMN en líquidos, lo podemos pensar como un ensamble de $N \sim 10^{24}$ moléculas no interactuantes.

Para formalizar un poco esta descripción escribamos cual es el Hamiltoniano que describe nuestra molécula hipotética. La parte que nos interesa es la nuclear y está descripta por el siguiente Hamiltoniano:

$$\mathbf{H}_{\mathrm{sc}} = \omega_1 \mathbf{I}_z^1 + \omega_2 \mathbf{I}_z^2 + 2\pi J_{12} \left(\mathbf{I}_x^1 \mathbf{I}_x^2 + \mathbf{I}_y^1 \mathbf{I}_y^2 + \mathbf{I}_z^1 \mathbf{I}_z^2\right) \tag{5.1}$$

donde $\mathbf{I}_j^k$ ($k = 1, 2$ y $j = x, y, z$) es la componente j del momento angular del espín k. Para espines 1/2: $\mathbf{I}_j = \frac{1}{2}\hbar\sigma_j$ con σ_j las matrices de Pauli. De aquí en más tomaremos $\hbar = 1$ salvo en algunos pasajes. A este tipo de interacción entre los espines se la conoce como acoplamiento fuerte cuando es de la forma: $\mathbf{I}^1 \cdot \mathbf{I}^2$. Si la diferencia entre las frecuencias de resonancia $|\omega_1 - \omega_2| \gg J_{12}$ la ecuación (5.1) es bien aproximada por el de acoplamiento débil:

$$\mathbf{H}_{\mathrm{wc}} = \omega_1 \mathbf{I}_z^1 + \omega_2 \mathbf{I}_z^2 + 2\pi J_{12}\, \mathbf{I}_z^1 \mathbf{I}_z^2 \tag{5.2}$$

Éste Hamiltoniano es mucho más simple por dos motivos: por un lado es automáticamente diagonal en la base de los auto-estados de $\mathbf{I}_z$ y por lo tanto no es necesario diagonalizarlo. La segunda ventaja es que los términos conmutan entre sí y resulta más fácil escribir el operador evolución temporal. En el caso que estamos considerando el Hamiltoniano es una excelente aproximación: la diferencia de frecuencias entre el ^{1}H y ^{13}C es del orden de los 300 MHz mientras que el acoplamiento típico entre protón y carbono esta en los cientos de Hz.

Conocido el Hamiltoniano de cada molécula se puede analizar cual es la dinámica de los estados. Como no es función explícita del tiempo el operador de evolución se obtiene exponenciando el Hamiltoniano: $\mathbf{U}(t) = \exp(-i\mathbf{H}t)$. Debido a que los términos de $\mathbf{H}$ conmutan, la evolución se puede descomponer en tres operaciones simples: dos rotaciones alrededor de z con ángulos $\omega_1 t$ y $\omega_2 t$ y una rotación en torno al "eje" $\mathbf{I}_z^1 \mathbf{I}_z^2$ en un ángulo $\pi J_{12} t$. Más adelante explicaremos esta observación un poco más.

5.2. Formalismo de operadores producto (FOP)

5.2.1. Describiendo estados cuánticos

Antes de poder ver cómo evolucionan estados debemos hablar acerca de como representar el estado de los espines. En mecánica cuántica, se acostumbra usar la matriz densidad de un sistema para describir su estado. Esta representación permite describir, además de un único sistema (dado por un rayo en el espacio de Hilbert) a una mezcla estadística como es nuestro ensamble de moléculas. En principio, para describir un sistema de $N \sim 10^{24}$ moléculas, se necesitaría una matriz densidad que es el producto tensorial de N matrices ρ_i: una por cada molécula. Sin embargo, como las moléculas no interactúan

es claro que sólo es necesario estudiar la evolución de la matriz densidad de una de ellas. Si además nuestras moléculas están en equilibrio termodinámico la matriz densidad que describe a nuestro ensamble es:

$$\rho = \mathrm{e}^{-\mathbf{H}/kT}/Z. \tag{5.3}$$

donde $Z = \mathrm{Tr}\rho$ es la normalización. Para el caso de nuestra molécula hipotética, ρ se puede escribir como:

$$\rho \;\alpha\; \mathrm{e}^{-(\omega_1 \mathbf{I}_z^1 + \omega_2 \mathbf{I}_z^2 + 2\pi J_{12}\, \mathbf{I}_z^1 \mathbf{I}_z^2)/kT}. \tag{5.4}$$

$$= 1 - \frac{\omega_1}{kT}\mathbf{I}_z^1 - \frac{\omega_2}{kT}\mathbf{I}_z^2 - \frac{2\pi J_{12}}{kT}\,\mathbf{I}_z^1\mathbf{I}_z^2 + \ldots \tag{5.5}$$

$$\sim 1 - \frac{\omega_1}{kT}\mathbf{I}_z^1 - \frac{\omega_2}{kT}\mathbf{I}_z^2. \tag{5.6}$$

para los campos magnéticos de los espectrómetros modernos. Al operador $\rho - \mathbf{1}$ se lo suele denominar matriz densidad desviación. Este operador es típicamente muy pequeño. En efecto la anterior es una excelente aproximación porque $\omega/kT = \gamma B_0/kT$ es aproximadamente una parte en un millón y además $\omega \gg J_{12}$. Esto quiere decir que la matriz densidad es prácticamente la identidad más una pequeña desviación. Sin embargo, es importante notar que la señal que se mide en los espectrómetros proviene, precisamente de la desviación de la identidad. Es claro que una matriz proporcional a la identidad no tiene ninguna dirección de polarización y por lo tanto no genera una señal detectable por lo que de ahora en más omitiremos ese término. Finalmente es conveniente omitir los factores ω_k/kT pues son simplemente un factor de escala sin importancia por lo que al operador densidad inicial de nuestro sistema lo escribiremos simplemente como: $\mathbf{I}_z^1 + \mathbf{I}_z^2$. Los espectroscopistas de RMN suelen escribir la matriz densidad como combinaciones lineales de "operadores producto": es decir, productos tensoriales de $\mathbf{I}_x^k$, $\mathbf{I}_y^k$ e $\mathbf{I}_z^k$. Cualquier matriz se puede expandir como una combinación lineal de estos productos tensoriales más la matriz identidad.

5.2.2. Evolución de operadores producto

La ventaja de usar esta notación es que es posible interpretar la evolución ante pulsos de radio-frecuencia (rf) y la evolución libre de una manera muy simple e intuitiva. Para empezar veamos cómo cambia un término I_x^1 cuando lo rotamos en un ángulo $\theta = \omega_1 t$ alrededor del eje z. El operador evolución es entonces: $\mathbf{U}(t) = \exp(-i\omega_1 t\, \mathbf{I}_z^1)$. Ante cualquier transformación unitaria, la matriz densidad evoluciona como $\rho \rightarrow \mathbf{U}(t)\rho\mathbf{U}^\dagger(t)$. Si el estado del sistema es $\rho = \mathbf{I}_x^1$ es fácil hacer ésta cuenta y resulta,

$$\mathbf{I}_x^1 \xrightarrow{[\theta]_z^1} \cos(\theta)\mathbf{I}_x^1 + \sin(\theta)\mathbf{I}_y^1. \tag{5.7}$$

De esta expresión vemos que $\mathbf{I}_x^1$ rota en torno al eje z en un ángulo θ como esperábamos intuitivamente (y como surge, además de observar que $\mathbf{I}^1$ y $\mathbf{I}^2$ son operadores vectoriales). En general, si tenemos un operador $\mathbf{I}_j$ y le aplicamos $\mathbf{U}(t) = \exp(-i\,\theta\,\mathbf{I}_k)$ el estado evoluciona de la siguiente manera:

$$\mathbf{I}_j \xrightarrow{[\theta]_k} \cos(\theta)\mathbf{I}_j + i\sin(\theta)[\mathbf{I}_k, \mathbf{I}_j] \tag{5.8}$$

siempre que $\mathbf{I}_j$ y $\mathbf{I}_k$ no conmuten. También es fácil calcular cómo evolucionan términos que son productos de dos o más operadores de espín. Definamos los siguientes productos de operadores: $\mathbf{C}_q = \{\mathbf{I}_j, 2\mathbf{I}_j\mathbf{I}_k, 4\mathbf{I}_j\mathbf{I}_k\mathbf{I}_l, \ldots\}$ entonces, si le aplicamos $\mathbf{U}(t) = \exp(-i\,\theta\,\mathbf{C}_q)$ a un estado $\mathbf{C}_p$, éste evoluciona según:

$$\mathbf{C}_p \xrightarrow{[\theta]\mathbf{C}_q} \cos(\theta)\mathbf{C}_p + i\sin(\theta)[\mathbf{C}_q, \mathbf{C}_p]. \tag{5.9}$$

Usando estas reglas de evolución podemos ver como cambia, por ejemplo, el estado $\mathbf{I}_x^1 + \mathbf{I}_z^2$: es decir, el espín 1 está en el planos x-y mientras que el espín 2 está en la dirección z. Como el estado no conmuta con el Hamiltoniano libre H_{wc} el mismo variará en el tiempo. Para calcular su evolución usaremos el hecho que los tres términos de (5.2) conmutan por lo que descomponemos la evolución en tres operaciones independientes: $\exp(-i\omega_1 t\mathbf{I}_z^1)$, $\exp(-i\omega_2 t\mathbf{I}_z^2)$ y $\exp(-iJ_{12}t\mathbf{I}_z^1\mathbf{I}_z^2)$. Veamos por separado cada término al evolucionarlo por un tiempo t con el Hamiltoniano libre. Como $\mathbf{I}_z^2$ conmuta con $\mathbf{H}_{wc}$ este estado no evoluciona asique:

$$\mathbf{I}_z^2 \xrightarrow{\mathbf{H}_{wc}t} \mathbf{I}_z^2.$$

El otro estado es un poco más complicado pero resulta:

$$\begin{aligned}
\mathbf{I}_x^1 &\xrightarrow{\omega_2 t\,\mathbf{I}_z^2} \mathbf{I}_x^1 \\
&\xrightarrow{2\pi J_{12}t\,\mathbf{I}_z^1\mathbf{I}_z^2} \cos\pi J_{12}t\,\mathbf{I}_x^1 + \sin\pi J_{12}t\,\mathbf{I}_y^1\mathbf{I}_z^2 \\
&\xrightarrow{\omega_1 t\,\mathbf{I}_z^1} \cos 2\pi J_{12}t\,(\cos\omega_1 t\,\mathbf{I}_x^1 + \sin\omega_1 t\,\mathbf{I}_y^1) + \\
&\qquad + \sin 2\pi J_{12}t\,(\cos\omega_1 t\,\mathbf{I}_y^1 - \sin\omega_1 t\,\mathbf{I}_x^1)\mathbf{I}_z^2
\end{aligned}$$

De esta expresión se ve que, a partir de un estado que se representaba por dos operadores producto: $\mathbf{I}_x^1$ y $\mathbf{I}_z^2$ obtenemos un estado con 5 términos: $\mathbf{I}_x^1$, $\mathbf{I}_y^1$, $\mathbf{I}_z^2$, $\mathbf{I}_x^1\mathbf{I}_z^2$ y $\mathbf{I}_y^1\mathbf{I}_z^2$. Poniendo $J = 0$ en esta expresión vemos que:

$$\mathbf{I}_x^1 + \mathbf{I}_z^2 \xrightarrow{\mathbf{H}_{wc}t} \cos\omega_1 t\,\mathbf{I}_x^1 + \sin\omega_1 t\,\mathbf{I}_y^1 + \mathbf{I}_z^2,$$

es decir, el espín 1 precede con frecuencia ω_1 mientras que el segundo espín no evoluciona. Si tenemos acoplamiento entre espines vemos que su efecto es

el de modular con un término $\cos(\pi J t)$ a la rotación y además aparecen dos operadores producto más. Luego veremos cómo es la señal que medimos en el espectrómetro debido a esta modulación.

5.2.3. Acoplamiento con pulsos de radio frecuencia

Éste formalismo es muy útil entonces para seguir la evolución de estados cuando el sistema evoluciona libremente (con el Hamiltoniano de la molécula). Pero también se puede usar para describir otra operación que se hace frecuentemente en los espectrómetros: pulsos de radiofrecuencia (rf). Es claro que como el estado inicial (térmico) esta polarizado en la dirección z y el Hamiltoniano de evolución conmuta con $\mathbf{I}_z^i$ entonces el estado inicial no cambiaría. Debemos encontrar alguna manera de convertir el estado que está polarizado en la dirección z al plano transversal para que esta magnetización preceda en torno al campo magnético B. Para ello se utilizan pulsos de rf que están en resonancia con alguna de las frecuencias ω_i. Un campo magnético oscilante de frecuencia ω polarizado en la dirección x introduce un término al Hamiltoniano libre $\mathbf{H}_{wc}$ de la forma: $\mathbf{V}(t) = \Omega_1 \cos(\omega t + \phi)\mathbf{I}_x^1 + \Omega_2 \cos(\omega t + \phi)\mathbf{I}_x^2$ de manera que el sistema evolucionará con un Hamiltoniano dependiente del tiempo $\mathbf{H}(t) = \mathbf{H}_{wc} + \mathbf{V}(t)$. En el caso de espines 1/2 se puede obtener una solución aproximada muy buena cuando ω es cercana a alguna de las ω_i. Si pasamos a la representación interacción (es decir: $\rho_I = \mathbf{U}_0^\dagger(t)\rho\mathbf{U}(t)$ donde $\mathbf{U}_0(t) = \exp[-i(\omega_1 \mathbf{I}_z^1 + \omega_2 \mathbf{I}_z^2)\, t]$ se puede demostrar que la evolución de ρ_I en la aproximación resonante ($\omega = \omega_1$) está dada por el siguiente Hamiltoniano:

$$\mathbf{V}_I = 2\pi J_{12}\mathbf{I}_z^1\mathbf{I}_z^2 + \Omega_1(\cos\phi\, \mathbf{I}_x^1 + \sin\phi\, \mathbf{I}_y^1) \tag{5.10}$$

y la ecuación de evolución para ρ_I queda:

$$\dot{\rho}_I = -i[\mathbf{V}_I, \rho_I]$$

Todavía se puede hacer una última aproximación: en la práctica, la frecuencia Ω_1 (que es proporcional a la intensidad del campo oscilatorio que genera el pulso) es mucho mayor que J. Típicamente $J \sim 100$ Hz mientras que, en un espectrómetro moderno $\Omega_1 \sim 25\text{kHz} \gg J$ por lo que durante el pulso de rf se puede considerar que la única evolución es la dada por el pulso y que no hay evolución J. Si la fase ϕ del pulso es 0 y encendemos el campo por un tiempo t el efecto sobre ρ_I es el de producir una rotación en torno al eje x en un ángulo $\Omega_1 t$. Si el tiempo durante el cual se aplica el pulso de rf es tal que $\Omega_1 t_{1/2} = \pi/2$ se obtiene una rotación en un ángulo de 90 grados. A este tipo de pulsos se lo conoce simplemente como "pulso $\pi/2$". También usaremos pulsos de 180 grados para cambiar la dirección de un espín. Más adelante veremos cómo se puede "apagar" las interacciones J_{ij} mediante tiempos de evolución libre y pulsos π.

5.2.4. Diseñando Hamiltonianos

Una de las características que hacen de RMN una implementación tan atractiva es el hecho que es posible, a partir del Hamiltoniano natural del sistema (que depende del tipo de molécula), obtener un *Hamiltoniano efectivo* distinto del original. Para entender cómo podemos hacer ésto veamos primero un ejemplo sencillo con la molécula de 2 qubits. El Hamiltoniano era:

$$\mathbf{H} = \omega \mathbf{I}_z^1 + \omega_2 \mathbf{I}_z^2 + 2\pi J_{12} \mathbf{I}_z^1 \mathbf{I}_z^2. \tag{5.11}$$

El operador de evolución temporal es entonces: $\mathbf{U}(t) = \exp(-i\mathbf{H}t)$. Como ya lo habíamos mencionado antes, el efecto de propagar un estado con dicho operador es equivalente a evolucionarlo con $\exp(-i\omega_1 t \mathbf{I}_z^1)$, luego con $\exp(-i\omega_2 t \mathbf{I}_z^2)$, y finalmente con $\exp(-iJ_{12}t2\mathbf{I}_z^1\mathbf{I}_z^2)$ (debido a que los términos del Hamiltoniano conmutan entre sí). Analicemos entonces la siguiente secuencia de operaciones: i) aplicar un pulso π alrededor de x sobre el espín 1, ii) evolucionar por un tiempo $t_1 = \alpha\tau$ con $\mathbf{U}(\alpha\tau)$ donde $0 \leq \alpha \leq 1$ y finalmente iii) aplicar un pulso final $-\pi$ alrededor de x sobre 1. No es difícil convencerse que el efecto neto de esta secuencia es evolucionar al sistema por un tiempo t_1 [1] con un Hamiltoniano efectivo que es equivalente a reemplazar $\mathbf{I}_z^1$ por $-\mathbf{I}_z^1$ en el Hamiltoniano original. Ahora combinemos esta evolución con una evolución libre que dura $t_2 = (1-\alpha)\tau$. La evolución completa está dada por un operador de evolución efectiva que es:

$$\begin{aligned} \mathbf{U}(\tau) &= e^{(-i(\omega \mathbf{I}_z^1 + \omega_2 \mathbf{I}_z^2 + 2\pi J_{12}\mathbf{I}_z^1\mathbf{I}_z^2)t_2)}\, e^{-i(-\omega \mathbf{I}_z^1 + \omega_2 \mathbf{I}_z^2 - 2\pi J_{12}\mathbf{I}_z^1\mathbf{I}_z^2 .)t_1} \\ &= e^{-i(\omega \mathbf{I}_z^1(t_2 - t_1) + \omega_2 \mathbf{I}_z^2(t_1+t_2) + 2\pi J_{12}\mathbf{I}_z^1\mathbf{I}_z^2(t_2-t_1)} \\ &= e^{-i((1-2\alpha)\omega \mathbf{I}_z^1 + \omega_2 \mathbf{I}_z^2 + 2\pi(1-2\alpha)J_{12}\mathbf{I}_z^1\mathbf{I}_z^2)\tau} \end{aligned} \tag{5.12}$$

La expresión anterior es muy interesante porque muestra que la evolución neta es la que se obtendría si propagamos al sistema por un tiempo τ con el Hamiltoniano efectivo:

$$\tilde{H} = 1 - 2\alpha)\omega(\mathbf{I}_z^1 + \omega_2 \mathbf{I}_z^2 + 2\pi(1-2\alpha)J_{12}\mathbf{I}_z^1\mathbf{I}_z^2$$

Es decir, hemos logrado cambiar la frecuencia ω_1 y la constante de acoplamiento J_{12} a $\tilde{\omega}_1 \equiv (1-2\alpha)\omega_1$ y $\tilde{J}_{12} \equiv (1-\alpha)J_{12}$ respectivamente. En particular, si hacemos el pulso justo a la mitad de la evolución ($\alpha = 1/2$) vemos que tanto $\tilde{\omega}$ como $\tilde{J}_{12}$ son cero! A este procedimiento se lo conoce como "reenfoque" de la evolución del espín 1. Mediante este ingenioso truco es posible anular el acoplamiento entre los espines 1 y 2.

En el caso general de tener más de 2 espines se pueden combinar pulsos de reenfoque en los distintos espines para apagar y prender distintos términos del Hamiltoniano. Así por ejemplo, es posible apagar todos los acoplamientos J salvo uno y hacer compuertas entre dos espines en moléculas de más de dos núcleos.

[1] Como siempre hemos despreciado el tiempo de duración de los pulsos

5.2.5. Observando los espines

Ahora que sabemos cómo describir el estado de nuestros espines, su evolución libre en un campo magnético intenso $B_0\hat{z}$ y cómo evolucionan al someterlos a pulsos de rf falta ver cómo observamos su estado. Para medir el estado de los espines se utiliza una bobina sobre la que se genera una f.e.m. debido a la variación de flujo generado por la precesión de los espines cuando giran en el plano x-y. El campo generado por un espín precediendo alrededor del eje z es proporcional a su momento dipolar $\vec{\mu}$. Si ponemos una bobina para medir dicha variación, el flujo que atraviesa a la bobina variará con el tiempo cuando el espín preceda en el campo. Se puede demostrar que la corriente inducida en la bobina es proporcional al valor medio de la magnetización transversal, o sea:

$$I(t) \; \alpha \; \mathrm{Tr}[\,\rho(t)\,(\mathbf{I}_+^1 + \mathbf{I}_+^2)] \tag{5.13}$$

donde $\mathbf{I}_\pm^k = \mathbf{I}_x^k \pm i\mathbf{I}_y^k$. A modo de ejemplo, estudiemos qué observaríamos en el espectrómetro después de hacer un pulso de 90 grados, frecuencia $\omega = \omega_1$ y fase $\phi = 90$ (pulso en torno al eje y) al estado inicial térmico. Si partimos de $\mathbf{I}_z^1 + \mathbf{I}_z^2$, después del pulso de 90 obtendríamos el estado: $\mathbf{I}_x^1 + \mathbf{I}_z^2$ cuya evolución calculamos en (5.10). Es fácil ver que si tomamos la traza con $\mathbf{I}_+^k$ los únicos términos que sobreviven son los que tienen $\mathbf{I}_{x,y}^k$. Todos los demás no aportan a la señal observada en la bobina por lo que la corriente inducida resulta:

$$\begin{aligned} I(t) &= I_0 \, \cos\pi J_{12}t \, \cos\omega_1 t + i \cos\pi J_{12}t \, \sin\omega_1 t \\ &= I_0 \cos\pi J_{12}t \; \mathrm{e}^{i\omega_1 t} \\ &= \frac{1}{2} I_0 \left[\mathrm{e}^{i(\omega_1 + \pi J_{12})t} + \mathrm{e}^{i(\omega_1 - \pi J_{12})t}\right] \end{aligned}$$

De aquí podemos ver que el estado $\mathbf{I}_x^1 + \mathbf{I}_z^2$ produce un espectro con dos picos centrados en la frecuencia $\nu_1 = \omega_1/2\pi$ separados por una distancia J_{12} (el primero en $\nu_1 - J_{12}/2$ y el otro en $\nu_1 + J_{12}/2$). En la frecuencia de resonancia ω_2 no observamos nada. En la figura 5.1. podemos ver algunos espectros generados por estados parecidos al anterior. En la realidad, la corriente aparece multiplicada por una exponencial decreciente debido a la pérdida de coherencia de los espines. Este fenómeno hace que los picos medidos experimentalmente sean Lorentzianas con un ancho de linea que dependerá de distintos fenómenos que hacen que el sistema pierda coherencia. El problema principal lo aporta la inhomogeneidad del campo magnético $B_0\hat{z}$ que hace que, en distintas partes de la muestra los espines precedan a distintas frecuencias. Afortunadamente este problema se puede solucionar mediantes pequeñas bobinas dentro del espectrómetro que se ajustan para homogeneizar el campo.

Dado el estado ρ expandido en la base de operadores producto se puede

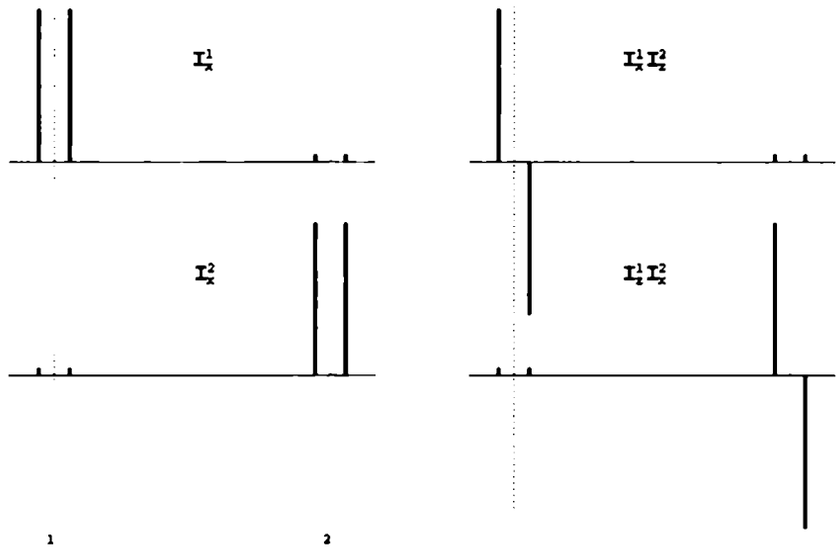


Figura 5.1: Espectros generados por distintos operadores producto: $\mathbf{I}_x^1$, $\mathbf{I}_x^2$, $\mathbf{I}_x^1\mathbf{I}_z^2$ y $\mathbf{I}_z^1\mathbf{I}_x^2$.

determinar qué se observará en el espectrómetro de manera relativamente fácil. Para ello observemos que

$$\begin{aligned} I(t) &= \mathrm{Tr}[(\mathbf{I}_+^1 + \mathbf{I}_+^2)\rho(t)] \\ &= \mathrm{Tr}[(\mathbf{I}_+^1 + \mathbf{I}_+^2)\mathbf{U}(t)\rho(0)\mathbf{U}^\dagger(t)] \\ &= \mathrm{Tr}[\mathbf{U}^\dagger(t)(\mathbf{I}_+^1 + \mathbf{I}_+^2)\mathbf{U}(t)\rho(0)] \\ &= \mathrm{Tr}[(\mathbf{I}_+^1(t) + \mathbf{I}_+^2(t))\rho(0)] \end{aligned}$$

donde $\rho(0)$ es el estado de la muestra justo antes de empezar a medir la corriente sobre la bobina receptora. Usando las reglas de evolución mostradas arriba podemos calcular $\mathbf{I}_+^1(t)$ y lo podemos escribir de la siguiente manera:

$$\mathbf{I}_+^1(t) = \mathrm{e}^{-i\omega_1 t}\left[\cos\pi Jt\,\mathbf{I}_x^1 + i\cos\pi Jt\,\mathbf{I}_y^1 + \sin\pi Jt\,\mathbf{I}_y^1\mathbf{I}_z^2 - i\sin\pi Jt\,\mathbf{I}_x^1\mathbf{I}_z^2\right] \quad (5.14)$$

y obtenemos una expresión muy similar para $\mathbf{I}_+^2(t)$. Si ahora reemplazamos sin() y cos() por su expansión en términos de exponenciales obtenemos la siguiente expresión:

$$\begin{aligned} \mathbf{I}_+^1(t) = \frac{1}{2}\Big\{ &\left(\mathrm{e}^{-i(-Jt/2)} + \mathrm{e}^{-iJt/2}\right)\mathbf{I}_x^1 + \left(-\mathrm{e}^{-i(-Jt/2)} + \mathrm{e}^{-iJt/2}\right)\mathbf{I}_x^1\mathbf{I}_z^2 + \\ &+i\left[\left(\mathrm{e}^{-i(-Jt/2)} + \mathrm{e}^{-iJt/2}\right)\mathbf{I}_y^1 + \left(-\mathrm{e}^{-i(-Jt/2)} + \mathrm{e}^{-iJt/2}\right)\mathbf{I}_y^1\mathbf{I}_z^2\right]\Big\}\mathrm{e}^{-i\omega_1 t} \end{aligned}$$

La ventaja de esta expresión es que ahora podemos ver fácilmente qué se observa cuando calculamos la transformada de Fourier de la traza de $\mathbf{I}_+^1(t)$ y $\rho(0)$. Supongamos que expandimos $\rho(0)$ en la base de productos de operadores:

$$\rho(0) = a_0 \mathbf{I}_x^1 + a_1 \mathbf{I}_y^1 + a_3 \mathbf{I}_z^1 + a_4 \mathbf{I}_x^1 \mathbf{I}_z^2 + \ldots \tag{5.15}$$

entonces, cuando hagamos el producto de $\rho(0)$ con $\mathbf{i}_+^1(t)$ y tomemos la traza, sólo quedarán 4 términos de la expansión de $\rho(0)$: $\mathbf{i}_x^1$, $\mathbf{i}_y^1$, $\mathbf{i}_x^1\mathbf{i}_z^2$ y $\mathbf{i}_y^1\mathbf{i}_z^2$. todos los demás términos darán cero debido a que la traza del producto de dos operadores producto *distintos* es cero. el espectro de fourier de la corriente que observaremos es la suma de los 4 espectros posibles multiplicados por su correspondiente peso a_i. los espectros están centrados en la frecuencia de resonancia ω_1 y tienen dos picos en $\omega_i - 2\pi j/2$ y $\omega_i + 2\pi j/2$. en el término $\mathbf{i}_x^1$ los dos picos tienen el mismo signo mientras que en el término $\mathbf{i}_x^1\mathbf{i}_z^2$ los picos tienen distinto signo. la figura 5.1 muestra todos los espectros correspondientes a cada término.

El resultado anterior es muy importante porque nos da una forma de hacer tomografía del estado y nos permite medir experimentalmente la matriz densidad. Observando la transformada de Fourier de la corriente podemos medir 4 coeficientes de la expansión de $\rho(0)$. Para obtener los otros coeficientes se debe someter la matriz densidad final a una secuencia simple de pulsos para transformar el operador producto que deseamos medir a uno de los 4 observables. Supongamos que queremos medir el coeficiente de expansión que acompaña al término $\mathbf{I}_z^1\mathbf{I}_z^2$. Ése término no es observable directamente pues $\mathrm{Tr}[\mathbf{I}_z^1\mathbf{I}_z^2 \cdot \mathbf{I}_+^1(t)] = \mathrm{Tr}[\mathbf{I}_z^1\mathbf{I}_z^2 \cdot \mathbf{I}_+^2(t)] = 0$. Sin embargo, si antes de empezar a medir hacemos un pulso de $\pi/2$ sobre el primer espín, ese término se transformará en $\mathbf{I}_x^1\mathbf{I}_z^2$ que *si* es observable. De esta manera podemos medir, mediante sucesivos experimentos todos los coeficientes de expansión de $\rho(0)$ y asi determinar el estado cuántico de nuestros espines (salvo la parte que es proporcional a la identidad).

EXPANSIONES DE OPERADORES

Otra forma de entender la expansión en productos de operadores y también la versión de la función de Wigner discreta es pensar que cualquier operador se puede expandir en una base de operadores. En el caso de la función de Wigner, dicha base está formada por los N^2 operadores $\{A(\alpha)\}$ mientras que en el caso del formalismo usado en RMN los operadores son la matriz identidad y las N^2-1 productos de operadores posibles $\{\mathbf{C}_\alpha\}$. Estas bases forman un espacio vectorial y sirven para expandir cualquier "vector" del espacio (aunque cada vector es en realidad un operador). En este espacio vectorial podemos definir el producto escalar de dos elementos. Dados dos vectores $\mathbf{A}$ y $\mathbf{B}$ su producto escalar es:

$$< \mathbf{A}, \mathbf{B} > \equiv \mathrm{Tr}[\mathbf{AB}]$$

con esta definición vemos que las bases generadas tanto por $\hat{A}(\alpha)$ como por los $\mathbf{C}_\alpha$ son bases ortogonales y por lo tanto podemos calcular explícitamente los coeficientes de expansión usados para expandir un elemento arbitrario ρ del espacio. Si escribimos:

$$\rho = \sum_\alpha a_\alpha \mathbf{A}_\alpha$$

entonces multiplicando por $\mathbf{A}_\beta$ y tomando la traza,

$$\begin{aligned} < \mathbf{A}_\beta, \rho > &= \sum_\alpha a_\alpha < \mathbf{A}_\beta, \mathbf{A}_\alpha > \\ &= \sum_\alpha a_\alpha \delta_{\alpha,\beta} \\ &= a_\beta, \end{aligned}$$

es decir, los coeficientes de expansión son la traza de ρ con cada elemento de la base. También es interesante destacar que si la base es hermítica, los coeficientes son reales como es el caso de las dos bases discutidas aquí.

5.3. Computación cuántica con estados mixtos

El principal problema de usar RMN para computación cuántica es que el sistema se encuentra en un estado mixto. Una forma de ver la dificultad que esto trae es pensar que en vez de tener una computadora cuántica tenemos un conjunto enorme de ellas, cada una preparada en un estado inicial *distinto* (de un conjunto finito de estados posibles). Esto quiere decir, que, cuando hagamos evolucionar a este sistema cada computadora está haciendo el mismo cálculo sobre condiciones iniciales distintas. Al final del día, cuando queremos inspeccionar el estado final del cálculo no podremos distinguir qué respuesta viene de qué estado inicial. Una alternativa es intentar poner a todos nuestros espines en el mismo estado inicial. Esto se puede lograr disminuyendo la temperatura de manera que el estado mixto tienda a un estado en el cual todas las moléculas están en el estado de menor energía: todas alineadas con el campo externo B_0. Desafortunadamente esto no se puede hacer en líquidos porque al disminuir T el líquido se congelaría. En este punto, la interacción dipolar entre moléculas ya cobra importancia y esto hace mucho más difícil trabajar con este sistema.

5.3.1. Método de promedio temporal

Afortunadamente, numerosos investigadores encontraron distintas maneras de generar estados que, si bien no son puros, para todos los fines prácticos se comportan como tales. Probablemente el método más simple de entender es el llamado "promedio temporal" [Knill98b]. Supongamos que queremos usar nuestro ensamble de moléculas de $CHCl_3$ para implementar en un algoritmo cuántico que empieza con el estado puro $|00\rangle$. Veamos cómo se lograr esto con un ensamble que está en equilibrio térmico. La matriz densidad del sistema es, escrita en la base $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$ y a temperatura ambiente (de manera que $\hbar\omega/kT \sim 10^{-5} \ll 1$):

$$\rho_1 = \frac{1}{4}\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} + 10^{-5}\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0.6 & 0 & 0 \\ 0 & 0 & -0.6 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}. \tag{5.16}$$

El primer término es proporcional a la identidad y por lo tanto no es observable. El segundo término es lo que llamamos la matriz densidad desviación. El objetivo es conseguir que esta matriz tenga los elementos nulos salvo el elemento $\langle 00|\rho_{dev.}|00\rangle$. Para lograrlo haremos tres experimentos. En cada experimento prepararemos un estado inicial distinto para luego aplicarle el algoritmo que queremos implementar. Al final del algoritmo promediaremos los resultados finales obtenidos. La idea es elegir los estados iniciales para que la computación generada por los tres estados no deseados $|01\rangle$, $|10\rangle$ y $|11\rangle$ promedie a cero. Vamos a poder hacer esto debido a la linealidad de todas las operaciones que hacemos en el espectrómetro.

Las tres preparaciones consisten en, no hacer nada con el estado ρ_1, aplicarle una secuencia de pulsos que implementen el operador $\hat{P}_1$ que permuta cíclicamente los estados $|01\rangle$, $|10\rangle$ y $|11\rangle$ una vez y el último consisten en aplicar $\hat{P}_1$ dos veces (que es equivalente a $P_1^2 = P^{-1} \equiv P_2$). Los tres estados iniciales generados son $\hat{\rho}_1$, $\hat{\rho}_2$:

$$\hat{\rho}_2 = \hat{P}_1\hat{\rho}_1\hat{P}_1^\dagger = \frac{1}{4}\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} + 10^{-5}\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -0.6 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0.6 \end{bmatrix}. \tag{5.17}$$

y el tercero consiste en aplicar la permutación inversa a $\hat{P}$ ($\hat{P}_2 = \hat{P}^\dagger$):

$$\hat{\rho}_3 = \hat{P}_2\hat{\rho}_1\hat{P}_2^\dagger = \frac{1}{4}\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} + 10^{-5}\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0.6 & 0 \\ 0 & 0 & 0 & -0.6 \end{bmatrix}. \tag{5.18}$$

El promedio de los tres estados resulta:

$$\begin{aligned}
\tilde{\rho} &= \frac{1}{3}\sum_i P_i \rho P_i^\dagger \\
&= \frac{1}{4}\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} + 10^{-5}\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -0.333 & 0 & 0 \\ 0 & 0 & -.333 & 0 \\ 0 & 0 & 0 & -.333 \end{bmatrix} \\
&= (\frac{1}{4} - 0.333 \cdot 10^{-5})\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} + 10^{-5}\begin{bmatrix} 1.333 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}.
\end{aligned}$$

Esta expresión muestra entonces como la matriz densidad promedio se descompone en una parte que es proporcional a la identidad (no observable) y la desviación es un estado puro. Combinando los resultados de los tres experimentos vemos que la contribución de los estados erróneos es anulada al promediar los resultados dejando sólo la parte que es proporcional a $|00\rangle$.

En resumen: partiendo del estado inicial mixto, haciendo tres experimentos distintos modificando el estado inicial de una manera cuidadosamente elegida y promediando los resultados finales es posible obtener el mismo resultado que obtendríamos si hiciésemos un único experimento con un estado puro. Claramente esta manera es poco eficiente (de hecho el número de experimentos que hay que hacer crece exponencialmente con el número de espines que queremos usar) pero para pocos qubits es relativamente sencillo de implementar.

5.3.2. Método de promedio espacial

La otra técnica usada para generar estados pseudo-puros es conocida como "promedio espacial" [Cory98a]. En el ejemplo anterior vimos cómo obtener un estado pseudo-puro promediando los resultados de tres experimentos distintos. La idea ahora es, en vez de promediar los resultados de distintos experimentos lo hacemos dividiendo a la muestra en secciones y haciendo que cada parte evolucione de manera distinta. Como el espectrómetro sólo ve el comportamiento de toda la muestra el resultado es obtener una matriz densidad promediada. Ésta operación de promediar, ya sea espacialmente o temporalmente tiene el efecto de hacer una operación no-unitaria fundamental para poder pasar de una matriz densidad mixta a una matriz pseudo-pura. Claramente, las dos matrices tienen distintos autovalores y cualquier operación unitaria los conservaría.

Para poder implementar la técnica de promedio espacial se utilizan pulsos de gradiente de campo. La idea es prender, durante un tiempo Δt un campo magnético no-homogéneo. Los espectrómetros modernos permiten poner campos magnéticos que son lineales en alguna de las tres coordenadas x, y o z. Formalmente pensamos que durante el tiempo Δt, a nuestro campo $\mathbf{B}$ le

sumamos un campo lineal en z: $\mathbf{B}(z) = B_g z\mathbf{z}$. El efecto de dicho campo es el de enroscar la magnetización alrededor al eje z. Es decir: si la matriz densidad desviación tiene términos del tipo $\mathbf{I}_x$ estos pasaran a un estado rotado $\cos(\gamma B_g z \Delta t)\,\mathbf{I}_x + \sin(\gamma B_g z \Delta T)\,\mathbf{I}_y$ donde el ángulo con el que rota depende de la coordenada z. Distintos núcleos tienen distinto γ por lo que el número de vueltas que darán los vectores de campo dependerán del tipo de núcleo. La matriz densidad promedio se obtiene integrando esta expresión sobre toda la muestra de altura L. Si $L\gamma B_g \Delta t \gg 1$, las funciones coseno y seno promediarán a cero de manera que términos de la forma $\mathbf{I}_x$ desaparecen de la expansión de ρ. En otras palabras, si la expansión de ρ tenía algún término con $\mathbf{I}_{x,y}$ o $\mathbf{I}_{x,y}\mathbf{I}_z$ estos términos no serán observables porque al promediarlos sobre toda la muestra se cancela su contribución. Esto es análogo a lo que sucedía en el promedio temporal: uno puede hacer que ciertos elementos de la matriz densidad no sean observables.

Sin embargo, de lo discutido anteriormente no es para nada obvio que podamos con esto generar estados pseudo-puros. Afortunadamente se han desarrollado secuencias de pulsos que demuestran que es posible hacerlo y Cory y Knill/Laflamme muestran técnicas generales para obtener este tipo de estados utilizando gradientes [Cory98a]. Para dar una idea de cómo funciona esta secuencia mostraremos cómo es posible obtener un estado pseudo-puro de dos qubits en nuestra molécula hipotética de $CHCl_3$. El objetivo es pasar de la matriz densidad térmica escrita en el formalismo de producto de operadores: $\rho_0 = \mathbf{I}_z^1 + \mathbf{I}_z^2$ a la de un estado $|00\rangle\langle 00|$ que, escrita en FOP es: $|00\rangle\langle 00| = 1/4(\mathbf{1} + 2\mathbf{I_z})^1(\mathbf{1} + 2\mathbf{I_z})^2 = (1/4\mathbf{1} + 1/2\mathbf{I}_\mathbf{z}^\mathbf{1} + 1/2\mathbf{I}_\mathbf{z}^\mathbf{2} + \mathbf{I}_\mathbf{z}^\mathbf{1}\mathbf{I}_\mathbf{z}^\mathbf{2})$. A continuación mostramos la secuencia de pasos que genera dicho estado:

$$
\begin{aligned}
& \mathbf{I}_z^1 + \mathbf{I}_z^2 \\
\xrightarrow{[\pi/3]_x^2} \quad & \mathbf{I}_z^1 + \mathbf{I}_z^2/2 - \mathbf{I}_y^2\sqrt{3}/2 \\
\xrightarrow{[\mathrm{grad}]_z} \quad & \mathbf{I}_z^1 + \mathbf{I}_z^2/2 \\
\xrightarrow{[\pi/4]_x^1} \quad & \mathbf{I}_z^1/\sqrt{2} + \mathbf{I}_z^2/2 - \mathbf{I}_y^1/\sqrt{2} \\
\xrightarrow{[1/(2J_{12})]} \quad & \mathbf{I}_z^1/\sqrt{2} + \mathbf{I}_z^2/2 + \sqrt{2}\,\mathbf{I}_x^1\mathbf{I}_z^2 \\
\xrightarrow{[-\pi/4]_y^1} \quad & \mathbf{I}_z^1/2 + \mathbf{I}_z^2/2 - \mathbf{I}_x^1/2 + \mathbf{I}_x^1\mathbf{I}_z^2 + \mathbf{I}_z^1\mathbf{I}_z^2 \\
\xrightarrow{[\mathrm{grad}]_z} \quad & \mathbf{I}_z^1/2 + \mathbf{I}_z^2/2 + \mathbf{I}_z^1\mathbf{I}_z^2
\end{aligned}
\tag{5.19}
$$

Los primeros pasos consisten en un pulso de $\pi/3$ en torno al eje x sobre el segundo espín seguido por un gradiente de campo magnético que elimina el término $\mathbf{I}_y^2$. Luego se aplica una secuencia de una rotación ento a x del primer espín en un ángulo de $\pi/4$ seguido por un tiempo de evolución $t = 1/(2J_{12})$ y finalmente un pulso de $\pi/4$ en torno a $-y$. La secuencia termina aplicando un gradiente que destruye la magnetización en el plano x-y dejando el estado que queríamos (salvo por la parte proporcional a la identidad que, de todos

modos, no es observable).

Es interesante destacar, que esta misma secuencia se puede usar como base para generar un estado pseudo-puro utilizando promedio temporal. El truco consiste en, por ejemplo, hacer dos secuencias: la original (sin el primer gradiente) y otra en la cual el primer pulso es $\pi/3$ pero ahora en la dirección $-x$ (también sin el primer gradiente). El efecto de esto es que, después de dicho pulso, el estado resulta $\mathbf{I}_z^1 + \mathbf{I}_z^2/2 + \mathbf{I}_y^2\sqrt{3}/2$ que tiene el término $\mathbf{I}_y^2$ cambiado de signo. Al sumar los resultados de este y del experimento original toda la contribución de $\mathbf{I}_y^2$ se cancelará logrando el mismo efecto del gradiente. Así hemos logrado eliminar la necesidad de un gradiente (a costa de tener que hacer 2 experimentos). Usando esta misma idea podemos eliminar el segundo gradiente: en lugar de hacer el último gradiente, rotamos π alrededor del eje z el primer spin. Esa operación sólo cambia el signo de los dos términos que tienen $\mathbf{I}_x^1$ y al promediar se eliminan. Los espectrómetros modernos permiten hacer este tipo de operaciones automáticamente usando lo que se conoce como 'phase cycling'.

En resumen: los métodos de promedio temporal y espacial permiten generar estados iniciales que tienen las mismas propiedades observables que los estados puros.

5.4. Implementación de circuitos cuánticos en RMN

Para poder implementar cualquier compuerta unitaria sobre L qubits es necesario ser capaz de hacer dos tipos de operaciones: operaciones arbitrarias unitarias sobre un qubit y un CNOT entre dos bits. Un interesante teorema muestra que con estas dos operaciones es posible implementar cualquier operador unitario [Barenco95]. Las operaciones sobre un qubit se pueden implementar mediante rotaciones. Como vimos en la sección 5.2 las rotaciones sobre un espín se pueden implementar utilizando pulsos de rf en resonancia con el espín que queremos transformar y el eje de rotación se cambia variando la fase del pulso (para rotaciones con un eje en el plano x-y). Si queremos hacer una rotación en torno al eje z tenemos dos opciones. Una consiste en utilizar la identidad: $\exp(-i\theta\,\mathbf{I}_z) = \exp(i\pi/2\,\mathbf{I}_y)\exp(-i\theta\,\mathbf{I}_x)\exp(-i\pi/2\,\mathbf{I}_y)$ que nos dice que rotar en torno al eje z es equivalente a rotar 90 grados alrededor de y (rotando al eje z sobre el eje x), rotar alrededor de x en el ángulo θ y volver el eje x al z rotando -90 grados alrededor de y. Es decir: una rotación alrededor de z se convierte en tres rotaciones alrededor de x e y. Esto implica que cada rotación alrededor de z se convierte en 3 operaciones de 1 bit lo cual no es bueno porque agrega una mayor cantidad de pulsos a nuestra secuencia introduciendo más errores. Afortunadamente siempre es posible ignorar las rotaciones alrededor del eje z utilizando una técnica conocida como seguimiento de fase. Para entender esta técnica conviene repasar el significado de la fase ϕ en los pulsos de rf. Cunado uno dice que $\phi = 0$ es un pulso alrededor del eje x uno está haciendo una elección arbitraria de lo que llama eje x. Una rotación alrededor de z no es otra cosa que cambiar la definición de lo que llamamos $\phi = 0$ y entonces lo que hay que hacer es no hacer esta rotación y 'recordar'

que hubo una redefinición de ϕ. La próxima vez que tengamos que hacer una rotación alrededor de, digamos x, lo que en realidad hacemos es una rotación en el mismo ángulo pero alrededor del eje rotado. De ésta manera se puede evitar todas las rotaciones alrededor de z. Esto es muy bueno porque, como veremos un poco después, muchas operaciones necesarias para implementar distintos algoritmos cuánticos usan este tipo de rotaciones.

Antes de mostrar cómo se implementa un CNOT en RMN conviene discutir un poco sobre una operación muy repetida en computación cuántica que se conoce como transformación de Hadamard. Dicha transformación mapea el estado $|0\rangle \rightarrow 1/\sqrt{2}(|0\rangle + |1\rangle)$ y $|1\rangle \rightarrow 1/\sqrt{2}(|0\rangle - |1\rangle)$. La matriz de dicha transformación:

$$\hat{U}_{\mathrm{H}} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}.$$

Es fácil verificar que esta operación se puede hacer mediante dos rotaciones: una rotación en un ángulo de 90 grados alrededor del eje $-y$ seguida de una rotación de 180 grados en torno al eje z. Es fácil verificarlo haciendo el producto de las dos matrices de rotación correspondientes. El resultado es una matriz que es proporcional a la de la transformada de Hadamard.

Para implementar el CNOT en RMN conviene escribirlo en función de otra compuerta: la compuerta que introduce un desfasaje controlado. Esta operación tiene el efecto de introducir un fase $\exp(i\phi)$ al estado $|11\rangle$ dejando a todos los demás estados sin cambio. La matriz que representa esta operación es (en la base $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$) es:

$$\hat{U}_{\mathrm{ps}}(\phi) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \mathrm{e}^{i\phi} \end{bmatrix}.$$

La ventaja de esta operación es que en RMN es muy fácil de implementar esta compuerta y no es difícil convencerse que el CNOT se puede escribir combinando la compuerta de fase con dos transformaciones de Hadamard sobre el bit target. Es decir un CNOT entre el bit 1 y 2 (bit 1 es el control) se escribe como: $\hat{U}^{1,2}_{\mathrm{CNOT}} = \hat{U}^{2}_{\mathrm{H}} \hat{U}^{1,2}_{\mathrm{ps}}(\pi) \hat{U}^{2}_{\mathrm{H}}$. Esta identidad se puede representar como muestra el circuito de la figura 5.2.

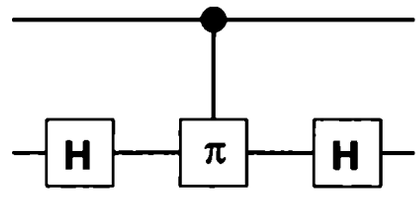


Figura 5.2: La compuerta CNOT se puede escribir combinando dos operaciones de Hadamard sobre el bit de destino y una compuerta de fase como se ve en la figura.

Es simple verificar que $\hat{U}_{\text{ps}}(\theta) = \exp(i\theta/2(\mathbf{I}_z^1 + \mathbf{I}_z^2 - 2\mathbf{I}_z^1\mathbf{I}_z^2))$ a menos de una fase global (que, sin embargo, no tiene efectos y por lo tanto los dos operadores son equivalentes). La ventaja de la segunda expresión es que revela inmediatamente cómo implementarla en RMN. La compuerta de fase se puede implementar entonces mediante dos rotaciones individuales de los qubits 1 y 2 en un ángulo $\theta/2$ alrededor de z y un acoplamiento J en un ángulo $-\theta/2$.

En resumen: mostramos como podemos hacer en RMN cualquier operación unitaria en L bits a partir de rotaciones en un bit y acoplamientos J entre dos bits que son las dos operaciones que se pueden hacer naturalmente en RMN. Sin embargo hay un detalle que hemos dejado de lado: ¿qué pasa cuando tenemos más de dos qubits? Todos los ejemplos los hemos hechos en nuestra molécula hipotética de dos qubits pero en la realidad deberemos ser capaces de operar con más bits. Trabajar con mayor cantidad de qubits introduce varios problemas. La dificultad principal de trabajar con más de dos bits proviene de la necesidad de hacer pulsos selectivos sobre un espín sin modificar el estado de los otros. En el ejemplo de nuestra molécula hipotética de $CHCl_3$ no había problema porque las dos frecuencias de resonancia de nuestros qubits estaban muy separadas. En RMN se acostumbra usar un número limitado de tipos de núcleos: ^{13}C, ^{1}H, ^{19}F, ^{14}N. A medida que agregamos qubits las frecuencias de resonancia se encuentran cada vez más pegadas y los efectos 'off-resonance' se hacen cada vez más importantes. Otra limitación es que la mayoría de los espectrómetros modernos sólo operan con dos tipos de núcleos a la vez: típicamente ^{1}H y otro núcleo. En el mejor de los casos se puede tener cabezales para cinco tipos de elementos. Esto hace que, al agregar bits tengamos que usar núcleos del mismo elemento para nuestros qubits. Si bien en las molécula dos carbonos distintos no tienen la misma frecuencia de resonancia debido al efecto del corrimiento químico las diferencias de frecuencia son relativamente chicas (típicamente del orden de los KHz en carbono) comparada con la diferencia $\omega_{\text{H}} - \omega_{\text{C}} \sim 375\,\text{MHz}$. Para poder obtener la suficiente selectividad se puede modular la amplitud de los pulsos pero esto implica que los mismos sean más largos cuanto más cerca estén los núcleos en frecuencia. También al agregar más núcleos el método de seguimiento de fase se hace más engorroso aunque esto no es una dificultad fundamental ya que se pueden escribir programas para hacerlo automáticamente.

Finalmente otra dificultad importante es el hecho que debemos introducir muchos pulsos de reenfoque para apagar las distintas interacciones como mostramos en la sección 5.2.4. Por ejemplo: si tenemos una molécula de 3 espines y queremos hacer una compuerta CNOT entre dos de ellos debemos, en el medio de la compuerta apagar la interacción restante haciendo un pulso de refocussing sobre el espín 3. Para ello hace falta ser capaz de hacer pulsos selectivos sobre dicho espín y a medida que agregamos qubits a nuestra computadora, mayor es la cantidad de pulsos que hay que hacer para apagar las distintas interacciones.

Cabe señalar que, aunque la mayoría de las dificultades mencionadas arriba son superables en teoría hay una dificultad de la cual no hemos dicho nada que hace que RMN no escalee a más de una decena de qubits. La dificultad

principal proviene del hecho de que en **todos** los métodos conocidos hasta el día para generar estados pseudopuros la relación señal-ruido decae exponencialmente con el número de qubits (por lo menos para RMN en líquidos a temperatura ambiente). Esta propiedad limita seriamente cualquier propuesta para hacer RMN con líquidos a temperatura ambiente.

5.5. Resultados experimentales

Presentaremos aquí una serie de resultados experimentales obtenidos en un espectrómetro comercial de 500 MHz. Estos resultados demuestran la posibilidad real de implementar en la práctica la mayoría de los conceptos discutidos anteriormente y muestran cuáles son las dificultades experimentales para implementar computación cuántica en un escenario real.

5.5.1. La molécula de tricloroetileno

Los experimentos se hicieron con una molécula de tricloroetileno (TCE) con carbono enriquecido (^{13}C en vez de ^{12}C) al 99%. La molécula tiene dos carbonos fuertemente acoplados $J \sim 100$ Hz, un hidrógeno acoplado con una constante $J \sim 200$ Hz con el carbono C_1 y $J \sim 10$Hz con el carbono C_2. En la figura 5.3 se puede ver la disposición geométrica de la molécula.

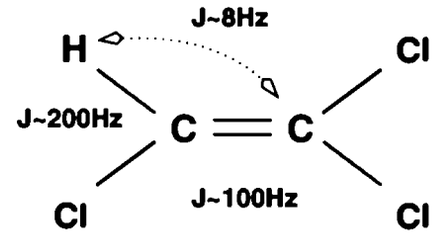


Figura 5.3: Molécula de tricloroetileno. La interacción entre el ^{1}H y los dos carbonos de de tipo $\mathbf{I}_z^1\mathbf{I}_z^2$ mientras que el acoplamiento entre carbonos es del tipo $\vec{I}^1 \cdot \vec{I}^2$. La diferencia de frecuencia entre los carbonos es, para un espectrómetro de 500 MHz, aproximadamente $\omega_{C_1} - \omega_{C_2} \sim 900$ Hz.

El Hamiltoniano de la molécula de TCE está muy bien descripto por la siguiente expresión:

$$\mathbf{H} = \omega_1 \mathbf{I}_z^1 + \omega_2 \mathbf{I}_z^2 + \omega_3 \mathbf{I}_z^3 + 2\pi J_{12} \mathbf{I}_z^1 \mathbf{I}_z^2 + 2\pi J_{13} \mathbf{I}_z^1 \mathbf{I}_z^3 + 2\pi J_{23} \tilde{\mathbf{I}}^2 \cdot \tilde{\mathbf{I}}^3 \qquad (5.20)$$

donde para la molécula que usamos las constantes son: $J_{12} = 200.76$ Hz, $J_{13} = 9.12$ Hz y $J_{23} = 103.06$ Hz mientras que el chemical shift C_1-C_2 es $\delta_{C_1C_2} = 908.88$ Hz. De ahora en más etiquetaremos con 1 al hidrógeno, 2 al carbono del medio y 3 al otro carbono. Para trabajar más cómodamente conviene hacer un cambio de representación a un sistema de referencia que rota junto con el hidrógeno y C_1 mediante el siguiente operador unitario:

$$\mathbf{U}_0(t) = \mathrm{e}^{-i(\omega_1 \mathbf{I}_z^1 + \omega_2 \mathbf{I}_z^2)t}$$

en la representación interacción ($\rho_I = \mathbf{U}_0^\dagger(t)\rho\mathbf{U}_0(t)$) la matriz densidad evoluciona con el Hamiltoniano:

$$\tilde{\mathbf{H}}_I = -2\pi\delta\,\mathbf{I}_z^3 + 2\pi J_{12}\,\mathbf{I}_z^1\mathbf{I}_z^2 + 2\pi J_{13}\,\mathbf{I}_z^1\mathbf{I}_z^3 + 2\pi J_{23}\,\tilde{\mathbf{I}}^2 \cdot \tilde{\mathbf{I}}^3 \tag{5.21}$$

donde δ es el chemical-shift. Aunque este Hamiltoniano es bastante simple (y se puede resolver analíticamente) todavía es complicado para diseñar secuencias de pulsos. Para simplificarlo un poco más haremos dos aproximaciones que lo hacen más manejable: la primera consiste en ignorar el acoplamiento J_{13} ya que es bastante más chico que los otros. La segunda aproximación consiste en suponer que el acoplamiento C_1-C_2 es débil. Esto es válido siempre que $J_{23} \ll \delta$ condición que se cumple en nuestro caso. Con estas dos simplificaciones llegamos al siguiente Hamiltoniano:

$$\mathbf{H}_I = -2\pi\delta\,\mathbf{I}_z^3 + 2\pi J_{12}\,\mathbf{I}_z^1\mathbf{I}_z^2 + 2\pi J_{23}\,\mathbf{I}_z^2\mathbf{I}_z^3 \tag{5.22}$$

Para verificar el impacto de estas aproximaciones desarrollamos un código C que simula la evolución con el Hamiltoniano sin ninguna aproximación. Como entrada al programa se le ingresa la secuencia de pulsos en un lenguaje casi idéntico al usado por el espectrómetro y a la salida se puede ver el resultado de la secuencia tanto en formalismo de producto de operadores como la transformada de Fourier de la corriente que se mediría en la bobina de carbono o hidrógeno. El programa también permite comparar los resultados con y sin acoplamiento fuerte para monitorear su efecto. La conclusión principal es que en lineas generales los efectos de acoplamiento fuerte no son muy importantes salvo cuando aparecen ciertos operadores producto. En ese caso hemos tenido en cuenta su efecto en forma sistemática.

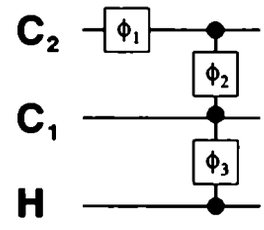


Figura 5.4: Esquema de la propagación de la molécula de tricloroetileno ignorando los efectos de acoplamiento fuerte y el acoplamiento C_2-H. Propagar durante un tiempo t es equivalente a una rotación alrededor del eje z de C_2, un acoplamiento $\mathbf{I}_z^H\mathbf{I}_z^{C_1}$ (que para abreviar llamaremos acoplamiento ZZ) y un acoplamiento ZZ entre C_1 y C_2.

La evolución descripta por (5.22) se puede entender como un acoplamiento ZZ entre los espines 1-2 y entre 2-3 más una rotación del espín 3 en torno al eje $-z$. En la figura 5.4 se puede ver diagramáticamente el efecto de propagar durante un tiempo t con $\mathbf{H}_I$. Como discutimos en la sección anterior, mediante períodos de evolución libre y pulsos sobre cada uno de los espines se logra

apagar y prender los distintos acoplamientos. Por ejemplo: evolucionando por un tiempo τ, aplicando un pulso π sobre ^{1}H, y evolucionando durante τ nuevamente se apaga la interacción J_{12} (y también la J_{13}). Es importante recordar que esta evolución no anula la rotación en z del bit 3. Para evolucionar sólo con J_{23} lo que se hace es evolucionar durante τ, aplicar un pulso sobre los dos carbonos *simultáneamente* y nuevamente evolucionar en τ. Dicha secuencia anula tanto el acoplamiento J_{12}, la rotación en z del bit 3 y el acoplamiento J_{13}.

Es importante recordar que, debido a que las frecuencias de C_1 y C_2 están relativamente cerca es muy difícil hacer pulsos selectivos sobre uno de los carbonos. El espectrómetro que usamos en el LANAIS RMN 500 no tiene la posibilidad de modular los pulsos para poder lograr la selectividad necesaria por lo que hay que recurrir a un ingenioso truco desarrollado por Raymond Laflamme y Emanuel Knill. La idea es utilizar la rotación en z de C_2 para transformarla en una rotación alrededor de otro eje sobre C_2. La siguiente secuencia pone en funcionamiento esa idea:

i. Aplicar un pulso $\pi/2$ alrededor de y sobre C_1 y C_2.

ii. Dejar evolucionar por un tiempo $\tau = 1/(8\delta)$.

iii. Aplicar un pulso $-\pi/2$ alrededor de y sobre C_1 y C_2.

Debido a que $\delta \gg J_{12}$ y $\delta \gg J_{23}$ podemos pensar que el único efecto de dejar evolucionar por un tiempo τ suficientemente corto al sistema con el Hamiltoniano (5.22) es el de rotar el carbono 2 en un ángulo $4\pi\delta\tau = \pi/2$ alrededor del eje $-z$. Si ahora rodeamos esta evolución con dos pulsos de $\pm\pi/2$ en los carbonos el efecto neto es el de no hacer nada en C_1 (pues primero rota $\pi/2$ en un dirección y después en la contraria) mientras que sobre C_2 el efecto es el de $R_y(90)R_z(4\pi\delta\tau)R_y(-90) = R_x(4\pi\delta\tau)$; es decir, convertimos la rotación alrededor de z en una rotación sobre el eje x. Mediante este truco es posible hacer pulsos selectivos sobre C_2 pero con un pulso adicional sobre los carbonos en un ángulo de $-4\pi\theta\tau$ se puede hacer que el pulso sea sólo sobre C_1. Es claro que esta secuencia no es exacta e introduce un error debido a que los acoplamiento J no son tanto más grandes que δ. Una manera de mejorarla un poco es aplicar un pulso π en la mitad de la evolución libre para anular la evolución J_{12} y J_{13}. Con esta secuencia se obtienen mejores resultados y resulta una forma relativamente efectiva de hacer pulsos selectivos en un tiempo bastante corto. Sin embargo se debe tener en cuenta que el método introduce errores y conviene tener esto en cuenta a la hora de diseñar secuencias para minimizar el número de pulsos selectivos sobre los carbonos.

5.5.2. Estados pseudo-puros

Una de los primeros objetivos que nos fijamos fue el de intentar generar estados pseudo-puros utilizando técnicas de promedio espacial, temporal y mixto. A modo de ejemplo mostraremos algunas secuencias simples que

generan estados pseudo-puros en dos espíns. Antes de presentar los resultados conviene ver cuáles son los operadores productos que generarán señal sobre las bobinas. La señal producida por los carbonos es, como vimos antes, proporcional a:

$$\begin{aligned} I(t) &= \mathrm{Tr}[(\mathbf{I}_+^2 + \mathbf{I}_+^3)\rho(t)] = \ldots \\ &= \mathrm{Tr}[(\mathbf{I}_+^2(t) + \mathbf{I}_+^3(t))\rho(0)]. \end{aligned} \tag{5.23}$$

El cálculo es ahora un poco más extenso pero veamos, a modo de ejemplo cómo es la señal observada en el espín 3 (C_2). Si calculamos $\mathbf{I}_+^3(t)$ usando como Hamiltoniano (5.21) pero considerando acoplamiento débil entre los carbonos obtenemos:

$$\begin{aligned} \mathbf{I}_+^3(t) = \mathrm{e}^{i2\pi\delta t} \; [\quad & \cos\pi J_{23}t \; \cos\pi J_{13}t \; \mathbf{I}_x^3 - \\ & -i\cos\pi J_{23}t \; \sin\pi J_{13}t \; 2\mathbf{I}_z^2\mathbf{I}_x^3 - \\ & -i\sin\pi J_{23}t \; \cos\pi J_{13}t \; 2\mathbf{I}_z^1\mathbf{I}_x^3 - \\ & -\sin\pi J_{23}t \; \sin\pi J_{13}t \; 4\mathbf{I}_z^1\mathbf{I}_z^2\mathbf{I}_x^3 \;] + \\ & +i \; \mathrm{e}^{i2\pi\delta t} \; [\quad \mathbf{I}_x^3 \leftrightarrow \mathbf{I}_y^3 \quad] \end{aligned} \tag{5.24}$$

Esta expresión muestra que la transformada de Fourier estará centrada en la frecuencia $2\pi\delta$. Al expandir las funciones trigonométricas en términos de exponenciales como hicimos en el ejemplo anterior la parte real (de la transformada de Fourier de la la corriente) proviene de los términos $\mathbf{I}_x^3$, $2\,\mathbf{I}_z^1\mathbf{I}_x^3$, $2\,\mathbf{I}_z^2\mathbf{I}_x^3$ y $4\,\mathbf{I}_z^1\mathbf{I}_z^2\mathbf{I}_x^3$ mientras que la parte imaginaria viene de los mismos términos cambiando $\mathbf{I}_x^3$ por $\mathbf{I}_y^3$ [2] La transformada de cada término tendrá cuatro picos en las frecuencias $-(J_{23}+J_{13})/2$, $-(J_{23}-J_{13})/2$, $(J_{23}-J_{13})/2$ y $(J_{23}+J_{13})/2$ mientras que los signos son todos + para el operador producto $\mathbf{I}_x^3$ y signos opuestos de a pares para los otros elementos (ver figura 5.5).

El cálculo de los picos de C_1 es exactamente igual cambiando los acoplamientos y ahora los picos estarán centrados en la frecuencia 0 (relativa a nuestro sistema rotante). En la figura 5.6 vemos un espectro típico obtenido después de un pulso $\pi/2$ sobre los carbonos a partir del estado térmico. El estado justo antes de empezar a medir la corriente en el espectrómetro es (suponiendo que el pulso fue alrededor del eje y) $\rho(0) = \mathbf{I}_z^1 + \mathbf{I}_x^2 + \mathbf{I}_x^3$. La corriente inducida en la bobina es:

[2] Cuando hablamos de la 'parte real' de la corriente nos referimos a la parte real de la transformada de Fourier de la corriente medida como función del tiempo. Por ejemplo, si la señal medida por el detector es un $cos(\omega t+\phi)$, la transformada de Fourier de la corriente tiene una parte real (que genera un pico con forma Lorenziana que se conoce como "espectro de absorción") y una parte imaginaria con un aspecto similar a la derivada de una Lorenziana, que se conococe como "espectro dispersivo". Ver [VandeVen95] para una explicación más completa de la detección en cuadratura.

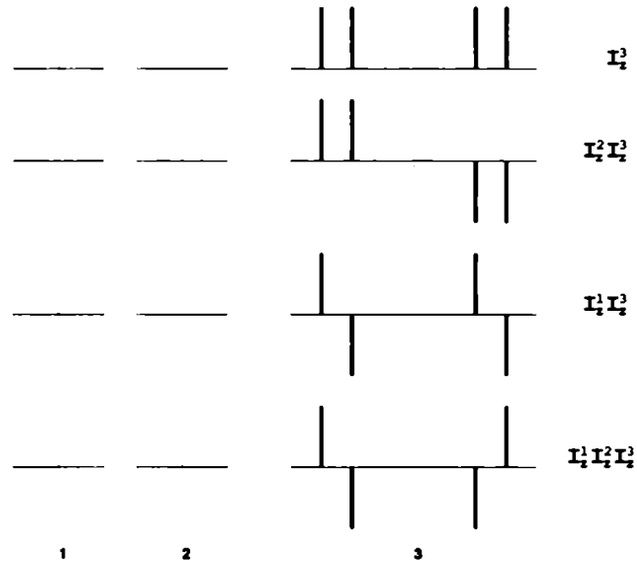


Figura 5.5: Espectros correspondientes a los 4 operadores producto visibles de 3 núcleos (después de un pulso de lectura sobre el espín 3): $\mathbf{I}_z^3$, $\mathbf{I}_z^2\mathbf{I}_z^3$, $\mathbf{I}_z^1\mathbf{I}_z^3$, y $\mathbf{I}_z^1\mathbf{I}_z^2\mathbf{I}_z^3$. En los gráficos supusimos $J_{13} < J_{32}$ como es el caso en la molécula de TCE.

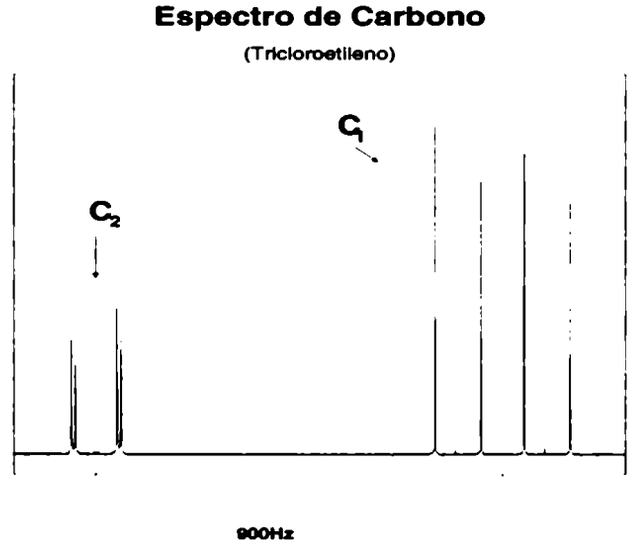


Figura 5.6: Espectro experimental del carbono a partir del estado térmico. Se observan 2 grupos de 4 picos cada uno separados por el chemical-shift que es $\sim$ 900Hz. La separación en 4 picos se debe a los acoplamientos J entre carbonos y con el ^{1}H. La distinta altura se debe al efecto de acoplamiento fuerte que fue ignorado en el tratamiento teórico. También se puede observar un pico muy pequeño entre los picos grande debido a la interacción entre moléculas que tienen 1 ^{12}C en lugar de ^{13}C.

$$\mathrm{Tr}\left[\rho(0)\left(\mathbf{I}_+^2(t)+\mathbf{I}_+^3(t)\right)\right] = \cos\pi J_{12}t\ \cos\pi J_{23}t + \mathrm{e}^{i2\pi\delta t}\ \cos\pi J_{23}t\ \cos\pi J_{13}t \tag{5.25}$$

cuya transformada de Fourier tiene 8 picos como se ve en la figura. Se ve que los picos no son todos de la misma altura (como indica nuestro cálculo teórico). El motivo es que en nuestro cálculo hemos ignorado los efectos de acoplamiento fuerte que se manifiestan en las amplitudes de los picos.

En la figura 5.7 vemos el espectro de un estado pseudo-puro de dos qubits obtenido utilizando promedio temporal. Para ello se diseñaron 4 secuencias que generan los estados: $\mathbf{I}_z^2$, $2\mathbf{I}_z^1\mathbf{I}_z^2$, $2\mathbf{I}_z^2\mathbf{I}_z^3$ y $4\mathbf{I}_z^1\mathbf{I}_z^2\mathbf{I}_z^3$. Si calculamos una matriz densidad que resulta de sumar los cuatro estados obtenemos la siguiente expresión: $(\mathbf{I}_z^2 + 2\mathbf{I}_z^1\mathbf{I}_z^2 + 2\mathbf{I}_z^2\mathbf{I}_z^3 + 4\mathbf{I}_z^1\mathbf{I}_z^2\mathbf{I}_z^3) = (1 + 2\mathbf{I}_z)\mathbf{I}_z(1 + 2\mathbf{I}_z) = |0\rangle\langle 0| \otimes \mathbf{I}_z \otimes |0\rangle\langle 0|$ Si le aplicamos un pulso de $\pi/2$ sobre C_1 obtenemos el estado $(\mathbf{I}_x^2 + 2\mathbf{I}_z^1\mathbf{I}_x^2 + 2\mathbf{I}_x^2\mathbf{I}_z^3 + 4\mathbf{I}_z^1\mathbf{I}_x^2\mathbf{I}_z^3)$ que es observable (su traza con $\mathbf{I}_+^2(t)$ es distinta de cero) y el espectro debería tener un sólo pico como se ve en el gráfico. Los otros tres espectros corresponden a los otros 3 posibles estados computacionales del H y C_2. En el apéndice 2 se muestran las secuencias usadas para generar los cuatro estados. Los resultados presentados aquí usan, en realidad, además de promedio temporal gradientes para "promediar a cero" ciertos operadores. Cómo habíamos dicho antes, se puede usar 'phase cycling' para obtener el mismo efecto.

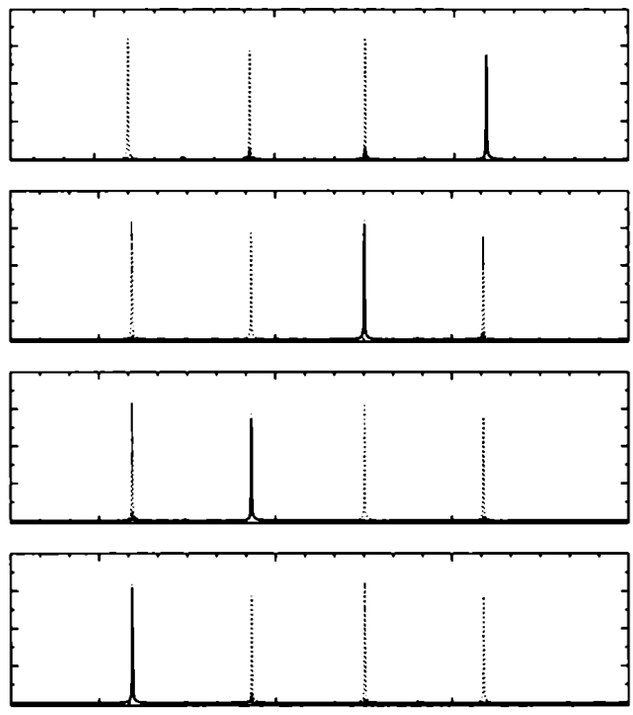


Figura 5.7: Estado pseudo-puro de dos qubits ($|0\rangle \otimes \mathbf{I}_z \otimes |0\rangle$ obtenidos utilizando promedio temporal.)

También hemos generado estados pseudo-puros de dos qubits usando promedio espacial. La secuencia fue diseñada por Raymond Laflamme y Em-

manuel Knill y se muestra en el apéndice 2. En la figura 5.8 mostramos el espectro de C_2 cuando el H y C_1 están en el estado $|0\rangle|0\rangle$.

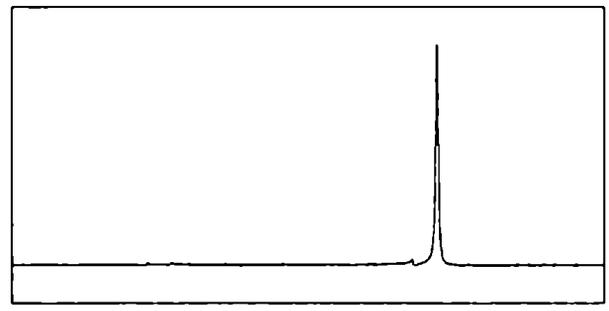
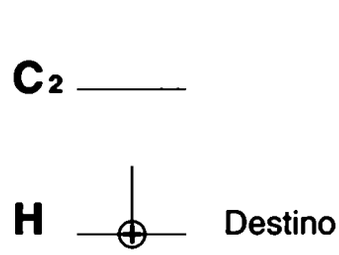


Figura 5.8: Espectro de C_2 del estado $|0\rangle \otimes |0\rangle \otimes \mathbf{I}_z$ donde el H y C_1 están en un estado puro. El gráfico muestra el espectro después de aplicarle un pulso de lectora sobre C_2.

5.5.3. Una compuerta cuántica

El próximo experimento simple consiste en implementar una compuerta CNOT entre dos espines. En este experimento generamos los cuatro estados computacionales $|00\rangle$, $|01\rangle$, $|10\rangle$ y $|11\rangle$ utilizando la secuencia del estado pseudo-puro. Luego le aplicamos la secuencia del CNOT que describimos en 5.4 y medimos el espectro final para comprobar que la compuerta satisface su tabla de verdad. Como qubit de control se uso el C_1 y el hidrógeno como bit afectado. En la figura 5.9 se muestra la molécula y la tabla de verdad del CNOT.



Entrada	→	Salida
0		0
1		1
0		1
1		0

Figura 5.9: Efecto de un CNOT sobre los cuatro estados posibles iniciales.

El efecto de un CNOT sobre los cuatro estados iniciales posibles se puede ver en la figura 5.10. En la parte izquierda del gráfico se ve el espectro sobre C_2 de los 4 estados computacionales iniciales. La columna derecha los muestra después de aplicarle el CNOT. Se ve claramente que el efecto es intercambiar los espectros correspondientes a $|10\rangle$ y $|11\rangle$. Normalmente (en el estado térmico) uno observa 4 resonancias (picos en la transformada de Fourier). Sin embargo, cuando los espines del H y el C_1 están en cada uno de los estados $00\ldots11$ en el espectro sólo se observa uno de los picos.

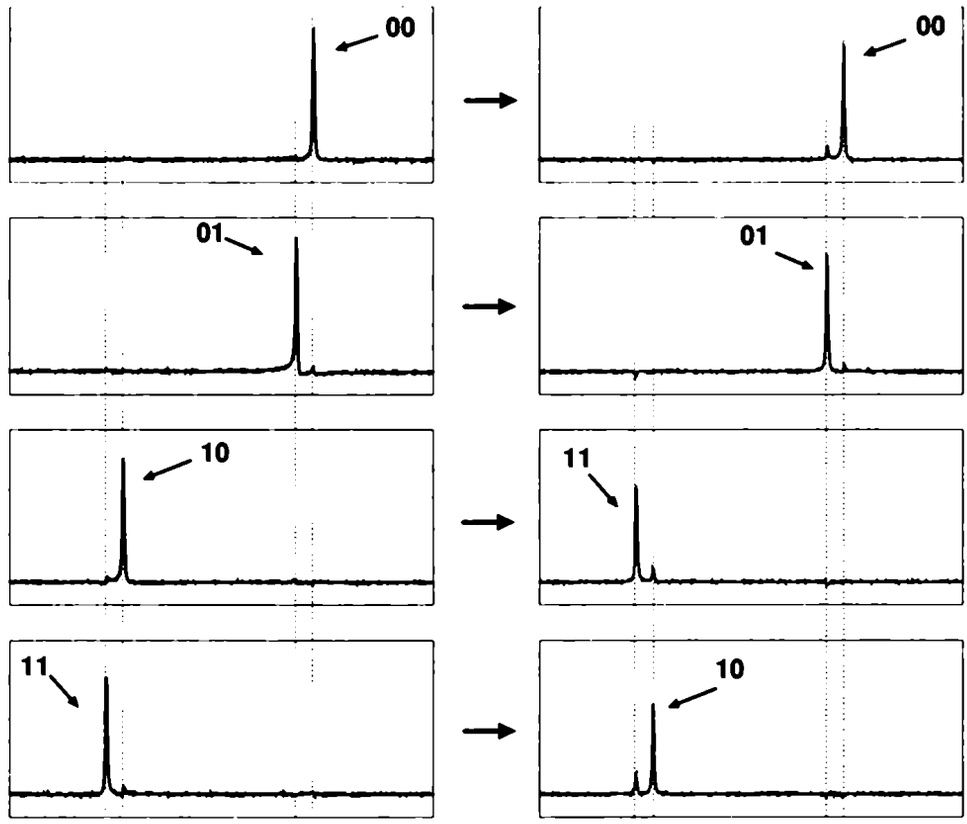


Figura 5.10: Espectro del carbono C_2 antes y después de aplicarle un CNOT entre el H y C_1. La columna izquierda es el espectro para cada uno de los 4 estados de entrada posibles. La otra columna muestra el espectro después de la compuerta CNOT. Nótese que se intercambian los estados $|10\rangle$ $|11\rangle$ que es el efecto del CNOT.

5.5.4. Medición de la función de Wigner de dos qubits.

En el capítulo anterior vimos que es posible definir una función de Wigner para sistemas discretos y presentamos una técnica para medirla experimentalmente utilizando una computadora cuántica. En esta sección mostraremos los detalles de una serie de experimentos llevados a cabo en el LANAIS de la FCEyN y en el Los Alamos National Laboratory. El objetivo de dichos experimentos era mostrar cómo se puede implementar los circuitos diseñados en el capítulo anterior utilizando las técnicas de RMN presentadas aquí. Estos resultados son totalmente novedosos y representan una nueva técnica tomográfica: mediante un único experimento es posible medir la función de Wigner en un punto del espacio de las fases. Si bien es posible hacer tomografía con RMN, esta es una forma novedosa que corresponde a medir los coeficientes de la expansión de la matriz densidad en una base distinta (la de operadores de espacio de fase) a la usada usualmente en RMN (operadores producto). La utililidad de esta técnica dependerá de lo que nos interese observar. Es claro que si estamos interesados en estudiar el límite semi-clásico de un sistema cuántico es más natural mirar al estado del sistema en el espacio de fases que en el de operadores producto.

Aquí mostraremos los detalles y resultados para un caso en el que el sistema a medir es un conjunto de dos qubits. En este caso el espacio de Hilbert del sistema tiene dimensión $N = 4$ y nuestro espacio de fase es una grilla de 8×8 puntos. La base computacional (de "posición") es $B_q = \{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$ donde hemos usado notación binaria para representar cada uno de los 4 estados posibles de la base.

Una vez elegida la forma de representar los estados debemos convertir el circuito de medición de la figura 4.4 en compuertas de uno y dos qubits: rotaciones en un espín y CNOT. Estas compuertas, sabemos, pueden ser implementadas mediante una secuencia de evoluciones libres y pulsos de rf. Para entender cómo se llega a la secuencia de pulsos final miremos el siguiente circuito que es una variante del de la figura 4.2:

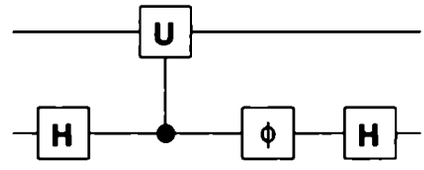


Figura 5.11: Circuito para medir la parte real o imaginaria de Tr $\left[\hat{\rho}\hat{U}\right]$. H es la transformada de Hadamard y $e^{-i\phi I_z}$ es el operador de un bit que rota en un ángulo ϕ alrededor del eje z

Si medimos el valor medio de la componente z del espín del qubit de control obtenemos (cuando el estado inicial del sistema está descripto por la matriz densidad $\rho \otimes \mathbf{I}_z$):

$$\langle \mathbf{I}_z \rangle = \mathrm{Tr}\left[\hat{\rho}\,\hat{U}^\dagger \exp(-i\phi)\right] + \mathrm{Tr}\left[\hat{\rho}\,\hat{U} \exp(i\phi)\right]$$

Si elegimos $\hat{U}(q,p) = \exp\left(-i\frac{2\pi}{N}q\hat{P}\right)\hat{R}\exp\left(i\frac{2\pi}{N}p\hat{Q}\right)$ y el ángulo $\phi = \frac{2\pi}{N}\frac{pq}{2}$ entonces el valor medio resulta:

$$\begin{aligned}\langle \mathbf{I}_z \rangle &= \mathrm{Tr}\left[\hat{\rho}\,\hat{A}^{\dagger}(q,p)\right] + \mathrm{Tr}\left[\hat{\rho}\,\hat{A}(q,p)\right] \\ &= 2\,\mathrm{Tr}\left[\hat{\rho}\,\mathrm{e}^{-i\frac{2\pi}{N}q\hat{P}}\,\hat{R}\,\mathrm{e}^{i\frac{2\pi}{N}p\hat{Q}}\,\mathrm{e}^{i\frac{2\pi}{N}\frac{pq}{2}}\right] \\ &= 2\,\mathrm{Tr}\left[\hat{\rho}\,\hat{A}(q,p)\right] \\ &= 2\,W(q,p)\end{aligned}$$

En otras palabras: midiendo el valor medio de la componente z del espín del bit de control obtenemos el valor de la función de Wigner en el punto q,p del espacio de las fases. Esto no es sorprendente porque el circuito 4.2 no hace otra cosa que medir la traza de $\hat{\rho}$ con $\hat{U}$. Si pensamos a $\hat{U} = \hat{U}(q,p)$ como un elemento (etiquetado por q,p) de alguna base en la cual expandimos $\hat{\rho}$ estamos entonces midiendo el coeficiente de expansión en dicha base. Si $\hat{U}(q,p)$ es uno de los operadores de punto de fase entonces el coeficiente de expansión no es otra cosa que el valor de la función de Wigner en q,p.

Los operadores $\hat{A}(q,p)$ se pueden descomponer en tres operaciones: traslación en la base de posición, reflexión y traslación en la base de momento. En el caso general, éste circuito puede ser bastante complicado: los operadores de desplazamiento espacial y de reflexión requieren de varias transformadas de Fourier discretas. Afortunadamente, para el caso simple que estamos considerando no hace falta tanta complicación. La idea central consiste en observar la acción de los operadores sobre los 4 estados posibles en notación binaria:

[1] El operador que traslada en una distancia q a un ket que es auto-estado de la posición $|n\rangle$ resulta:

$$\mathrm{e}^{-i\frac{2\pi}{N}q\hat{P}}|n\rangle = |n + q \bmod N\rangle$$

Si hacemos las cuatro traslaciones posibles utilizando notación binaria vemos que su efecto puede ser implementado con los circuitos siguientes:

[2] El operador de reflexión $\hat{R}$ puede implementarse por medio de un CNOT con el control en el bit menos significativo y el destino en el más significativo.

[3] El operador de traslación en posición puede ser implementado por rotaciones de un bit alrededor del eje z. Los ángulos de rotación son: $\phi_1 = -\pi p$ y $\phi_2 = -\frac{\pi}{2}p$.

El circuito puede ser simplificado aún más combinando las operaciones de desplazamiento en momento y la reflexión como se ve en la siguiente figura:

En la figura 5.14 vemos el circuito completo para todos los 16 operadores de fase posibles cuyo valor de expectación debemos medir (recordar que aunque el espacio de fase tiene 64 puntos, conociendo la función de Wigner en 16 puntos es suficiente debido a la relación: (4.29)). Como muestra la figura, en

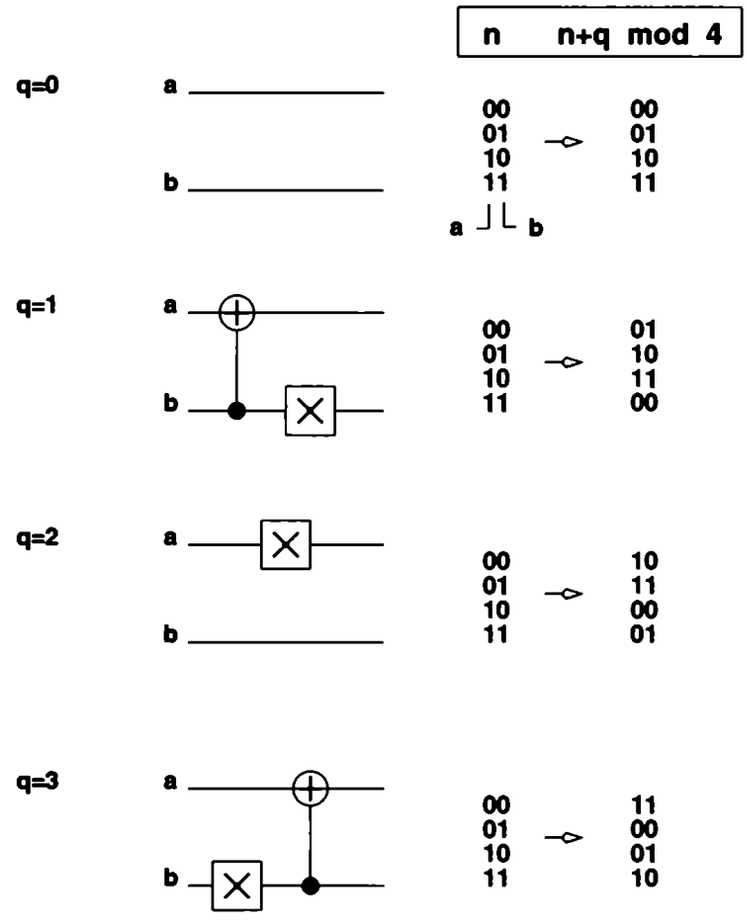


Figura 5.12: Circuito que implementa $e^{-i\frac{2\pi}{N}q\hat{P}}$ para los cuatro valores posibles de q cuando $N = 4$. a es el bit mas significativo mientras que b es el menos significativo.

el caso más complicado tenemos que hacer 2 compuertas de fase, un CNOT y una compuerta Toffoli. Dicha compuerta aparece porque aunque en el circuito que implementa el operador $\hat{A}(q,p)$ sólo aparecen compuertas de dos qubits, todas la operaciones deben estar controladas por el bit de control. Obsérvese también que las fases en las compuertas de fase son $\phi_1 = -\pi p$ y $\phi_2 = -\pi p/2$ asique, cuando $p = 0$ no se aplica ninguna compuerta y cuando $p = 2$ sólo se aplica una de las dos compuertas.

Si bien las secuencias son relativamente simples porque involucran un número chico de compuertas en la práctica, debido a las imperfecciones de los pulsos de rf en los espectrómetros y a las numerosas aproximaciones que hacemos cuando usamos TCE es fundamental reducir los pulsos al mínimo. Para ellos es posible hace una gran cantidad de simplificaciones al pasar de los circuitos a las secuencias de pulsos.

- Todas las rotaciones alrededor del eje z sobre el hidrógeno y sobre los dos carbonos se pueden ignorar utilizando el método de seguimiento de fase.

- Si miramos la secuencia de pulsos para cada uno de los 16 circuitos posibles desde la última operación hacia la primera podemos descartar

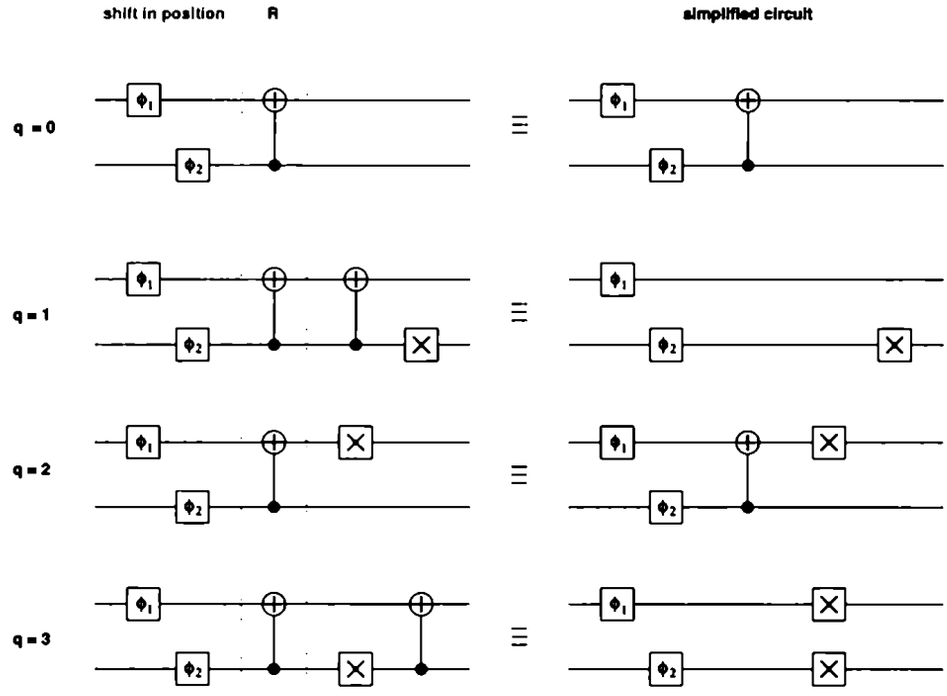


Figura 5.13: Combinación de compuertas que generan $\hat{U}(p,q)$

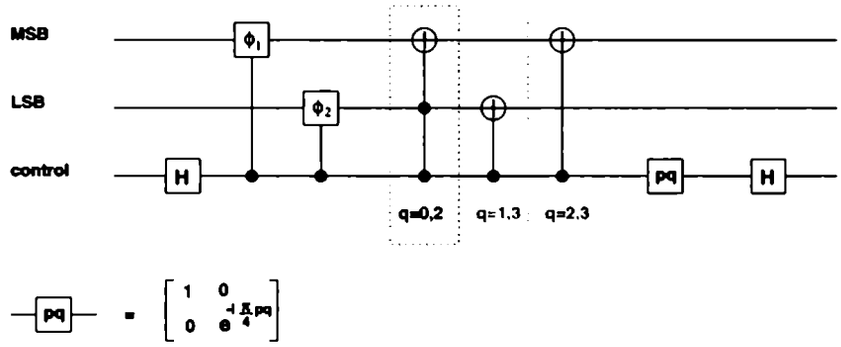


Figura 5.14: Circuitos para medir el valor de expectación de $\hat{A}(q,p)$ en cada uno de los 16 puntos posibles del espacio de fase.

todas las operaciones sobre los bits que no sean el control. Estas no modifican el valor de $\langle \hat{I}_z \rangle$ por lo que pueden ser ignoradas.

- La última Hadamard sobre el bit de control puede ser ignorada si, en lugar de mirar $\langle \mathbf{I}_z \rangle$ (que involucra una pulso final de lectura que cambia $z \longrightarrow x$) miramos directamente el valor medio de σ_x: $\langle \mathbf{I}_x \rangle$.
- Se pueden agregar pares de Hadamard (cuyo efecto neto es no hacer nada pues son auto-inversos) sobre alguno de los carbonos para evitar tener que hacer pulsos selectivos (ver más adelante).

Debido a las limitaciones del espectrómetro y de los parámetros de nuestra molécula de TCE hay que tomar en cuenta diversos factores: la dificultad de hacer operaciones de un bit sobre uno de los carbonos, el acoplamiento débil

entre H y C_2, etc., cuando se elije cual de los núcleos es el bit de control. Para los experimentos llevados a cabo aquí la mejor elección al tomar en cuenta todos estos factores resultó ser usar C_1 como bit de control, H como bit menos significativo y C_2 como bit más significativo. La razón principal para esta elección es que la mayoría de las operaciones de dos qubits involucran al bit de control y alguno de los dos restantes. Es natural entonces que el acoplamiento sea lo más grande posible para que estas operaciones sean lo más rápidas posibles.

Uno de los obstáculos más importantes para poder llevar a cabo este experimento con un error razonable es la implementación de la compuerta de Toffoli. Afortunadamente, la mitad de los puntos del espacio de fase se pueden medir sin dicha compuerta (los valores de $w(q,p)$ con $q = 0$ y $q = 2$). Para poder medir el resto es necesario un poco de ingenio. La dificultad principal es la siguiente: debido a que sólo podemos implementar operaciones de dos qubits es necesario descomponer la compuerta de Toffoli en compuertas CNOT y operaciones de un qubit. Al hacer dicha descomposición nos damos cuenta que hace falta hacer compuertas CNOT entre los bits 1-2, 1-3 y 2-3. Afortunadamente, la molécula de TCE tiene un acoplamiento muy chico entre 1-3 por lo que hacer un CNOT entre esos qubits es muy difícil. Sin embargo, debido a que algunas operaciones del final pueden ser ignoradas lo que se puede hacer es diseñar una compuerta Toffoli cuya última compuerta CNOT este entre H y C_2 y esa operación puede ser ignorada por no modificar el valor de $\langle \mathbf{I}_z \rangle$. En la figura 5.15 vemos la descomposición que usamos.

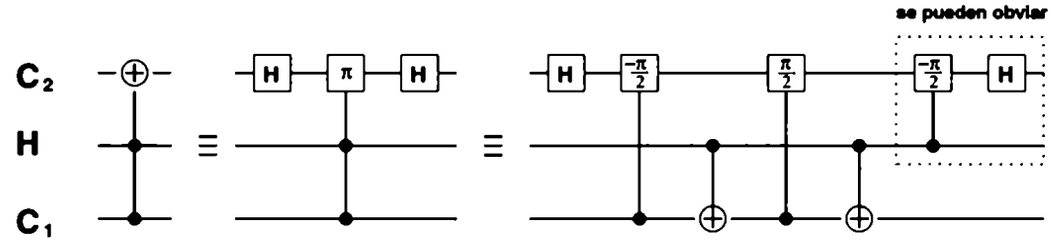


Figura 5.15: Descomposición de la compuerta de Toffoli en operaciones de 1 y 2 qubits.

Una vez que se tienen todos los elementos para poder hacer los experimentos bien definidos y pensados es importante hacer el último proceso de optimización para reducir al mínimo el número de pulsos. Este proceso es similar al que un programador encara cuando intenta optimizar las secciones críticas de código. En esos casos se busca reducir el tiempo de ejecución del programa utilizando diversas técnicas. Análogamente, es posible (e importante) reducir las secuencias a su mínima expresión. En el caso de RMN el objetivo no es tanto reducir el tiempo de ejecución del programa sino minimizar los errores que se generan a lo largo del cómputo. Para ello conviene hacer las secuencias lo más simples y cortas posible. Uno de los trucos más provechosos para esto es reducir al mínimo la cantidad de rotaciones en torno al eje z utilizando el método de seguimiento de fase. Para las mediciones de la función de Wigner este truco junto con la compuerta de Toffoli simplificada fueron fundamentales.

Para hacer los experimentos generamos los 4 estados computaciones de nuestras dos espines utilizando la técnica de promedio temporal [Knill98b] para obtener la parte de la matriz densidad que no es proporcional a la identidad. Los resultados de medir la función de Wigner se pueden ver en la figura 5.16.

Para estos gráficos se hicieron un total de 64 experimentos para medir la función de Wigner discreta en los 16 posibles puntos del espacio de fase para cada uno de los siguientes 4 estados del sistema: $\mathbf{1}$, $\mathbf{I_z^1}$, $\mathbf{I_z^2}$ y $\mathbf{I_z^1 I_z^2}$ de nuestros espínes. Utilizando promedio temporal se reconstruyó la función de Wigner de los 4 estados de la base computacional. Por ejemplo, para reconstruir la función de Wigner del estado $|00\rangle\langle 00|$ usamos que $|00\rangle\langle 00| = 1/4(\mathbf{1}+\mathbf{I_z})^1(\mathbf{1}+\mathbf{I_z})^2 = 1/4(\mathbf{1}+\mathbf{I_z^1}+\mathbf{I_z^2}+\mathbf{I_z^1 I_z^2})$ por lo que sumando las distribuciones de Wigner de los 4 estados iniciales obtenemos el resultado de la primer figura de 5.16. Las otras se obtienen sumando los cuatro estados con los signos apropiados. Los resultados fueron obtenidos con dos espectrómetros comerciales de 500MHz (un modelo AM-500 del LANAIS y un DRX-500 del Los Alamos National Laboratory ambos de la marca Bruker). Se usó un cabezal de 5mm sintonizado a ^{13}C y ^{1}H a las frecuencias de 125.77 MHz y 500.13 MHz, respectivamente. El error experimental medido es de aproximadamente el 15% en el peor de los casos. Las fuentes más importantes de errores son entendidas y provienen de efectos de acoplamiento fuerte (que modifican el comportamiento de nuestras compuertas para ciertos estados) e incerteza numérica al integrar los espectros. Sin embargo, es claro que los resultados del experimento demuestran el método tomográfico de la función de Wigner y concuerdan muy bien con los resultados teóricos. De los resultados experimentales es posible reconstruir la matriz densidad utilizando la relación (4.30). Los resultados se pueden ver en la figura 5.17 donde se grafica $|\langle n|\hat{\rho}|n'\rangle|$ a partir de las cuatro funciones de Wigner obtenidas experimentalmente. La figura da una mejor idea del error experimental presente en nuestros resultados.

5.6. Resumen histórico de experimentos en RMN

Es importante aclarar que la mayoría de las técnicas presentadas aquí fueron desarrolladas por numerosos investigadores a lo largo de los últimos cuatro años. Es conveniente, por lo tanto, hacer un breve resumen histórico para poner en contexto los resultados originales presentados en esta tesis. La presente explicación sigue a Jones [Jones01b] y recomendamos leer ese artículo para tener un buen pantallazo del estado actual de la computación cuántica y RMN.

La idea original de utilizar RMN fue propuesta por Gershenfeld y Chuang [Gershenfeld97]. Allí los autores presentan su idea y muestran una técnica relativamente complicada de generar estados pseudo-puros (que es, sin duda, el obstáculo más importante de superar para poder mostrar que es posible hacer computación cuántica con RMN). Sin embargo, fue David Cory quien en [Cory98a] marcó el camino para futuras investigaciones en RMN. Allí no sólo presenta la teoría y la notación que usamos en esta tesis (y en muchas otras publicaciones) sino que además presenta los primeros resultados experi-

mentales simples. Entre ellos: la generación de estados pseudo-puros en dos qubits, implementación de una compuerta cuántica en dos qubits y una compuerta de Toffoli. Los resultados presentados en la tesis son simplemente una reproducción de estos primeros pasos para evaluar las limitaciones de nuestro espectrómetro y explorar sus posibilidades.

Los trabajos de Gershenfeld y Chuang junto con el de Cory y sus colaboradores, abrieron el camino para implementar todo tipo de algoritmos cuánticos. Los primeros algoritmos utilizaban 2 y en el mejor de los casos 3 qubits. Entre los de dos qubits se puede destacar: el algoritmo de Deutch [Chuang98, Jones98a], Deutch-Jozsa (una generalización del algoritmo de Deutch), búsqueda en una base de datos utilizando el algoritmo de Grover [Jones98b, Jones01a], y "quantum counting" (una generalización del algoritmo de Grover) [Jones99]. Con tres qubits se logró demostrar la eficiencia de los códigos de corrección de errores. Para ello se demostró cómo es posible proteger información utilizando el código de 3 bits [Cory98b]. También se llevó a cabo un experimento de teleportación [Nielsen98] donde se 'teleportó' el estado de un qubit entre núcleos de una molécula de TCE.

A pesar del éxito de todos estos experimentos es claro que trabajar con un número mayor de qubits resulta bastante complicado y engorroso. No obstante hay resultados alentadores: se demostró que es posible operar con sistemas de hasta 7 qubits [Knill00], se implementó el código perfecto de corrección de 5 bits [Knill01] y hasta se pudo demostrar, primero parte del algoritmo de Shor con 5 qubits [Vandersypen00] y, recientemente, el algoritmo completo [Vandersypen01] utilizando 7 qubits. Estos últimos experimentos constituyen el "state of the art" del momento y los investigadores que los llevan a cabo cuentan con espectrómetros de última generación, químicos que sintetizan moléculas artificialmente para contar con Hamiltonianos apropiados y un saludable financiamiento. En este contexto nuestro aporte al campo de RMN puede ser considerado modesto aunque el circuito genérico de medición de la función de Wigner es potencialmente muy útil. Si en poco tiempo somos capaces de simular sistemas cuánticos con una computadora cuántica será natural querer mirar los resultados en una representación adecuada: el espacio de las fases. Nuestro circuito resultará entonces de utilidad.

5.7. Resumen

En este último capítulo presentamos con algún cuidado todas las herramientas para poder entender cómo se puede implementar una computadora cuántica usando un espectrómetro de RMN. Después de explicar el marco teórico general nos enfocamos un poco en el caso particular de la molécula de tricloroetileno que es la que usamos para nuestros experimentos. Presentamos los resultados experimentales de cómo preparar estados psudo-puros utilizando promedio temporal y promedio espacial y cómo hacer una compuerta CNOT simple. Finalmente presentamos como resultado original de esta tesis la implementación de un algoritmo cuántico para medir la función de Wigner de un sistema de dos qubits.

Láminas color

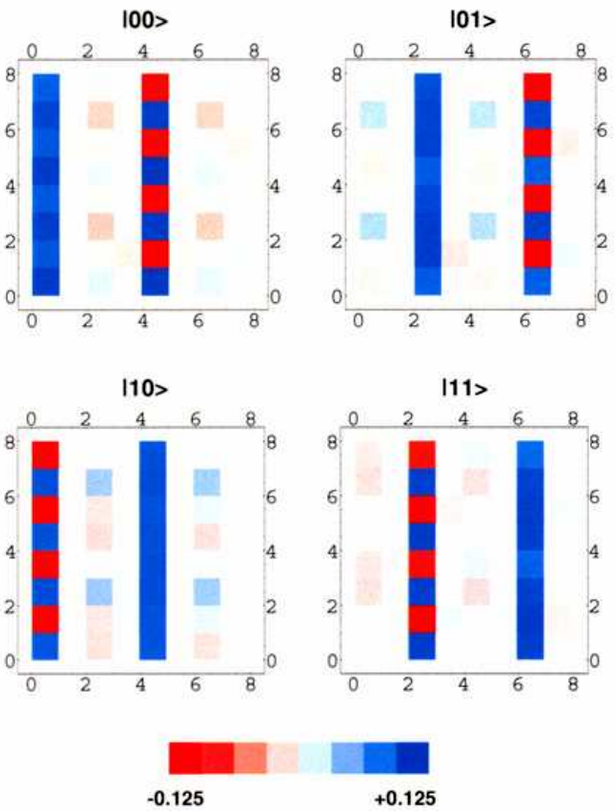


Figura 5.16: Resultados experimentales de la medición de la función de Wigner de los cuatro estados computacionales para un sistema de 2 qubits. Idealmente la funciones de Wigner son distintas de cero sólo en dos franjas verticales. En una de ellas toma el valor $1/8$ (para todo p) mientras que en la otra alterna entre $\pm 1/8$. El error experimental es en el peor de los casos aproximadamente el 15%.

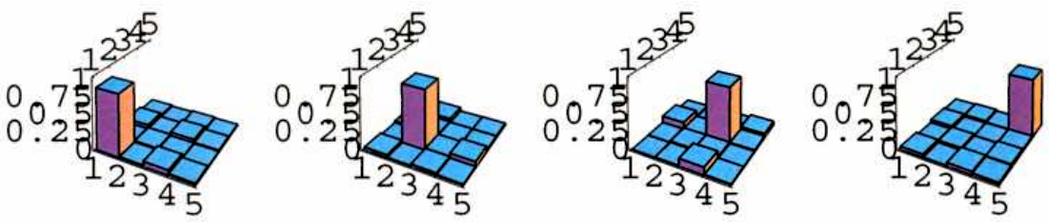


Figura 5.17: Matrices densidad reconstruidas a partir de los datos experimentales de la función de Wigner para los cuatro estados: $|00\rangle$, $|01\rangle$, $|10\rangle$ y $|11\rangle$. En el gráfico se muestra el valor absoluto de $\langle n|\hat{\rho}|n'\rangle$.

6. Conclusiones y perspectivas

Resumiremos aquí los resultados fundamentales de esta tesis. Uno de los obstáculos más difíciles de superar en computación cuántica es la sensibilidad de la misma a su acoplamiento con el entorno y a los errores posibles que se puedan generar al implementar algoritmos cuánticos. Nuestro trabajo [Miquel97] fue la primer simulación completa de un algoritmo relativamente complicado para una computadora cuántica basada en la trampa de iones. Allí mostramos claramente que cualquier intento por implementar un algoritmo relativamente complicado va a fracasar si no se utiliza algún método de corrección de errores. Lo que es más importante, encontramos una expresión simple para *predecir* la fidelidad como función del número de pulsos láser y la dispersión en los errores. Esta fórmula aproxima sorprendentemente bien los resultados numéricos y por lo tanto es de gran utilidad práctica.

A pesar de las dificultades que se presentan debido a la sensibilidad de las computadoras cuánticas es teóricamente posible implementar en la práctica cualquier algoritmo utilizando códigos de corrección de errores y computación tolerante a fallas. En esta área nuestra contribución con el código de 5 qubits fue pionera: no sólo encontramos el mínimo código que corrige todos los posibles errores de un qubit sino que encontramos un circuito que, servía tanto para codificar como para decodificar (invirtiendolo). Estas observaciones sirvieron para encontrar una descripción más sistemático de los códigos de corrección: la descripción utilizando el formalismo de estabilizadores.

Finalmente el último aporte interesante que hemos presentado aquí es la utilización de la función de Wigner discreta para entender los algoritmos de computación cuántica. Nuestro enfoque intenta utilizar la función de Wigner como herramienta para entenderlos y posiblemente hallar nuevos. Vale la pena aclarar que aquí sólo hemos presentado indicios de que puede resultar útil tener una representación en el espacio de fases de la computadora cuántica. Creemos que todavía hay bastantes cosas que investigar en ésta dirección. Además la función de Wigner y nuestro circuito para medirla son importantes a la hora de usar la computación cuántica para simular sistemas físicos cuánticos. Sin duda esta es un área en la cual veremos avances interesantes en los próximos años y disponer de un circuito simple para medir directamente la función de Wigner resultará ventajoso.

Agradecimientos

En primer lugar quiero agradecer a mi director Juan Pablo Paz. Desde fines de mi carrera he tenido la suerte de trabajar con él tanto en mi tesis de licenciatura como en mi doctorado. Sin duda puedo decir que es el mejor físico que conocí y es un honor haber hecho mi doctorado con él. Sin sus consejos y su apoyo esta tesis no existiría. También quiero agradecer a mis padres, el resto de mi familia y amigos por todo el apoyo que me dieron durante todos estos años de trabajo. Aquí hay una lista (espero que completa!) de toda la gente que ya sea académicamente como personalmente contribuyeron y me acompañaron durante estos años:

Juan Pablo Paz
Marcos Saraceno
Christof Zalka
Claudio Dorso
Susana Mirande
Maria Miquel
Malu
Alejandro Strachan
Juan Peralta
UBA
Raymond Laflamme
Wojciech Zurek
Michael Nielsen
Diego Dalvit
Cesar A. Miquel
Lucia Mirande
Viviana Visus
Lucas Mingarro
LANAIS
Emanuel Knill
Camille Negrevergne
D. DiVincenzo
Fernando Lombardo
Facundo Miquel
Mario Santomauro
Ariel Chernomoretz
Pablo Balenzuela
Dto. Fisica FCEyN

Notas y observaciones de los jurados

A. Implementación de Cirac y Zoller

En este apéndice describiremos la propuesta de Cirac y Zoller para hacer una computadora cuántica utilizando iones fríos. En una trampa de iones se puede confinar y enfriar un conjunto de iones a temperaturas cercanas a los μK. Los iones son atrapados utilizando láseres que los enfrían mediante colisiones inelásticas con los fotones emitidos por el láser. En la implementación propuesta por Cirac y Zoller los iones son atrapados en un arreglo unidimensional como vemos en la figura A.1. Una vez atrapados, los iones oscilan en un potencial armónico efectivo en la dirección $\hat{x}$ debido a su repulsión coulombiana y la interacción con campos electromagnéticos de confinamiento.

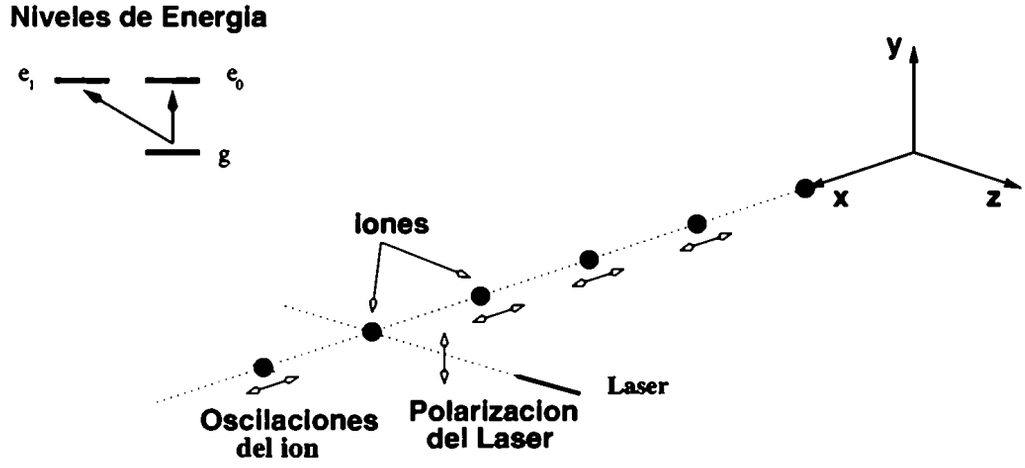


Figura A.1: Esquema de la implementación propuesta por Cirac y Zoller. Los iones están confinados a moverse en una dimensión por campos electromagnéticos. Los iones son excitados por láseres e interactúan entre sí a atravez de las oscilaciones del modo colectivo del centro de masa.

Los iones tienen niveles de energía internos que se usan para almacenar la información de un qubit. En esta implementación son necesarios tres estados: $|g\rangle$, $|e_0\rangle$ y $|e_1\rangle$. El primero es el estado fundamental mientras que $|e_0\rangle$ y $|e_1\rangle$ son dos estados excitados. Para excitar los electrones a uno u otro estado se utiliza un láser con distinta polarización q.

El movimiento de los iones puede ser descripto por un potencial armónico anisótropo de frecuencias $\nu_x \ll \nu_y, \nu_z$ donde los iones están alineados en la dirección $\hat{x}$. Enfriándolos con láseres, los iones ejecutan pequeñas oscilaciones entorno a la posición de equilibrio. Se los puede describir entonces en términos de modos normales. Vamos a suponer que los iones están oscilando en el estado fundamental (ningún fonón).

Si se le aplica un pulso láser al j'ésimo ion por un tiempo $t = k\pi/\Omega$ con frecuencia Ω (la frecuencia de Rabi) y una polarización elegida para excitar el estado $|e_0\rangle$, el estado del ion evoluciona con el siguiente Hamiltoniano:

$$\hat{H}_j = (\Omega/2)\left[|e_0\rangle_j\langle g|e^{-i\Phi} + |g\rangle_j\langle e_o|e^{i\phi}\right]$$

donde Φ es la fase del láser. Para que éste sea el Hamiltoniano del sistema deben verificarse ciertas suposiciones acerca de la posición del ion con respecto a la onda estacionaria del láser (Ver [Cirac95]). La evolución de un estado $|\psi\rangle$ esta dado por el operador de evolución temporal:

$$\hat{V}_j^k(\Phi) = \exp\left[-ik\frac{\pi}{2}(|e_0\rangle_j\langle g|e^{-i\Phi} + C.C.)\right]$$

Es fácil ver que la matriz del operador V_j^k en la base $|g\rangle$, $|e\rangle_0$ es:

$$V_j^k(\Phi) = \begin{pmatrix} \cos k\pi/2 & -ie^{-i\Phi}\sin k\pi/2 \\ -ie^{i\Phi}\sin k\pi/2 & \cos k\pi/2 \end{pmatrix} \tag{A.1}$$

Este operador corresponde a una rotación en el plano complejo. Con esta operación y una compuerta no-controlado tenemos conjunto de compuertas universales. Veamos cómo podemos implementar una compuerta no-controlada con la trampa de iones.

Para ello, debemos encontrar algún mecanismo para que se modifique el estado de uno de los iones dependiendo del estado del otro. De alguna manera debemos "comunicar" información entre los dos bits. Para lograr esto se usan las excitaciones de las oscilaciones del centro de masa del sistema. Supongamos que inicialmente el sistema está en el estado fundamental (cero fogones). Si le aplicamos un pulso láser de frecuencia $\omega = \omega + \nu_x$ (donde ω es la frecuencia necesaria para excitar al i'esimo ion) por un tiempo τ además de excitar el ion, la energía sobrante excitara el primer modo del centro de masa. Si el ion ya estaba excitado el estado no cambiará pues el resto de la energía no tendráádonde ir. Si ahora hacemos una operación similar sobre otro ion, el resultado final dependerá del estado del primer ion. Vemos que transfiriendo parte de la energía al c.m. podemos hacer operaciones sobre un ion condicionadas por el estado del otro.

Veamos más concretamente como podemos explotar ésta característica del modelo. Si aplicamos un pulso láser con polarización q sobre el n'esimo ion con una frecuencia $\omega = \omega - \nu_x$ el Hamiltoniano que describe esta interacción es:

$$\hat{H}_{n,q} = \frac{\eta}{\sqrt{N}}\frac{\Omega}{2}\left[|e_q\rangle_n\langle g|ae^{-i\Phi} + |g\rangle\langle e_q|_n a^\dagger e^{i\Phi}\right] \tag{A.2}$$

Aquí a y $a^\dagger$ son los operadores que destruyen y crean (respectivamente) un fonón correspondiente a un modo normal del centro de masa, $\eta = [\hbar k_\theta^2/(2M\nu_x)]^{1/2}$ es parámetro Lamb-Dicke, $k_\theta = k\cos\theta$ es la proyección del vector de onda k del láser sobre el eje x, M es la masa del ion y N el número de iones en la trampa. El subíndice $q = 0, 1$ se refiere a la polarización del láser y define cual de los dos estados del ion es excitado ($\langle e_0|$ o $\langle e_1|$).

Si el láser se prende por un tiempo $t = k\pi/(\Omega\eta/\sqrt{N})$, el operador de evolución temporal será:

$$\hat{U}_n^{k,q} = \exp\left[-ik\frac{\pi}{2}\left(|e_q\rangle_n\langle g|ae^{-i\Phi} + C.C.\right)\right] \tag{A.3}$$

Este operador evoluciona los estados $|g\rangle_n|1\rangle$ y $|e_q\rangle_n|0\rangle$ de la siguiente manera:

$$\begin{aligned} |g\rangle_n|1\rangle &\rightarrow \cos(k\pi/2)|g\rangle_n|1\rangle - ie^{i\Phi}\sin k\pi/2|e_q\rangle_n|0\rangle, \\ |e_q\rangle_n|0\rangle &\rightarrow \cos(k\pi/2)|e_q\rangle_n|0\rangle - ie^{-i\Phi}\sin k\pi/2|g\rangle_n|1\rangle \end{aligned} \tag{A.4}$$

Los vectores $|0\rangle$ y $|1\rangle$ representan el estado con ningún y un fonón respectivamente. Si definimos el operador $U_{m,n} \equiv \hat{U}_m^{1,0}\hat{U}_n^{2,1}\hat{U}_m^{1,0}$ que opera sobre el n'esimo y el m'esimo ione, es fácil verificar que:

	$\hat{U}_m^{1,0}$		$\hat{U}_n^{2,1}$		$\hat{U}_m^{1,0}$	
$\|g\rangle_m\|g\rangle_n\|0\rangle$	$\rightarrow$	$\|g\rangle_m\|g\rangle_n\|0\rangle$	$\rightarrow$	$\|g\rangle_m\|g\rangle_n\|0\rangle$	$\rightarrow$	$\|g\rangle_m\|g\rangle_n\|0\rangle$
$\|g\rangle_m\|e_0\rangle_n\|0\rangle$	$\rightarrow$	$\|g\rangle_m\|e_0\rangle_n\|0\rangle$	$\rightarrow$	$\|g\rangle_m\|e_0\rangle_n\|0\rangle$	$\rightarrow$	$\|g\rangle_m\|e_0\rangle_n\|0\rangle$
$\|e_0\rangle_m\|g\rangle_n\|0\rangle$	$\rightarrow$	$-i\|g\rangle_m\|g\rangle_n\|1\rangle$	$\rightarrow$	$i\|g\rangle_m\|g\rangle_n\|1\rangle$	$\rightarrow$	$\|e_0\rangle_m\|g\rangle_n\|0\rangle$
$\|e_0\rangle_m\|e_0\rangle_n\|0\rangle$	$\rightarrow$	$-i\|g\rangle_m\|e_0\rangle_n\|1\rangle$	$\rightarrow$	$-i\|g\rangle_m\|e_0\rangle_n\|1\rangle$	$\rightarrow$	$-\|e_0\rangle_m\|e_0\rangle_n\|0\rangle$

Se puede ver que el único resultado de la interacción es cambiarle el signo al estado $|e_0\rangle_m|e_0\rangle_n|0\rangle$ mientras que deja a los demás estados invariantes. Notese también que el estado del centro de masa siempre vuelve al estado fundamental (el estado con cero fonones). La ecuación anterior es equivalente al no-controlado. Si definimos $|\pm\rangle \equiv \frac{1}{\sqrt{2}}(|g\rangle \pm |e\rangle_0)$ entonces la secuencia de operaciones $\hat{U}_{m,n}$ aplicadas al vector $|g\rangle|\pm\rangle \rightarrow |g\rangle|\pm\rangle$ mientras que aplicado al estado $|e\rangle|\pm\rangle \rightarrow |g\rangle|\mp\rangle$. La transformación entre los estados $|g\rangle_n, |e_0\rangle_n$ y $|+\rangle_n, |-\rangle_n$ se puede implementar con el operador $V_n^{1/2}(\pi/2)$ que mostramos al principio.

El no-controlado se construye entonces aplicando las siguientes cinco transformaciones a nuestros iones:

$$\hat{U}_{no-cont.} = \hat{V}_n^{1/2}(\pi/2)\hat{U}_m^{1,0}\hat{U}_n^{2,1}\hat{U}_m^{1,0}\hat{V}_n^{1/2}(-\pi/2) \tag{A.6}$$

Con estos cinco pulsos láseres logramos nuestro propósito: la negación del bit m controlada por el bit n. Cirac y Zoller muestran que, usando los operadores $\hat{U}_{m,n}$ y $V_n^k(\Phi)$ también es posible implementar una compuerta de Toffoli (usando 7 pulsos láseres) y cualquier otra compuerta no-(controlada)^p. [1]

[1] La compuerta no-(controlada)^p es la generalización de la compuerta no-controlada pero en vez de uno tiene p bits de control. En ésta notación, la compuerta de Toffoli es una no-(controlada)^2.

De la implementación presentada aquí vemos las fuentes de errores que estudiamos en el capítulo 3. Si el tiempo durante el cual aplicamos un pulso láser sobre un ion para realizar la operación $V_j^k(\Phi)$ no es exactamente un múltiplo de π/Ω entonces el ion puede quedar en una combinación lineal incorrecta. En el capítulo 3 analizamos los efectos de errores en la duración de los pulsos y las fases de los láseres.

B. Secuencias de pulsos RMN

En este apéndice mostramos las secuencias de pulsos para generar estados pseudo-puros utilizando promedio temporal. Primero presentaremos las cuatro secuencias necesarias para generar el estado $|0\rangle\langle 0| \otimes \mathbf{I}_z \otimes |0\rangle\langle 0|$. Para generar este estado necesitamos generar: $\mathbf{I}_z^2$, $2\mathbf{I}_z^1\mathbf{I}_z^2$, $2\mathbf{I}_z^2\mathbf{I}_z^3$ y $4\mathbf{I}_z^1\mathbf{I}_z^2\mathbf{I}_z^3$. Las secuencias parten del estado térmico $\mathbf{I}_z^1 + \mathbf{I}_z^2 + \mathbf{I}_z^3$.

SECUENCIA PARA GENERAR $\mathbf{I}_z^2$

$$\mathbf{I}_z^1 + \mathbf{I}_z^2 + \mathbf{I}_z^3$$
$$\xrightarrow{[\pi/2]_x^{23}} \mathbf{I}_z^1 - \mathbf{I}_y^2 - \mathbf{I}_y^3$$
$$\xrightarrow{[\mathrm{grad}]_z} \mathbf{I}_z^1$$
$$\xrightarrow{[\pi/2]_y^1} \mathbf{I}_x^1$$
$$\xrightarrow{[\pi/2]_{J_{12}}} 2\mathbf{I}_y^1\mathbf{I}_z^2$$
$$\xrightarrow{[\pi/2]_x^1} 2\mathbf{I}_z^1\mathbf{I}_z^2$$
$$\xrightarrow{[\pi/2]_y^2} 2\mathbf{I}_z^1\mathbf{I}_x^2$$
$$\xrightarrow{[\pi/2]_{J_{12}}} \mathbf{I}_y^2$$
$$\xrightarrow{[\pi/2]_x^2} \mathbf{I}_z^2$$

SECUENCIA PARA GENERAR $2\mathbf{I}_z^2\mathbf{I}_z^3$

$$\mathbf{I}_z^1 + \mathbf{I}_z^2 + \mathbf{I}_z^3$$
$$\xrightarrow{[\pi/2]_x^{23}} \mathbf{I}_z^1 - \mathbf{I}_y^2 - \mathbf{I}_y^3$$
$$\xrightarrow{[\mathrm{grad}]_z} \mathbf{I}_z^1$$
$$\xrightarrow{[\pi/2]_y^1} \mathbf{I}_x^1$$
$$\xrightarrow{[\pi/2]_{J_{12}}} 2\mathbf{I}_y^1\mathbf{I}_z^2$$
$$\xrightarrow{[\pi/2]_x^1} 2\mathbf{I}_z^1\mathbf{I}_z^2$$
$$\xrightarrow{[\pi/2]_y^2} 2\mathbf{I}_z^1\mathbf{I}_x^2$$
$$\xrightarrow{[\pi/2]_{J_{12}}} \mathbf{I}_y^2$$
$$\xrightarrow{[\pi/2]_{J_{23}}} 2\mathbf{I}_x^2\mathbf{I}_z^3$$
$$\xrightarrow{[-\pi/2]_y^2} 2\mathbf{I}_z^2\mathbf{I}_z^3$$

SECUENCIA PARA GENERAR $2\mathbf{I}_z^1\mathbf{I}_z^2$

$$\mathbf{I}_z^1 + \mathbf{I}_z^2 + \mathbf{I}_z^3$$
$$\xrightarrow{[\pi/2]_x^{23}} \mathbf{I}_z^1 - \mathbf{I}_y^2 - \mathbf{I}_y^3$$
$$\xrightarrow{[\mathrm{grad}]_z} \mathbf{I}_z^1$$
$$\xrightarrow{[\pi/2]_y^1} \mathbf{I}_x^1$$
$$\xrightarrow{[\pi/2]_{J_{12}}} 2\mathbf{I}_y^1\mathbf{I}_z^2$$
$$\xrightarrow{[\pi/2]_x^1} 2\mathbf{I}_z^1\mathbf{I}_z^2$$

SECUENCIA PARA GENERAR $4\mathbf{I}_z^1\mathbf{I}_z^2\mathbf{I}_z^3$

$$\mathbf{I}_z^1 + \mathbf{I}_z^2 + \mathbf{I}_z^3$$
$$\xrightarrow{[\pi/2]_x^{23}} \mathbf{I}_z^1 - \mathbf{I}_y^2 - \mathbf{I}_y^3$$
$$\xrightarrow{[\mathrm{grad}]_z} \mathbf{I}_z^1$$
$$\xrightarrow{[\pi/2]_y^1} \mathbf{I}_x^1$$
$$\xrightarrow{[\pi/2]_{J_{12}}} 2\mathbf{I}_y^1\mathbf{I}_z^2$$
$$\xrightarrow{[\pi/2]_x^1} 2\mathbf{I}_z^1\mathbf{I}_z^2$$
$$\xrightarrow{[-\pi/2]_x^2} 2\mathbf{I}_z^1\mathbf{I}_y^2$$
$$\xrightarrow{[\pi/2]_{J_{23}}} -4\mathbf{I}_z^1\mathbf{I}_x^2\mathbf{I}_z^3$$
$$\xrightarrow{[-\pi/2]_y^{23}} 4\mathbf{I}_z^1\mathbf{I}_z^2\mathbf{I}_x^3$$
$$\xrightarrow{[-\pi/2]_y^3} 4\mathbf{I}_z^1\mathbf{I}_z^2\mathbf{I}_z^3$$

Si bien en teoría la secuencias son sencillas hay que recordar que hay varias limitaciones al trabajar con un espectrómetro de dos canales (^{1}H y ^{13}C. Para ver las dificultades que hay al pasar las secuencias a programas del espectrómetro aquí mostramos el listado (comentado) de la secuencia que genera $4\mathbf{I}_z^1\mathbf{I}_z^2\mathbf{I}_z^3$ en un formato casi idéntico al usado por el espectrómetro Bruker DRX 500.

Todas las secuencias comienzan de la misma manera: un pulso sobre los carbonos para crear polarización en en C y después un gradiente para destruirla:

```
; pi/2 X sobre C
(p1 ph0):f1
3u
```

2. Nociones básicas

2.1. Computadoras clásicas

Para poder entender muchas de las ideas y conceptos usados en computación cuántica es conveniente hacer una breve discusión acerca de la teoría clásica de la computación. Introduciremos primero los conceptos de algoritmo y máquina de Turing, para después hablar sobre complejidad algorítmica y las distintas clases de complejidad. Después haremos una breve introducción a los circuitos clásicos usados para describir algoritmos y finalmente introduciremos el análogo cuántico y usaremos el algoritmo de Grover para ver cómo es posible utilizar una computadora cuántica para resolver un problema más eficientemente que lo que puede hacer cualquier computadora clásica.

2.1.1. Algoritmos, máquinas de Turing y complejidad algorítmica

Un algoritmo no es otra cosa que un procedimiento sistemático que consiste de una serie de pasos a repetir hasta llegar a la solución de un problema. Para definir más rigurosamente la noción de algoritmo y su implementación en computadoras hace falta formalizar el concepto de "computadora" y "procedimiento sistemático". Alan Turing encontró una forma de hacerlo creando una máquina abstracta que hoy se conoce como máquina de Turing [Turing37].

Una máquina de Turing es una computadora llevada al nivel más básico que podemos imaginar. Posee un número finito de estados internos que llamaremos n_i, $i = 0, \ldots, M$. La máquina tiene además un cabezal $\check{\ell}$ que es capaz de hacer dos tipos de símbolos sobre una cinta infinita (ver figura 2.1). La cinta pasa por debajo del cabezal y se puede mover en dos direcciones: para adelante o para atrás. Esta dividida en celdas $\ldots, -1, 0, 1 \ldots$ donde se puede escribir dos tipos de símbolos etiquetados: 0 y 1.

¿Cómo opera la máquina? La computadora está en el estado interno n_i definido con el cabezal sobre una de las celdas de la cinta. Lee el símbolo s que está debajo del cabezal y realiza alguna combinación de las siguientes acciones: modifica el valor sobre la cinta a s', cambia de estado interno a n_j y/o mueve el cabezal hacia una de las dos direcciones posibles $\sigma = \pm 1, 0$. El conjunto de cinco números: $n_i(s, s', \sigma)n_j$ define a nuestra máquina de Turing y el algoritmo que se está implementando.

En la definición anterior de máquina de Turing el programa está codificado en la tabla de estados internos y las posibles transiciones. Sin embargo se podría incluir el programa como parte de la entrada en la cinta. A esta máquina se la conoce como máquina de Turing Universal. En la teoría de la computación se "define" algoritmo como aquel procedimiento sistemático que puede ser llevado a cabo por una máquina de Turing Universal. La palabra "define" aparece entre comillas porque en realidad esto no es una definición de

```
gon:ngrad ; gradiente
gcrush
goff:ngrad
gwait
200m
```

Aunque pueda parecer poco inteligente en un principio la idea es ahora transferir la polarización restante (la que está en ^{1}H) al resto del sistema. La razón es que la polarización del ^{1}H es cuatro veces más grande que la de ^{13}C. Además, al haber menos operadores producto se acoplan menos términos iniciales y eso resulta en estados mas "limpios". Observar también el tiempo de espera de $200ms$ para dejar que desparezcan las corrientes inducidas al variar prender y apagar los gradientes.

El próximo paso consiste en transferir la polarización al resto del sistema utilizando el acoplamiento J_{12} entre H y C_1. Dejando actuar el acoplamiento por tiempo $1/(2J_{12})$ se pueden transformar (usando también pulsos selectivos) términos de la forma $\mathbf{I}_z^1 \Leftrightarrow 2\mathbf{I}_z^1\mathbf{I}_z^2$. Aquí usamos ese truco. Observar además que, al no haber polarización en C_2 se puede ahorrar un pulso selectivo simplemente usando un pulso "duro" sobre ambos carbonos:

```
; pi/2 Y sobre H
(p2 ph1):f2
; XII

; pi/2 acoplamiento H-C1
"d6=0.5s/cnst1"
d6
; YZI

(p1 ph2):f1  ; pi/2 -X sobre C1-C2
3u
(p2 ph0):f2  ; pi/2 X sobre H
; ZYI
```

Ahora hacemos un acoplamiento J_{23}. Para ello se hace un pulso de reenfoque sobre ambos carbonos (para matar el acoplamiento H-C_1, H-C_2 y la rotación en C_2. Esta parte termina con un pulso π sobre ambos carbonos para cambiar el signo al operador producto:

```
; pi/2 C1-C2 coupling
"d7=(0.5s/cnst2 - p3)/2"
d7
(p3 ph0):f1   ; pi X sobre C
d7
```

```
; -ZXZ + basura

(p1 ph3):f1    ; pi/2 -Y sobre C
3u
; ZZX + basura
```

Debido a acoplamientos no deseados queda, en este punto, además del operador $4\mathbf{I}_z^1\mathbf{I}_z^2\mathbf{I}_x^3$ otros términos no deseados ("basura"). Después nos ocuparemos de ellos. La secuencia sigue con un pulso selectivo sobre C_2:

```
; ZZX + basura
; Pulso selectivo sobre C2

(p1 ph0):f1     ; pi/2 X sobre C
"d24 = 1s/(8*cnst4) - p2 - p1"
d24
(p4 ph0):f2     ; pi X sobre H
d24
(p1 ph2):f1     ; pi/2 -X sobre C

; ZZZ + basura
```

Notar el pulso de re-enfoque sobre H en el medio para matar parte del acoplamiento J_{12} y J_{13} que pueda surgir durante este periodo.

La secuencia termina con un gradiente para deshacernos de los términos sobrantes ("basura") que tenga polarización transversal.

```
gon:ngrad ; gradiente para eliminar la basura
gcrush
goff:ngrad
gwait
```

Bibliografía

[Aharanov96] Aharonov, D. y Ben-Or, M., *Fault tolerant quantum computation with constant error*, quant-ph/9611025.

[Arnold68] Arnold, V.I. y Avez, A., *Ergodic problems in classical mechanics*, Benjamin, New York, 1968.

[Bacon00] Bacon, D., Kempe, J., Lidar, D.A., Whaley, K.B., *Universal fault-tolerant quantum computation on decoherence-free subspaces*, Phys. Rev. Lett. **85**, 1758 (2000).

[Barenco95] Barenco, A., *A Universal two-bit gate for quantum computation*, quant-ph/9505016.

[Bennet73] Bennet, C.H., IBM Journal of Research and Development, **17**, 525. (1973).

[Bennet96] Bennett, C.H., DiVincenzo, D.P., Smolin, J.A., Wooters, W.K, *Mixed state entanglement and quantum error correction*, quant-ph/9604024.

[Bertrand87] Bertrand, J. y Bertrand, P., Found. Phys. **17**, 397 (1987)

[Bianucci01] Bianucci, P., Miquel, C., Paz, J.P. y Saraceno, M., *Discrete Wigner functions and the phase space representation of quantum computers*, Phys. Lett. A **299** 353 (2002). También en:quant-ph/0105091.

[Bouzouina96] Bouzouina, A. y De Bievre, S., Comm. Math. Phys. **178** (1996).

[Calderbank96] Calderbank, A.R., Rains, E.M., Shor, P.W. y Sloane, N.J., *Quantum error correction and orthogonal geometry*, Phys. Rev. Lett. **78**, 405 (1997).

[Chuang98] Chuang, I.L., Vandersypen, L.M., Zhou, X., Leung, D.W. y Lloyd, S., *Experimental realization of a quantum algorithm*, Nature **393**, 143 (1998).

[Cirac95] Cirac, J.I. y Zoller, P., *Quantum computation with cold trapped ions*, Phys. Rev. Lett., **74**, pg 4091, (1995).

[Cleve98] Cleve, R., Ekert, A., Macchiavello, C., y Mosca, M., *Quantum algorithms revisited*, Proc. R. Soc. Lond. A **454**, 339 (1998).

[Coppersmith94] Coppersmith, D., *An approximate fourier transform useful in quantum factoring*, IBM Research Report RC 19642. (1994).

[Cory98a] Cory, D.G., Price, M.D., y Havel, T.F., *Nuclear magnetic resonance spectroscopy: an experimentally accessible paradigm for quantum computing*, Physica D **120**, 82 (1998).

[Cory98b] Cory, D.G., Price, M.D., Maas, W., Knill, E., Laflamme, R., Zurek, W., Havel, T.F. y Somaroo, S.S., *Experimental quantum error correction*, Phys. Rev. Lett. **81**, 2152 (1998).

[Deutch85] Deutch, D., *Quantum theory, the church-turing principle and the universal quantum computer*, Proc. R. Soc. Lond., A **400**, pg 97, (1985).

[Deutch88] Deutch, D., *Quantum computational networks*, Proc. R. Soc. Lond. A **425**, pg 73, (1989).

[Deutch92] Deutch, D. y Jozsa, R., *Rapid solution of problems by quantum computation*, Proc. R. Soc. Lond. A **439**, pg 553, (1992).

[DiVincenzo00] DiVincenzo, D., *The physical implementation of quantum computation*, `quant-ph/0002077`.

[Dunn94] Dunn, T.J. *et al.*, Phys. Rev. Lett. **74** (1994) 884; Leibfried, D. *et al.* Phys. Rev. Lett. **77** (1996) 4281; ver además Physics Today **51** no. 4 (1998) 22.

[Duan98] Duan, L.M., Guo, G.C., Phys. Rev. A, **57**, 737 (1998).

[Fano57] Fano, U., Review of Modern Physics, **29**, 75 (1957).

[Feynman82] Feynman, R., *Simulating physics with computers*, Int. J. Theor. Phys. **21**, pg 467 (1982).

[Fredkin82] Fredkin, E., y Toffoli, T., Int. J. Theor. Phys. **21**, pg 219 (1982).

[Gershenfeld97] Gershenfeld, N.A. y Chuang, I.L., *Bulk spin-resonance quantum computation*, Science, **275**, 350 (1997).

[Gottesman96] Gottesman, D., *Class of quantum error-correcting codes saturating the quantum Hamming bound*, Phys. Rev. A **54**, 1862 (1996).

[Gottesman98] Gottesman, D., *A theory of fault-tolerant quantum computation*, Phys. Rev. A **57**, 127 (1998).

[Gottesman00] Gottesman, D., *An introduction to quantum error correction*, quant-ph/00004072.

[Grover96] Grover, L., Phys. Rev. Lett., **78**, 3252 (1997).

[Hannay80] Hannay, J.H. y Berry, M.V., Physica **1D**, 267 (1980).

[Jones98a] Jones, J. y Mosca, M., J. Chem. Phys., **109**, 1648 (1998).

[Jones98b] Jones, J., Mosca, M. y Hansen, R.H., Nature, **393**, 344 (1998).

[Jones99] Jones, J. y Mosca, M., Phys. Rev. Lett., **83**, 1050 (1999).

[Jones01a] Jones, J., Prog. NMR Spectrosc., **38**, 325 (2001).

[Jones01b] Jones, J. *Quantum computation and nuclear magnetic resonance*, quant-ph/0106067.

[Keyes89] Keyes, R.W., *Physics of digital devices*, Rev. Mod. Phys. **61**, No. 2 pg. 279 (1989).

[Kitaev96] Kitaev, A., *Quantum computing: algorithms and error correction*. Preprint en Ruso.

[Kitaev97] Kiteav, A., *Fault-tolerant quantum computation by anyons*, quant-ph/9707021.

[Knill98a] Knill, E., Laflamme, R., Zurek, W., *Resilient quantum computation: error models and thresholds*, Proc. R. Soc. Lond. A **454**, 365 (1998).

[Knill98b] Knill, E., Chuang, I. y Laflamme, R., Phys. Rev. A **57**, 3348 (1998).

[Knill00] Knill, E., Laflamme, R., Martinez, R. y Tseng, C.H., Nature, **404**, 368 (2000).

[Knill01] Knill, E., Laflamme, R., Martinez, R. y Negrevergne, C., quant-ph/0101034.

[Knuth81] Knuth, D.E., *The art of computer programming: seminumerica algorithms Vol 2*, Addison-Wesley, 1981.

[Kraus83] Kraus, K., *States, effects and operations*, Springer-Verlag, Berlin, 1983.

[Laflamme95] Laflamme, R., Miquel, C., Paz, J.P. y Zurek, W.H., *Perfect quantum error correction code*, Phys. Rev. Lett. **77**, 198 (1995).

[Landauer82] Landauer, R., *Uncertainty principle and minimal energy dissipation in a computer*, Int. J. Theor. Phys. **21**, pg 283 (1982).

[Lakshminarayan93] Lakshminarayan, A. y Balazs, N., *On the quantum cat and sawtooth maps: return to generic behaviour*, quant-ph/9307005.

[Leonhardt96] Leonhardt, U., Phys. Rev. Lett. **74**, 4101 (1995). Leonhardt, U., Pys. Rev. A **53**, 2998 (1996).

[Lidar98] Lidar, D.A., Chuang, I. y Whaley, K.B., *Decoherence-free subspaces for quantum computation*, Phys. Rev. Lett. **81**, 2594 (1998).

[Lidar99] Lidar, D.A., Bacon, D. y Whaley, K.B., *Concatenating decoherence free subspaces with quantum error correcting codes*, Phys. Rev. Lett. **82** 4556 (1999).

[Lloyd93] Lloyd, S., *Universal quantum simulators*, Science, **273** 1073 (1993), Lloyd, S., *Universal quantum simulators: correction*, Science, **279** 1113 (1998).

[Lutterbacj97] Lutterbach, L.G. y Davidovich, L., Phys. Rev. Lett. **78** (1997) 2547; Optics Express **3** (1998) 147.

[MacWilliams77] MacWilliams, F.J. y Sloane, N.J., *The theory of error-correcting codes*, North-Holland Publishing Company, New-York, 1977.

[Miquel96] Miquel, C., Paz, J.P. y Perazzo, R., *Factoring in a dissipative quantum computer*, Phys. Rev. A, **54** 2605 (1996).

[Miquel97] Miquel, C., Paz, J.P. y Zurek, W., *Quantum computation with phase drift errors*, Phys. Rev. Lett., **78**, 3971 (1997).

[Miquel02a] Miquel, C., Paz, J.P. y Saraceno, M., *Quantum computers in phase space*, Phys. Rev. A **65**, 062309 (2002). También en quant-ph/0204149.

[Miquel02b] Miquel, C., Paz, J.P., Saraceno, M., Knill, E., Laflamme, R. y Negrevergne, C., *Interpretation of tomography and spectroscopy as dual forms of quantum computation*, Nature **418**, 59 (2002). También en quant-ph/0109072.

[Nielsen98] Nielsen, M.A., Caves, C.M., Schumacher, B. y Barnum, H., *Information-theoretic approach to quantum error correction and reversible measurement*, Proc. R. Soc. Lond. A **454**, 261 (1998).

[Nielsen98] Nielsen, M.A., Knill, E., y Laflamme, R., Nature, **396**, 52 (1998).

[Nagues00] Nogues, G. et al, Phys. Rev. **A** (2000); ver además Maitre, X. et al, Phys. Rev. Lett. **79** (1997) 769.

[Nielsen00] Nielsen, M.A., y Chuang, I.L., *Quantum computation and quantum information*, Cambridge University Press, 2000.

[Obenlan98] Obenland, K., y Despain, A., *Simulating the effect of decoherence and inaccuracies on a quantum computer*, `quant-ph/9804038`. Obenland, K. y Despain, A., *Models to reduce the complexity of simulating a quantum computer*, `quant-ph/9712004`. Obenland, K. y Despain, A., *A parallel quantum computer simulator*, `quant-ph/9804039`.

[Paz98] Paz, J.P. y Zurek, W.H., *Continuous error correction*, Proc. R. Soc. Lond. A **454**, 355 (1998).

[Preskill97] Preskill, J., *Fault-tolerant quantum computation*, `quant-ph/9712048`.

[Preskill98] Preskill, J., *Reliable quantum computers*, Proc. R. Soc. Lond. A **454**, 385 (1998).

[Pringe97] Pringe, H., *Códigos cuánticos de corrección de errores*, Tesis de Licenciatura, Departamento de Física, FCEyN, UBA, 1997.

[Rivest78] Rivest, R.L., Shamir, A., Adleman, L., *A method of obtaining digital signatures and public-key cryprosystems*, Comm. ACM **21**, pg 120 (1978).

[RSA02] *The RSA factoring challenge FAQ*, `http://www.rsasecurity.com/rsalabs/ /challenges/factoring/faq.html`.

[Schak98] Schack, R. *Using a quantum computer to investigate quantum chaos*, Phys. Rev. A **57**, 1634 (1998). (tambien en `quant-ph/9705016`).

[Shor94] Shor, P.W., *Polynomial-time algorithm for prime factorization and discrete logarithms on a quantum computer*, Proceedings of the 35th Annual Symposium on Fundamentals of Computer Science, Los Alamitos, CA, IEEE Press 124 (1994).

[Shor95] Shor, P.W., *Scheme for reducing decoherence in quantum memory*, Phys. Rev. A **52**, 2493 (1995).

[Shor96a] Shor, P.W., *Fault-tolerant quantum computation*, Proc. Symp. on the Foundations of Computer Science. Los Alamitos, CA: IEEE Press (tambien en `quant-ph/9605011`).

[Silverman02] Silverman, R.D., *A cost-based security analysis of symmetric and asymmetric key lengths*, `http://www.rsasecurity.com/rsalabs/ /bulletins/bulletin13.html`.

[Somaroo99] Somaroo, S., Tseng, C.H., Havel, T.F., Laflamme, R. y Cory, D.G., *Quantum simulations on a quantum computer*, Phys. Rev. Lett. **82**, 5381, (1999).

[Steane96a] Steane, A.M., *Error correcting codes in quantum theory*, Phys. Rev. Lett., **77**, 793 (1996).

[Steane96b] Steane, A.M., *Multiple particle interference and quantum error correcion*, Proc. R. Soc. Lond. A **452**, 2551 (1996).

[Takami01] Takami, A., Hashimoto, T., Horibe, M. y Hayashi, A.,`http://xxx.lanl.gov`, hep-lat/0010002 (2001).

[Turing37] Turing, A.M., *On computable numbers, with an application to the Entscheidungsproblem*, Proc. Lond. Math Soc., **45**, pg 161, (1937).

[Vandersypen00] Vandersypen, L.M., Steffen, M., Breyta, G., Yannoni, C.S., Cleve, R. y Chuang, I.L., Phys. Rev. Lett., **85**, 5452 (2000).

[Vandersypen01] Vandersypen, L.M., Steffen, M., Breyta, G., Yannoni, C.S., Sherwood, M.H. y Chuang, I.L., Nature **414**, 883 (2001).

[VandeVen95] Van de Ven, F.J., *Multidimensional NMR in liquids*, VCH Publishers Inc, 1995.

[Wigner84] Hillery, M., O'Connell, R.F., Scully, M.O. y Wigner, E.P., Phys. Rep. **106** (1984), 121.

[Wooters82] Wooters, W.K. y Zurek, W.H., Nature **299**, 802 (1982).

[Wooters87] Wooters, W.K., Ann. Phys. NY **176** (1987).

[Zalka98] Zalka, C., *Simulating quantum systems on a quantum computer*, Proc. R. Soc. Lond. A **454**, 319 (1998).

[Zanardi97] Zanardi, P. y Rasetti, M., Mod. Phys. Lett. B, **11**, 1085 (1997).

[Zurek84] Zurek, W.H., *Reversibility and stability of information processing systems*, Phys. Rev. Lett. **53**, pg. 391 (1984)