TRABAJO DE FIN DE GRADO

Control óptimo para diseño de circuitos de computación cuántica: Implementación del algoritmo de Krotov

---

Autor:
Martín Gros Breto

Director:
Alberto Castro Barrigón

UNIVERSIDAD DE ZARAGOZA

FACULTAD DE CIENCIAS

2021

Resumen

En este trabajo se han utilizado técnicas computacionales para el diseño de pulsos electromagnéticos capaces de generar puertas lógicas cuánticas en un modelo de molécula magnética candidata para actuar como un sistema de qubits. El núcleo magnético de la molécula utilizada (GdW_{30}) consiste en un sistema de 8 niveles, equivalente a un sistema de 3 qubits acoplados. Para diseñar los pulsos, hemos utilizado el marco teórico de la *teoría de control óptimo cuántico* (QOCT). Más concretamente, hemos implementado el algoritmo de *Krotov*, uno de los algoritmos clásicos en teoría de control. Usando este método, hemos generado pulsos capaces de inducir transiciones entre estados vecinos de este sistema, así como operadores evolución equivalentes a las puertas lógicas cuánticas de Toffoli y Hadamard, con unos tiempos de duración dos órdenes de magnitud menores que el tiempo de coherencia del sistema.

Índice

1. Introducción y objetivos del trabajo

La computación cuántica consiste en la manipulación de las unidades mínimas de información cuántica, los qubits, mediante transformaciones determinadas por las puertas lógicas cuánticas [1]. Ciertos algoritmos que en computación clásica tardan un tiempo que crece exponencialmente con el tamaño del problema pueden ejecutarse, teóricamente, en un tiempo que crece de forma polinomial utilizando computación cuántica (por ejemplo el problema en el que se basa la encriptación). Uno de los principales retos para el desarrollo tecnológico de la computación cuántica es la fragilidad de los estados cuánticos. Los sistemas no están aislados, sino que interactúan con el entorno recibiendo ruido. Es por esto que se busca que el tiempo de las transformaciones sobre los qubits sea lo más breve posible.

Hay numerosos sistemas físicos sobre los que pueden construirse sistemas de qubits. Los nanoimanes moleculares son moléculas sintetizadas artificialmente que contienen un núcleo magnético formado por uno o varios iones, rodeados de ligandos no magnéticos, y típicamente insertas en sólidos o superficies [2]. Pueden ser útiles como *hardware* cuántico debido a su estructura de niveles de spin. Dentro de este grupo, nos centraremos en concreto en el GdW_{30} [3], un compuesto con características especialmente interesantes para la computación cuántica.

Para poder producir transiciones entre los niveles de spin, y con ellas crear las puertas cuánticas con las que manejar la información, es necesario producir pulsos magnéticos que alteren y conecten estos estados. El diseño de estos pulsos debe ser tal que su tiempo de duración sea muy pequeño, para evitar el problema de la fragilidad de los estados. Generalmente, las transiciones y las puertas se crean mediante concatenaciones de pulsos monocromáticos resonantes. Una propuesta alternativa es usar la teoría de control óptimo cuántico (QOCT) [4] para diseñar pulsos más complejos que puedan inducir las puertas de manera más rápida. Este marco teórico nos permite obtener una serie de ecuaciones cuya solución nos da pulsos que se adaptan de una manera óptima al problema y a las condiciones que les exijamos. La resolución de estas ecuaciones no es trivial. Uno de los algoritmos clásicos que aborda este problema es el algoritmo de Krotov [5].

En este trabajo se ha implementado este algoritmo, y se ha demostrado su efectividad utilizándolo para la optimización de pulsos magnéticos aplicados sobre un modelo de GdW_{30}. Hemos escrito el código necesario para la realización de este trabajo añadiéndolo al paquete "qocttools"[6], un paquete escrito en python que permite realizar cálculos de control óptimo cuántico sobre sistemas modelo. El código es software libre (GPL) y por lo tanto queda a disposición de cualquier usuario interesado para su uso o desarrollo.

Los objetivos principales de este trabajo han sido:

1. Entender los fundamentos de la computación cuántica, así como la comprensión de las puertas puertas lógicas cuánticas y su realización como operadores de evolución.

2. Estudiar y comprender los fundamentos del marco de la teoría de control óptimo cuántico, así como del algoritmo de Krotov.

3. Implementar un código que realice el algoritmo de Krotov.

4. Ejecutar los cálculos de optimización para implementación de puertas lógicas en un sistema de 3 qubits.

En la sección 2 se hace una breve introducción a la computación cuántica. En la sección 3 proseguimos con una breve descripción del sistema que estamos modelizando. Ya en la sección 4 se presenta el marco de la QOCT así como el algoritmo de Krotov que hemos implementado. Tras esto, en las secciones 5 y 6 presentamos los resultados obtenidos: optimizaciones de transiciones entre estados, y diseño de las puertas de Toffoli y Hadamard usando el algoritmo de Krotov. Finalmente en la sección 7 están las conclusiones del trabajo.

2. Introducción a la computación cuántica

Vamos a comenzar con una introducción al marco teórico de la computación cuántica, cuyo estudio ha sido necesario para realizar el trabajo. Este será un breve resumen, por lo que en el caso de buscar una introducción más completa y extensa, recomendamos consultar, por ejemplo, [1].

2.1. Bits cuánticos

La computación clásica consiste en la manipulación de una unidad de mínima información, el bit, que puede tomar dos valores (0/1). La computación cuántica es análoga, usando en este caso como unidad mínima de información cuántica el qubit (*quantum bit*). Un qubit puede visualizarse como un sistema cuántico cuya medida también únicamente nos puede devolver 2 posibles valores. Algunos ejemplos de sistemas que pueden considerarse como qubits pueden ser el spin de un electrón en el eje Z, o la polarización dextrógira o levógira de un fotón.

Podemos denotar a cada uno de estos 2 estados como $|0\rangle$ y $|1\rangle$, de manera análoga a los valores de los bits de la computación clásica. Ahora bien, la diferencia que tenemos con respecto a esta es que, al tratarse los qubits

de objetos cuánticos, poseen la propiedad de superposición de estados. De esta forma, entre medidas, el qubit puede estar en todos los estados

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle\ , \tag{2.1}$$

donde $\alpha, \beta \in \mathbb{C}$ y $|\alpha|^2 + |\beta|^2 = 1$. Es decir, puede estar en un estado (normalizado) cualquiera del espacio de Hilbert de dos dimensiones. Nótese que, de esta manera, mientras los bits son sistemas discretos que únicamente pueden tener 2 valores, los qubits pertenecen a un sistema contínuo, hasta que los medimos, por lo que pueden contener mucha más información.

Debido a este comportamiento cuántico, mientras que en computación clásica podemos estar constantemente leyendo y comparando bits entre sí, en computación cuántica tendremos que considerar el *problema de la medida*. Este nos dice que al medir un qubit lo estamos alterando, haciendo que su función de onda colapse en uno de los 2 estados posibles, con una probabilidad determinada por los coeficientes $|\alpha|^2$ y $|\beta|^2$.

En el caso de tener más de un qubit, debemos de nuevo aplicar las leyes de la mecánica cuántica, que nos dice que para sistemas compuestos por varios estados, el sistema total viene definido por el producto tensorial de cada uno de los subsistemas. Por lo tanto, una base del espacio compuesto viene dada por todos los estados $|a_N\rangle_N \otimes |a_{N-1}\rangle_{N-1} \otimes \cdots \otimes |a_1\rangle_1$, donde cada $a_j = 0, 1$. Una forma compacta de poder describir estos estados con varios qubits es la base computacional, en la que cada uno de los estados se describe como

$$|a_N\rangle_N \otimes |a_{N-1}\rangle_{N-1} \otimes \cdots \otimes |a_1\rangle_1 \equiv |a_N a_{N-1} \cdots a_1\rangle \equiv |c\rangle$$

donde $c = a_N 2^{N-1} + a_{N-1} 2^{N-2} \cdots + a_1$. De esta manera, la base computacional se expresa como $\{|0\rangle, |1\rangle, \cdots, |2^N - 1\rangle\}$.

Esta definición de los estados producto de varios subsistemas implica que, en cierto modo, el espacio compuesto es "más grande" que la suma de los espacios iniciales. Es decir, contiene más estados, en concreto los llamados "estados entrelazados". Una de las propiedades más interesantes que tienen estos estados es que ciertas acciones sobre alguno de los qubits, como por ejemplo la medida, provocan un cambio sobre el resto. Podemos ejemplificar este hecho mediante el estado de 2 qubits definido por

$$|\psi\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |3\rangle) = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$

Si hacemos una medida sobre el primer qubit y definimos su valor, automáticamente sabremos también el valor del otro qubit. Es decir, que aunque no hayamos medido directamente el segundo qubit, hemos alterado su valor midiendo el primero.

2.2. Puertas lógicas cuánticas

Una vez establecidos cuales son los *estados* o unidades de información de computación cuántica, es preciso entender cómo se opera sobre estos estados, para lo cual vamos a introducir el concepto de puerta lógica cuántica.

Haremos de nuevo una analogía con la computación clásica, en la que podemos hacer ciertas transformaciones a nuestros bits, definidas por las *puertas lógicas clásicas:* funciones que, dependiendo de los valores de los bits de entrada (el *input*), generan unos ciertos valores de salida (el *output*). Las puertas lógicas cuánticas operan de manera similar, pero de nuevo tendremos que estar sujetos a la condición de que los qubits son objetos cuánticos, que en general pueden estar en superposición.

Podemos describir estas puertas lógicas de forma matemática como matrices en el espacio 2^N dimensional que describe al sistema total. Al usar matrices es muy importante escoger una base inteligente en la que trabajar. La base computacional es especialmente útil en este caso, debido a que nos ayuda a entender de una forma intuitiva las transformaciones de esta puertas lógicas.

Comencemos con un ejemplo sencillo: supongamos que queremos transformar un solo qubit, de manera que si está en estado $|0\rangle$ pase a $|1\rangle$ y viceversa. Como en el caso general el estado será de la forma de (2.1), la acción sobre un estado espacio arbitrario se deduce de la acción sobre los estados de la base y aplicando linealidad:

$$|\psi\rangle_{\text{ini}} = \alpha\,|0\rangle + \beta\,|1\rangle \qquad \rightarrow \qquad |\psi\rangle_{\text{fin}} = \alpha\,|1\rangle + \beta\,|0\rangle \tag{2.2}$$

Lo que es equivalente a aplicar al sistema la siguiente matriz 2×2

$$U_{\text{NOT}} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \tag{2.3}$$

Esta puerta lógica sería el equivalente a la puerta NOT clásica, que invierte el valor del bit de entrada, pero en el caso cuántico puede trabajar también con superposiciones de estados. En general, una puerta lógica operando sobre un qubit puede representarse mediante una matriz unitaria ($U^\dagger U = \mathbb{I}$).

Las puertas lógicas pueden definirse sobre sistemas con un número arbitrario de qubits. Imaginemos, por ejemplo que tenemos un sistema de dos qubits y queremos cambiar el estado del segundo qubit como antes, pero solamente si el valor de primero es $|1\rangle$. Esta es la llamada puerta CNOT (*controlled NOT*), de la cual se puede deducir su representación matricial

viendo como cambian los estados de la base computacional:

$$\begin{aligned}
|0\rangle = |00\rangle &\longrightarrow |00\rangle = |0\rangle \\
|1\rangle = |01\rangle &\longrightarrow |01\rangle = |1\rangle \\
|2\rangle = |10\rangle &\longrightarrow |11\rangle = |3\rangle \\
|3\rangle = |11\rangle &\longrightarrow |10\rangle = |2\rangle
\end{aligned} \qquad U_{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

Se puede extrapolar esta forma de trabajo para matrices que actúen sobre un número de qubits arbitrario.

Construir "tablas de la verdad" como las anteriores (i.e. definiendo cómo actúa la puerta sobre cada uno de los estados de la base computacional) no es la única forma de obtener la representación matricial de una puerta. En el caso de que la puerta se componga de actuaciones independientes sobre cada uno de los qubits, podemos utilizar el producto tensorial. Por ejemplo, supongamos una puerta que aplica NOT sobre el segundo qubit, independientemente del estado del primero, y no hace nada en el primero. En ese caso:

$$U = \mathbb{I}_1 \otimes U_{NOT\,2} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \otimes \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \tag{2.4}$$

Ahora que sabemos crear puertas lógicas para uno o varios qubits, podemos definir los circuitos cuánticos, que no serán otra cosa que la aplicación de una sucesión de puertas cuánticas sobre el sistema. Matemáticamente, la aplicación de estas puertas lógicas se hace mediante el producto matricial (siempre respetando el orden de las operaciones debido a la no conmutatividad de este proceso).

Podemos concluir por lo tanto que se puede representar un circuito entero mediante una matriz, y que existen diferentes combinaciones de puertas que pueden construir un mismo circuito, es decir, que nos llevan de los mismos *input* a los mismos *output*. Este hecho sugiere la pregunta de si existen conjuntos de puertas *universales*, esto es, un conjunto reducido de puertas que permita representar todas las transformaciones posibles, con un cierto grado de exactitud. Esto no es simplemente una curiosidad teórica, sino que a nivel experimental, ciertas puertas son más sencillas de conseguir que otras, o tienen unas características que las hacen mejores (por ejemplo menores tiempos de ejecución), por lo que encontrar formas alternativas de construir circuitos es muy importante.

La respuesta a esta duda de la universalidad es que sí que se pueden encontrar ciertos conjuntos que cumplen esto. Es por ello que hemos utilizado la puerta de Toffoli para realizar los cálculos en este trabajo: si bien no es

universal por si sola en computación cuántica,[1] unida a alguna puerta extra como la de Hadamard, forma un conjunto universal [7]. Otra de las razones por la que nos interesa esta puerta es que es una puerta que actúa sobre 3 qubits, que como veremos, es el caso del sistema con el que vamos a trabajar.

2.3. Puertas lógicas como operadores de evolución. Motivación del uso de QOCT

A un nivel menos abstracto y más físico, las puertas lógicas cuánticas son en esencia operadores de evolución temporal $U(T, 0)$, es decir, operan mediante la acción de un Hamiltoniano que genera un cambio en el sistema en el intervalo temporal $t \in (0, T)$, donde T es el tiempo de operación, que puede variar de una puerta a otra. Es por esto por lo que toda puerta lógica cuántica debe cumplir la propiedad de ser unitaria, $U^\dagger U = \mathbb{I}$, lo cual implica entre otras cosas reversibilidad temporal.

En principio, uno puede tratar de diseñar una puerta cuántica buscando el Hamiltoniano cuya ecuación de Schrödinger asociada induzca la unitaria que define la puerta. Ahora bien, en el mundo real no es tan sencillo, ya que raramente podremos tener sistemas aislados, por lo que el ruido acaba afectando a estos, y pierden ciertas propiedades como la *coherencia*. Esta perdida de la coherencia cuántica es un problema grave, ya que la implementación de sistemas de información cuántica necesita que los estados, y por lo tanto la información que contienen, se mantenga inalterada. La decoherencia inducida por el ruido externo es el mayor problema que impide el desarrollo de esta tecnología. Desde el punto de vista teórico, también es un inconveniente, ya que supone que la ecuación de Schrödinger deje de ser válida para describir el sistema, y tendremos que usar otras ecuaciones válidas para sistemas abiertos, como por ejemplo la ecuación de Lindblad, cuyo tratamiento es mucho más complejo.

Por este motivo, nos interesa que los tiempos de actuación T de las puertas sean lo más cortos posibles, para que el exterior afecte lo menos posible. Esta es la motivación para usar la *teoría de control óptimo cuántico* (QOCT): tratar de encontrar campos externos que generen estas transformaciones en tiempos suficientemente pequeños para permanecer en el rango de validez de la ecuación de Schrödinger.

3. Descripción del sistema

La motivación para este trabajo surge del proyecto SUMO (*Scaling Up quantum computation with MOlecular spins*), dirigido por el investigador F.

[1] Se puede demostrar que en computación clásica sí que es universal.

Luis en el Instituto de Ciencia de Materiales de Aragón (ICMA). En este proyecto se desarrollan nanoimanes moleculares para investigar su posible uso como sistemas de qubits. Nos hemos centrado en uno de ellos en concreto, que puede modelarse con 8 niveles cuánticos, lo que se puede reinterpretar como un sistema de 3 qubits.

Aunque no ha sido objetivo de este trabajo entender y estudiar este sistema, procedemos a describirlo brevemente. Es este sistema con el que obtendremos los resultados numéricos en las secciones 5 y 6. Para una descripción más detallada consultar [2] y [3].

3.1. Sistema experimental

El sistema con cuyo modelo comprobaremos nuestro algoritmo es uno de los llamados *molecular nanomagnets* (nanoimanes moleculares): moléculas artificiales diseñadas y sintetizadas mediante métodos químicos. Consisten en un núcleo magnético rodeado de ligandos no magnéticos, que permanecen estables en diferentes tipos de materiales desde cristales a soluciones, e incluso cuando son depositados en substratos sólidos [2]. Su estructura de niveles de espín puede usarse como sistema sistema cuántico sobre el que se pueden realizar cálculos de computación cuántica.

Un ion que posee ciertas características que le hacen propicio para formar el núcleo magnético de la molécula es el átomo de Gadolinio triplemente ionizado, Gd^{+3}. Uno de los motivos principales de su elección es que, al contrario que otros lantánidos, la separación de sus niveles energéticos a campo nulo es suficientemente pequeña como para que todos sean accesibles experimentalmente usando generadores de microondas sencillos. Además, este ion tiene un momento angular orbital L nulo, y posee un espín $S = 7/2$, de manera que el espín electrónico forma como un sistema de 8 niveles, que puede interpretarse como un sistema de tres qubits. Es debido a estas características que resulta especialmente interesante su uso.

En el laboratorio, el ion Gd^{+3} no se encuentra aislado, sino que está inserto en una estructura bastante compleja, abreviada como GdW_{30} [3]. Su entorno cambia sus características, de modo que por ejemplo rompe degeneraciones, lo que que facilita el acceso a estados diferentes. Esta es otra de las razones para la elección de esta molécula como soporte de qubits.

Finalmente, otra de las ventajas del GdW_{30} son los tiempos de decoherencia de las transiciones entre sus niveles energéticos, que son bastante altos a temperaturas moderadas. Estos tiempos limitan las duraciones de los pulsos que queremos diseñar. Así, se ha encontrado experimentalmente [2] que los diferentes tiempos de coherencia de esta molécula son del orden de 0.5 μs.

3.2. Modelización Teórica

Para poder trabajar computacionalmente con el GdW_{30} necesitamos poder describirlo con un modelo matemático. En sistemas de este estilo, es muy conveniente usar los llamados *Hamiltonianos de espín*, debido a que pueden describir la parte relevante del sistema con gran precisión. El Hamiltoniano que describe el GdW_{30} (ver [8] y [9]) es:

$$\hat{H} = D\left[\hat{S}_z^2 - \frac{1}{3}S(S+1)\right] + E\left(\hat{S}_x^2 - \hat{S}_y^2\right) - g\mu_B\hat{\vec{S}}\cdot\vec{H}\,, \tag{3.1}$$

donde D y E son los llamados términos anisotrópicos de segundo orden, cuyos valores, medidos experimentalmente son:

$$D = 1281 \text{ MHz} \qquad\qquad E = 294 \text{ MHz}\,.$$

Además, $g = 2$ es la razón giromagnética, μ_B es el magnetón de Bohr, $\vec{H}$ es un campo magnético estático externo al que puede estar sometido el sistema, y $\hat{\vec{S}}$ es el operador vector de spin (referido a los ejes anisotrópicos del cristal). Por último, recordemos que para este sistema, $S = 7/2$.

El Hamiltoniano anterior es estático, y no puede utilizarse sin más para generar operaciones; debemos añadirle una perturbación dependiente del tiempo cuya forma temporal será precisamente la que vamos a diseñar computacionalmente. En el montaje experimental que estamos describiendo, las acciones sobre el sistema se efectúan mediante campos magnéticos en el rango de las microondas. Por lo tanto, el Hamiltoniano completo es:

$$\hat{H}(t) = \hat{H}_0 + f(t)\hat{V} \tag{3.2}$$

donde $\hat{H}_0$ vendrá descrito por 3.1 y

$$\hat{V} = -g\mu_B\hat{\vec{S}}\cdot\vec{H}_m \tag{3.3}$$

Notar que esta es una aproximación que solo funciona si la perturbación magnética introducida no es muy grande.

4. Metodología

Tras presentar el modelo del sistema, describiremos en esta sección el marco teórico que nos permitirá calcular los pulsos óptimos.

La forma habitual utilizada para generar transiciones entre estados y puertas cuánticas consiste en usar los llamados *pulsos-π*. No obstante, los tiempos de aplicación de estos pueden ser demasiado grandes, lo que implica el riesgo de pérdida de la coherencia cuántica antes de poder hacer un número relevante

de operaciones con ellos [10]. Es por esto que la QOCT aparece como una alternativa para calcular pulsos con tiempos de duración mucho menores.

Presentamos a continuación un breve resumen de esta teoría. Después, vamos a describir uno de los algoritmos clásicos de optimización en OCT, el algoritmo de Krotov [5], cuya posterior implementación en código es uno de los principales objetivos de este trabajo.

4.1. Introducción a la teoría de control óptimo

La teoría de control óptimo estudia el problema de encontrar las soluciones que maximicen o minimicen un funcional definido sobre un sistema dinámico, sometido a una perturbación que actúa como *control*.

Supongamos la siguiente ecuación diferencial que describe la dinámica de un proceso genérico:

$$\dot{y}(t) = f(y(t), u, t) \tag{4.1a}$$
$$y(0) = y_0 \tag{4.1b}$$

donde $u \equiv \{u_i\}$ es un conjunto de parámetros que introducimos en el sistema, los parámetros de control.

Definimos ahora el siguiente funcional:

$$F(y, u) = F^{\text{term}}(y(T), u) + \int_0^T dt F^{\text{td}}(y(t), u)\,. \tag{4.2}$$

El funcional "terminal", F^{term}, depende únicamente del estado del sistema en el instante final T, mientras que el funcional "time-dependent" F^{td} depende de toda la trayectoria. La definición concreta de F depende del problema concreto: hay que diseñarla de forma que tome valores altos si el comportamiento del sistema es el deseado, y bajos en caso contrario. El objetivo entonces será encontrar el máximo de este funcional.

Dado que la trayectoria del sistema está determinada por la elección de los parámetros de control, podemos definir una función

$$G(u) := F(y[u], u)\,, \tag{4.3}$$

donde llamamos $y[u](t)$ a la solución de la trayectoria inducida por los parámetros u. Notar que G ya solo depende de u, debido a que estamos considerando la trayectoria específica determinada por estos parámetros de control.

El objetivo es, por lo tanto, maximizar esta función G. Hay muchos algoritmos disponibles para encontrar máximos de funciones; los más eficientes son aquellos que utilizan el gradiente de la función. En este caso, puede derivarse una expresión para el gradiente de esta función usando optimización condicionada con multiplicadores de Lagrange: es posible demostrar [4] que

$$\begin{aligned}\frac{\partial G}{\partial u_i}(u) = &\int_0^T dt\ \frac{\partial F^{td}}{\partial u_i}(y(t),u)\bigg|_{y(t)=y[u](t)} + \frac{\partial F^{term}}{\partial u_i}(y(T),u)\bigg|_{y(T)=y[u](T)} \\ &+ \sum_m \int_0^T dt\ \lambda_m[u](t)\frac{\partial f_m}{\partial u_i}(y(t),u)\bigg|_{y(t)=y[u](t)}\end{aligned} \tag{4.4}$$

donde $\lambda_m[u]$ es el llamado *coestado*, el multiplicador de Lagrange que se obtiene resolviendo las siguientes ecuaciones dinámicas:

$$\frac{d}{dt}\lambda_m[u](t) = -\sum_n \lambda_n(t)\frac{\partial f_n}{\partial y_m}(y[u](t),u,t) - \frac{\partial F^{td}}{\partial y_m}(y[u](t),u) \tag{4.5a}$$

$$\lambda_m[u](T) = \frac{\partial F^{term}}{\partial y_m}(y[u](T),u) \tag{4.5b}$$

Esta sería una ecuación bastante general válida para multitud de sistemas dinámicos. Puede aplicarse al caso de un sistema cuántico. En ese caso, lo que requerimos es que la ecuación dinámica, 4.1a, sea una ecuación perteneciente al formalismo cuántico, como por ejemplo la ecuación de Schrödinger o la de Lindblad. Así, vamos a explorar a continuación los dos casos que hemos implementado en código: la ecuación de Schrödinger para funciones de onda y para operadores evolución.

4.1.1. QOCT para funciones de onda

En este caso, la ecuación dinámica se reduce al caso particular

$$\frac{d}{dt}\left|\psi[u](t)\right\rangle = -i\hat{H}(u,t)\left|\psi[u](t)\right\rangle \tag{4.6a}$$

$$\left|\psi[u](0)\right\rangle = \left|\psi_0\right\rangle \tag{4.6b}$$

donde $|\psi(t)\rangle$ es la función de onda del sistema cuántico. La notación $\psi[u](t)$ enfatiza el hecho de que la especificación de un conjunto concreto de parámetros u determina la evolución del sistema.

Vamos a realizar varias asunciones comunes para simplificar el problema. La primera es que podemos expresar el hamiltoniano del sistema $\hat{H}(u,t)$ de manera perturbativa como

$$\hat{H}(u,t) = \hat{H}_0 + f(u,t)\hat{V} \tag{4.7}$$

donde en nuestro caso $\hat{H}_0$ viene descrito por 3.1, $f(u,t)$ es la perturbación (o pulso) dependiente del tiempo, y $\hat{V}$ viene descrito por 3.3.

En segundo lugar, consideraremos que el funcional a optimizar no posee parte dependiente del tiempo, i.e. depende únicamente del estado final. Frecuentemente, el funcional a optimizar será de la forma:

$$F^{term}(\psi(T),u) = \langle\psi(T)|\, O\, |\psi(T)\rangle + P(u) \tag{4.8}$$

donde O es un operador cuyo valor esperado queremos optimizar, y $P(u)$ es la llamada *función de penalty*, una función que puede diseñarse para penalizar pulsos con propiedades que no nos interesen, como por ejemplo tener demasiada amplitud.

La expresión 4.4 nos queda entonces de la siguiente forma:

$$\frac{\partial G}{\partial u_i}(u) = \frac{\partial P}{\partial u_i}(u) + 2\,\mathrm{Im}\int_0^T dt \frac{\partial f}{\partial u_i}(u,t)\left\langle \chi^\dagger[u](t)\left|\hat{V}\right|\psi[u](t)\right\rangle . \tag{4.9}$$

y $\chi[u]$ está definido por las siguientes ecuaciones dinámicas

$$\frac{d}{dt}|\chi[u](t)\rangle = -iH(u,t)\,|\chi[u](t)\rangle \tag{4.10a}$$

$$|\chi[u](T)\rangle = O\,|\psi[u](T)\rangle\ . \tag{4.10b}$$

Debemos notar que en 4.10b no tenemos una condición inicial, sino una condición final. Esto es relevante a la hora de resolver esta ecuaciones computacionalmente, pues tendremos que hacerlo *hacia atrás*.

En los cálculos que mostraremos a continuación, uno de los objetivos ha sido el diseño de pulsos que inducen la transición del sistema desde su estado inicial hacia un estado target $|\Psi_{\text{target}}\rangle$. En ese caso, generalmente se elige como operador O a la proyección sobre ese estado:

$$O = |\Psi_{\text{target}}\rangle\langle\Psi_{\text{target}}|\,. \tag{4.11}$$

4.1.2. QOCT para operadores evolución

Sabemos que podemos describir la evolución unitaria de un sistema mediante un operador de evolución temporal: $|\psi(t)\rangle = U(t,t')\,|\psi(t')\rangle$. En lo sucesivo, usaremos por conveniencia $U(t) := U(t,0)$.

Puede entonces reescribirse la ecuación de Schrödinger para el propagador:

$$\frac{d}{dt}U(t) = -i\hat{H}(u,t)U(t) \tag{4.12a}$$

$$U(0) = \mathbb{I} \tag{4.12b}$$

Este formalismo es especialmente interesante para nuestro caso, ya que buscaremos que $U[u](t)$ sea equivalente a un cierto operador U_{target}. Una manera de definir el funcional para conseguir este tipo de optimización es:

$$F^{\text{term}}(U(T), u) = \left| \frac{1}{d} \operatorname{Tr}\left[U^{\dagger}(T) \cdot U_{target} \right] \right|^2 + P(u) \tag{4.13}$$

Hemos utilizado el producto de Frobenius entre operadores

$$(A, B) = \frac{1}{d} \operatorname{Tr}\left[A^{\dagger} \cdot B \right] , \tag{4.14}$$

donde d es la dimensión del espacio de Hilbert. Este producto posee la característica de que para operadores unitarios no puede tomar valores mayores que 1 (en valor absoluto), y valdrá uno solo si los dos operadores son equivalentes (i.e. se diferencian únicamente en un factor de fase global).

Definiendo, como antes, $G(u) = F(U[u](T), u)$, la expresión 4.4 nos queda:

$$\frac{\partial G}{\partial u_i}(u) = \frac{\partial F^{\text{term}}}{\partial u_i}(U(T), u)\bigg|_{U(T)=U[u](T)} + 2 \operatorname{Im} \int_0^T dt \operatorname{Tr}\left[B^{\dagger}[u](t) \frac{\partial \hat{H}}{\partial u_i}(u, t) U[u](t) \right] \tag{4.15}$$

donde $B[u](t)$ es el operador evolución coestado, el cual está determinado por las siguientes ecuaciones dinámicas

$$\frac{d}{dt} B[u](t) = -i \hat{H}(u, t) B[u](t) \tag{4.16a}$$

$$B[u](T) = \frac{\partial F^{\text{term}}}{\partial U^*} \left(U[u](T), u \right) . \tag{4.16b}$$

De nuevo, recalcamos que 4.16b es una condición final del operador evolución coestado.

4.2. Función de penalización

Hemos visto que en las expresiones de la QOCT puede introducirse una función de penalización que será muy importante a la hora de hacer los cálculos, ya que con ella podemos imponer restricciones a los pulsos que queremos obtener. Hay muchas formas de crear funciones de este estilo en función del tipo de pulso que queramos obtener, y las limitaciones que tengamos en nuestro sistema o instrumental. Para nuestros cálculos hemos utilizado:

$$P(u) = -\alpha \int_0^T dt \frac{u^2(t)}{S(t)} \tag{4.17}$$

En esta función $\alpha > 0$ y regula la fuerza de la penalización al valor de G, lo que nos permite crear pulsos más o menos restrictivos. Por otro lado,

el numerador de la integral castiga el crear perturbaciones con amplitudes excesivamente altas. Esto es muy importante, ya que como se ha dicho antes, el valor máximo de la amplitud de los pulsos que se puede generar en la práctica está acotado, por lo que tendremos que restringirnos al rango alcanzable por nuestros instrumentos.

Por último, la función $S(t)$ en el denominador será una cuyo valor en el centro del intervalo temporal sea 1, pero en los extremos $t \to 0$ y $t \to T$ tienda a 0. Con esto se fuerza a que la amplitud del pulso en los instantes iniciales y finales sea nula, y se aproxime a cero de manera suave, tal y como lógicamente ocurre en un experimento. Hay varias opciones para definir $S(t)$; nosotros hemos escogido una de las más típicas:

$$S(t) = \text{erf}\left(r\frac{t}{T}\right)\text{erf}\left(r\frac{T-t}{T}\right) \tag{4.18}$$

donde erf es la función error de Euler. Notar que se puede seleccionar r para obtener transiciones más o menos suaves.

De esta manera, alterando los valores de α y r, podremos adaptar el pulso a nuestras preferencias o necesidades experimentales, buscando el mejor pulso posible para cada caso particular.

4.3. El algoritmo de Krotov

En las secciones 4.1.1 y 4.1.2 hemos obtenido una serie de ecuaciones, las cuales, en general, no son resolubles analíticamente. Es por esto que requerimos de métodos numéricos para ser capaces de obtener las soluciones de estas, y por tanto ser capaces de resolver el problema que nos interesa, el cálculo de pulsos óptimos. Un problema de los métodos numéricos es que muchas veces requieren de muchos recursos si queremos obtener un resultado lo suficientemente preciso. Es por ello que en general interesa tener y buscar métodos o algoritmos que sean muy eficientes para cada problema, ya que nos puede ahorrar muchos costos computacionales.

Es aquí donde surge la importancia del algoritmo de Krotov, ya que nos permite aproximarnos rápidamente a la solución de las ecuaciones presentadas en el apartado anterior. El ingrediente esencial necesario para ejecutar este algoritmo es la propagación de las ecuaciones del movimiento – esencialmente en nuestro caso, la ecuación de Schrödinger. Este es un problema numérico bien conocido y optimizado, para el que tenemos un gran número de propagadores veloces como el *Runge-Kutta 4* [11].

El algoritmo de Krotov require de una parametrización del pulso *en tiempo*

real. Es decir, los parámetros u_i que introducíamos anteriormente son directamente los valores que toma la función f en cada instante del tiempo. Numéricamente, el intervalo de propagación $(0, T)$ debe discretizarse, de forma que utilizaremos como parámetros los valores que toma f en los tiempos discretizados t_0, t_1, ..., T. En la descripción que hacemos a continuación, sin embargo, utilizaremos la función entera que renombraremos como $u(t)$.

Este algoritmo propone una manera de aproximarnos a la solución de las ecuaciones de la QOCT mediante un proceso iterativo. Iremos obteniendo una sucesión $u^{(0)}, u^{(1)}, ..., u^{(k)}, ...$ de campos de perturbación, con la propiedad [5] de que esta sucesión es tal que el valor de G crece de manera monótona con el numero de iteración $G(u^{(k+1)}) \geq G(u^{(k)})$.

Vamos a describir de manera diferenciada los dos casos estudiados en la sección anterior: el caso en que se propagan funciones de onda y el caso en que se propagan operadores evolución.

4.3.1. Algoritmo de Krotov para evolución de estados

En este primer caso, nos tendremos que remontar a las expresiones obtenidas en la sección 4.1.1, donde recordemos que ψ hace referencia a las coordenadas del estado y χ a las del coestado.

Podemos dividir el algoritmo en 2 bloques, la inicialización, y el núcleo del algoritmo, ambos con 3 pasos internos:

Inicialización: Este bloque prepara los estados y el pulso para entrar posteriormente al núcleo del programa.

1. Se propone un pulso inicial $u^{(0)}$, que puede, o no, ser aleatorio.

2. Se resuelve la ecuación dinámica para este pulso inicial:

$$\frac{d}{dt}\left|\psi^{(0)}(t)\right\rangle = -i\left[H_0 + u^{(0)}(t)V\right]\left|\psi^{(0)}(t)\right\rangle \tag{4.19a}$$

$$\left|\psi^{(0)}(0)\right\rangle = |\psi_0\rangle \tag{4.19b}$$

3. Se resuelve la ecuación dinámica del coestado $|\chi(t)\rangle$

$$\frac{d}{dt}\left|\chi^{(0)}(t)\right\rangle = -i\left[H_0 + u^{(0)}(t)V\right]\left|\chi^{(0)}(t)\right\rangle \tag{4.20a}$$

$$\left|\chi^{(0)}(T)\right\rangle = O\left|\psi^{(0)}(T)\right\rangle \tag{4.20b}$$

Notar que estamos definiendo un coestado, tal y como hacíamos en la sección anterior para escribir la ecuación para el gradiente, con una ecuación

semejante. El algoritmo de Krotov, aunque no explícitamente, utiliza por lo tanto el gradiente de la función objetivo. Así, hemos obtenido $u^{(0)}$, $\psi^{(0)}$ y $\chi^{(0)}$, por lo que ya podemos entrar al núcleo del algoritmo.

Núcleo: En esta parte, iremos obteniendo una sucesión de pulsos, hasta llegar a la convergencia, o al máximo número de iteraciones. Comenzamos con $k = 1$, y para cada iteración aumentamos $k = k + 1$:

1. Obtendremos $\psi^{(k)}$ resolviendo la siguiente ecuación diferencial

$$\frac{d}{dt}\left|\psi^{(k)}(t)\right\rangle = -i\left[H_0 + u^{(k)}(t)V\right]\left|\psi^{(k)}(t)\right\rangle \tag{4.21a}$$

$$\left|\psi^{(k)}(0)\right\rangle = |\psi_0\rangle \tag{4.21b}$$

   donde $u^{(k)}$ está dado por la siguiente expresión:

$$u^{(k)}(t) = \frac{S(t)}{\alpha}\,\mathrm{Im}\left\langle\chi^{(k-1)}(t)\right| V \left|\psi^{(k)}(t)\right\rangle \tag{4.22}$$

   Debemos notar que $u^{(k)}$ está acoplado a $\psi^{(k)}$, por lo que en la práctica nos enfrentamos a una ecuación diferencial no lineal, la cual podemos escribir de forma explícita como:

$$\frac{d}{dt}\left|\psi(t)^{(k)}\right\rangle = -i\left[H_0 + \frac{S(t)}{\alpha}\,\mathrm{Im}\Big\{\left\langle\chi^{(k-1)}(t)\right| V \left|\psi^{(k)}(t)\right\rangle\Big\}V\right]\left|\psi^{(0)}(t)\right\rangle \tag{4.23}$$

2. Una vez calculados el nuevo estado y el nuevo pulso, es el momento en el que evaluamos si es preciso terminar el algoritmo, o si por el contrario, necesitamos continuar, para lo cual hay que obtener un nuevo coestado. Para ello, analizaremos la convergencia del algoritmo,[2] es decir, si el nuevo campo de control es lo suficientemente parecido al viejo ($u^{(k)} \approx u^{(k-1)}$), o si el valor de $G(u^{(k)})$ es lo suficientemente grande.

3. Si encontramos que necesitamos continuar el algoritmo, necesitamos calcular $\chi^{(k)}$, para lo cual resolveremos la ecuación dinámica

$$\frac{d}{dt}\left|\chi^{(k)}(t)\right\rangle = -i\left[H_0 + u^{(k)}(t)V\right]\left|\chi^{(k)}(t)\right\rangle \tag{4.24a}$$

$$\left|\chi^{(k)}(T)\right\rangle = O\left|\psi^{(k)}(T)\right\rangle \tag{4.24b}$$

4.3.2. Algoritmo de Krotov para operadores evolución

Nos remitimos ahora a las ecuaciones obtenidas en 4.1.2, de las que trataremos de hallar una solución mediante un algoritmo con la misma estructura que la que hemos explorado en la sección anterior. Recordemos que U hace referencia al operador evolución del sistema y que B hace referencia al operador evolución coestado.

[2] O si hemos llegado al número máximo de evaluaciones que buscamos.

Inicialización: Este bloque inicializa el pulso, el operador evolución y el operador coestado para posteriormente poder entrar al núcleo del algoritmo.

1. Se propone un pulso inicial $u^{(0)}$.

2. Se resuelve la ecuación dinámica del operador evolución U

$$\frac{d}{dt}U^{(0)}(t) = -i\left[H_0 + u^{(0)}V\right]U^{(0)}(t) \tag{4.25a}$$
$$U^{(0)}(0) = \mathbb{I} \tag{4.25b}$$

3. Se resuelve la ecuación dinámica del operador coestado B

$$\frac{d}{dt}B^{(0)}(t) = -i\left[H_0 + u^{(0)}V\right]B^{(0)}(t) \tag{4.26a}$$
$$B^{(0)}(T) = \frac{1}{d^2}\,\mathrm{Tr}\left[U^{(0)\dagger}(T)U_{\text{target}}\right]U_{\text{target}} \tag{4.26b}$$

Ya tenemos $u^{(0)}(t)$, $U^{(0)}(t)$ y $B^{(0)}(t)$, por lo que estamos listos para entrar al núcleo del algoritmo.

Núcleo: Al igual que en el caso anterior, iremos obteniendo una sucesión de campos de perturbación, comenzando con $k = 1$ y haciendo $k + 1$ cada vez que completemos una iteración del algoritmo.

1. Calculamos $U^{(k)}$ resolviendo la ecuación dinámica

$$\frac{d}{dt}U^{(k)}(t) = -i\left[H_0 + u^{(k)}V\right]U^{(k)}(t) \tag{4.27a}$$
$$U^{(k)}(0) = \mathbb{I} \tag{4.27b}$$

   donde $u^{(k)}$ viene dado por la siguiente expresión

$$u^{(k)} = \frac{S(t)}{\alpha}\,\mathrm{Im}\,\mathrm{Tr}\left[B^{(k-1)}(t)VU^{(k)}(t)\right] \tag{4.28}$$

   Al igual que antes, estas ecuaciones en realidad se resumen en una ecuación diferencial no lineal de primer orden:

$$\frac{d}{dt}U^{(k)}(t) = -i\left[H_0 + \frac{S(t)}{\alpha}\,\mathrm{Im}\,\mathrm{Tr}\left[B^{(k-1)}(t)VU^{(k)}(t)\right]V\right]U^{(k)}(t) \tag{4.29}$$

2. Una vez calculados el nuevo operador evolución y campo de perturbaciones, es momento de evaluar si se debe para el algoritmo o seguir una iteración más. Evaluaremos si el campo de perturbación ha convergido lo suficiente ($u^{(k)} \approx u^{(k-1)}$), o si el valor de $G(u^{(k)})$ es suficientemente grande.

3. Si continuamos el algoritmo, necesitaremos un nuevo coestado, por lo que resolveremos la siguiente ecuación dinámica para calcular $B^{(k)}$

$$\frac{d}{dt}B^{(k)}(t) = -i\left[H_0 + u^{(k)}V\right]B^{(k)}(t) \tag{4.30a}$$

$$B^{(k)}(T) = \frac{1}{d^2}\,\mathrm{Tr}\left[U^{(k)\dagger}(T)U_{target}\right]U_{target} \tag{4.30b}$$

Cuanto más iteraciones hagamos de este algoritmo, más cerca estaremos de encontrar la solución óptima que se ajuste a los valores de α, T y $u^{(0)}$ que hemos impuesto.

5. Transiciones entre estados vecinos

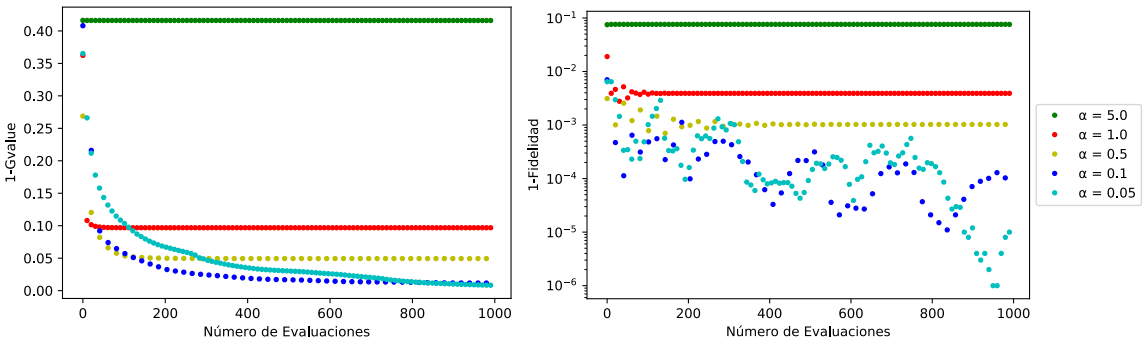
Aunque el objetivo último de este trabajo es calcular puertas lógicas cuánticas hemos empezado abordando un problema más sencillo, como es el de calcular los pulsos óptimos para provocar transiciones entre estados vecinos. Tal y como mencionábamos anteriormente, la forma habitual de resolver este problema consiste en utilizar los llamados *pulsos-*π, que son pulsos capaces de generar transiciones entre estados vecinos de forma prácticamente perfecta. El problema de estos es que generalmente el tiempo necesario para ello puede ser grande, y dependiendo del sistema puede ser mayor que el tiempo de coherencia del sistema. Así, para nuestro problema concreto, en la práctica no nos sirven (para un estudio más detallado de los tiempos de estos pulsos y sus propiedades consultar [10]). Nos podemos hacer la pregunta de si con QOCT y con el algoritmo de Krotov, podemos obtener pulsos con precisiones similares a los pulsos-π, pero en tiempos mucho menores.

Como vimos en la sección 3, el tiempo de coherencia del sistema con el que estamos trabajando es $\sim$0.5 μs, de esta manera, buscaremos pulsos cuya duración sea unos 2 ordenes de magnitud menor, de forma que seamos capaces de aplicar un numero amplio de transformaciones antes de que el estado cuántico comience a "estropearse".

En el algoritmo de Krotov, los parámetros α y T se introducen a priori y no se optimizan, por lo que conviene hacer un estudio previo para ver como afectan estos valores al pulso óptimo generado.

La función objetivo F es por lo tanto la probabilidad de que el estado final esté en el nivel que queremos. Para el caso de transiciones entre estados, si partimos del estado $|i\rangle$, y queremos transicionar al $|i+1\rangle$:

$$F(\psi(T), u) = |\,\langle i+1|\psi(T)\rangle\,|^2 + P(u)\,. \tag{5.1}$$



(a) (b)

Figura 1: (a) Convergencia del valor del funcional G con el número de iteraciones del algoritmo de Krotov para un pulso optimizado para generar la transición entre estados $|0\rangle$ y $|1\rangle$. (b) Infidelidad para el mismo cálculo.

En computación cuántica generalmente se utiliza el término *fidelidad*, que para este caso sencillo se define simplemente como:

$$\text{Fidelidad} = |\langle i+1|\psi(T)\rangle|^2 \tag{5.2}$$

La *infidelidad* es lógicamente la diferencia $1 -$ Fidelidad. Cuanto menor sea esta, mejor se comportará nuestro pulso para nuestros objetivos. Buscaremos pulsos con una infidelidad del orden de $10^{-2} - 10^{-3}$.

5.1. Estudio de la convergencia del algoritmo

Vamos a comenzar confirmando que el algoritmo esté bien implementado computacionalmente, y que funcione como esperábamos. Para ello, vamos a estudiar la propiedad mencionada anteriormente de que el valor del funcional G converge de manera monótona con el número de iteraciones del núcleo del algoritmo.

Para ello, vamos a representar el valor de la infidelidad de G, que definiremos como $1 - Gvalue$, de un pulso optimizado para inducir la transición $|0\rangle \to |1\rangle$. Presentamos en la gráfica 1a el comportamiento de esta variable frente al número de iteraciones para varios valores de α.

Podemos observar en la gráfica que en efecto este valor de la infidelidad de G decrece de manera monótona tal y como esperábamos, lo que nos confirma que el algoritmo funciona correctamente. Además, podemos ver que este comportamiento ocurre para distintos valores de α, y que además, cuanto mayor es el valor de esta, más rápido llegamos a la convergencia del algoritmo.

No obstante, no es el valor de G el que realmente nos interesa, sino la población del estado objetivo, es decir, la fidelidad. Presentamos en la gráfica 1b un estudio de la infidelidad en función del número de iteraciones del algoritmo. Podemos observar que hay una cierta convergencia con el número de evaluaciones, como cabría esperar, ya que cuantas más evaluaciones más cerca de la solución óptima estamos. Ahora bien, esta convergencia no es monótona como en el caso anterior. Esto es algo normal, ya que recordemos que el algoritmo maximiza el valor de una función que tiene en cuenta una función de penalización, y está diseñado para ser monótono únicamente en la función G completa.

También podemos observar la dependencia de la infidelidad con α. Cuanto mayor es α, más complicado es obtener una infidelidad baja, ya que más peso tiene la función penalización, lo que impide explorar soluciones que nos darían mucha más fidelidad. La decisión sobre qué pulso elegir, de entre todos los obtenidos para diversos valores de α para aplicarlo en la práctica debe establecer un compromiso entre la fidelidad buscada, y el hecho de que un valor alto de α probablemente implique un pulso solución con amplitudes grandes, quizá no realizable experimentalmente.

5.2. Estudio de la influencia de la duración del pulso

Vamos ahora a estudiar el efecto del tiempo de aplicación del pulso T, cómo afecta a la fidelidad de las soluciones encontradas. Para hacer este estudio, para cada valor de α, hemos seguido el siguiente protocolo: comenzamos, para el valor de T más bajo, con un pulso aleatorio inicial, y hacemos la optimización. Tras esto, aumentamos T a $T + \Delta t$ y, usando como pulso inicial el óptimo obtenido en el tiempo T (unido a un pulso nulo para rellenar el tiempo restante entre T y $T + \Delta T$), volvemos a optimizar. Iteraremos este proceso hasta llegar al tiempo de aplicación máximo que queramos estudiar. Aunque podamos creer que haciendo esto mantendremos como mínimo la fidelidad del pulso, ya que un pulso de un tiempo menor está siempre incluido en el espacio de configuraciones de un pulso de mayor tiempo, esto no es exactamente así. Esto es debido a que la función de penalización cambia ligeramente con el tiempo total del pulso,[3] por lo que es posible que al cambiar el pulso para minimizar esta, disminuya la fidelidad de nuestro pulso. Así, no debemos esperar un comportamiento monótono, aunque si que esperamos que las desviaciones respecto a este sean pequeñas.

Presentamos en la gráfica 2 el estudio de las infidelidades frente al tiempo total del pulso, para el caso de varios valores de α. El rango de tiempo de pulso se ha escogido de manera que cumplamos con el hecho de estar co-

[3] Recordemos la definición de $S(t)$ dada en 4.18

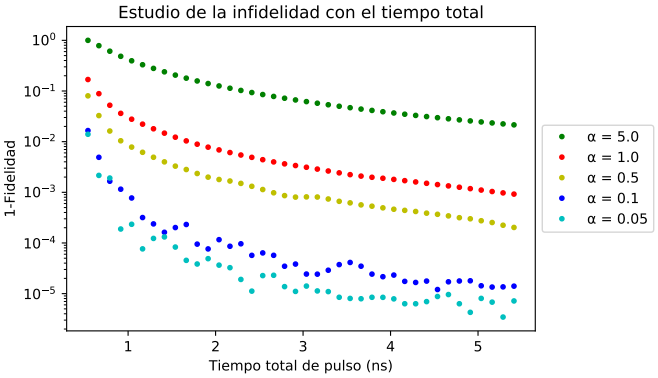


Figura 2: Estudio de la infidelidad de los pulsos óptimos para distintos valores de tiempo máximo de propagación. Para el cálculo de cada pulso se ha usado como semilla el pulso obtenido con el tiempo anterior.

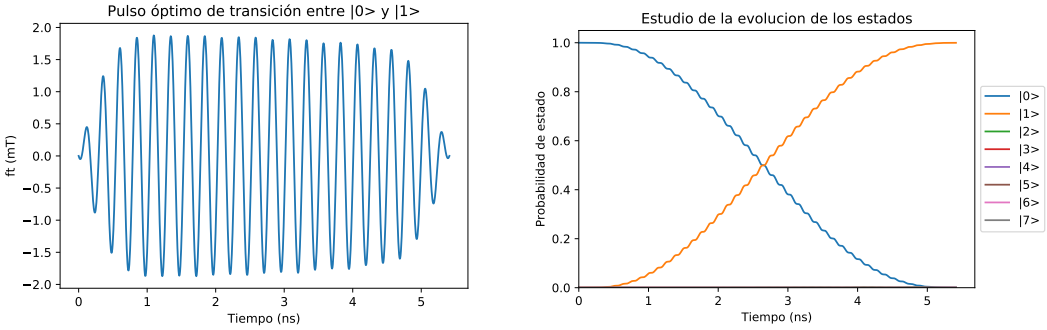
mo mínimo 2 órdenes de magnitud por debajo del tiempo de coherencia del sistema, que recordemos es de $\sim$0.5 μs. Observando esta figura, podemos ver como, claramente, conforme mayor es el tiempo de pulso, mejor es el pulso que podemos generar. Esto es esperable, ya que las transiciones son más fáciles si hay más tiempo. También podemos notar que conforme menos restringimos el pulso (valores de α más bajos), encontramos que menos infidelidad tenemos en el pulso optimizado. Este comportamiento se puede explicar ya que con amplitudes más altas podemos excitar más a los estados y conseguir transiciones mejores en menos tiempo.

Desde el punto de vista del espacio de configuraciones, estos dos resultados son esperables, ya que conforme más relajamos las condiciones (tiempos más largos y α más pequeñas), vamos descubriendo nuevos máximos globales, y no se pierden los que se tienen con condiciones más restrictivas.

Por último, veamos uno de los pulsos obtenidos en uno de los cálculos anteriores; hemos escogido el caso en el que el tiempo T es 5.41 ns, y α =0.5, por ejemplo. Presentamos en la figura 3 el pulso calculado, así como la evolución de cada uno de los niveles energéticos al aplicar este pulso al sistema. Podemos observar en esa evolución cómo el sistema comienza en el estado $|0\rangle$ y termina en el estado $|1\rangle$, sin prácticamente excitar otros estados intermedios, lo que da muestra de la gran calidad del pulso obtenido.

6. Diseño de puertas lógicas cuánticas

Finalmente, hemos resuelto el objetivo último planteado para este trabajo: el diseño de pulsos cuyo operador evolución inducido sea equivalente a una puerta cuántica. En concreto, vamos a generar la puerta de Toffoli, ya que es una puerta muy relevante para sistemas de 3 qubits. Además de esta, diseñaremos también una puerta Hadamard para el primer qubit, la cual



(a) (b)

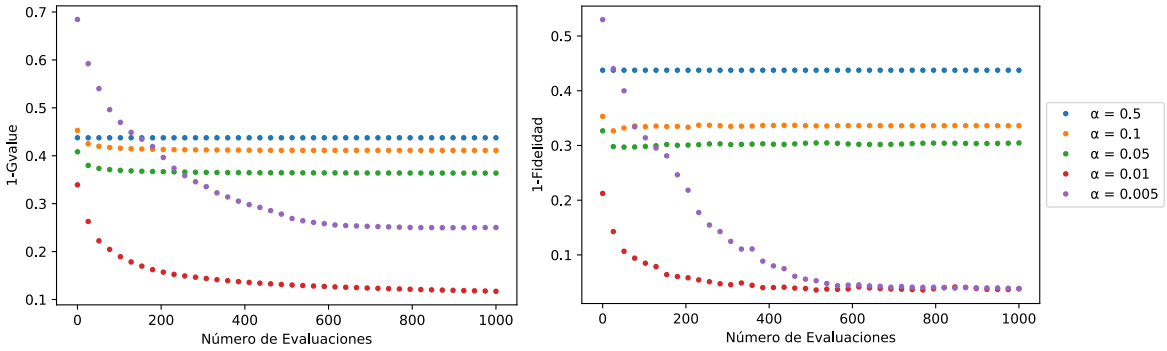
Figura 3: (a) Pulso óptimo para la transición $|0\rangle \rightarrow |1\rangle$ calculado usando el algoritmo de Krotov, para un valor de $\alpha = 0.5$ y $T = 5.41$ ns. El pulso posee una infidelidad de $2.84{\cdot}10^{-4}$. (b) Estudio de la evolución de los estados para el pulso calculado.

si se combina con la anterior nos puede dar lugar a un conjunto *universal* [7].

Haremos un pequeño estudio semejante al de la sección anterior, es decir, analizaremos cómo se comporta el algoritmo en función de los parámetros α y T (aunque esta vez no nos detendremos tanto, ya que muchas de las conclusiones a las que llegamos anteriormente nos sirven aquí también).

Así, presentamos en la figura 4 un estudio de la convergencia del valor de G y de la infidelidad en función del número de evaluaciones, en concreto para la generación de la puerta de Toffoli. Observamos que se comportan como cabría esperar. Así, el valor de la infidelidad de G converge monótonamente, confirmando que el algoritmo está bien implementado, y el valor de la infidelidad desciende conforme aumentamos las evaluaciones y bajamos α. Este es un comportamiento semejante al que ocurría cuando el objetivo era una transición entre estados.

Por otro lado presentamos en la figura 5 el valor de la infidelidad como función del tiempo máximo del pulso, también para el caso de la optimización de una para una puerta Toffoli. Tal y como ocurría para el caso de las transiciones de estado a estado, para valores altos de α la fidelidad es bastante mala independientemente del tiempo de pulso, y cuando el valor de esta es bajo encontramos una buena convergencia. Observando esta figura también podemos ver que encontramos una cierta *transición* cuando pasamos de un tiempo de pulso de 5 ns, en la cual la fidelidad baja notablemente para valores de α intermedios.



(a) (b)

Figura 4: (a) Estudio de la convergencia del valor del funcional G con el número de evaluaciones, para la optimización de un pulso equivalente a una puerta lógica cuántica de Toffoli, usando el algoritmo de Krotov. (b) Estudio de la evolución de la infidelidad con el número de evaluaciones para el mismo pulso anterior.

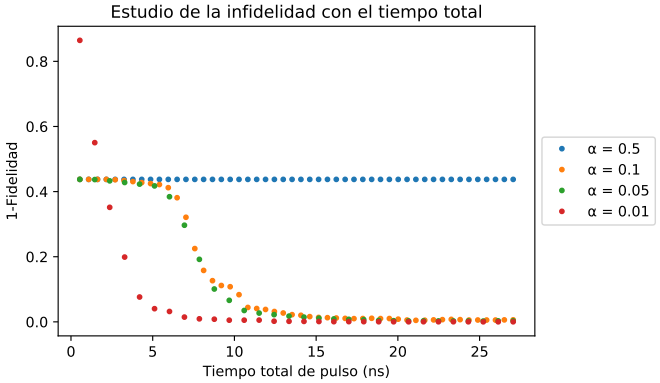
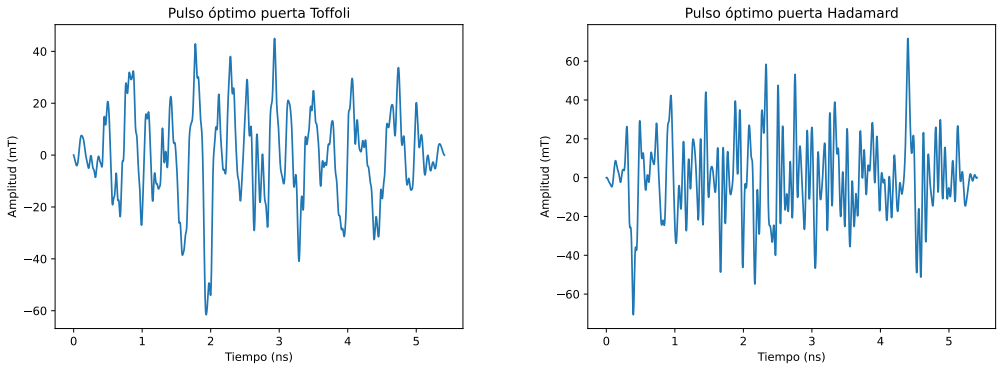


Figura 5: Estudio de la fidelidad con el tiempo máximo para un pulso equivalente a una puerta de Toffoli, usando el algoritmo de Krotov.

Terminaremos mostrando los pulsos obtenidos para ambas puertas (puerta de Toffoli, y puerta de Hadamard extendida a tres qubits), para alguno de los valores de α y T. En concreto, utilizaremos el valor de $\alpha = 0.01$, ya que nos permite obtener una fidelidad muy alta en el tiempo objetivo de $T = 5.41$ ns. Un problema que aparece al utilizar un valor de α pequeño es que hemos tenido que reducir el valor de la discretización del tiempo en el programa, ya que al trabajar con perturbaciones de mucha amplitud la propagación es numéricamente más complicada. El uso de una discretización temporal más fina implica aumentar el tiempo de computación.

Comencemos por la puerta de Toffoli. El funcionamiento de esta puerta es sencillo, ya que aplica la puerta NOT al tercer qubit del sistema, siempre que los otros dos estén en el estado $|1\rangle$. Por ello, también es llamada la puerta CCNOT. La forma matricial de esta matriz se puede expresar de manera



(a) (b)

Figura 6: Cálculos de las perturbaciones equivalente a un operador evolución que representen (a) la puerta de Toffoli, y (b) la puerta de Hadamard. Los cálculos se ha realizado con α =0.01, T =5.41 ns y 3000 evaluaciones del algoritmo de Krotov. La infidelidad de los pulsos es de (a) $2.611{\cdot}10^{-3}$ y (b) $9.615{\cdot}10^{-3}$.

sencilla usando la base computacional:

$$U_{\text{Toffoli}} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \tag{6.1}$$

Presentamos el pulso obtenido para la generación de esta puerta en la figura 6a.

La puerta de Hadamard transforma un qubit en una combinación lineal equilibrada de sus estados $|0\rangle$ y $|1\rangle$. Si queremos extender esta puerta a sistemas de tres qubits, podemos por ejemplo requerir que la puerta se aplique sobre el primer qubit, independientemente de los otros dos. Usando el método explicado en la sección 2 para tratar con puertas que afectan a un qubit en un sistema de múltiples qubits, se deduce que:

$$U_{\text{Hadamart}} = H \otimes \mathbb{I}_2 \otimes \mathbb{I}_2 = \frac{1}{\sqrt{2}} \begin{pmatrix} \mathbb{I}_4 & \mathbb{I}_4 \\ \mathbb{I}_4 & -\mathbb{I}_4 \end{pmatrix} \tag{6.2}$$

donde H representa la puerta Hadamard sobre un qubit

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \tag{6.3}$$

$\mathbb{I}_2$ se refiere a la identidad de dimensión 2 y $\mathbb{I}_4$ a la identidad de dimensión 4. Hemos presentamos el pulso obtenido equivalente a esta puerta en la figura 6b.

7. Conclusiones

En este trabajo, hemos implementado el algoritmo de Krotov, uno de los algoritmos "clásicos" en teoría de control óptimo, con el objetivo de realizar optimizaciones en sistemas cuánticas. En concreto, hemos trabajado sobre un modelo de una molécula magnética (GdW_{30}).

Hemos conseguido mostrar que, usando este algoritmo, podemos encontrar la forma de las perturbaciones que debemos introducir para que el operador evolución resultante tras un tiempo T sea equivalente a una puerta lógica. En concreto, hemos trabajado con las puertas de Toffoli y de Hadamard. En el algoritmo, debe fijarse de antemano el tiempo total de aplicación de la puerta. Como ejemplo, hemos mostrado casos en los que las puertas se han generado en un tiempo de 5.41 ns, dos órdenes de magnitud menor que el tiempo de coherencia de las transiciones características del sistema ($\sim$0.5 μ s), lo que permitiría aplicar muchas de estas transformaciones seguidas antes de perder la coherencia del sistema, y mantenernos dentro de la región de validez de la ecuación de Schrödinger.

Las perturbaciones que hemos obtenido poseen una infidelidad dentro del orden que buscábamos ($10^{-2} - 10^{-3}$). En el caso de que buscásemos pulsos con mayor fidelidad hay dos alternativas. Una, bajar el valor de la penalización que se incluye en el algoritmo para amplitudes altas. Esto sin embargo provocaría que la amplitud del pulso solución sea mayor, y por lo tanto lo haría más costoso a nivel experimental. Otra, subir el tiempo total de ejecución de las puertas, lo que reduciría el número de transformaciones que pueden hacerse en el rango de validez de la ecuación de Schrödinger.

El algoritmo de Krotov es únicamente uno de los muchos que pueden utilizarse para resolver las ecuaciones de la QOCT. Este estudio debiera completarse, en un futuro trabajo, con un estudio de la eficiencia comparada del algoritmo en relación con alguna de las alternativas. Así mismo, otra línea de trabajo que queda abierta sería la extensión del algoritmo para operar sobre ecuaciones para sistemas abiertos (como la ecuación de Lindblad), para poder tener en cuenta explícitamente la acción del entorno.

Bibliografía

[1] M. Nelsen, and I. Chuang. "Quantum Computation and Quantum Information". Cambridge University Press, 2010.

[2] M.D. Jenkins y col. "Coherent manipulation of three-qubit states in a molecular single-ion magnet". *Physical Review B* 95(6):064423, 2017.

[3] M.D. Jenkins y col. "A scalable architecture for quantum computation with molecular nanomagnets". *Dalton Transactions* 45(42):16682-16693, 2016.

[4] Constantin Brif and Raj Chakrabarti and Herschel Rabitz. "Control of quantum phenomena: past present and future". *New Journal of Physics* 12(7):075008, 2010.

[5] Yvon Maday and Gabriel Turinici. "New formulations of monotonically convergent quantum control algorithms". *The Journal of Chemical Physics*, 118(18):8191-8196, 2013.

[6] https://gitlab.com/acbarrigon/qocttools

[7] Yaoyun Shi. "Both Toffoli and Controlled-NOT need little help to do universal quantum computation". https://arxiv.org/abs/quant-ph/0205115

[8] M.J. Martínez-Pérez y col. "Gd-based single-ion magnets with tunable magnetic anisotropy: molecular design of spin qubits". *Physical review letters* 108(24):247213, 2012.

[9] José J. Baldoví y col. "Coherent manipulation of spin qubits based on polyoxometalates: the case of the single ion magnet [GdW 30 P 5 O 110] 14". *Chemical Communications* 49(79):8922-8924, 2013.

[10] Adrián García Carrizo. "Construcción computacional de puertas lógicas cuánticas mediante la teoría de control óptimo". Trabajo de Fin de Máster, Universidad de Zaragoza, 2020.

[11] Uri M. Ascher and Linda R. Petzold. "Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations". Philadelphia: Society for Industrial and Applied Mathematics, 1998.