

Università degli studi di Milano-Bicocca

Corso di Laurea Triennale in Fisica



Simulazioni di sistemi molecolari attraverso  
metodi di calcolo variazionale (VQE)  
implementando ansatz basati su impulsi

Relatore:  
Prof. Andrea Giachero

Correlatore:  
Dott. Rodolfo Carobene

Tesi di Laurea di:  
Riccardo Marega  
Matr. N. 865371

ANNO ACCADEMICO 2023/2024



# Sommario

Il seguente elaborato presenta lo studio eseguito per determinare le energie dello stato fondamentale delle molecole  $H_2$  (idrogeno molecolare) ed  $LiH$  (idruro di litio). Tale analisi è stata effettuata sfruttando metodi di computazione quantistica ed algoritmi variazionali.

L'argomentazione della tesi sarà strutturata in 4 capitoli.

Il primo fra questi si propone di esporre le basi nozionistiche necessarie per affrontare lo studio dell'argomento di tesi. All'interno di questo capitolo sarà introdotto il problema elettronico associato alle molecole di nostro interesse.

Il secondo capitolo si incentrerà sulla definizione delle basi del *quantum computing* e sui metodi computazionali. All'interno di questo capitolo sarà introdotto l'algoritmo utilizzato per lo studio dell'energia dello stato fondamentale delle molecole indicate: il *Variational Quantum Eigensolver* (VQE). Lo studio proseguirà con l'esposizione degli strumenti necessari per la scrittura ed esecuzione di un codice su di un computer quantistico e l'analisi per l'implementazione della VQE mediante ansatz generato da un programma di impulsi.

Nel terzo capitolo si raccolgono i metodi operativi e l'analisi dati. I risultati ottenuti saranno poi confrontati con i valori accertati dalla comunità scientifica e con i risultati ottenuti sfruttando metodi di computazione classica.

Nell'ultimo capitolo saranno esposte le conclusioni tratte dallo studio e verranno brevemente elencate possibili nuove applicazioni per i modelli ottenuti.

Si noti che al termine dell'elaborato sono riportate le referenze agli articoli e libri impiegati per la scrittura del seguente testo ed è inoltre disponibile il link alla pagina *github* dove è possibile trovare l'insieme dei codici utilizzati ed i dati ottenuti.



# Indice

<b>Sommario</b>	<b>III</b>
<b>1 Cenni teorici</b>	<b>5</b>
1.1 Algebra lineare . . . . .	5
1.2 Problema elettronico . . . . .	6
1.2.1 Approssimazione di Born-Oppenheimer . . . . .	8
1.2.2 Approssimazione di Hartree . . . . .	9
1.2.3 Approssimazione di Hartree-Fock . . . . .	10
1.2.4 Introduzione alle basi Gaussiane . . . . .	11
1.3 Seconda quantizzazione . . . . .	13
1.3.1 Bosoni e Fermioni . . . . .	14
1.3.2 Hamiltoniane di seconda quantizzazione . . . . .	14
1.3.3 Spazio di Fock . . . . .	15
<b>2 Quantum computing ed algoritmi variazionali</b>	<b>17</b>
2.1 Introduzione al quantum computing . . . . .	17
2.1.1 Gate e circuiti quantistici . . . . .	17
2.1.2 Variational Quantum Algorithms . . . . .	22
2.1.3 Qiskit ed IBM . . . . .	25
2.2 Pulse Based VQA . . . . .	29
2.2.1 Hardware Efficient Ansatz . . . . .	33
<b>3 Analisi Computazionale</b>	<b>39</b>
3.1 Introduzione al codice . . . . .	39
3.2 Risultati computazionali . . . . .	47
3.2.1 Idrogeno molecolare ( $H_2$ ) . . . . .	47

3.2.2	Idruro di litio ( <i>LiH</i> )	62
<b>4</b>	<b>Conclusioni</b>	<b>65</b>
4.1	Analisi dei risultati computazionali	65
4.2	Analisi del lavoro svolto	66
4.3	Possibili applicazioni e spunti per ulteriori analisi	67
4.3.1	Pulse ansatz optimization as a reinforcement learning problem	67
4.3.2	Zero noise extrapolation [16]	68

# Elenco delle figure

1	Sfera di Bloch implementata attraverso qiskit	4
2.1	Schema di un Variational Quantum Algorithm	22
2.2	Grafici rappresentati l'andamento di due possibili Cost function	23
2.3	LC vs JJ	31
2.4	Giunzione Josephson	31
2.5	Circuito annesso ad un transom	32
2.6	Forma funzionale di un impulso Gaussiano	37
3.1	Andamenti energetici di una moleola di $H_2$ sul simulatore FakeManila in presenza di rumore in funzione del numero di qubit e del tipo di mapping	41
3.2	Andamenti energetici di una moleola di $H_2$ al variare della base	42
3.3	Andamenti energetici di una moleola di $H_2$ al variare della distanza e del numero di iterazioni	48
3.4	Andamenti energetici di una moleola di $H_2$ al variare della distanza e del numero di iterazioni	49
3.5	Andamento energetico in funzione del numero di iterazioni attraversi <i>ibmq_montreal</i> con ottimizzatore di tipo non-gradient.	50
3.6	Stima dell'energia di ground della molecola $H_2$ in funzione del numero di iterazioni eseguite dall'ottimizzatore COBYLA.	51
3.7	Stima dell'energia di ground della molecola $H_2$ in funzione del numero di iterazioni eseguite dall'ottimizzatore COBYLA.	54
3.8	Andamenti energetici di una moleola di $H_2$ al variare della distanza e del numero di iterazioni	55

---

3.9	Confronto fra ansatz differenti (dove $D$ rappresenta un impulso DRAG e $C$ un impulso di controllo) per lo studio del numero di iterazioni necessarie alla convergenza . . . . .	56
3.10	Confronto fra tempi computazionali [s] associati ad ansatz differenti . . . . .	57
3.11	VQE con ansatz pulse based eseguito sul computer quantistico <i>ibm_nazca</i> . . . . .	58
3.12	VQE con ansatz pulse based eseguito sui computer quantistici <i>ibm_nazca</i> ed <i>ibm_osaka</i> . . . . .	59
3.13	Circuito RY-CNOT . . . . .	61
3.14	Confronto andamenti energetici in funzione del numero di iterazioni tra <i>ibm_nazca</i> ed <i>ibm_osaka</i> . . . . .	61
4.1	Risultati $H_2$ . . . . .	65
4.2	Risultati $LiH$ . . . . .	65



# Introduzione

L'idea di un computer il cui funzionamento fosse fondato sulle regole della meccanica quantistica fu introdotta per la prima volta tra la fine del 1970 e l'inizio del 1980 dai fisici ed informatici *Charles H. Bennet*, del centro di ricerca dell' IBM Thomas J. Watson, *Paul A. Benioff*, dall' Argonne National Laboratory in Illinois, *David Deustch*, dall'università di Oxford e *Richard P. Feynman* dalla Caltech. Ad oggi, i computer quantistici sono diventati una realtà con la quale chiunque può interracciarci, questo anche grazie ai servizi per il pubblico messi a disposizione dall'IBM.

Un computer quantistico è un dispositivo che espande le capacità computazionali di un computer classico attraverso l'elaborazione di informazione quantistica. L'unità base dell'informazione processata da un computer quantistico è il qubit. Definendo gli stati base di un qubit come  $|0\rangle$  ed  $|1\rangle$ , in generale lo stato di un qubit è una superposizione di questi due stati  $|\psi\rangle = \sum_{z=0,1} c_z |z\rangle$  con  $c_z$  numeri complessi in grado di soddisfare la condizione di normalizzazione:  $\sum_z |c_z|^2 = 1$ . Un modo comune per visualizzare lo stato di un singolo qubit è quello di parametrizzarlo con la funzione d'onda  $|\psi\rangle = \cos(\theta/2)|0\rangle + e^{i\phi}\sin(\theta/2)|1\rangle$ , dove gli angoli  $\theta$  e  $\phi$  mappano lo stato in un punto posto sulla superficie di una sfera che prende il nome di sfera di Bloch, dove i poli rappresentano gli "stati classici".

Lo studio riportato sfrutta le tecnologie messe a disposizione dall'azienda IBM. L'insieme dei codici scritti si basa su *Qiskit*, un tool-kit open-source sviluppato dalla stessa IBM del quale avremo modo di approfondire le funzionalità all'interno del terzo capitolo.

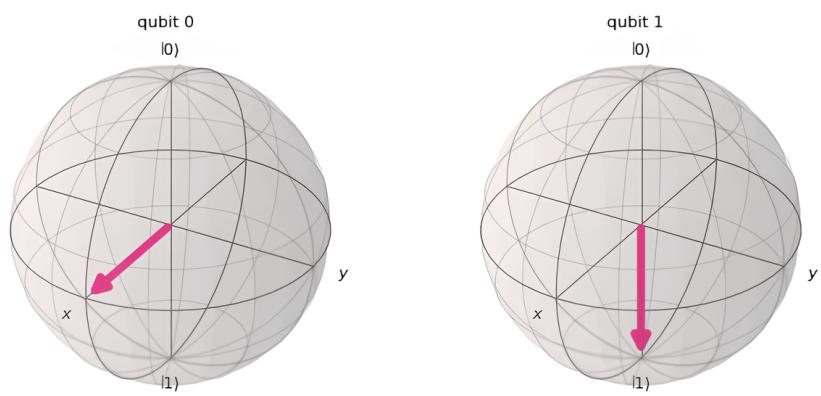


Figura 1: Sfera di Bloch implementata attraverso qiskit

# Capitolo 1

## Cenni teorici

### 1.1 Algebra lineare

Riprendendo quanto brevemente presentato nel paragrafo d'introduzione, un qubit può essere rappresentato come un vettore  $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ , dove  $\alpha$  e  $\beta$  sono numeri complessi che soddisfano la condizione di normalizzazione:  $|\alpha|^2 + |\beta|^2 = 1$ . Il simbolo " $| \rangle$ " costituisce la notazione standard (notazione di Dirac) per specificare gli stati in meccanica quantistica. La base ortonormale dello spazio bidimensionale  $\{|0\rangle, |1\rangle\}$  espressa come:

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

prende il nome di base computazionale.

Gli stati quantistici si combinano attraverso il prodotto tensoriale:

$$|\psi_1\rangle \otimes |\psi_2\rangle = \begin{bmatrix} \psi_{1,0} \\ \psi_{1,1} \end{bmatrix} \otimes \begin{bmatrix} \psi_{2,0} \\ \psi_{2,1} \end{bmatrix} = \begin{bmatrix} \psi_{1,0}\psi_{2,0} \\ \psi_{1,0}\psi_{2,1} \\ \psi_{1,1}\psi_{2,0} \\ \psi_{1,1}\psi_{2,1} \end{bmatrix}$$

Le matrici codificano le operazioni sullo spazio. In particolare, facendo riferimento al secondo postulato della meccanica quantistica<sup>1</sup>, le matrici Hermitiane rappresentano le osservabili del sistema. Una porta quantistica è descritta da una matrice unitaria e consente di cambiare lo stato di un qubit. Poiché queste matrici sono

---

<sup>1</sup>Ad ogni osservabile  $A$  corrisponde un operatore lineare ed autoaggiunto  $\hat{A}$  in  $\mathcal{H}$ .

unitarie, la trasformazioni da esse implementate sono invertibili. Questa è una differenza fondamentale rispetto alle porte dei computer classici, in grado di implementare esclusivamente trasformazioni non invertibili. Se il vettore di input è un n-qubit, allora è necessario lavorare con matrici unitarie  $n \times n$  con elementi complessi. In questa situazione, il vettore di output è un n-qubit.

Le proprietà matematiche dei qubit sono regolamentate dalle proprietà delle matrici di Pauli:

$$\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \quad \sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \quad \sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

che soddisfano le seguenti regole di commutazione ed anticommutazione:

$$[\sigma_\alpha, \sigma_\beta] = 2i\epsilon_{\alpha\beta\gamma}\sigma_\gamma, \quad \{\sigma_\alpha, \sigma_\beta\} = 2\delta_{\alpha\beta}\mathbb{1}$$

dove  $\epsilon_{\alpha\beta\gamma}$  è il tensore di Levi-Civita e  $\delta_{\alpha\beta}$  è la delta di Kronecker.

## 1.2 Problema elettronico

Tra gli oggetti degli studi svolti nell'ambito della chimica quantistica vi è quello di determinare gli autostati associati all'Hamiltoniana elettronica delle molecole. L'Hamiltoniana che descrive un sistema di elettroni interagenti sottoposti al potenziale generato dai nuclei atomici può essere scritta come:

$$\hat{H} = \hat{H}_1 + \hat{H}_2 \tag{1.1}$$

dove:  $\hat{H}_1 = \sum_{i=1}^N -\frac{1}{2}\Delta_i^2 + V(r_i)$  e  $\hat{H}_2 = \sum_{1 \leq i < j \leq K} \frac{1}{|r_i - r_j|}$ . In questi termini  $K$  identifica il numero di elettroni,  $r_i$  è l'operatore posizione dell'i-esimo elettrone,  $\Delta_i$  è il corrispettivo Laplaciano e  $V(r)$  è il potenziale elettronico generato dai nuclei.

Il termine  $\hat{H}_1$  include i termini cinetici e l'energia potenziale degli elettroni non interagenti mentre il termine  $\hat{H}_2$  costituisce il termine di interazione Coulombiana. In questo tipo di descrizione si trascurano completamente i termini relativistici e si adotta l'approssimazione di Born-Oppenheimer (la cui trattazione è ripresa successivamente 1.2.1). Ogni elettrone è descritto quanto-mecanicamente dalla sua posizione  $r_i \in R^3$  e dal suo spin  $\omega_i \in \{\uparrow, \downarrow\}$ . Di conseguenza, un sistema costituito da  $K$  elettroni può essere descritto dalla funzione d'onda  $\psi(\vec{x}_1, \dots, \vec{x}_K)$

dove  $\vec{x}_i = (r_i, \omega_i)$ . Questa funzione d'onda deve rispettare la statistica di Fermi e dunque  $\psi$  deve essere anti-simmetrica per lo scambio di coordinate di un elettrone  $x_i$  con un elettrone  $x_j$ . Si sfrutta il determinante di Slater per ottenere una funzione d'onda totalmente anti-simmetrica:

$$\psi(\vec{x}_1, \dots, \vec{x}_K) \propto \det \begin{bmatrix} \psi_1(x_1) & \cdots & \psi_1(x_K) \\ \vdots & \ddots & \vdots \\ \psi_K(x_1) & \cdots & \psi_K(x_K) \end{bmatrix} \quad (1.2)$$

Quello che si è fatto è l'aver troncato lo spazio di Hilbert associato ad un singolo elettrone ad uno spazio finito le cui basi sono le funzioni d'onda  $\psi_1, \dots, \psi_N$  (dove  $N$  identifica la dimensione dello spazio di Hilbert) che prendono il nome di orbitali. In questa rappresentazione dunque l'insieme dei determinanti di Slater può esser identificato come il sottospazio anti-simmetrico dell'insieme  $(C^N)^{\otimes K}$ . La proiezione dell'Hamiltoniana  $\hat{H}$  nello spazio delle funzioni anti-simmetriche risulta essere:

$$\hat{H} = \sum_{i=1}^K \sum_{p,q=1}^N t_{pq} |p\rangle\langle q|_i \quad (1.3)$$

dove  $|p\rangle \equiv |\psi_p\rangle$  e  $t_{pq} = \langle \psi_p | (-\frac{\Delta}{2} + V) | \psi_q \rangle$ .

Ogni elettrone nello spazio di Hilbert  $\mathbb{C}^N$  è codificato in un registro di  $\log_2 N$  qubit, questo vuol dire che sono necessari un totale di  $K \log_2 N$  qubits per descrivere l'intero sistema.

Un parametro importante che influenza i tempi necessari per una simulazione di questo tipo è il grado di "sparsity" dell'Hamiltoniana. Un' Hamiltoniana  $\hat{H}$  si dice essere d-sparse se la matrice  $\hat{H}$  nella base di  $n$  qubit possiede al massimo  $d$  elementi non-nulli in ogni riga (o equivalentemente in ogni colonna).

Un secondo metodo applicabile per la descrizione di un sistema come quello precedente consiste nel considerare un'Hamiltoniana della forma:

$$H = \sum_{p,q=1}^N t_{p,q} \hat{c}_p^\dagger \hat{c}_q + \frac{1}{2} \sum_{p,q,r,s=1}^N u_{pqrs} \hat{c}_p^\dagger \hat{c}_q^\dagger \hat{c}_r \hat{c}_s \quad (1.4)$$

dove  $\hat{c}_p^\dagger$  e  $\hat{c}_p$  sono gli operatori di creazione e distruzione dell'orbite  $\psi_p$ .

L'Hamiltoniana in questione è definita nello spazio di Fock, questo spazio è generato da  $2^N$  vettori  $|n_1, \dots, n_N\rangle$  dove  $n_p \in \{0, 1\}$  si riferisce all'orbitale  $\psi_p$ .

*La scelta di una descrizione di questo tipo è giustificata successivamente con l'introduzione teorica all'approssimazione di Hartree-Fock.*

### 1.2.1 Approssimazione di Born-Oppenheimer

Per comprendere l'approssimazione di Born-Oppenheimer risulta necessario richiamare l'Hamiltoniana di una molecola:

$$\hat{H}|\Psi(\vec{X}, \vec{x})\rangle = (\hat{T}_N + \hat{T}_e + \hat{V}_{NN} + \hat{V}_{eN} + \hat{V}_{ee})|\Psi(\vec{X}, \vec{x})\rangle \quad (1.5)$$

con

$$\hat{T}_N := -\sum_{\alpha=1}^N \frac{1}{2M_\alpha} \nabla_\alpha^2, \quad (1.6)$$

$$\hat{T}_e := -\sum_{i=1}^N \frac{1}{2} \nabla_i^2, \quad (1.7)$$

$$\hat{V}_{NN} := \frac{1}{2} \sum_{\alpha \neq \beta} Z_\alpha Z_\beta \frac{1}{R_{\alpha\beta}}, \quad (1.8)$$

$$\hat{V}_{eN} := -\sum_{i=1}^N \sum_{\alpha=1}^{NN} \frac{Z_\alpha}{R_{\alpha i}}, \quad (1.9)$$

$$\hat{V}_{ee} := \frac{1}{2} \sum_{i \neq j} \frac{1}{r_{ij}}, \quad (1.10)$$

dove  $\hat{T}_N$  rappresenta il termine cinetico associato ai nuclei,  $\hat{T}_e$  rappresenta il termine cinetico associato agli elettroni,  $\hat{V}_{NN}$  rappresenta il termine di energia potenziale di interazione nucleo-nucleo ed analogamente  $\hat{V}_{eN}$  e  $\hat{V}_{ee}$  rappresentano il termine di energia potenziale per le interazioni elettrone-nucleo ed elettrone-elettrone.

Il punto di partenza dell'approssimazione di Born-Oppenheimer consiste nell'assumere che esista una soluzione dell'equazione (1.5) scrivibile come

$$\Psi(\vec{X}, \vec{x}) = \Psi_N(\vec{X}) \Psi_e(\vec{X}; \vec{x}),$$

ovvero come prodotto di una funzione d'onda associata ai nuclei della molecola con una funzione d'onda associata alla componente elettronica<sup>2</sup>.

Nel quadro della teoria di Born-Oppenheimer, si fa un'ulteriore ipotesi: si assume che siano note le equazioni che regolano le parti elettroniche e nucleari della funzione d'onda. Pertanto, si assume che  $\Psi_e(\vec{x}; \vec{X})$  sia una soluzione dell'equazione di Schrödinger stazionaria con nuclei fissi,

$$(\hat{T}_e + \hat{V}_{eN}(r; R) + \hat{V}_{ee}(r))|\Psi_e(\vec{x}; R)\rangle := \hat{H}_e(R)|\Psi_e(\vec{x}; R)\rangle,$$

dove l'operatore hamiltoniano elettronico  $\hat{H}_e(R)$  e l'energia elettronica  $E_e(R)$  (entrambi dipendenti dalla posizione dei nuclei) sono stati definiti e, poiché gli spin nucleari non entrano nell'espressione, abbiamo indicato esplicitamente che  $\Psi_e$  dipende parametricamente da  $R$  e non da  $\vec{X}$ . A questo punto è sufficiente studiare il problema nucleare con operatore hamiltoniano cinetico e un termine di potenziale efficacie dato dalla soluzione del problema elettronico.

Si noti come la seguente approssimazione ha portato alla scrittura dell'Hamiltoniana della forma

$$\hat{H} = \hat{H}_1 + \hat{H}_2,$$

dove i seguenti termini sono quelli indicati nella parte introduttiva del problema elettronico (1.1).

### 1.2.2 Approssimazione di Hartree

Una delle prime e più semplici approssimazioni impiegate per la risoluzione del problema elettronico presentato è proposta da Hartree nel 1927. In questa approssimazione la funzione d'onda viene scritta come prodotto (*prodotto di Hartree*) di  $N$  funzioni d'onda a un elettrone (orbitali) dove vengono trascurati lo spin e l'antisimmetria:

$$\Phi(r_1, \dots, r_N) = \prod_{i=1}^N \phi_i(r_i). \quad (1.11)$$

---

<sup>2</sup>L'utilizzo della seguente notazione ";" nella scrittura della funzione d'onda associata alla componente elettronica di una molecola serve ad indicare che essa è funzione del set di coordinate elettroniche e dipende "parametricamente" dal set di coordinate associate ai nuclei

Una delle condizioni richieste è la normalizzazione delle funzioni d'onda a singolo elettrone:

$$\langle \phi_i | \phi_i \rangle = 1, \quad i = 1, \dots, N$$

Definito dunque il funzionale  $F[\Psi] := \langle \Psi | \hat{H} | \Psi \rangle$ , è possibile risolvere il problema della ricerca degli autovalori associati ad un Hamiltoniana sfruttando i *moltiplicatori di Lagrange*. Si introduce un funzionale associato  $\tilde{F}[\Psi] := F[\Psi] + \lambda (\langle \Psi | \Psi \rangle - 1)$  e se si deriva quest'ultimo per il complesso coniugato della funzione d'onda e si impone la derivata esser nulla, si trovano gli autovalori associati all'Hamiltoniana i cui autostati soddisfano la condizione di normalizzazione.

Per quanto concerne l'approssimazione di Hartree, si può riscrivere il funzionale associato ad  $F[\Psi]$  come:

$$\tilde{F}[\{\phi_i\}] = \left\langle \prod_{i=1}^N \phi_i(r_i) \mid \hat{H} \mid \prod_{i=1}^N \phi_i(r_i) \right\rangle - \sum_{i=1}^N \epsilon_i (\langle \phi_i | \phi_i \rangle - 1),$$

dove  $\epsilon_i$  rappresentano N moltiplicatori di Lagrange. Risolvendo si ottiene

$$\frac{\partial \tilde{F}[\{\phi_i\}]}{\partial \phi_k^*} = \left( -\frac{1}{2} \nabla^2 + \hat{V}_e(r) + \hat{V}_k^e(r) - \epsilon_k \right) \phi_k(r),$$

da cui, imponendo nulla la derivata per  $k = 1, \dots, N$ , si arriva all'*Equazione di Hartree*:

$$\hat{H}_k[\phi] \phi_k(r) := \left( -\frac{1}{2} \nabla^2 + \hat{V}_N(r) + \hat{V}_k^e(r) \right) \phi_k(r) = \epsilon_k \phi_k(r) \quad (1.12)$$

### 1.2.3 Approssimazione di Hartree-Fock

Sebbene l'approssimazione di Hartree costituiscia uno degli approcci fondamentali proposti per la risoluzione del problema elettronico, esso non è quasi mai impiegato poichè manca nel considerare l'indistinguibilità degli elettroni. Questa osservazione fu per la prima volta effettuata separatamente dai fisici Fock e Slater nel 1930 che ne proposero una correzione introducendo un nuovo ansatz per la funzione d'onda elettronica che prese il nome di determinante di Slater.

La conseguenza matematica del considerare l'indistinguibilità di un set di N particelle risiede nello scivere una funzione d'onda che rimanga invariata (simmetrica) per lo scambio di coordinate di due particelle oppure cambi il segno (antisimmetrica), questo in funzione del tipo di particelle presenti nel sistema. Nel primo di

questi due casi le particelle prendono il nome di *bosoni* e sono caratterizzate da spin intero, mentre nel secondo caso le particelle prendono il nome di *fermioni* e sono caratterizzate da uno spin semi-intero.

Nello studio del problema elettronico associato ad una molecola ci si trova a dover descrivere il comportamento di un insieme di  $N$  elettroni, particelle che appartengono al gruppo dei fermioni in quanto caratterizzate da spin  $\frac{1}{2}$ . Si procede dunque con la costruzione di una funzione d'onda antisimmetrica, sfruttando il determinante di Slater:

$$\Psi(x_1, \dots, x_N) = \frac{1}{\sqrt{N!}} \begin{vmatrix} \psi_1(x_1)\psi_2(x_1)\dots\psi_N(x_1) \\ \psi_1(x_2)\psi_2(x_2)\dots\psi_N(x_2) \\ \vdots & \vdots & \vdots \\ \psi_1(x_N)\psi_2(x_N)\dots\psi_N(x_N) \end{vmatrix} \quad (1.13)$$

Definita pertanto la funzione d'onda di un insieme di  $N$  elettroni, si richiede che le singole funzioni d'onda (orbitali)  $\psi_i$  non solo soddisfino la condizione di normalizzazione ma che siano ortogonali tra loro:

$$\langle \psi_i | \psi_j \rangle = \delta_{ij}$$

Si può dunque procedere col calcolo del valore di aspettazione dell'energia associata ad uno stato scrivibile con una funzione d'onda come quella presentata precedentemente:

$$\langle \Psi | \hat{H} | \Psi \rangle = \sum_i \langle \Psi | \hat{h}_i | \Psi \rangle + \frac{1}{2} \sum_{i \neq j} \langle \Psi | \frac{1}{r_{ij}} | \Psi \rangle$$

dove

$$\hat{h}_i := -\frac{\nabla_i^2}{2} - \sum_{\alpha=1}^N \frac{Z_\alpha}{|\mathbf{R}_\alpha - \mathbf{r}_i|}$$

#### 1.2.4 Introduzione alle basi Gaussiane

Gli unici problemi nell'ambito della chimica quantistica in grado di essere risolti esattamente sfruttando la meccanica quantistica non relativistica sono quelli riconducibili agli atomi idrogenoidi: sistemi formati da un nucleo di carica  $Z$  ed un solo elettrone.

Le funzioni d'onda spaziali dell'Hamiltoniana di questo tipo di problemi (esprese in coordinate sferiche) sono scrivibili come:

$$\phi_{nlm}(r, \theta, \phi) = \sqrt{\frac{2Z^n}{n^3}} \frac{(n-l-1)!}{2n[(n+l)!]^3} \left(\frac{2Z^n}{r}\right)^l L_{n-1}^{2l+1} \left(\frac{2Z^n}{r}\right) e^{-\frac{Zr}{n}} Y_{lm}(\theta, \phi)$$

dove  $n$ ,  $l$  ed  $m$  sono i numeri quantici associati.

Sfruttando questo modello, vengono introdotte dai fisici Slater e Zener, nel 1930, funzioni d'onda del tipo

$$\chi_{\text{STO}a}(r; \mathbf{R}_{\alpha a}) = N_{\text{STO}a} \tilde{Y}_{c,s}^{\lambda_a}(\theta_{\alpha a}, \phi_{\alpha a}) |r - \mathbf{R}_{\alpha a}|^{n_a-1} \exp(-\zeta_a |r - \mathbf{R}_{\alpha a}|)$$

che prendono il nome di *Slater-type-orbitals* (STOs).

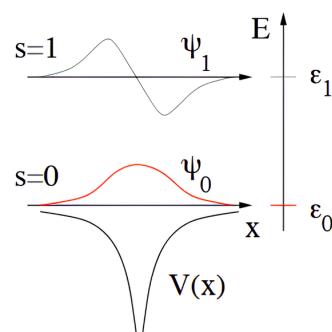
In quest'espressione  $N_{\text{STO}a}$  è la costante di normalizzazione e  $\zeta_a$  è un parametro della funzione.

Le funzioni  $\chi_{\text{STO}a}$  sono caratterizzate da una serie di proprietà che le rendono particolarmente utili. Una fra le più importanti è la presenza di una cuspide per  $|r - \mathbf{R}_{\alpha a}| \rightarrow 0$ , come richiesto nel teorema di Kato<sup>3</sup>; sono inoltre caratterizzate da una decrescita esponenziale quando  $|r - \mathbf{R}_{\alpha a}| \rightarrow \infty$ .

Nonostante ciò, la proiezione di una funzione d'onda molecolare scritta attraverso l'approssimazione di Hartree-Fock su una base gaussiana presenta alcune limitazioni computazionali: sebbene gli integrali associati al termine  $\langle \Psi | \hat{h} | \Psi \rangle$  siano calcolabili analiticamente, il termine  $\langle \Psi | \frac{1}{r_{ij}} | \Psi \rangle$  non può essere risolto analiticamente.

---

<sup>3</sup>Il teorema di Kato asserisce che per un generico potenziale Coulombiano, la densità di elettroni ha una cuspide in concomitanza con la posizione dei nuclei



Funzione d'onda associata ai primi due livelli energetici di un elettrone in presenza di un potenziale Coulombiano

Questo problema divenne noto come "*the nightmare of the integrals*" e portò alla necessità di costruire una nuova base su cui espandere le funzioni d'onda, vennero così introdotte le funzioni *Cartesian-Gaussian-type orbitals* (cGTO):

$$\chi_{\text{cGTO}a}(r; \mathbf{R}_{\alpha a}) = N_{\text{cGTO}a} (r_1 - R_{1\alpha a})^l x_a (r_2 - R_{2\alpha a})^l y_a (r_3 - R_{3\alpha a})^l z_a e^{(-\zeta_a |r - \mathbf{R}_{\alpha a}|^2)}$$

dove  $r_p$  e  $R_{p\alpha a}$ , con  $p = 1, 2, 3$ , sono le coordinate euclidee dell'elettrone e del  $\alpha a$ -esimo nucleo rispettivamente. Gli interi  $l_x^a$ ,  $l_y^a$ , e  $l_z^a$ , che assumono valori da 0 a  $\infty$ , sono chiamati numeri quantici orbitali.

## 1.3 Seconda quantizzazione

Seconda quantizzazione è il nome che si da ad una serie di tecniche in grado di trattare sistemi costituiti da un gran numero di particelle. Alla base di questo strumento matematico vi è la descrizione quantitativa del numero di particelle presenti in un certo stato, si parla dunque di *occupation number representation*. Nello studio di un sistema di particelle in meccanica quantistica si esordisce definendo il numero di quest'ultime. In uno studio effettuato attraverso la seconda quantizzazione si rilassa questa condizione e si assume che il numero di particelle possa variare. Per una descrizione di questo tipo si introducono un operatore di creazione e distruzione, in maniera analoga a quanto fatto in meccanica quantistica, in grado di creare oppure distruggere direttamente le particelle di un sistema.

**Definizione** Sia  $|\alpha\rangle$  un set di stati a singola particella ortonormali che soddisfano la condizione:

$$\langle \alpha | \beta \rangle = \delta_{\alpha, \beta}.$$

Si definisce l'operatore  $a_\alpha^\dagger$  e lo stato  $|0\rangle$ , chiamato **vuoto**, tale che:

$$a_\alpha^\dagger |0\rangle = |\alpha\rangle = |1_\alpha\rangle. \quad (1.14)$$

L'operatore  $a_\alpha^\dagger$  pertanto crea una particella in uno stato a singola particella  $|\alpha\rangle$ . Si noti che la notazione  $|1_\alpha\rangle$  viene utilizzata per evidenziare il fatto che lo stato sia a singola particella.

**Definizione** Si definisce l'operatore  $a_\alpha$  tale che:

$$a_\alpha |\beta\rangle = \delta_{\alpha,\beta} |0\rangle. \quad (1.15)$$

La caratterizzazione del numero di particelle appartenenti ad un sistema è definita attraverso l'operatore **Numero**:

$$N = \sum_\alpha a_\alpha^\dagger a_\alpha$$

### 1.3.1 Bosoni e Fermioni

La seconda quantizzazione nasce dalla necessità di descrivere con maggior accuratezza sistemi di particelle identiche. Nel 1940 il fisico Wolfgang Pauli pubblica un articolo intitolato "*The connection between spin and statistics*", dove mostra che, come conseguenza del gruppo di Lorentz della relatività ristretta, particelle identiche posso comportarsi in uno dei due seguenti modi:

- Bosoni:  $a_\beta^\dagger a_\alpha^\dagger |0\rangle = a_\alpha^\dagger a_\beta^\dagger |0\rangle$
- Fermioni:  $a_\beta^\dagger a_\alpha^\dagger |0\rangle = -a_\alpha^\dagger a_\beta^\dagger |0\rangle$

### 1.3.2 Hamiltoniane di seconda quantizzazione

Al fine di comprendere come esprimere l'Hamiltoniana di seconda quantizzazione di un sistema si procede dall'analisi di quanto già noto di un sistema semplice.

**Particella libera** Data l'Hamiltoniana di una particella libera  $H = \frac{p^2}{2m}$ , l'Hamiltoniana di seconda quantizzazione di una particella libera viene rappresentata come  $H = \sum_k \frac{k^2}{2m} a_k^\dagger a_k$ , dove quello che si è fatto è stato contare il numero di particelle accomunate dallo stesso momento  $k$ .

**Hamiltoniana di interazione elettronica** A fronte di quanto discusso nella trattazione del problema elettronico, considerando inoltre quanto appena introdotto, è possibile scrivere l'Hamiltoniana associata ad un sistema di elettroni interagenti come:

$$H = \sum_{k\sigma} \epsilon_k a_{k\sigma}^\dagger a_{k\sigma} + \frac{1}{2} \sum_{k_1 k'_1 \sigma k_2 k'_2 \sigma'} U_{k'_1 k'_2; k_2 k_1} a_{k'_1 \sigma}^\dagger a_{k'_2 \sigma'}^\dagger a_{k_2 \sigma'} a_{k_1 \sigma}, \quad (1.16)$$

dove gli elementi matriciali dell'interazione tra elettroni sono dati da

$$U_{k'_1 k'_2; k_2 k_1} = \int d^3r \int d^3r' \psi_{k'_1}^*(r) \psi_{k'_2}^*(r') U(r - r') \psi_{k_2}(r') \psi_{k_1}(r)$$

ed il primo termine dell'Hamiltoniana risulta essere

$$\epsilon_k = \int d^3r \psi_k^*(r) \left[ -\frac{\hbar^2}{2m} \nabla^2 + V(r) \right] \psi_k(r).$$

### 1.3.3 Spazio di Fock

Sulla base di quanto asserito fin'ora, siamo in grado di definire la funzione d'onda di un sistema costituito da N particelle come:

$$\frac{1}{\sqrt{\prod_\lambda n_\lambda}} a_{\lambda_N}^\dagger \dots a_{\lambda_1}^\dagger |\Omega\rangle = |\lambda_1 \dots \lambda_N\rangle$$

dove gli operatori di creazione e distruzione devono soddisfare le seguenti proprietà di commutazione:

$$[a_\lambda, a_\mu^\dagger] = \delta_{\lambda\mu}, \quad [a_\lambda, a_\mu]_{-\zeta} = 0, \quad [a_\lambda^\dagger, a_\mu^\dagger]_{-\zeta} = 0,$$

dove  $\zeta = -1$  per fermioni e  $\zeta = +1$  per bosoni.

Se si definisce  $F_N$  la combinazione lineare degli stati di N particelle  $F_N = |\lambda_1 \dots \lambda_N\rangle$ , allora lo spazio di Fock  $\mathbf{F}$  risulta essere:

$$F = \bigoplus_{N=0}^{\infty} F_N. \tag{1.17}$$



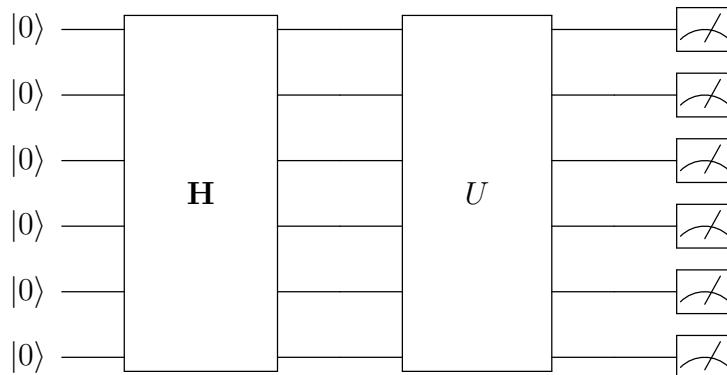
# Capitolo 2

## Quantum computing ed algoritmi variazionali

### 2.1 Introduzione al quantum computing

#### 2.1.1 Gate e circuiti quantistici

**Definizione di Quantum Computing** Si definisce un'operazione di computazione quantistica l'insieme  $\{\mathbf{H}, U, M_m\}$  dove  $\mathbf{H} = \mathbb{C}^{2^n}$  è lo spazio di Hilbert del registro di n-qubit,  $U \in U(2^n)$  rappresenta l'algoritmo quantistico ed  $M_m$  costituisce l'insieme degli operatori di misura.



In figura è riportata una rappresentazione grafica di un generico circuito quantistico che implementa un set di operazioni  $U$  ad un registro di qubit il cui stato dipende dall'insieme delle operazioni  $H$ .

L'insieme delle operazioni eseguite su di un qubit è implementata attraverso quantum gates (la cui natura concettuale richiama quella delle porte logiche utilizzate nella computazione classica). La proprietà di linearità permette di studiare l'azione di un quantum gate su di un qubit trasponendo l'operazione agli elementi della base computazione.

Di seguito sono elencati alcuni dei quantum gates principali:

Gate	Rappresentazione Matriciale	Circuito Quantistico
Hadamard (H)	$\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$	$ 0\rangle \xrightarrow{\boxed{H}} \frac{ 0\rangle +  1\rangle}{\sqrt{2}}$ $ 1\rangle \xrightarrow{\boxed{H}} \frac{ 0\rangle -  1\rangle}{\sqrt{2}}$
Pauli-X (X)	$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$	$ 0\rangle \xrightarrow{\boxed{X}}  1\rangle$ $ 1\rangle \xrightarrow{\boxed{X}}  0\rangle$
Pauli-Y (Y)	$\begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$	$ 0\rangle \xrightarrow{\boxed{Y}} i 1\rangle$ $ 1\rangle \xrightarrow{\boxed{Y}} -i 0\rangle$
Pauli-Z (Z)	$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$	$ 0\rangle \xrightarrow{\boxed{Z}}  0\rangle$ $ 1\rangle \xrightarrow{\boxed{Z}} - 1\rangle$
C-NOT	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$	$ 00\rangle \xrightarrow{\bullet \oplus}  00\rangle$ $ 10\rangle \xrightarrow{\bullet \oplus}  11\rangle$ $ 11\rangle \xrightarrow{\bullet \oplus}  10\rangle$ $ 01\rangle \xrightarrow{\bullet \oplus}  01\rangle$
S	$\begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$	$ 0\rangle \xrightarrow{\boxed{S}}  0\rangle$ $ 1\rangle \xrightarrow{\boxed{S}} i 1\rangle$
T	$\begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix}$	$ 0\rangle \xrightarrow{\boxed{T}}  0\rangle$ $ 1\rangle \xrightarrow{\boxed{T}} e^{i\pi/4} 1\rangle$

### Quantum gates universali

**Teorema** Il set dei gate a singolo qubit ed il gate CNOT sono universali. Ogni gate unitario applicato ad un registro di  $n$  qubit può essere implementato attraverso gate a singolo qubit e gate CNOT.

Sfruttando l'interpretazione di un qubit come un punto appartenente alla superficie di una sfera unitaria, è possibile introdurre il concetto di rotazione. L'implementazione di un'operazione di rotazione è permessa attraverso l'applicazione allo stato associato al qubit di un'adeguata matrice unitaria 2x2 della forma

$$U_i(\theta) \equiv e^{-i\frac{\theta}{2}\sigma_i} = \sigma_0 \cos\left(\frac{\theta}{2}\right) - i\sigma_i \sin\left(\frac{\theta}{2}\right),$$

dove  $\sigma_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$  costituisce la matrice identità.

**Lemma** Sia  $U \in SU(2)$ <sup>1</sup>. Si dimostra esistere  $\alpha, \beta, \gamma \in \mathbf{R}$  tali che  $U = R_z(\alpha)R_y(\beta)R_z(\gamma)$ , dove

$$\begin{aligned} R_z(\alpha) &= \exp\left(\frac{i\alpha\sigma_z}{2}\right) = \begin{bmatrix} e^{i\alpha/2} & 0 \\ 0 & e^{-i\alpha/2} \end{bmatrix} \\ R_y(\beta) &= \exp\left(\frac{i\beta\sigma_y}{2}\right) = \begin{bmatrix} \cos(\beta/2) & \sin(\beta/2) \\ -\sin(\beta/2) & \cos(\beta/2) \end{bmatrix}. \end{aligned}$$

**Dimostrazione** Dato

$$R_z(\alpha)R_y(\beta)R_z(\gamma) = \begin{bmatrix} e^{i(\alpha+\gamma)/2} \cos(\beta/2) & e^{i(\alpha-\gamma)/2} \sin(\beta/2) \\ -e^{-i(-\alpha+\gamma)/2} \sin(\beta/2) & e^{-i(\alpha+\gamma)/2} \cos(\beta/2) \end{bmatrix}.$$

Qualunque  $U \in SU(2)$  può essere scritta nella forma

$$U = \begin{bmatrix} a & b \\ -b^* & a^* \end{bmatrix} = \begin{bmatrix} \cos(\theta)e^{i\lambda} & \sin(\theta)e^{i\mu} \\ -\sin(\theta)e^{-i\mu} & \cos(\theta)e^{-i\lambda} \end{bmatrix},$$

dove abbiamo utilizzato il fatto che  $\det U = |a|^2 + |b|^2 = 1$ . Considerando dunque  $U = R_z(\alpha)R_y(\beta)R_z(\gamma)$ , si ottiene:

$$\theta = \frac{\beta}{2}, \lambda = \frac{\alpha + \gamma}{2}, \mu = \frac{\alpha - \gamma}{2}. \quad \square$$

<sup>1</sup>Il gruppo  $SU(2)$  rappresenta il gruppo speciale unitario di ordine 2, ovvero il gruppo delle matrici 2x2 con determinante unitario

**Lemma** Sia  $U \in SU(2)$ . Si dimostra esistere  $A, B, C \in SU(2)$  tali che  $U = AXBXCX$  e  $ABC = \mathbf{1}$ , dove  $X = \sigma_x$ .

**Dimostrazione** Dato il primo di questi due lemmi, è possibile scrivere  $U$  come  $U = R_z(\alpha)R_y(\beta)R_z(\gamma)$  per  $\alpha, \beta, \gamma \in \mathbf{R}$ .

Sia

$$A = R_z(\alpha)R_y\left(\frac{\beta}{2}\right), B = R_y\left(-\frac{\beta}{2}\right)R_z\left(-\frac{\alpha+\gamma}{2}\right), C = R_z\left(-\frac{\alpha-\gamma}{2}\right).$$

allora:

$$\begin{aligned} AXBXC &= R_z(\alpha)R_y\left(\frac{\beta}{2}\right)XR_y\left(-\frac{\beta}{2}\right)R_z\left(-\frac{\alpha+\gamma}{2}\right)XR_z\left(-\frac{\alpha-\gamma}{2}\right) \\ &= R_z(\alpha)R_y\left(\frac{\beta}{2}\right)XR_y\left(-\frac{\beta}{2}\right)X \cdot XR_z\left(-\frac{\alpha+\gamma}{2}\right)X \cdot XR_z\left(-\frac{\alpha-\gamma}{2}\right) \\ &= R_z(\alpha)R_y\left(\frac{\beta}{2}\right)R_y\left(\frac{\beta}{2}\right)R_z\left(\frac{\alpha+\gamma}{2}\right)R_z\left(-\frac{\alpha-\gamma}{2}\right) \\ &= R_z(\alpha)R_y(\beta)R_z(\gamma) = U. \end{aligned}$$

Si dimostra così inoltre la relazione:

$$\begin{aligned} ABC &= R_z(\alpha)R_y\left(\frac{\beta}{2}\right)R_y\left(-\frac{\beta}{2}\right)R_z\left(-\frac{\alpha+\gamma}{2}\right)R_z\left(-\frac{\alpha-\gamma}{2}\right) \\ &= R_z(\alpha)R_y(0)R_z(-\alpha) \\ &= I. \quad \square \end{aligned}$$

**Teorema (Solovay-Kitaev)** Sia  $G$  un set di istruzioni per  $SU(d)$ , e sia  $\epsilon > 0$ , parametro di accuratezza. Si dimostra esistere una costante  $c$  tale che per ogni  $U \in SU(d)$  esiste una sequenza finista  $S$  di gate appartenenti a  $G$  di lunghezza  $O(\log^c(\frac{1}{\epsilon}))$  tale che  $d(U, S) < \epsilon$ .

La dimostrazione del seguente teorema è altamente formale nonostante ciò risulta funzionale comprenderne a pieno l'importanza in termini di capacità computazionali.

Il teorema asserisce infatti che è sempre possibile codificare un single qubit gate con un set finito di gate universali. Si vede di seguito un esempio notevole riportato all'interno dell' IBM Quantum Documentation [9].

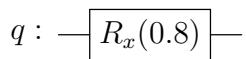
## IBM Quantum Documentation - SolovayKitaev

```

1 import numpy as np
2 from qiskit.circuit import QuantumCircuit
3 from qiskit.transpiler.passes.synthesis import
4     SolovayKitaev
5
6 circuit = QuantumCircuit(1)
7 circuit.rx(0.8, 0)
8
9 print("Original circuit:")
10 print(circuit.draw())
11
12 skd = SolovayKitaev(recursion_degree=2)
13
14 discretized = skd(circuit)
15
16 print("Discretized circuit:")
17 print(discretized.draw())
18
19 print("Error:", np.linalg.norm(Operator(circuit).data -
    Operator(discretized).data))

```

Original circuit:



Discretized circuit:

global phase:  $\frac{7\pi}{8}$



Error: 2.828408279166474

### 2.1.2 Variational Quantum Algorithms

I variational quantum algorithms (VQA) vengono implementati per l'ottimizzazione di un circuito quantistico mediante un ottimizzatore classico. L'obiettivo di un VQA è quello di trovare il modello che meglio approssimi i risultati sperimentali, ovvero, ricerca i parametri in grado di descrivere al meglio i dati in analisi. Un VQA è scritto seguendo i seguenti passaggi: costruzione di una cost function, costruzione di un ansatz e costruzione di una funzione per la ricalibrazione dei parametri.

Tra i principali VQA si annoverano il *variational quantum eigensolver* (VQE), il *quantum approximation optimization algorithm* (QAOA) ed il *quantum machine learning* (QML).

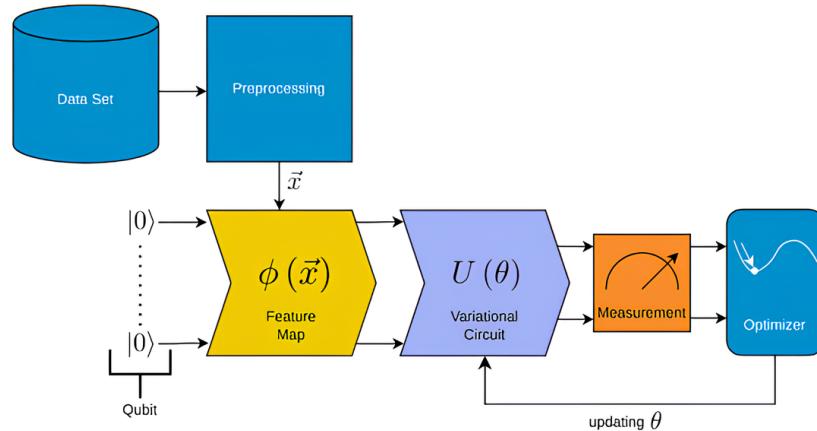


Figura 2.1: Schema di un Variational Quantum Algorithm

**Variational Quantum Eigensolver** Il variational quantum eigensolver è un algoritmo quantistico, che sfrutta un ottimizzatore classico per ottimizzare una serie di parametri associati ad un circuito quantistico, impiegato nella risoluzione di problemi agli autovalori. Il VQE trova ampia applicazione nello studio delle energie di ground e delle energie associate a stati debolmente eccitati.

*Una trattazione più approfondita di questo algoritmo variazionale è da ritrovarsi nei paragrafi successivi di questo capitolo.*

**Cost function** Uno degli aspetti fondamentali di una VQA è quello di codificare il problema in una cost function. In termini astratti, la cost function rappresenta un' iper-superficie che prende il nome di cost landscape dove il compito dell'ottimizzatore diviene quello di trovare il minimo globale.

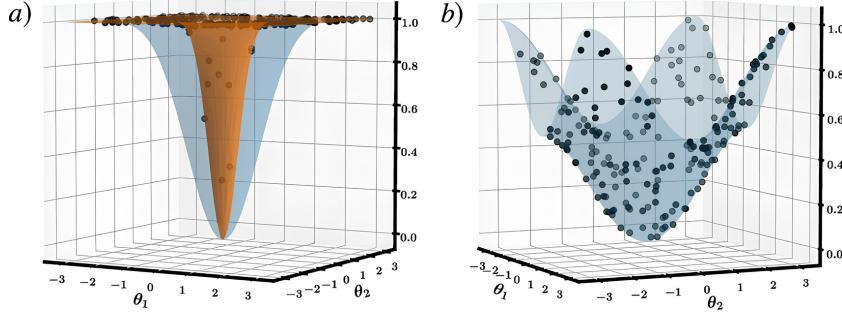


Figura 2.2: Grafici rappresentati l'andamento di due possibili Cost function

Senza perdita di generalità si può esprimere una cost function come

$$C(\theta) = f(\{\rho_k\}, \{O_k\}, \{U_k\})$$

dove  $f$  rappresenta una generica funzione,  $U(\theta)$  è una matrice unitaria parametrizzata da  $\theta$  che rappresenta un'insieme di parametri discreti e continui,  $\{\rho_k\}$  rappresentano gli stati in input ed  $\{O_k\}$  rappresenta l'insieme di tutte le osservabili del sistema.

Si noti come è spesso utile esprimere la cost function come

$$C(\theta) = \sum_k f_k(Tr[O_k U(\theta) \rho_k U^\dagger(\theta)]).$$

Una cost function deve altresì rispettare una serie di criteri, primo fra tutti è necessario che il minimo globale della funzione corrisponda alla soluzione del problema. In secondo luogo, è necessario che sia efficacemente computabile l'insieme dei valori di  $C(\theta)$  eseguendo misurazioni su un computer quantistico eventualmente susseguito da un classical post-processing. Infine la cost function deve essere "trainable", il che vuol dire che possano essere efficacemente ottimizzati i parametri  $\theta$ .

Nell'ambito della VQE la cost function viene definita in maniera differente a seconda del subspace, che può essere pesato oppure non-pesato. Per un subspace

pesato la cost function risulta avere la seguente forma analitica

$$C(\theta) = \sum_{i=0}^m w_i \langle \phi_i | U(\theta)^\dagger H U(\theta) | \phi_i \rangle$$

dove i pesi sono tali che  $w_0 > w_1 > \dots w_m$ , mentre per un subspace non pesato (che vedremo essere quello adottato in questo studio) la cost function risulta essere

$$C(\theta) = \sum_{i=0}^m \langle \phi_i | U(\theta)^\dagger H U(\theta) | \phi_i \rangle$$

**Ansatz** L'ansatz costituisce l'architettura fondamentale di un circuito quantistico. Esso è costituito da un set di gate applicati ad uno specifico sottosistema. Dal punto di vista concettuale, un ansatz costituisce una serie di assunzioni che si fanno precedentemente all'esecuzione completa del circuito il cui scopo è quello di approssimare con maggior precisione il sistema in che si sta studiando.

La corretta scelta di un ansatz risulta essere di fondamentale importanza per ottenere una soluzione quanto più prossima allo stato che si vuole descrivere.

**Ottimizzatori ed ottimizzazione numerica** Lo studio dei metodi di ottimizzazione numerica si rifà ad una serie di tecniche la cui validità risulta spesso essere caso-specifica. In variational hybrid quantum-classical algorithms, come la VQE ci si limita a metodi in cui definita una funzione  $f : \Omega \in R^n \rightarrow R$  sul dominio  $\Omega$ , caratterizzata da un estremo inferiore e superiore nel dominio, si considerano non note informazioni sui valori assunti dal gradiente o dalle derivate ma ci si limita alla sola valutazione della funzione in ogni punto del dominio.

### Criteri di selezione per un ottimizzatore

1. Abilità di trovare una buona soluzione in presenza di rumore<sup>2</sup>, potenzialmente in grado di utilizzare metodi d'ottimizzazione differenti in funzione del tipo di rumori presenti

---

<sup>2</sup>Col termine rumore ci si riferisce all'insieme di tutti quegli effetti che possono portare ad un errore durante un processo di computazione quantistica. A differenza della computazione classica, dove l'errore comporta una variazione randomica dei dati, nella computazione quantistica l'errore ha una natura più complessa e diverse sono le tecniche messe in uso per andare a mitigare i vari tipi di errore.

2. Capacità di variare i tempi computazionali in funzione del numero di parametri

**COBYLA** Nell'ambito del seguente lavoro di tesi si è sfruttato il metodo d'ottimizzazione COBYLA (Constrained optimization by linear approximation), quest'ultimo è un metodo di ottimizzazione numerica impiegato in problemi caratterizzati da soluzioni al contorno in cui le derivate della funzione in analisi si assumono non note, nello specifico questo metodo d'ottimizzazione consiste nell'approssimare iterativamente il problema con condizioni al contorno con problemi di tipo linear programming (LP).

### 2.1.3 Qiskit ed IBM

Qiskit [10] è un framework che consente di progettare e testare i propri circuiti su un simulatore, eventualmente installato su macchina locale, oppure su un computer quantistico reale (simulatori e hardware prendono il nome di backends). Nello studio eseguito, sono stati sfruttati sia simulatori che computer quantici messi a disposizione dall'IBM, azienda che si è inoltre occupata dello sviluppo di qiskit. In tabella è riportato l'insieme di simulatori e computer utilizzati in questa tesi:

Nome	Numero di qubits	Processore	Architettura
●ibm_osaka	127	Eagle r3	
●ibm_nazca	127	Eagle r2	
●FakeManila	5	- - -	
●FakeLagos	7	- - -	

● Computer quantico ● Simulatore

**Quantum error correction (QEC)** I quantum hardware di cui disponiamo oggi risultano essere significativamente affetti da errori computazionali spesso dovuti alle interazioni che i sistemi di qubit possono avere con l'ambiente circostante. Sebbene il tipo di interazione che i qubit possono avere è specifica per ogni sistema, la presenza di errori è un qualcosa che accomuna tutti gli hardware.

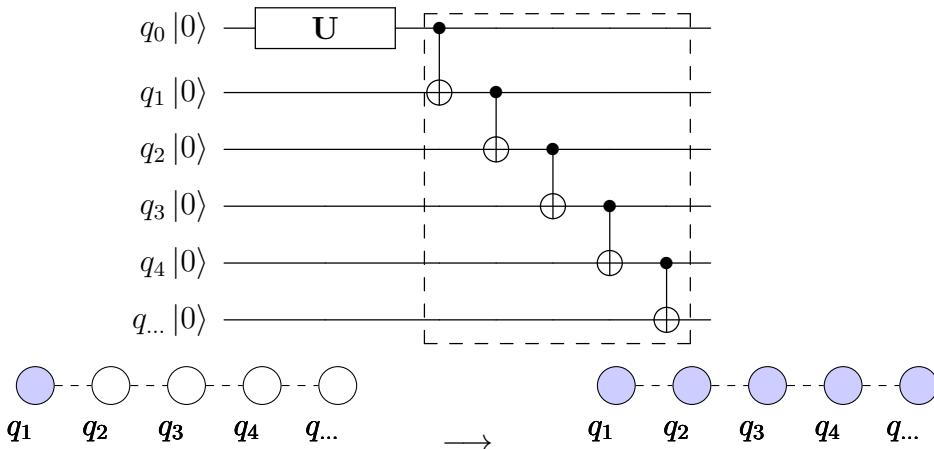
Sono molteplici le tecniche di mitigazione dell'errore che sono state proposte nel corso degli anni. Di seguito si affronta lo studio di una delle tecniche di computazione classica per la mitigazione degli errori più semplice: Repetition codes.

### Repetition codes

I Repetition codes fanno leva sulla ridondanza dell'informazione al fine di tutelarla dagli effetti del rumore esterno. Poichè uno degli errori più comuni che coinvolgono i qubit di quantum hardware superconduttori è lo spin-flip di un qubit quello che si presenta di seguito è un codice finalizzato alla correzione di spin-flip indesiderati. Si assume esistere un qubit (root qubit) nello stato  $a|0\rangle + b|1\rangle$ . Attraverso altri  $n$  qubit nello stato  $|0\rangle$ , è possibile eseguire il seguente mapping:

$$(a|0\rangle + b|1\rangle)|0\dots 0\rangle \rightarrow a|0\dots 0\rangle + b|0\dots 0\rangle.$$

Questa operazione può essere eseguita implementando per esempio  $n$  volte il CNOT gate, come riportato.

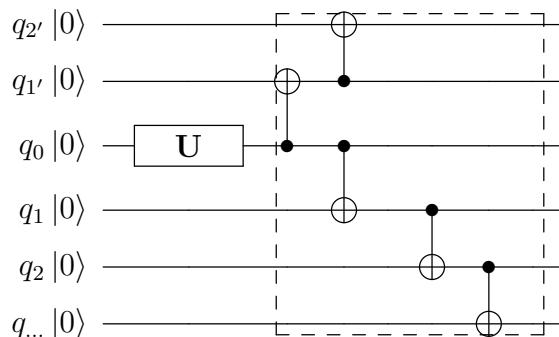


Formalmente questa operazione è generata da  $n$  stabilizzatori  $\{Z_0, Z_1, Z_2, \dots, Z_{n-1}, Z_n\}$  dove  $Z_i$  rappresenta la matrice di Pauli  $\sigma_z$  applicata all'  $i$ -esimo qubit. Si noti che

lo stato  $a|0\dots0\rangle + b|1\dots1\rangle$  non è equivalente ad  $n+1$  copie dello stato  $(a|0\rangle + b|1\rangle)$  poichè questo andrebbe a violare il teorema di non clonazione ma solamente gli autovalori associati all'operatore  $Z$  del qubit originario sono immagazzinati molteplici volte. Di conseguenza, solamente errori dovuti allo spin-flip possono essere individuati ed eventualmente corretti.

Si noti che circuiti la cui espressione logica è quella appena descritta vengono applicati immediatamente prima alla fase di misura.

Questa tecnica di correzione dell'errore, per come è stata correntemente introdotta, potrebbe eventualmente comportare una propagazione dell'errore associato ad un qubit a tutti i livelli sottostanti. Al fine di ovviare a questo problema si sviluppa l'approccio split encoding nel quale una sequanza di qubit viene suddivisa in più sottosequenze dando origine a strutture complesse che possiamo altresì ritrovare nelle architetture dei computer quantici dell'IBM.



## 2.2 Pulse Based VQA

Come precedentemente introdotto, uno degli approcci più comunemente usati nell'implementazione di VQA è quello gate-based, ovvero mediante l'utilizzo di quantum gate. Nello specifico, ci si concentra su un desing gate-based di uno specifico circuito quantistico, l'ansatz, che ricordiamo essere la prima approssimazione dello stato che crediamo meglio approssimare il sistema nel suo livello fondamentale. Questo tipo di approccio risulta essere però non sempre efficiente a causa dell'insieme di ripetizioni che possono presentarsi nell'espressione del circuito e dell'inefficienza legata ai limiti di esecuzione di algoritmi quantistici particolarmente complessi su hardware rumoroso. Al fine di ovviare questo tipo di problema vengono introdotti i Parametrized Quantum Pulse Circuit, la cui implementazione risulta differire in base al tipo di problema affrontato.

Al fine di comprendere al meglio in cosa consista l'implementazione di una serie di operazione attraverso gli impulsi, ci si vuole anzitutto ricondurre a qiskit ed alla tecnologia con cui sono progettati i qubits forniti dall'azienda IBM.

**Programmazione con impulsi attraverso qiskit** Una delle caratteristiche principali di qiskit è la possibilità che fornisce agli utenti di strutturare le operazioni che si desiderano eseguire attraverso l'implementazione di una serie di quantum gate oppure di un circuito di impulsi. Questo tipo di programmazione è permessa poichè quando si utilizzano qubit superconduttori, il sistema esegue le operazioni sui qubit inviando impulsi nelle microonde ai qubit stessi.

### circuit Quantum ElectroDynamics, Josephson Junction e Trasmon qubit

La cQED(circuit Quantum ElectroDynamics) si occupa di descrivere l'interazione della luce, tipicamente a frequenze delle microonde, con la materia costituita da elementi circuituali superconduttori.

L'oscillatore armonico è spesso utilizzato come base per formulare modelli quantistici di sistemi più complessi. Nell'ambito della cQED, la manifestazione fisica di un oscillatore armocino è un circuito LC che si costituisce di un induttore ed un condensatore. Questo semplice circuito può essere portato a mostrare un comportamento puramente quantistico se costruito con materiali supercondutti-

vi e mantenuto a temperature sufficientemente basse. Questo porta i circuiti LC superconduttori a raggiungere una conduzione senza perdite e con massima soppressione del rumore termico. Questo dispositivo gioca un ruolo fondamentale nei dispositivi cQED. Un resonator superconduttivo è semplicemente un insieme di oscillatori quantistici LC, dove ogni modo di oscillazione del resonator corrisponde ad una specifica frequenza di risonanza. In questi termini è possibile descrivere l'intera dinamica di un resonator superconduttore attraverso  $\phi$  che è il risultato della quantizzazione del flusso magnetico che passa attraverso l'induttore, da cui si deriva che il flusso magnetico totale è  $\Phi = \Phi_0\phi$  dove  $\Phi_0 = \hbar/2e$  è il "reduced magnetic flux quantum" ed  $n$  che rappresenta la differenza nel numero di cariche tra i due piatti del condensatore.

Si noti valere la relazione di commutazione:

$$[\phi, n] = i$$

In questi termini l'Hamiltoniana dell'oscillatore superconduttivo LC è data da:

$$H = \frac{E_L}{2}\phi^2 + 4E_Cn^2 \quad (2.1)$$

dove  $E_L = \frac{\Phi_0^2}{L}$  e  $E_C = \frac{e^2}{2C}$ .

Alternativamente è possibile esprimere l'Hamiltoniana utilizzando i parametri  $\omega$  per la frequenza e  $Z$  l'impedenza. Si ottiene dunque che, dato  $\omega = \frac{\sqrt{8E_L E_C}}{\hbar} = \frac{1}{\sqrt{LC}}$  e  $Z_0 = \sqrt{\frac{L}{C}}$ , si ha:

$$H = \hbar\omega \left[ \left( \frac{R_Q}{2Z_0} \right) \phi^2 + \left( \frac{Z_0}{2R_Q} \right) n^2 \right]. \quad (2.2)$$

I risonatori superconduttori da soli non costituiscono un utile mezzo per implementare informazione quantistica. Questo a causa del fatto che i livelli energetici di un risonatore sono equamente distanziati di un valore pari a  $\hbar\omega$ . Per questo motivo vi è la necessità di introdurre un elemento non lineare al fine di ottenere miglior controllo della fisica quantistica del circuito. Nell'ambito della cQED la non linearità è ottenuta attraverso la così detta Josephson Junction (JJ) che viene prediletta per la sua natura non dispersiva. La JJ è un elemento circuitale costituito da due elettrodi superconduttori ed una barriera isolante (che sono rappresentate dalle zone A,B e C in figura 2.4), nello specifico questi sono costruiti

sovrapponendo due stati di film superconduttori separati da uno strato di materiale ossidato. In figura è rappresentato uno schema dei differenti risultati ottenuti con i circuiti precedentemente descritti.

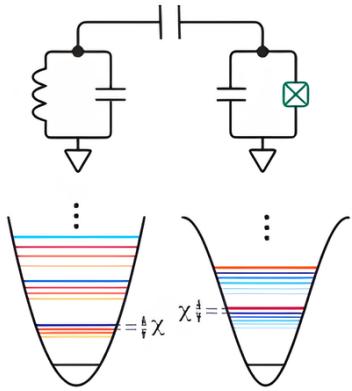


Figura 2.3: LC vs JJ

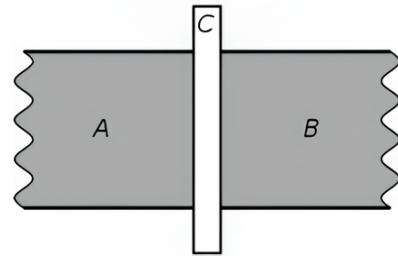


Figura 2.4: Giunzione Josephson

Iniziamo considerando ogni lato della JJ come un’isola superconduttriva contenente un certo numero di coppie di Cooper  $n$ , che è la stessa osservabile utilizzata nella precedente caratterizzazione dei circuiti LC. Come le coppie di Cooper fanno effetto tunnel attraverso la giunzione,  $n$  varia di conseguenza. Ne segue che l’Hamiltoniana di una Josephson junction può essere scritta nella base delle cariche come:

$$H_{J/\hbar} = \frac{E_J}{2} \sum_{n=-\infty}^{\infty} (|n\rangle\langle n+1| + |n+1\rangle\langle n|)$$

dove  $E_J$  è la Josephson energy, che è tradizionalmente espressa in unità di frequenza diviso  $\hbar$ .

**Trasmon** A questo punto si posseggono tutti gli elementi necessari per la costruzione di un circuito in grado di codificare efficacemente informazione espressa in quantum bits. Gli elementi non lineari sono meglio conosciuti come qubit superconduttori. Nella fattispecie, i trasmon qubit, che sono deboli oscillatori anarmonici, rappresentano gli elementi maggiormente utilizzati nei dispositivi cQED. Le frequenze di transizione tra lo stato  $|0\rangle$  e lo stato  $|1\rangle$  dei trasmon sono di solito

intorno ai 5GHz.

La programmazione con impulsi offre una maggiore versatilità rispetto ad una programmazione gate-level ma risente in termini di precisione. Nel circuito in figura

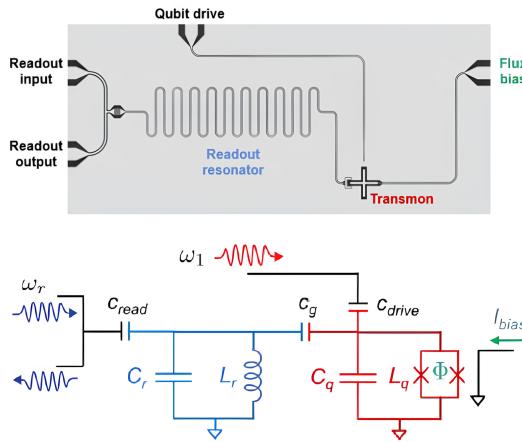


Figura 2.5: Circuito annesso ad un transom

ra, gli impulsi microonde sono implementati attraverso la qubit drive line mentre attraverso la flux bias è possibile regolare dinamicamente la frequenza d'operazione del qubit. Lo stato del transmon viene invece monitorato attraverso degli impulsi sonda nelle microonde applicati attraverso le porte di readout input e ricevuti nelle porte di readout output. Si noti come in vero la precisione sia cruciale per algoritmi del tipo non-variazionali poiché richiedono rotazioni ed operazioni esatte, nonostante ciò, per algoritmi variazionali l'obiettivo è quello di approssimare gli stati e non è richiesta la necessità di ottenere risultati perfettamente accurati.

Si noti inoltre che il passaggio da un tipo di programmazione gate-based ad un tipo di programmazione pulse-based può avvenire in maniera differente a seconda del tipo di operazioni, è possibile ad esempio associare ad una rotazione un numero variabile di impulsi. Vedremo come ciò sarà rilevante nella scelta e nel numero di parametri associati agli ansatz proposti nella tesi.

A fronte di ciò è possibile riprendere con la trattazione sull'implementazione di VQA attraverso programmi di impulsi.

Nell'ambito del seguente elaborato si sfrutta la programmazione pulse-based per

definire l'ansatz, nella fattispecie il tipo di ansatz che si è deciso di adottare prende il nome di Hardware Efficient.

### 2.2.1 Hardware Efficient Ansatz

Si consideri un generico circuito quantistico

$$U(\theta) = \prod_l e^{-i\theta_l H_l} V_l, \quad (2.3)$$

l'Hardware Efficient Ansatz (HEA) è il nome associato a tutti quegli ansatz che si concentrano sulla riduzione della depth (ovvero in numero di gates) del circuito scegliendo opportuni gates  $V_l$  e generatori  $H_l$  a seconda del tipo di connessioni ed interazioni dello specifico computer quantico utilizzato.

**Effetto di barren plateau** Come ampiamente trattato nella letteratura riguardante la non-trainability dei VQAs, è facile incorrere nel fenomeno di barren plateau, fenomeno in cui la funzione obiettivo ha gradienti estremamente piccoli od addirittura nulli su gran parte dello spazio dei parametri<sup>3</sup>. Nello specifico ci si trova difronte ad un fenomeno di barren plateau della cost function se quest'ultima diviene esponenzialmente piatta all'aumentare del numero di qubit, da qui la necessità di introdurre un ansatz in grado di essere implementato sfruttando un numero maggiormente ridotto di qubit.

Si consideri nello specifico la cost function  $f(\theta)$ , durante il fenomeno di barren plateau è possibile trovarsi in una delle due seguenti codizioni: deterministic concentration<sup>4</sup> (tutto il landscape risulta esser piatto) o probabilistic concentration (la maggior parte del landscape è piatto).

**Deterministic concentration** Sia il valore della cost function pari a  $f_{trv} := \frac{1}{2^n} Tr[O]$ . Allora la funzione  $f(\theta)$  è  $\epsilon$ -limitata se e solo se esiste un  $\epsilon$  positivo tale che per ogni  $\theta$  vale che

$$|f(\theta) - f_{trv}| \leq \epsilon,$$

dove si definisce con  $Tr[U(\theta), |\psi\rangle]$  la più generica cost function.

---

<sup>3</sup>Gradienti molto piccoli rendono estremamente difficile l'addestramento del modello in quanto il processo di ottimizzazione richiede gradienti significativi per convergere verso un minimo globale o un punto di minimo locale

<sup>4</sup>Col termine concentration ci si riferisce alla piattezza

**Probabilistic concentration** Sia  $\langle \cdot \rangle$  la media in funzione dei parametri  $\theta_1 \in \Theta_1, \theta_2 \in \Theta_2, \dots$ , nei rispettivi domini di definizione  $\{\Theta\}$ . Allora la funzione  $f(\theta)$  si definisce probabilistic centratrated se per ogni  $\theta'$  vale

$$E_\theta |f(\theta) - f(\theta')| \leq \epsilon$$

dove la media è presa sui domini  $\{\Theta\}$ .

La scelta di questo tipo di ansatz viene giustificata dai risultati esposti nella pubblicazione **On the practical usefulness of the Hardware Efficient Ansatz** di Lorenzo Leone et al. [11] dove (sebbene come da loro riportato la seguente conclusione richiederebbe analisi maggiormente approfondate) si mostra che la scelta di un ansatz di questo tipo sia effettivamente vantaggiosa in alcuni regimi sia in termini di correzione dell'effetto barren plateau che in termini di costi computazionali.

**HEA con qiskit** Qiskit permette l'implementazione di una moltitudine di hardware efficient ansatz, tra questi si annoverano il qUCSSD (la cui struttura nasce dall'analisi del diretto problema elettronico), l'EfficientSU2 (appartenente alla categoria dei TwoLocal) ecc. Nell'ambito del seguente studio però ci si è concentrati sulla scrittura di un HEA attraverso l'utilizzo di un programma di impulsi. Prima di soffermarci su quest'ultimo, si vedono brevemente gli ansatz sopracitati.

**qUCCSD** Il seguente anstaz deriva unitary coupled-cluster theory (UCC) dove la funzione d'onda  $|\psi\rangle$  viene scritta come prodotto della funzione d'onda di Hartree-Fock (HF) per un operatore che prende il nome di operatore di cluster che racchiude l'informazione sugli stati d'eccitazione roto-vibrazionali

$$|\psi\rangle = e^{\hat{T}} |\psi_{HF}\rangle$$

dove l'operatore di cluster è esprimibile come

$$T(\vec{\theta}) = \sum_{p,q}^{p \in vir, q \in occ} \theta_q^p \hat{T}_q^p + \sum_{p>q, r>s}^{p,q \in vir, r,s \in occ} \theta_{rs}^{pq} \hat{T}_{rs}^{qp}$$

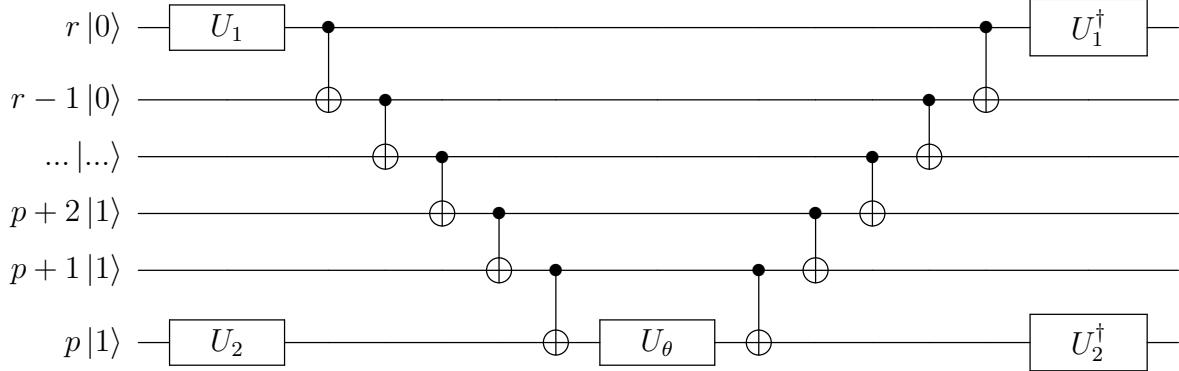
dove con il primo termine si considerano i contributi di singola eccitazione e con il secondo i contributi di doppia eccitazione (si noti che sono inoltre compresi tutti

i processi di diseccitazione).

L'ansatz qUCCSD risulta dunque essere:

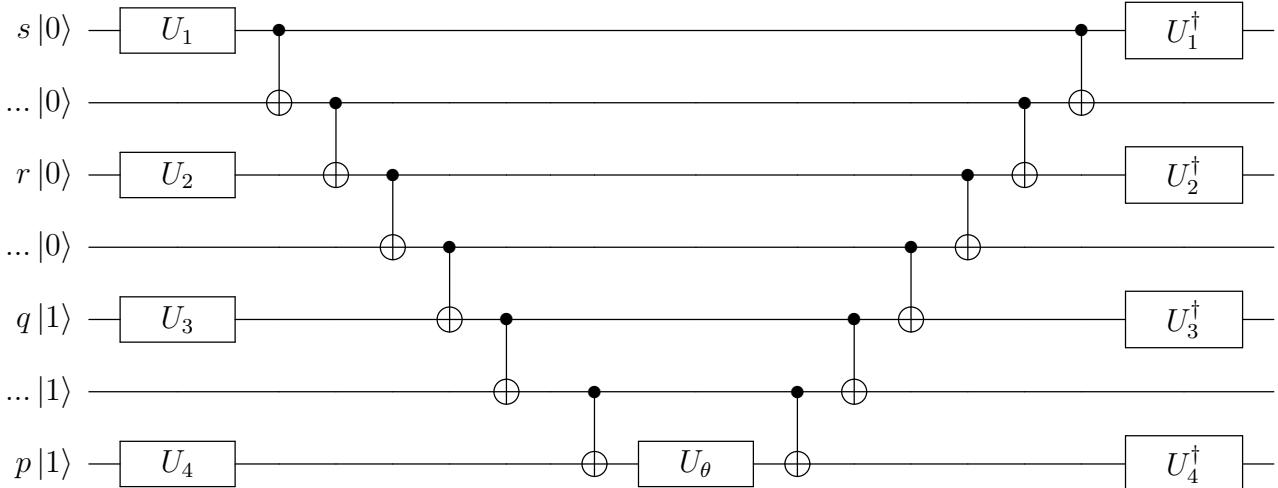
$$|\psi\rangle = e^{\hat{T} - \hat{T}^\dagger} |\psi_{HF}\rangle$$

I circuiti per l'implementazione di una funzione d'onda con qUCCSD sono costruiti come segue:



dove

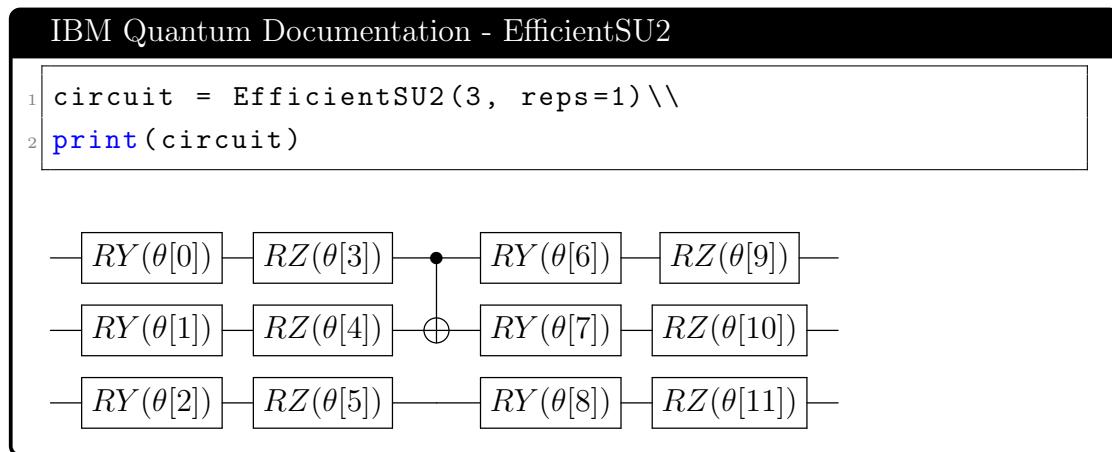
$$(U_1, U_2) = \{(Y, H), (H, Y)\}; Y = R_x(-\frac{\pi}{2})$$



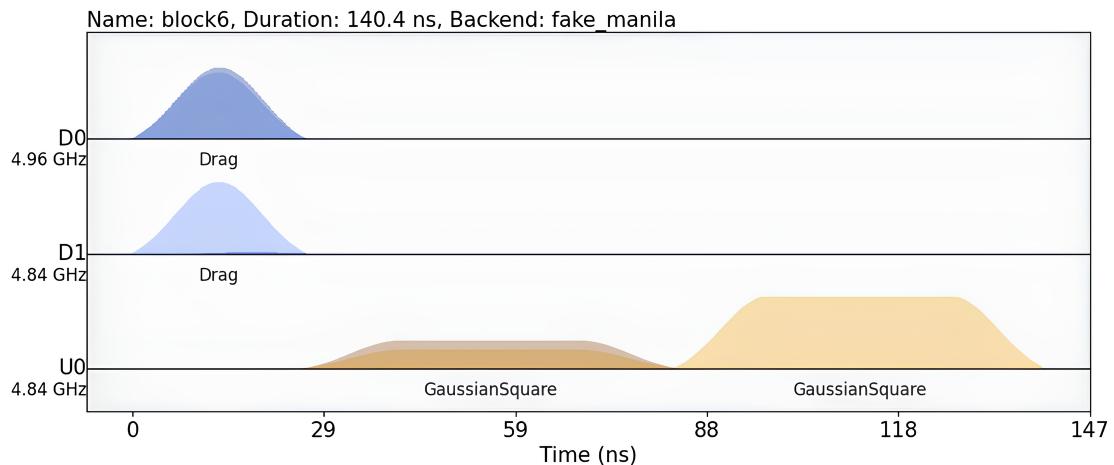
dove:  $(U_1, U_2, U_3, U_4) = \{(H, H, Y, H), (Y, H, Y, Y), (H, Y, Y, Y), (H, H, H, Y), (Y, H, H, H), (H, Y, H, H), (Y, Y, Y, H), (Y, Y, H, Y)\}$ .

**EfficientSU2** Il circuito EfficientSU2 consiste in una serie di operazioni a singolo qubit implementate con matrici SU(2) ed entanglement implementato attraverso il quantum gate CX.

Un esempio di ansatz EfficientSU(2) implementato su 3 qubit potrebbe essere il seguente:



**HEA pulse based** Nell’ambito del presente elaborato, la funzione d’onda associata al sistema in analisi viene implementata all’interno del codice attraverso un programma di impulsi (la cui descrizione dettagliata sarà parte d’interesse del terzo capitolo) in grado quest’ultimo di essere eventualmente rappresentato con un circuito quantistico. Lo schema ad impulsi riportato fornisce un’anticipazione del



tipo di schemi che si presenteranno nel capitolo successivo, essendo infatti esso uno tra gli ansatz impiegati per la descrizione di una molecola di idrogeno attraverso un sistema a 2 qubit. Si noti che, come precedentemente anticipato nella trattazione dei trasmoni, le frequenze di transizione dei qubit si aggirano intorno ai 5GHz. Dal

punto di vista elettronico, un impulso gaussiano introdotto all'interno del sistema risulta avere la seguente forma funzionale:

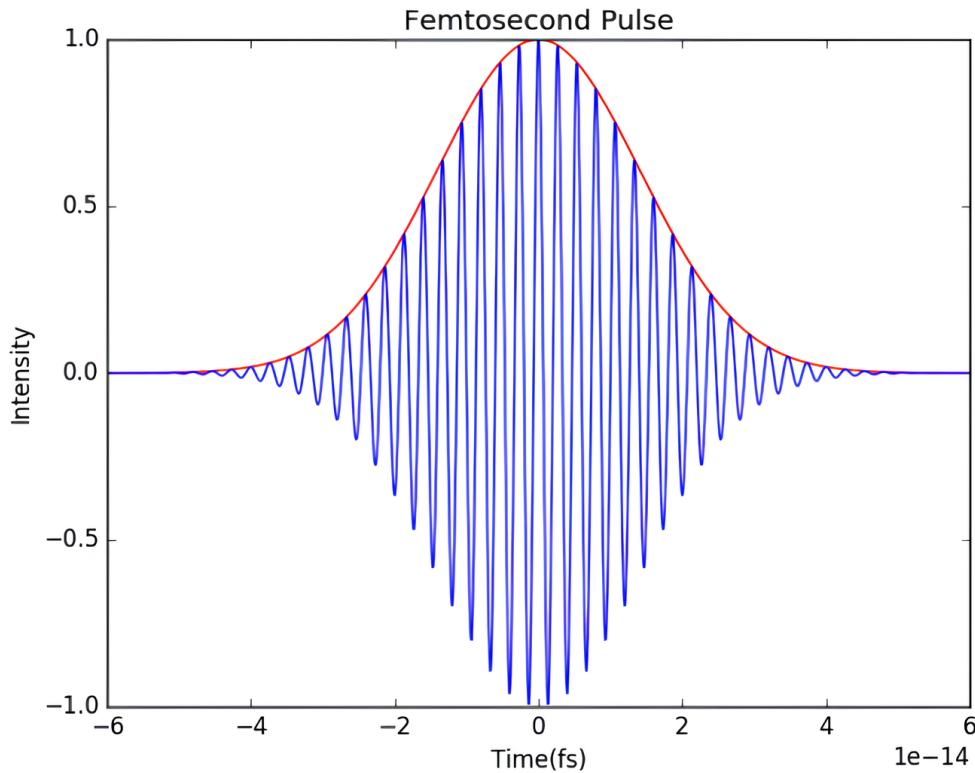


Figura 2.6: Forma funzionale di un impulso Gaussiano

In generale la tensione viene implementata con la forma funzionale:

$$V(t) = V_0 \epsilon(t) \cos(\omega t - \phi) \quad (2.4)$$

dove  $\epsilon(t)$  definisce il profilo della funzione e  $\phi$  è il fattore di fase.



# Capitolo 3

## Analisi Computazionale

### 3.1 Introduzione al codice

L'oggetto della tesi risulta essere lo studio delle energie dello stato fondamentale di alcune semplici molecole quali l'idrogeno molecolare ( $H_2$ ) e l'idruro di litio ( $LiH$ ) implementando l'ansatz attraverso un programma di impulsi. L'obiettivo sarà quello di ottenere l'andamento dell'energia di ground in funzione della distanza tra gli atomi costituenti la molecola. Ciò che ci si aspetta dall'analisi computazionale è di trovare risultati confrontabili con i valori teorici a partire da ansatz di impulsi di dimensioni inferiori agli ansatz gate-based e di osservare inoltre rispetto a questi tempi computazionali inferiori.

In termini di programmazione, l'implementazione di sistemi molecolari è permessa attraverso la classe di qiskit PySCFDriver:

```
1 ultra_simplified_ala_string = f"""
2     H 0.0 0.0 0.0
3     H 0.0 0.0 {dist}
4     """
5
6     driver = PySCFDriver(
7         atom=ultra_simplified_ala_string.strip(),
8         basis='sto3g',
9         unit=DistanceUnit.ANGSTROM
10    )
```

```
11 qmolecule = driver.run()
```

Nel seguente codice si è implementata una molecola di  $H_2$  definendo due atomi di idrogeno posti ad una distanza  $dist$  e rappresentando il sistema nella base sto3g. Si procede dunque col definire il problema elettronico e col mapparlo in termini di operatori di Pauli.

```
1 num_active_electrons = 2
2 num_active_orbitals = 2
3 as_transformer = ActiveSpaceTransformer(num_active_electrons,
   num_active_orbitals)
4 problem = as_transformer.transform(qmolecule)
5 mapper = ParityMapper(num_particles = problem.num_particles)
6 qubit_op = mapper.map(problem.second_q_ops() [0])
```

Nel codice viene presentato il numero di elettroni ed il numero di orbitali associati al problema elettronico (*problem*) e quest'ultimo viene espresso in termini di operatori di Pauli (*qubit\_op*). Sfruttando la funzione *ActiveSpaceTransformer* si ha la possibilità di escludere dal problema il contributo dovuto alla presenza di elettroni secondari, ovvero la simulazione non tiene conto degli orbitali interni (il cui contributo è immagazzinato all'interno di un termine di potenziale) ma viene eseguita solo sugli orbitali di valenza, questo però senza perdere precisione nella descrizione del sistema. Questa scelta (che potrebbe trovar maggior giustificazione per sistemi molecolari complessi) è adottata poichè vantaggiosa in termini di tempi computazionali. Ci si concentra sull'idrogeno molecolare per definire quale sia il miglior mapping<sup>1</sup> da utilizzare in funzione del numero di qubit introdotti nel sistema. I risultati esposti sono stati ottenuti sfruttando il simulatore FakeManila e sono stati successivamente confermati sul quantum computer nazca.

---

<sup>1</sup>Nell'ambito del presente elaborato si introducono (data la natura elettronica del problema) dei mapper per fermioni la cui funzione è quella di codificare in operazioni applicabili a qubit gli operatori fermionici della meccanica quantistica (essendo i qubit sistemi bosonici è importante mantenere le corrette relazioni di commutazione ed anticommutazione).

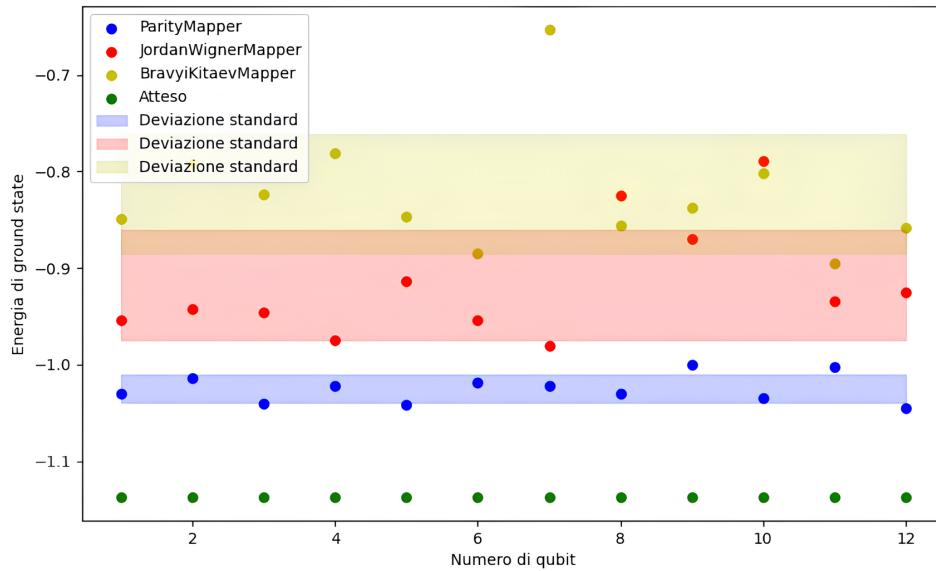


Figura 3.1: Andamenti energetici di una moleola di  $H_2$  sul simulatore FakeManila in presenza di rumore in funzione del numero di qubit e del tipo di mapping

Da questo punto in poi tutte le simulazioni eseguite hanno sfruttato il parity mapping dove l'informazione sulla parità è localmente immagazzinata in un qubit mentre l'informazione sullo stato di occupazione è delocalizzata su tutti i qubit. Lo studio dell'energia associato alle molecole  $H_2$  ed  $LiH$  ha inizio con l'implementazione di un codice in grado di fornire risultati corretti per la stima dell'energia dello stato fondamentale delle seguenti molecole. Successivamente si intraprende uno studio incentrato sulla scelta dei parametri associati all'implementazione dei sistemi molecolari. Nell' immagine che segue si possono vedere a confronto gli andamenti energetici delle molecola di idrogeno molecolare al variare della base utilizzando un *noisy estimator*. La scelta di eseguire le misure sfruttando un *noisy estimator* risiede nel fatto di non conoscere anticipatamente il grado di rumore caratteristico di un qubit fisico, l'analisi ha infatti l'obiettivo di simulare un sistema altamente rumoroso per iniziare a comprendere come poter affrontare una condizione di quel tipo.

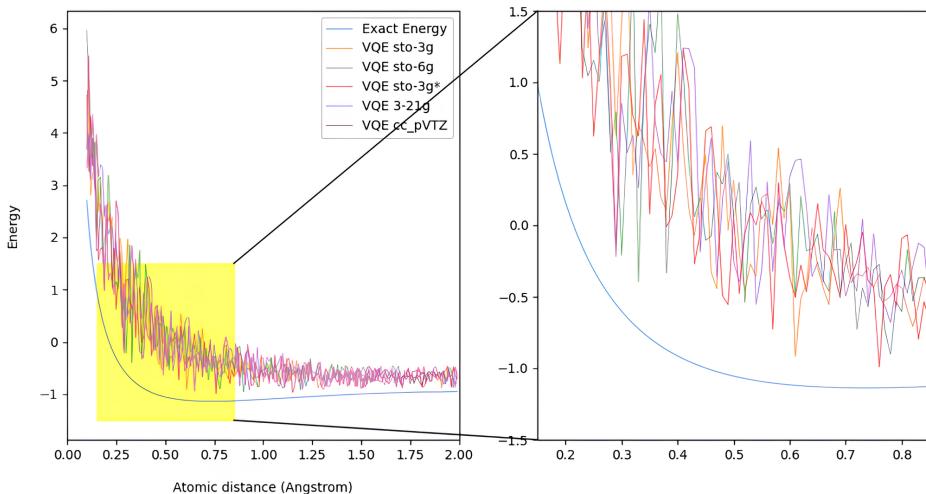


Figura 3.2: Andamenti energetici di una moleola di  $H_2$  al variare della base

Una prima analisi non è in grado di fornire risultati apprezzabili circa la scelta della migliore base da impiegare nella definizione del sistema molecolare. Nonostante ciò, è facilmente dimostrabile come, in assenza di un preponderante rumore, la base che meglio approssima l'andamento teorico risulta essere quella caratterizzata dal maggior numero di dimensioni.

Terminata la stesura di un programma in grado di fornire risultati confrontabili con i risultati accertati dalla comunità scientifica, si procede con l'implementazione dell'ansatz come programma di impulsi e, nel merito della seguente tesi, si introduce un Hardware Efficient ansatz.

```

1 def HE_pulse(backend, amp, angle, width):
2     with pulse.build(backend) as my_program1:
3         sched_list = []
4         with pulse.build(backend) as sched1:
5             qubits = (1, 0)
6             for i in range(2):
7                 pulse.play(drag_pulse(backend, amp[i], angle[
8                     i]), DriveChannel(qubits[i]))
9             sched_list.append(sched1)

```

```

9
10    with pulse.build(backend) as sched2:
11        uchan = pulse.control_channels(0, 1)[0]
12        pulse.play(ecr_pulse(backend, amp[0], angle[2],
13                           width[0]), uchan)
14        sched_list.append(sched2)
15
16    with pulse.build(backend) as sched4:
17        uchan = pulse.control_channels(0, 1)[0]
18        pulse.play(ecr_pulse(backend, amp[1], angle[3],
19                           width[0]), uchan)
20        sched_list.append(sched4)
21
22    with pulse.build(backend) as my_program:
23        with pulse.transpiler_settings(initial_layout=[0,
24                                                       1]):
25            with pulse.align_sequential():
26                for sched in sched_list:
27                    pulse.call(sched)
28
29
30    return my_program

```

Nel merito del seguente codice, l'ansatz viene implementato come una funzione i cui argomenti sono il backend su cui verrà eseguito il codice e tre parametri (ampiezza, angolo e larghezza) i cui valori sono ricavati a partire dai parametri inseriti prima di iniziare la simulazione. Il programma implementa una lista vuota alla quale sono appese una serie di altre liste contenenti gli impulsi a cui i due qubit devono essere sottoposti. Il programma di impulsi si struttura in una serie di impulsi di controllo e DRAG pulses. Il Derivative Removal by Adiabatic Gate (DRAG) pulse è un impulso standard Gaussiano al quale è addizionato un termine di derivata della Gaussiana<sup>2</sup> :

$$\text{DRAG}(t) = A \exp\left(-\frac{(t - \mu)^2}{2\sigma^2}\right) + i\beta \frac{d}{dt} \left[ A \exp\left(-\frac{(t - \mu)^2}{2\sigma^2}\right) \right]$$

---

<sup>2</sup>Il termine aggiunto è immaginario, perciò la shape della DRAG è complessa (al contrario di una gaussiana)

```

1 from qiskit.pulse import Schedule, GaussianSquare, Drag,
2   Delay, Play, ControlChannel, DriveChannel
3
4 def drag_pulse(backend, amp, angle):
5   backend_defaults = backend.defaults()
6   inst_sched_map = backend_defaults.instruction_schedule_map
7   x_pulse = inst_sched_map.get('x', (0)).filter(channels =
8     DriveChannel(0)), instruction_types=[Play]).instructions
9   [0][1].pulse
10  duration_parameter = x_pulse.parameters['duration']
11  sigma_parameter = x_pulse.parameters['sigma']
12  beta_parameter = x_pulse.parameters['beta']
13  pulse1 = Drag(duration=duration_parameter, sigma=
14    sigma_parameter, beta=beta_parameter, amp=amp, angle=
15    angle)
16
17  return pulse1

```

Il seguente codice presenta l’implementazione di un impulso *DRAG* a partire dai parametri associati al gate *X* che generalmente corrisponde ad una rotazione di  $\pi$  attorno all’asse *x* sulla sfera di Bloch.

Per quanto concerne gli impulsi di controllo, essi vengono implementati attraverso un impulso di tipo *ECR* (Echoed Cross-Resonance).

$$\text{ECR} = \begin{pmatrix} \frac{1}{\sqrt{2}} & 0 & 0 & \frac{1}{\sqrt{2}} \\ 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ 0 & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 \\ \frac{1}{\sqrt{2}} & 0 & 0 & -\frac{1}{\sqrt{2}} \end{pmatrix}$$

Si noti che il codice presentato prevede l’implementazione di un sistema a due qubit questo poichè deriva direttamente dal codice per lo studio delle energie associate all’idrogeno molecolare. La ridefinizione per un numero superiore di qubit segue in maniera automatica dalle funzioni riportate.

Si vuole ora presentare il risultato dell’implementazione dell’ansatz attraverso un programma di impulsi, nonostante ciò, la scelta di introdurre il programma di impulsi attraverso uno *ScheduleBlock* a discapito di un più classico *Schedule* presenta come diretto svantaggio l’impossibilità di ottenere un risultato grafico ap-

prezabile come quello presentato nel paragrafo HEA pulse based sebbene abbia il vantaggio di potersi considerare di natura "dinamica", essendo infatti caratterizzato dalla creazione di consequenziali timeslots durante l'esecuzione del programma stesso. In questi termini, il risultato grafico migliore per poter osservare la natura dei programmi di impulsi implementati risulta essere della forma:

```
1 ScheduleBlock(
2     ScheduleBlock(
3         ScheduleBlock(
4             Play(
5                 Drag(
6                     duration=160, sigma=40, beta
7                         = -0.589247519648376, amp=0.0, angle
8                         =0.0
9                 ),
10                DriveChannel(0),
11            ),
12            Play(
13                Drag(
14                    duration=160, sigma=40, beta
15                        = -0.589247519648376, amp=0.0, angle
16                        =0.0
17                ),
18                DriveChannel(1),
19            ),
20            name="block6",
21            transform=AlignLeft(),
22        ),
23        ScheduleBlock(
24            Play(
25                GaussianSquare(duration=256, sigma=64, width
26                    =0, amp=0.0, angle=0.0),
27                ControlChannel(0),
28            ),
29            name="block7",
30            transform=AlignLeft(),
31        ),
32    )
```

```
27     ScheduleBlock(
28         Play(
29             GaussianSquare(duration=256, sigma=64, width
30                 =0, amp=0.0, angle=0.0),
31             ControlChannel(0),
32         ),
33         name="block8",
34         transform=AlignLeft(),
35     ),
36     ScheduleBlock(
37         Play(
38             Drag(
39                 duration=160, sigma=40, beta
40                     =-0.589247519648376, amp=0.0, angle
41                     =0.0
42             ),
43             DriveChannel(0),
44         ),
45         Play(
46             Drag(
47                 duration=160, sigma=40, beta
48                     =-0.589247519648376, amp=0.0, angle
49                     =0.0
50             ),
51             DriveChannel(1),
52         ),
53         name="block6",
54         transform=AlignLeft(),
55     ),
56     ScheduleBlock(
57         Play(
58             GaussianSquare(duration=256, sigma=64, width
59                 =0, amp=0.0, angle=0.0),
60             ControlChannel(0),
61         ),
62         name="block9",
```

```
57         transform=AlignLeft(),
58     ) ,
59     ScheduleBlock(
60         Play(
61             GaussianSquare(duration=256, sigma=64, width
62                 =0, amp=0.0, angle=0.0),
63             ControlChannel(0),
64         ),
65         name="block10",
66         transform=AlignLeft(),
67     ),
68     name="block12",
69     transform=AlignSequential(),
70 ),
71     name="block11",
72     transform=AlignLeft(),
73 )
```

Dove il seguente codice risulta essere il prodotto di una stampa dell'ansatz precedentemente introdotto.

Si procede dunque col presentare i primi risultati computazionali.

## 3.2 Risultati computazionali

La trattazione sui risultati ottenuti dalle simulazioni ha inizio con l'argomentazione delle energie stimate attraverso l'ansatz introdotto nel precedente paragrafo. Le simulazioni sono state eseguite sfruttando due tipologie di backend distinti: simulatori e computer quantistici.

### 3.2.1 Idrogeno molecolare ( $H_2$ )

**Idrogeno molecolare su simulatore (FakeManila) con sistema a due qubit** La stima dei parametri associati all'ansatz risulta esser tanto migliore quanto maggiore è il numero di iterazioni eseguite dall'ottimizzatore classico.

```

1 vqe_res = minimize(vqe, params, args=(my_dict, pulse_backend,
                                         gate_backend, n_qubit,n_shot),method=optimizer,
                                         constraints=LC,options={'rhobeg':0.1,'maxiter':100})

```

Nel seguente codice si implementa il processo di simulazione definendo con il parametro *maxiter* il numero massimo di iterazioni eseguibili dall’ottimizzatore e con il parametro *rholbeg* il passo associato ad ogni iterazione per la stima dei parametri. Di seguito è riportato un primo risultato computazionale, ottenuto attraverso un ciclo di 26 interazioni da parte dell’ottimizzatore COBYLA.

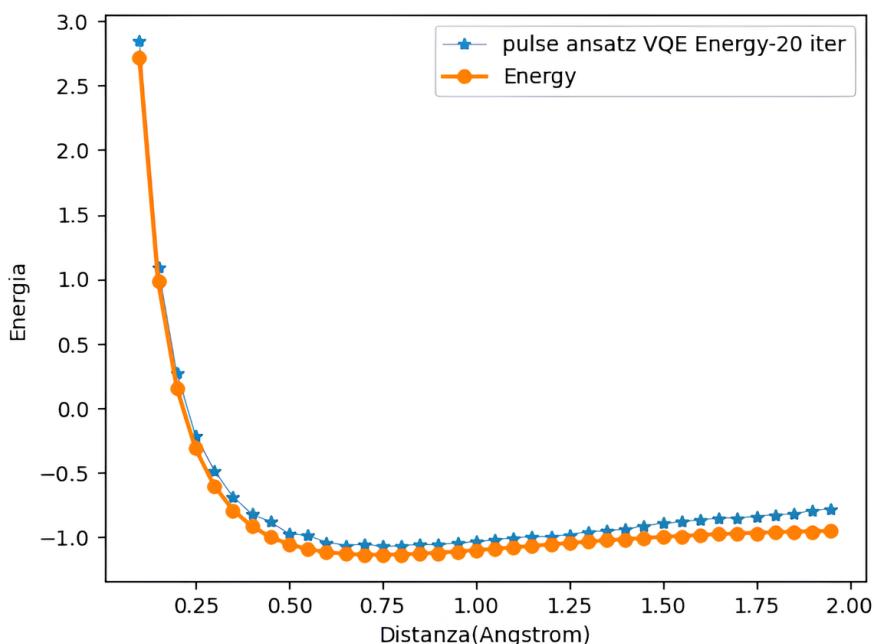


Figura 3.3: Andamenti energetici di una moleola di  $H_2$  al variare della distanza e del numero di iterazioni

Per lo scopo del seguente elaborato si è deciso di condurre nuovamente la simulazione aumentando il numero di iterazioni. I risultati sono riportati nella tabella che segue.

In questa seconda simulazione si è incrementato il numero di iterazioni da 26 a 65. I risultati ottenuti sono i seguenti:

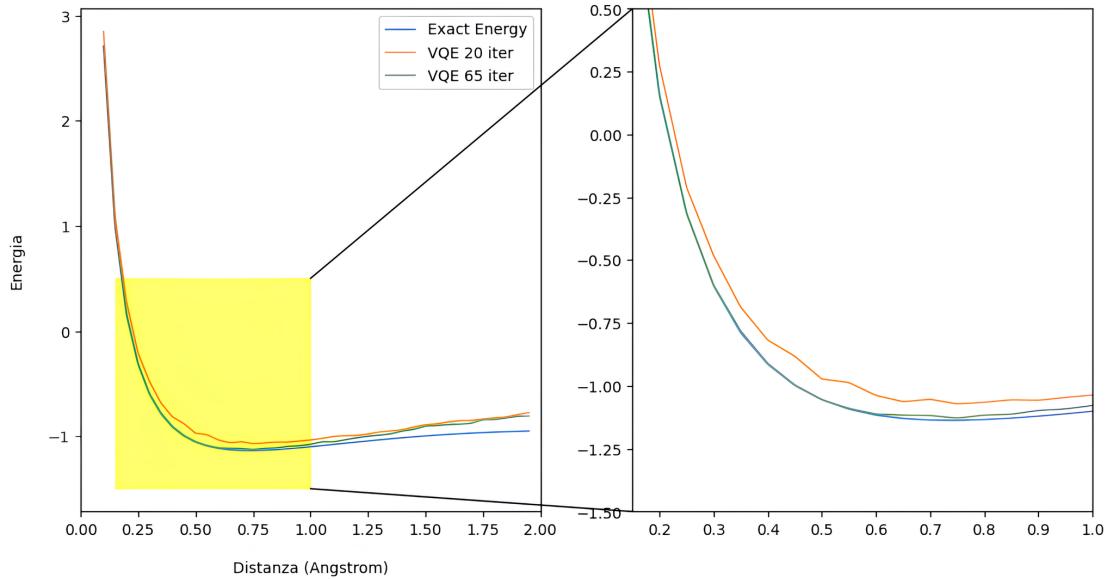


Figura 3.4: Andamenti energetici di una molecola di  $H_2$  al variare della distanza e del numero di iterazioni

Iterazioni	Energia stato fondamentale [ $E_H$ ]
26	-1.072
65	-1.129

Risulta dunque fondamentale andare a comprendere nel dettaglio l’andamento dell’approssimazione dell’energia dello stato fondamentale delle molecole in funzione del numero di iterazioni. Al fine di comprendere la validità del risultato computazionale ottenuto si introducono i risultati ottenuti nello studio *NAPA: Intermediate-level Variational Native-pulse Ansatz for Variational Quantum Algorithms* [12] dove, almeno in un primo momento, si procede ad effettuare un confronto fra dati ottenuti tramite simulatore e dati ottenuti mediante hardware.

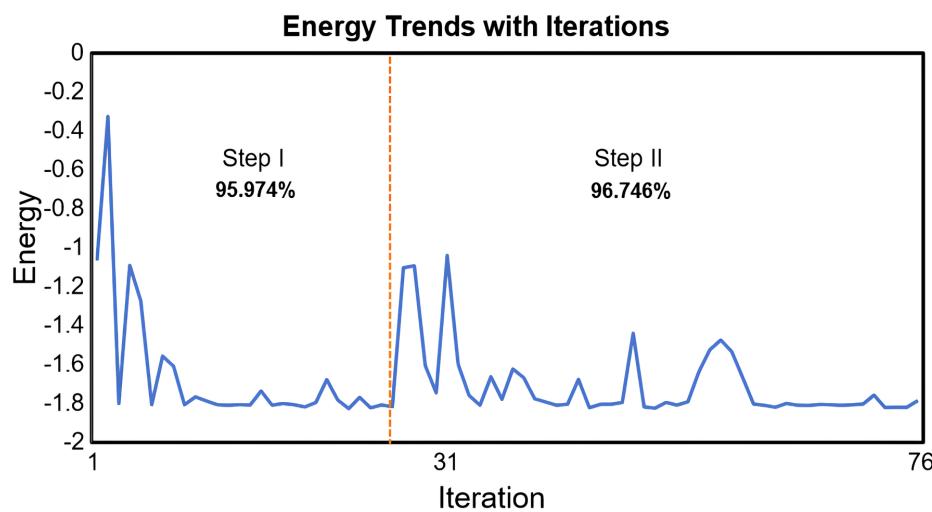


Figura 3.5: Andamento energetico in funzione del numero di iterazioni attraversi *ibmq\_montreal* con ottimizzatore di tipo non-gradient.

La linea verticale al centro separa i passaggi nell'apprendimento progressivo. Possiamo vedere diversi "picchi" sulle curve perché i tentativi dell'ottimizzatore potrebbero essere stati fatti in una "direzione" sbagliata. L'accuratezza raggiunta nel Passo I, che comprende le iterazioni di ottimizzazione dalla uno alla 26, è del 95.974%. Successivamente, il Passo II, che copre le iterazioni dalla 27 alla 76, migliora ulteriormente l'accuratezza al 96.746%.

Si procede nel condurre lo stesso studio utilizzando il simulatore FakeManila. Il risultato atteso dovrebbe risultare analogo a quello presentato nello studio introdotto. Si noti inoltre che la simulazione caratterizzata da un numero di iterazioni pari a 65 è stata condotta impostando il valore *maxiter* a 100, questo comporta che l'ottimizzatore abbia, nei limiti parametrici introdotti nella definizione del problema, ottenuto un valore di convergenza che riusita essere il minimo assoluto della cost function.

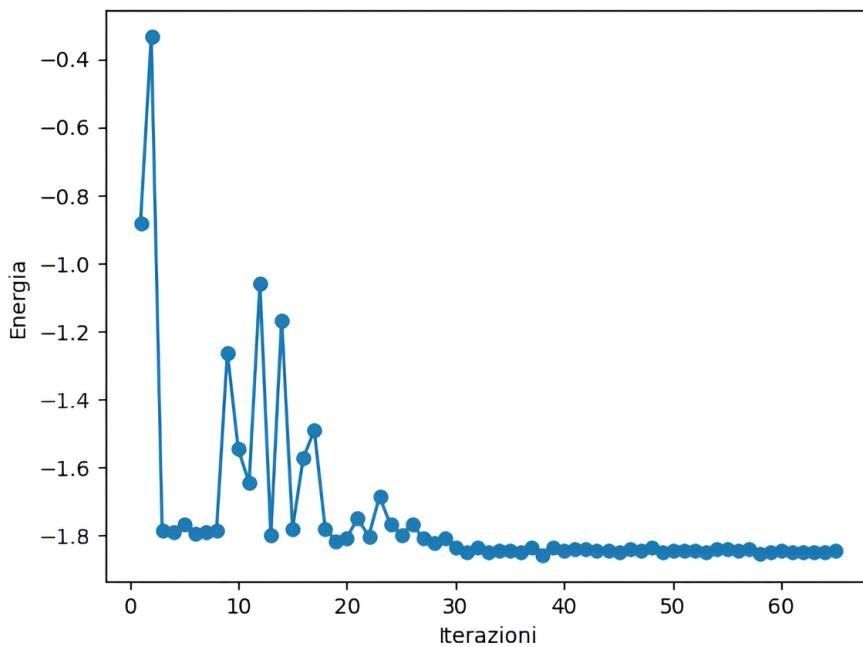


Figura 3.6: Stima dell’energia di ground della molecola  $H_2$  in funzione del numero di iterazioni eseguite dall’ottimizzatore COBYLA.

Seguendo la partizione del numero di iterazioni eseguite dall’ottimizzatore riportata nello studio citato, si riscontrano i seguenti risultati:

Step	Accuratezza
1	$\sim 94,26\%$
2	$\sim 99,26\%$

Questa prima parte dell’analisi computazionale si è svolta implemendando l’ansatz come riportato precedentemente<sup>3</sup>. Si vuole ora implementare un ansatz differente e studiare l’andamento delle simulazioni.

Si definisce il nuovo ansatz come:

```
1 def HE_pulse(backend, amp, angle, width):
```

<sup>3</sup>FakeManila non presenta tra i propri quantum gate il gate *ecr*; per la sua implementazione si è sfruttata la combinazione dei gate *cx* ed *rz*

```

2     with pulse.build(backend) as my_program1:
3         sched_list = []
4         with pulse.build(backend) as sched1:
5             qubits = (0, 1)
6             for i in range(2):
7                 pulse.play(drag_pulse(backend, amp[i], angle[
8                     i]), DriveChannel(qubits[i]))
9             sched_list.append(sched1)
10
11
12         with pulse.build(backend) as sched2:
13             uchan = pulse.control_channels(0, 1)[0]
14             pulse.play(cr_pulse(backend, amp[0], angle[2],
15                         width[0]), uchan)
16             sched_list.append(sched2)
17
18
19         with pulse.build(backend) as sched4:
20             qubits = (0, 1)
21             for i in range(2):
22                 pulse.play(drag_pulse(backend, amp[i], angle[
23                     i]), DriveChannel(qubits[i]))
24             sched_list.append(sched4)
25
26
27         with pulse.build(backend) as my_program:
28             with pulse.transpiler_settings(initial_layout=[0,
29                 1]):
30                 with pulse.align_sequential():
31                     for sched in sched_list:
32                         pulse.call(sched)
33
34
35     return my_program

```

dove si osserva l'introduzione di un nuovo tipo di impulso (cr\_pulse) la cui implementazione è data da:

```
1 def cr_pulse(backend, amp, angle, duration):
2     backend_defaults = backend.defaults()
3     inst_sched_map = backend_defaults.instruction_schedule_map
4     cr_pulse = inst_sched_map.get('cx', (0, 1)).filter(channels
5         = [ControlChannel(0)], instruction_types=[Play]).instructions[0][1].pulse
6     cr_params = {}
7     cr_params['duration'] = cr_pulse.parameters['duration']
8     cr_params['amp'] = cr_pulse.parameters['amp']
9     cr_params['angle'] = cr_pulse.parameters['angle']
10    cr_params['sigma'] = cr_pulse.parameters['sigma']
11    cr_params['width'] = cr_pulse.parameters['width']
12    cr_risefall = (cr_params['duration'] - cr_params['width'])
13        / (2 * cr_params['sigma'])
14    angle_parameter = angle
15    duration_parameter = duration
16    sigma_parameter = cr_pulse.parameters['sigma']
17    width_parameter = int(duration_parameter - 2 * cr_risefall
18        * cr_params['sigma'])
19
20    pulse1 = GaussianSquare(duration = duration_parameter, amp
21        = amp, angle = angle_parameter, sigma = sigma_parameter,
22        width=width_parameter)
23    return pulse1
24
25    return my_program
```

Si procede con l'effettuare la stessa analisi introdotta per l'ansatz precedente.

L'andamento dell'energia in funzione del numero di iterazioni risulta essere:

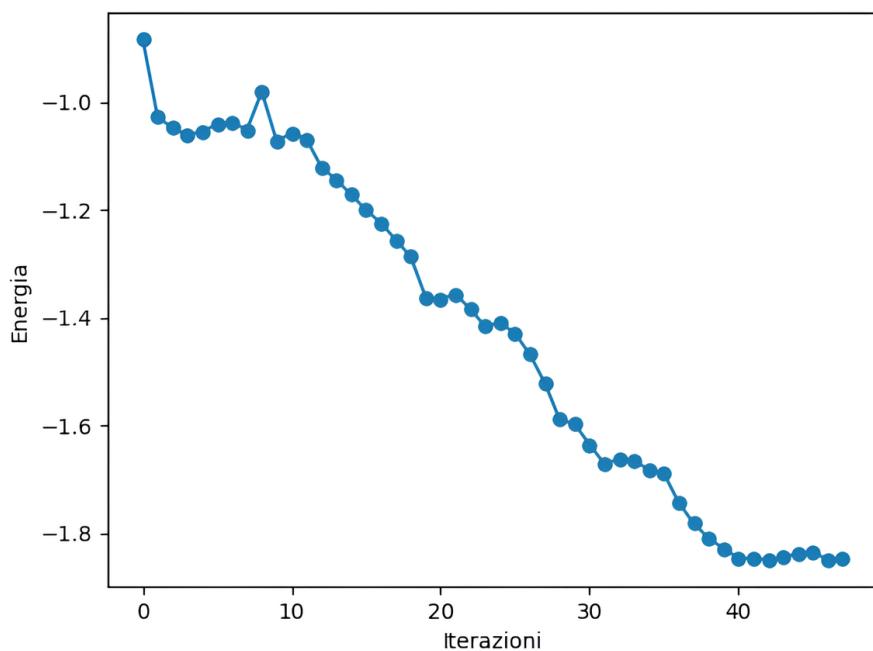


Figura 3.7: Stima dell’energia di ground della molecola  $H_2$  in funzione del numero di iterazioni eseguite dall’ottimizzatore COBYLA.

Associato anche in questo caso al parametro *maxiter* un valore di 100 si osserva che la simulazione ha raggiunto un valore di convergenza dell’energia associata allo stato fondamentale. Nonostante ciò, risulta evidente la differenza dell’ andamento dell’energia in funzione del numero di iterazioni rispetto ai dati analizzati in precedenza.

Iterazioni	Energia stato fondamentale [ $E_H$ ]
26	-0.780
47	-1.123

Step	Accuratezza
1	~ 68,59%
2	~ 98,79%

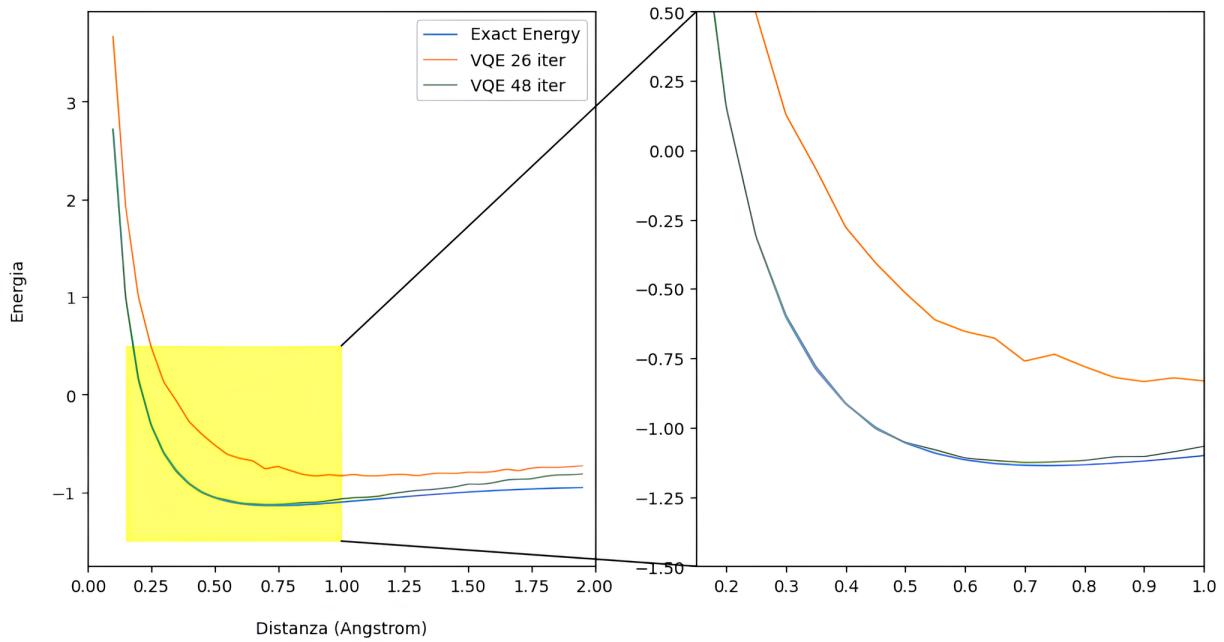


Figura 3.8: Andamenti energetici di una molecola di  $H_2$  al variare della distanza e del numero di iterazioni

Si vuole indagare la natura di questo discostamento eseguendo una serie di simulazioni nelle quali, senza far variare il numero di parametri associati ai singoli impulsi, si introducono una serie di ansatz differenti nel numero di impulsi costituenti. L’obiettivo è quello di indagare i "tempi" di convergenza della simulazione in funzione del grado di complessità di cui si costituisce l’ansatz di impulsi. Per far ciò, data l’articolata natura delle combinazioni di impulsi utilizzabili, si procede utilizzando ansatz costituiti in maniera progressiva da impulsi di tipo *DRAG* e *cr* (le cui forme funzionali sono state già definite). Per comprendere nel dettaglio la variazione dell’andamento in funzione del tipo di ansatz si è deciso di assegnare per ogni simulazione i medesimi valori dei parametri iniziali.

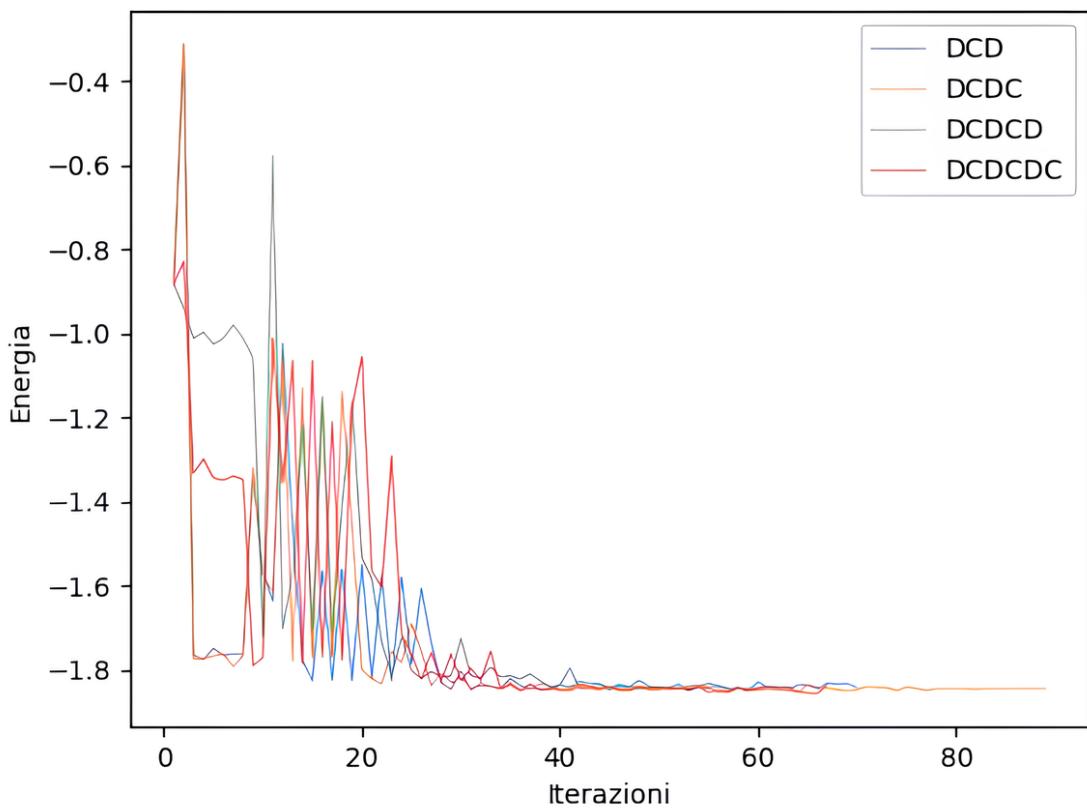


Figura 3.9: Confronto fra ansatz differenti (dove  $D$  rappresenta un impulso DRAG e  $C$  un impulso di controllo) per lo studio del numero di iterazioni necessarie alla convergenza

Lo studio eseguito non permette di evidenziare una correlazione tra la dimensione dell’ansatz ed il numero di iterazioni necessarie a convergere al minimo della cost function. Si conclude inoltre che l’andamento di convergenza associato al secondo ansatz implementato costituisca esclusivamente uno dei potenziali comportamenti osservabili da parte dell’ottimizzatore nel processo di stima del minimo della funzione. In merito a ciò, si giustifica il confronto (che verrà nuovamente eseguito per le simulazioni su hardware) con i risultati esposti nella seguente pubblicazione: [12].

L’analisi fin’ora svolta ha permesso di validare la bontà del codice scritto ed ha permesso inoltre di iniziare a comprendere alcune delle relazioni fondamentali che legano la natura del codice, come la scelta dell’ansatz e la sua articolazione, ai risultati già noti.

Si vuole ora presentare un ultimo risultato la cui applicazione a sistemi hardware è tra gli elementi fondamentali per cui risulta maggiormente vantaggioso eseguire operazioni su qubit attraverso programmi di impulsi: analisi dei tempi computazionali.

Il grafico che segue mette a confronto i tempi computazionali richiesti per l'esecuzione di simulazioni su ansatz progressivamente complessi.

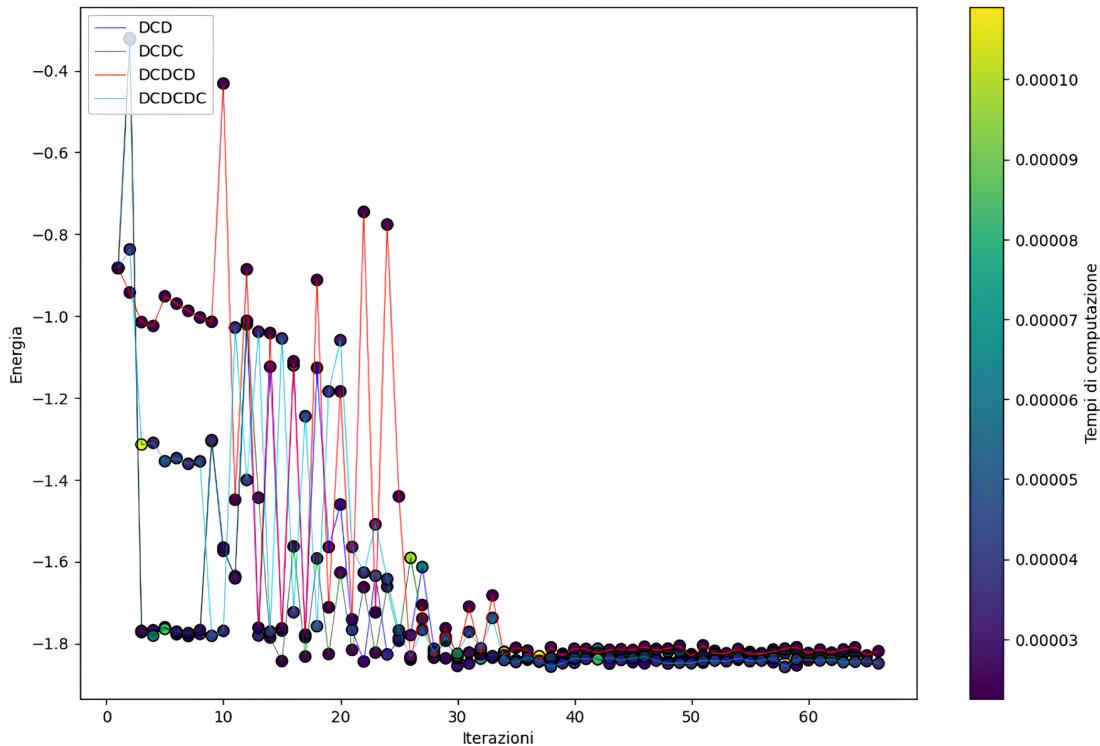


Figura 3.10: Confronto fra tempi computazionali [s] associati ad ansatz differenti

Quello che si è in grado di evincere dal grafico è che non è presente una marca differenza fra i tempi computazionali in funzione della progressiva complicazione dell'ansatz. I risultati ottenuti in questo caso sono indice delle capacità computazionali del computer su cui è stata eseguita la simulazione, nonostante ciò, l'applicazione di un'analisi di questo tipo ad un computer quantistico è in grado di evidenziare le differenze in termini di tempi computazionali tra ansatz gate based ed ansatz pulse based.

Si vuole ora procedere con l'eseguire simulazioni su computer quantistici reali. L'aspettativa è quella di ottenere risultati in una certa misura concordi a quelli ot-

tenuti data la natura intrinseca del simulatore FakeManila, programmato secondo le caratteristiche del computer quantistico *imbq\_manila*.

**Idrogeno molecolare su computer quantistici con sistema a due qubit**  
Sfruttando il primo tra gli ansatz definiti nelle simulazioni dell'idrogeno molecolare con FakeManila, si procede implementando come backend il computer quantistico *ibm\_nazca*.

Di seguito è riportato un primo risultato computazionale ottenuto impostando il parametro *maxiter* a 50. Questa scelta, che ritrova una prima giustificazione nei risultati precedentemente riportanti, sarà nuovamente giustificata di seguito.

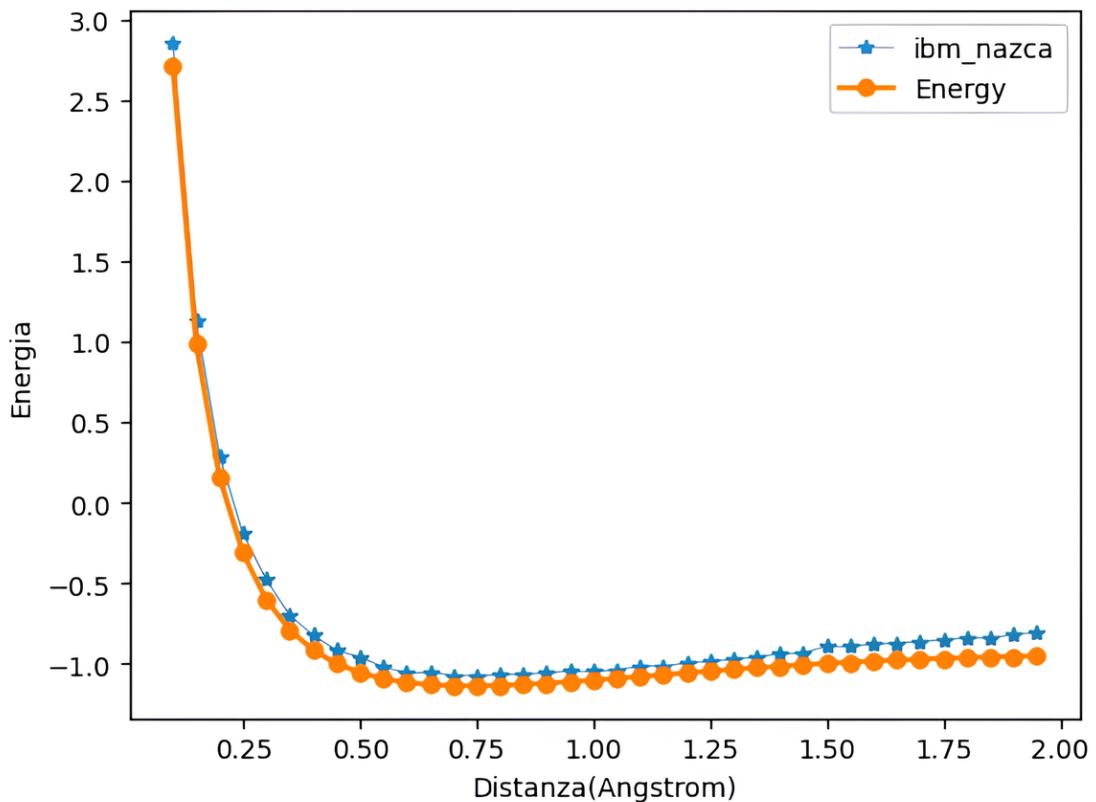


Figura 3.11: VQE con ansatz pulse based eseguito sul computer quantistico *ibm\_nazca*

L'energia dello stato fondamentale ricavata attraverso la seguente simulazione ri-

sulta essere pari a  $-1.075 E_H$ .

Si procede ora con l'eseguire la stessa operazione utilizzando il computer quantistico *ibm\_osaka* e si rappresenta l'andamento in un grafico.

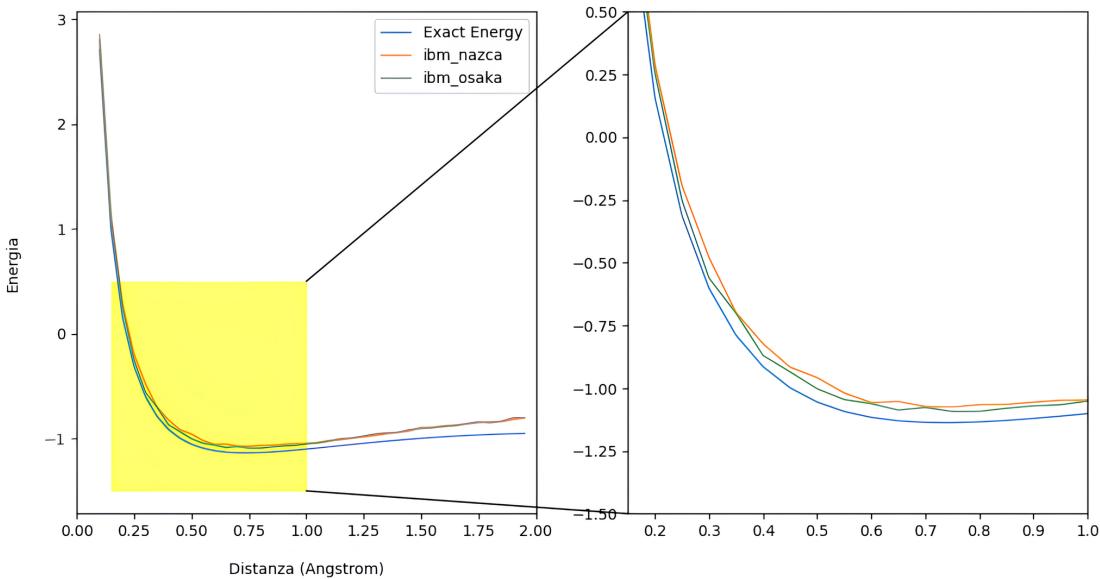


Figura 3.12: VQE con ansatz pulse based eseguito sui computer quantistici *ibm\_nazca* ed *ibm\_osaka*

L'energia dello stato fondamentale stimata attraverso quest'ultima simulazione risulta esser pari a  $-1.096 E_H$ .

Il lavoro svolto fin'ora ha come obiettivo quello di presentare i risultati ottenuti implementando il processo di VQE attraverso un ansatz di impulsi. Per comprendere a pieno il valore del lavoro svolto e le sue potenzialità, si vuole procedere ad eseguire un confronto con processi di VQE implementati attraverso ansatz gate based.

L'analisi delle differenze fra un ansatz gate based ed un ansatz pulse based ha inizio con la considerazione che attraverso un programma di impulsi è possibile accedere a parametri inaccessibili nel caso di una programmazione gate based. Questo ha un effetto diretto sulla trainability dei circuiti e di conseguenza sui tempi computazionali. I risultati presenti di seguito mostrano questa differenza (i dati riferiti ai gate derivano dalla calibrazione riportata nel segunete studio di ricerca [13]).

Tabella 3.1: Pulse Ansatz

Backend	Pulse	Pulse time [ns] a 0.735Å	N iter	Energia [ $E_H$ ]	T [s]
<i>ibm_nazca</i>	DRAG	35.2	50	-1.075	40.37
	ECR	56.32			
<i>ibm_osaka</i>	DRAG	35.2	38	-1.096	58.11
	ECR	56.32			

Tabella 3.2: Gate Ansatz

Backend	Gate	Gate time [ns]	N iter	Energia [ $E_H$ ]	T [s]
<i>ibmq_manila</i>	CNOT				
	X	0_1: 277.3	14	$-1.006 \pm 0.019$	$334 \pm 40$
	$S_X$	1_0: 312.8			
	$R_z$				
<i>aqt_marmot</i> <sup>4</sup>	CNOT				
	X	0_1: –	14	$-1.100 \pm 0.010$	$13297 \pm 625$
	$S_X$	1_0: –			
	$R_z$				

Dove con  $T$  ci si riferisce al tempo computazionale richiesto dall'intera simulazione.

In prima analisi si osserva come i tempi computazionali associati alle simulazioni eseguite nell'ambito dello studio qui presentato siano inferiori in relazione al numero di iterazioni effettuate dall'ottimizzatore, questo giustificato dalla complessità dei gate ansatz implementati al fine di ottenere risultati confrontabili con i valori d'aspettazione. Nell'analisi riportata il gate ansatz è un circuito RY-CNOT con una depth di 17 e 35 gate.

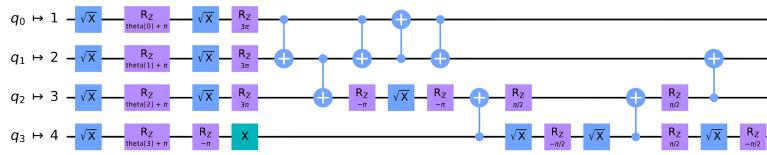


Figura 3.13: Circuito RY-CNOT

Di seguito è riportato l'andamento della stima dell'energia dello stato fondamentale in funzione del numero di iterazioni per le simulazioni eseguite.

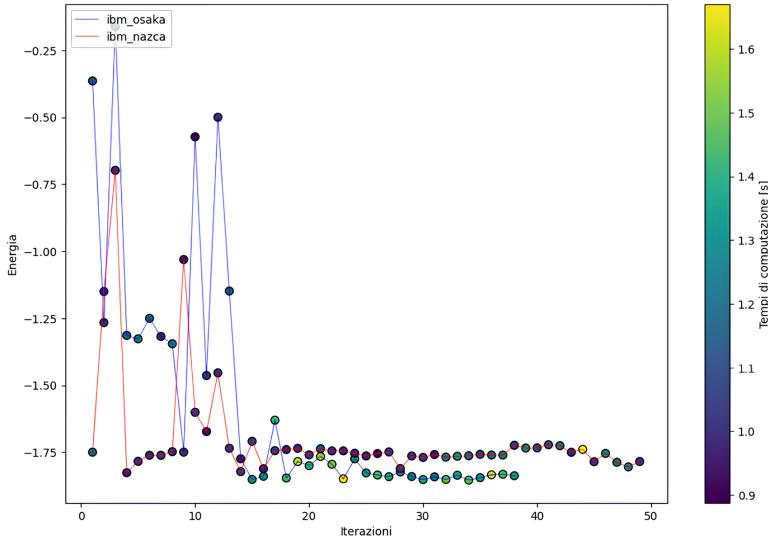


Figura 3.14: Confronto andamenti energertici in funzione del numero di iterazioni tra *ibm\_nazca* ed *ibm\_osaka*

A differenza di quanto osservato nel caso di simulazioni su simulatore, è possibile osservare una prima distinzione tra i tempi computazionali in funzione del tipo di backend (argomento principale dello studio precedentemente citato [13]). Questo aspetto non viene però ulteriormente approfondito nell'ambito del presente elaborato.

Si presenta infine il confronto fra la durata di un programma di impulsi ed un circuito costituito da gate. Per far ciò, si sfruttano in parte i risultati presenti nella pubblicazione [12] e li si mette a confronto con quelli ottenuti nelle presenti simulazioni.

Backend	Tipo di circuito	Durata circuito a $0.735\text{\AA}$ [ns]
<i>ibm_jakarta</i>	Gate	682.7
<i>ibm_lagos</i>	Gate	380
<i>ibm_lagos</i>	Pulse	50
<i>ibm_nazca</i>	Pulse	128
<i>ibm_osaka</i>	Pulse	128

Si noti che il primo risultato è ottenuto da [12] attraverso una generazione randomica dell'ansatz come circuito di gate. Si inoltre noti che nell'ambito del presente elaborato sarebbe stato possibile implementare un ansatz caratterizzato da un programma d'impulsi di dimensioni inferiori (come quello implementato nel presente studio [14]) ma per gli scopi ricercati (ottenere uno scostamento dai risultati ottenuti con circuiti di gate) gli ansatz implementati sono risultati soddisfacenti.

### 3.2.2 Idruro di litio (*LiH*)

L'analisi svolta per l'idruro di litio segue direttamente da quella eseguita per l'idrogeno molecolare. L'obiettivo è quello di introdurre un ansatz di impulsi e confrontare i risultati così ottenuti con un ansatz gate based. Si procede dunque implementando il seguente ansatz:

```

1 def HE_pulse(backend, amp, angle, width):
2     with pulse.build(backend) as my_program1:
3         # layer 1
4         sched_list = []

```

```
5     with pulse.build(backend) as sched1:
6         qubits = (0,1,2,3)
7         for i in range(4):
8             pulse.play(drag_pulse(backend, amp[i], angle[i]
9                 ]), DriveChannel(qubits[i]))
10        sched_list.append(sched1)
11
12    with pulse.build(backend) as sched2:
13        uchan = pulse.control_channels(0, 1)[0]
14        pulse.play(ecr_pulse(backend, amp[4], angle[4],
15            width[0]), uchan)
16        sched_list.append(sched2)
17
18    with pulse.build(backend) as sched4:
19        uchan = pulse.control_channels(1, 2)[0]
20        pulse.play(ecr_pulse(backend, amp[5], angle[5],
21            width[1]), uchan)
22        sched_list.append(sched4)
23
24    with pulse.build(backend) as sched6:
25        uchan = pulse.control_channels(2, 3)[0]
26        pulse.play(ecr_pulse(backend, amp[6], angle[6],
27            width[2]), uchan)
28        sched_list.append(sched6)
29
30    with pulse.build(backend) as my_program:
31        with pulse.transpiler_settings(initial_layout=
32            [0,1,2,3]):
33            with pulse.align_sequential():
34                for sched in sched_list:
35                    pulse.call(sched)
36
37    return my_program
```

La complessità computazionale di questa molecola risulta molto maggiore rispetto a quella dell'igrodeno molecolare, per questo motivo l'analisi non ha permesso di

trovare risultati pienamente soddisfacenti. Nonostante ciò, al fine di estendere le conclusioni raggiunte attraverso lo studio dell'idrogeno molecolare, si integrano i risultati ottenuti dalle simulazioni con alcuni risultati noti pubblicati nei seguenti articolo [12]. Nella tabella che segue sono riportati nelle prime 2 righe i risultati ottenuti nell'ambito del presente elaborato.

Backend	Tipo di circuito	qubit	Durata circuito a 1.595Å [ns]	Energia [ $E_H$ ]
<i>FakeManila</i>	Pulse	4	248	-6.483
<i>ibm_osaka</i>	Pulse	4	206,22	-6.780
<i>ibmq_manila</i>	Gate	4	380	-7.581
<i>ibmq_manila</i>	Pulse	4	50	-7.861
<i>ibm_montreal</i>	Pulse	4	199	-7.590
<i>ibm_montreal</i>	Gate	6	7296	-6.914

In maniera analoga a quanto precedentemente osservato per la molecola di idrogeno si riscontrano dimensioni circuitali inferiori per i circuiti costituiti da programmi di impulsi.

Sebbene i risultati presentati in quest'ultima analisi non possano di per se definirsi soddisfacenti poichè troppo distanti dal minimo della funzione, è possibile ugualmente osservare come nel caso dei circuiti implementati tramite programmi di impulsi, l'ottimizzatore classico sia in grado di approssimare il valore teorico in un minor numero di iterazioni, infatti, mentre nel caso delle prime due simulazioni i valori ottenuti sono il prodotto di un numero di iterazioni pari a 30, al fine di ottenere il valore riscontrato nella terza simulazione, l'ottimizzatore è stato portato ad eseguire un numero di iterazioni pari ad 800.

L'aspettativa dunque di un'analisi con un maggior numero di processi d'ottimizzazione dei parametri per un ansatz pulse based è quella di ottenere risultati migliori rispetto a quelli ottenuti, mantenendo ugualmente tempi computazionali inferiori rispetto a quelli richiesti per un circuito gate based.

# Capitolo 4

## Conclusioni

### 4.1 Analisi dei risultati computazionali

Con il presente studio si è andato ad introdurre ansatz come programmi di impulsi nel merito di processi di VQE (Variational Quantum Eigensolver) per le molecole di  $H_2$  ed  $LiH$ . Lo studio è stato svolto implementando ansatz del tipo Hardware Efficient (HE) per marcare le differenze in termini di dimensioni circuitali e tempi computazionali con i circuiti gate based.

L'insieme dei risultati computazionali associati alle simulazioni eseguite su computer quantistici sono riportati di seguito.

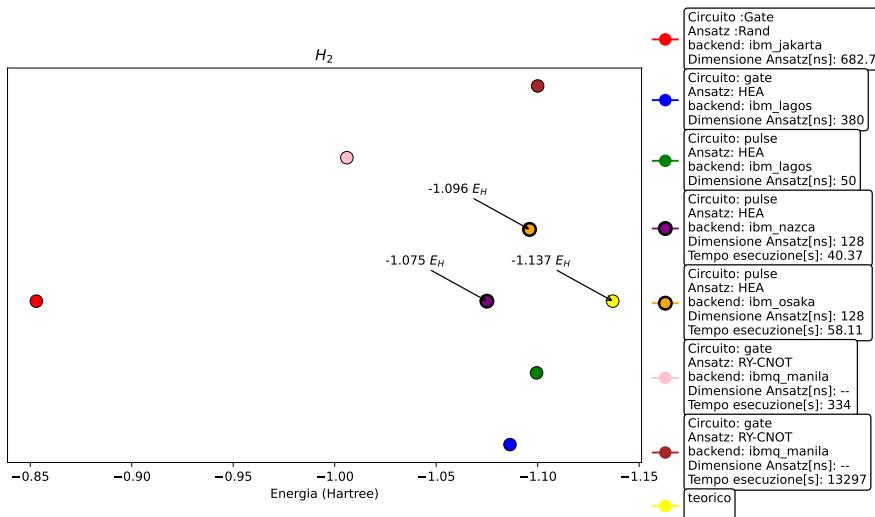


Figura 4.1:  
Risul-  
tati  $H_2$

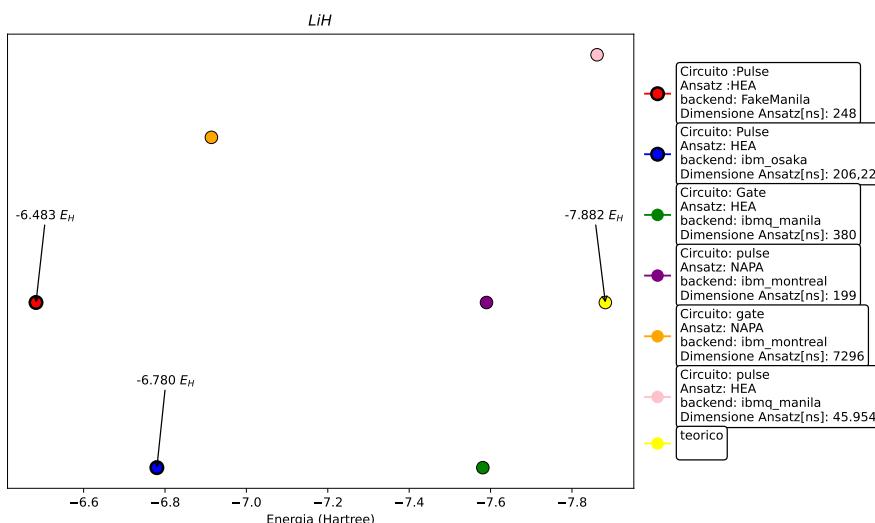


Figura 4.2:  
Risul-  
tati  
 $LiH$

Per quanto concerne l'idrogeno molecolare, le simulazioni su computer quantistico hanno portato ad una stima dell'energia dello stato fondamentale pari a  $-1.075E_H$  e  $-1.096E_H$ . Entrambe sono state eseguite implementando un ansatz di impulsi HE di durata pari a 128ns<sup>1</sup> caratterizzate da tempi computazionali pari a 40.37s e 58.11s rispettivamente. Si osserva come i risultati computazionali ottenuti siano compatibili con quelli ottenuti attraverso l'implementazione di ansatz gate based ma come questi siano il prodotto di una più elaborata operazione computazionale (nel grafico sono riportati i tempi computazionali e le rispettive durate degli ansatz).

Per quanto concerne l'idruro di litio, data la maggiore complessità computazione rispetto all'idrogeno molecolare, le simulazioni non hanno portato a risultati prossimi al valore teorico, viene però riportato (nel rispettivo grafico con il colore rosa), al fine di giustificare la validità delle conclusioni elaborate per l'idrogeno molecolare, il risultato prodotto dallo studio di ricerca PANSATZ (Pulse-based Ansatz for Variational Quantum) [14] che presenta un dato ( $-7.862E_H$ ) in forte accordo col valore teorico ( $-7.882E_H$ ). Si noti infine, che la riduzione della width dell'ansatz (come nel caso dello studio citato, dove quest'ultimo è costituito da un solo layer) comporta una più approssimativa descrizione del sistema e di conseguenza pone un limite al grado di approssimazione dell'andamento energetico studiato, sebbene nel limite del presente elaborato il risultato costituisca la miglior stima presentata. Un possibile sviluppo dell'analisi associata a questa molecola potrebbe consistere dunque nell'esecuzione di simulazioni implementando ansatz con width superiori ad 1 fino al raggiungimento della convergenza al valore teorico.

## 4.2 Analisi del lavoro svolto

Questo paragrafo ha lo scopo di riassumere brevemente il lavoro svolto nell'ambito del presente elaborato al fine di comprendere quali possano essere eventuali spunti per ulteriori analisi.

Lo studio intrapreso ha avuto come obiettivo quello di indagare le implicazioni associate all'implementazione di ansatz, nel merito di processi di VQE, come

---

<sup>1</sup>Nel limite di convergenza al valore teorico si è osservata l'ottimizzazione dei parametri associati alla durata degli impulsi che ha portato alla definizione di un circuito caratterizzato da questa durata.

programmi di impulsi invece che come circuiti gate based. L’analisi ha visto l’introduzione di ansatz di tipo Hardware Efficient al fine di marcare le differenze, in termini di tempi computazionali e dimensioni circuitali, osservabili effettuando un confronto con circuiti gate based. L’intera analisi è stata svolta sfruttando il toolkit qiskit ed il processo di ottimizzazione associato al metodo di calcolo variazionale ha interessato unicamente i parametri matematici associati ai singoli impulsi ed ha tralasciato l’ottimizzazione dei parametri naturali (ad esempio la frequenza). Lo studio ha visto come oggetto delle simulazioni le molecole  $H_2$  ed  $LiH$  ma gli esiti ottenuti (riportati nel paragrafo 4.1) risultano essere di carattere generale.

## 4.3 Possibili applicazioni e spunti per ulteriori analisi

### 4.3.1 Pulse ansatz optimization as a reinforcement learning problem

Lo studio affrontato ha permesso di evidenziare i vantaggi computazionali ottenibili nell’ambito della VQE implementando l’ansatz come un programma di impulsi piuttosto che come un circuito di gate. L’analisi è stata però eseguita implementando esclusivamente ansatz del tipo HE per poter trarre il maggior vantaggio in termini di tempi computazionali, ma questi non sempre costituiscono la miglior opzione. A questo proposito molti studi hanno proposto l’introduzione di ansatz la cui natura potesse essere quanto più rappresentativa possibile, introducendo ansatz ibridi del tipo HE e Unitary Coupled Cluster (UCC). Uno degli approcci principali per la costruzione di ansatz di questa natura avviene attraverso il deep reinforcement learning (RL): si rappresenta l’ottimizzazione dell’ansatz come un ambiente di reinforcement learning dove lo stato costituisce il circuito e l’azione rappresenta l’addizione di un gate al circuito. Ad ogni tempo  $t$ , l’agente (ovvero la componente che si occupa di ottenere informazioni dall’ambiente e portare avanti il processo d’ottimizzazione) ottiene una ricompensa  $R$  in funzione del grado di approssimazione del valore computato col valore teorico, ad esempio:

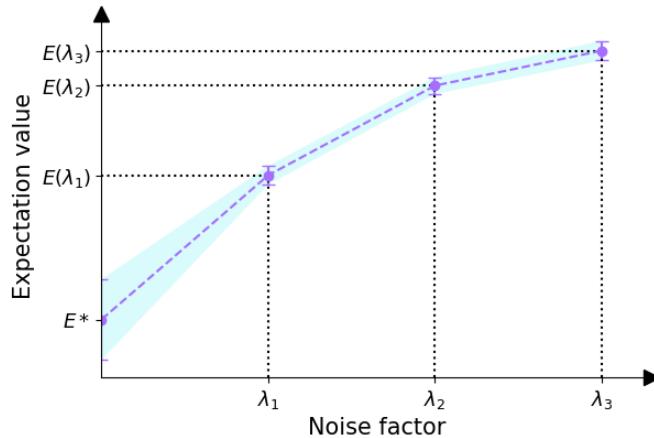
$$R = \begin{cases} 5 & \text{if } E_t < \xi \\ -5 & \text{if } t > L \text{ and } E_t > \xi \\ \max\left(\frac{E_{t-1}-E_t}{E_{t-1}-E_{\min}}, -1\right) & \text{otherwise} \end{cases}$$

dove  $\xi$  rappresenta un limite di accettazione del dato ottenuto.

Un ulteriore sviluppo per il seguente elaborato potrebbe vedere l'introduzione di ansatz implementati come programma di impulsi all'interno di un ambiente di reinforcement learning, andando a studiare le differenze con ansatz gate based già ampiamente sfruttati nell'ambito delle VQE e con ansatz gate based costruiti sfruttando questo approccio [15].

### 4.3.2 Zero noise extrapolation [16]

Lo studio portato avanti in questo elaborato non ha avuto tra i suoi obiettivi l'analisi dell'errore dovuto al rumore dei computer quantistici usati. Un possibile sviluppo dello studio potrebbe vedere l'introduzione della tecnica di Zero Noise Extrapolation (ZNE), il cui obiettivo è quello di comprendere le risposte di un computer quantistico stimolato con gradi diversi di rumore (artificialmente introdotti e parametrizzati da un fattore  $\lambda$ ), al fine di estrapolare il suo comportamento in condizione di zero rumore, per poi andare a confrontare i risultati ottenuti con quelli presentati precedentemente.



# Bibliografia

- [1] Qiskit. PAntza. *GitHub repository*. Available at: <https://github.com/qismib/PAntza>
- [2] Renato Portugal. *Basic Quantum Algorithms*. Available at: <https://arxiv.org/abs/2201.10574>
- [3] Guanru Feng, Dawei Lu, Jun Li, Tao Xin, Bei Zeng *Quantum Computing: Principles and Applications*. Available at: <https://arxiv.org/abs/2310.09386>
- [4] Pablo Echenique, J. L. Alonso *A mathematical and computational review of Hartree-Fock SCF methods in Quantum Chemistry*. Available at: <https://arxiv.org/abs/0705.0337>
- [5] Etienne Bernard, Francesco Salvarani *Basic mathematics for quantum computing*. Available at: [https://www.google.com/url?sa=t&source=web&rct=j&opi=89978449&url=https://mate.unipv.it/~salvaran/IMEDiL/Basic\\_mathematics\\_quantum\\_computing.pdf&ved=2ahUKEwiNwsmZsqGHAxWlhPOHHWxND1kQFnoECCkQAQ&usg=A0vVaw0b0PWeT5qy8vsazIKuPQF5](https://www.google.com/url?sa=t&source=web&rct=j&opi=89978449&url=https://mate.unipv.it/~salvaran/IMEDiL/Basic_mathematics_quantum_computing.pdf&ved=2ahUKEwiNwsmZsqGHAxWlhPOHHWxND1kQFnoECCkQAQ&usg=A0vVaw0b0PWeT5qy8vsazIKuPQF5)
- [6] Gabriel T. Landi. *Second Quantization*. University of São Paulo. Available at: <https://www.google.com/url?sa=t&source=web&rct=j&opi=89978449&url=http://www fmt if usp br/~gtlandi/courses/second-quantization-4.pdf&ved=2ahUKEwiGoICHs6GHAxUag-OHHVkB6AQFnoECA8QAQ&usg=A0vVaw35pT3fxU201PVKhfevoyz9>
- [7] Quantum Gates, Quantum Circuit and Quantum Computation. Available at: <https://www.google.com/url?sa=t&source=web&rct=j&opi=>

- 89978449&url=https://cklixx.people.wm.edu/teaching/QC2021/  
QC-chapter5.pdf&ved=2ahUKEwiZs961rKGHAxUKg\_0HHUnrATAQFnoECBQQAQ&  
usg=A0vVaw3MwZDZtm1-62rDerBzL4VH
- [8] Junhan Qin *Review of Ansatz Designing Techniques for Variational Quantum Algorithms*. Available at: <https://arxiv.org/abs/2212.04913>
  - [9] *SolovayKitaev theorem* Available at: <https://docs.quantum.ibm.com/api/qiskit/qiskit.transpiler.passes.SolovayKitaev>
  - [10] *Qiskit* Available at: <https://www.ibm.com/quantum/qiskit>
  - [11] Lorenzo Leone, Salvatore F.E. Oliviero, Lukasz Cincio, M. Cerezo *On the practical usefulness of the Hardware Efficient Ansatz* Available at: <https://arxiv.org/abs/2211.01477>
  - [12] Zhiding Liang, Jinglei Cheng, Hang Ren, Hanrui Wang, Fei Hua, Zhixin Song, Yongshan Ding, Fred Chong, Song Han, Xuehai Qian, Yiyu Shi *NAPA: Intermediate-level Variational Native-pulse Ansatz for Variational Quantum Algorithms* Available at: <https://arxiv.org/abs/2208.01215>
  - [13] Amine Bentellis, Andrea Matic-Flierl, Christian B. Mendl, Jeanette Miriam Lorenz *Benchmarking the Variational Quantum Eigensolver using different quantum hardware* Available at: <https://arxiv.org/abs/2305.07092>
  - [14] Dekel Meirom, Steven H. Frankel *PANSATZ: Pulse-based Ansatz for Variational Quantum Algorithms* Available at: <https://arxiv.org/abs/2212.12911>
  - [15] Mateusz Ostaszewski, Lea M. Trenkwalder, Wojciech Masarczyk, Eleanor Scerri, Vedran Dunjko *Reinforcement learning for optimization of variational quantum circuit architectures* Available at: <https://arxiv.org/abs/2103.16089>
  - [16] Alexey Uvarov, Daniil Rabinovich, Olga Lakhmanskaya, Kirill Lakhman-skiy, Jacob Biamonte, and Soumik Adhikary *Mitigating quantum gate errors for variational eigensolvers using hardware-inspired zero-noise extrapolation* Available at: <https://journals.aps.org/prabSTRACT/10.1103/PhysRevA.110.012404>