JhedAI Research Paper Series

Simulación Cuántico-Inspirada y Optimización AutoML de Circuitos Cuánticos Variacionales en Arquitecturas GPU Clásicas

Julio Hofflinger — Director de Operaciones, JhedAI, Chile
Contacto: julio@jhedai.com

Abstract

This paper proposes a quantum-inspired simulation framework for the automatic optimization of Variational Quantum Circuits (VQCs) on classical GPU architectures.

The approach combines vectorized simulation of VQCs with an AutoML module based on Graph Neural Networks (GCN) that predicts circuit performance and ranks candidate architectures.

Additionally, the framework integrates SABRE transpilation, heavy-hex qubit layout, and realistic noise modeling (T_1, T_2, readout) for reproducible quantum-classical experiments.

It also leverages Tensorcore-accelerated simulation (TANQ-Sim) and Qiskit Aer's *StatevectorSimulator* for hybrid GPU performance, creating a bridge between theoretical quantum design and practical classical execution.

Resumen

Este trabajo propone un marco de simulación cuántico-inspirada para la optimización automática de *Variational Quantum Circuits* (VQC) sobre arquitecturas clásicas basadas en GPU.

El enfoque combina la simulación vectorizada de VQC con un módulo AutoML basado en *Graph Neural Networks* (GCN) que predice el rendimiento de los circuitos y clasifica las arquitecturas candidatas.

El marco integra la transpilación SABRE, el layout *heavy-hex*, la simulación acelerada por Tensorcore (TANQ-Sim) y el modelado de ruido realista (T_1, T_2, *readout*) para experimentos cuántico-clásicos reproducibles.

El resultado es un puente entre la computación cuántica teórica y la optimización práctica sobre hardware clásico GPU.

1. Introducción

Los *Variational Quantum Circuits* (VQC) constituyen un paradigma fundamental del *Quantum Machine Learning* al usar compuertas cuánticas parametrizadas para representar funciones complejas.

Dada la limitada disponibilidad de hardware cuántico real y la fragilidad de los dispositivos NISQ, las simulaciones cuántico-inspiradas en GPU representan una alternativa eficiente para investigar el comportamiento, expresividad y estabilidad de los VQC.

Este trabajo propone una metodología que combina simulación vectorizada, AutoML predictivo y modelado de ruido físico.

El objetivo del estudio es diseñar y validar un marco híbrido GPU-QML que permita optimizar automáticamente arquitecturas VQC, considerando restricciones de topología real (heavy-hex), transpilación SABRE y métricas reproducibles de ruido y fidelidad.

2. Marco Teórico y Fundamentos

2.1. Formalismo Matemático de los VQC

- Un **VQC** se define como una unidad unitaria parametrizada **W(Θ),** ompuesta por **L** bloques variacionales **Vj(θj).**
- Un codificador de entrada **U(λ)** mapea los datos clásicos **λ** a ángulos de rotación.
- La medición produce valores de expectación de **Pauli-Z** en cada qubit, generando un vector de *embedding* **z(λ):**

$$\langle Z_k \rangle = \langle 0 | U^\dagger(\lambda) W^\dagger(\Theta) Z_k W(\Theta) U(\lambda) | 0 \rangle \qquad (1)$$

Donde:
$\langle Z_k \rangle$: Valor esperado del operador Pauli-Z sobre el qubit k.
$|0\rangle$: Estado base inicial.
U(λ): Codificador de datos.
El Operador †: Conjugado transpuesto.

Define la medición promedio del observable $\mathbf{Z_k}$ sobre el estado final del sistema.

Forma equivalente (estado puro):

$$\langle Z_k \rangle = \langle \Psi | Z_k | \Psi \rangle, \text{ con } |\Psi\rangle = W(\Theta)U(\lambda)|0\rangle^{\otimes n}$$

2.2. Simulación Cuántico-Inspirada

La simulación vectorizada se implementa sobre **GPU** con soporte Tensorcore, siguiendo la metodología de ***TANQ-Sim*** **(2025).**

Esta permite operar con matrices densas complejas en precisión mixta **FP16/FP32,** logrando hasta 4× de mejora frente a simulaciones **FP64**.

Se adopta un mapeo angular suave mediante función arctangente para reducir aliasing numérico **[Sim et al., 2019]** y conservar continuidad de fase.

La fase global del vector de estado se ignora por no afectar los valores medibles.

La arquitectura híbrida utiliza un modelo *noisy-statevector*, en el que cada operación puede incorporar canales de Kraus representando efectos físicos de decoherencia **(T_1, T_2)** y errores de lectura.

Esta estrategia permite reproducir condiciones experimentales de dispositivos reales en entornos **GPU** controlados.

$$x' = \arctan(W_x x + b) \qquad (2)$$

Donde:
x**:** Vector de datos clásicos ($\mathbf{R}^d$);
W_x**:** Matriz de proyección (n×d);
b**:** Vector de sesgos;
x'**:** Vector de ángulos $((-\pi/2, \pi/2)^n)$;
a**rctan:** Mapeo suave y diferenciable.

Reemplaza el clipping clásico por una función continua para preservar gradientes y estabilidad en GPU.

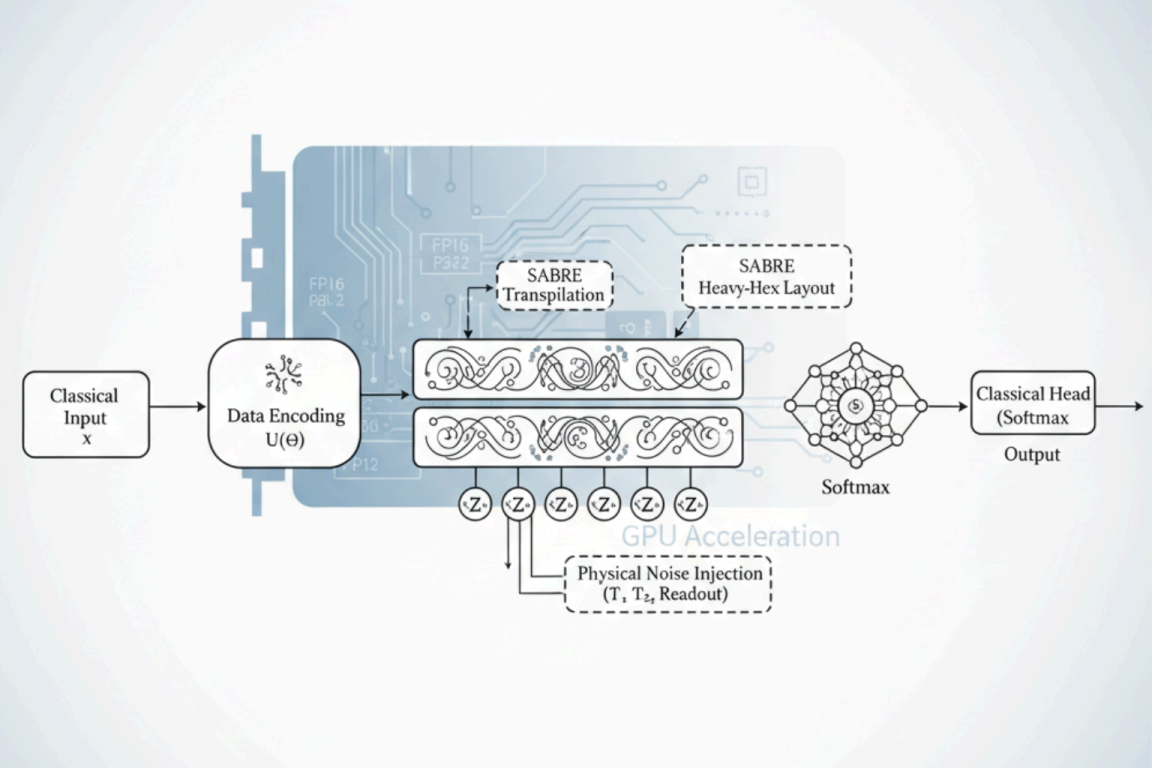

Figura 1. Flujo cuántico-inspirado ejecutado sobre GPU. El pipeline integra codificación U(***λ***), bloques V(***θ***), medición $\langle Z_k \rangle$ y cabeza clásica (***softmax***). Incluye además la transpilación **SABRE**, layout heavy-hex y la inyección de ruido físico.

2.3. Integración AutoML-VQC

El módulo **AutoML** representa los circuitos como grafos, donde los nodos corresponden a compuertas y las aristas a dependencias lógicas.

Una ***Graph Convolutional Network*** (GCN) se pre-entrena mediante reconstrucción de características enmascaradas (***masked feature reconstruction***) y se ajusta finamente con una pérdida de ranking tipo *hinge loss*:

$$L=\max(0,m-(\hat{y}_j-\hat{y}_i)) \qquad (6)$$

Pérdida ***hinge-ranking*** entre predicciones $\hat{\mathbf{y}}_j,\hat{\mathbf{y}}_i$ con margen $m>0$. *(Basada en Situa et al., 2025).* De este modo, las arquitecturas pueden ordenarse según rendimiento estimado y seleccionarse las más prometedoras para entrenamiento completo.

2.4. Ruido, Transpilación y Layout

Para garantizar correspondencia con hardware IBM real, se aplica transpilación SABRE en tres etapas:

1. **Layout selection:** asigna qubits lógicos a físicos según la conectividad ***heavy-hex*.**
2. **Routing optimization:** inserta **SWAPs** mínimos para respetar restricciones topológicas.
3. **Gate simplification:** fusiona y cancela compuertas redundantes.

El flujo permite evaluar la fidelidad antes y después de la transpilación, midiendo la profundidad y el costo computacional en **GPU**.

Los efectos de ruido (decoherencia y lectura) se modelan mediante canales de Kraus integrados, generando una simulación *noise-aware* reproducible.

$$\varrho' = \sum_i E_i \rho E_i^\dagger \tag{9}$$

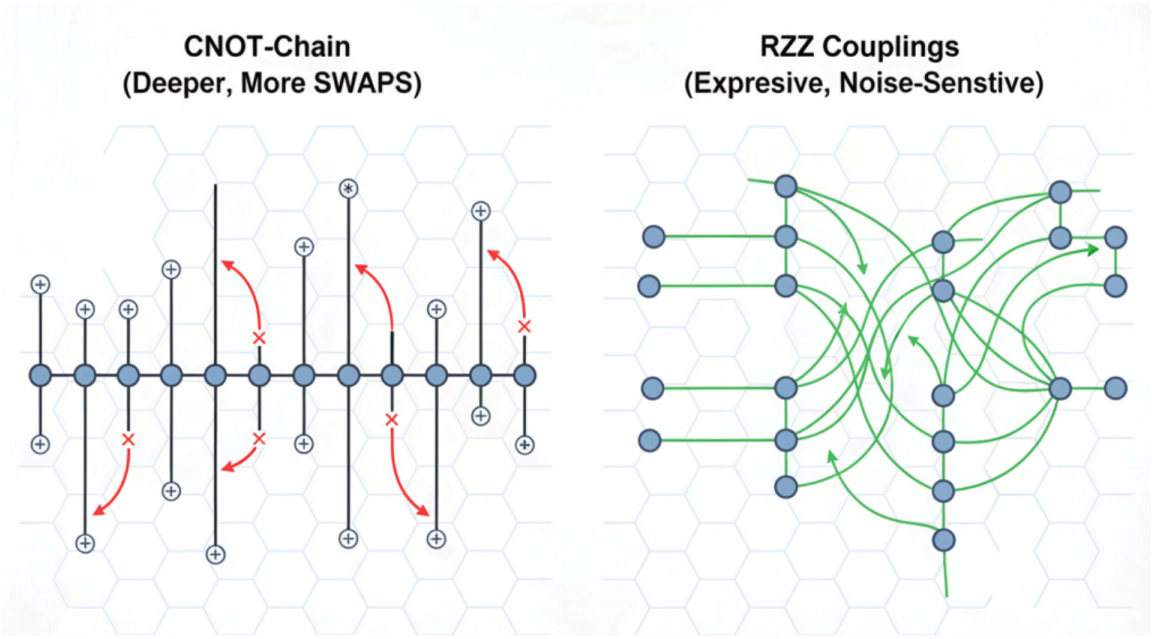

Figura 2. Comparación en layout heavy-hex: izquierda, cadena **CNOT** con **SWAPs** (mayor profundidad); derecha, **acoplamientos RZZ** más expresivos pero sensibles al ruido.

3. Metodología Propuesta

La metodología desarrollada busca integrar en un mismo marco computacional tres componentes centrales:

1. **Simulación cuántico-inspirada en GPU**,
2. **Optimización automática mediante AutoML-GCN**, y
3. **Entrenamiento jerárquico conjunto de modelos híbridos cuántico-clásicos.**

Este sistema permite evaluar y comparar arquitecturas variacionales bajo condiciones realistas, incluyendo ruido físico, limitaciones de conectividad (***heavy-hex***) y transpilación efectiva (**SABRE**).

A continuación, se detallan los elementos que conforman el flujo completo de procesamiento y entrenamiento.

3.1. Arquitectura y Flujo de Entrenamiento

El sistema integra tres componentes principales:

1. **Simulador VQC en GPU (PyTorch/CUDA):** ejecutado sobre el backend StatevectorSimulator de **Qiskit Aer**, configurado con device='GPU' y precision='single'. El objetivo es emular la evolución cuántica de los circuitos variacionales mediante operaciones matriciales vectorizadas.

2. **Modelo híbrido cuántico-clásico:** las salidas del **VQC** —representada por el vector de expectativas $z(x)$—se conectan a una cabeza **MLP.** El modelo híbrido aprende a mapear representaciones cuánticas a etiquetas o valores continuos en tareas de clasificación o regresión.

3. **Predictor AutoML-GCN:** Un modelo **AutoML** basado en ***Graph Convolutional Networks*** analiza la estructura topológica de cada circuito **VQC** (nodos como compuertas y aristas como dependencias) y estima su rendimiento antes del entrenamiento completo, permitiendo seleccionar las arquitecturas más prometedoras (***Top-K***).

3.2. Estructura Matemática del Modelo

El modelo híbrido conecta el circuito variacional con una capa densa final a través de una transformación lineal:

$$y = \text{softmax}(W_c z(x) + b_c) \tag{5}$$

Donde:
W_c, b_c**:** pesos y sesgos de la capa **MLP** (modelo clásico); **softmax:** normaliza el vector de salida como probabilidades; $z(x)$**:** vector de expectativas de **Pauli-Z** obtenido tras la medición cuántica.

El conjunto de parámetros $\boldsymbol{\theta}$ (**cuánticos**) y $\boldsymbol{\phi} = (W_c, b_c)$ (**clásicos**) se optimiza de manera conjunta mediante retropropagación híbrida, con gradientes calculados sobre amplitudes vectorizadas.

$$H^{(\iota+1)} = \sigma(\tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} H^{(l)} W^{(\iota)}) \tag{7}$$

Propagación normalizada de una **GCN.**

Donde:
$\mathbf{H}^{(\iota)}$**:** matriz de activaciones en la capa ι;

$\tilde{A} = \boldsymbol{A} + \boldsymbol{I}$**:** adyacencia con auto-conexiones;

$\tilde{\mathbf{D}}$**:** matriz diagonal de grados normalizada ($\tilde{D}$);

Definimos $\tilde{\mathbf{D}}_{ii} = \sum_j \tilde{A}_{ij}$, de modo que $\tilde{\mathbf{D}}^{-\frac{1}{2}} \tilde{A} \tilde{\mathbf{D}}^{-\frac{1}{2}}$ es la forma simétricamente normalizada.

$\mathbf{W}(\iota)$**:** pesos de la capa ι**.**

$\sigma(\cdot)$**:** función de activación no lineal, en este caso función **ReLU**.

3.4. Pérdida AutoML y entrenamiento jerárquico

La **GCN** se entrena con (6) y se usa para filtrar ***Top-K***. Luego se entrena íntegramente ese subconjunto en GPU. La pérdida total conjunta:

$$L_{total} = L_{AutoML} + \alpha L_{VQC} \tag{12}$$

Donde: $\mathbf{L_{AutoML}}$**:** Ranking (6); $\mathbf{L_{VQC}}$**:** Pérdida de tarea; $\boldsymbol{\alpha}$**:** Ponderación.

3.5. Métrica de validación

$$\rho = 1 - \frac{6 \sum_i d_i^2}{N\left(N^2 - 1\right)} \tag{10}$$

Donde:
ϱ**:** correlación de Spearman; $\mathbf{d_i}$**:** diferencia de rangos real-predicho; ***N*****:** número de arquitecturas. $\varrho \in$**[−1,1].**

3.6. Pipeline operativo (Figura 2)

1. Simulación GPU: U(λ) → V(Θ)→ ⟨Zk⟩ (Ecs. 2, 9).
2. Predicción AutoML: GCN (Ecs. 7, 6).
3. Optimización final: *Top-K* + pérdida conjunta (Ec. 12).

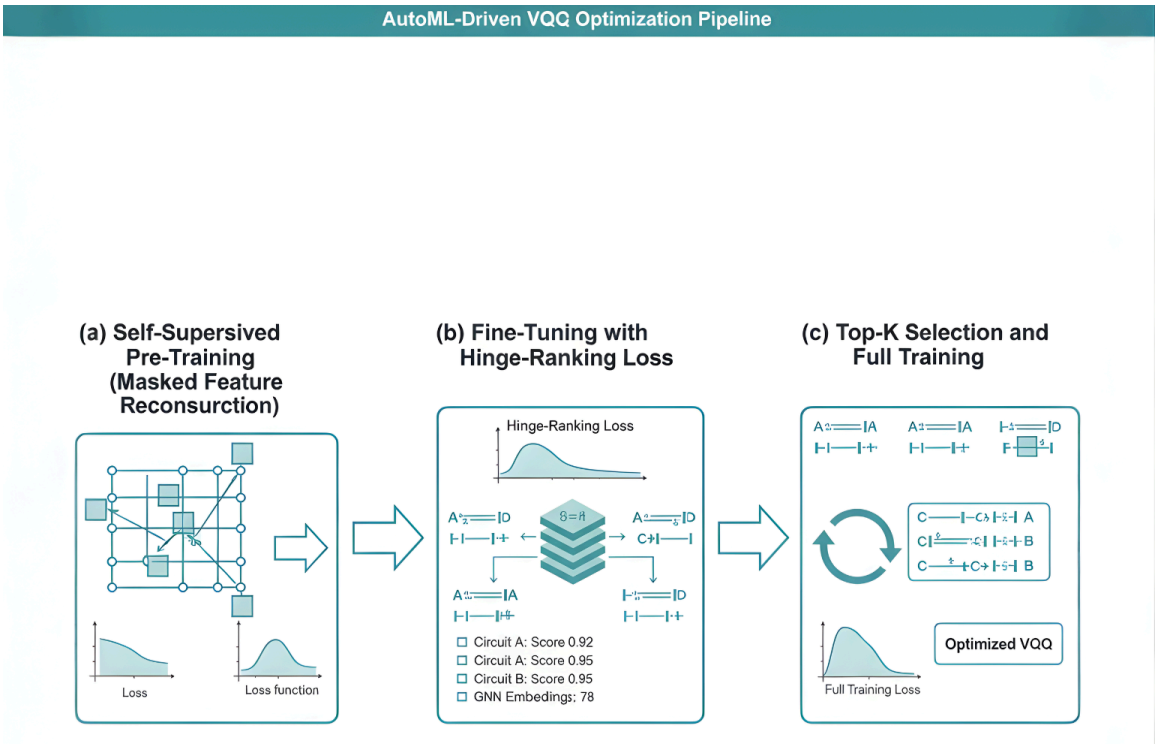

Figura 3. Flujo AutoML-VQC en tres etapas: *pre-entrenamiento auto-supervisado (reconstrucción enmascarada), ajuste fino con pérdida de ranking y selección Top-K con entrenamiento completo.*

3.7. Hiperparámetros y configuración

AdamW, ***lr*** =2 * 10^{-3}, ***batch*** **128**, **30–50 épocas**, **FP16** (***autocast***), ***gradient clipping*** **1.0**, ***checkpointing*** en circuitos profundos, **n≤16** qubits.

4. Ecuaciones Clave del Circuito

(2) Mapa de características

$$x' = \arctan(W_x x + b) \tag{2}$$

Esta ecuación transforma el vector clásico de entrada x (dimensión d) en un vector angular x' (dimensión n, igual al número de qubits).

W_x es una matriz de pesos que proyecta los datos al espacio de los qubits, y b es un vector de sesgos.

La función $\arctan(\cdot)$ realiza un mapeo suave al rango $(-\pi/2,\pi/2)$, garantizando que los valores puedan interpretarse como ángulos de rotación físicos para compuertas cuánticas.

Este mapeo reemplaza al clipping tradicional $x'=\mathbf{clip}(W_x x,[-\pi,\pi])$, introduciendo continuidad y estabilidad de gradiente. La elección de arctan se justifica porque su derivada $\frac{d}{dx}$ **arctan** $(x) = \frac{1}{1+x^2}$**,** es continua y acotada, lo que evita saturaciones fuertes y favorece gradientes estables en **GPU**."

(3) Bloque variacional

$$V_j(\theta_j) = \prod_k R_y(\theta_{j,k}) \mathrm{CNOT}_{(k,k+1)} \tag{3}$$

El bloque variacional $\mathbf{V_j(\theta_j)}$ se compone de rotaciones alrededor del eje **Y** $\mathbf{(R_y(\theta_{j,k}))}$ aplicadas a cada qubit, seguidas de una cadena de compuertas **CNOT** que entrelazan qubits adyacentes.

El producto $\prod_k$ indica aplicación secuencial de compuertas sobre los qubits **k**.

Cada parámetro $\boldsymbol{\theta_{j,k}}$ es entrenable y define la amplitud de rotación individual.

Esta formulación sintetiza el esquema hardware-efficient ansatz descrito por Sim et al. (2019), adaptado a **GPU**. El orden rotación–**CNOT**–rotación se eligió por su estabilidad numérica en entornos vectorizados.

(4) Vector de Expectativas Cuánticas

$$z(x)=[\langle Z_1\rangle,\langle Z_2\rangle,\ldots,\langle Z_n\rangle] \tag{4}$$

El vector $z(x)$ agrupa los valores esperados de **Pauli-Z** de cada qubit tras la medición. Cada componente $\langle \mathbf{Z_k}\rangle$ está en el rango **[−1,1]** y describe la probabilidad relativa de medir **0** o **1** en el qubit **k**.

Este vector actúa como representación (***embedding***) cuántico-inspirada que se conecta a una red neuronal clásica (**MLP**).

(8) Estado cuántico final

$$|\psi(\Theta,\lambda)\rangle=W(\Theta)\,U(\lambda)\,|0\rangle^{\otimes n} \tag{8}$$

*E*volución unitaria explícita desde el estado base. *(Propuesta; conecta con Ec. 1).*

(11) Expectación con estados mixtos

$$\langle Zk\rangle=\mathrm{Tr}(\rho\, Z_k) \tag{11}$$

Cálculo de expectativa sobre $\boldsymbol{\rho}$ (**matriz de densidad**). *(Propuesta; extensión ruidosa de Ec. 1).*

5. Resultados Esperados e Hipótesis

- **H1:** Los ***embeddings*** **cuántico-inspirados** superarán a una **MLP** clásica equivalente en datasets no lineales.
- **H2: AutoML-GCN** logrará $\boldsymbol{\rho}$**>0.7**, entrenando sólo **10–20 %** de las arquitecturas con **≥ 95 %** del mejor rendimiento.
- **H3:** Simulación de **~16 qubits viable en GPU** gama media-alta con **FP16** y **ruido modelado**.

6. Discusión y Limitaciones

Escalabilidad: coste **$O(2^n)$. Barren plateaus:** gradientes pequeños en ansätze profundos. **Entrelazamiento físico:** emulado matemáticamente. **Ruido:** se incluye decoherencia **(T_1, T_2)** y ***readout***, no fluctuaciones térmicas.

Evidencia previa (*Situa et al., 2025*) reporta ρ**=0.84** entre rendimiento real y predicho (**7–8 qubits**), apoyando la viabilidad del enfoque **AutoML-GCN. SABRE + TANQ-Sim** aportan realismo y reproducibilidad.

7. Conclusión

Se presenta un marco cuántico-inspirado que integra simulación vectorizada en **GPU**, **AutoML-GCN** predictivo, transpilación **SABRE** y modelado de ruido realista. El enfoque reduce coste de exploración, prioriza ***Top-K*** prometedoras y analiza el impacto del *routing* y el ruido en el rendimiento. Este puente operativo entre teoría variacional y práctica en **GPU** habilita un nuevo paradigma de diseño cuántico autoasistido.

7.1. Aportes

(1) Simulación GPU ruidosa reproducible; **(2)** AutoML estructural con GCN; **(3)** Entrenamiento jerárquico conjunto; **(4)** Transpilación y *layout* realista; **(5)** Pipeline transferible.

7.2. Futuro

Meta-aprendizaje (Quantum-Train Agent), mitigación adaptativa de ***barren plateaus***, **compresión tensorial** (MPS/TTN/PEPS), **validación en hardware** (IBM Q, IonQ) y **escalado multi-GPU** (Ray/DeepSpeed).

8. Referencias

[1] Cerezo et al., *Variational Quantum Algorithms*, NRP 3(9):625–644, 2021.
[2] Sim, Johnson & Aspuru-Guzik, *Adv. Quantum Technologies* 2(12):1900070, 2019.
[3] Liu et al., *Programming Variational Quantum Circuits with Quantum-Train Agent*, Preprint, 2024.
[4] Situa et al., *AutoML-Driven Optimization of VQCs*, SSRN 5167459, 2025.
[5] IBM Quantum Docs — *Transpilation Optimizations with SABRE*, 2025.
[6] IBM Quantum Docs — *Transpiler Stages Guide*, 2025.
[7] arXiv:2404.13184v2 — *Noise-Aware Quantum Simulation on GPUs*, 2024.
[8] *TANQ-Sim: Tensorcore Accelerated Noisy Quantum Simulation*, 2025.

9. Nota Institucional

Este trabajo corresponde a una investigación personal desarrollada por **Julio Hofflinger**, como parte de un estudio independiente sobre simulación cuántico-inspirada y optimización AutoML aplicada a circuitos variacionales.

El contenido integra conocimientos adquiridos durante su trabajo en **JhedAI**, pero no representa necesariamente una línea institucional oficial.

Su propósito es contribuir al avance de la investigación aplicada en computación cuántica e inteligencia artificial desde América Latina.