# Tensor Networks for Quantum-Inspired Simulations

Author: Jack Benarroch Jedlicki, jbenarje7@alumnes.ub.edu
*Facultat de Física, Universitat de Barcelona, Diagonal 645, 08028 Barcelona, Spain.*

Advisors: Bruno Julià-Díaz (bruno@fqa.ub.edu), Artur García-Saez, Stefano Carignano

**Abstract:** Quantum algorithms have the potential to accelerate computation and reduce memory requirements on advanced quantum computers. However, current hardware limitations hinder their application to complex problems. In this work, we investigate a promising approach that bypasses the need for quantum hardware by leveraging tensor networks to simulate quantum algorithms on classical computers. We assess the performance of quantum-inspired simulators relative to classical methods in terms of memory, runtime, and accuracy. Our results demonstrate that quantum-inspired simulators can surpass their classical counterparts in accuracy while using less than half the memory. Additionally, we show that operators based on higher-precision approximations can reduce errors in quantum-inspired simulations without compromising memory requirements. Finally, we explore the capability of quantum-inspired simulators to address memory-intensive problems beyond the reach of conventional algorithms.

**Keywords:** Tensor Networks, Partial Differential Equations, Numerical Simulations.
**SDGs:** 4, 9, 12.

## I. INTRODUCTION

Quantum computers can offer exponential speedups and memory reductions compared to their classical counterparts [1]. However, they fail to successfully allocate more than a few hundred qubits due to noise and decoherence, significantly limiting their current real-world applications. In this thesis, we focus on a line of research that has been recently gaining attention [2–4]: using tensor networks to imitate quantum registers on classical computers in order to address computationally expensive problems without requiring quantum hardware. In particular, in 2021 Garcia et al. [2, 4] introduced a new family of quantum-inspired (QI) algorithms using matrix-product states approximating low-entanglement states of quantum registers, including new algorithms for interpolating, performing quantum Fourier transforms, or solving differential equations. This set of techniques could pave the way to considerable memory and time reductions in classical computers. In this thesis we further explore the capabilities offered by QI algorithms, focusing on their ability to solve partial differential equations (PDEs) compared to their classical counterparts. In Section II we provide a brief introduction to tensor networks-based objects, and in Section III we describe their implementation for solving PDEs along with the potential memory advantages. In Section IV we study their performance in comparison to their classical counterparts using the diffusion equation as a benchmark problem. The results obtained lead us to explore two new directions in Subsections IV C and IV D.

## II. TENSOR NETWORKS

Assume we want to describe a one-dimensional function $f(x)$ where $x \in I = [a, b]$ for some real numbers $a < b$, and the interval is numerically represented with a regular grid with $2^N$ points. We refer to $N$ as the *discretization number.* In a standard algorithm on a classical computer, $f$ is represented with a $2^N$-element array and the time-evolution operator is a matrix with $2^{2N}$ elements, making the memory scale with $2^N$ or $2^{2N}$ depending on the problem. This makes many problems intractable on standard laptops for $N$ as low as 15–20. On the other hand, a quantum computer with $N$ qubits can store $2^N$ complex numbers as components of the quantum register wavefunction, given by $|\psi\rangle = \sum_{i=0}^{2^N-1} a_i |s_i\rangle$, where the sum goes over all $2^N$ basis states $|s_i\rangle$ and $a_i$ are the amplitudes associated. As a consequence, an exponential reduction of memory is obtained compared to a classical computer. To imitate this idea on a classical computer, we can approximate the representation of quantum states using Tensor Networks (TNs). To do so, we express each spatial location as $x = \frac{b-a}{2^N} s + b$, where $s = s_1 s_2 \dots s_N = \sum_{i=1}^{N} 2^{N-i} s_i$ and each $s_i$ is either 0 or 1. The function $f$ can then be seen as a quantum state $|f\rangle$ given by:

$$|f\rangle = \sum_{s_1,\dots,s_N} f_{s_1\dots s_N} |s_1, \dots, s_N\rangle, \tag{1}$$

where $f_{s_1\dots s_N} = f(x_{s_1\dots s_N})$. We can approximate this state with a product of $N$ tensors, where the value of $f$ at each point $s_1 \dots s_N$ is given by

$$|f\rangle_{s_1\dots s_N} = \sum_{\alpha} A^{s_1}_{\alpha_1} A^{s_2}_{\alpha_1,\alpha_2} \dots A^{s_N}_{\alpha_N}. \tag{2}$$

Each internal tensor $A^{s_k}_{\alpha_{k-1},\alpha_k} \in \mathbb{C}^{2\times\chi_{k-1}\times\chi_k}$ has internal indices $\alpha_{k-1}, \alpha_k$ (with *bond dimensions* $\chi_{k-1}$ and $\chi_k$, respectively), and physical index $s_k$ (with dimension 2). We refer to a representation of a function in the form (2) as a Matrix Product State (MPS). The contraction of an

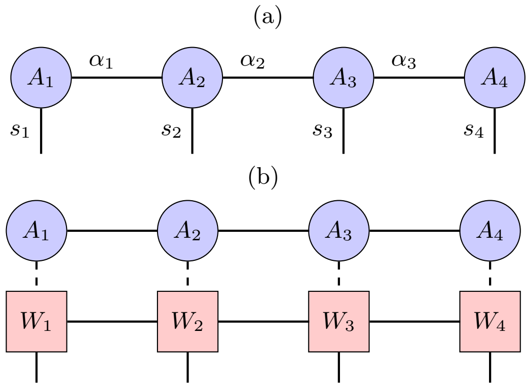

FIG. 1: (a) Diagrammatic representation of an MPS with open boundary conditions for $N = 4$, with physical indices $s_i$ and bond indices $\alpha_i$. (b) Contraction of an MPS (top) and MPO (bottom) with $N = 4$.

MPS is diagrammatically represented in Figure 1(a). The bond dimensions depend on the entanglement between tensors, and with sufficiently large bond dimensions it is always possible to find an exact MPS representation of a continuous function $f$ by recursive Schmidt decomposition [2]. However, in practical applications we want to bound the bond dimensions to maintain computational efficiency. Hence, in tensor network-based simulations a *maximal bond dimension* (MBD) $\chi$ is often imposed on the MPSs. Therefore, given an MBD $\chi$ the MPS requires $2(N-2)\cdot\chi^2+4\chi$ parameters. This represents a potential exponential gain of memory compared to the $2^N$ parameters used in a classical vector. Additionally, the MPS representation induces a similar representation for operators acting on those objects: the Matrix Product Operators (MPOs). These can be seen as the generalization of an MPS to the case of "matrices" that act on MPSs and consist of a contraction of $N$ tensors, where each internal tensor has two physical indices and two internal (bond) indices. Figure 1(b) shows the diagrammatic representation of a contraction between an MPS and MPO, resulting in a new MPS.

## III. QUANTUM-INSPIRED SIMULATORS

### A. Finite differences QI algorithm

Assume we want to solve the time evolution of a one-dimensional function $p(x,t)$, governed by a PDE of the form $\partial_t |p(t)\rangle = \hat{G}\,|p(t)\rangle$, where $\hat{G}$ is a linear operator. The finite differences QI algorithm enables us to solve this problem as follows [2]. First, to avoid potential exponentially growing numerical instabilities, we use the second-order implicit method

$$\frac{\partial}{\partial t}p(x,t) \simeq \frac{p(t+\delta t)-p(t)}{\delta t} \simeq \hat{G}\frac{(|p(t)\rangle + |p(t+\delta t)\rangle)}{2}. \tag{3}$$

After rearranging terms, this results in:

$$|p(t+\delta t)\rangle \simeq (1-\frac{\delta t}{2}\hat{G})^{-1}(1+\frac{\delta t}{2}\hat{G})\,|p(t)\rangle\,. \tag{4}$$

To implement equation (4) numerically, we first express the operators $\hat{G}_- = (1-\frac{\delta t}{2}\hat{G})$ and $\hat{G}_+ = (1+\frac{\delta t}{2}\hat{G})$ as MPOs. To do so, we express partial derivatives in terms of MPOs:

$$\partial_x |p\rangle \simeq \frac{S^+ - S^-}{2\delta x}\,|p\rangle\,,\ \partial_{xx}|p\rangle \simeq \frac{S^+ - 2\mathrm{Id} + S^-}{\delta x^2}\,|p\rangle\,,$$

where $S_+$ and $S_-$ are the up and down ladder operators, respectively (i.e., $S_\pm$ transforms $|p(x,t)\rangle$ into $|p(x\pm\delta x,t)\rangle$). Then, at each simulation iteration, the QI simulator works in two steps. In Step 1, we construct the MPS that results from the product $\hat{G}_+\,|p(t)\rangle = |\phi_1\rangle$, using simplification techniques to get the simplest and best approximation while keeping a bounded bond dimension. To do so, the tensors $\hat{G}_+$ and $|p(t)\rangle$ are first contracted, resulting in a new, larger MPS. To simplify this new MPS into one with smaller bond dimension, an iterative process is performed, sweeping from left to right and back on the MPS until convergence. At each step, a two-site tensor is split into two one-site tensors by orthonormalization. More precisely, the two-site tensor is reshaped into a matrix $M$ and the singular value decomposition is performed, yielding matrices $U, S,$ and $V$ such that $M = USV^t$, where $S$ has the singular values. By truncating $S$ below some threshold according to the maximal bond dimension and desired error, we obtain a new smaller tensor that is reinserted into the MPS. In general, this produces a new MPS with minimal error. Step 2 of the QI algorithm consists in finding a solution $|\phi_2\rangle$ to $|\phi_2\rangle = \hat{G}_-^{-1}\,|\phi_1\rangle$. Instead of applying $\hat{G}_-^{-1}$ to $|\phi_1\rangle$ (which requires finding the inverse of $\hat{G}_-^{-1}$ and is in general time-prohibitive), we follow the approach used by [2]. The method relies on iteratively optimizing a state $|\phi_2\rangle$ to find an optimal solution to $\hat{G}_-\,|\phi_2\rangle = |\phi_1\rangle$. Starting with an initial guess $|\phi_2\rangle = 0$, the MPS is recursively updated using the conjugate gradient method until reaching convergence or a maximum number of iterations. The final solution $|\phi_2\rangle$ corresponds to a close approximation of $|\phi(t+\delta t)\rangle$. In the following, we will refer to this method as the "QI" algorithm. For more details about the algorithms and code, see Appendix V A. On the other hand, we will compare the QI algorithm to its classical counterpart, the classical finite differences method, only differing in the representation of information: the MPS state is stored as a vector, and $\hat{G}_-$ and $\hat{G}_+$ are represented as two sparse matrices. As a consequence, in Step 1 of each evolution step, the vector $p(t)$ is multiplied by the matrix representing $\hat{G}_+$. In Step 2, the inverse problem $p(t+\delta t) = \hat{G}_-^{-1}p(t)$ is solved using the numerical solver `linalg.spsolve` from the `spicy` library. In the following, we refer to this method as the "Classical" algorithm.

### B. Memory advantages

In the QI method, the total number of parameters used is given by the shape of the MPS describing the state and the shapes of the two MPOs. Using a maximal bond dimension $\chi$, in the worst-case scenario, the numbers of parameters used are $2(N-2)\cdot\chi^2+4\chi$ for the MPS and $4(N-2)\cdot\chi^2+8\chi$ for each MPO, resulting in a total number of parameters $P_{\text{MPS}}+2P_{\text{MPO}}=10(N-2)\cdot\chi^2+20\chi$. Thus, the memory allocated scales linearly with $N$ (multiplied by a factor $10\chi^2$ that remains bounded during the simulation). On the other hand, the general Classical method employs $2^N$ parameters for representing the vector state and $(2^N)^2=2^{2N}$ parameters for each of the two matrices in the worst case. If we assume a relatively simple operator that can admit a sparse representation—as in the case of the diffusion equation—, the number of parameters for each sparse matrix can grow as $C\cdot 2^N$ for some constant $C$ (3 for first-order operators and 5 for second-order operators in the diffusion equation). Thus, the memory allocated by Classical approaches scales at least with $2^N$, which is exponentially worse than the QI method. We plot the memory usage as a function of discretization number for the diffusion equation in Figure 2. For $N\geq 12$ the QI method presents a memory reduction.

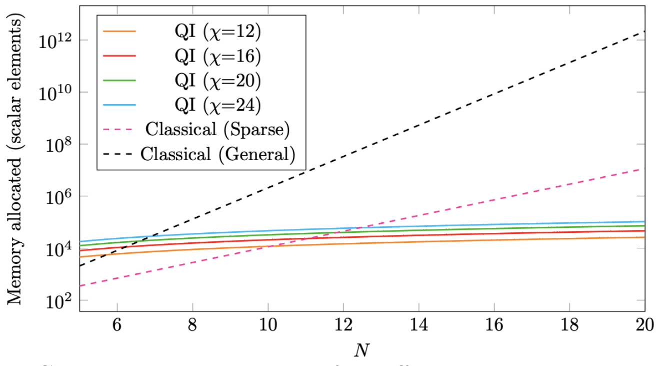

FIG. 2: Memory allocation for different simulators as a function of discretization number $N$.

## IV. APPLICATIONS: DIFFUSION EQUATION

We focus on a benchmark problem that represents an essential equation in physics: the one-dimensional diffusion equation, given by

$$\frac{\partial p(x,t)}{\partial t}=\alpha\frac{\partial^2 p(x,t)}{\partial x^2}, \tag{5}$$

where $\alpha$ is the diffusion coefficient. This equation describes the propagation of heat in a substance, where $p(x,t)$ represents the temperature at position $x$ and time $t$. Additionally, if the initial state $p(x,0)$ is a Gaussian distribution centered at 0 and with standard deviation $\sigma$, we can obtain the analytical solution, which is given by

$$p(x,t)=\frac{1}{\sqrt{2\pi(\sigma^2+2\alpha t)}}e^{-x^2/(2\sigma^2+4\alpha t)}. \tag{6}$$

### A. Simulations and accuracy

We solve the diffusion equation with $\alpha=15$ using the QI and Classical methods for discretization numbers $N=12,14,16$, as they represent the region where the QI method starts offering memory advantages. For each $N$, we run the QI simulation with MBDs $\chi=12,16,20,24$. In all cases we keep the spatial discretization to $\delta x=0.02$ and temporal discretization to $\delta t=0.033$ to avoid numerical instabilities. The initial state in all tests is a Gaussian distribution with standard deviation $\sigma=1$. For each $N$, we compare the outcomes of the five simulations to the analytical solution. For animations of the evolution of the states, see Appendix V B. We observe that while the QI solutions maintain the "physics" of the problem—a curve that flattens and decays to 0—, they decay to 0 much faster than expected. To formally compare the simulations to the true analytical solution, we calculate the time evolution of the relative error between two solutions $\phi_{\text{True}}$ and $\tilde{\phi}$ at time $t$, which we define as

$$\epsilon(t)=\frac{1}{\phi_{\text{True}}(0,t)}\cdot\sup_{x\in I}\left|\phi_{\text{True}}(x,t)-\tilde{\phi}(x,t)\right|,$$

where $I$ is the spatial interval where the state is defined. The results, shown in Figure 3, seem to indicate that a higher MBD corresponds to a lower error in the MPS solution. This assumption holds in general for the three values of $N$, indicating that MPSs with low MBD can not capture all the true information, leading to inaccuracies. Additionally, the MPS solution even outperforms the classical method for high enough MBD when $N$ is 12 and 14. Remarkably, for $N=14$ and $\chi=20$, the QI simulator achieves a lower error than the classical method while only employing 42% of the memory. However, indiscriminately increasing MBD is not always optimal, as an excessively high MBD may make the MPS capture too fine-grained information, leading to unrealistic solutions. This is the case of $\chi=24$ in Figure 3(b). Hence, appropriately selecting the MBD is a key aspect of each simulation.

### B. Time cost

Figure 4 shows the computation time per evolution step of the Classical and QI algorithms for the diffusion equation, using different MBDs $\chi$ and discretization numbers $N$. We have multiplied in the figure the times of the Classical method by 1000 to compare the scaling laws of the two algorithms. In fact, although for $N\leq 20$ the Classical method is orders of magnitude faster than the QI method, the dependences on $N$ are very different: the time for the Classical method seems to increase exponentially with $N$, while in the QI approach the computation time seems to increase linearly with $N$, with $\chi$ controlling the rate. These observations are in agreement with theoretical predictions: in the classical method, the time of each evolution step is dominated by Step 2 of the two-step algorithm, which is $\mathcal{O}(T_{\text{cgs}}2^{2N})$ [5], where $T_{\text{cgs}}$ is the number of iterations in the conjugate-gradient method and $2^{2N}$ is related to the cost of each matrix-vector multiplication. On the other hand, in the QI approach the time per step is also dominated by Step 2, and is $\mathcal{O}(T_{\text{cgs}}T_{\text{mult}})$ [2], where $T_{\text{mult}}$ is the cost of a single MPS-MPO multiplication and simplification. This

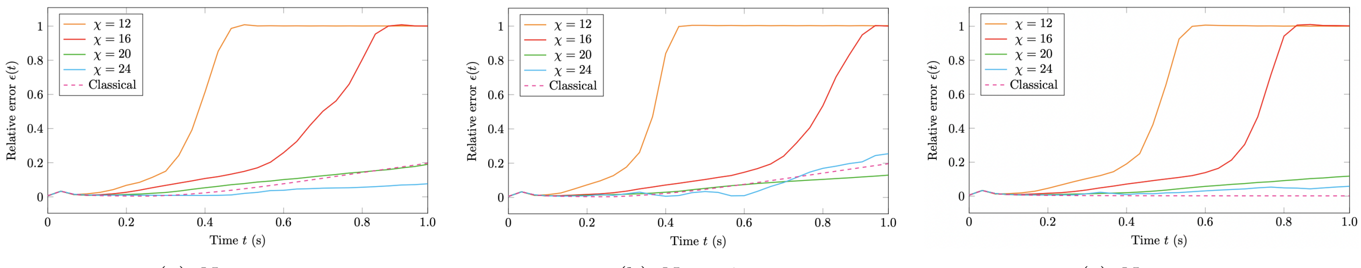

FIG. 3: Time evolution of the relative error throughout the QI simulation with different bond dimensions $\chi$, and the Classical simulation, for three different discretization numbers $N$.

time is $\mathcal{O}(T_{\text{sweeps}} N \chi^3)$ [6], where $T_{\text{sweeps}}$ is the number of sweeps in the simplification algorithm. As a consequence, assuming $\chi$ remains bounded, the computational time scales exponentially with $N$ for the classical finite differences method while it only scales linearly for the QI method. We would thus expect exponential gains in time and memory for the QI method with large enough (multi-dimensional) grids. Note that these scaling laws are general and not limited to the diffusion equation. However, in explicit problems, the requirement for high values of $\chi$ may offset the scaling advantages of the QI method.

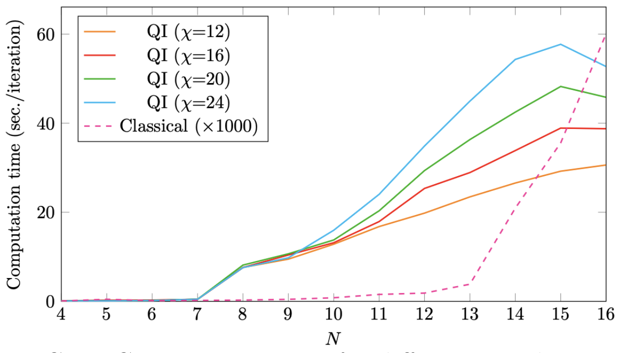

FIG. 4: Computation time for different simulation methods as a function of the discretization number $N$.

## C. A higher-precision QI solver

Although we have seen that high MBDs can reduce errors in the QI method, we want to find an alternative way of reducing errors without needing to increase MBD and reaching undesired high memory costs. To this aim, we construct a new higher precision (HP) QI solver. In particular, we use second-order approximations of the spatial derivatives, which involve calculating $f(x \pm 2\delta x)$. We use the fact that $f(x + 2\delta x)$ corresponds to two consecutive applications of the ladder operator $S^+$, i.e., $(S_+)^2 |f\rangle$, and analogously for $f(x - 2\delta x)$ and $S_-$. This allows us to construct the new operator for the diffusion equation:

$$\hat{G}_{\text{HP}} = \frac{\alpha}{(\delta x)^2}(16S_+ + 16S_- + \text{Id} - S_+^2 - S_-^2).$$

In this case, the error of the approximation is $\mathcal{O}(x^4)$, whereas the error of the previous operator was $\mathcal{O}(x^2)$. To test the efficiency of $\hat{G}_{\text{HP}}$ in comparison to the first-order operator, we perform a proof-of-concept experiment. We use a simulation from Figure 3 that yielded considerable error, and select the case $N = 12$, $\chi = 16$. Maintaining the same hyperparameters (time and space interval, initial state and diffusion coefficient), we run the HP-QI, QI, and Classical simulations. We plot in Figure 5 the resulting evolution of the relative error with respect to the true solution for the three algorithms. This shows that the relative error of the HP-QI method remains below 0.25 throughout the entire simulation, while the first-order QI simulation reaches an error of 1 after $\sim 0.8$ sec. In other words, we have achieved an error reduction by a factor of 4 using the same MBD. Nevertheless, the HP-QI method comes with a cost: it is 5 times more time-consuming than the QI algorithm. This is probably due to the fact that a more complex time operator leads to more iterations needed in the two-step algorithm at each evolution step.

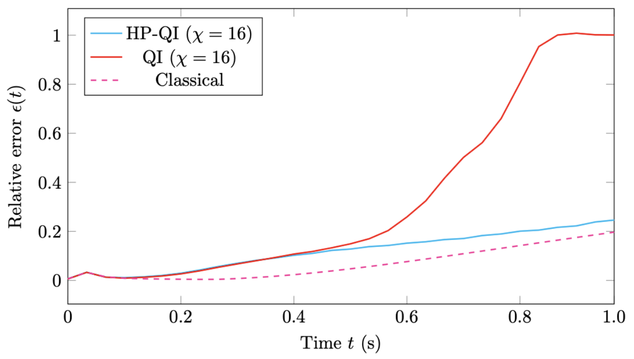

FIG. 5: Comparison of the HP-QI method to first-order QI and Classical methods, for $N = 12$.

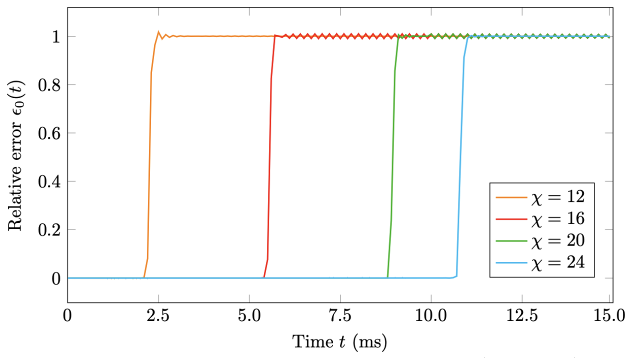

FIG. 6: Non-classical QI simulations ($N = 24$).

### D. Breaking the classical limit

Having seen that the QI method is impractical in small problems due to highly extensive runtimes, we aim to see how QI simulators behave in the limit where classical techniques fail due to memory constraints. In fact, a standard laptop can not simulate the temporal evolution of a function on a grid with $2^N$ points when $N > 14$, since it would require over 1 billion scalars for representing the matrix operators, surpassing memory capabilities. Even assuming a sparse representation, the numerical solver required for the classical finite differences approach, `linalg.spsolve`, crashes when $N \geq 24$, as it has to handle over 67 million parameters. On the other hand, when $N = 24$ the QI algorithm with MBD $\chi \leq 24$ requires less than 128,000 parameters, only 0.1% of the 117 million parameters required in the classical method. We thus aim to see if the QI algorithm can successfully simulate the time evolution of a function in this "non-classical" regime. To do so, we simulate the evolution of a Gaussian state with $\sigma = 1$ subject to the diffusion equation with $\alpha = 0.3$ and discretization number $N = 24$. Additionally, we reduce the temporal step to 0.0001 s to avoid numerical instabilities. To avoid the expensive process of computing and comparing the analytical solution over the entire grid, we compare the MPS solution to the analytical solution at $x = 0$. In particular, we track the relative error $\epsilon_0(t) = |\phi_{\text{QI}}(0,t) - \phi_{\text{True}}(0,t)| / \phi_{\text{True}}(0,t)$ across the simulation. We plot in Figure 6 the resulting curves for different MBDs over the first 150 iterations. Each simulation maintains the relative error below 1% for several iterations (over 100 for $\chi = 24$); however, the MPSs end up rapidly collapsing to 0 after a short time. This phenomenon occurs earlier for lower MBDs and can be delayed using higher MBDs, indicating that it may be due to a lack of information captured by small MPSs. This seems to indicate that the failure of the simulation is not caused by numerical instabilities, but rather by errors induced by truncations and approximations when simplifying MPSs. Furthermore, it is interesting to note that the behavior of functions in Figure 6 could be seen as the "asymptotic limit" of the behavior of low-MBD simulators of Figure 3. In that case, the collapse to 0 was due to weak approximations in small MPSs, but large enough MBDs were able to overcome that problem, even surpassing the classical method in terms of accuracy. Thus, we could expect a similar behavior for high-MBD simulators when $N = 24$. This is a promising conjecture, and we expect future research in this line, especially on large multi-dimensional grids.

## V. CONCLUSIONS

In this work, we have studied the ability of QI algorithms to solve PDEs. We have derived scaling laws for memory and time costs, showing potential exponential gains of memory and runtime when working with large grids and complex enough problems. Next, we have experimentally studied the ability of QI algorithms to simulate the time evolution of a Gaussian distribution subject to the diffusion equation. Our results have demonstrated that, if the MBD is carefully selected, the QI simulator can yield higher accuracy than its classical counterpart while only employing 42% of the memory. To reduce errors without the need for higher MBDs, we have developed a higher-order simulator and showed in a proof-of-concept example that it can reduce simulation error by a factor of 4 compared to the first-order approach. Despite these promising results, QI simulators remain impractical in small problems due to time-intensive runtimes. As a consequence, we have tested the efficacy of QI algorithms in the limit where classical approaches fail due to memory constraints. The results obtained indicate that the QI method can effectively simulate a true property of the state for short durations, however, it collapses to 0 due to weak approximations in MPS simplification. Nevertheless, the correlation we have found between MBD and accuracy is a promising result, indicating that high-MBD QI algorithms could be used in the future on complex high-dimensional problems, potentially addressing problems currently intractable in classical computers.

**Acknowledgments**

I would like to thank Dr. Bruno Julià-Díaz, Dr. Artur García-Saez, and Dr. Stefano Carnignano for their guidance and advice. Additionally, I am deeply grateful to the Barcelona Supercomputing Center for enabling me to conduct my thesis with the Quantic group.

[1] Peter W Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th annual symposium on foundations of computer science*, pages 124–134. Ieee, 1994.

[2] Juan José García-Ripoll. Quantum-inspired algorithms for multivariate analysis: from interpolation to partial differential equations. *Quantum*, 5:431, 2021.

[3] Martin Kiffner and Dieter Jaksch. Tensor network reduced order models for wall-bounded flows. *arXiv preprint arXiv:2303.03010*, 2023.

[4] Paula García-Molina, Luca Tagliacozzo, and Juan José García-Ripoll. Global optimization of mps in quantum-inspired numerical analysis. *arXiv preprint arXiv:2303.09430*, 2023.

[5] William H Press, Saul A Teukolsky, William T Vetterling, and Brian P Flannery. Numerical recipes (cambridge, 1992.

[6] Román Orús. A practical introduction to tensor networks: Matrix product states and projected entangled pair states. *Annals of physics*, 349:117–158, 2014.

# Xarxes Tensorials per Simulacions Inspirades per la Quàntica

Author: Jack Benarroch Jedlicki, jbenarje7@alumnes.ub.edu
*Facultat de Física, Universitat de Barcelona, Diagonal 645, 08028 Barcelona, Spain.*

Advisors: Bruno Julià-Díaz (bruno@fqa.ub.edu), Artur García-Saez, Stefano Carignano

**Resum:** Els algorismes quàntics tenen el potencial d'accelerar la computació i reduir requisits de memòria en ordinadors quàntics prou avançats. Tanmateix, limitacions en maquinari quàntic dificulten la seva aplicació a problemes complexos. En aquest treball, investiguem un enfocament prometedor que evita la necessitat de maquinari quàntic utilitzant xarxes tensorials per simular algorismes quàntics en ordinadors clàssics. Avaluem el rendiment de simuladors inspirats en la quàntica en relació amb mètodes clàssics en termes de memòria, temps d'execució i precisió. Els resultats demostren que els simuladors inspirats en la quàntica poden superar els seus homòlegs clàssics en precisió tot utilitzant menys de la meitat de la memòria. A més, mostrem que operadors basats en aproximacions de major precisió poden reduir errors en simulacions inspirades en la quàntica sense comprometre els requisits de memòria. Finalment, explorem la capacitat dels simuladors inspirats en la quàntica per abordar problemes de gran demanda de memòria que estan fora de l'abast d'algorismes convencionals.

**Paraules clau:** Xarxes Tensorials, Equacions en Derivades Parcials, Simulacions Numèriques.
**ODSs:** 4, 9, 12.

**Objectius de Desenvolupament Sostenible (ODSs o SDGs)**

| | | | |
|---|---|---|---|
| 1. Fi de la es desigualtats | | 10. Reducció de les desigualtats | |
| 2. Fam zero | | 11. Ciutats i comunitats sostenibles | |
| 3. Salut i benestar | | 12. Consum i producció responsables | X |
| 4. Educació de qualitat | X | 13. Acció climàtica | |
| 5. Igualtat de gènere | | 14. Vida submarina | |
| 6. Aigua neta i sanejament | | 15. Vida terrestre | |
| 7. Energia neta i sostenible | | 16. Pau, justícia i institucions sòlides | |
| 8. Treball digne i creixement econòmic | | 17. Aliança pels objectius | |
| 9. Indústria, innovació, infraestructures | X | | |

El contingut d'aquest TFG, part d'un grau universitari de Física, es relaciona amb l'ODS 4, ja que contribueix a l'educació a nivell universitari en relació amb les xarxes tensorials i simulacions inspirades per la quàntica. També es relaciona amb l'ODS 9, ja que s'estudien mètodes per realitzar simulacions amb potencials estalvis exponencials de memòria, el qual podria contribuir a la realització eficient de simulacions computacionalment complexes en acadèmia o indústria sense la necessitat de grans infraestructures computacionals. A més, els mètodes tractats en aquest TFG poden contribuïr a tractar problemes actualment intractables computacionalment, el qual també els relaciona amb la innovació mencionada a l'ODS 9. Aquest TFG també es relaciona amb l'ODS 12, ja que promou l'ús responsable d'energia i recursos computacionals a través de l'ús de xarxes tensorials i algorismes inspirats per la quàntica.

## SUPPLEMENTARY MATERIAL

### A. Algorithms and code

The algorithms and code have been made available in the repository https://github.com/JackBJ23/Quantum-Sim, which includes a library we have developed, `quantumsim`, that allows the direct installation of quantum-inspired simulators for the diffusion equation. It builds upon previous work on tensor networks by Garcia et al. [2, 4], and provides new algorithms for efficient implementation on large grids, new operators for the case of the diffusion equation, new operators based on higher-order approximations, and implementations in limits where classical methods fail with conventional approaches. It also includes tools for visualization. We have also included in the repository the main code used for running the experiments.

### B. Visualizations of simulations

We have made available animations of the simulations performed in Section IV A in order to give a more insightful understanding of the dynamics of the solutions. They can be found in https://github.com/JackBJ23/Quantum-Sim. In particular, evolutions_N12.gif corresponds to the time evolutions for $N = 12$, evolutions_N14.gif corresponds to $N = 14$, and evolutions_N16.gif corresponds to $N = 16$. Additionally, we provide visualizations of the collapses of functions in Figure 6, which we believe is an interesting phenomenon that should be further studied. The animations are given for the two extreme value cases: evolution_N24_MBD12.gif for $\chi = 12$, and evolution_N24_MBD24.gif for $\chi = 24$.