Supplemental: Hybrid classical-quantum computation of heat diffusion in multilayer materials

Sasan Moradi

Center for medical physics and biomedical engineering, Medical university of Vienna, Währinger Straße, Vienna, 1090, Vienna, Austria.

Corresponding author(s). E-mail(s): sassan.moradi@gmail.com;

The math and details are in support of the main text for "Hybrid classical-quantum computation of heat diffusion in multilayer materials"

Appendix A Nondimensionalization of heat diffusion equation

Consider the dimensional diffusion equation,

$$\frac{\partial U_i}{\partial t} = D_i \frac{\partial^2 U_i}{\partial x^2} \tag{A1}$$

using standard non-dimensionalisation, let

$$t = t_0 t^*, U_i = U_0 U_i^*, x = L x^* \tag{A2}$$

t_0 is the typical time scale, U_0 is the typical temperature, and L is the typical length. Substituting these into Eq. A1, we have

$$\frac{\partial U_i^*}{\partial t^*} = D_i \frac{t_0}{L^2} \frac{\partial^2 U_i^*}{\partial x^{*2}} \tag{A3}$$

However since D_i is a function of space, this must be done carefully. Let $D_i = D_{max} D_i^*$ where D_{max} is the largest diffusivity, such that $0 \le D_i^* \le 1$ and $t_0 = \frac{L^2}{D_{max}}$

$$\frac{\partial U_i^*}{\partial t^*} = D_i^* \frac{\partial^2 U_i^*}{\partial x^{*2}} \tag{A4}$$

Appendix B Matrix-Vector multiplication

We first start with small size matrix 4×4 and vector 4×1. Let's begin by performing the following matrix-vector multiplication for 4×4 matrix and 4×1 vector

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \tag{B5}$$

and $\mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix}$. Let's begin by introducing the following definitions:

$$\begin{aligned} \vec{A}_1 &= (a_{11}, a_{12}, a_{13}, a_{14}) \\ \vec{A}_2 &= (a_{21}, a_{22}, a_{23}, a_{24}) \\ \vec{A}_3 &= (a_{31}, a_{32}, a_{33}, a_{34}) \\ \vec{A}_4 &= (a_{41}, a_{42}, a_{43}, a_{44}) \end{aligned} \tag{B6}$$

This allows us to express the matrix-vector multiplication as

$$Ab = \begin{bmatrix} A_1.b \\ A_2.b \\ A_3.b \\ A_4.b \end{bmatrix} = \begin{bmatrix} a_{11}b_1 + a_{12}b_2 + a_{13}b_3 + a_{14}b_4 \\ a_{21}b_1 + a_{22}b_2 + a_{23}b_3 + a_{24}b_4 \\ a_{31}b_1 + a_{32}b_2 + a_{33}b_3 + a_{34}b_4 \\ a_{41}b_1 + a_{42}b_2 + a_{43}b_3 + a_{44}b_4 \end{bmatrix} \tag{B7}$$

For each row, we can define the following states

$$\begin{aligned} |A_1\rangle &= a_{11}|00\rangle + a_{12}|01\rangle + a_{13}|10\rangle + a_{14}|11\rangle, \sum_{j=1}^{4} |a_{1j}|^2 = 1 \\ |A_2\rangle &= a_{21}|00\rangle + a_{22}|01\rangle + a_{23}|10\rangle + a_{24}|11\rangle, \sum_{j=1}^{4} |a_{2j}|^2 = 1 \\ |A_3\rangle &= a_{31}|00\rangle + a_{32}|01\rangle + a_{33}|10\rangle + a_{34}|11\rangle, \sum_{j=1}^{4} |a_{3j}|^2 = 1 \\ |A_4\rangle &= a_{41}|00\rangle + a_{42}|01\rangle + a_{43}|10\rangle + a_{44}|11\rangle, \sum_{j=1}^{4} |a_{4j}|^2 = 1 \\ |b\rangle &= b_1|00\rangle + b_2|01\rangle + b_3|10\rangle + b_4|11\rangle, \sum_{i=1}^{4} |b_i|^2 = 1 \end{aligned} \tag{B8}$$

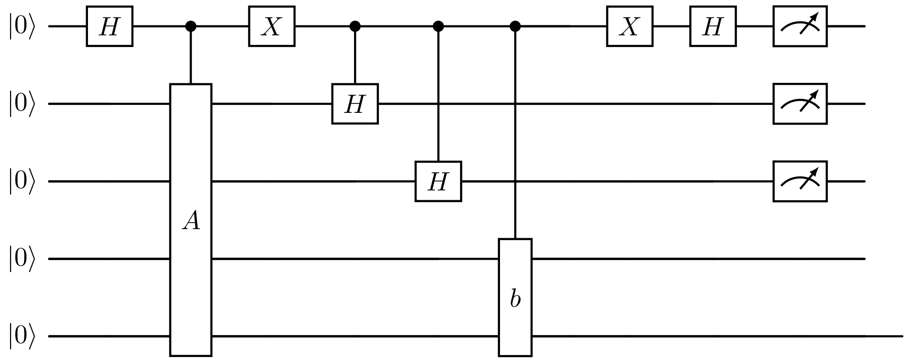

Fig. B1 Hadamard Test quantum circuit to generate the matrix-vector multiplications, where matrix is a 4×4 and vector is a 4×1. H is a Hadammard gate and X gate is PauliX gate. The lower quantum wires carries the register qubits, the upper wires are the ancilla qubits. If the ancilla registers are measured, the probabilities of measuring ancilla qubits give the values of matrix-vector multiplication as given in Eq. B14.

Then we have the whole matrix as a state vector

$$|A\rangle = \frac{1}{2}(|00\rangle|A_1\rangle + |01\rangle|A_2\rangle + |10\rangle|A_3\rangle + |11\rangle|A_4\rangle, \tag{B9}$$

If you have a 4×4 matrix and a 4×1 vector, you must redefine the vector $|b\rangle$ to have a size of 4×4 in order to design the quantum circuit for matrix-vector multiplication. This can be represented as

$$|\bar{b}\rangle = \frac{1}{2}(|00\rangle|b\rangle + |01\rangle|b\rangle + |10\rangle|b\rangle + |11\rangle|b\rangle), \tag{B10}$$

The next step is to generate the following quantum state using a Hadamard Test quantum circuit, as illustrated in Figure. B1

$$|\Phi\rangle = \frac{1}{2}|0\rangle(|A\rangle + |\bar{b}\rangle) + \frac{1}{2}|1\rangle(|A\rangle - |\bar{b}\rangle), \tag{B11}$$

By inserting Eqs. B9 and B10 in Eq. B11, we derive Eq. B12

$$\begin{aligned}|\Phi\rangle =& \frac{1}{4}|0\rangle(|00\rangle(|A_1\rangle + |b\rangle) + |01\rangle(|A_2\rangle + |b\rangle)\\ &+ |10\rangle(|A_3\rangle + |b\rangle) + |11\rangle(|A_4\rangle + |b\rangle))\\ &+ \frac{1}{4}|1\rangle(|00\rangle(|A_1\rangle - |b\rangle) + |01\rangle(|A_2\rangle - |b\rangle)\\ &+ |10\rangle(|A_3\rangle - |b\rangle) + |11\rangle(|A_4\rangle - |b\rangle))\end{aligned}, \tag{B12}$$

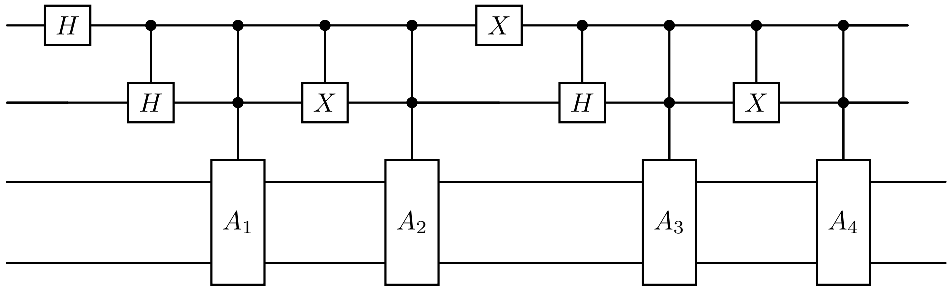

Fig. B2 Quantum circuit to generate the matrix A given in Eq. B9. A_1, A_2, A_3, and A_4 are row vectors that are normalized and then encoded using the divide and conquer approach in this reference [1].

If matrix A and vector b are real-valued, the probability of measuring the basis states $|000\rangle$, $|001\rangle$, $|010\rangle$, and $|011\rangle$ for the first three qubits can be determined by

$$\begin{aligned} Pr(|000\rangle) &= \frac{1}{8}(1 + \langle A_1|b\rangle) \\ Pr(|001\rangle) &= \frac{1}{8}(1 + \langle A_2|b\rangle) \\ Pr(|010\rangle) &= \frac{1}{8}(1 + \langle A_3|b\rangle) \\ Pr(|011\rangle) &= \frac{1}{8}(1 + \langle A_4|b\rangle) \end{aligned}, \tag{B13}$$

Using Eq.B13, the inner product $\langle A|b\rangle$ can be obtained

$$\begin{aligned} \langle A_1|b\rangle &= 8Pr(|000\rangle) - 1 \\ \langle A_2|b\rangle &= 8Pr(|001\rangle) - 1 \\ \langle A_3|b\rangle &= 8Pr(|010\rangle) - 1 \\ \langle A_4|b\rangle &= 8Pr(|011\rangle) - 1, \end{aligned} \tag{B14}$$

The values of probabilities in Eq. B14 can be obtained using a quantum simulator or from a NISQ quantum machine.

Appendix C Divide and Conquer loading an $N \times N$ matrix and an $N \times 1$ vector

Basically we encode the matrix elements by the calculation of rotation angles, the creation of a binary tree (Btree) and traversing the Btree in preorder strategy to arrange the set of quantum gates (X(NOT) gate, $CNOT$, controlled Hadamard, and controlled rotation R_y).

For this, we implement recursion, an approach of divide and conquer strategy to obtain the rotation angles. The root and leaves of the tree are represented by a data structure, containing an arbitrary data value, a level of depth and the subsequent

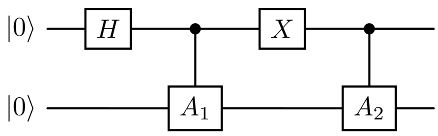

Fig. C3 Quantum circuit for encoding matrix $A_{2\times 2}$. The quantum state representing the matrix $A_{2\times 2}$ is $|A_{12}\rangle = \frac{1}{\sqrt{2}}(|0\rangle\,|A_1\rangle + |1\rangle\,|A_2\rangle)$

branches. The tree is filled recursively for as many elements as available in the passed row vector of the matrix. In our case this is $2N_x - 1$ locations. The preorder traverse is defined as a type of tree traversal that follows the Root-Left-Right policy. The root node of the subtree is visited first, then the left subtree and at last the right subtree is traversed. The preorder traverse is also recursive. The arguments are: the quantum circuit which is going to be built, the current node inside the binary tree pointed to, the offset index (qubit index) to the lowest significant quantum bit (usually 1), the index of the vector of angles inside the vector field, the data structure which stores the calculated angles. To encode a matrix $A_{N\times N}$, where N is a big number, we begin with designing the simplest quantum circuit for encoding the matrix $A_{2\times 2}$ as depicted in Fig. C3.

For instance, to encode

$$A_{4\times 4} = \begin{bmatrix} A_1 \\ A_2 \\ A_3 \\ A4 \end{bmatrix}$$

, first, we divide the input into two sub-matrices

$$A_{12} = \begin{bmatrix} A_1 \\ A_2 \end{bmatrix}$$

and

$$A_{34} = \begin{bmatrix} A_3 \\ A_4 \end{bmatrix}$$

, where

$$A_{4\times 4} = \begin{bmatrix} A_{12} \\ A_{34} \end{bmatrix}$$

. The first matrix A_{12} includes rows A_1 and A_2 and the second matrix A_{34} includes rows A_3 and A_4

$$|A_{4\times 4}\rangle = \frac{1}{2}(|0\rangle\,|A_{12}\rangle + |1\rangle\,|A_{34}\rangle) \tag{C15}$$

$A_{4\times 4}(A_{1234})$ is the root of the tree and A_{12} and A_{34} are the left leaf and the right leaf of the tree, respectively. Fig. C5 shows the quantum circuit for the encoding $A_{4\times 4}$. With the same procedure, we can encode a matrix $A_{N\times N}$. First, we divide the input matrix into two matrices. The first matrix contains the first $N/2$ rows, while the second matrix includes the next $N/2$ rows of the matrix. This division process continues until each leaf contains a matrix with only two rows and N columns. The depth of the tree is $log_2(N)-1$, the number of edges are $N/4(N/4+1)$ and the number of leaves are $N/2$. If $A_{N\times N}$ is a sparse matrix with s as the number of nonzero values and $s << N$, then we need $log_2(s)$ number of qubits to encode the nonzero values and $n = log_2(N)$ qubits for addressing the edges [1], while for a dense matrix $A_{N\times N}$, $2n$ qubits are utilized for encoding. For a sparse vector with s nonzero values, $s-2$ two-qubit gates and $s-1$ single-qubits gates are required. Therefore, the computational cost of encoding an s-sparse matrix $N \times N$ has an order $O(sN)$, while the order for an $N \times N$ dense matrix is at the best case $O(N^2)$.

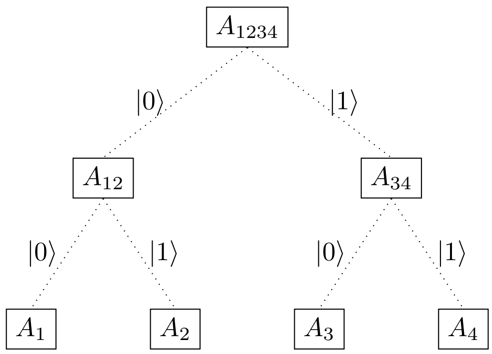

Fig. C4 A binary tree with roots and leaves. The main root is a matrix A_{1234} ($A_{4\times4}$) and its leaves are matrices A_{12} and A_{34}. Each of which has two rows and four columns. The matrix A_{12} is encoded by using the quantum circuit in Fig. C3. By replacing A_1 with A_3 and A_2 with A_4 in Fig. C3, the matrix A_{34} is also encoded. Finally, the quantum circuit in Fig. C5 encodes the $A_{4\times4}$. The preorder traversal follows the Root-Left-Right policy. At first the root will be visited, i.e. node A_{1234}. After this, traverse in the left subtree. Now the root of the left subtree is visited i.e., node A_{12} is visited. Again the left subtree of node A_{12} is traversed and the root of that subtree i.e., node A_1 is visited. There is no subtree of A_1 and the left subtree of node A_{12} is visited. So now the right subtree of node A_{12} will be traversed and the root of that subtree i.e., node A_2 will be visited. The left subtree of node A_{1234} is visited. So now the right subtree of node A_{1234} will be traversed and the root node i.e., node A_{34} is visited. Now the left subtree of node A_{34} is traversed and the root of that subtree i.e., node A_3 is visited. There is no subtree of A_3 and the left subtree of node A_{34} is visited. So the right subtree will be traversed and the root of the subtree i.e., node A_4 will be visited. After that there is no node that is not yet traversed. So the traversal ends.

For encoding an $N \times 1$ vector, we used the encoding approach implemented in this reference [2].

Appendix D Flowchart for simulation of heat diffusion in 1D

Basically, creating the quantum circuit for matrix-vector multiplication involves arranging quantum gates in a specific order. This arrangement can be done either manually, by sequentially placing gates, or algorithmically, by looping through qubits and positioning gates with well-defined behaviors. Before generating any quantum circuits, first, the sparse diffusion matrix is prepared by introducing the diffusion coefficients and constructing the main diagonal and side diagonals. The initial vector with

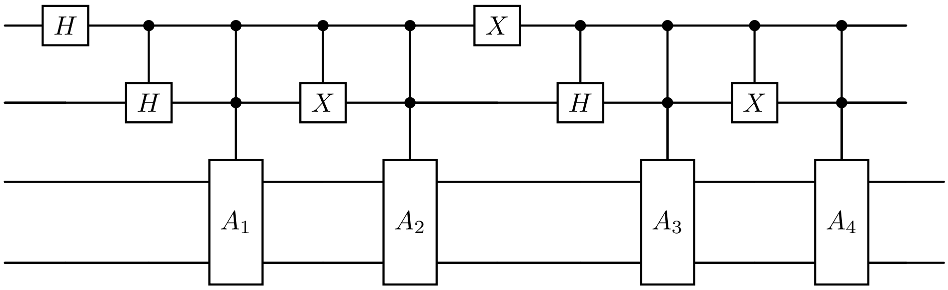

Fig. C5 Quantum circuit to generate the matrix A_{1234} which is a 4×4 matrix. The quantum state representing the matrix $A_{4\times4}$ is $|A_{1234}\rangle = \frac{1}{2}(|0\rangle|A_{12}\rangle + |1\rangle|A_{34}\rangle) = \frac{1}{2}(|00\rangle|A_1\rangle + |01\rangle|A_2\rangle + |10\rangle|A_3\rangle + |11\rangle|A_4\rangle)$.

its starting intensity is also prepared classically. Then the rotation angles for encoding the matrix A into quantum space are approximated, but this is done for a single time step. Since the result vector, or $U(t)(= b)$, changes with each iteration, normalization and the calculation of angles for the rotation gates will be performed in each iteration. Each result vector serving as the input for the next iteration. When solving the heat equation in quantum space, we typically begin with an initial quantum state of zero and iterate through the discretized location points. The number of qubits required to define the size of the quantum circuit and the quantum state are calculated based on the number of location points.

The quantum circuit for MVM is the Hadamard test. First a Hadamard gate (H) is applied on the first ancilla qubit, then the sparse diffusion matrix A is encoded (see Appendix C). Next an X gate as part of the Hadamard test is applied. The vector $U(t)$ does not require as many qubits as the matrix for encoding. The vector must be encoded in a higher dimensional Hilbert space. This is done by applying H gates on the ancilla qubits that are not used for the encoding $U(t)$. The application of X gate uncomputes the first ancilla qubit. The Hadamard Test circuit concludes with the application of the $H^\dagger$ gate. $H^\dagger = H$ holds in the case of qubits. By measuring the $log(N) + 1$ ancilla qubits, the measurement probabilities in different basis states yield the elements of the matrix-vector multiplication (MVM). The output vector is $U(t + \Delta t)$. Fig. D6 in Appendix B shows the flowchart for simulating heat diffusion in 1D space.

References

[1] Park, D.K., Petruccione, F., Rhee, J.-K.K.: Circuit-based quantum random access memory for classical data. Scientific Reports **9**(1), 3949 (2019) https://doi.org/10.1038/s41598-019-40439-3

[2] Araujo, I.F., Park, D.K., Petruccione, F., Silva, A.J.: A divide-and-conquer algorithm for quantum state preparation. Scientific Reports **11**, 6329 (2021) https://doi.org/10.1038/s41598-021-85474-1

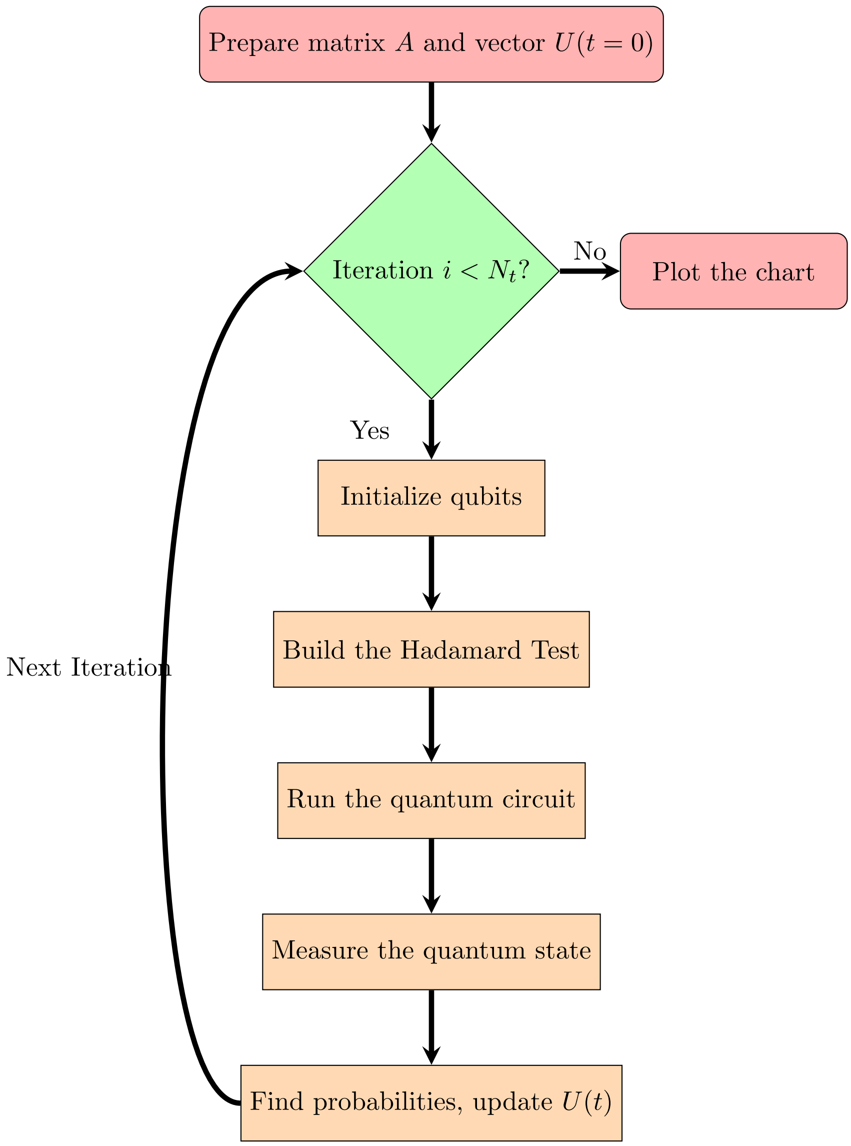

Fig. D6 Flowchart for simulation of heat diffusion in 1D. It illustrates the iterative process for matrix preparation, quantum circuit execution, and result analysis. First, the tridiagonal matrix A and the vector representing the heat source $U(t)$ at $t = 0$ are prepared in a classical computer. Then the rotation angles are calculated using the divide and conquer algorithm [1]. At the next step, the iteration is started. Inside the loop, the quantum circuit for the multiplication of matrix A with vector $U(t = 0)$ is designed. The qubits are initialized at $|0\rangle^{\otimes(2n+1)}$. Then the sequence of Hadamard, R_y rotation, and X gates are applied. This quantum circuit is the Hadamard Test. The quantum circuit runs to prepare the quantum state. The quantum state is measured on the first $log_2 N + 1$ qubits. Finally, the probabilities of measurement are approximated. The outcomes of the probabilities of measurement are the new vector $U(t)$ at $t = \Delta t$. The new vector $U(t)$ must be normalized with a classical computer. The iteration process continues $i = N_t - 1$ times. For $i = N_t$, the iteration is terminated and the chart is plotted.