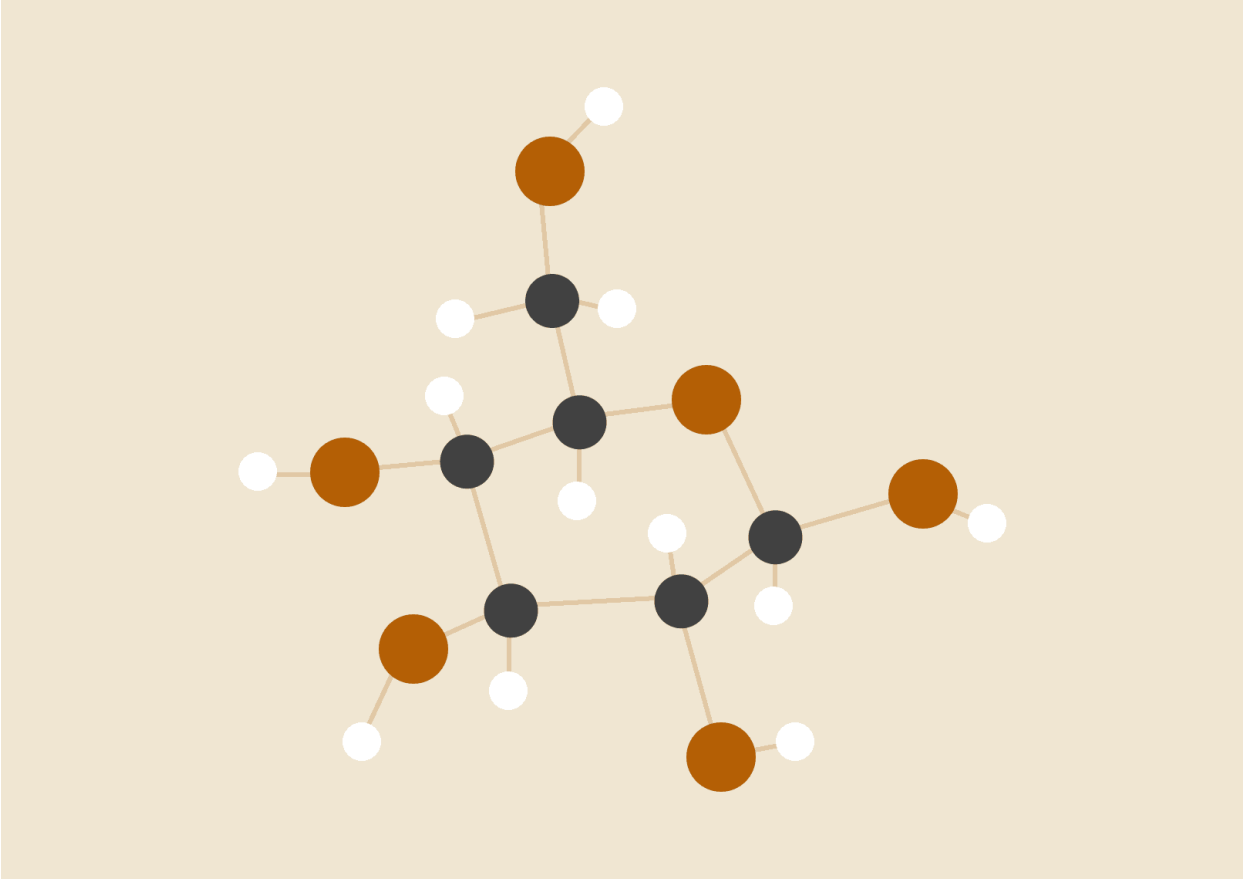
SOLVING HARMONIC OSCILLATOR

Using Suzuki-Trotter Decomposition to Simulate Time Evolution



By Team Duality

09.28.2024

FROM THE PAPER

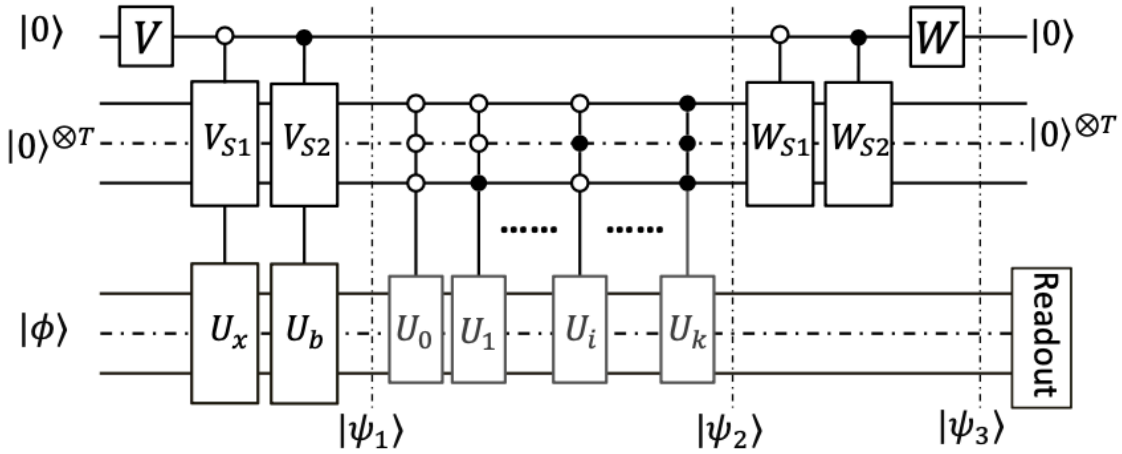
The paper given shows the algorithm of solving a linear differential equation in the general form:

$$\frac{dx(t)}{dt} = Mx(t) + b$$

The solution to this involves the matrix exponential e^{Mt}. In the paper, this is approximated by using a Taylor Series expansion. This algorithm constructs a circuit that simulates the power of matrix M.

The vector can be represented using a series of quantum states(qubits), and the matrix M can be represented by a unitary operator, which gives us insight on building a time evolution operator.

DEEP DIVE INTO THE CIRCUIT



The initial state $x(0)$ and b are stored in the first register qubit, a series of ancilla qubits are stored in the second register, lastly work qubit is represented by $|\phi>$. The first controlled operation U_x and U_b entangle the first register qubit with the first ancilla qubit, this controls whether the work qubit evolve according to $x(0)$ or b. The Taylor coefficients of $x(0)$ are stored in the first row of matrix V_{S1} and the coefficients of b are

stored in the first row of matrix V_{S2}. After operation U_x and U_b, the zeroth order terms of the Taylor Series are introduced . V_{S1} and V_{S2} can be thought of as arms fetching the Taylor coefficients and placing them into our quantum state. The subsequent controlled operation entangles the first qubit with the following ancilla qubits one by one, introducing higher order Taylor terms one by one. At the end the entire Taylor series is represented with a global superposition(Eq.7)

APPLY TO HO

The given Harmonic Oscillator:

$\frac{d^2y}{dt^2} + \omega^2 y = 0,\ \omega = 1$

Boundary condition: $y(0) = 1,\ \frac{dy}{dt}|_{t=0} = 1.$

Converting this second order equation into the first order, we get:

$$\frac{d}{dt}\begin{pmatrix} x_1(t) \\ x_2(t) \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}\begin{pmatrix} x_1(t) \\ x_2(t) \end{pmatrix}$$

And therefore,

$M = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$ and $\vec{b} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$

Now the above equation can be solved using the method described in the paper.

Solving the original HO equation on paper, gives us a solution of :

$$y(t) = cos(t) + sin(t)$$

This equation is the represented in the form of $x(t) = e^{Mt}x(0) + (e^{Mt} - 1)M^{-1}b$, where $x(t) = (y(t)\ ,\ y'(t))^T$, and the values of M and b are defined above

Taking b=0, gives us:

$x(t) = e^{Mt}x(0)$

Thus, the problem is now converted to simulating the time-evolution of a Hamiltonian e^{-iHt}, in the equation $A|\psi\rangle = e^{-iHt}|\psi\rangle$, where $H = M$

TIME-EVOLUTION OF KINETIC AND POTENTIAL ENERGY

To evaluate the kinetic energy and potential energy as a function of time in the time interval [0,1]:

1. We choose a Hamiltonian operator as our time evolution operator here, the Hamiltonian operator can be separate into a potential energy operator and a kinetic energy operator:
   $\widehat{H} = \widehat{V} + \widehat{T}$, where
   $\widehat{V} = cos^2(t) \Rightarrow$ PE operator
   $\widehat{T} = sin^2(t) \Rightarrow$ KE operator
   Therefore, $\widehat{H} = cos^2(t) + sin^2(t)$, this equation represents the total energy of the harmonic system.

   Time evolution operator can be written as:
   $U(t) = e^{-i\widehat{H}t/h_{bar}}$, where $\widehat{H}$ has been defined above.

   We have chosen these functions for the KE and PE operators based on the fact that $cos^2(t)$ and $sin^2(t)$ vary sinusoidally and as one increases, the other one decreases, emulating the behavior of the kinetic energy and potential energy of a Classical Harmonic Oscillator(CHO).
   Moreover, the sum of the KE and PE operators ie. $sin^2(t)$ and $cos^2(t)$, comes out to be 1, ie. constant, similar to how the energies in a mechanical system always remain constant classically.

2. We now convert our operators into the Quantum Gate form, ie. we use Pauli Gates(I,X,Y,Z) to represent our energy operators

   We represent our PE operator as 'ZI' + 'IZ'

   - The reason being that Pauli-Z distinguishes between the two energy levels in a two-level system, subsequently, it is convenient to map the position of the particles to the eigenvalues of Z (position of the particle is reflected in the eigenvalue +1 or −1)

   - This can be understood by looking at the following equations:

- $Z|0\rangle = |0\rangle$ and $Z|1\rangle = -|1\rangle$
- The addition sums up the contributions from two different sites or qubits, representing the potential energy operator.

On the other hand, we represent our KE operator as 'XI' + 'IX'

- Since we are measuring the KE operator in the momentum basis, and the basis is conjugate to the position basis, we use Pauli-X to flip between the eigenstates of the Pauli-Z(which is measured in the position basis), similar to how the position and momentum operators counter each other in the real world.
- This can be understood by looking at the following equations:
- $X|0\rangle = |1\rangle$ and $X|1\rangle = |0\rangle$
- Again, the addition sums up the contributions from two different sites or qubits, representing the kinetic energy operator.

Since the KE and PE operators do not commute, ie. $[P, Y] = PY - YP \neq 0$**, therefore we apply the Trotter-Suzuki decomposition to reduce the operators**

SUZUKI-TROTTER DECOMPOSITION

Suzuki-Trotter Decomposition can be written as:

$U(t) \approx (e^{-i\hat{V}dt}e^{-i\hat{T}dt})^n$, $n = time\ steps$, $dt = \frac{t}{n}$.

- Potential energy is mapped to position basis, we represent this by Pauli-Z matrices:

  $V = 2(Z_1 + Z_2)$

- Corresponding to momentum basis, we represent KE operator by Pauli-X matrices:

  $T = \frac{1}{2}(X_1 + X_2)$

CODE:

We have executed two parts here

- One for simulating the output of the given harmonic oscillator equation
- And the other one for graphically representing the time-evolution of the Kinetic and potential energies of the system.

For both the cases, we use a 2-qubit system

- In the first case, we use a 2-qubit system to represent the intricacies of the function $y(t) = cos(t) + sin(t)$, as this is now converted to optimizing the simulation of a function, and using a single qubit does give the correct output but the graph is very smooth, not capturing the fine details. The qubits are also used to initialize the values of $x(0)$ and b.

- In the second case, we use 2 qubits to represent the two energy levels of a system, like Spin states as well as Two Coupled particles

1. Solution to the HO equation
    - We have already defined the matrix M to be used in the time-evolution e^{Mt}, since we are using a two qubit system, we have $2^2 = 4$ total states, and we have the following equation:

$$|\psi\rangle = cos(\theta_1)|00\rangle + sin(\theta_1))cos(\theta_2))|01\rangle + sin(\theta_1))sin(\theta_2))cos(\phi)|10\rangle + sin(\theta_1))sin(\theta_2))sin(\phi)|11\rangle$$

    - Thus we define the parameters θ_1, θ_2 and ϕ as

```
# Define the angles for the two-qubit state
theta1 = np.pi / 4
theta2 = np.pi / 4
phi = np.pi / 2
```

    - We initialize the time evolution operator as e^{-iHt}, where$H = U = iM$(we have multiplied with i to make it a unitary complex matrix

```
# Define the system matrix M for the harmonic oscillator
M = np.array([[0, 1], [-1, 0]])
U = 1j * M  # Define the Hamiltonian as a unitary operator

# Compute the time evolution operator exp(-iMt)
time_evolution_operator = expm(-1j * U * t)
```

- Based on the 2-qubit state equation, we calculate the values of the amplitudes, and then apply the time evolution operator e^{-iHt}, followed by measurement

```
# Create the initial state vector
initial_state = [c00, c01, c10, c11]
qc.initialize(initial_state, [0, 1])  # Initialize both qubits

# Step 2: Apply the time evolution operator exp(-iMt)
qc.unitary(time_evolution_operator, [0, 1], label="Time Evolution")
```

- Finally, we measure the circuit for each iteration and calculate the probabilities based on each outcome, and approximate our solution using it

```
# Step 3: Measure the qubits to get probabilities
qc.measure_all()

# Simulate the circuit
simulator = Aer.get_backend('qasm_simulator')
compiled_circuit = transpile(qc, simulator)
qobj = assemble(compiled_circuit)
result = simulator.run(qc, shots=num_shots).result()

# Get the results of the simulation
counts = result.get_counts(qc)

# Calculate probabilities for each measurement outcome
probabilities = {state: count / num_shots for state, count in counts.items()}

# Calculate y(t) = cos(t) + sin(t) based on probabilities
y_t = (probabilities.get('00', 0) * 1 + probabilities.get('01', 0) * 1 +
       probabilities.get('10', 0) * 0 + probabilities.get('11', 0) * 0)

y_results.append(y_t)
```

2. The second part of the problem is time-evolution of the KE and PE operators
    - We import the required libraries and Pauli gates and define our Hamiltonian components (the KE and PE operators) as described in the Time Evolution section

```
# Use strings to represent Pauli matrices
Z = Pauli('Z')
X = Pauli('X')
I = Pauli('I')
# Define time points
t_vals = np.linspace(0, 1, 100)

# Initialize arrays for KE, PE, and total energy
KE_values = []
PE_values = []
total_energy_values = []

# Define the Hamiltonian components: Potential energy (Pauli-Z) and Kinetic energy (Pauli-X)
V = SparsePauliOp(Pauli('ZI')) + SparsePauliOp(Pauli('IZ'))  # Potential energy operator
T = SparsePauliOp(Pauli('XI')) + SparsePauliOp(Pauli('IX'))  # Kinetic energy operator
```

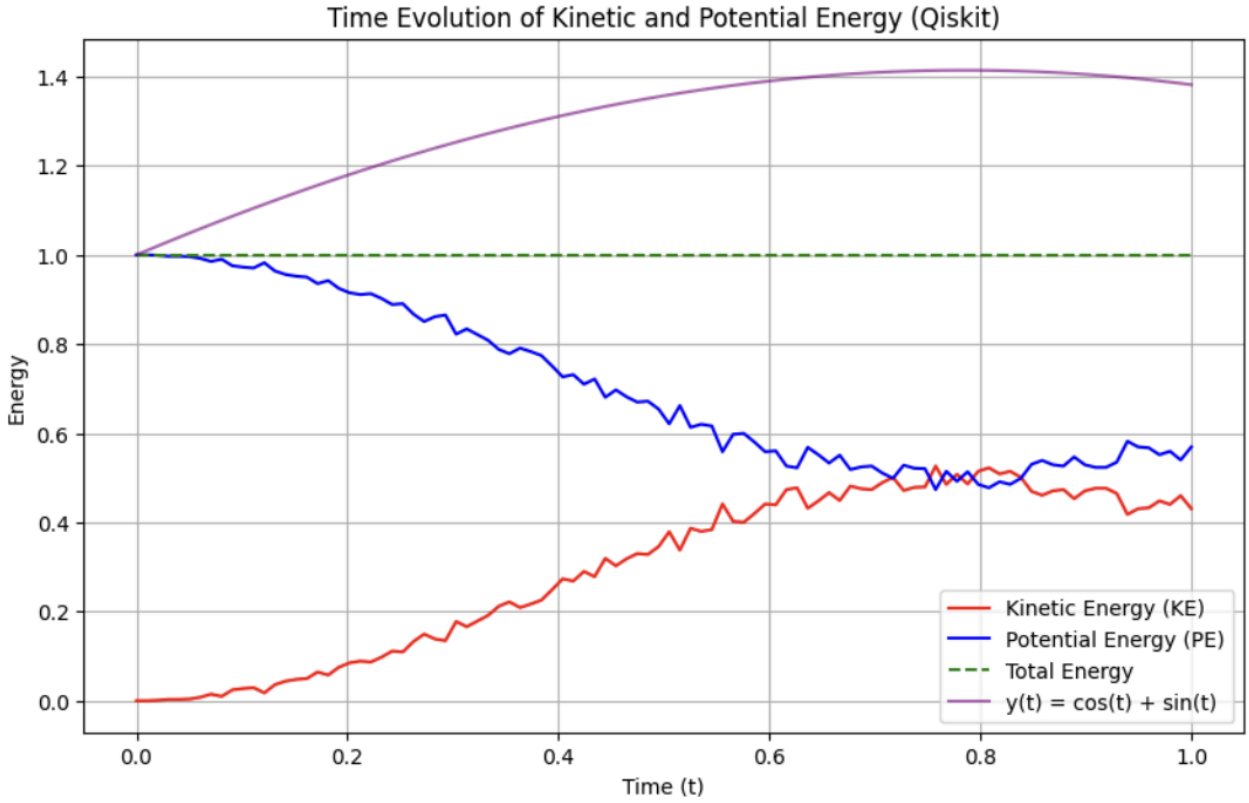
- For each time interval, the time evolution gates e^{-iVt} and e^{-iTt} are applied and the qubits are measured

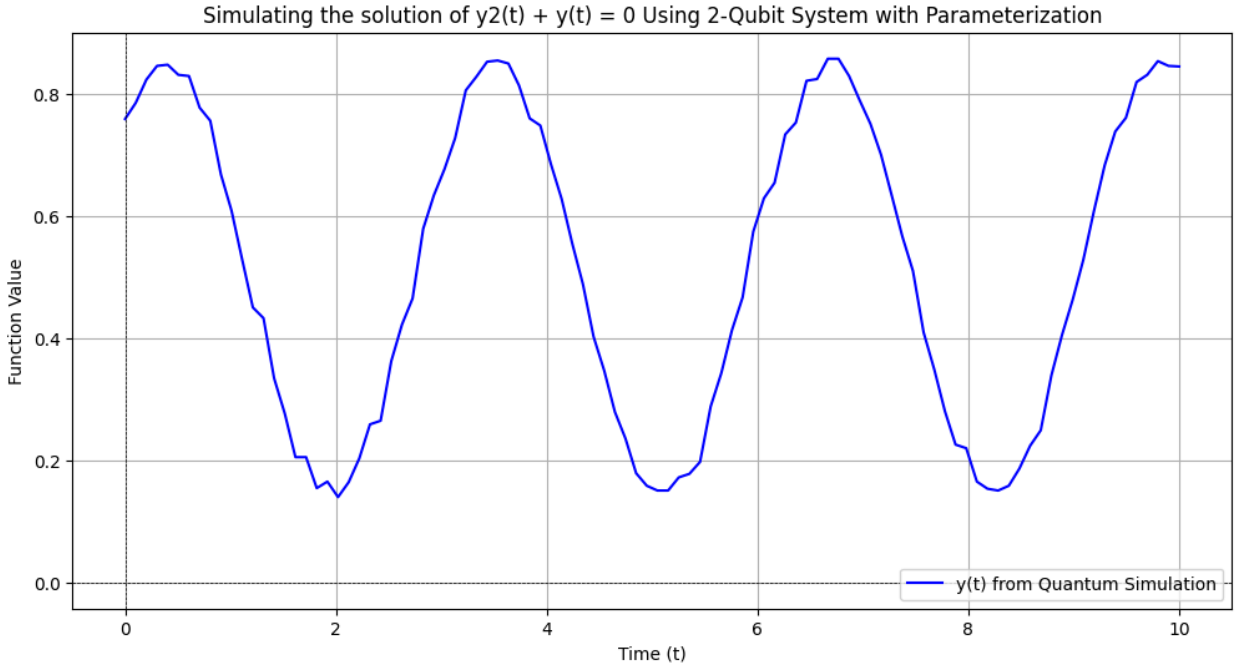
```
# Simulate for each time step
for t in t_vals:
    # Create quantum circuit for 2 qubits
    qc = QuantumCircuit(2)

    # Apply time evolution operator using Trotter-Suzuki decomposition
    # First, apply the potential energy evolution exp(-iVt)
    qc.unitary(Operator(expm(-1j * V.to_matrix() * t)), [0, 1], label="exp(-iVt)")

    # Then, apply the kinetic energy evolution exp(-iTt)
    qc.unitary(Operator(expm(-1j * T.to_matrix() * t)), [0, 1], label="exp(-iTt)")
    # Add measurement gates to the circuit
    qc.measure_all()
```

- We map the KE operator probabilities to the $|01\rangle$ *and* $|10\rangle$ states and the PE probabilities to the $|00\rangle$ *and* $|11\rangle$ states of the 2-qubit system.
  The expectation values for both the operators are calculated and are stored in an empty string

```
    # Measure expectation values
    counts = result.get_counts()
    probabilities = {state: count / 1024 for state, count in counts.items()}

    # Calculate expectation values for kinetic energy (using Pauli-X basis)
    KE = probabilities.get('01', 0) + probabilities.get('10', 0)
    KE_values.append(KE)

    # Calculate expectation values for potential energy (using Pauli-Z basis)
    PE = probabilities.get('00', 0) + probabilities.get('11', 0)
    PE_values.append(PE)

    # Total energy should be KE + PE
    total_energy_values.append(KE + PE)
```

RESULTS


Time Evolution of Kinetic and Potential Energy (Qiskit)
Energy
Time (t)
Kinetic Energy (KE)
Potential Energy (PE)
Total Energy
y(t) = cos(t) + sin(t)


Simulating the solution of y2(t) + y(t) = 0 Using 2-Qubit System with Parameterization
Function Value
Time (t)
y(t) from Quantum Simulation

BURNING QUESTIONS

This challenge provides a great opportunity for us to dive deep into multiple algorithms and try to understand them from first principle. Although the more we explore, the more questions we encounter:

1. Do we need to apply QFT for this problem?
2. The algorithm shown in the paper used Taylor expansion, is Trotterization a better way to do approximation
3. If operator T and V do commute with each other, does trotterization provide an exact solution?
4. What does “a qubit evolve in time” really means?