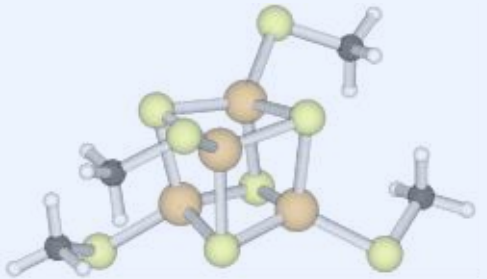
Hamiltonian Simulation

Learning to run quantum circuits through domain-specific problem

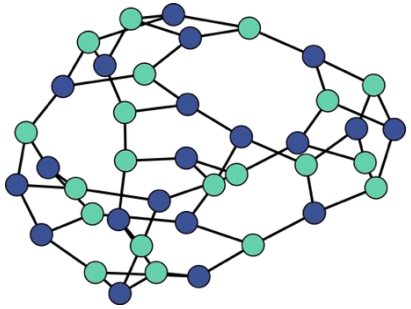
Yuri Kobayashi
IBM Quantum

Domain specific problems: Promising quantum computational areas

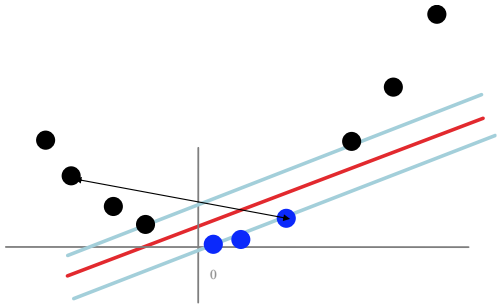
Hamiltonian simulation



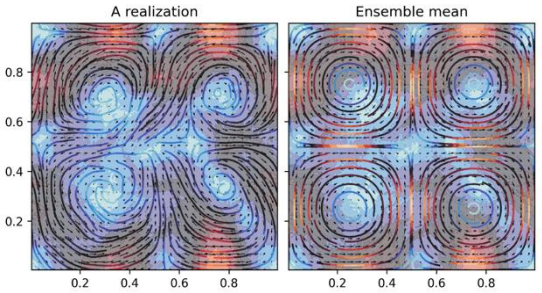
Optimization



Machine learning



Differential equations



Install and set up Qiskit 2.x (macOS)

- Reference URL： https://docs.quantum.ibm.com/guides/install-qiskit(For non-macOS users, please refer this.)
- For those who are using Qiskit prior to 1.x. It is strongly recommended to start a new virtual environment to install Qiskit 2.x.
- An extremely fast Python installer uv (10—100x faster than pip) is also available. Please see next slide to utilize uv.

1. Create a new virtual environment, using Python 3.8 or later.

```
python3 -m venv venv
```

2. Activate the environment.

```
source venv/bin/activate
```

3. Install Qiskit.

```
pip install qiskit
```

4. Install the necessary packages one-by-one.

```
pip install qiskit-ibm-runtime
pip install qiskit[visualization]
pip install jupyter
pip install qiskit-aer
```

zsh users will need to put *'qiskit[visualization]'* in single quotes.

5. With the following command, you can launch Jupyter notebook and start using Qiskit.

```
jupyter notebook
```

6. When switching back to your global environment or to another virtual environment, you can deactivate the environment with the following command.

```
deactivate
```

A guide for the impatient - Install and set up Qiskit 2.x (macOS) lightning fast

- Reference URL： https://docs.quantum.ibm.com/guides/install-qiskit(For non-macOS users, please refer this.)
- For those who are using Qiskit prior to 1.x. It is strongly recommended to start a new virtual environment to install Qiskit 2.x.
- An extremely fast Python installer uv (10—100x faster than pip) is also available. Please see instructions below to use uv

1. Install **uv**, a Python installer for saving time.
   python3 -m pip install *uv*

2.Create a new virtual environment
   uv venv *venv* #python will be installed by default

3. Activate the environment.
   source *venv*/bin/activate

4. Install Qiskit.
   uv pip install *qiskit*

5. Install the necessary packages below one-by-one.
   uv pip install *qiskit-ibm-runtime*
   uv pip install *qiskit[visualization]* # **zsh** users will need to put *'qiskit[visualization]'* in single quotes.
   uv pip install *qiskit-aer*
   uv pip install *jupyter*

5. With the following command, you can launch Jupyter notebook and start using Qiskit.
   jupyter notebook

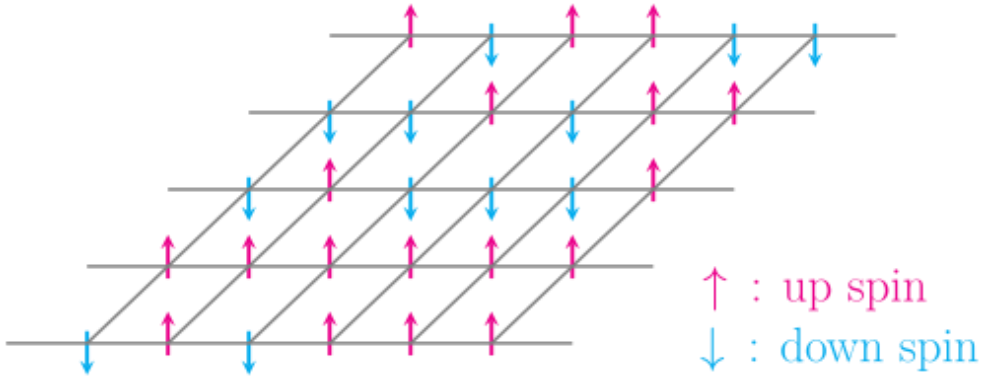
6. When switching back to your global environment or to another virtual environment, you can deactivate the environment with the following command.
   deactivate

Hamiltonian Simulation 1-D Ising Model

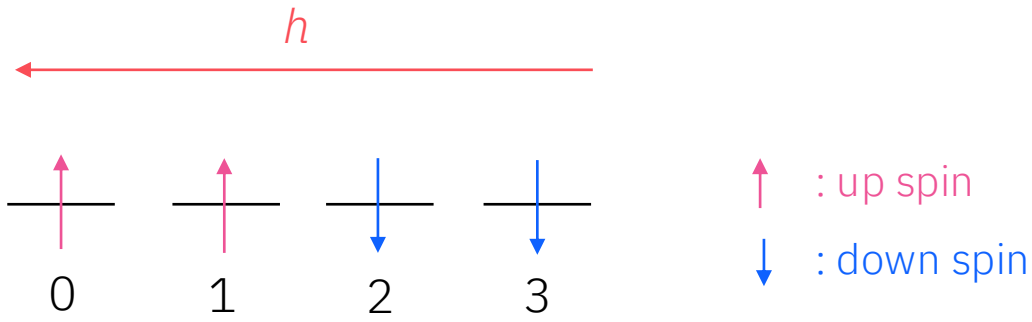
Getting up to speed with Qiskit through simulating a 1-D Transverse Ising Model

Ising Models describe magnetic dipole moments of atomic "spins" that can be in one of two states (+1 or –1). The spins are arranged in a graph, usually a lattice allowing each spin to interact with its neighbors. In today's exercise, we will look at a 1-D case.

2-D Ising Model



1-D Ising Model



Background information for mapping our problem

Hamiltonian Simulation for Electronic Structure Problems
The goal: solve the Schrödinger equation for $\Psi(t)$

Time-dependent Schrödinger equation

$$\hat{H}\Psi(t) = i\hbar\frac{\partial}{\partial t}\Psi(t)$$

$\hat{H}$ **Hamiltonian**
corresponds to the total energy of the system

Ψ **Wavefunction**
tells us about the characteristics of a particle system (e.g., energy, position, momentum, spin)

The goal is to compute the wave function that satisfies the above

$$|\Psi(t)\rangle = e^{-i\hat{H}t}|\Psi(0)\rangle$$

The wave function contains vital information about a quantum system

$\hat{E}$	$\hat{x}$	$\hat{p}$	$\hat{L}$
energy	position	momentum	orbital angular momentum ~ Spin

Hamiltonian in general

Hamiltonian of a quantum system is an operator representing the total energy of the system

$$\hat{H} = \underbrace{\hat{T}}_{\text{Kinetic Energy}} + \underbrace{\hat{V}}_{\text{Potential Energy}}$$

Important in many fields

- Quantum chemistry (material science), Condensed matter physics, High-energy physics
- Optimization problems where the cost function is defined as a Hamiltonian

Spin Hamiltonian

In Hamiltonian Simulations, these energies can come from spin 1/2 interactions and external interactions

$$\hat{H} = \hat{T} + \hat{V}$$

KE: spin interactions

PE: e.g., interactions between electron and Nuclei
interaction between Nuclei-Nuclei
external magnetic field

Electronic structure Hamiltonian of a molecule

The molecular Hamiltonian is

$$\hat{\mathbf{H}} = \sum_{i}^{electrons} \frac{-\hbar^2}{2m_e}\nabla_i^2 + \sum_{A}^{nuclei} \frac{-\hbar^2}{2m_A}\nabla_A^2 + \sum_{i}^{electrons}\sum_{A}^{nuclei} \frac{-e^2 Z_A}{r_{iA}} + \sum_{i>j}^{electrons} \frac{e^2}{r_{ij}} + \sum_{A>B}^{nuclei} \frac{e^2 Z_A Z_B}{r_{AB}}$$

Kinetic energy of electrons

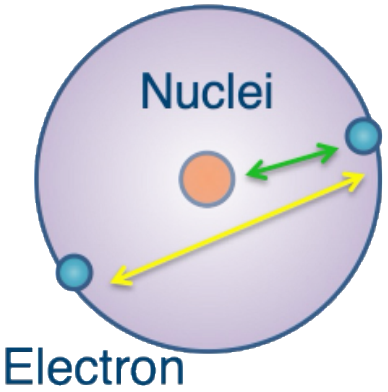
Kinetic energy of the nuclei

Electron-nuclei attraction

Electron-electron repulsion

Nuclei-nuclei repulsion

The Born-Oppenheimer approximation neglects the motion of the atomic nuclei

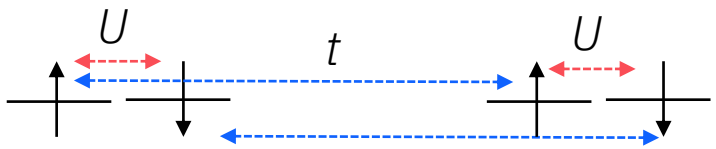


Second quantization

– Hubbard model

Describe conducting and insulating systems

$$H = -t \sum_{i,\sigma} \left(\hat{a}^{\dagger}_{i,\sigma} \hat{a}_{i+1,\sigma} + \hat{a}^{\dagger}_{i+1,\sigma} \hat{a}_{i,\sigma} \right) + U \sum_{i} \hat{n}_{i,\uparrow} \hat{n}_{i,\downarrow}$$



$$\hat{n}_{i,\sigma} = \hat{a}^{\dagger}_{i,\sigma} \hat{a}_{i,\sigma}$$

Creation operator

Annihilation operator

– Quantum Chemistry Hamiltonian

$$\hat{H}_{ele}(\mathbf{r};\mathbf{R}) = -\sum_{i}^{N_{ele}} \frac{1}{2} \nabla_i^2 - \sum_{A}^{N_{nuc}} \sum_{i}^{N_{ele}} \frac{Z_A}{r_{iA}} + \sum_{i>j}^{N_{ele}} \frac{1}{r_{ij}}$$

Kinetic energy of electrons

Electron-nucleus attraction

Electron-electron repulsion

Complexity & Computational resources

Second quantization: Hartree-Fock method

The Hartree–Fock method is a mean-field approximation typically used to solve the Schrödinger equation for a multi-electron atom or molecule as described in the Born–Oppenheimer approximation and is considered a good starting point for solving this problem.

This method approximates an N-body problem by N one-body problems where each electron evolves in the mean-field of the others.

$$\hat{H}_{elec} = \sum_{pq} h_{pq} \hat{a}_p^\dagger \hat{a}_q + \frac{1}{2} \sum_{pqrs} h_{pqrs} \hat{a}_p^\dagger \hat{a}_q^\dagger \hat{a}_r \hat{a}_s$$

Mapping the Hamiltonian

Map the second-quantized Hamiltonian to qubits:

$$H = -t \sum_{i,\sigma} \left(\hat{a}^{\dagger}_{i,\sigma} \hat{a}_{i+1,\sigma} + \hat{a}^{\dagger}_{i+1,\sigma} \hat{a}_{i,\sigma} \right) + U \sum_{i} \hat{n}_{i,\uparrow} \hat{n}_{i,\downarrow}$$

Hubbard, Quantum chemistry

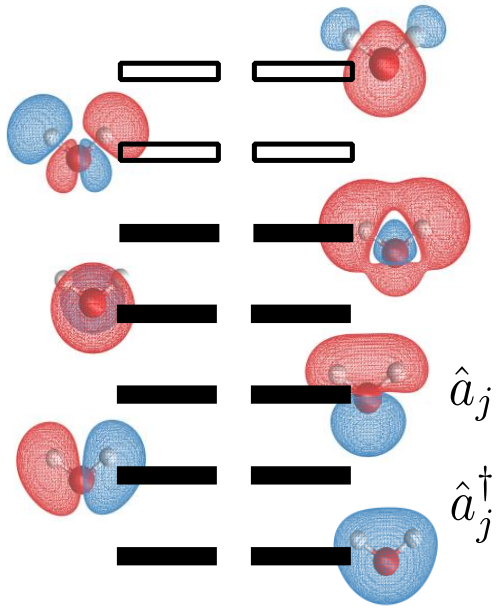
$$\hat{n}_{i,\sigma} = \hat{a}^{\dagger}_{i,\sigma} \hat{a}_{i,\sigma}$$

Fermions

$$\{a_i, a_i\} = 0,\ \{a_i^{\dagger}, a_i^{\dagger}\} = 0, \{a_i, a_i^{\dagger}\} = \delta_{i,j}$$

Spins

$$[\sigma_i, \sigma_i] = 0,\ [\sigma_i^{\dagger}, \sigma_i^{\dagger}] = 0, [\sigma_i, \sigma_j^{\dagger}] = \delta_{i,j}$$

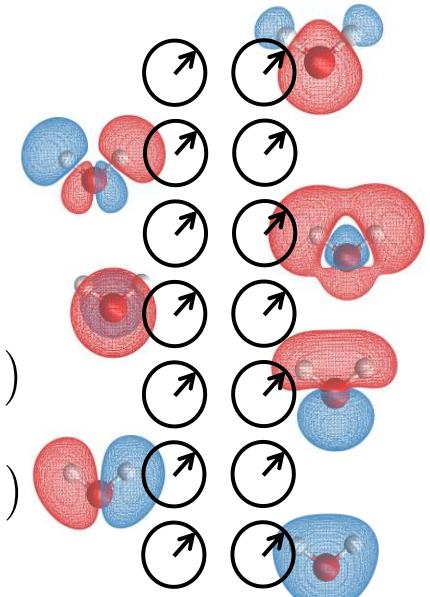


mapping

Jordan-Wigner

$$\hat{a}_j = \bigotimes_{i=1}^{j-1} \hat{\sigma}_i^z \otimes (\hat{\sigma}_j^x + i\hat{\sigma}_j^y)$$

$$\hat{a}_j^{\dagger} = \bigotimes_{i=1}^{j-1} \hat{\sigma}_i^z \otimes (\hat{\sigma}_j^x - i\hat{\sigma}_j^y)$$



$$\hat{H}_{elec} = \sum_{pq} h_{pq} \hat{a}_p^{\dagger} \hat{a}_q + \frac{1}{2} \sum_{pqrs} h_{pqrs} \hat{a}_p^{\dagger} \hat{a}_q^{\dagger} \hat{a}_r \hat{a}_s$$

$$\hat{H}_{elec} = \sum_{i} c_i \hat{P}_i$$

Pauli matrices

- Pauli matrices play a critical role in describing spin ½. (Usually written as $\boldsymbol{\sigma_x, \sigma_y, \sigma_z,}$)
- Linear algebraic characteristics: Hermitian, traceless, and unitary
- They represent operators which gives us a projection of spin of electrons along the x, y, z, axis
- And the good news is that they can be implemented using quantum gates on a quantum computer!

Pauli Operators

Pauli x $\quad \sigma_1 = \sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$

Pauli y $\quad \sigma_2 = \sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$

Pauli z $\quad \sigma_3 = \sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$

Rotational gates

$$\boldsymbol{R_x(\theta) = e^{-i\frac{\theta}{2}\sigma_x} = \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) & -i\sin\left(\frac{\theta}{2}\right) \\ -I\sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{pmatrix}}$$

$$\boldsymbol{R_y(\theta) = e^{-i\frac{\theta}{2}\sigma_y} = \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) & -\sin\left(\frac{\theta}{2}\right) \\ \sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{pmatrix}}$$

$$\boldsymbol{R_z(\theta) = e^{-i\frac{\theta}{2}\sigma_z} = \begin{pmatrix} e^{-i\frac{\theta}{2}} & 0 \\ 0 & e^{i\frac{\theta}{2}} \end{pmatrix}}$$

Jordan–Wigner Mapping

Fermionic Hamiltonian $\hat{H}_M = \sum_{pq} h_{pq} a_p^\dagger a_q + \sum_{pqrs} h_{pqrs} a_p^\dagger a_q^\dagger a_r a_s$

Jordan–Wigner mapping

Creation operator $a_p^\dagger = \frac{1}{2}\left(X_p - iY_p\right) \otimes Z_{p-1} \otimes \cdots \otimes Z_1$

Annihilation operator $a_q = \frac{1}{2}\left(X_q + iY_q\right) \otimes Z_{q-1} \otimes \cdots \otimes Z_1$

These our Pauli operators!

$$\sigma_{X_i}, \sigma_{Y_i}, \sigma_{Z_i} = X_i, Y_i, Z_i$$

Hydrogen molecule (bond length=0.735 Angstrom, STO-3G basis set. 4 spin orbitals and 36 terms)

$$H_f = -1.26 a_0^\dagger a_0 - 0.47 a_1^\dagger a_1 - 1.26 a_2^\dagger a_2 - 0.47 a_3^\dagger a_3$$

$$+0.34 a_0^\dagger a_0^\dagger a_0 a_0 + 0.33 a_0^\dagger a_1^\dagger a_1 a_0 + 0.34 a_0^\dagger a_2^\dagger a_2 a_0 + 0.33 a_0^\dagger a_3^\dagger a_3 a_0 + \cdots$$

$$+0.09 a_0^\dagger a_2^\dagger a_3 a_1 + \cdots$$

Mapping the one-body term using Pauli operators

Use these relations

$$\sigma_{X_i}, \sigma_{Y_i}, \sigma_{Z_i} = X_i, Y_i, Z_i$$

$$\sigma_X\sigma_Y = -\sigma_Y\sigma_X = i\sigma_Z$$
$$\sigma_Y\sigma_Z = -\sigma_Z\sigma_Y = i\sigma_X$$
$$\sigma_Z\sigma_X = -\sigma_X\sigma_Z = i\sigma_Y$$

By using the above relations, we can describe the one-body terms with Pauli operators as shown below

$$a_3^\dagger a_3 = \frac{1}{2}\left(X_3 - iY_3\right) \otimes Z_2Z_1Z_0 \times \frac{1}{2}\left(X_3 + iY_3\right) \otimes Z_2Z_1Z_0$$

$$= \frac{1}{4}\left(X_3Z_2Z_1Z_0 - iY_3Z_2Z_1Z_0\right) \times \left(X_3Z_2Z_1Z_0 + iY_3Z_2Z_1Z_0\right) = \frac{1}{4}\left(I + I + iX_3Y_3 - iY_3X_3\right)$$

$$= \frac{1}{4}\left(I + I + iX_3Y_3 - iY_3X_3\right) = \frac{1}{2}\left(I + iX_3Y_3\right) = \frac{1}{2}\left(I - Z_3\right)$$

Mapping the two-body term using Pauli operators

Use these relations

$$\sigma_X^2 = \sigma_Y^2 = \sigma_Z^2$$

$$\sigma_X\sigma_Y = -\sigma_Y\sigma_X = i\sigma_Z$$

$$\sigma_Y\sigma_Z = -\sigma_Z\sigma_Y = i\sigma_X$$

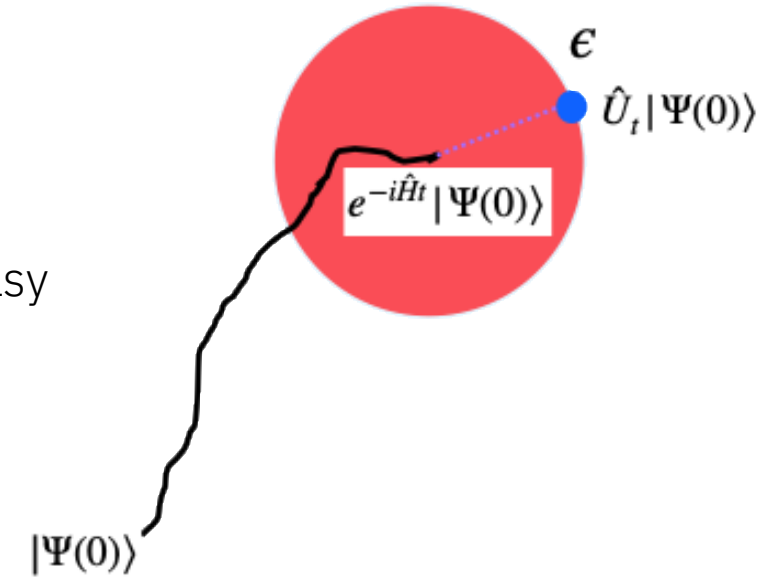
$$\sigma_Z\sigma_X = -\sigma_X\sigma_Z = i\sigma_Y$$

By using the above relations, we can describe the two-body terms with Pauli operators as shown below

$$a_0^\dagger a_2^\dagger a_3 a_1 = \frac{1}{2}\left(X_0 - iY_0\right) \times \frac{1}{2}\left(X_2 - iY_2\right) \otimes Z_1Z_0 \times \frac{1}{2}\left(X_3 + iY_3\right) \otimes Z_2Z_1Z_0 \times \frac{1}{2}\left(X_1 + iY_1\right) \otimes Z_0$$

$$= \frac{1}{16}\left[-X_3Y_2X_1Y_0 - iX_3Y_2Y_1Y_0 - iY_3Y_2X_1Y_0 + Y_3Y_2Y_1Y_0 - iX_3X_2X_1Y_0 + X_3X_2Y_1Y_0 + Y_3X_2X_1Y_0 + iY_3X_2Y_1Y_0\right.$$
$$\left. - iX_3Y_2X_1X_0 + X_3Y_2Y_1X_0 + Y_3Y_2X_1X_0 + iY_3Y_2Y_1X_0 + X_3X_2X_1X_0 + iX_3X_2Y_1X_0 + iY_3X_2X_1X_0 - Y_3X_2Y_1X_0\right]$$

Approximation techniques



Compute wavefunction at time *t* ← computing this is not easy

$$|\Psi(t)\rangle = e^{-i\hat{H}t}|\Psi(0)\rangle$$

We apply an approximation

$$|\Psi(t+\Delta t)\rangle = e^{-i\hat{H}\Delta t}|\Psi(t)\rangle \approx \left(1 - iH\Delta t - \frac{\hat{H}^2\Delta t^2}{2} + \ldots\right)|\Psi(t)\rangle$$

Very small time slice

Trotterization

Let us focus on a simple Hamiltonian

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

Lie Product Formula (also known as the Trotter Formula)

$$e^{-it(H_1+H_2)} = \lim_{n\to\infty} \left(e^{-iH_1\frac{t}{n}} e^{-iH_2\frac{t}{n}} \right)^n$$

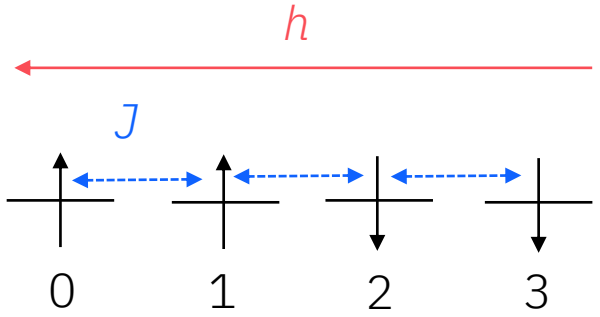
We will take "n" to be finite

$$e^{-i(\hat{H}_1+\hat{H}_2)\Delta t} = e^{-i\hat{H}_1\Delta t} e^{-i\hat{H}_2\Delta t}$$

This only holds when H_1 and H_2 commute, but this is often not the case

Example: Trotterization (first-order)
Transverse Ising model

$$H = -\sum_{\langle i,j \rangle}^{N-1} J\sigma_{Z_i}\sigma_{Z_j} - \sum_{i}^{N} h_i\sigma_{X_i}$$



N: Number of qubits

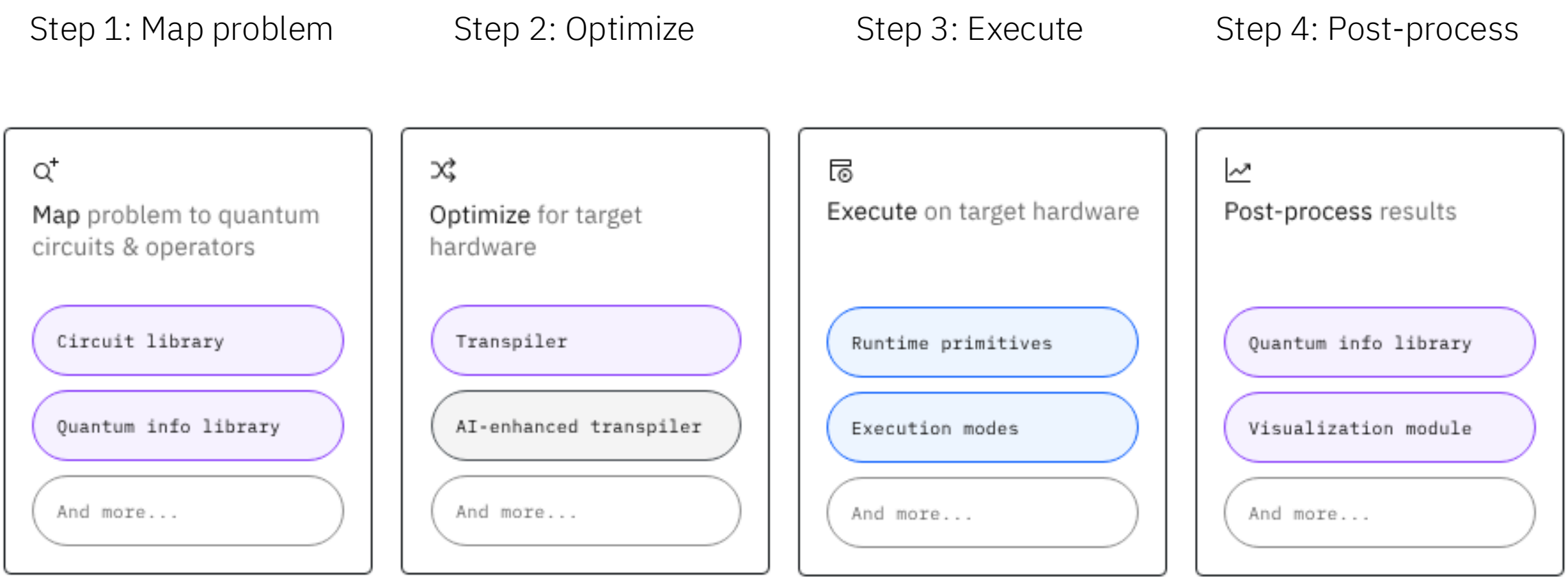
$$e^{-i\hat{H}\Delta t} = e^{-i\Delta t(-\sum_{i,j}^{N} J\sigma_{Z_i}\sigma_{Z_j} - \sum_{i}^{N} h_i\sigma_{X_i})} \approx e^{-i\Delta t(-\sum_{i,j}^{N} J\sigma_{Z_i}\sigma_{Z_j})} e^{-i\Delta t(-\sum_{i}^{N} h_i\sigma_{X_i})}$$

$R_{ZZ}(-2J\Delta t)$ $\qquad$ $R_X(-2h\Delta t)$

$$R_{ZZ}(\theta) = e^{-i\frac{\theta}{2}\sigma_Z\sigma_Z} = \begin{pmatrix} e^{-i\frac{\theta}{2}} & 0 & 0 & 0 \\ 0 & e^{i\frac{\theta}{2}} & 0 & 0 \\ 0 & 0 & e^{i\frac{\theta}{2}} & 0 \\ 0 & 0 & 0 & e^{-i\frac{\theta}{2}} \end{pmatrix}$$

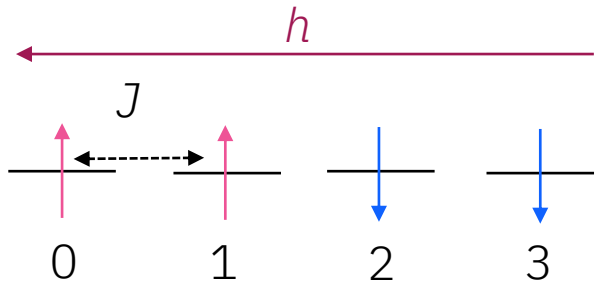
$$R_X(\theta) = e^{-i\frac{\theta}{2}\sigma_X} = \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) & -i\sin\left(\frac{\theta}{2}\right) \\ -i\sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{pmatrix}$$

Qiskit Patterns



Step 1: Map problem to circuit – 1-D Transverse Ising Model with 1st order trotterization

$$H = -\sum_{\langle i,j \rangle}^{N-1} J\sigma_{Z_i}\sigma_{Z_j} - \sum_{i}^{N} h_i\sigma_{X_i}$$



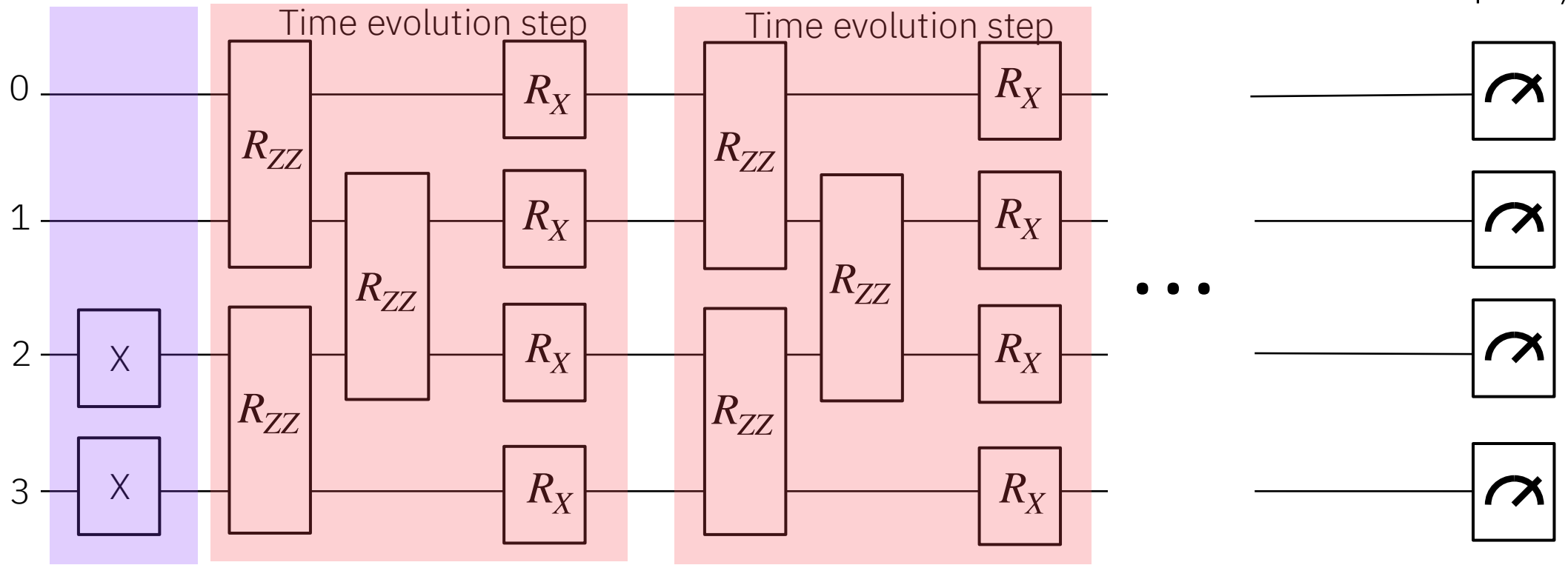
0: up spin

1: down spin

Bit strings

$|0011\rangle$

$|1100\rangle$



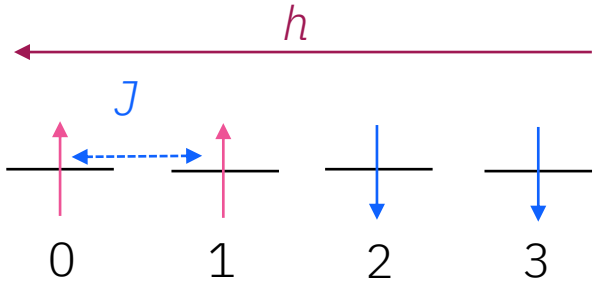
State preparation

By repeating this, we can get the wavefunction at time t

$$|\Psi(t)\rangle = e^{-i\hat{H}t}|\Psi(0)\rangle$$

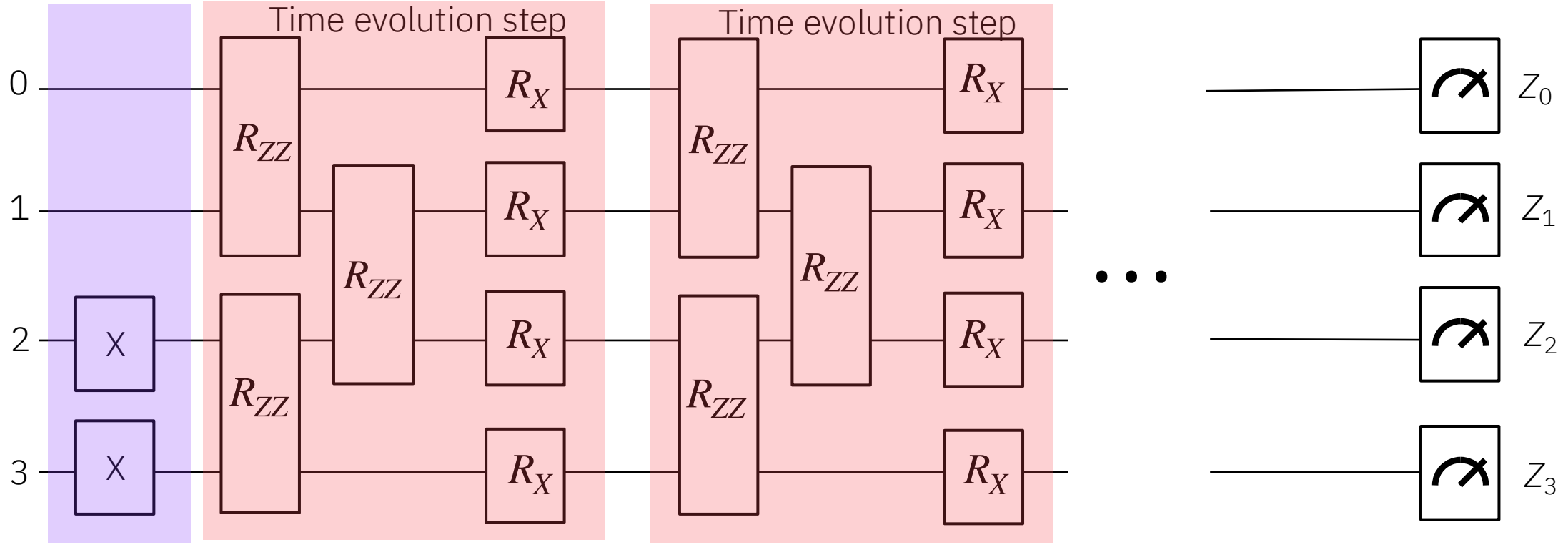
Measuring a physical observable

$$H = -\sum_{\langle i,j \rangle}^{N-1} J\sigma_{Z_i}\sigma_{Z_j} - \sum_{i}^{N} h_i\sigma_{X_i}$$



Magnetization

$$\sum_{i}^{N} Z_i/N$$



State preparation

Measure expectation values of observables

Suzuki-Trotter Formula (2nd order)

Hamiltonian (general form)

$$\hat{H} = \sum_{i=1}^{L} a_i P_i$$

Second-order Suzuki–Trotter formula

$$U_{ST2} = \prod_{j=1}^{L} e^{-ia_j P_j \frac{t}{2}} \prod_{j'=L}^{1} e^{-ia_{j'} P_{j'} \frac{t}{2}}$$

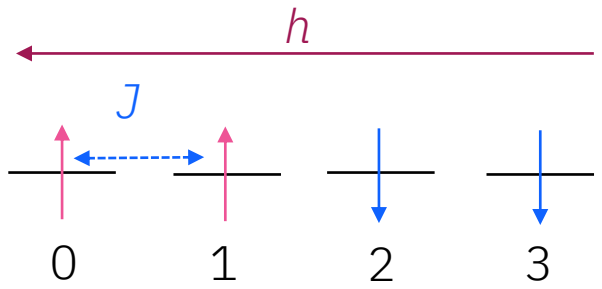
Again, let us focus on a simple Hamiltonian

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

$$\hat{U}_{\mathrm{ST2}} = e^{-i\hat{H}_1 \frac{\Delta t}{2}} e^{-i\hat{H}_2 \Delta t} e^{-i\hat{H}_1 \frac{\Delta t}{2}}$$

Example: Trotterization (second-order) Transverse Ising model

$$H = -\sum_{\langle i,j \rangle}^{N-1} J\sigma_{Z_i}\sigma_{Z_j} - \sum_{i}^{N} h_i\sigma_{X_i}$$



$$e^{-i\hat{H}\Delta t} = e^{-i\Delta t(-\sum_{i,j}^{N} J\sigma_{Z_i}\sigma_{Z_j} - \sum_{i}^{N} h_i\sigma_{X_i})}$$

$$\approx e^{-i\frac{\Delta t}{2}(-J\sigma_{Z_0}\sigma_{Z_1})} e^{-i\frac{\Delta t}{2}(-J\sigma_{Z_1}\sigma_{Z_2})} e^{-i\frac{\Delta t}{2}(-J\sigma_{Z_2}\sigma_{Z_3})} e^{-i\frac{\Delta t}{2}(-h\sigma_{X_0})} e^{-i\frac{\Delta t}{2}(-h\sigma_{X_1})} e^{-i\frac{\Delta t}{2}(-h\sigma_{X_2})}$$

$$e^{-i\Delta t(-h\sigma_{X_3})}$$

$$e^{-i\frac{\Delta t}{2}(-h\sigma_{X_2})} e^{-i\frac{\Delta t}{2}(-h\sigma_{X_1})} e^{-i\frac{\Delta t}{2}(-h\sigma_{X_0})} e^{-i\frac{\Delta t}{2}(-J\sigma_{Z_2}\sigma_{Z_3})} e^{-i\frac{\Delta t}{2}(-J\sigma_{Z_1}\sigma_{Z_2})} e^{-i\frac{\Delta t}{2}(-J\sigma_{Z_0}\sigma_{Z_1})}$$

Error in Suzuki-Trotter Formula (2nd order)

$$\hat{U}_{\text{exact}} = e^{-i(\hat{H}_1+\hat{H}_2)\Delta t}$$

$$\hat{U}_{\text{ST2}} = e^{-i\hat{H}_1\frac{\Delta t}{2}}e^{-i\hat{H}_2\Delta t}e^{-i\hat{H}_1\frac{\Delta t}{2}}$$

The exact Taylor expansion truncated at the 3rd order

$$\hat{U}_{\text{exact}_3} = \mathbb{I} + (-i\Delta t)(\hat{H}_1+\hat{H}_2) + \frac{(-i\Delta t)^2}{2}\left(\hat{H}_1+\hat{H}_2\right)^2 + \frac{(-i\Delta t)^3}{6}\left(\hat{H}_1+\hat{H}_2\right)^3$$

The Taylor expansion for each term (3rd order) in the 2nd order Suzuki-Trotter Formula

$$\hat{U}_{\text{ST2}_3} = \left[\mathbb{I} + (-i\Delta t/2)\hat{H}_1 + \frac{(-i\Delta t/2)^2}{2}\left(\hat{H}_1^2\right) + \frac{(-i\Delta t/2)^3}{6}\left(\hat{H}_1^3\right)\right]$$

$$\left[\mathbb{I} + (-i\Delta t)\hat{H}_2 + \frac{(-i\Delta t)^2}{2}\left(\hat{H}_2^2\right) + \frac{(-i\Delta t)^3}{6}\left(\hat{H}_2^3\right)\right]$$

$$\left[\mathbb{I} + (-i\Delta t/2)\hat{H}_1 + \frac{(-i\Delta t/2)^2}{2}\left(\hat{H}_1^2\right) + \frac{(-i\Delta t/2)^3}{6}\left(\hat{H}_1^3\right)\right]$$

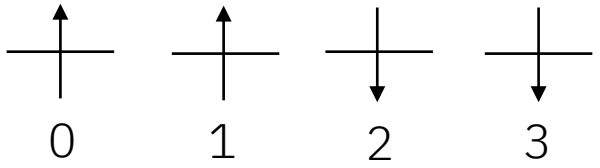
$$\hat{U}_{\text{ST2}_3} \approx \mathbb{I} + (-i\Delta t/2)(\hat{H}_1+\hat{H}_2) + \frac{(-i\Delta t/2)^2}{2}\left(\hat{H}_1+\hat{H}_2\right)^2$$

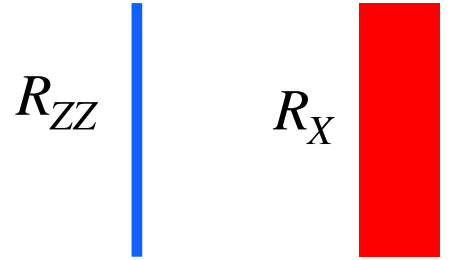
$$+\frac{(-i\Delta t/2)^3}{6}\left[\hat{H}_1^3 + \frac{3}{2}(\hat{H}_1\hat{H}_2^2 + \hat{H}_2^2\hat{H}_1 + \hat{H}_1\hat{H}_2\hat{H}_1) + \frac{3}{2}(\hat{H}_2\hat{H}_1^2 + \hat{H}_1^2\hat{H}_2 + \hat{H}_2^3)\right]$$

Error of the second-order Suzuki-Trotter

$$\|\hat{U}_{\text{exact}_3} - \hat{U}_{\text{ST2}_3}\| \leq \frac{1}{24}\|\hat{H}_2^2\hat{H}_1 + \hat{H}_1\hat{H}_2^2 + \hat{H}_1\hat{H}_2\hat{H}_1 + \hat{H}_2\hat{H}_1\hat{H}_2\|\Delta t^3$$

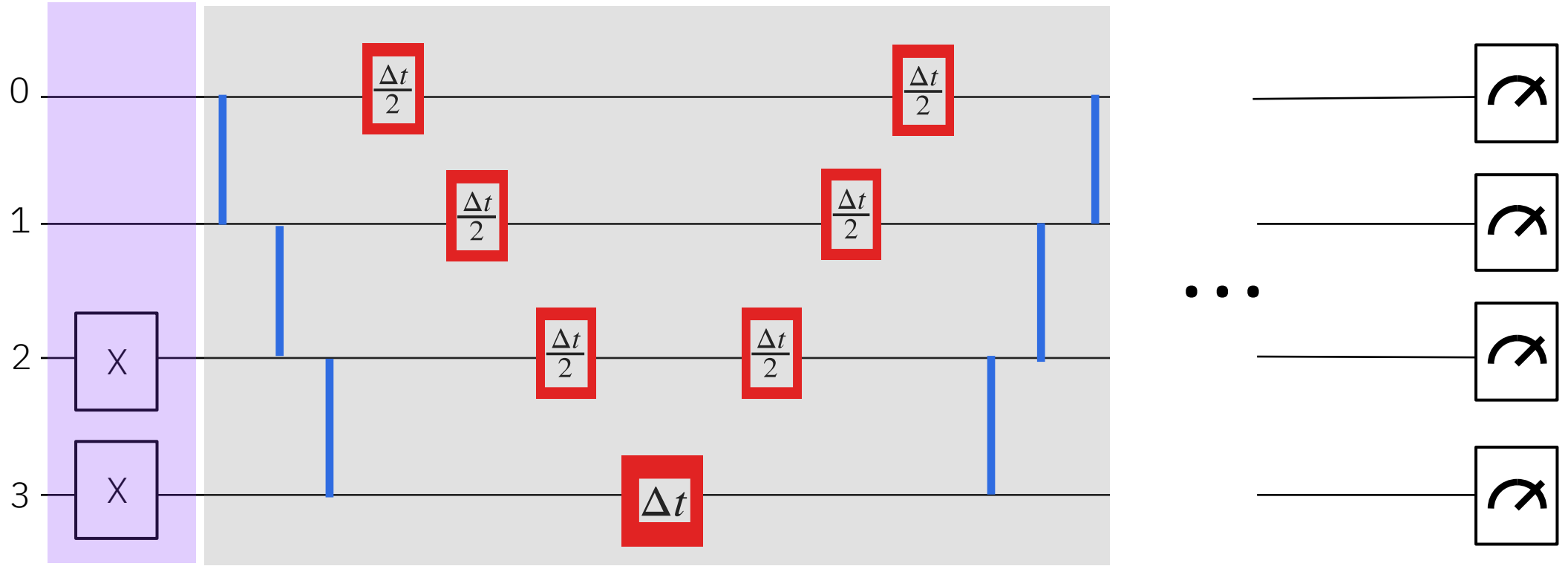
Example: Trotterization (second-order) Transverse Ising model





$$H = -\sum_{\langle i,j \rangle}^{N-1} J\sigma_{Z_i}\sigma_{Z_j} - \sum_{i}^{N} h_i\sigma_{X_i}$$

Time evolution step



State preparation

By repeating this, we can get the wavefunction of time t

Suzuki-Trotter recursion formula for higher ($\mathcal{P}^{\text{th}}$) order

Second-order Suzuki–Trotter formula $$e^{-itH} \approx \hat{U}_{ST2}(t) = \prod_{j=1}^{L} e^{-ia_j P_j \frac{t}{2}} \prod_{j'=L}^{1} e^{-ia_{j'} P_{j'} \frac{t}{2}}$$

$$\mathcal{P} = 2k$$

Recursion relation $$U_{ST(2k)}(t) = \left[U_{ST(2k-2)}\left(p_k t\right)\right]^2 U_{ST(2k-2)}\left((1-4p_k)t\right)\left[U_{ST(2k-2)}\left(p_k t\right)\right]^2$$

$$p_k = 1/\left(4 - 4^{\frac{1}{2k-1}}\right)$$

Fourth order Suzuki–Trotter $$\hat{U}_{ST4}(t) = \left[\hat{U}_{ST2}\left(p_2 t\right)\right]^2 \hat{U}_{ST2}\left((1-4p_2)t\right)\left[\hat{U}_{ST2}\left(p_2 t\right)\right]^2$$

$$p_2 = 1/\left(4 - 4^{\frac{1}{2*2-1}}\right) = 1/\left(4 - 4^{\frac{1}{3}}\right) \approx 0.4145$$

$$\hat{U}_{ST4}(\Delta t) = \hat{U}_{ST2}\left(0.4145\Delta t\right)\hat{U}_{ST2}\left(0.4145\Delta t\right)\hat{U}_{ST2}\left(-0.6579\Delta t\right)\hat{U}_{ST2}\left(0.4145\Delta t\right)\hat{U}_{ST2}\left(0.4145\Delta t\right)$$

Reasons to study product formulas (trotterization)

- The method is intuitive and easy to implement
- Number of qubits required is minimal (no auxiliary qubits required)
- The scaling of the gate depth against the error is not optimal
  - First-order trotter error: $O\left(t^2/\epsilon\right)$
  - Second-order trotter error: $O\left(t^{1.5}/\epsilon^{0.5}\right)$

Qiskit Coding Session

Please have the Jupyter notebook '**hamiltonian-simulation.ipynb**' downloaded and ready

https://github.com/qiskit-community/japan-pf-2026/tree/main/hamiltonian_simulation

What we will demonstrate in this coding session:

1. Quantum simulation with a simulator
   - Time evolution of an observable
   - "Estimator" in Qiskit
2. Quantum simulation with a quantum hardware
   - Time evolution of the wavefunction
   - "Sampler" in Qiskit

Qiskit Circuit Library

Evolution of the transverse Ising model

The PauliEvolutionGate implements $U(t) = e^{-itH}$

Ising model on spin-1/2 particles:

$$H = \underbrace{-J \sum_{jk} Z_j Z_k}_{H_{ZZ}} + \underbrace{h \sum_{j} X_j}_{H_X}$$

The SparsePauliOp is used to implement the Hamiltonian

```python
from qiskit.circuit.library import PauliEvolutionGate
from qiskit.quantum_info import Statevector, SparsePauliOp

def get_hamiltonian(nqubits, J, h, alpha):

    # List of Hamiltonian terms as 3-tuples containing
    # (1) the Pauli string,
    # (2) the qubit indices corresponding to the Pauli string,
    # (3) the coefficient.
    Hzz = [("ZZ", [i, i + 1], -J) for i in range(0, nqubits - 1)]
    Hz = [("Z", [i], -h * np.sin(alpha)) for i in range(0, nqubits)]
    Hx = [("X", [i], -h * np.cos(alpha)) for i in range(0, nqubits)]

    # We create the Hamiltonian as a SparsePauliOp, via the method
    # `from_sparse_list`, and multiply by the interaction term.
    hamiltonian = SparsePauliOp.from_sparse_list([*Hzz, *Hz, *Hx], num_qubits=nqubits)
    return hamiltonian.simplify()
```


IBM quantum utility paper: https://www.nature.com/articles/s41586-023-06096-3
API docs: https://docs.quantum.ibm.com/api/qiskit/qiskit.circuit.library.PauliEvolutionGate

Summary

Hamiltonian Simulations in the context of simulating electronic structures are one of the most promising areas for quantum computing to show advantage.

Solving the wave function is key to simulation and obtaining physical observables of interest

However, calculating the wave function exactly is very challenging and algorithms that can approximate the solution effectively is an active area of research

Trotterization is a widely used mathematical approach to simulate Hamiltonians on a quantum computer

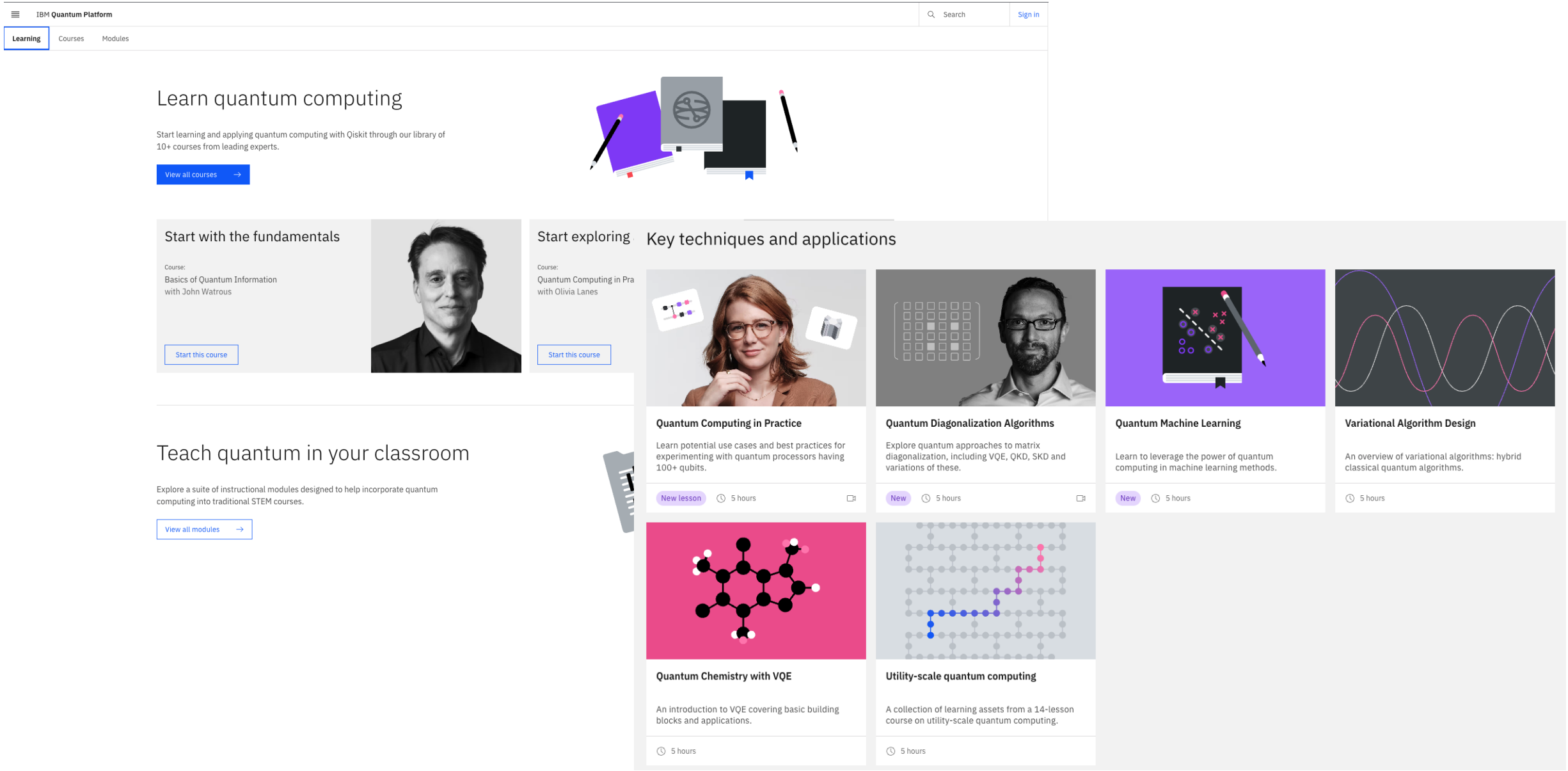
While higher order trotterization theoretically should provide better accuracy of results, it increases circuit depth which is something to consider when computing on near-term quantum computers.

This should give us inspiration to learning more about error mitigation techniques which is the next topic that will be covered by our partners!

Topics we covered today

1. IBM Quantum Platform now on IBM Cloud (since *July 2025*)
2. Verifying your Account
3. Support Center (how to create cases to get issues resolved)
4. Navigating the platform
5. Qiskit – users' top choice quantum SDK
6. Qiskit Runtime Primitives
7. Qiskit Patterns – framework for running domain specific problems
8. Promising quantum computational areas
9. Installing Qiskit (traditional way vs fast way)
10. Getting up to speed with Qiskit (through a domain specific problem)
    - Taking Hamiltonian Simulation as an example
    - Qiskit coding tutorial: 1-D Transverse Ising Model
11. Summary

Visit IBM Quantum Learning to continue your quantum journey




IBM