Quantum algorithms for Dynamics Simulation: Hamiltonian Simulation and Linear Differential Equations

Di Fang

Department of Mathematics
Duke Quantum Center
Duke University

IPAM Tutorial, 2023

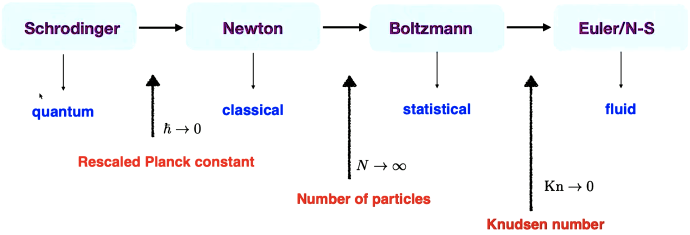
Outline

Part 1: Hamiltonian Simulation (time-independent case)

Different Levels of Physics

multiscale physics fig by Prof. Qin Li

Different Levels of Physics



multiscale physics fig by Prof. Qin Li

Different Levels of Physics

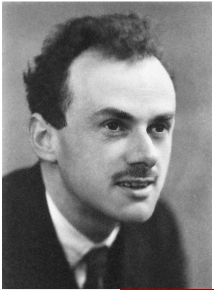
Schrodinger → Newton → Boltzmann → Euler/N-S

quantum | classical | statistical | fluid

$\hbar \to 0$ Rescaled Planck constant

$N \to \infty$ Number of particles

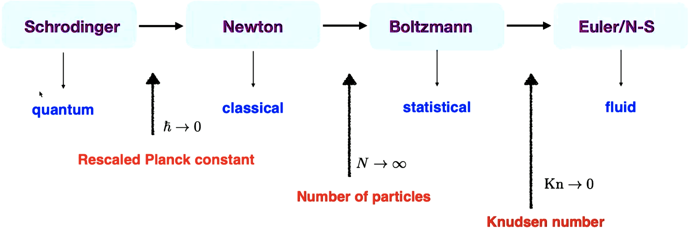
$\mathrm{Kn} \to 0$ Knudsen number

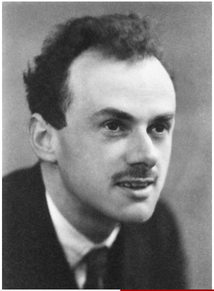


"the underlying physical laws necessary for the mathematical theory of a large part of physics and the whole of chemistry are thus completely known."

Paul A. M. Dirac (1929)

Different Levels of Physics

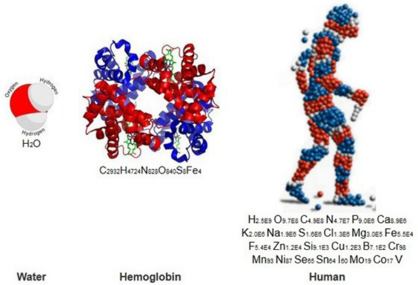




"the underlying physical laws necessary for the mathematical theory of a large part of physics and the whole of chemistry are thus completely known, and the difficulty is only that the exact application of these laws leads to equations much too complicated to be soluble."

Paul A. M. Dirac (1929)

Schrödinger equation for Molecular Dynamics

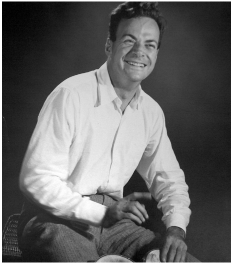


To describe its behaviour: (x: nuclei coordinates, y: electronic coordinates, M: mass of a nucleus, m: mass of an electron.)

$$\hat{H}_{\text{total}} = -\frac{\hbar^2}{2M}\Delta_x - \frac{\hbar^2}{2m}\Delta_y + V(x,y), \quad x \in \mathbb{R}^d, y \in \mathbb{R}^n$$

$$i\hbar\partial_t\psi = \hat{H}_{\text{total}}\psi$$

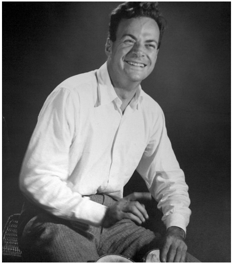
Quantum Computing 101



"... nature isn't classical, dammit, and if you want to make a simulation of nature, you'd better make it quantum mechanical, and by golly it's a wonderful problem, because it doesn't look so easy."

Richard Feynman (1981)

Quantum Computing 101

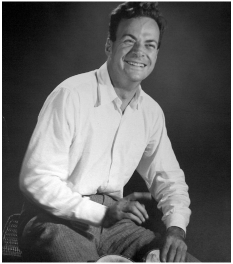


"... nature isn't classical, dammit, and if you want to make a simulation of nature, you'd better make it quantum mechanical, and by golly it's a wonderful problem, because it doesn't look so easy."

Richard Feynman (1981)

Hamiltonian Simulation Problem (original motivation for quantum computers): Given a description of the Hamiltonian $H(t)$, an evolution time t and an initial state $|\psi(0)\rangle$, to produce the final state $|\psi(t)\rangle$ within in some error tolerance ϵ.

Quantum Computing 101



"... nature isn't classical, dammit, and if you want to make a simulation of nature, you'd better make it quantum mechanical, and by golly it's a wonderful problem, because it doesn't look so easy."

Richard Feynman (1981)

Hamiltonian Simulation Problem (original motivation for quantum computers): Given a description of the Hamiltonian $H(t)$, an evolution time t and an initial state $|\psi(0)\rangle$, to produce the final state $|\psi(t)\rangle$ within in some error tolerance ϵ.

$$\mathrm{i}\partial_t |\psi(t)\rangle = H(t) |\psi(t)\rangle, \quad |\psi(0)\rangle = |\psi_0\rangle.$$

To simulate $\mathcal{T}e^{-\mathrm{i}\int_0^t H(s)\,ds}$ for H of very high dimension!

Hamiltonian Simulation

Hamiltonian Simulation Problem: Given a description of the Hamiltonian $H(t)$, an evolution time t and an initial state $|\psi(0)\rangle$, to produce the final state $|\psi(t)\rangle$ within in some error tolerance ϵ.

Hamiltonian Simulation

Hamiltonian Simulation Problem: Given a description of the Hamiltonian $H(t)$, an evolution time t and an initial state $|\psi(0)\rangle$, to produce the final state $|\psi(t)\rangle$ within in some error tolerance ϵ.

Time-independent: $H(t) \equiv H$ is a $2^n \times 2^n$ matrix

$$\mathrm{i}\partial_t |\psi(t)\rangle = H |\psi(t)\rangle, \quad |\psi(0)\rangle = |\psi_0\rangle.$$

Hamiltonian Simulation

Hamiltonian Simulation Problem: Given a description of the Hamiltonian $H(t)$, an evolution time t and an initial state $|\psi(0)\rangle$, to produce the final state $|\psi(t)\rangle$ within in some error tolerance ϵ.

Time-independent: $H(t) \equiv H$ is a $2^n \times 2^n$ matrix

$$\mathrm{i}\partial_t |\psi(t)\rangle = H |\psi(t)\rangle, \quad |\psi(0)\rangle = |\psi_0\rangle.$$

$$\left\| \mathcal{U}_{\text{app}} |\psi_0\rangle - e^{-\mathrm{i}Ht} |\psi_0\rangle \right\|$$

Hamiltonian Simulation

Hamiltonian Simulation Problem: Given a description of the Hamiltonian $H(t)$, an evolution time t and an initial state $|\psi(0)\rangle$, to produce the final state $|\psi(t)\rangle$ within in some error tolerance ϵ.

Time-independent: $H(t) \equiv H$ is a $2^n \times 2^n$ matrix

$$\mathrm{i}\partial_t |\psi(t)\rangle = H |\psi(t)\rangle, \quad |\psi(0)\rangle = |\psi_0\rangle .$$

$$\left\| \mathcal{U}_{\text{app}} |\psi_0\rangle - e^{-\mathrm{i}Ht} |\psi_0\rangle \right\| \leq \left\| \mathcal{U}_{\text{app}} - e^{-\mathrm{i}Ht} \right\| \leq \epsilon.$$

Hamiltonian Simulation

Hamiltonian Simulation Problem: Given a description of the Hamiltonian $H(t)$, an evolution time t and an initial state $|\psi(0)\rangle$, to produce the final state $|\psi(t)\rangle$ within in some error tolerance ϵ.

Time-independent: $H(t) \equiv H$ is a $2^n \times 2^n$ matrix

$$\mathrm{i}\partial_t |\psi(t)\rangle = H |\psi(t)\rangle , \quad |\psi(0)\rangle = |\psi_0\rangle .$$

$$\left\| \mathcal{U}_{\text{app}} |\psi_0\rangle - e^{-\mathrm{i}Ht} |\psi_0\rangle \right\| \leq \boxed{\left\| \mathcal{U}_{\text{app}} - e^{-\mathrm{i}Ht} \right\| \leq \epsilon}.$$

Hamiltonian Simulation

Hamiltonian Simulation Problem: Given a description of the Hamiltonian $H(t)$, an evolution time t and an initial state $|\psi(0)\rangle$, to produce the final state $|\psi(t)\rangle$ within in some error tolerance ϵ.

Time-independent: $H(t) \equiv H$ is a $2^n \times 2^n$ matrix

$$\mathrm{i}\partial_t |\psi(t)\rangle = H |\psi(t)\rangle, \quad |\psi(0)\rangle = |\psi_0\rangle.$$

$$\left\| \mathcal{U}_{\text{app}} |\psi_0\rangle - e^{-\mathrm{i}Ht} |\psi_0\rangle \right\| \leq \boxed{\left\| \mathcal{U}_{\text{app}} - e^{-\mathrm{i}Ht} \right\| \leq \epsilon}.$$

Examples of H: many-body Hamiltonian

$$H = \sum_{E \in S \subset \{I,X,Y,Z\}^{\otimes n}} \lambda_E E,$$

k-local Hamiltonian (TFIM, Heisenberg models, etc), etc.

Error Metrics: Specific case v.s. Worst case

- Taking into account **initial conditions**: consider *vector norm*, instead of operator norm.

$$\left\| \mathcal{U}_{\text{app}} \left|\psi_0\right\rangle - e^{-\mathrm{i}Ht} \left|\psi_0\right\rangle \right\| \leq \left\| \mathcal{U}_{\text{app}} - e^{-\mathrm{i}Ht} \right\| \leq \epsilon.$$

[Sahinoglu-Somma 2021], [An-Fang-Lin 2020], [Tres-Fang 2022], [BornsWeil-Fang 2022], [Zhao-Zhou-Shaw-Li-Childs 2022], etc

Error Metrics: Specific case v.s. Worst case

- Taking into account **initial conditions**: consider *vector norm*, instead of operator norm.

$$\left\|\mathcal{U}_{\text{app}} |\psi_0\rangle - e^{-\mathrm{i}Ht} |\psi_0\rangle\right\| \leq \left\|\mathcal{U}_{\text{app}} - e^{-\mathrm{i}Ht}\right\| \leq \epsilon.$$

- Taking into account the **observable**: consider *observable error bounds*. $\|O\| \leq 1$

$$\begin{aligned}
&\left\|\mathcal{U}_{\text{app}}^\dagger O \mathcal{U}_{\text{app}} - e^{iHt} O e^{-\mathrm{i}Ht}\right\| \\
=&\left\|\mathcal{U}_{\text{app}}^\dagger O \mathcal{U}_{\text{app}} - \mathcal{U}_{\text{app}}^\dagger O e^{-\mathrm{i}Ht} + \mathcal{U}_{\text{app}}^\dagger O e^{-\mathrm{i}Ht} - e^{iHt} O e^{-\mathrm{i}Ht}\right\| \\
\leq& 2\left\|\mathcal{U}_{\text{app}} - e^{-\mathrm{i}Ht}\right\| \leq \epsilon.
\end{aligned}$$

[Sahinoglu-Somma 2021], [An-Fang-Lin 2020], [Tres-Fang 2022], [BornsWeil-Fang 2022], [Zhao-Zhou-Shaw-Li-Childs 2022], etc

Error Metrics: Specific case v.s. Worst case

- Taking into account **initial conditions**: consider *vector norm*, instead of operator norm.

$$\left\|\mathcal{U}_{\text{app}}\left|\psi_0\right\rangle - e^{-\mathrm{i}Ht}\left|\psi_0\right\rangle\right\| \leq \left\|\mathcal{U}_{\text{app}} - e^{-\mathrm{i}Ht}\right\| \leq \epsilon.$$

- Taking into account the **observable**: consider *observable error bounds*. $\|O\| \leq 1$

$$\begin{aligned}
&\left\|\mathcal{U}_{\text{app}}^{\dagger}O\mathcal{U}_{\text{app}} - e^{iHt}Oe^{-\mathrm{i}Ht}\right\| \\
=&\left\|\mathcal{U}_{\text{app}}^{\dagger}O\mathcal{U}_{\text{app}} - \mathcal{U}_{\text{app}}^{\dagger}Oe^{-\mathrm{i}Ht} + \mathcal{U}_{\text{app}}^{\dagger}Oe^{-\mathrm{i}Ht} - e^{iHt}Oe^{-\mathrm{i}Ht}\right\| \\
\leq&2\left\|\mathcal{U}_{\text{app}} - e^{-\mathrm{i}Ht}\right\| \leq \epsilon.
\end{aligned}$$

- Taking into account **both** and consider *observable expectation* $\left|\left\langle\psi_0\right|\mathcal{U}_{\text{app}}^{\dagger}O\mathcal{U}_{\text{app}}\left|\psi_0\right\rangle - \left\langle\psi_0\right|e^{iHt}Oe^{-\mathrm{i}Ht}\left|\psi_0\right\rangle\right|$

[Sahinoglu-Somma 2021], [An-Fang-Lin 2020], [Tres-Fang 2022], [BornsWeil-Fang 2022], [Zhao-Zhou-Shaw-Li-Childs 2022], etc

Specific case v.s. Worst case: Take-away

"Spectrum" of various error measurements:

$$\boxed{\begin{array}{c}\text{Observable}\\\text{expectation}\end{array}} \ll \begin{array}{c}\boxed{\text{Vector norm}}\\\boxed{\text{Observable error bounds}}\end{array} \ll \boxed{\text{Operator norm}}$$

[1] The cost can be improved for product formulas.

Specific case v.s. Worst case: Take-away

"Spectrum" of various error measurements:

$$\boxed{\text{Observable expectation}} \ll \boxed{\text{Vector norm}} \; \boxed{\text{Observable error bounds}} \ll \boxed{\text{Operator norm}}$$

Specific case v.s. **Worst case**
Taking into account the specific instance, the error (and hence the cost) can be improved. [1]

In this lecture, we only focus on the worst case, i.e. error in the operator norm of unitaries.

[1] The cost can be improved for product formulas.

Expected Complexity?

2 [Feymann 1985]

Expected Complexity?

No-fast-forwarding Theorem: *(informal)*
Simulating Hamiltonian dynamics for time t requires complexity $\Omega(t)$.

[2] [Feymann 1985]

Expected Complexity?

No-fast-forwarding Theorem: *(informal)*
Simulating Hamiltonian dynamics for time t requires complexity $\Omega(t)$.

Exponential Quantum Advantage (EQA)? (often this is also used to refer superpolynomial speedup)

Criteria to claim EQA:

2 [Feymann 1985]

Expected Complexity?

No-fast-forwarding Theorem: *(informal)*
Simulating Hamiltonian dynamics for time t requires complexity $\Omega(t)$.

Exponential Quantum Advantage (EQA)? (often this is also used to refer superpolynomial speedup)

Criteria to claim EQA:

- There is a QA with quantum complexity $\leq \mathrm{poly}\,(n)$.

[2] [Feymann 1985]

Expected Complexity?

No-fast-forwarding Theorem: *(informal)*
Simulating Hamiltonian dynamics for time t requires complexity $\Omega(t)$.

Exponential Quantum Advantage (EQA)? (often this is also used to refer superpolynomial speedup)

Criteria to claim EQA:

- There is a QA with quantum complexity $\leq \mathrm{poly}\,(n)$.
- Classical intractable

$$\begin{cases} \text{(A) Best-known Classical Alg. has complexity } \geq e^{\mathrm{poly}\,(n)} \\ \\ \end{cases}$$

[2] [Feymann 1985]

Expected Complexity?

No-fast-forwarding Theorem: *(informal)*
Simulating Hamiltonian dynamics for time t requires complexity $\Omega(t)$.

Exponential Quantum Advantage (EQA)? (often this is also used to refer superpolynomial speedup)

Criteria to claim EQA:

- There is a QA with quantum complexity $\leq \operatorname{poly}(n)$.
- Classical intractable

$$\begin{cases} \text{(A) Best-known Classical Alg. has complexity } \geq e^{\operatorname{poly}(n)} \\ \text{(B) Show that the task is BQP-hard or BQP-complete} \\ \text{(Any Classical Alg. under reasonable complexity conjectures)} \end{cases}$$

2 [Feymann 1985]

Expected Complexity?

No-fast-forwarding Theorem: *(informal)*
Simulating Hamiltonian dynamics for time t requires complexity $\Omega(t)$.

Exponential Quantum Advantage (EQA)? (often this is also used to refer superpolynomial speedup)

Criteria to claim EQA:

- There is a QA with quantum complexity $\leq \mathrm{poly}\,(n)$.
- Classical intractable

$$\begin{cases} \text{(A) Best-known Classical Alg. has complexity } \geq e^{\mathrm{poly}\,(n)} \\ \text{(B) Show that the task is BQP-hard or BQP-complete} \\ \text{(Any Classical Alg. under reasonable complexity conjectures)} \end{cases}$$

Hamiltonian Simulation is BQP-hard.
(Any quantum circuit can be efficiently implemented by the dynamics of a local Hamiltonian. [2])

[2] [Feymann 1985]

Hamiltonian Simulation Algorithms

- Trotterization (= Product Formulae = Time/Operator Splitting)
  1st-order Trotter formula (Lie-Trotter) for $H = H_1 + H_2$

$$e^{-\mathrm{i}Ht} \approx \left(e^{-\mathrm{i}H_2 t/L} e^{-\mathrm{i}H_1 t/L}\right)^L$$

Cost/Complexity?

Hamiltonian Simulation Algorithms

- Trotterization (= Product Formulae = Time/Operator Splitting)
  1st-order Trotter formula (Lie-Trotter) for $H = H_1 + H_2$

  $$e^{-\mathrm{i}Ht} \approx \left(e^{-\mathrm{i}H_2 t/L} e^{-\mathrm{i}H_1 t/L}\right)^L$$

  Cost/Complexity? error estimate and circuit implementation

Hamiltonian Simulation Algorithms

- Trotterization (= Product Formulae = Time/Operator Splitting)
  1st-order Trotter formula (Lie-Trotter) for $H = H_1 + H_2$

$$e^{-\mathrm{i}Ht} \approx \left(e^{-\mathrm{i}H_2 t/L} e^{-\mathrm{i}H_1 t/L}\right)^L$$

Cost/Complexity? error estimate and circuit implementation

$$e^{-\mathrm{i}Ht} = \left(e^{-\mathrm{i}H_2 t/L} e^{-\mathrm{i}H_1 t/L}\right)^L + \mathcal{O}\left(\|[H_1, H_2]\| t^2 / L\right)$$

The number of Trotter steps $L = \mathcal{O}\left(\|[H_1, H_2]\| t^2 / \epsilon\right)$

Hamiltonian Simulation Algorithms

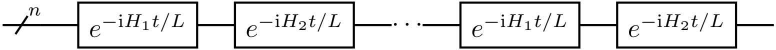
- Trotterization (= Product Formulae = Time/Operator Splitting)
  1st-order Trotter formula (Lie-Trotter) for $H = H_1 + H_2$

$$e^{-\mathrm{i}Ht} \approx \left(e^{-\mathrm{i}H_2 t/L} e^{-\mathrm{i}H_1 t/L}\right)^L$$

  Cost/Complexity? error estimate and circuit implementation

$$e^{-\mathrm{i}Ht} = \left(e^{-\mathrm{i}H_2 t/L} e^{-\mathrm{i}H_1 t/L}\right)^L + \mathcal{O}\left(\|[H_1, H_2]\| t^2/L\right)$$

  The number of Trotter steps $L = \mathcal{O}\left(\|[H_1, H_2]\| t^2/\epsilon\right)$



Hamiltonian Simulation Algorithms

- Trotterization (= Product Formulae = Time/Operator Splitting)
  1st-order Trotter formula (Lie-Trotter) for $H = H_1 + H_2$

$$e^{-\mathrm{i}Ht} \approx \left(e^{-\mathrm{i}H_2 t/L} e^{-\mathrm{i}H_1 t/L}\right)^L$$

Cost/Complexity? error estimate and circuit implementation

$$e^{-\mathrm{i}Ht} = \left(e^{-\mathrm{i}H_2 t/L} e^{-\mathrm{i}H_1 t/L}\right)^L + \mathcal{O}\left(\|[H_1, H_2]\| t^2/L\right)$$

The number of Trotter steps $L = \mathcal{O}\left(\|[H_1, H_2]\| t^2/\epsilon\right)$

$$\overset{n}{\not-}\!\!\!-\boxed{e^{-\mathrm{i}H_1 t/L}}-\boxed{e^{-\mathrm{i}H_2 t/L}}-\cdots-\boxed{e^{-\mathrm{i}H_1 t/L}}-\boxed{e^{-\mathrm{i}H_2 t/L}}-$$

$\Rightarrow \mathcal{O}\left(\|[H_1, H_2]\| t^2/\epsilon\right)$ queries to $e^{-\mathrm{i}H_1 s}$ and $e^{-\mathrm{i}H_2 s}$.

Hamiltonian Simulation Algorithms

- Trotterization (= Product Formulae = Time/Operator Splitting)
  1st-order Trotter formula (Lie-Trotter) for $H = H_1 + H_2$

$$e^{-\mathrm{i}Ht} \approx \left(e^{-\mathrm{i}H_2 t/L} e^{-\mathrm{i}H_1 t/L}\right)^L$$

Cost/Complexity? error estimate and circuit implementation

$$e^{-\mathrm{i}Ht} = \left(e^{-\mathrm{i}H_2 t/L} e^{-\mathrm{i}H_1 t/L}\right)^L + \mathcal{O}\left(\|[H_1, H_2]\| t^2/L\right)$$

The number of Trotter steps $L = \mathcal{O}\left(\|[H_1, H_2]\| t^2/\epsilon\right)$

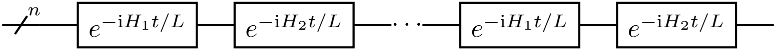
$$\overset{n}{/}\!\!-\boxed{e^{-\mathrm{i}H_1 t/L}}-\boxed{e^{-\mathrm{i}H_2 t/L}}-\cdots-\boxed{e^{-\mathrm{i}H_1 t/L}}-\boxed{e^{-\mathrm{i}H_2 t/L}}-$$

$\Rightarrow \mathcal{O}\left(\|[H_1, H_2]\| t^2/\epsilon\right)$ queries to $e^{-\mathrm{i}H_1 s}$ and $e^{-\mathrm{i}H_2 s}$.

High order (p-th): query complexity $\mathcal{O}\left(\alpha_H t^{1+1/p}/\epsilon^{1/p}\right)$. [3]

Hamiltonian Simulation Algorithms

- Trotterization (= Product Formulae = Time/Operator Splitting)
  1st-order Trotter formula (Lie-Trotter) for $H = H_1 + H_2$



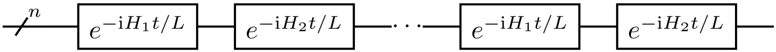
  High order (p-th): query complexity $\mathcal{O}\left(\alpha_H t^{1+1/p}/\epsilon^{1/p}\right)$. [3]
  - Everything is unitary! No ancilla needed.
  - But it needs $e^{-\mathrm{i}H_j s}$ efficiently implementable.

[3] [Suzuki 91], [Descombes-Thalhammer 2010], [Childs-Su-et Al. 2021]

[4] truncated Taylor [Berry-Childs-Cleve-Kothari-Somma 2015], QSP/QSVT [Low-Chuang 2016 and 2017], [Gilyen-Su-Low-Wiebe 2019]

Hamiltonian Simulation Algorithms

- Trotterization (= Product Formulae = Time/Operator Splitting) 1st-order Trotter formula (Lie-Trotter) for $H = H_1 + H_2$



  High order (p-th): query complexity $\mathcal{O}\left(\alpha_H t^{1+1/p}/\epsilon^{1/p}\right)$.
  - Everything is unitary! No ancilla needed.
  - But it needs $e^{-\mathrm{i}H_j s}$ efficiently implementable.
- Post-Trotter, e.g., truncated Taylor series, quantum signal processing (QSP), quantum singular value transformation (QSVT) [3], etc.

$$e^{-\mathrm{i}Ht} \approx \sum_{k=0}^{K} \frac{(-\mathrm{i}Ht)^k}{k!} = \sum_{k=0}^{K} \sum_{\ell_1,\cdots,\ell_k} \frac{(-\mathrm{i}t)^k}{k!} H_{\ell_1} H_{\ell_2} \cdots H_{\ell_k}.$$

Upshot: $\Rightarrow \mathcal{O}(t \log(t/\epsilon))$

[3] truncated Taylor [Berry-Childs-Cleve-Kothari-Somma 2015], QSP/QSVT [Low-Chuang 2016 and 2017], [Gilyen-Su-Low-Wiebe 2019]

Hamiltonian Simulation Algorithms

- Trotterization (= Product Formulae = Time/Operator Splitting) 1st-order Trotter formula (Lie-Trotter) for $H = H_1 + H_2$

$$\overset{n}{/}\!-\!\boxed{e^{-\mathrm{i}H_1 t/L}}\!-\!\boxed{e^{-\mathrm{i}H_2 t/L}}\!-\cdots-\!\boxed{e^{-\mathrm{i}H_1 t/L}}\!-\!\boxed{e^{-\mathrm{i}H_2 t/L}}\!-$$

  High order (p-th): query complexity $\mathcal{O}\left(\alpha_H t^{1+1/p}/\epsilon^{1/p}\right)$.
  - Everything is unitary! No ancilla needed.
  - But it needs $e^{-\mathrm{i}H_j s}$ efficiently implementable.
- Post-Trotter, e.g., truncated Taylor series, quantum signal processing (QSP), quantum singular value transformation (QSVT) [3], etc.

$$e^{-\mathrm{i}Ht} \approx \sum_{k=0}^{K} \frac{(-\mathrm{i}Ht)^k}{k!} = \sum_{k=0}^{K} \sum_{\ell_1,\cdots,\ell_k} \frac{(-\mathrm{i}t)^k}{k!} H_{\ell_1} H_{\ell_2} \cdots H_{\ell_k}.$$

Upshot: $\Rightarrow \mathcal{O}(t \log(t/\epsilon))$ $\quad\Rightarrow$ Even better, say, $\mathcal{O}(t + \log(1/\epsilon))$?

[3] truncated Taylor [Berry-Childs-Cleve-Kothari-Somma 2015], QSP/QSVT [Low-Chuang 2016 and 2017], [Gilyen-Su-Low-Wiebe 2019]

Block-Encoding – Definition

Let A be a general $2^n \times 2^n$ matrix.

Idea:

$$U_A = \begin{pmatrix} A & * \\ * & * \end{pmatrix} \quad \rightarrow \text{ancilla qubits}$$

Block-Encoding – Definition

Let A be a general $2^n \times 2^n$ matrix. $\|A\| \leq \alpha$

Idea:

$$U_A = \begin{pmatrix} \frac{A}{\alpha} & * \\ * & * \end{pmatrix} \quad \to \text{ancilla qubits,}$$

Block-Encoding – Definition

Let A be a general $2^n \times 2^n$ matrix. $\|A\| \leq \alpha$

Idea:

$$U_A = \begin{pmatrix} \frac{\tilde{A}}{\alpha} & * \\ * & * \end{pmatrix} \qquad \left\|\tilde{A} - A\right\| \leq \epsilon$$

$\rightarrow m$ ancilla qubits,

Block-Encoding – Definition

Let A be a general $2^n \times 2^n$ matrix. $\|A\| \le \alpha$

Idea:

$$U_A = \begin{pmatrix} \frac{\tilde{A}}{\alpha} & * \\ * & * \end{pmatrix} \qquad \begin{array}{c} \left\| \tilde{A} - A \right\| \le \epsilon \\ \to m \text{ ancilla qubits,} \end{array}$$

Definition (Block-encoding)

U_A is an (α, m, ϵ)-block-encoding of A, if

$$\|A - \alpha \left(\langle 0^m| \otimes I_n\right) U_A \left(|0^m\rangle \otimes I_n\right)\| \le \epsilon,$$

for some $\alpha \ge \|A\|$, $m > 0$ and $\epsilon > 0$. Here α is called the *subnormalization* factor and m is the number of ancilla qubits, and n is the number of system qubits. When $\epsilon = 0$, it is also called an (α, m)-block-encoding.

Block-Encoding – Definition

Let A be a general $2^n \times 2^n$ matrix. $\|A\| \leq \alpha$

Idea:

$$U_A = \begin{pmatrix} \frac{\tilde{A}}{\alpha} & * \\ * & * \end{pmatrix} \qquad \left\|\tilde{A} - A\right\| \leq \epsilon$$

$\rightarrow m$ ancilla qubits,

Definition (Block-encoding)

U_A is an (α, m, ϵ)-block-encoding of A, if

$$\|A - \alpha\left(\langle 0^m| \otimes I_n\right) U_A \left(|0^m\rangle \otimes I_n\right)\| \leq \epsilon,$$

for some $\alpha \geq \|A\|$, $m > 0$ and $\epsilon > 0$. Here α is called the *subnormalization* factor and m is the number of ancilla qubits, and n is the number of system qubits. When $\epsilon = 0$, it is also called an (α, m)-block-encoding.

Understanding: $U_A : 2^{m+n} \times 2^{m+n}$.

Block-Encoding – Definition

Let A be a general $2^n \times 2^n$ matrix. $\|A\| \leq \alpha$

Idea:

$$U_A = \begin{pmatrix} \frac{\tilde{A}}{\alpha} & * \\ * & * \end{pmatrix} \qquad \left\|\tilde{A} - A\right\| \leq \epsilon$$

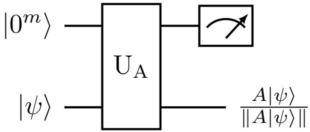
$\rightarrow m$ ancilla qubits,

Definition (Block-encoding)

U_A is an (α, m, ϵ)-block-encoding of A, if

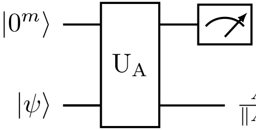
$$\|A - \alpha\left(\langle 0^m| \otimes I_n\right) U_A \left(|0^m\rangle \otimes I_n\right)\| \leq \epsilon,$$

for some $\alpha \geq \|A\|$, $m > 0$ and $\epsilon > 0$.



(upon getting 0 in measurement)

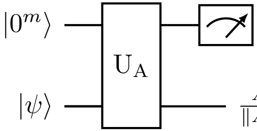
Block-Encoding – Definition cont'd



$\frac{A|\psi\rangle}{\|A|\psi\rangle\|}$ (upon getting 0 in measurement)

$$|0,\psi\rangle = |0\rangle \otimes |\psi\rangle = \begin{pmatrix} |\psi\rangle \\ 0 \end{pmatrix}, \quad U_A\,|0,\psi\rangle = \begin{pmatrix} \frac{\tilde{A}}{\alpha} & * \\ * & * \end{pmatrix} \begin{pmatrix} |\psi\rangle \\ 0 \end{pmatrix} = \begin{pmatrix} \tilde{A}\,|\psi\rangle \\ * \end{pmatrix}.$$

Block-Encoding – Definition cont'd

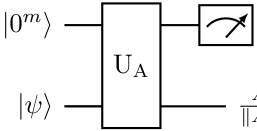


$\frac{A|\psi\rangle}{\||A|\psi\rangle\||}$ (upon getting 0 in measurement)

$$|0,\psi\rangle = |0\rangle \otimes |\psi\rangle = \begin{pmatrix} |\psi\rangle \\ 0 \end{pmatrix}, \quad U_A\,|0,\psi\rangle = \begin{pmatrix} \frac{\tilde{A}}{\alpha} & * \\ * & * \end{pmatrix} \begin{pmatrix} |\psi\rangle \\ 0 \end{pmatrix} = \begin{pmatrix} \tilde{A}\,|\psi\rangle \\ * \end{pmatrix}.$$

Question: Well-defined? Not an empty set?

Block-Encoding – Definition cont'd



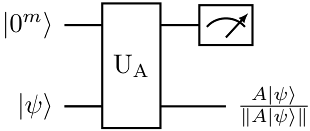
$\frac{A|\psi\rangle}{\| A|\psi\rangle \|}$ (upon getting 0 in measurement)

$$|0, \psi\rangle = |0\rangle \otimes |\psi\rangle = \begin{pmatrix} |\psi\rangle \\ 0 \end{pmatrix}, \quad U_A |0, \psi\rangle = \begin{pmatrix} \frac{\tilde{A}}{\alpha} & * \\ * & * \end{pmatrix} \begin{pmatrix} |\psi\rangle \\ 0 \end{pmatrix} = \begin{pmatrix} \tilde{A} |\psi\rangle \\ * \end{pmatrix}.$$

Question: Well-defined? Not an empty set?

- Trivial example (unitary): U is a $(1, 0, 0)$-block-encoding of U.

Block-Encoding – Definition cont'd



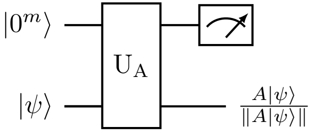
(upon getting 0 in measurement)

$$|0,\psi\rangle = |0\rangle \otimes |\psi\rangle = \begin{pmatrix} |\psi\rangle \\ 0 \end{pmatrix}, \quad U_A\,|0,\psi\rangle = \begin{pmatrix} \frac{\tilde{A}}{\alpha} & * \\ * & * \end{pmatrix} \begin{pmatrix} |\psi\rangle \\ 0 \end{pmatrix} = \begin{pmatrix} \tilde{A}\,|\psi\rangle \\ * \end{pmatrix}.$$

Question: Well-defined? Not an empty set?

- Trivial example (unitary): U is a $(1,0,0)$-block-encoding of U.
- $(\alpha, 1)$-block-encoding is general. WLOG, assume $\|A\| \leq 1$.

Block-Encoding – Definition cont'd



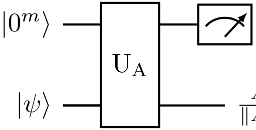
(upon getting 0 in measurement)

Question: Well-defined? Not an empty set?

- Trivial example (unitary): U is a $(1, 0, 0)$-block-encoding of U.
- $(\alpha, 1)$-block-encoding is general. WLOG, assume $\|A\| \leq 1$.
  Proof: $A = W\Sigma V^\dagger$. All singular values $\in [0, 1]$.

References: Lecture notes by Lin Lin, QSVT [Gilyen-Su-Low-Wiebe 2018/2019], see also QSP [Low-Chuang 2017], qubitization [Low-Chuang 2016]

Block-Encoding – Definition cont'd



$\frac{A|\psi\rangle}{\|A|\psi\rangle\|}$ (upon getting 0 in measurement)

Question: Well-defined? Not an empty set?

- Trivial example (unitary): U is a $(1, 0, 0)$-block-encoding of U.
- $(\alpha, 1)$-block-encoding is general. WLOG, assume $\|A\| \leq 1$.
  Proof: $A = W\Sigma V^\dagger$. All singular values $\in [0, 1]$.

$$U_A := \begin{pmatrix} W & 0 \\ 0 & I_n \end{pmatrix} \begin{pmatrix} \Sigma & \sqrt{I_n - \Sigma^2} \\ \sqrt{I_n - \Sigma^2} & -\Sigma \end{pmatrix} \begin{pmatrix} V^\dagger & 0 \\ 0 & I_n \end{pmatrix}$$
$$= \begin{pmatrix} A & W\sqrt{I_n - \Sigma^2} \\ \sqrt{I_n - \Sigma^2}V^\dagger & -\Sigma \end{pmatrix}$$

References: Lecture notes by Lin Lin, QSVT [Gilyen-Su-Low-Wiebe 2018/2019], see also QSP [Low-Chuang 2017], qubitization [Low-Chuang 2016]

Sparse Input Model

Question: Efficient to construct?

[Gilyen-Su-Low-Wiebe 2018], see also [Berry-Childs 2012], [Low-Chuang 2017]

Sparse Input Model

Question: Efficient to construct?

Upshot: sparse matrix (Hamiltonian) is ok!

[Gilyen-Su-Low-Wiebe 2018], see also [Berry-Childs 2012], [Low-Chuang 2017]

Sparse Input Model

Question: Efficient to construct?

Upshot: sparse matrix (Hamiltonian) is ok!

We assume that H is a s-sparse matrix with $\|H\|_{\max} \le 1$. The information of H is given through the following oracles:

$$\begin{aligned} U_{\text{row}} \left|j, s\right\rangle &= \left|j, \text{row}(j,s)\right\rangle, \\ U_{\text{col}} \left|j, s\right\rangle &= \left|j, \text{col}(j,s)\right\rangle, \\ U_{\text{val}} \left|j, k, z\right\rangle &= \left|j, k, z \oplus H_{jk}(t)\right\rangle. \end{aligned} \tag{1}$$

Here $\text{row}(j,s)$ is the row index of the sth nonzero element in the jth column, $\text{col}(j,s)$ is the column index of the sth nonzero element in the jth row.

A $(s, n+3, \epsilon)$-block-encoding of H can be constructed via $\mathcal{O}(1)$ queries to above oracles and $\mathcal{O}(n + \log^{5/2}(s/\epsilon)$ primitive gates.

[Gilyen-Su-Low-Wiebe 2018], see also [Berry-Childs 2012], [Low-Chuang 2017]

Block-Encoding – Properties

Properties: Let U_A be an (α, a, ϵ)-BE of A; U_B be a (β, b, δ)-BE of B

Block-Encoding – Properties

Properties: Let U_A be an (α, a, ϵ)-BE of A; U_B be a (β, b, δ)-BE of B

1. (BE of cA)

Block-Encoding – Properties

Properties: Let U_A be an (α, a, ϵ)-BE of A; U_B be a (β, b, δ)-BE of B

1. (BE of cA) U_A is an $(c\alpha, a, c\epsilon)$-BE of cA.

Block-Encoding – Properties

Properties: Let U_A be an (α, a, ϵ)-BE of A; U_B be a (β, b, δ)-BE of B

1. (BE of cA) U_A is an $(c\alpha, a, c\epsilon)$-BE of cA.
2. (BE of AB)
   $W = (I_b \otimes U_A)(I_a \otimes U_B)$ is an $(\alpha\beta, a+b, \alpha\delta + \beta\epsilon)$-BE of AB.

Block-Encoding – Properties

Properties: Let U_A be an (α, a, ϵ)-BE of A; U_B be a (β, b, δ)-BE of B

1. (BE of cA) U_A is an $(c\alpha, a, c\epsilon)$-BE of cA.
2. (BE of AB)
   $W = (I_b \otimes U_A)(I_a \otimes U_B)$ is an $(\alpha\beta, a+b, \alpha\delta + \beta\epsilon)$-BE of AB.
   Proof:

$$\left\| AB - \alpha\beta(\langle 0|^{\otimes a+b} \otimes I)(I_b \otimes U_A)(I_a \otimes U_B)(|0\rangle^{\otimes a+b} \otimes I) \right\|$$
$$= \left\| AB - \underbrace{\alpha(\langle 0|^{\otimes a} \otimes I)U_A(|0\rangle^{\otimes a} \otimes I)}_{\tilde{A}} \underbrace{\beta(\langle 0|^{\otimes b} \otimes I)U_B(|0\rangle^{\otimes b} \otimes I)}_{\tilde{B}} \right\|$$

Block-Encoding – Properties

Properties: Let U_A be an (α, a, ϵ)-BE of A; U_B be a (β, b, δ)-BE of B

1. (BE of cA) U_A is an $(c\alpha, a, c\epsilon)$-BE of cA.
2. (BE of AB)
   $W = (I_b \otimes U_A)(I_a \otimes U_B)$ is an $(\alpha\beta, a+b, \alpha\delta + \beta\epsilon)$-BE of AB.
   Proof:

$$\left\| AB - \alpha\beta(\langle 0|^{\otimes a+b} \otimes I)(I_b \otimes U_A)(I_a \otimes U_B)(|0\rangle^{\otimes a+b} \otimes I) \right\|$$

$$= \left\| AB - \underbrace{\alpha(\langle 0|^{\otimes a} \otimes I)U_A(|0\rangle^{\otimes a} \otimes I)}_{\tilde{A}}\underbrace{\beta(\langle 0|^{\otimes b} \otimes I)U_B(|0\rangle^{\otimes b} \otimes I)}_{\tilde{B}} \right\|$$

$$= \left\| AB - \tilde{A}B + \tilde{A}B - \tilde{A}\tilde{B} \right\| = \left\| (A - \tilde{A})B + \tilde{A}(B - \tilde{B}) \right\|$$

Block-Encoding – Properties

Properties: Let U_A be an (α, a, ϵ)-BE of A; U_B be a (β, b, δ)-BE of B

1. (BE of cA) U_A is an $(c\alpha, a, c\epsilon)$-BE of cA.
2. (BE of AB)
   $W = (I_b \otimes U_A)(I_a \otimes U_B)$ is an $(\alpha\beta, a+b, \alpha\delta + \beta\epsilon)$-BE of AB.
   Proof:

$$
\begin{aligned}
&\left\| AB - \alpha\beta(\langle 0|^{\otimes a+b} \otimes I)(I_b \otimes U_A)(I_a \otimes U_B)(|0\rangle^{\otimes a+b} \otimes I) \right\| \\
=&\left\| AB - \underbrace{\alpha(\langle 0|^{\otimes a} \otimes I)U_A(|0\rangle^{\otimes a} \otimes I)}_{\tilde{A}}\underbrace{\beta(\langle 0|^{\otimes b} \otimes I)U_B(|0\rangle^{\otimes b} \otimes I)}_{\tilde{B}} \right\| \\
=&\left\| AB - \tilde{A}B + \tilde{A}B - \tilde{A}\tilde{B} \right\| = \left\| (A - \tilde{A})B + \tilde{A}(B - \tilde{B}) \right\| \\
\leq&\alpha\delta + \beta\epsilon.
\end{aligned}
$$

Block-Encoding – Properties

Properties: Let U_A be an (α, a, ϵ)-BE of A; U_B be a (β, b, δ)-BE of B

1. (BE of cA) U_A is an $(c\alpha, a, c\epsilon)$-BE of cA.
2. (BE of AB)
   $W = (I_b \otimes U_A)(I_a \otimes U_B)$ is an $(\alpha\beta, a+b, \alpha\delta + \beta\epsilon)$-BE of AB.
3. (BE of $A + B$)

Block-Encoding – Properties

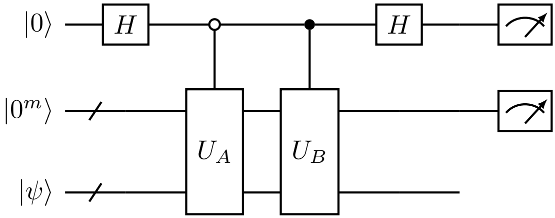
Properties: Let U_A be an (α, a, ϵ)-BE of A; U_B be a (β, b, δ)-BE of B

1. (BE of cA) U_A is an $(c\alpha, a, c\epsilon)$-BE of cA.
2. (BE of AB)
   $W = (I_b \otimes U_A)(I_a \otimes U_B)$ is an $(\alpha\beta, a+b, \alpha\delta + \beta\epsilon)$-BE of AB.
3. (BE of $A+B$) U_A: $(1, m, \epsilon)$-BE of A; U_B: $(1, m, \delta)$-BE of B

Block-Encoding – Properties

Properties: Let U_A be an (α, a, ϵ)-BE of A; U_B be a (β, b, δ)-BE of B

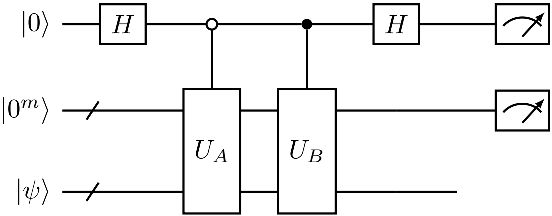
1. (BE of cA) U_A is an $(c\alpha, a, c\epsilon)$-BE of cA.
2. (BE of AB)
   $W = (I_b \otimes U_A)(I_a \otimes U_B)$ is an $(\alpha\beta, a+b, \alpha\delta + \beta\epsilon)$-BE of AB.
3. (BE of $A+B$) U_A: $(1, m, \epsilon)$-BE of A; U_B: $(1, m, \delta)$-BE of B
   The following circuit constructs a $(2, m, \delta+\epsilon)$-BE of $A+B$.



Block-Encoding – Properties

Properties: Let U_A be an (α, a, ϵ)-BE of A; U_B be a (β, b, δ)-BE of B

1. (BE of cA) U_A is an $(c\alpha, a, c\epsilon)$-BE of cA.
2. (BE of AB)
   $W = (I_b \otimes U_A)(I_a \otimes U_B)$ is an $(\alpha\beta, a + b, \alpha\delta + \beta\epsilon)$-BE of AB.
3. (BE of $A + B$) U_A: $(1, m, \epsilon)$-BE of A; U_B: $(1, m, \delta)$-BE of B
   The following circuit constructs a $(2, m, \delta + \epsilon)$-BE of $A + B$.



More generally, linear combination of block-encodings can be constructed via Linear Combination of Unitaries (LCU) Lemma.

LCU Lemma

LCU Lemma: $T = \sum_{j \in [L]} c_j U_j$ for unitaries U_j. $\|c\|_1 = \sum_{j \in [L]} |c_j|$.

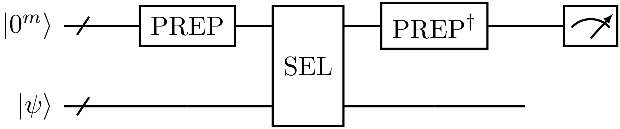
LCU [Berry-Childs-Kothari 2015], General LCBE [Gilyen-Su-Low-Wiebe 2018]

LCU Lemma

LCU Lemma: $T = \sum_{j\in[L]} c_j U_j$ for unitaries U_j. $\|c\|_1 = \sum_{j\in[L]} |c_j|$.

One can get a $(\|c\|_1, \lceil \log_2 L \rceil)$-block-encoding by:

- $\mathrm{SEL} := \sum_{j\in[L]} |j\rangle\langle j| \otimes U_j$
- $\mathrm{PREP}\,|0\rangle = \frac{1}{\sqrt{\|c\|_1}} \sum_{j\in[L]} \sqrt{c_j}\, |j\rangle$.



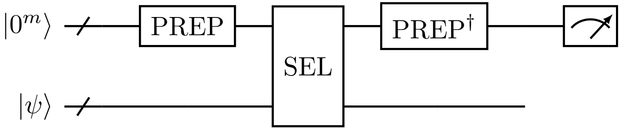
LCU [Berry-Childs-Kothari 2015], General LCBE [Gilyen-Su-Low-Wiebe 2018]

LCU Lemma

LCU Lemma: $T = \sum_{j\in[L]} c_j U_j$ for unitaries U_j. $\|c\|_1 = \sum_{j\in[L]} |c_j|$.

One can get a $(\|c\|_1, \lceil \log_2 L \rceil)$-block-encoding by:

- $\mathrm{SEL} := \sum_{j\in[L]} |j\rangle\langle j| \otimes U_j$
- $\mathrm{PREP}|0\rangle = \frac{1}{\sqrt{\|c\|_1}} \sum_{j\in[L]} \sqrt{c_j}\,|j\rangle$.



General LCBE: $\max_j m_j + \lceil \log_2 L \rceil$ ancillas

LCU [Berry-Childs-Kothari 2015], General LCBE [Gilyen-Su-Low-Wiebe 2018]

Matrix Function and Truncated Taylor series

We can "$+$" and "$\times$" $\Rightarrow$ we can BE $\mathrm{poly}(A)$

Matrix Function and Truncated Taylor series

We can "$+$" and "$\times$" $\Rightarrow$ we can BE $\mathrm{poly}(A)$

$\Rightarrow$ $\boxed{\text{We can BE } f(A).}$ Super Powerful!!!

Matrix Function and Truncated Taylor series

We can "$+$" and "$\times$" $\Rightarrow$ we can BE $\mathrm{poly}(A)$

$\Rightarrow$ $\boxed{\text{We can BE } f(A).}$ Super Powerful!!!

e.g., e^{-iHt} Hamiltonian Simulation, $e^{-\beta H}$ Gibbs distribution, A^{-1} matrix inversion, etc.

Matrix Function and Truncated Taylor series

We can "$+$" and "$\times$" $\Rightarrow$ we can BE $\mathrm{poly}(A)$

$\Rightarrow$ $\boxed{\text{We can BE } f(A).}$ Super Powerful!!!

e.g., e^{-iHt} Hamiltonian Simulation, $e^{-\beta H}$ Gibbs distribution, A^{-1} matrix inversion, etc.

Truncated Taylor Series $\|H\| \leq \alpha$.

$$e^{-\mathrm{i}H\Delta t} \approx \sum_{k=0}^{K} \frac{(-\mathrm{i}H\Delta t)^k}{k!} = \sum_{k=0}^{K} \sum_{\ell_1,\cdots,\ell_k} \frac{(-\mathrm{i}\Delta t)^k}{k!} H_{\ell_1} H_{\ell_2} \cdots H_{\ell_k}.$$

Number of time steps $L = t/\Delta t = \mathcal{O}(\alpha t)$; $K = \mathcal{O}(\frac{\log(t/\epsilon)}{\log\log(t/\epsilon)})$

$\Rightarrow$ Query Complexity: $\mathcal{O}(\alpha t \frac{\log(t/\epsilon)}{\log\log(t/\epsilon)})$.

Matrix Function and Truncated Taylor series

We can "$+$" and "$\times$" $\Rightarrow$ we can BE $\mathrm{poly}(A)$

$\Rightarrow$ $\boxed{\text{We can BE } f(A).}$ Super Powerful!!!

e.g., e^{-iHt} Hamiltonian Simulation, $e^{-\beta H}$ Gibbs distribution, A^{-1} matrix inversion, etc.

Truncated Taylor Series $\|H\| \leq \alpha$.

$$e^{-\mathrm{i}H\Delta t} \approx \sum_{k=0}^{K} \frac{(-\mathrm{i}H\Delta t)^k}{k!} = \sum_{k=0}^{K} \sum_{\ell_1,\cdots,\ell_k} \frac{(-\mathrm{i}\Delta t)^k}{k!} H_{\ell_1} H_{\ell_2} \cdots H_{\ell_k}.$$

Number of time steps $L = t/\Delta t = \mathcal{O}(\alpha t)$; $K = \mathcal{O}(\frac{\log(t/\epsilon)}{\log\log(t/\epsilon)})$

$\Rightarrow$ Query Complexity: $\mathcal{O}(\alpha t \frac{\log(t/\epsilon)}{\log\log(t/\epsilon)})$.

But $A + A^2 + \cdots + A^d$
Number of ancillas: $m; 2m; \cdots; dm \Rightarrow dm + \log(d)$ HUGE!

Question: Can we do better?

Matrix Function and Truncated Taylor series

We can "$+$" and "$\times$" $\Rightarrow$ we can BE $\mathrm{poly}(A)$

$\Rightarrow$ $\boxed{\text{We can BE } f(A).}$ Super Powerful!!!

e.g., e^{-iHt} Hamiltonian Simulation, $e^{-\beta H}$ Gibbs distribution, A^{-1} matrix inversion, etc.

But $A + A^2 + \cdots + A^d$
Number of ancillas: $m; 2m; \cdots ; dm \Rightarrow dm + \log(d)$ HUGE!

Question: Can we do better? Yes! 1 additional ancilla is sufficient!

Quantum Singular Value Transformation (QSVT) / Quantum Signal Processing (QSP)

QSVT

$A = W\Sigma V^\dagger \quad f^\diamond(A) := W f(\Sigma) V^\dagger$ Generalized Matrix Function

QSVT

$A = W\Sigma V^\dagger \quad f^\diamond(A) := W f(\Sigma) V^\dagger$ Generalized Matrix Function

Theorem (QSVT with odd real polynomial)

Let U_A be a $(1, m)$-block-encoding of $A \in \mathbb{C}^{2^n \times 2^n}$. Given an odd polynomial $P_{\Re}(x) \in \mathbb{R}[x]$ of odd degree d satisfying

$$|P_{\Re}(x)| \leqslant 1, \forall x \in [-1, 1].$$

We can find a sequence of phase factors $\Phi \in \mathbb{R}^{d+1}$ and construct a $(1, m+1)$-block-encoding of $P^\diamond_{\Re}(A)$ that uses $U_A, U_A^\dagger$, m-qubit controlled NOT, and single qubit rotation gates for $\mathcal{O}(d)$ times.

QSVT

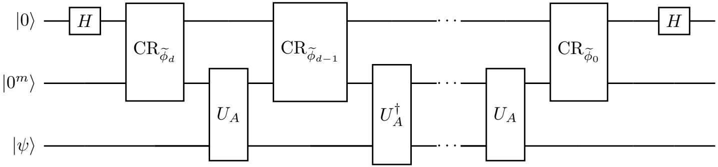
$A = W\Sigma V^\dagger \quad f^\diamond(A) := W f(\Sigma) V^\dagger$ Generalized Matrix Function

Theorem (QSVT with odd real polynomial)

Let U_A be a $(1, m)$-block-encoding of $A \in \mathbb{C}^{2^n \times 2^n}$. Given an odd polynomial $P_{\Re}(x) \in \mathbb{R}[x]$ of odd degree d satisfying

$$|P_{\Re}(x)| \leqslant 1, \forall x \in [-1, 1].$$

We can find a sequence of phase factors $\Phi \in \mathbb{R}^{d+1}$ and construct a $(1, m+1)$-block-encoding of $P_{\Re}^\diamond(A)$ that uses $U_A, U_A^\dagger$, m-qubit controlled NOT, and single qubit rotation gates for $\mathcal{O}(d)$ times.



QSVT (Hermitian matrix + arbitary parity)

Theorem (QSVT for Polynomial eigenvalue transformation)
Let U_A be a (α, m, ϵ)-block-encoding of a Hermitian matrix $A \in \mathbb{C}^{2^n \times 2^n}$. Given a d-degree polynomial $P_{\Re}(x) \in \mathbb{R}[x]$ satisfying

$$|P_{\Re}(x)| \leqslant 1/2, \forall x \in [-1, 1].$$

Then for $\delta \geq 0$, there is a quantum circuit that constructs a $(1, m+2, 4d\sqrt{\epsilon/\alpha} + \delta)$-block-encoding of $P_{\Re}^{\diamond}(A/\alpha)$ that uses a single application of controlled-U_A, and d applications of $U_A, U_A^{\dagger}$, and $\mathcal{O}((m+1)d)$ other one- and two-qubit gates.

Optimal Hamiltonian Simulation by QSVT

Given U_H: an $(\alpha, m, 0)$-block-encoding of H.
Goal: an algorithm that makes $\mathcal{O}(t + \log(1/\epsilon))$ queries to U_H.

Optimal Hamiltonian Simulation by QSVT

Given U_H: an $(\alpha, m, 0)$-block-encoding of H.
Goal: an algorithm that makes $\mathcal{O}(t + \log(1/\epsilon))$ queries to U_H.

- $e^{iHt} = e^{i\frac{H}{\alpha}\alpha t}$. WLOG, assume $\alpha = 1$.

Optimal Hamiltonian Simulation by QSVT

Given U_H: an $(\alpha, m, 0)$-block-encoding of H.
Goal: an algorithm that makes $\mathcal{O}(t + \log(1/\epsilon))$ queries to U_H.

- $e^{iHt} = e^{i\frac{H}{\alpha}\alpha t}$. WLOG, assume $\alpha = 1$. $e^{itx} = \cos(tx) + i\sin(tx)$

Optimal Hamiltonian Simulation by QSVT

Given U_H: an $(\alpha, m, 0)$-block-encoding of H.
Goal: an algorithm that makes $\mathcal{O}(t + \log(1/\epsilon))$ queries to U_H.

- $e^{iHt} = e^{i\frac{H}{\alpha}\alpha t}$. WLOG, assume $\alpha = 1$. $e^{itx} = \cos(tx) + i\sin(tx)$
- Jacobi-Anger expansion on $[-1, 1]$:

$$\cos(tx) = J_0(t) + 2\sum_{k=1}^{\infty}(-1)^k J_{2k}(t) T_{2k}(x),$$

$$\sin(tx) = 2\sum_{k=0}^{\infty}(-1)^k J_{2k+1}(t) T_{2k+1}(x).$$

$J_\nu(t)$ denotes Bessel functions of the first kind.

Optimal Hamiltonian Simulation by QSVT

Given U_H: an $(\alpha, m, 0)$-block-encoding of H.
Goal: an algorithm that makes $\mathcal{O}(t + \log(1/\epsilon))$ queries to U_H.

- $e^{iHt} = e^{i\frac{H}{\alpha}\alpha t}$. WLOG, assume $\alpha = 1$. $e^{itx} = \cos(tx) + i\sin(tx)$
- Jacobi-Anger expansion on $[-1, 1]$:

$$\cos(tx) = J_0(t) + 2\sum_{k=1}^{\infty}(-1)^k J_{2k}(t)T_{2k}(x),$$

$$\sin(tx) = 2\sum_{k=0}^{\infty}(-1)^k J_{2k+1}(t)T_{2k+1}(x).$$

- This series converges rapidly. Truncating it with

$$r = \Theta\left(t + \frac{\log(1/\epsilon)}{\log(e + \log(1/\epsilon)/t)}\right)$$

terms gives a polynomial approximation (with precision ϵ and degree $2r+1$) of $\cos(tx) + i\sin(tx) = e^{itx}$.

Optimal Hamiltonian Simulation by QSVT

Given U_H: an $(\alpha, m, 0)$-block-encoding of H.
Goal: an algorithm that makes $\mathcal{O}(t + \log(1/\epsilon))$ queries to U_H.

- $e^{iHt} = e^{i\frac{H}{\alpha}\alpha t}$. WLOG, assume $\alpha = 1$. $e^{itx} = \cos(tx) + i\sin(tx)$

Query Complexity: ($\alpha \geq \|H\|$.)

$$\mathcal{O}\left(\alpha t + \frac{\log(1/\epsilon)}{\log(e + \log(1/\epsilon)/(\alpha t))}\right).$$

Optimal Hamiltonian Simulation by QSVT cont'd

Given U_H: an $(\alpha, m, 0)$-block-encoding of H.
Goal: an algorithm that makes $\mathcal{O}(t + \log(1/\epsilon))$ queries to U_H.

- We now have block-encodings of both $\cos(tH)$ and $i\sin(tH)$ and use LCU.

4 [Berry-Childs-Cleve-Kothari-Somma 2014/2015]

Optimal Hamiltonian Simulation by QSVT cont'd

Given U_H: an $(\alpha, m, 0)$-block-encoding of H.
Goal: an algorithm that makes $\mathcal{O}(t + \log(1/\epsilon))$ queries to U_H.

- We now have block-encodings of both $\cos(tH)$ and $i\sin(tH)$ and use LCU. Or equivalently, directly use QSVT for polynomial eigenvalue transformation with arbitrary parity. We have a $(1, m+2, \epsilon/6)$ of $e^{itH}/2$.

[4] [Berry-Childs-Cleve-Kothari-Somma 2014/2015]

Optimal Hamiltonian Simulation by QSVT cont'd

Given U_H: an $(\alpha, m, 0)$-block-encoding of H.
Goal: an algorithm that makes $\mathcal{O}(t + \log(1/\epsilon))$ queries to U_H.

- We now have block-encodings of both $\cos(tH)$ and $i\sin(tH)$ and use LCU. Or equivalently, directly use QSVT for polynomial eigenvalue transformation with arbitrary parity. We have a $(1, m+2, \epsilon/6)$ of $e^{itH}/2$.
- Last step: We want to turn $e^{itH}/2$ to e^{itH}.

[4] [Berry-Childs-Cleve-Kothari-Somma 2014/2015]

Optimal Hamiltonian Simulation by QSVT cont'd

Given U_H: an $(\alpha, m, 0)$-block-encoding of H.
Goal: an algorithm that makes $\mathcal{O}(t + \log(1/\epsilon))$ queries to U_H.

- We now have block-encodings of both $\cos(tH)$ and $i\sin(tH)$ and use LCU. Or equivalently, directly use QSVT for polynomial eigenvalue transformation with arbitrary parity. We have a $(1, m+2, \epsilon/6)$ of $e^{itH}/2$.
- Last step: We want to turn $e^{itH}/2$ to e^{itH}. Notice this can be done by QSVT with $P(x) = 3x - 4x^3$.

[4] [Berry-Childs-Cleve-Kothari-Somma 2014/2015]

Optimal Hamiltonian Simulation by QSVT cont'd

Given U_H: an $(\alpha, m, 0)$-block-encoding of H.
Goal: an algorithm that makes $\mathcal{O}(t + \log(1/\epsilon))$ queries to U_H.

- We now have block-encodings of both $\cos(tH)$ and $i\sin(tH)$ and use LCU. Or equivalently, directly use QSVT for polynomial eigenvalue transformation with arbitrary parity. We have a $(1, m+2, \epsilon/6)$ of $e^{itH}/2$.
- Last step: We want to turn $e^{itH}/2$ to e^{itH}. Notice this can be done by QSVT with $P(x) = 3x - 4x^3$.

  Oblivious Amplitude Amplification (OAA) [4]

  Given a block-encoding U_A of A (and A is close to unitary): $A = \alpha(\langle 0|_m \otimes I_n)U_A(|0\rangle_s \otimes I_n)$. Define $R := I_m - 2\,|0\rangle_m \langle 0|_m$ and

$$W = -U_A(R \otimes I_n)U_A^\dagger(R \otimes I_n)U_A.$$

$$(\langle 0|_m \otimes I_n)W(|0\rangle_m \otimes I_n) = \frac{3}{\alpha}A - \frac{4}{\alpha^3}AA^\dagger A \approx (\frac{3}{\alpha} - \frac{4}{\alpha^3})A.$$

[4] [Berry-Childs-Cleve-Kothari-Somma 2014/2015]

Summary of Hamiltonian Simulation

- Hamiltonian simulation: motivation; set-up
- Expected cost: No-fast-forwarding theorem and BQP-hardness
- Algorithms
  - Trotterization
  - Revisit of Block-encoding; truncated Taylor series
  - Optimal Hamiltonian Simulation via QSVT

Summary of time-independent Ham Sim cont'd

- Trotterization:
  1st-order Trotter formula

$$e^{-\mathrm{i}Ht} = \left(e^{-\mathrm{i}H_1 t/L} e^{-\mathrm{i}H_2 t/L}\right)^L + \mathcal{O}(\|[H_1, H_2]\| t^2/L)$$

  High order (p-th): $\mathcal{O}(\|\text{Comm}\|^{1/p} \frac{t^{1+1/p}}{\epsilon^{1/p}})$
  Randomized product formula, e.g., qDRIFT: $\mathcal{O}\left(\alpha^2 t^2/\epsilon\right)$.
  (*weak convergence* wrt the diamond norm of Quantum channels)

- LCU, e.g. Truncated Taylor series:

$$\mathcal{O}\left(\alpha t \frac{\log(t/\epsilon)}{\log\log(t/\epsilon)}\right).$$

- QSP/QSVT: $\mathcal{O}(\alpha t + \frac{\log(1/\epsilon)}{\log(e + \log(1/\epsilon)/(\alpha t))})$

Helpful References

- Lecture notes on Quantum Algorithms for Scientific Computations by Lin Lin (UC Berkeley) [arXiv:2201.08309]
- Lecture notes on Quantum Algorithms by Andrew Childs (U Maryland)
- A. Gilyen, Y. Su, G.H. Low, N. Wiebe. Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics, [arXiv 1806.01838]

Thank you for your attention!