# CS 59300 – Algorithms for Data Science

## Classical and Quantum approaches

### Lecture 16 (11/4)

### Quantum linear algebra toolkits (I)

https://ruizhezhang.com/course_fall_2025.html

The slides are borrowed from Dong An's talk at IPAM

# Quantum linear algebra toolbox

- Basic linear algebra operations
  - Input models for vectors and matrices
  - Matrix-vector multiplication
  - Matrix/vector addition: linear combination of unitaries (LCU)
  - Matrix multiplication
- Linear systems of equations
- Eigenvalue problems (revisted)
- Matrix functions
  - Functions of Hermitian matrices: quantum signal processing (QSP), qubitization
  - Functions of general matrices: quantum singular value transformation (QSVT), linear combinations of Hamiltonian simulations (LCHS)

# Input model: vectors

Vector as quantum state:

$$O_u : |0\rangle \mapsto \sum_{j=0}^{2^n-1} u_j |j\rangle \qquad \text{(state preparation oracle)}$$

- Constructing $\mathcal{O}_u$ is generally hard, but easy in special cases (Grover-Rudolph ’02; Zhang-Li-Yuan ’22)

# Input model: matrices

Matrix as quantum gate (block-encoding):

Let $A$ be a $2^n$-by-$2^n$ matrix. A block-encoding of $A$ is a $2^{n+a}$-by- $2^{n+a}$ unitary $U_A$ such that

$$A \approx \alpha(\langle 0^a| \otimes I)U_A(|0^a\rangle \otimes I)$$

Or equivalently,

$$U_A \approx \begin{bmatrix} A/\alpha & * \\ * & * \end{bmatrix}$$

- $\alpha$ is called the block-encoding factor and should satisfy $\alpha \geq ||A||$
- $U_A$ is called an $(\alpha, a, \epsilon)$-block-encoding of $A$
- Constructing $U_A$ is generally hard, but easy in special cases such as unitaries, sparse matrices, special structured matrices (Gilyen et al. '18; Camps et al. '22)

# Matrix-vector multiplication

**Inputs:**

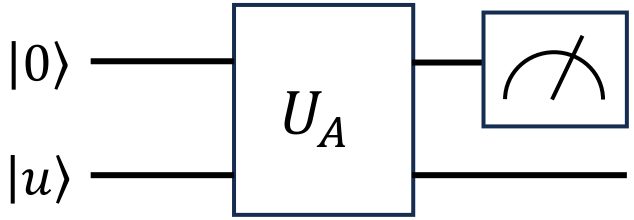
- $U_A$: an $(\alpha, a, \epsilon)$-block-encoding for $A \in \mathbb{C}^{2^n \times 2^n}$ i.e.

$$U_A = |0\rangle\langle 0| \otimes \frac{A}{\alpha} + |0\rangle\langle 1| \otimes * + |1\rangle\langle 0| \otimes * + |1\rangle\langle 1| \otimes *$$

- $U_b$: state preparation oracle for $u \in \mathbb{C}^{2^n}$ i.e.

$$U_b|0\rangle = |u\rangle = \sum_{j=0}^{2^n-1} u_j |j\rangle$$

**Algorithm:** 'apply the block-encoding'



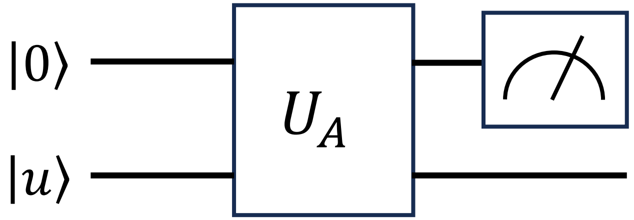

$$U_A|0\rangle|u\rangle \approx |0\rangle \otimes \frac{A}{\alpha}|u\rangle + |1\rangle \otimes *$$

$$\begin{bmatrix} A/\alpha & * \\ * & * \end{bmatrix} \begin{bmatrix} u \\ 0 \end{bmatrix} = \begin{bmatrix} Au/\alpha \\ * \end{bmatrix}$$

It succeeds if the measurement outcome equals to $0^a$

# Matrix-vector multiplication

**Algorithm:** ‘apply the block-encoding’




$\frac{A|u\rangle}{\||A|u\rangle\||}$ (upon measure 0)

$$U_A|0\rangle|u\rangle \approx |0\rangle \otimes \frac{A}{\alpha}|u\rangle + |1\rangle \otimes *$$

$$\begin{bmatrix} A/\alpha & * \\ * & * \end{bmatrix} \begin{bmatrix} u \\ 0 \end{bmatrix} = \begin{bmatrix} Au/\alpha \\ * \end{bmatrix}$$

- The success probability $= (\||A|u\rangle\||/\alpha)^2$
- The number of repeats: $\mathcal{O}\big((\alpha/\||A|u\rangle\||)^2\big)$ or $\mathcal{O}(\alpha/\||A|u\rangle\||)$ (by amplitude amplification)

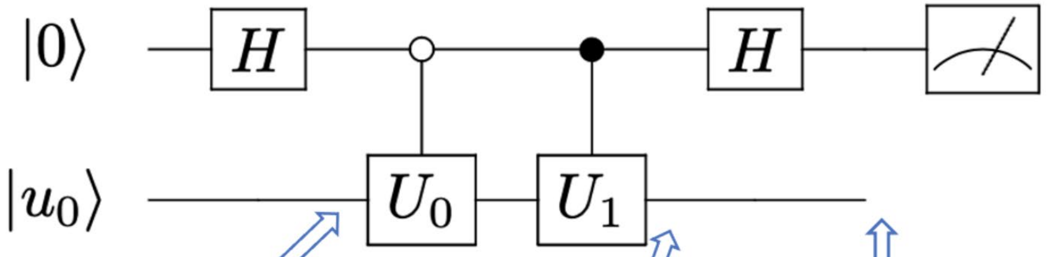
# Matrix addition: LCU

Linear combinations of unitaries (LCU): given a set of unitaries $\{U_j\}$ and coefficients $\{c_j\}$, compute

$$\sum_j c_j U_j$$

- Toy example:

$$\frac{1}{2}U_0 + \frac{1}{2}U_1$$




$$\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\,|u_0\rangle$$

$$\frac{1}{\sqrt{2}}(|0\rangle\, U_0\,|u_0\rangle + |1\rangle\, U_1\,|u_0\rangle)$$

$$\frac{1}{2}\,|0\rangle\,(U_0 + U_1)\,|u_0\rangle + \frac{1}{2}\,|1\rangle\,(U_0 - U_1)\,|u_0\rangle)$$

# Matrix addition: LCU

Linear combinations of unitaries (LCU): given a set of unitaries $\{U_j\}$ and coefficients $\{c_j\}$, compute

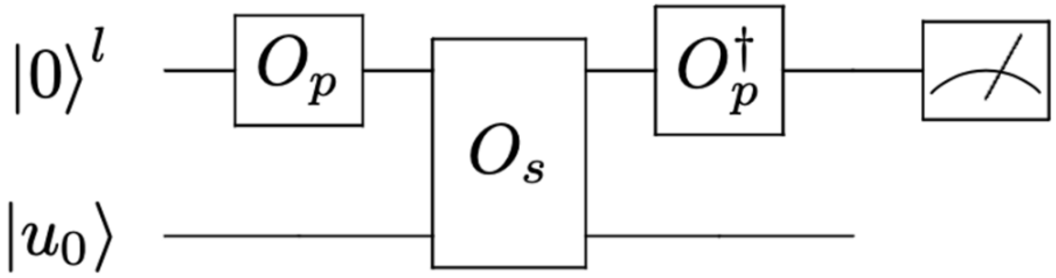
$$\sum_j c_j U_j$$

$$|0\rangle|u_0\rangle \xrightarrow{O_p} \frac{1}{\sqrt{\|c\|_1}} \sum_j \sqrt{c_j}|j\rangle\,|u_0\rangle$$

$$\xrightarrow{O_s} \frac{1}{\sqrt{\|c\|_1}} \sum_j \sqrt{c_j}|j\rangle\, U_j|u_0\rangle$$

$$\xrightarrow{O_p^\dagger} |0\rangle \sum_j \frac{c_j}{\|c\|_1} U_j|u_0\rangle + |\perp\rangle$$

First register $\neq 0$




$$\text{Prepare Oracle } O_p : |0\rangle \to \frac{1}{\sqrt{\|c\|_1}} \sum_j \sqrt{c_j}\,|j\rangle$$

$$\text{Select Oracle } O_s = \sum_j |j\rangle\langle j| \otimes U_j$$

# Interlude: Uncompute

$$\frac{1}{\sqrt{\|c\|_1}}\sum_j \sqrt{c_j}|j\rangle\, U_j|u_0\rangle \xrightarrow{O_p^\dagger} |0\rangle \sum_j \frac{c_j}{\|c\|_1} U_j|u_0\rangle + |\perp\rangle$$

- By definition,

$$O_p^\dagger\colon \frac{1}{\sqrt{\|c\|_1}}\sum_j \sqrt{c_j}|j\rangle \mapsto |0\rangle$$

- $O_p^\dagger$ is a unitary matrix, so for any $|v\rangle \perp \frac{1}{\sqrt{\|c\|_1}}\sum_j \sqrt{c_j}|j\rangle$, $|v\rangle \xrightarrow{O_p^\dagger} |\perp\rangle$

- Thus, for each $j$, we can decompose $\frac{\sqrt{c_j}}{\sqrt{\|c\|_1}}|j\rangle$ into the parallel part and the orthogonal part:

$$\left\langle \frac{\sqrt{c_j}}{\sqrt{\|c\|_1}}|j\rangle, \frac{1}{\sqrt{\|c\|_1}}\sum_j \sqrt{c_j}|j\rangle \right\rangle = \frac{c_j}{\|c\|_1}$$

# Interlude: Uncompute

$$\frac{1}{\sqrt{\|c\|_1}} \sum_j \sqrt{c_j} |j\rangle \, U_j |u_0\rangle \xrightarrow{O_p^\dagger} |0\rangle \sum_j \frac{c_j}{\|c\|_1} U_j |u_0\rangle + |\perp\rangle$$

- By definition,

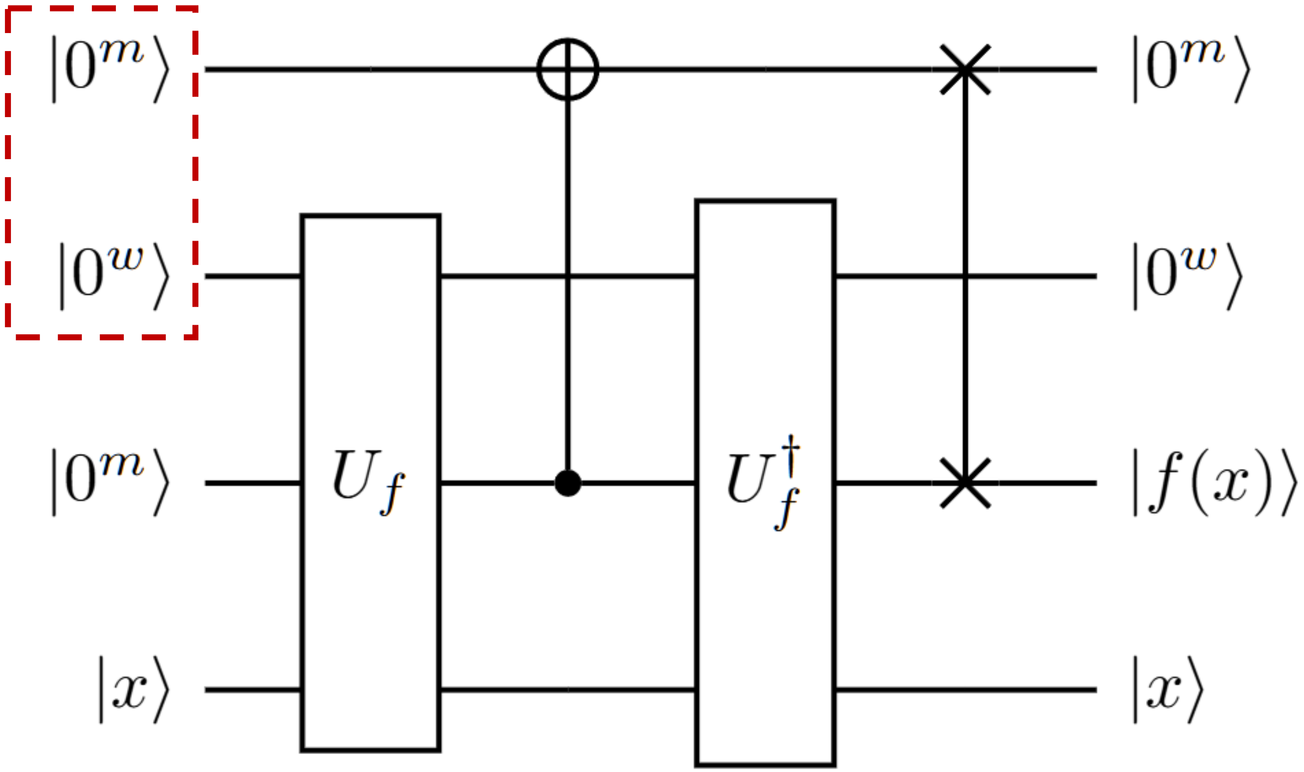
$$O_p^\dagger \colon \frac{1}{\sqrt{\|c\|_1}} \sum_j \sqrt{c_j} |j\rangle \mapsto |0\rangle$$

- $O_p^\dagger$ is a unitary matrix, so for any $|v\rangle \perp \frac{1}{\sqrt{\|c\|_1}} \sum_j \sqrt{c_j} |j\rangle$, $|v\rangle \xrightarrow{O_p^\dagger} |\perp\rangle$
- Thus, for each $j$, we can decompose $\frac{\sqrt{c_j}}{\sqrt{\|c\|_1}} |j\rangle$ into the parallel part and the orthogonal part:

$$\frac{\sqrt{c_j}}{\sqrt{\|c\|_1}} |j\rangle \xrightarrow{O_p^\dagger} \frac{c_j}{\|c\|_1} |0\rangle$$

# Interlude: Uncompute

ancilla registers




$$|0^{m+w}\rangle|0^m\rangle|x\rangle \mapsto |0^{m+w}\rangle|f(x)\rangle|x\rangle$$

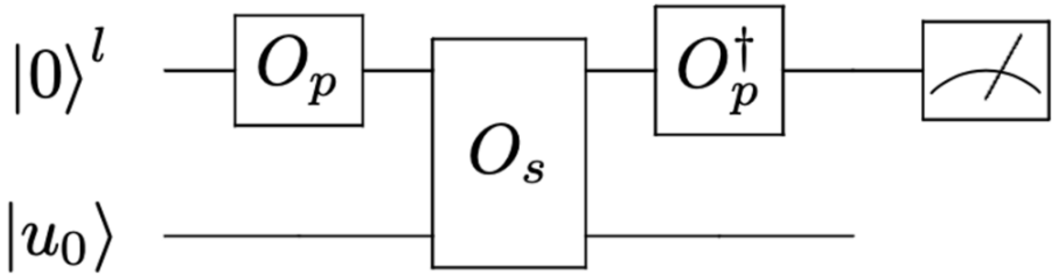
See Sec. 1.7 in *Lecture Notes on Quantum Algorithms for Scientific Computation (QASC)*

# Matrix addition: LCU

Linear combinations of unitaries (LCU): given a set of unitaries $\{U_j\}$ and coefficients $\{c_j\}$, compute

$$\sum_j c_j U_j$$

$$|0\rangle|u_0\rangle \xrightarrow{O_p^\dagger} |0\rangle \sum_j \frac{c_j}{\|c\|_1} U_j |u_0\rangle + |\perp\rangle$$




- Cost: $\mathcal{O}\left(\|c\|_1 / \left\|\sum_j c_j U_j |u_0\rangle\right\|\right)$

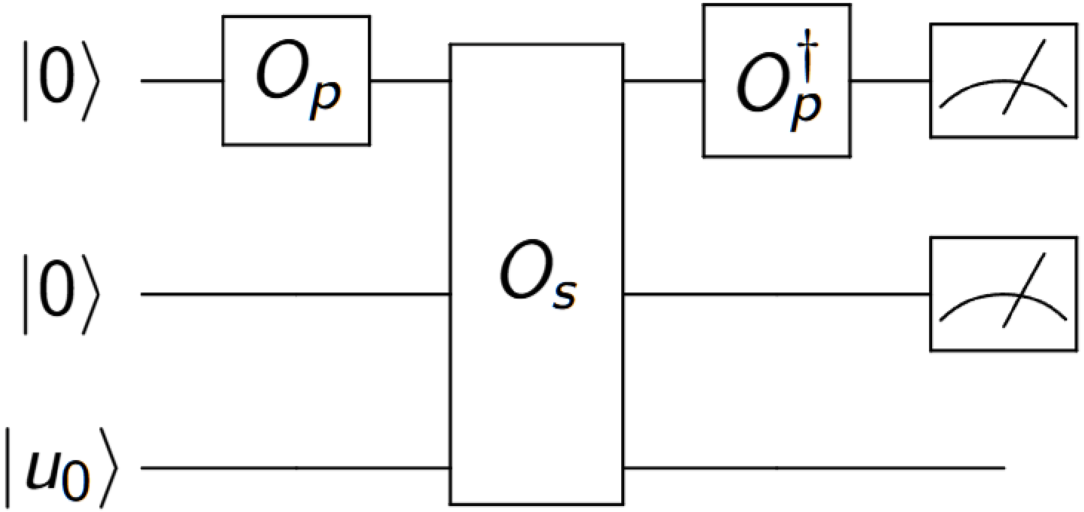
Prepare Oracle $O_p : |0\rangle \to \frac{1}{\sqrt{\|c\|_1}} \sum_j \sqrt{c_j}\,|j\rangle$

Select Oracle $O_s = \sum_j |j\rangle\langle j| \otimes U_j$

# Matrix addition

Given a set of block-encodings for general matrices $\{A_j\}$ with $\|A_j\| \leq 1$ and coefficients $\{c_j\}$, compute the block-encoding for $\sum_j c_j A_j$

LCU for block-encodings:



# Vector addition

Given a set of vectors $\{u_j\}$ and coefficients $\{c_j\}$, compute $\sum_j c_j u_j$

- Construct the state preparation oracles $O_{u_j}$:

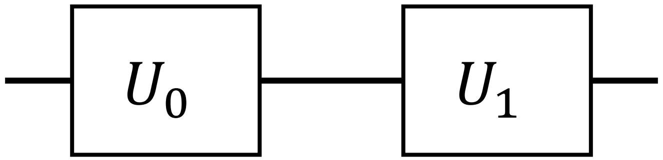
$$O_{u_j} : |0\rangle \mapsto |u_j\rangle$$

- Apply LCU to $\{O_{u_j}\}$ and $\{c_j\}$:

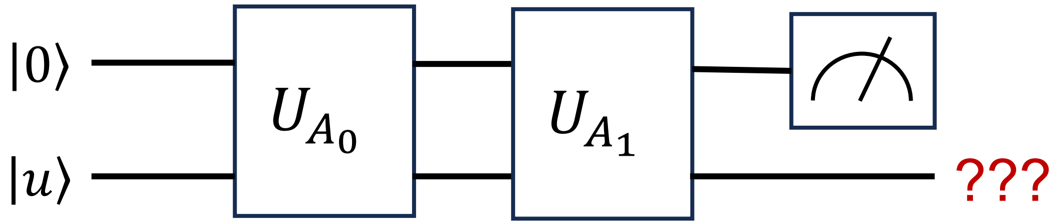
$$\sum_j c_j O_{u_j} |0\rangle = \sum_j c_j |u_j\rangle$$

# Matrix multiplication

- Example 1: two unitaries $U_0 U_1$:

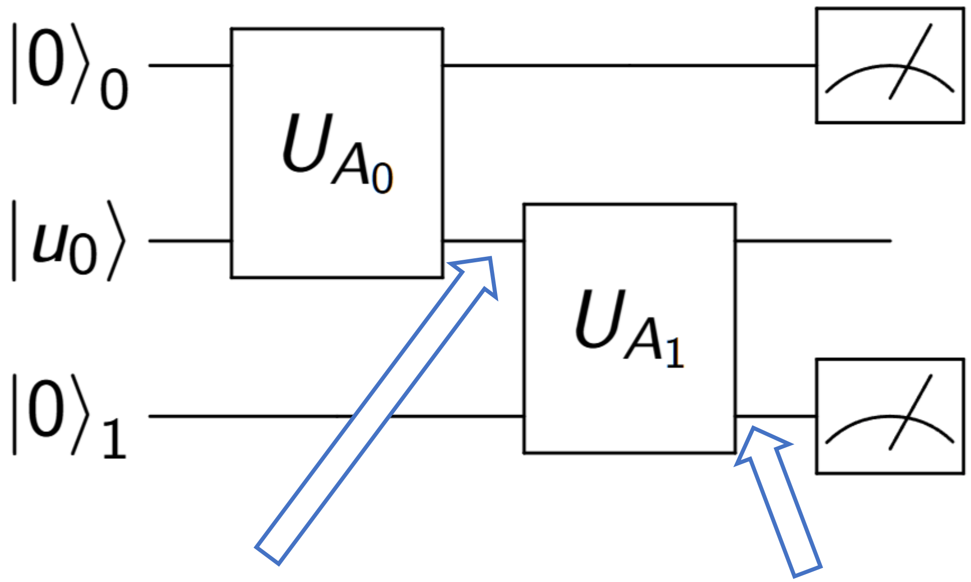



- Example 2: two matrices $A_0 A_1$ (with block-encodings):




$$\begin{bmatrix} A_1 & * \\ * & * \end{bmatrix} \begin{bmatrix} A_0 & * \\ * & * \end{bmatrix} \begin{bmatrix} u \\ 0 \end{bmatrix} = \begin{bmatrix} A_1 & * \\ * & * \end{bmatrix} \begin{bmatrix} A_0 u \\ * \end{bmatrix} = \begin{bmatrix} A_1 A_0 u + * \\ * \end{bmatrix} \neq \begin{bmatrix} A_1 A_0 u \\ * \end{bmatrix}$$

# Matrix multiplication



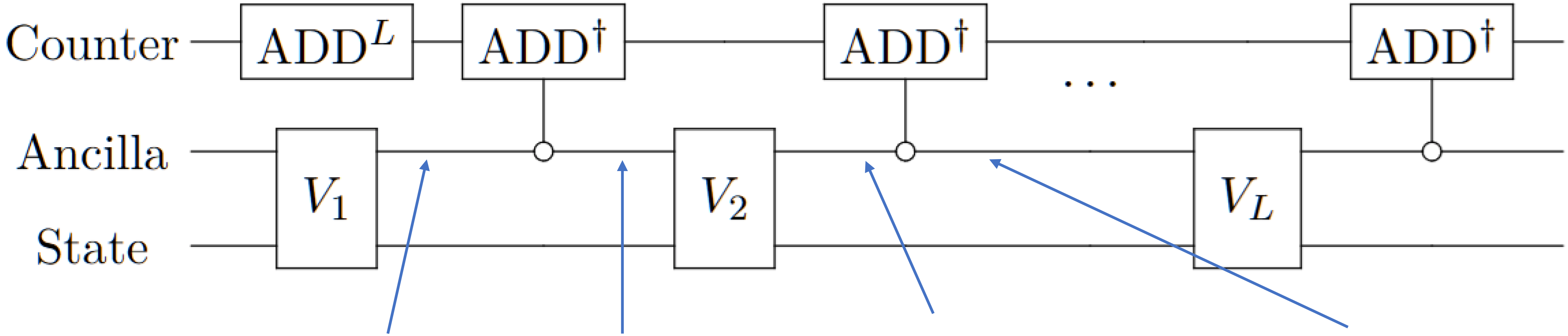

$(|0\rangle_0 A_0|u_0\rangle + |\perp\rangle_0|*\rangle)|0\rangle_1$

$|0\rangle_0(A_1A_0|u_0\rangle|0\rangle_1 + |*\rangle|\perp\rangle_1) + |\perp\rangle_0|*\rangle|*\rangle_1$
$= |00\rangle_{01}A_1A_2|u_0\rangle + |\perp\rangle_{01}|*\rangle$

Drawback: multiplication of $L$ matrices requires $\mathcal{O}(L)$ ancilla qubits

# Matrix multiplication: qubit-efficient approach

Compression gadget (Low-Wiebe '18; Fang-Lin-Tong '22): $\mathcal{O}(\log L)$ extra ancilla qubits




$|L\rangle_c|0\rangle_a A_1|u\rangle + |L\rangle_c|\perp\rangle_a|*\rangle$

$|L-1\rangle_c|0\rangle_a A_1|u\rangle$
$+ |L\rangle_c|\perp\rangle_a|*\rangle$

$|L-1\rangle_c|0\rangle_a A_2 A_1|u\rangle$
$+ |L\rangle|*\rangle_a|*\rangle$
$+ |L-1\rangle_c|\perp\rangle_a|*\rangle$

$|L-2\rangle_c|0\rangle_a A_2 A_1|u\rangle$
$+ |L-1\rangle_c|0\rangle_a|*\rangle$
$+ |L\rangle_c|\perp\rangle_a|*\rangle$

Final state: $|0\rangle_c|0\rangle_a A_L \cdots A_1|u\rangle + \sum_{j>0}|j\rangle_c|*\rangle_a|*\rangle$

$\mathrm{ADD}: |0\rangle \to |1\rangle \to |2\rangle \to \cdots \to |L\rangle \to |0\rangle$

# Quantum linear algebra toolbox

- Basic linear algebra operations
  - Input models for vectors and matrices
  - Matrix-vector multiplication
  - Matrix/vector addition: linear combination of unitaries (LCU)
  - Matrix multiplication
- Linear systems of equations

# Quantum linear system problem (QLSP)

Classical version: given an $N$-by-$N$ Hermitian matrix $A$, and a vector $b$, compute

$$x = A^{-1}b$$

- For non-Hermitian $A$, consider dilation: $\begin{bmatrix} 0 & A \\ A^{\dagger} & 0 \end{bmatrix} \begin{bmatrix} 0 \\ x \end{bmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix}$

Quantum version: given a block-encoding for $A$ (assuming $\|A\| = 1$) and a state preparation oracle for $|b\rangle$, compute an $\epsilon$-approximation of the quantum state:

$$\| |\tilde{x}\rangle - |x\rangle \| \leq \epsilon \qquad |x\rangle = \frac{A^{-1}|b\rangle}{\|A^{-1}|b\rangle\|}$$

Parameters:

- Condition number: $\kappa \coloneqq \|A\| \|A^{-1}\| = \|A^{-1}\|$
- Accuracy $\epsilon$

# Harrow-Hassidim-Lloyd (HHL)

QLSP as an eigenvalue transformation problem:

- Consider the eigen-decomposition: $A = \sum_j \lambda_j |v_j\rangle\langle v_j|$
- $|b\rangle = \sum_j \beta_j |v_j\rangle$

$$A^{-1}|b\rangle = \left(\sum_j \lambda_j^{-1} |v_j\rangle\langle v_j|\right)\left(\sum_j \beta_j |v_j\rangle\right) = \sum_j \frac{\beta_j}{\lambda_j} |v_j\rangle$$

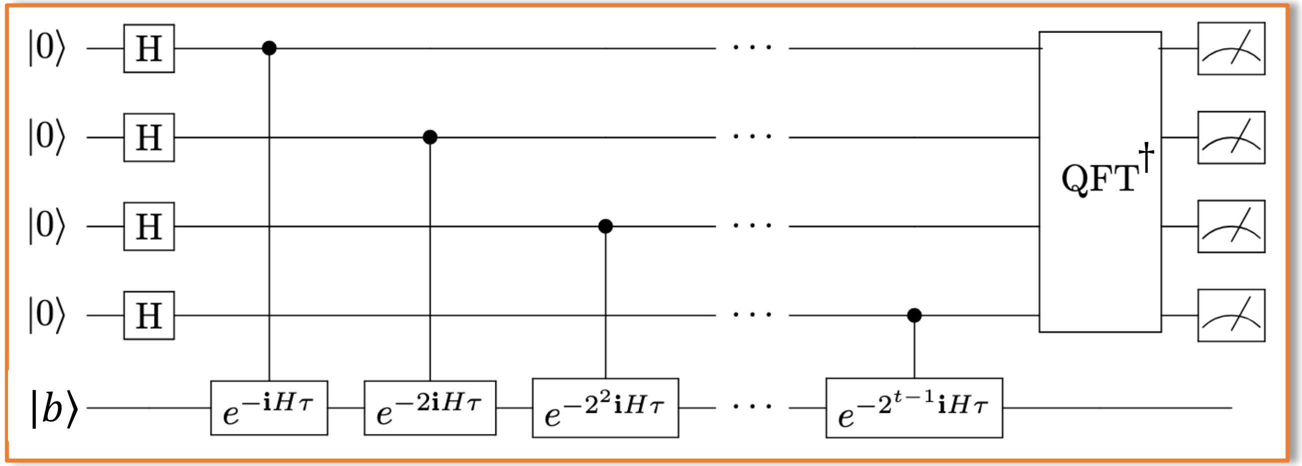
Need to do:

1. Store the information (binary encoding) of $\lambda_j$'s in an ancilla register coherently $\longleftarrow$ QPE
2. Multiply the factor $\lambda_j^{-1}$ to each eigenvector $|v_j\rangle$

# Harrow-Hassidim-Lloyd (HHL)

Need to do:

1. Store the information (binary encoding) of $\lambda_j$'s in an ancilla register coherently




$$U_{\mathrm{QPE}}|0\rangle|b\rangle = \sum_j \beta_j |\tilde{\lambda}_j\rangle |v_j\rangle$$

# Harrow-Hassidim-Lloyd (HHL)

Need to do:

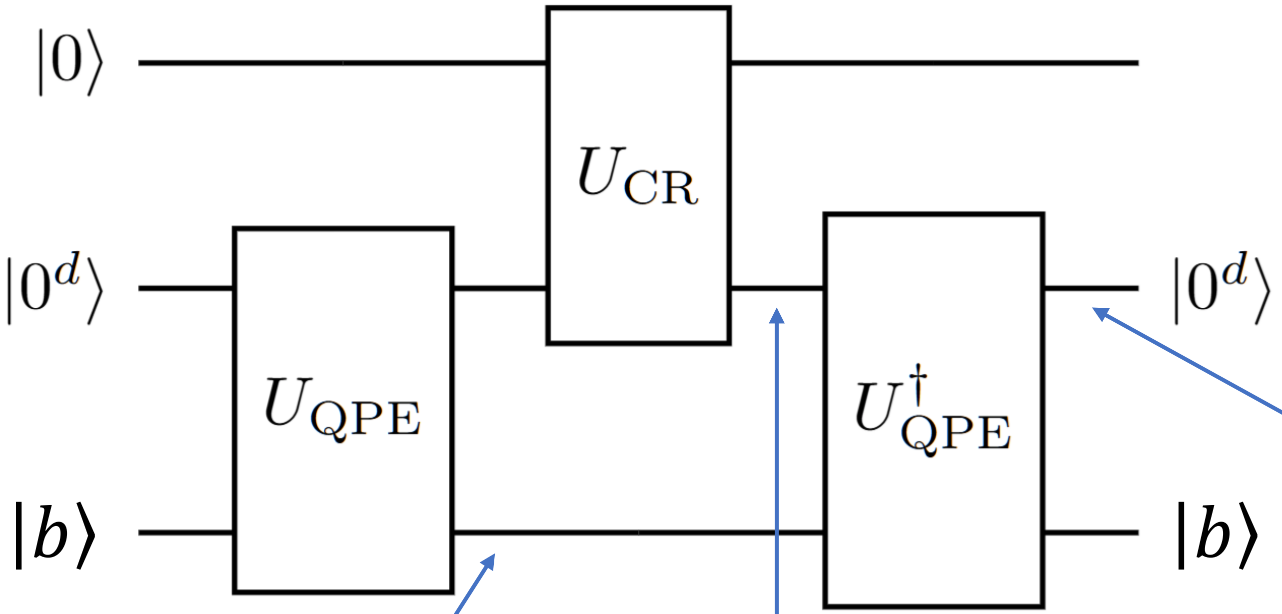
1. Store the information (binary encoding) of $\lambda_j$'s in an ancilla register coherently
2. Multiply the factor $\lambda_j^{-1}$ to each eigenvector $|v_j\rangle$

Controlled rotation:

$$U_{\mathrm{CR}}|0\rangle|\lambda_j\rangle = \left(\frac{C}{\lambda_j}|0\rangle + \sqrt{1 - \frac{C^2}{\lambda_j^2}}|1\rangle\right)|\lambda_j\rangle$$

- See Sec. 4.3.2 of *QASC* if you want to know how to implement $U_{\mathrm{CR}}$

# Harrow-Hassidim-Lloyd (HHL)




$$\sum_j C\frac{\beta_j}{\lambda_j}|0\rangle|0\rangle|v_j\rangle + |1\rangle|0\rangle|*\rangle$$

$$|0\rangle\sum_j \beta_j|\lambda_j\rangle|v_j\rangle$$

$$\sum_j \beta_j\left(\frac{C}{\lambda_j}|0\rangle + \sqrt{1-\frac{C^2}{\lambda_j^2}}|1\rangle\right)|\lambda_j\rangle|v_j\rangle$$
$$= \sum_j C\frac{\beta_j}{\lambda_j}|0\rangle|\lambda_j\rangle|v_j\rangle + *\,|1\rangle|\lambda_j\rangle|v_j\rangle$$

# Harrow-Hassidim-Lloyd (HHL)

Final state of HHL:

$$|0\rangle \sum_j C \frac{\beta_j}{\lambda_j} |v_j\rangle + |\perp\rangle = |0\rangle C A^{-1} |b\rangle + |\perp\rangle$$

- We need to choose $C$ so that $\left|\frac{C}{\lambda_j}\right| \leq 1$, i.e., $C \leq |\lambda_0|$
- The success probability $= \| C A^{-1} |b\rangle \|^2 = C^2 \|x\|^2$, so we want $C$ to be large

$C \sim 1/\kappa$

$$\Pr[\text{succ}] = \| C A^{-1} |b\rangle \|^2 = \kappa^{-2} \cdot \| A^{-1} |b\rangle \|^2 \geq \kappa^{-2} \cdot \frac{\| |b\rangle \|^2}{\|A\|^2} = \kappa^{-2}$$

- Number of repeats: $\mathcal{O}(\kappa^{-1})$
- What is the cost of QPE?

# Harrow-Hassidim-Lloyd (HHL)

Due to the approximation errors from QPE, the actual final state is

$$|0\rangle \sum_j C \frac{\beta_j}{\tilde{\lambda}_j} |v_j\rangle + |\perp\rangle \approx |0\rangle C A^{-1} |b\rangle + |\perp\rangle$$

where $\tilde{\lambda}_j = \lambda_j (1 + e_j)$ with $|e_j| \leq \epsilon/4$

- For the unnormalized solution,

$$\|\tilde{x} - x\| = \left\| \sum_j \beta_j \left( \frac{1}{\tilde{\lambda}_j} - \frac{1}{\lambda_j} \right) |v_j\rangle \right\| \leq \left\| \sum_j \frac{\beta_j}{\lambda_j} \left( \frac{-e_j}{1 + e_j} \right) |v_j\rangle \right\| = \mathcal{O}(\epsilon) \|x\|$$

- For the normalized solution state,

$$\| |\tilde{x}\rangle - |x\rangle \| = \left\| \frac{\tilde{x}}{\|\tilde{x}\|} - \frac{x}{\|x\|} \right\| \leq \left| 1 - \frac{\|\tilde{x}\|}{\|x\|} \right| + \frac{\|\tilde{x} - x\|}{\|x\|} \leq \epsilon$$

# Harrow-Hassidim-Lloyd (HHL)

Due to the approximation errors from QPE, the actual final state is

$$|0\rangle \sum_j C \frac{\beta_j}{\widetilde{\lambda}_j} |v_j\rangle + |\perp\rangle \approx |0\rangle C A^{-1} |b\rangle + |\perp\rangle$$

where $\widetilde{\lambda}_j = \lambda_j (1 + e_j)$ with $|e_j| \leq \epsilon/4$

- Thus, QPE requires an addition error to within $\epsilon_{\mathrm{QPE}} = \mathcal{O}(\lambda_0 \epsilon) \leq \epsilon/\kappa$
- The cost of $U_{\mathrm{QPE}}$ is $\mathcal{O}(\kappa/\epsilon)$
- Overall complexity of HHL: $\mathcal{O}(\kappa^2/\epsilon)$
- Caveat: QPE requires Hamiltonian simulation $e^{-\mathbf{i}\tau A}$ and the $1/\epsilon$ in the complexity

# QLSP via LCU (Fourier approach)

- Key identity:

$$\frac{1}{x} = \frac{\mathbf{i}}{\sqrt{2\pi}} \int_0^\infty \mathrm{d}y \underbrace{\int_{-\infty}^\infty z e^{-z^2/2} e^{-\mathbf{i}xyz} \mathrm{d}z}_{-\sqrt{2\pi}\mathbf{i}te^{-t^2/2}\big|_{t=xy}}$$

$$\forall \text{ odd } f(y) \text{ s.t. } \int_0^\infty f(y)\mathrm{d}y = 1, \qquad \int_0^\infty f(xy)\mathrm{d}y = \frac{1}{x}$$

# QLSP via LCU (Fourier approach)

- Key identity:

$$\frac{1}{x} = \frac{\mathbf{i}}{\sqrt{2\pi}} \int_0^{\infty} \mathrm{d}y \int_{-\infty}^{\infty} z e^{-z^2/2} e^{-\mathbf{i}xyz} \mathrm{d}z$$

- It also holds for matrix:

$$\begin{aligned} A^{-1} &= \frac{\mathbf{i}}{\sqrt{2\pi}} \int_0^{\infty} \mathrm{d}y \int_{-\infty}^{\infty} z e^{-z^2/2} e^{-\mathbf{i}yzA} \mathrm{d}z \\ &\approx \frac{\mathbf{i}}{\sqrt{2\pi}} \int_0^{Y} \mathrm{d}y \int_{-Z}^{Z} z e^{-z^2/2} e^{-\mathbf{i}yzA} \mathrm{d}z \\ &\approx \frac{\mathbf{i}}{\sqrt{2\pi}} \sum_{j=0}^{J-1} \Delta_y \sum_{k=-K}^{K} \Delta_z z_k e^{-z_k^2/2} e^{-\mathbf{i}y_j z_k A} \end{aligned}$$

$$Y = \mathcal{O}\left(\kappa\sqrt{\log(\kappa/\epsilon)}\right),$$
$$Z = \mathcal{O}\left(\sqrt{\log(\kappa/\epsilon)}\right)$$

$$\text{LCU cost} \approx \|c\|_1 \approx \int_0^Y \mathrm{d}y \int_0^Z z e^{-z^2/2} \mathrm{d}z = \mathcal{O}\left(\kappa\sqrt{\log(\kappa/\epsilon)}\right)$$

# QLSP via LCU (Fourier approach)

- It also holds for matrix:

$$A^{-1} \approx \frac{\mathbf{i}}{\sqrt{2\pi}} \sum_{j=0}^{J-1} \Delta_y \sum_{k=-K}^{K} \Delta_z z_k e^{-z_k^2/2} e^{-\mathbf{i} y_j z_k A}$$

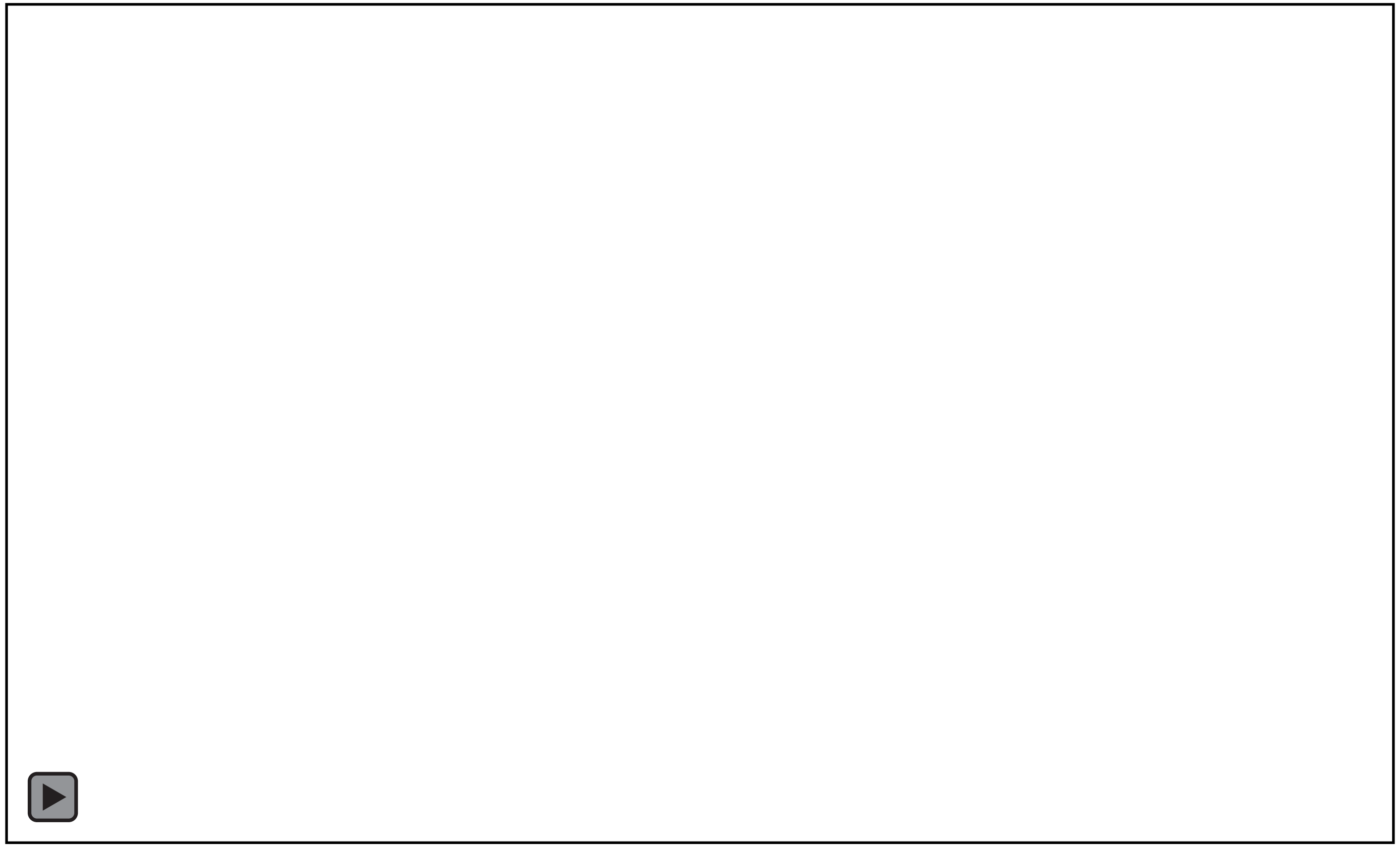
$$Y = \mathcal{O}\left(\kappa\sqrt{\log(\kappa/\epsilon)}\right),$$
$$Z = \mathcal{O}\left(\sqrt{\log(\kappa/\epsilon)}\right)$$

- LCU cost: $\mathcal{O}\left(\kappa\sqrt{\log(\kappa/\epsilon)}\right)$
- Hamiltonian simulation $e^{-\mathbf{i}AT}$ cost: $T\text{polylog}(1/\epsilon) = \kappa\text{polylog}(\kappa/\epsilon)$
- Overall complexity:

$$\kappa\sqrt{\log(\kappa/\epsilon)} \times \kappa\text{polylog}(\kappa/\epsilon) = \kappa^2\text{polylog}(\kappa/\epsilon)$$

# QLSP via LCU (Chebyshev approach)

$$\frac{1}{x} \approx \frac{1-(1-x^2)^d}{x} \qquad \forall x \in [-1,-\kappa] \cup [\kappa,1] \qquad d \sim \kappa^2 \log(\kappa/\epsilon)$$



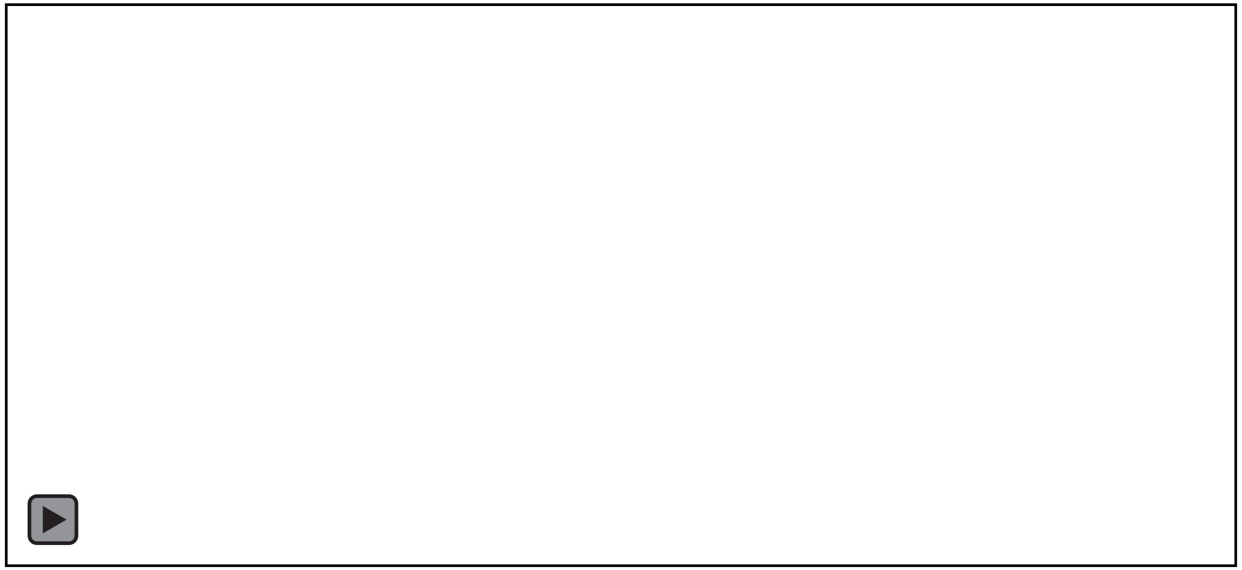
# QLSP via LCU (Chebyshev approach)

$$\frac{1}{x} \approx \frac{1-(1-x^2)^d}{x} \qquad \forall x \in [-1,-\kappa] \cup [\kappa, 1] \qquad d \sim \kappa^2 \log(\kappa/\epsilon)$$

Chebyshev polynomials

$$T_n(\cos\theta) = \cos(n\theta)$$

$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x), \qquad T_0(x) = 1, T_1(x) = x$$



# QLSP via LCU (Chebyshev approach)

$$\frac{1}{x} \approx \frac{1-(1-x^2)^d}{x} \qquad \forall x \in [-1,-\kappa] \cup [\kappa, 1] \qquad d \sim \kappa^2 \log(\kappa/\epsilon)$$

Chebyshev polynomials

$$T_n(\cos\theta) = \cos(n\theta)$$
$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x), \qquad T_0(x) = 1, T_1(x) = x$$

$$\frac{1-(1-x^2)^d}{x} = 4\sum_{j=0}^{d-1}(-1)^j \left(2^{-2d}\sum_{i=j+1}^{d}\binom{2d}{d+i}\right)T_{2j+1}(x)$$

$$\approx 4\sum_{j=0}^{J}(-1)^j \left(2^{-2d}\sum_{i=j+1}^{d}\binom{2d}{d+i}\right)T_{2j+1}(x) \qquad J \sim \sqrt{d\log(d/\epsilon)}$$

# QLSP via LCU (Chebyshev approach)

$$A^{-1} \approx 4 \sum_{j=0}^{\kappa \mathrm{polylog}(\kappa/\epsilon)} (-1)^j \left( 2^{-2d} \sum_{i=j+1}^{d} \binom{2d}{d+i} \right) T_{2j+1}(A)$$

- The LCU cost $\approx \|c\|_1 = \mathcal{O}\left(\sqrt{d}\right) = \mathcal{O}\left(\kappa \sqrt{\log(\kappa/\epsilon)}\right)$
- $\left\|T_{2j+1}(A)\right\| \leq 1$ but it is non-unitary, so we need to construct a block-encoding for it
  - Suppose it has a cost $\mathcal{O}(j) \leq \kappa \mathrm{polylog}(\kappa/\epsilon)$
- Overall complexity:

$$\kappa \sqrt{\log(\kappa/\epsilon)} \times \kappa \mathrm{polylog}(\kappa/\epsilon) = \kappa^2 \mathrm{polylog}(\kappa/\epsilon)$$

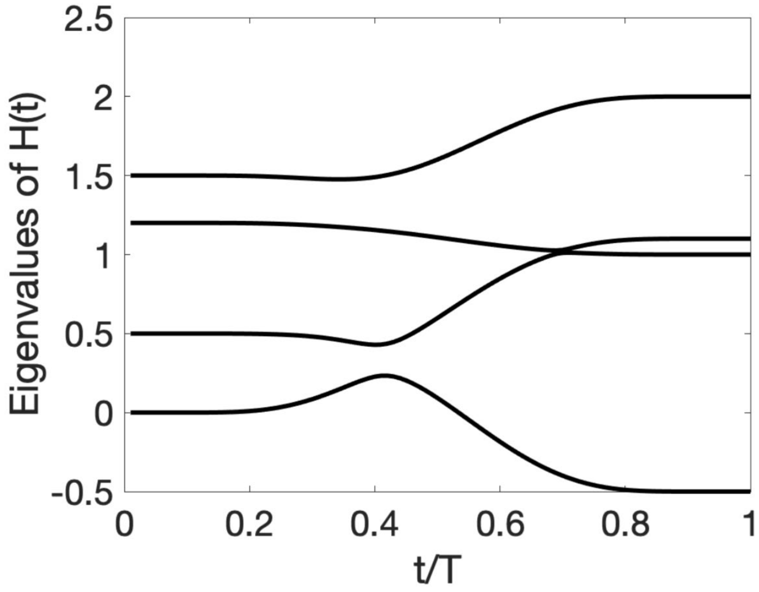
# QLSP via Adiabatic quantum computation (AQC)

Schrödinger equation:

$$i\partial_t|\psi(t)\rangle = H(t/T)|\psi(t)\rangle, \qquad t \in [0,T]$$
$$H|\psi(0)\rangle = \lambda|\psi_0\rangle$$

Adiabatic evolution:

- Start from an eigenstate of the initial Hamiltonian $H(0)$
- $H(t/T)$ is changing slowly (i.e. $T$ is large enough)
- Gap condition is satisfied
- The final state $|\psi(T)\rangle$ will approximate the eigenstate of $H(1)$



# QLSP via Adiabatic quantum computation (AQC)

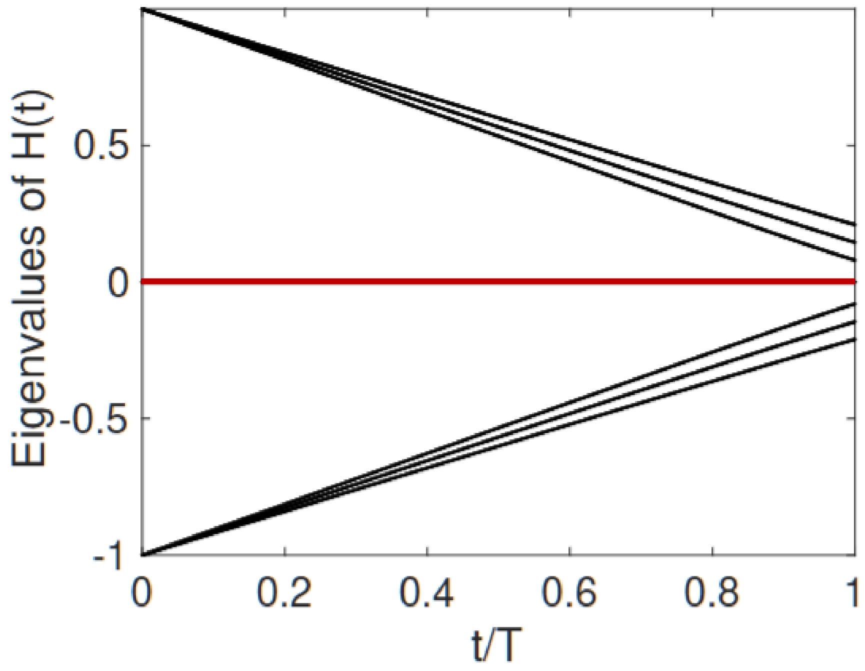
(Vanilla) AQC for QLSP:

$$H_0 = \begin{bmatrix} 0 & Q_b \\ Q_b & 0 \end{bmatrix}, \qquad H_1 = \begin{bmatrix} 0 & AQ_b \\ Q_b A & 0 \end{bmatrix}$$

$$Q_b := I - |b\rangle\langle b|$$

$$H(s) = (1-s)H_0 + sH_1$$

- $H_0$ has two eigenstates with eigenvalue 0: $\begin{bmatrix} b \\ 0 \end{bmatrix}$ and $\begin{bmatrix} 0 \\ b \end{bmatrix}$
- $H_1$ has two eigenstates with eigenvalue 0: $\begin{bmatrix} x \\ 0 \end{bmatrix}$ and $\begin{bmatrix} 0 \\ b \end{bmatrix}$
- If we start from $\begin{bmatrix} b \\ 0 \end{bmatrix}$, it will be evolved to $\begin{bmatrix} x \\ 0 \end{bmatrix}$
- QLSP $\Longrightarrow$ (Time-dependent) Hamiltonian simulation



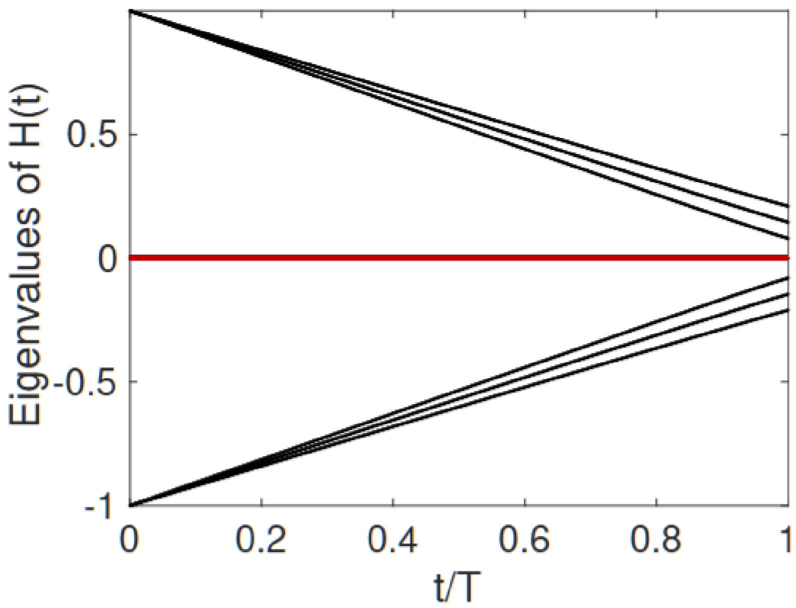
# QLSP via Adiabatic quantum computation (AQC)

**Quantum adiabatic theorem** (Jansen-Ruskai-Seile '06).

Assume gap $\Delta(s)$, then the distance between the dynamics and the eigenvector can be bounded by

$$\eta(s) = \mathcal{O}\left(\frac{\|H'(0)\|}{T\Delta(0)^2} + \frac{\|H'(s)\|}{T\Delta(s)^2} + \frac{1}{T}\int_0^s \left(\frac{\|H''(\tau)\|}{\Delta(\tau)^2} + \frac{\|H'(\tau)\|^2}{\Delta(\tau)^3}\right) d\tau\right)$$

- To bound the error by $\epsilon$, $T \sim \Delta_*^{-3}\epsilon^{-1}$
- In QLSP, $\Delta_* \sim \kappa^{-1}$
- Thus, $T \sim \kappa^3\epsilon^{-1}$



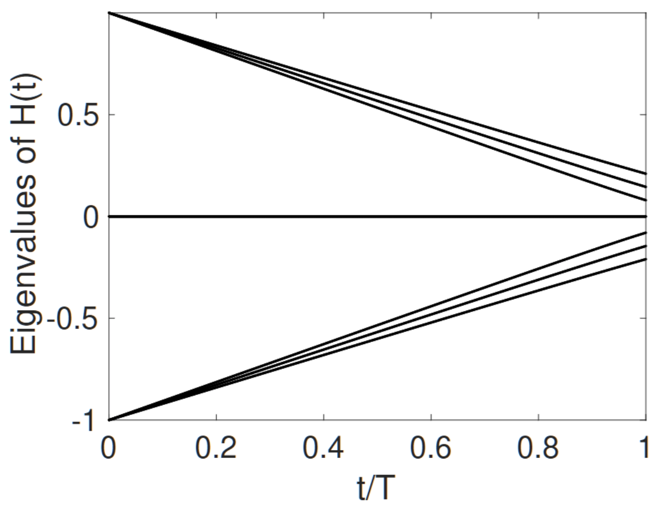
# Time-optimal AQC for QLSP

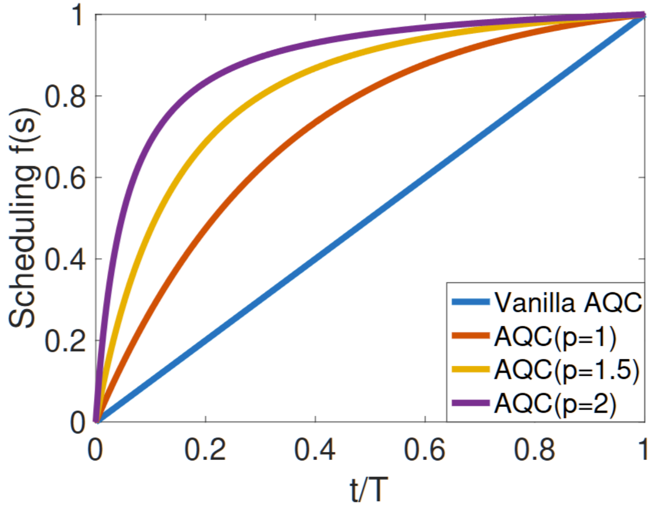
$$i\partial_t|\psi(t)\rangle = H(t/T)|\psi(t)\rangle, \qquad t \in [0,T]$$

- We can interpolate $H(s) = \big(1 - f(s)\big)H_0 + f(s)H_1$ with a clever design of $f(s)$ such that $\|H'(s)\|$ is small when the gap $\Delta(s)$ is small
- **An-Lin '19:**

AQC($p$): $$\frac{\mathrm{d}f(s)}{ds} = c\Delta\big(f(s)\big)^p \quad \Longrightarrow \quad f(s) = \frac{\kappa}{\kappa-1}\left(1 - \big(1 + s(\kappa^{p-1} - 1)\big)^{\frac{1}{1-p}}\right)$$

$$T = \mathcal{O}(\kappa/\epsilon)$$

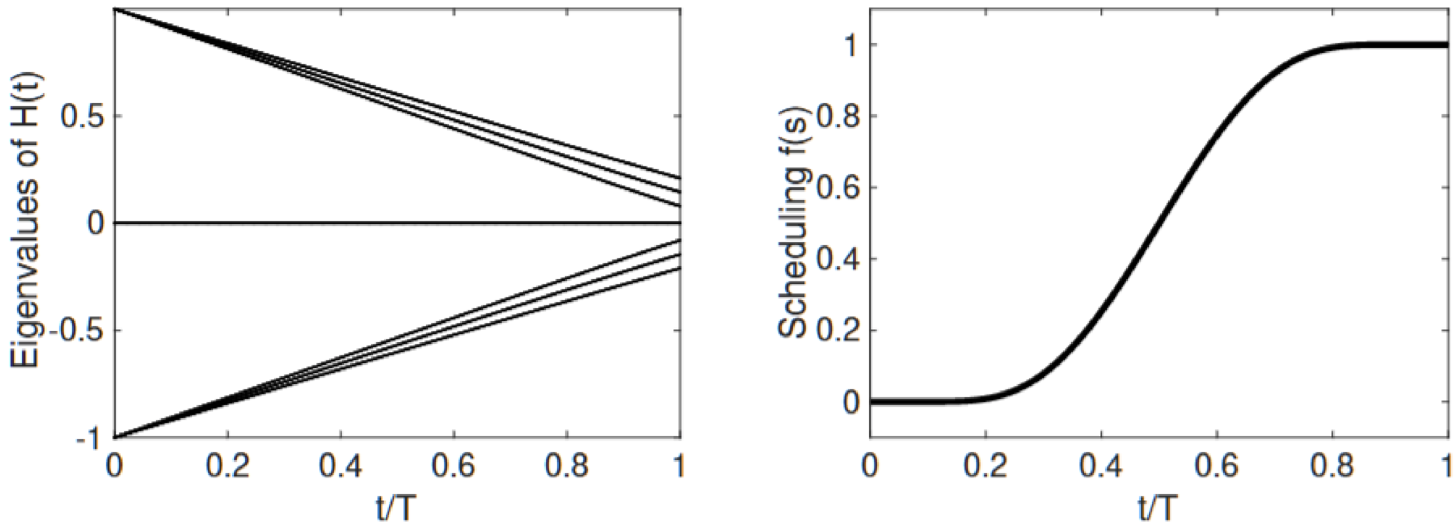


- Quantum Adiabatic Theorem gives a linear rate error decaying
- To improve the $\epsilon$-dependence, we need higher-order convergence, i.e. error $\sim T^{-k}$
- This is true if $f(s)$ satisfies the boundary cancellation condition: all derivatives of $H\big(f(s)\big)$ vanishes at the boundary $s = 0$ and $s = 1$
- **An-Lin '19:**

AQC(exp): $$f(s) = c' \int_0^s \exp\left(-\frac{1}{x(1-x)}\right) \mathrm{d}x$$

- $T = \kappa \mathrm{polylog}(\kappa/\epsilon)$
- Lower bound for QLSP:



# Quantum linear algebra toolbox

- Basic linear algebra operations
  - Input models for vectors and matrices
  - Matrix-vector multiplication
  - Matrix/vector addition: linear combination of unitaries (LCU)
  - Matrix multiplication
- Linear systems of equations
  - HHL: $\kappa^2/\epsilon$
  - LCU (Fourier and Chebyshev): $\kappa^2 \log(1/\epsilon)$ (Can be improved to $\kappa \log(1/\epsilon)$ via VTAA)
  - AQC: $\kappa \log(1/\epsilon)$ (optimal)
  - Recent survey: *arXiv:2411.02522v3*

# Input model: matrices

Matrix as quantum gate (block-encoding):

Let $A$ be a $2^n$-by-$2^n$ matrix. A block-encoding of $A$ is a $2^{n+a}$-by- $2^{n+a}$ unitary $U_A$ such that

$$A \approx \alpha(\langle 0^a| \otimes I)U_A(|0^a\rangle \otimes I)$$

Or equivalently,

$$U_A \approx \begin{bmatrix} A/\alpha & * \\ * & * \end{bmatrix}$$

- $\alpha$ is called the block-encoding factor and should satisfy $\alpha \geq \|A\|$
- $U_A$ is called an $(\alpha, a, \epsilon)$-block-encoding of $A$
- Constructing $U_A$ is generally hard, but easy in special cases such as unitaries, sparse matrices, special structured matrices (Gilyen et al. ’18; Camps et al. ’22)

# Construct block-encodings

Block-encoding always exists:

- Let $A$ be a general matrix with $\|A\| \leq 1$
- Consider the SVD $A = W\Sigma V^\dagger$, where $\Sigma$'s diagonal entries are in $[0,1]$
- We have a $(1,1,0)$-block-encoding for $A$:

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

# Construct block-encodings

Diagonal matrix:

- Let $A$ be a diagonal matrix given by the quantum oracle:

$$O_A : \ |0\rangle|i\rangle \mapsto \left( A_{ii}|0\rangle + \sqrt{1 - |A_{ii}^2|}|1\rangle \right)|i\rangle$$

- $U_A \coloneqq O_A$ is a $(1,1,0)$-block-encoding for $A$:

$$\langle 0|\langle j|U_A|0\rangle|i\rangle = A_{ii}\delta_{ij} \quad \forall i,j \in [N]$$

# Construct block-encodings

Sparse matrix:

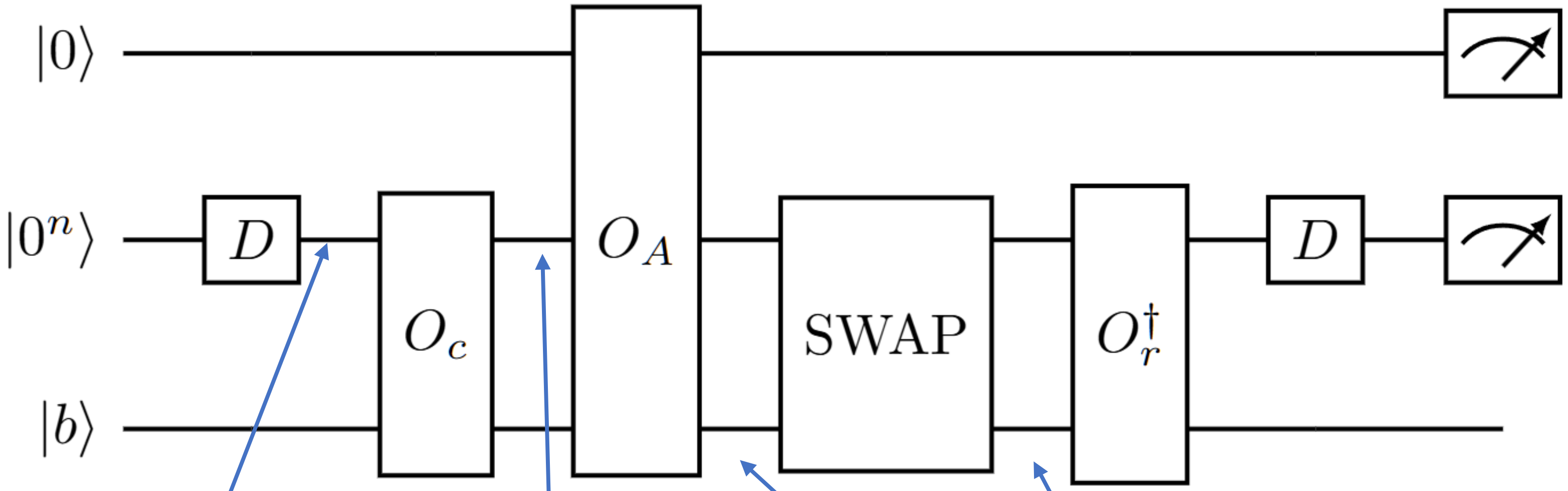
- Let $A$ be a sparse matrix such that each row/column has $\leq S = 2^s$ nonzero entries
- Three input oracles:

$$O_r|l\rangle|i\rangle = |r(i,l)\rangle|i\rangle$$
$$O_c|r\rangle|j\rangle = |c(j,r)\rangle|j\rangle$$
$$O_A|0\rangle|i\rangle|j\rangle = \left(A_{ii}|0\rangle + \sqrt{1 - |A_{ii}^2|}|1\rangle\right)|i\rangle|j\rangle$$

- $r(i,l)$ is the position of the $l$-th nonzero entry in the $i$-th row
- $c(j,r)$ is the position of the $r$-th nonzero entry in the $j$-th column

# Construct block-encodings

Sparse matrix:




$$|0\rangle \frac{1}{\sqrt{S}} \sum_{r=0}^{S-1} |r\rangle\, |j\rangle$$

$$|0\rangle \frac{1}{\sqrt{S}} \sum_{r=0}^{S-1} |c(j,r)\rangle\, |j\rangle$$

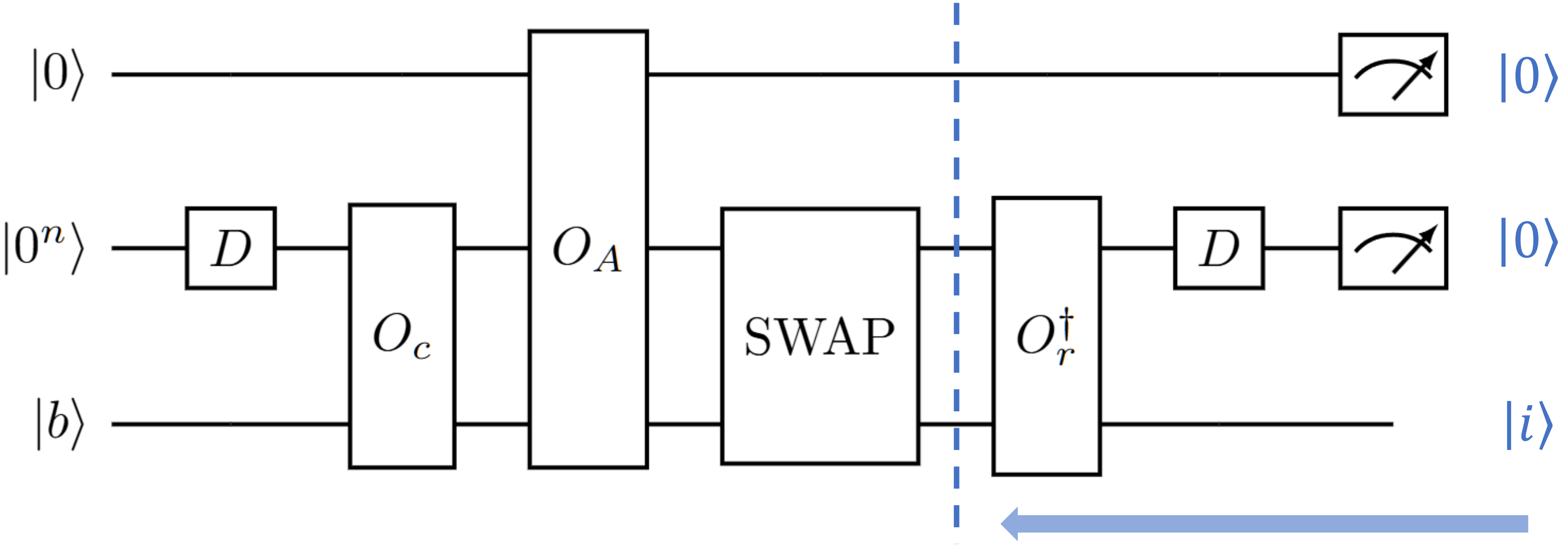
$$\frac{1}{\sqrt{S}} \sum_{r=0}^{S-1} A_{c(j,r),j} |0\rangle |c(j,r)\rangle\, |j\rangle + |1\rangle |*\rangle$$

$$\frac{1}{\sqrt{S}} \sum_{r=0}^{S-1} A_{c(j,r),j} |0\rangle\, |j\rangle |c(j,r)\rangle + |1\rangle |*\rangle$$

# Construct block-encodings

Sparse matrix:

$(S, n+1, 0)$-b.e.




$$\frac{1}{\sqrt{S}}\sum_{r=0}^{S-1} A_{c(j,r),j}|0\rangle\,|j\rangle|c(j,r)\rangle + |1\rangle|*\rangle \qquad \frac{1}{\sqrt{S}}\sum_{l=0}^{S-1}|0\rangle\,|r(i,l)\rangle|i\rangle$$

$$\langle 0|\langle 0|\langle i|U_A|0\rangle|0\rangle|j\rangle = \frac{1}{S}\sum_{l,r=0}^{S-1} A_{c(j,r),j}\,\delta_{i,c(j,r)}\delta_{j,r(i,l)} = \frac{1}{S}A_{ij}$$