A survey of quantum computing for PDEs

Matthias Möller

Delft University of Technology
Delft Institute of Applied Mathematics

Research interests and collaborations

Quantum computing in aerospace engineering applications

Research interests and collaborations

Quantum algorithms and benchmarks for NISQ devices

This survey is based on …

- *A review of quantum algorithms for partial differential equations in fluid and structural mechanics*
  G. Balducci, B. Chen, MM, and R. de Breuker

  in preparation for the special issue: Quantum Computing Applications in Computational Engineering in Frontiers in Mechanical Engineering, 2022
- *Quantum Algorithms for Solving Partial Differential Equations*
  Report by A. Pesah, 2020
- about 50 articles and reports cited in the works above

Different quantum computing principles

- **Discrete-variable quantum computing (DVQC)**: eigenstates of a discrete variable form the computational basis of a finite-dimensional Hilbert space

$$|\psi\rangle = \sum_{i=0}^{2^n-1} c_i |b_i\rangle\,, \qquad \sum_{i=0}^{2^n-1} |c_i|^2 = 1, \qquad \langle b_i | b_j \rangle = \delta_{ij}$$

- **Continuous-variable quantum computing (CVQC)**: eigenstates of a continuous variable form the basis of an infinite-dimensional Hilbert space

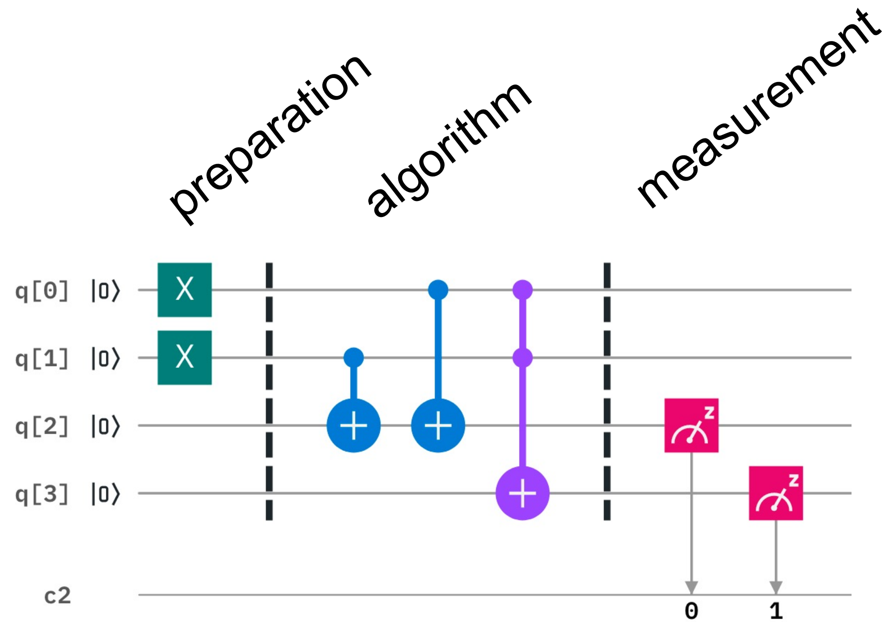
$$|\psi\rangle = \int_{-\infty}^{\infty} c(x)|x\rangle dx\,, \qquad \langle x' | x \rangle = \delta(x' - x)$$

DVQC: Gate-based universal quantum computers

- Mathematical model

$$|\psi_{out}\rangle = U_m \cdot \ldots \cdot U_1 |\psi_0\rangle$$

- Hardware realizations with ~100 superconducting qubits, e.g., by IBM, Google, Rigetti, Intel, …



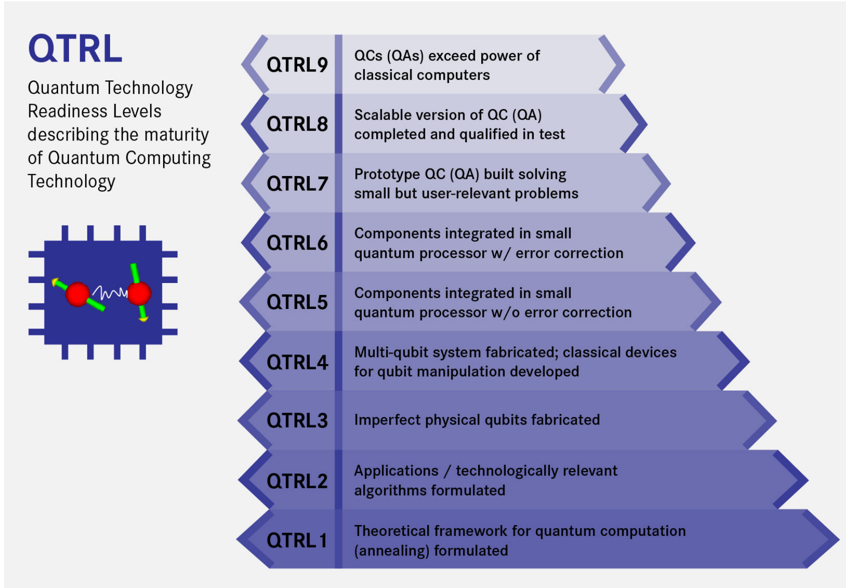
DVQC: Gate-based universal quantum computers

- Mathematical model

$$|\psi_{out}\rangle = U_m \cdot \ldots \cdot U_1 |\psi_0\rangle$$

- Hardware realizations with ~100 superconducting qubits, e.g., by IBM, Google, Rigetti, Intel, …

- QTRL 4-5 / 9 (expected in 2035)



https://www.fz-juelich.de/ias/jsc/EN/Research/ModellingSimulation/QIP/QTRL/_node.html

DVQC: Quantum annealing

- Mathematical model

$$|\psi_0\rangle = \underset{|\psi\rangle}{\arg\min}\langle\psi|H|\psi\rangle$$

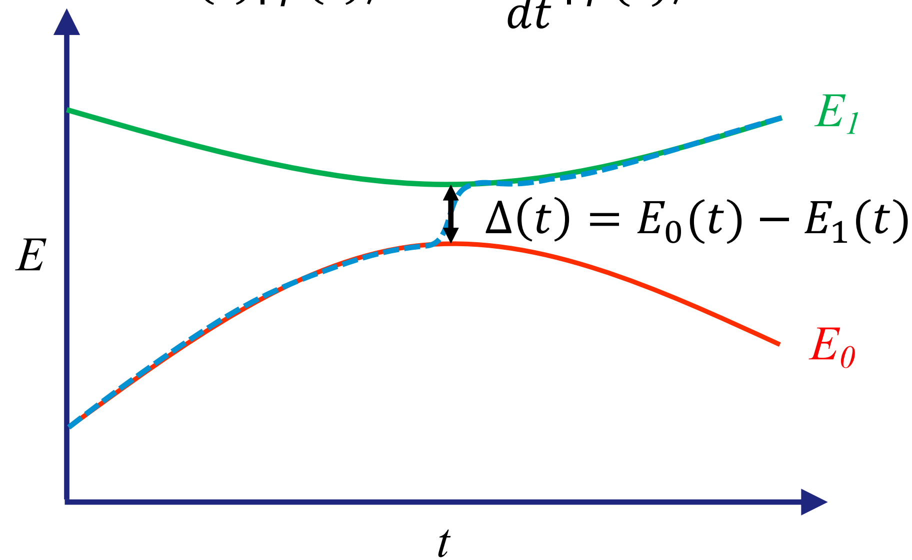
- Path of Hamiltonians for $t \in [0, T]$

$$H(t) = \big(1 - f(t)\big)H_I + f(t)H_P$$

with easy-to-compute ground state $|\psi_0\rangle$ for the initial Hamiltonian H_I

- Ground-state evolution

$$H(t)|\psi(t)\rangle = -i\frac{d}{dt}|\psi(t)\rangle$$



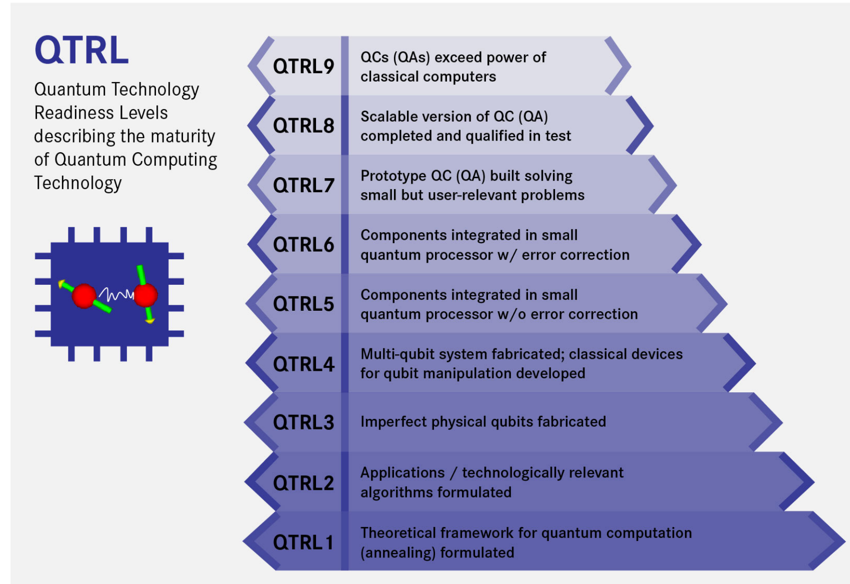
DVQC: Quantum annealing

- Mathematical model

$$|\psi_0\rangle = \underset{|\psi\rangle}{\arg\min} \langle\psi|H|\psi\rangle$$

- Hardware realizations by D-Wave with up to 5000 qubits

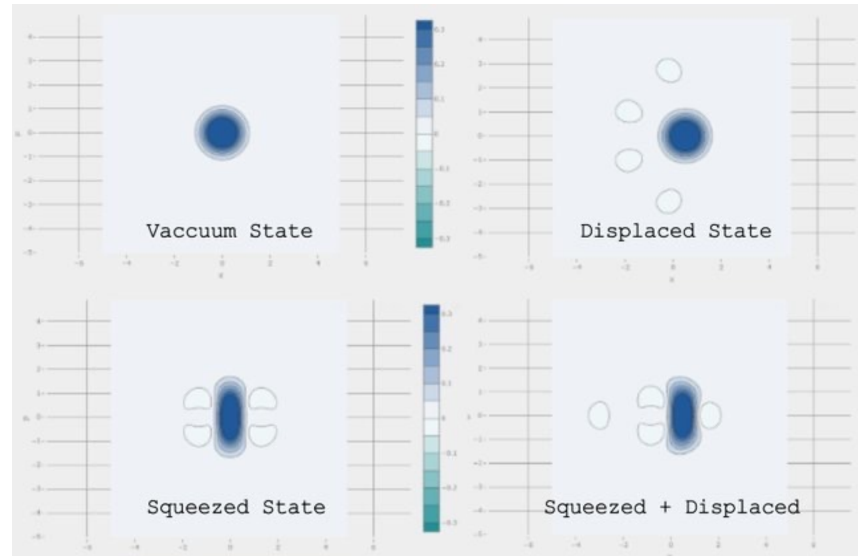
- QTRL 8-9



https://www.fz-juelich.de/ias/jsc/EN/Research/ModellingSimulation/QIP/QTRL/_node.html

CVQC

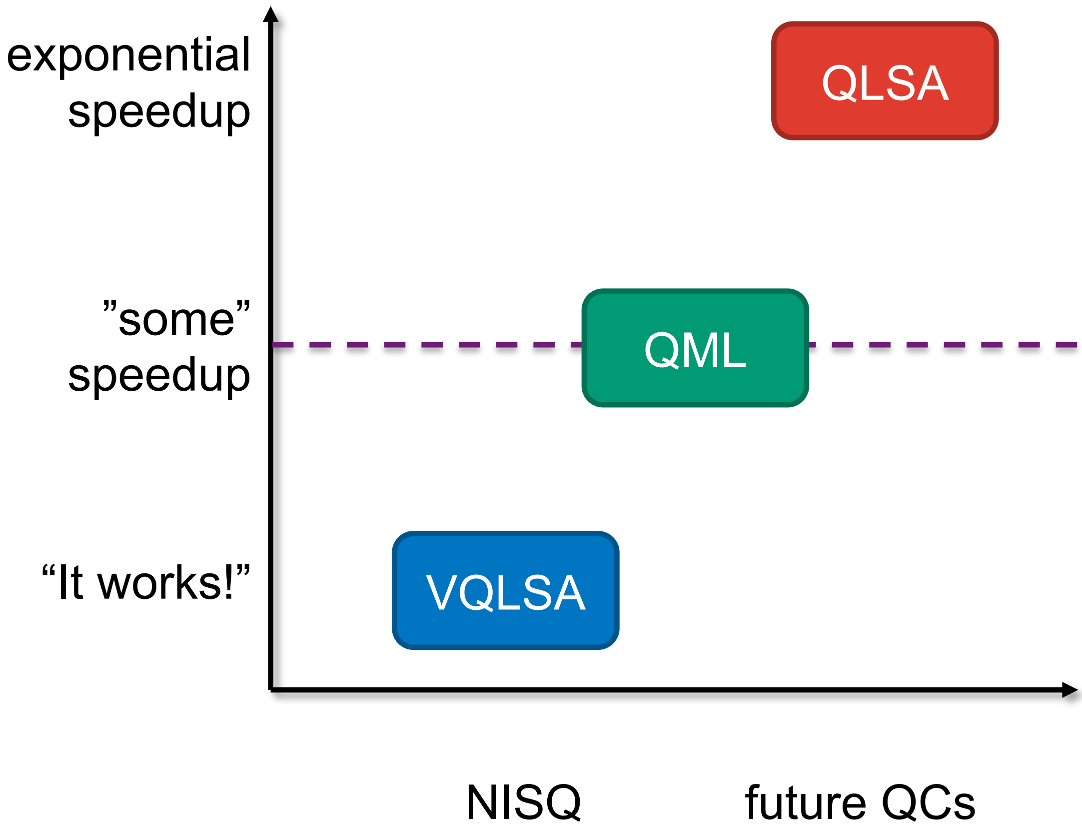
- Hardware realizations by Xanadu (photonic), concepts for trapped ions [Maslennikov et al. 2019]
- Comprehensive introduction by S.Buck et al. (2021)
- Quantum search by A.K. Pati et al. (2020) and D. Su et al. (2018)



Different interpretations of the word "quantum algorithm"

- **Theoretical analysis of an algorithm** – *HHL paper (2008) claims exponential speedup for "solving" Ax=b under conditions on the matrix A*
- **Application-specific theoretical analysis** – *Montanaro et al. (2016) shows that no exponential speedup for P_k-FEM can be achieved for fixed dimension*
- **Application-specific circuit + cost estimation** – *Cao et al. (2013) present textbook quantum circuits for solving Poisson's equation with FDM*
- **Execution of application-specific circuit on QVM** – *Wang et al. (2020) demonstrate a fast Poisson solver on Sunway TaihuLight (yes, it works 🙂)*
- **Execution of application-specific circuit on QPU –** *Morrell et al. (2021) show that solving a 2x2 system on IBM-Q fails due to noise (doesn't work 😪)*

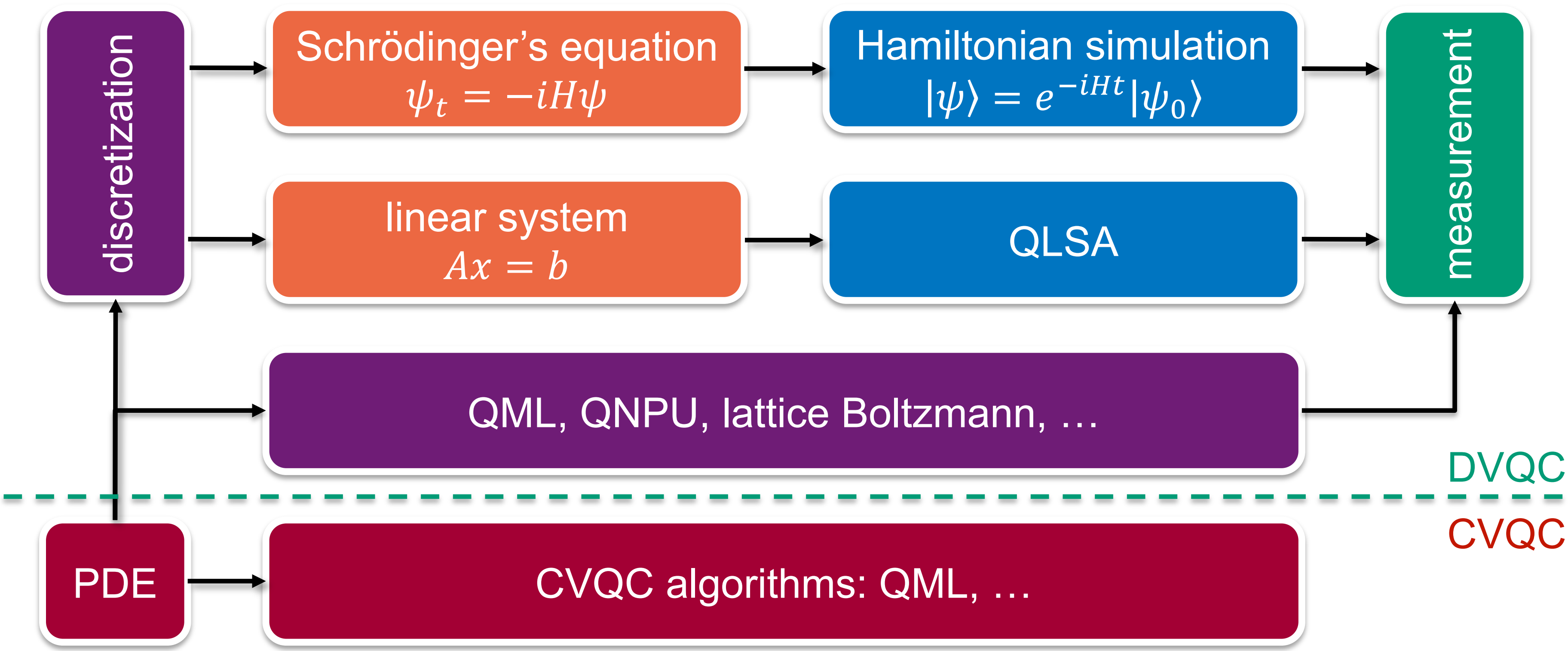
What should we aim for?



M. Troyer: super-quadratic speedup is a must because

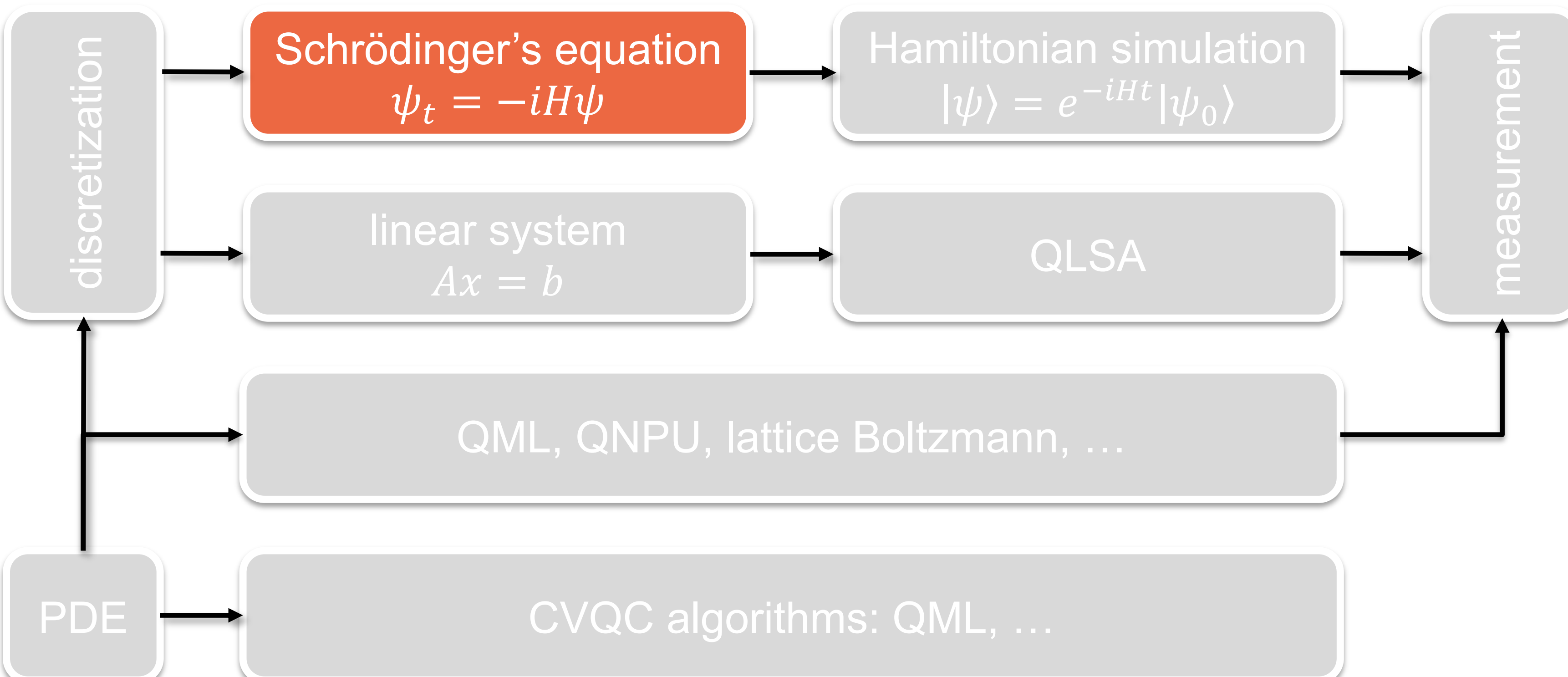
- operations will be 10-12 orders of magnitude slower
- I/O will be 10.000x slower

Quantum algorithms for solving PDEs



Inspired by A. Pesah's report "Quantum Algorithms for Solving Partial Differential Equations" 2020.

Quantum algorithms for solving PDEs



Inspired by A. Pesah's report "Quantum Algorithms for Solving Partial Differential Equations" 2020.

Schrödinger's equation
$\psi_t = -iH\psi$

Given: $\dot{x} = Ax,\ x(t_0) = x_0$

- Von Neumann measurement [von Neumann 1932, Childs et al. 2002]

$$\begin{pmatrix} 0 & iA^\dagger \\ -iA & 0 \end{pmatrix} \Rightarrow H = iA^\dagger \otimes |0\rangle_P\langle 1| - iA \otimes |1\rangle_P\langle 0|$$

- State after Hamiltonian simulation [Leyton, Osborne 2008]

$$|\Psi\rangle = e^{iHt}|\psi\rangle|0\rangle_P = \sum_{k=0}^{\infty} \frac{(iHt)^k}{k!}|\psi\rangle|0\rangle_P = |\psi\rangle|0\rangle_P + tA|\psi\rangle|1\rangle_P - \cdots$$

- Post-selection on "1" after measurement on the ancillary qubit
- Procedure from [HHL 2008] to correct for first-order truncation
- **Caveat**: success probability $\frac{1}{2}t^2$ (roughly $16/t^2$ 'fresh' states $|\psi\rangle$ needed)

Schrödinger's equation
$\psi_t = -iH\psi$

Given: $\dot{x} = Ax,\ x(t_0) = x_0$

- Matrix decomposition $A = A_H + A_A$
- Baker–Campbell–Hausdorff formula

$$e^{iAt} = e^{iA_H t} \cdot e^{iA_A t}, \qquad \text{if } [A_H, A_A] = 0$$

- Hamiltonian simulation of A_H and A_A via unitary dilation of $\hat{O} = e^{iA_A t}$

$$\begin{pmatrix} \hat{O} & \sqrt{1-\hat{O}^2} \\ \sqrt{1-\hat{O}^2} & -\hat{O} \end{pmatrix} |\psi\rangle|0\rangle = \hat{O}|\psi\rangle|0\rangle + \sqrt{1-\hat{O}^2}|\psi\rangle|1\rangle$$

- Black-Scholes equation [Gonzalez-Conde et al. 2021]

$$f_t = af + bf_x - cf_{xx} = (ib(-i\partial_x) + aI + c(-i\partial_x)^2)f$$

- **Caveat**: exponential scaling in t if $[A_H, A_A] \neq 0$ [Berry 2014]

Schrödinger's equation
$\psi_t = -iH\psi$

Given: $\dot{x} = Ax,\ x(t_0) = x_0$

- Derivative of Schrödinger's equation [Costa et al. 2019]

$$\psi_{tt} = -H^2\psi$$

- Hermitian matrix

$$H = \begin{pmatrix} 0 & B \\ B^\dagger & 0 \end{pmatrix} \quad \Rightarrow \quad H^2 = \begin{pmatrix} BB^\dagger & 0 \\ 0 & B^\dagger B \end{pmatrix}$$

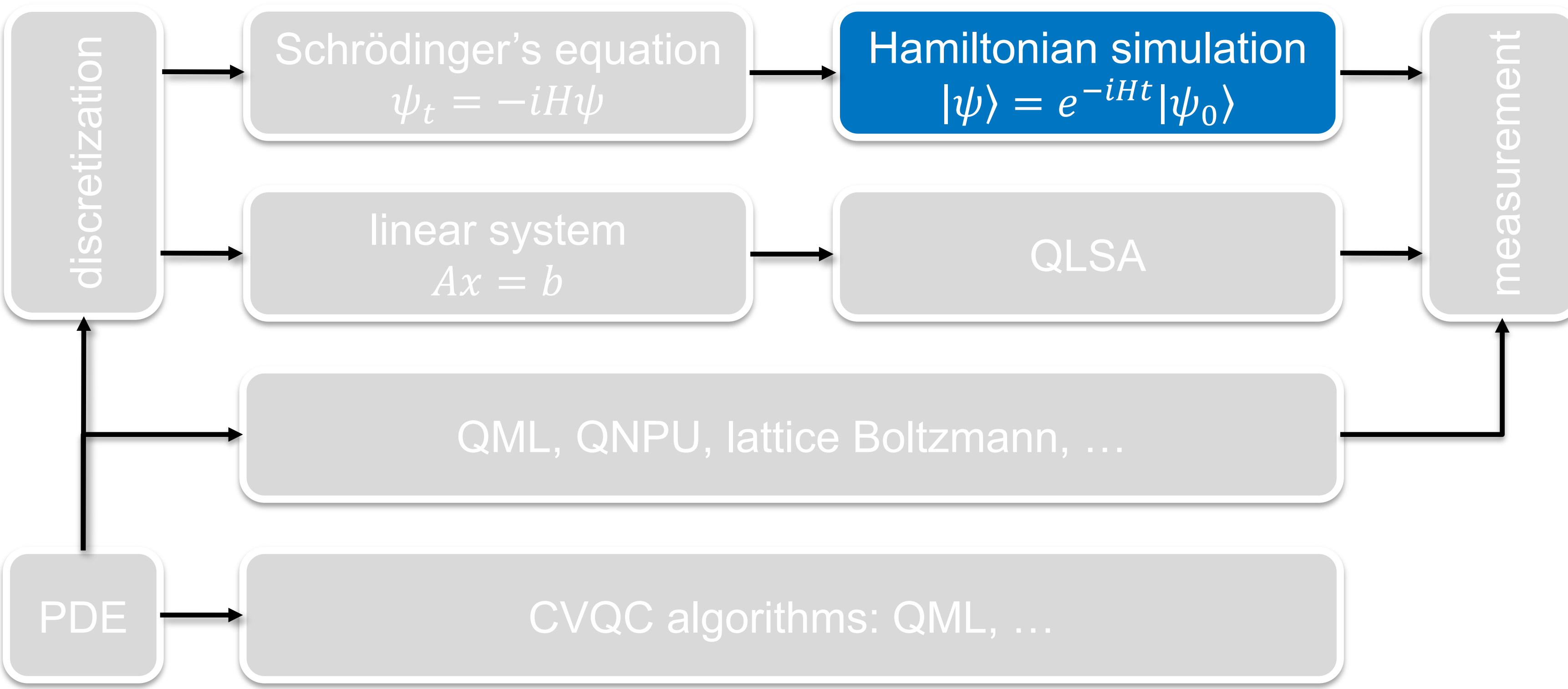
- Wave equation

$$\partial_{tt} f = -\Delta f \approx Af \quad \Rightarrow \quad \text{find } A = BB^\dagger$$

- **Example**: graph Laplacian

$$B_{ev} = \begin{cases} 1 & e = (v,w), v < w \\ -1 & e = (v,w), v > w \\ 0 & \text{otherwise} \end{cases} \quad \Rightarrow \quad B_{ev} = \begin{pmatrix} 1 & -1 & 0 & 0 \\ 0 & \ddots & \ddots & 0 \\ 0 & 0 & 1 & -1 \end{pmatrix}$$

Quantum algorithms for solving PDEs



Inspired by A. Pesah's report "Quantum Algorithms for Solving Partial Differential Equations" 2020.

Hamiltonian simulation
$|\psi\rangle = e^{-iHt}|\psi_0\rangle$

Hamiltonian simulation (approach #1)

- **Given**: Hamiltonian H ($2^n \times 2^n$ Hermitian on n qubits), time t, and error ϵ
- **Goal**: find an algorithm to approximate U such that $\left\|U - e^{iHt}\right\| \leq \epsilon$

- Decomposition into local Hamiltonians [Lloyd 1996]

$$H = \sum_{\ell=1}^{L} H_\ell \,,$$

$$\left(e^{A\frac{t}{r}}e^{B\frac{t}{r}}\right)^r = e^{(A+B)t+\frac{1}{2}[A,B]\frac{t^2}{r}+\mathcal{O}\left(\frac{t^3}{r^2}\right)}$$

- Suzuki-Trotter decomposition [Suzuki 1991]

$$e^{-iHt} \approx \left(\prod_{\ell=1}^{L} e^{-iH_\ell\frac{t}{r}}\right)^r, \qquad r \gg 1$$

Hamiltonian simulation (approach #2)

Hamiltonian simulation
$|\psi\rangle = e^{-iHt}|\psi_0\rangle$

- **Given**: Hamiltonian H ($2^n \times 2^n$ Hermitian on n qubits), time t, and error ϵ
- **Goal**: find an algorithm to approximate U such that $\|U - e^{iHt}\| \leq \epsilon$

- Truncated Taylor expansion

$$\mathrm{e}^{iHt} = I - iHt - \tfrac{1}{2}H^2t^2 + \tfrac{i}{6}H^3t^3 + \cdots$$

- Linear combination of unitary operators [Berry et al. 2015]

$$H = \sum\nolimits_{\ell} \alpha_\ell H_\ell \quad \Rightarrow \quad H^n = \sum\nolimits_{\ell_1,\dots,\ell_n} \alpha_{\ell_1} \dots \alpha_{\ell_k} H_{\ell_1} \dots H_{\ell_n}$$

Hamiltonian simulation (complexity)

Hamiltonian simulation
$|\psi\rangle = e^{-iHt}|\psi_0\rangle$

	Gate complexity [1]	Query complexity [2]-[5]
1st-order Trotter	$\mathcal{O}(t^2/\epsilon)$	$\mathcal{O}\left(s^3 t (st/\epsilon)^{\frac{k}{2}}\right)$
Taylor expansion	$\mathcal{O}\left(\frac{t \log^2(t/\epsilon)}{\log\log t/\epsilon}\right)$	$\mathcal{O}\left(\frac{s^2 \Vert H\Vert_{\max} \log s^2 \Vert H\Vert_{\max}/\epsilon}{\log\log s^2 \Vert H\Vert_{\max}/\epsilon}\right)$
Quantum walk	$\mathcal{O}(t/\sqrt{\epsilon})$	$\mathcal{O}(s\Vert H\Vert_{\max} t/\sqrt{\epsilon})$
Quantum signal processing	$\mathcal{O}(t + \log 1/\epsilon)$	$\mathcal{O}\left(st\Vert H\Vert_{\max} + \frac{\log 1/\epsilon}{\log\log 1/\epsilon}\right)$

[1] Childs 2017, [2] Kothari 2017, [3] Berry 2015, [4] Berry 2015, [5] Low 2017

Leyton and Osborne *2008*

- First-order systems of the form

$$\dot{\boldsymbol{x}}(t) = \begin{pmatrix} f_1(\boldsymbol{x}) \\ \vdots \\ f_N(\boldsymbol{x}) \end{pmatrix}, \qquad f_j(\boldsymbol{x}) = \sum_{k,l=1}^{N} a_{kl}^{(j)} x_k x_l, \qquad \sum_{j=1}^{N} |x_j|^2 = 1$$

- **Example**: Orszag-McLaughlin dynamical system

$$\dot{x}_j = x_{j+1}x_{j+2} + x_{j-1}x_{j-2} - 2x_{j+1}x_{j-1}, \qquad j = 1, \dots N$$

- $|a_{ij}| = \mathcal{O}(1)$ and A is s-sparse, i.e., each f_j involves at most $s/2$ monomials and each variable x_j appears in at most $s/2$ polynomials f_j
- Lipschitz constant: $\|F(\boldsymbol{x} - \boldsymbol{y})\| \leq \mathcal{O}(1) \cdot \|\boldsymbol{x} - \boldsymbol{y}\|$ in ball $\|\boldsymbol{x}\| \leq 1, \|\boldsymbol{y}\| \leq 1$
- We assume that the initial state can be prepared efficiently

Leyton and Osborne *2008*

Schrödinger's equation $\psi_t = -iH\psi$ → Hamiltonian simulation $|\psi\rangle = e^{-iHt}|\psi_0\rangle$ → M

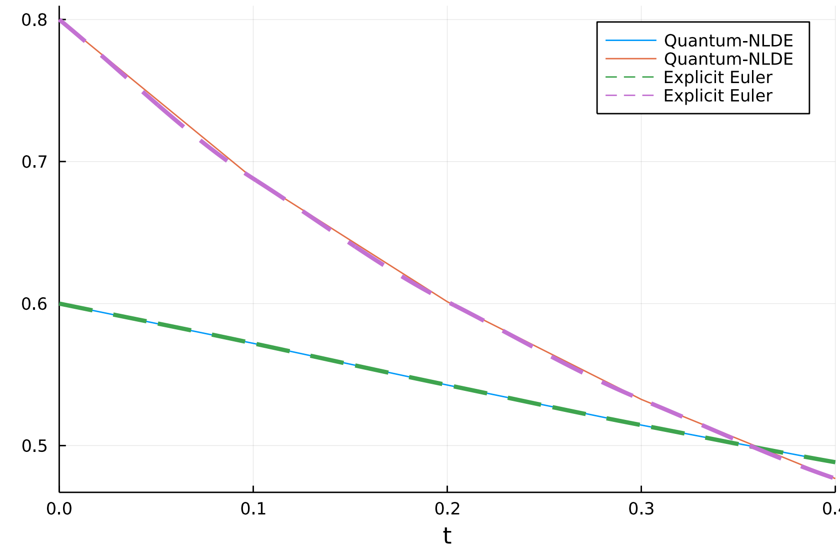
- Explicit Euler method

$$|\psi'\rangle = e^{iH\Delta t}|\psi\rangle|0\rangle \quad \Rightarrow \quad |\psi(t+\Delta t)\rangle = |\psi(t)\rangle + \Delta t A|\psi(t)\rangle$$

- Success probability of a single step $\frac{1}{2}\Delta t^2$; $16/\Delta t^2$ 'fresh' $|\psi\rangle$ needed
- Temporal scaling $\left(\frac{16}{\Delta t^2}\right)^m$, spatial scaling $\left(\frac{16}{\Delta t^2}\right)^m \log N$ for m steps
- Hamiltonian simulation must be performed with error $\delta < \left(3\mathcal{O}(1)\right)^{-m}$ to ensure that the m-th iterate is exponentially close to the desired state

QuDiffEq

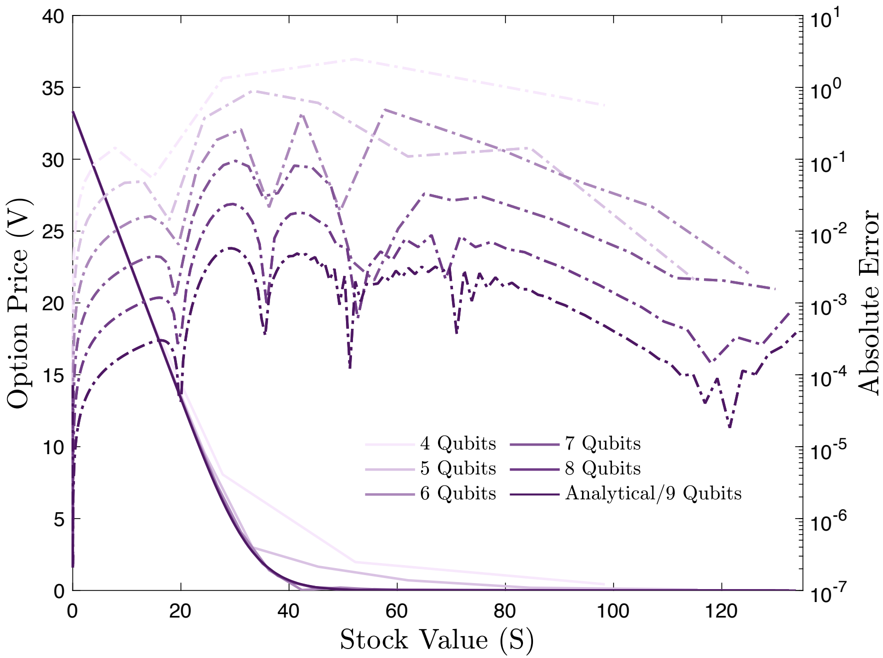
- Quantum algorithms for linear and nonlinear differential equations
- Papers with Code
  - [Leyton, Osborne 2008]
  - [Berry et al. 2010]
  - [Xin et al. 2018]

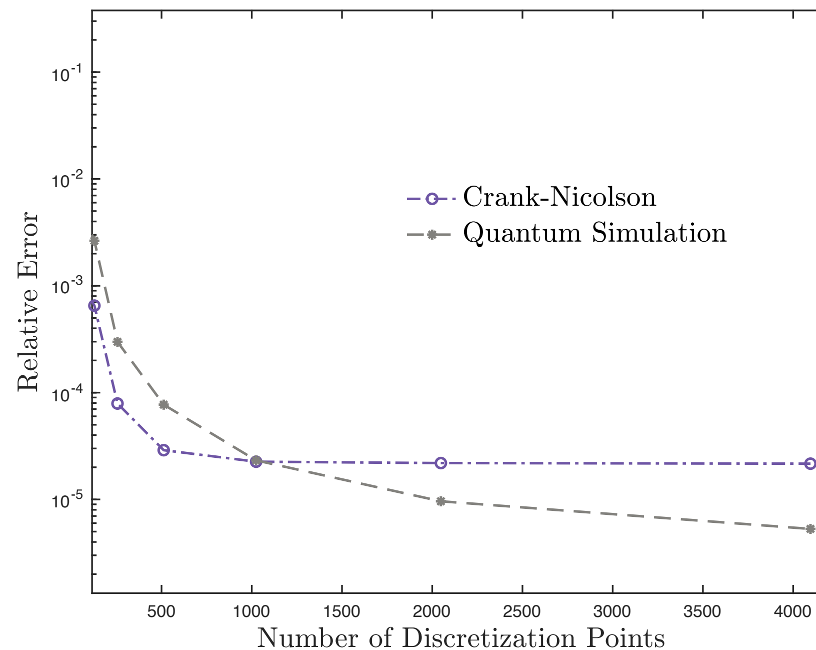


$$\dot{x}_1 = x_2 - 3x_1^2$$
$$\dot{x}_2 = -x_2^2 - x_1 x_2$$

Gonzalez-Conde et al. 2022

Black-Scholes equation for European put options

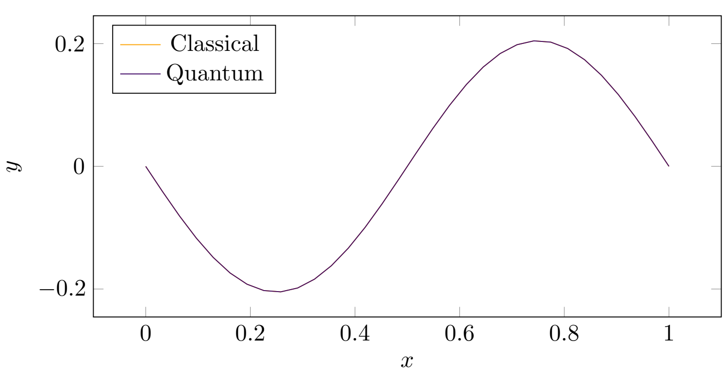


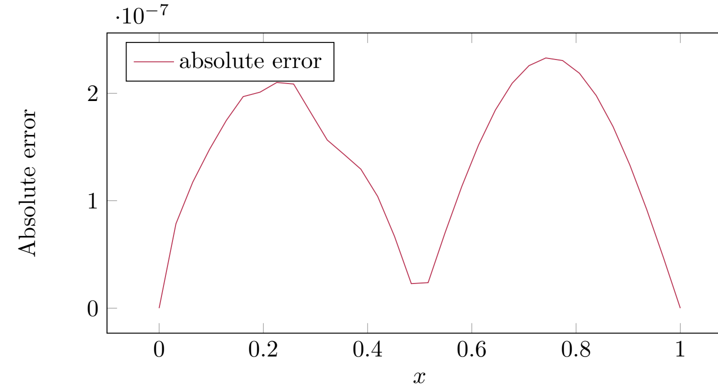


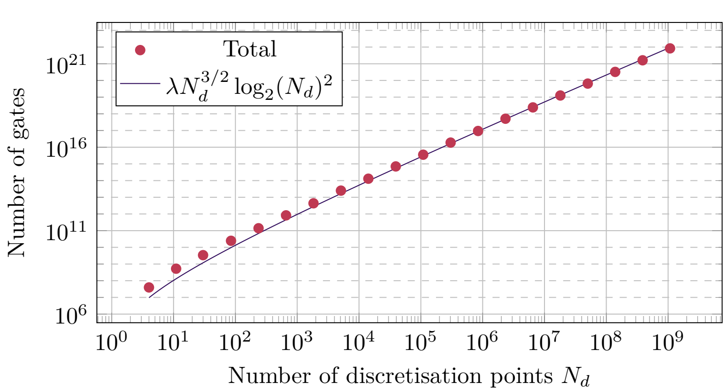
Precision comparable to classical methods with 10 qubits and 94 entangling gates on fault-tolerant QC. Complexity $\mathcal{O}(\text{poly}\, n)$. Success probability 0.6.

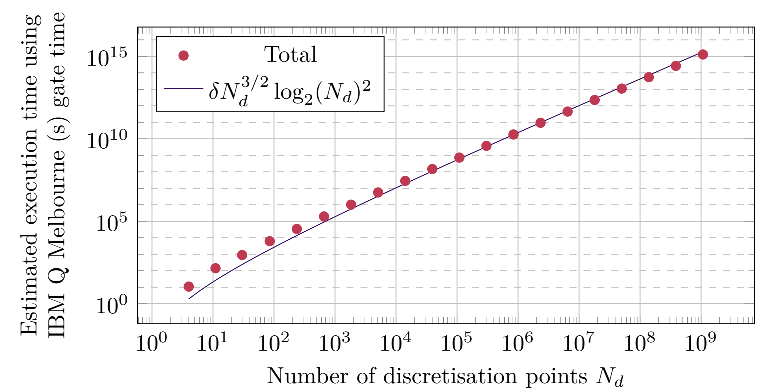
Suau et al. 2022

Wave equations

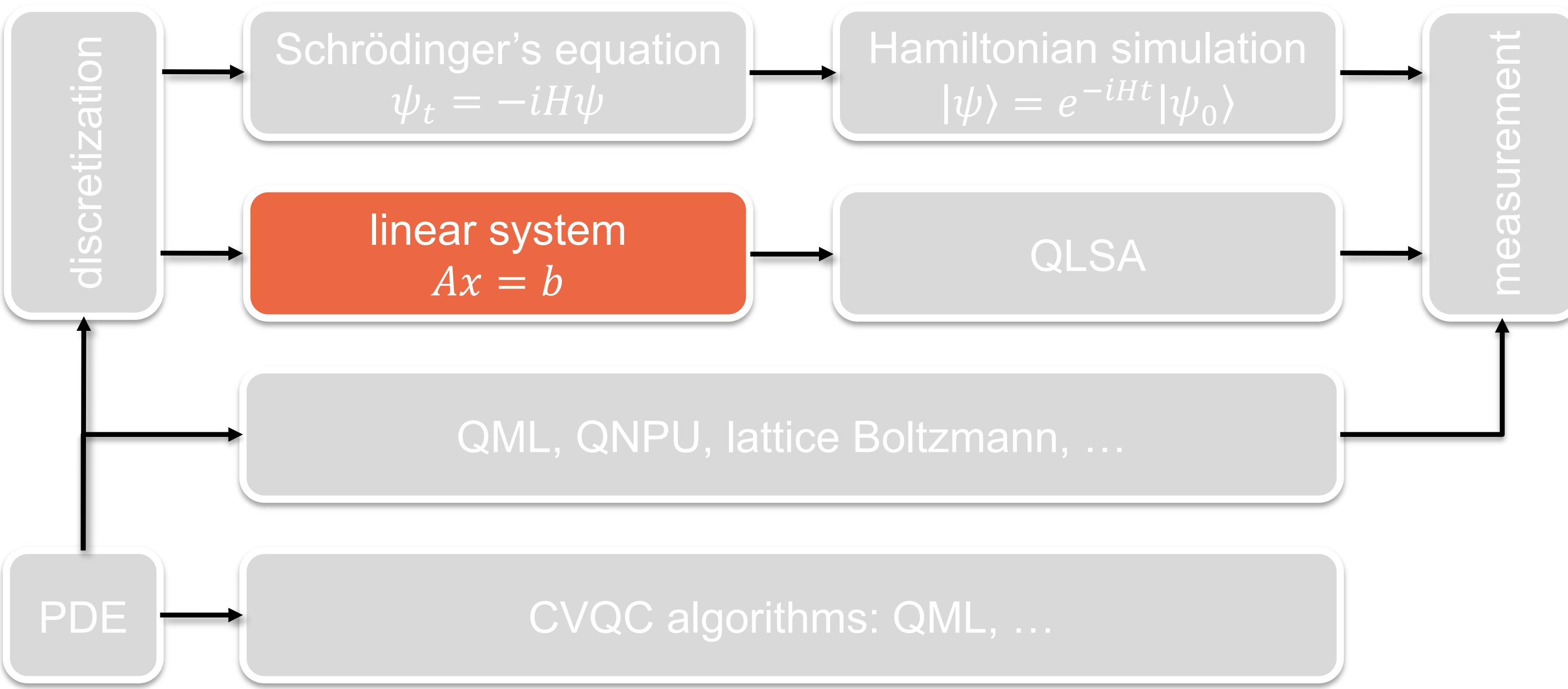








Quantum algorithms for solving PDEs



Inspired by A. Pesah's report "Quantum Algorithms for Solving Partial Differential Equations" 2020.

Given: $\dot{x} = Ax + b,\ x(t_0) = x_0$

linear system
$Ax = b$

$\mathcal{O}\big(\mathrm{poly}(1/\epsilon)\big)$

- Unroll Euler method in time

$$\begin{pmatrix} I & 0 & 0 & 0 \\ -(I + \Delta t A) & I & 0 & 0 \\ \ddots & \ddots & \ddots & \ddots \\ 0 & 0 & -(I + \Delta t A) & I \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_m \end{pmatrix} = \begin{pmatrix} x_{in} \\ \Delta t b \\ \vdots \\ \Delta t b \end{pmatrix}$$

- Apply HHL-type algorithm to obtain the solution at all times

$$|x\rangle = \sum_{j=0}^{m} |t_j\rangle |x_j\rangle$$

- Application and analysis for the heat equation yields poor scaling with precision [Linden et al. 2020] even with the improved variant of the QLSA 'solver' [Berry et al. 2017]

Solution: $x(t) = e^{At}x_0 + (e^{At} - I)A^{-1}b$

linear system
$Ax = b$

$\mathcal{O}\big(\text{poly}\log(1/\epsilon)\big)$

- Truncated exponentials

$$e^z \approx \sum_{j=0}^{k} \frac{z^j}{j!}, \qquad (e^z - 1)z^{-1} \approx \sum_{j=1}^{k} \frac{z^{j-1}}{j!}$$

- Linear system [Berry et al. 2017]

$$C_{m,k,p}(\Delta t A)|x\rangle = |0\rangle|x_0\rangle + \Delta t \sum_{j=0}^{m-1} |j(k+1)+1\rangle|b\rangle$$

$$\begin{pmatrix}
I &&&&&&&&&& \\
-\Delta t A & I &&&&&&&&& \\
& -\Delta t A/2 & I &&&&&&&& \\
&& -\Delta t A/3 & I &&&&&&& \\
-I & -I & -I & -I & I &&&&&& \\
&&&& -\Delta t A & I &&&&& \\
&&&&& -\Delta t A/2 & I &&&& \\
&&&&&& -\Delta t A/3 & I &&& \\
&&&& -I & -I & -I & -I & I && \\
&&&&&&&& -I & I & \\
&&&&&&&&& -I & I
\end{pmatrix}_{2,3,2}, \quad
\begin{pmatrix}
|x_0\rangle \\ \Delta t|b\rangle \\ 0 \\ 0 \\ 0 \\ \Delta t|b\rangle \\ 0 \\ 0 \\ 0 \\ 0 \\ 0
\end{pmatrix}$$

linear system
$Ax = b$

$\mathcal{O}\big(\text{poly}\log(1/\epsilon)\big)$

Given: $\dot{x} = A(t)x + b(t),\ x(t_0) = x_0$

- Chebyshev pseudo-spectral approximation

$$x(t) = \sum_{k=0}^{n} c_k T_k(t) \quad \Rightarrow \quad \dot{x}(t_l) = A(t_l)x(t_l) + b(t_l), \qquad t_l = \cos\frac{l\pi}{n}$$

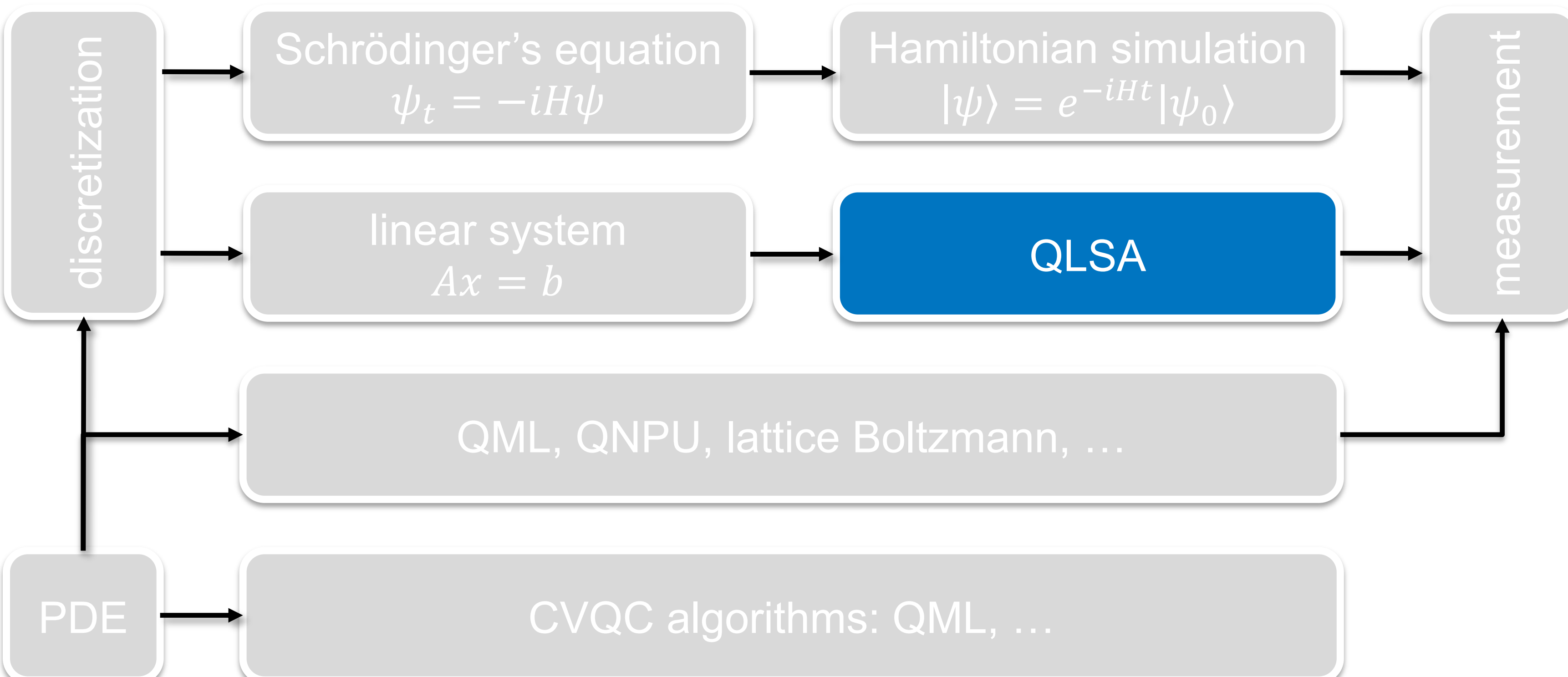
- Rescaled differential equation (Childs and Liu 2020)

$$\dot{x}\big(\gamma(t)\big) = -\tfrac{t^{n+1}-t^n}{2}\Big[A\big(\gamma(t)\big)x\big(\gamma(t)\big) + b\big(\gamma(t)\big)\Big],$$

where $\gamma\colon [t^n, t^{n+1}] \mapsto [-1,1]$ is defined as $\gamma\colon t \mapsto 1 - \frac{2(t-t^n)}{t^{n+1}-t^n}$

- Combined with the $C_{m,k,p}$-approach from [Berry et al. 2017] this extends their work to ODEs with time-dependent coefficient matrices and vectors

Quantum algorithms for solving PDEs



Inspired by A. Pesah's report "Quantum Algorithms for Solving Partial Differential Equations" 2020.

Quantum linear 'solver' algorithm

	QLSA
Problem: $a = \langle x^{\dagger} \| M \| x \rangle \quad s.t. \quad A\|x\rangle = \|b\rangle$	$\mathcal{O}(s\sqrt{\kappa}N \log 1/\epsilon)$
Original HHL algorithm [Harrow et al. 2008]	$\mathcal{O}(s^2\kappa^2 \log(N)/\epsilon)$
Improved versions of HHL	
– VTAA [Ambainis 2010]	$\mathcal{O}(s^2\kappa \log^3 \kappa \log(N)/\epsilon^3)$
– AQC [Subasi et al. 2019]	$\mathcal{O}(\kappa^2 \log(\kappa)/\epsilon)$
– AQC [An and Lin 2019]	$\mathcal{O}(\kappa \text{ poly} \log(\kappa/\epsilon))$
QLSA w/o phase estimation [Childs et al. 2017]	$\mathcal{O}(\text{poly} \log(1/\epsilon))$
Dense matrices [Wossnig et al. 2018]	$\mathcal{O}\left(\kappa^2\sqrt{N}\text{poly} \log(N)/\epsilon\right)$

Original HHL algorithm

$$A = \sum_{j=0}^{2^n-1} \lambda_j |u_j\rangle\langle u_j|, \quad \boxed{b = \sum_{j=0}^{2^n-1} b_j |u_j\rangle,} \quad |x\rangle = A^{-1}|b\rangle = \sum_{j=0}^{2^n-1} \lambda_j^{-1} b_j |u_j\rangle$$

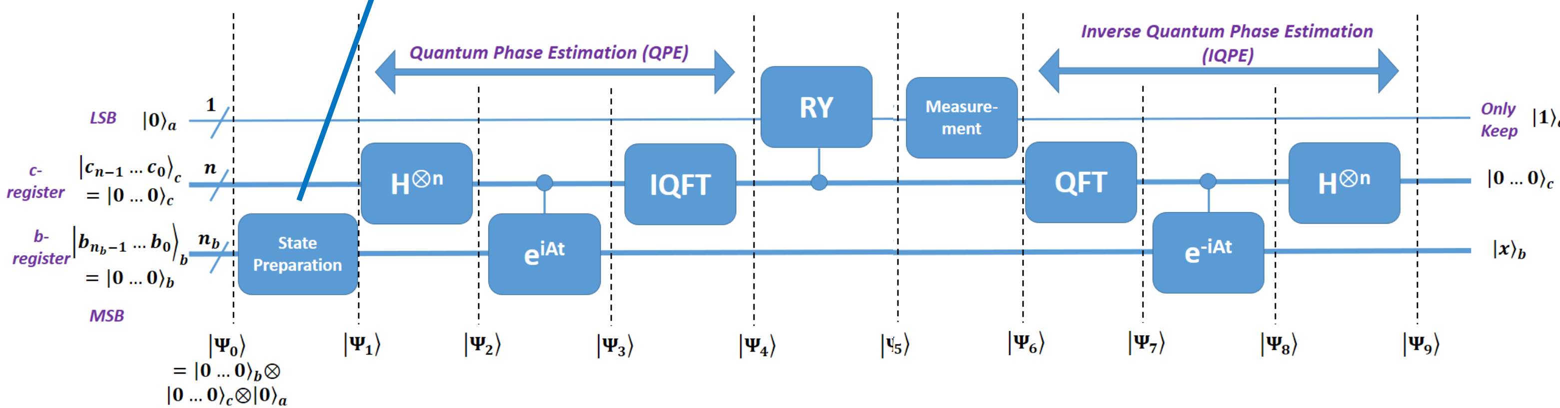


Figure 1 from [Morrell and Wong 2021]

State preparation: $|\psi_{init}\rangle = U_{prep}|0\rangle$

- General states cannot be prepared efficiently, not even approximated

$$N \text{ grid points} \quad \Rightarrow \quad n = \log N \text{ qubits} \quad \Rightarrow \quad \left|U_{prep}\right| = \mathcal{O}(N)$$

  uniformly controlled rotations [Mottonen et al. 2004] using $\mathcal{O}(2^n)$ gates

- Certain states of the form $|\psi\rangle = \sum_i \sqrt{p_i}\,|i\rangle$ can be prepared efficiently, e.g., using quantum GANs [Zoufal et al. 2019] using $\mathcal{O}(\text{poly } n)$ gates

- Reducing time complexity by adding ancillary qubits
  - Low-depth approach: $\mathcal{O}(n^2)$ using $\mathcal{O}\left(2^{n^2}\right)$ ancillae [Zhang et al. 2021]
  - s-sparse states: $\Theta(\log ns)$ using $\mathcal{O}(ns \log s)$ ancillae [Zhang et al. 2022]

Does any of this work in practice?

- QLSA for $Ax = b$
  - HW-realization for 2×2 matrix [Cai et al. 2013], [Barz et al. 2013], [Pan et al. 2013], and 8×8 matrix [Wen et al. 2018]
  - 2×2, 4×4, and 8×8 on IBM, Rigetti, IonQ [Cornelissen et al. 2021]
  - Other authors report that "*due to imperfection and noise in a real quantum computer [ibmq_santiago], the hardware execution of the same circuit does not give satisfactory results*" [Morrell and Wong 2021]
- Okay, so no chance for solving ODEs / transient PDEs with QC in near term
- How about solving Poisson's equation discretized by FDM / FEM?

$\mathcal{O}(s\sqrt{\kappa}N \log 1/\epsilon)$ versus $\mathcal{O}(\text{poly} \log(1/\epsilon))$

- General state preparation is exponentially expensive, i.e., $\mathcal{O}(N)$
  - Polynomials/functions with local support can be prepared efficiently
- $\kappa = \mathcal{O}\left(N^{d/2}\right)$ in standard FEM ⇒ no exponential speedup
  - Quantum-SPAI precondioner, i.e. $PAx = Pb$ [Clader et al. 2013]
    - $\mathcal{O}(s^2)$ queries to PA-oracle; $\mathcal{O}(s^3)$ runtime
    - $\kappa = \mathcal{O}(1)$ or $\kappa = \mathcal{O}(\log N)$
- $s = ?$
- $1/\epsilon = \mathcal{O}(N)$ in most discretization schemes ⇒ no exponential speedup

No exponential speedup for elliptic problems for fixed d

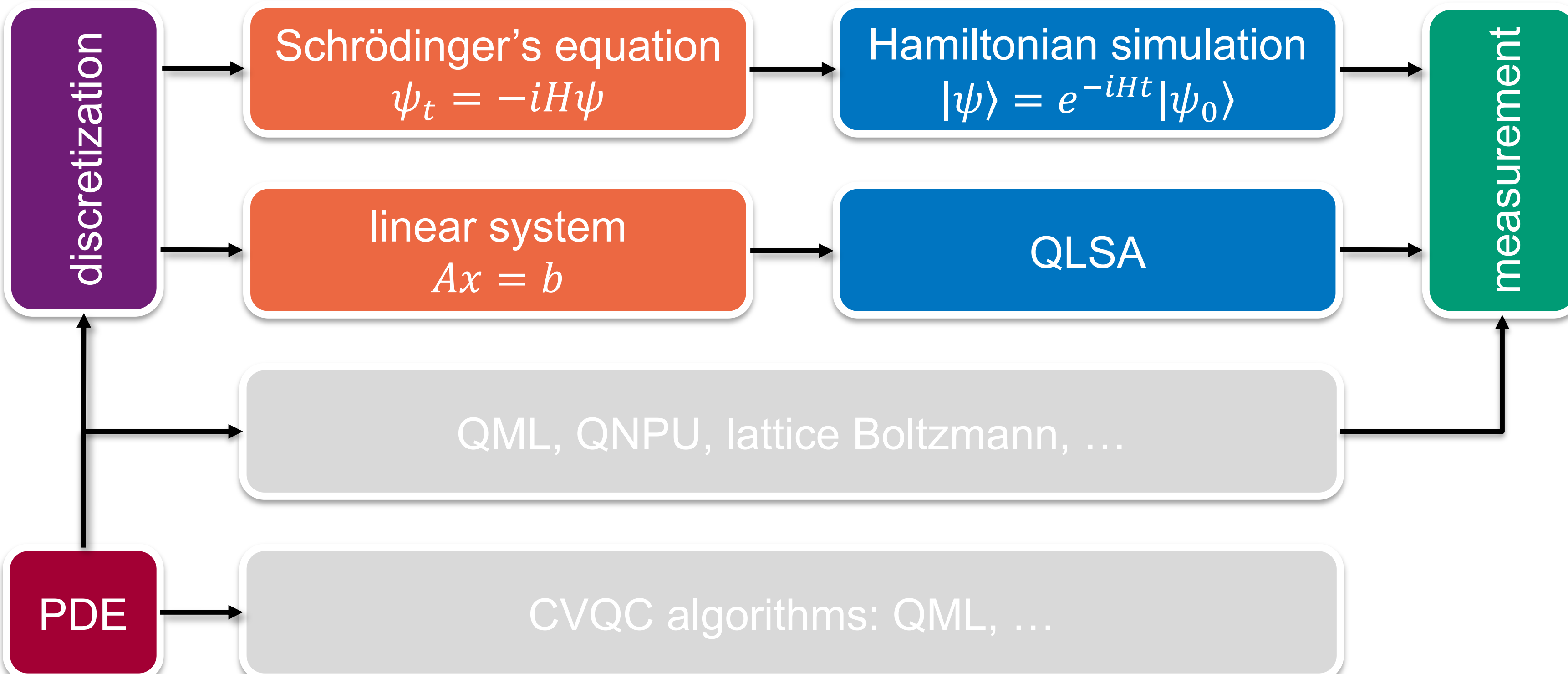
algorithm	w/o preconditioner	optimal preconditioner
Conjugate Gradients	$\tilde{\mathcal{O}}\left(\left(\frac{\lvert x\rvert_2}{\epsilon}\right)^{\frac{d+1}{2}}\right)$	$\tilde{\mathcal{O}}\left(\left(\frac{\lvert x\rvert_2}{\epsilon}\right)^{\frac{d}{2}}\right)$
Childs et al. 2017	$\tilde{\mathcal{O}}\left(\left(\frac{\lVert x\rVert_1 \lvert x\rvert_2^2}{\epsilon^3}\right)\right)$	$\tilde{\mathcal{O}}\left(\frac{\lVert x\rVert_1}{\epsilon}\right)$

[Montanaro, Pallister 2016]:

$$\tilde{\mathcal{O}}\big(h(n)\big) = \mathcal{O}\big(h(n)\log^k n\big)$$

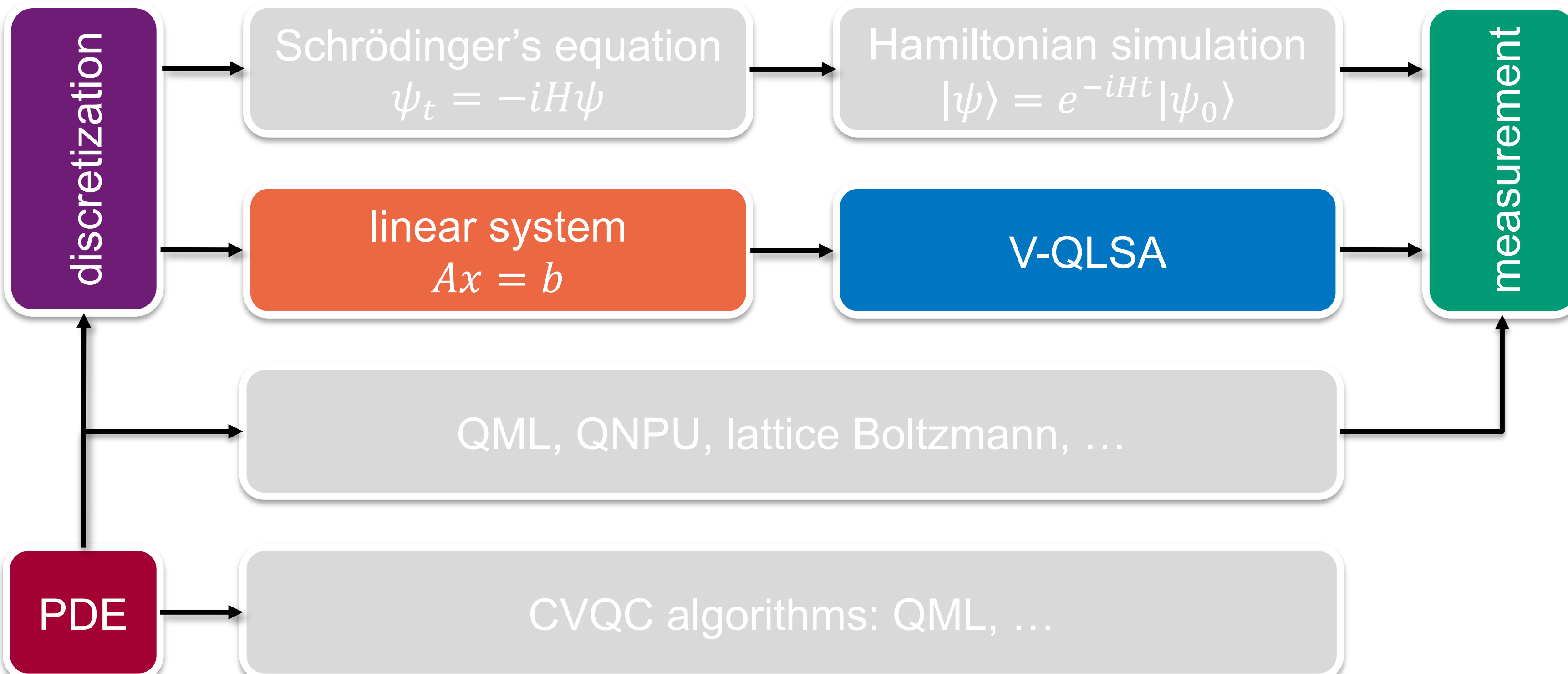
- State preparation + q-SPAI preconditioner + PA-oracle in $\mathcal{O}(\log(1/\epsilon))$
- To distinguish between two ϵ-close states requires $\mathcal{O}\left(\sqrt{1/\epsilon}\right)$ queries

Quantum algorithms for solving PDEs



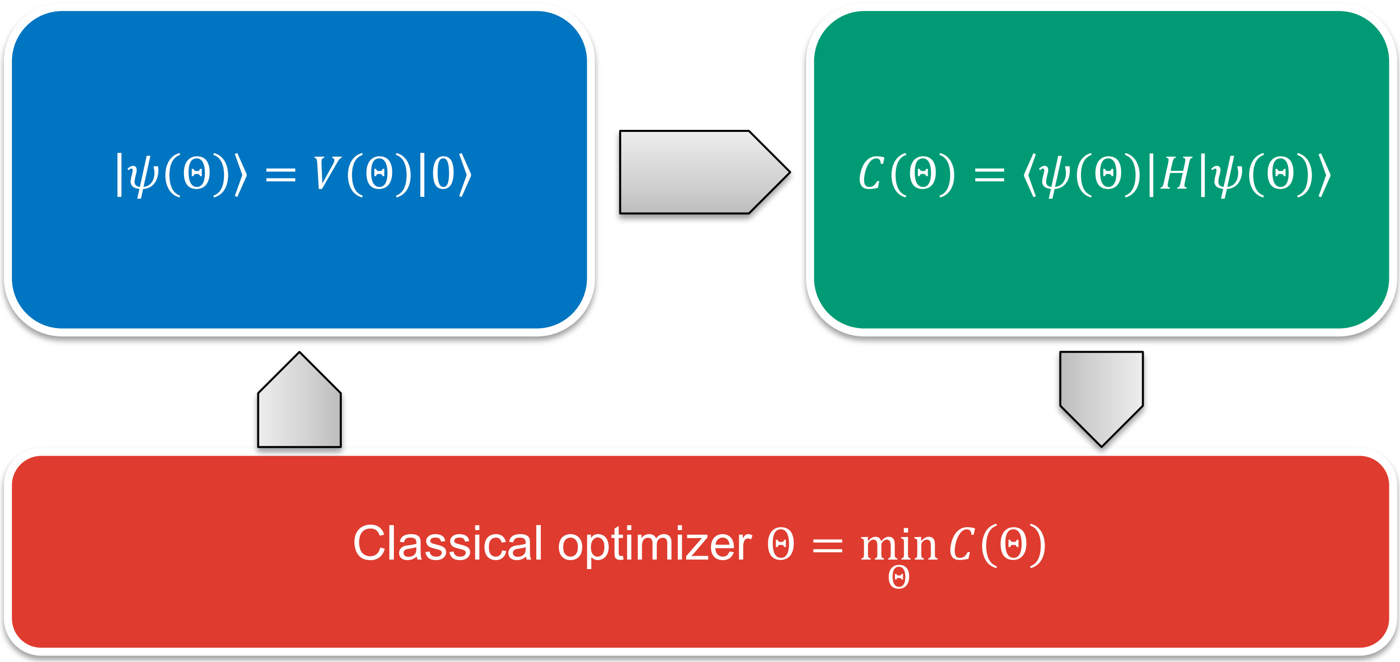
Inspired by A. Pesah's report "Quantum Algorithms for Solving Partial Differential Equations" 2020.

Quantum algorithms for solving PDEs



Inspired by A. Pesah's report "Quantum Algorithms for Solving Partial Differential Equations" 2020.

Variational quantum algorithms [Cerezo et al. 2020]



Variational quantum linear solver [Bravo-Prieto et al. 2020]

- Efficient(!) decomposition into unitaries + efficient(!) state preparation

$$A = \sum_k \alpha_k A_k \,, \qquad |b\rangle = B|0\rangle$$

- Cost function

$$\left.\begin{array}{lcll} |\Phi\rangle \perp |b\rangle & \Rightarrow & C(\Theta) & \text{large} \\ |\Phi\rangle \parallel |b\rangle & \Rightarrow & C(\Theta) & \text{small} \end{array}\right\} \qquad |\Phi\rangle = A|\psi(\Theta)\rangle$$

- Ground-state Hamiltonian

$$H = A^\dagger(\mathbb{I} - |b\rangle\langle b|)A$$

- Cost function

$$C(\Theta) = \langle\psi(\Theta)|H|\psi(\Theta)\rangle = \langle\Phi|\Phi\rangle - \langle\Phi|b\rangle\langle b|\Phi\rangle$$

Variational quantum linear solver [Bravo-Prieto et al. 2020]

- Efficient(!) decomposition into unitaries + efficient(!) state preparation

$$A = \sum_k \alpha_k A_k \,, \qquad |b\rangle = B|0\rangle$$

- Cost function

$$\left.\begin{array}{lcl} |\Phi\rangle \perp |b\rangle & \Rightarrow & C(\Theta) \;\; \text{large} \\ |\Phi\rangle \parallel |b\rangle & \Rightarrow & C(\Theta) \;\; \text{small} \end{array}\right\} \qquad |\Phi\rangle = A|\psi(\Theta)\rangle$$

- Ground-state Hamiltonian

$$H = A^\dagger(\mathbb{I} - |b\rangle\langle b|)A$$

- Normalized cost function

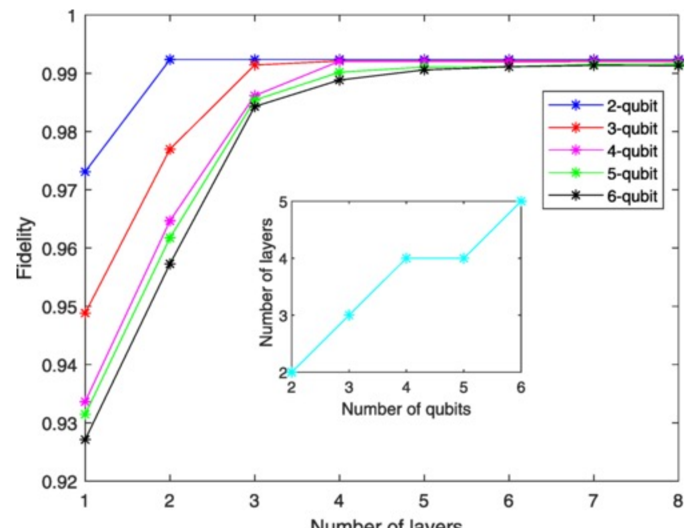
$$\hat{C}(\Theta) = 1 - \frac{|\langle\Phi|b\rangle|^2}{\langle\Phi|\Phi\rangle}$$

Variational quantum linear solver [Bravo-Prieto et al. 2020]

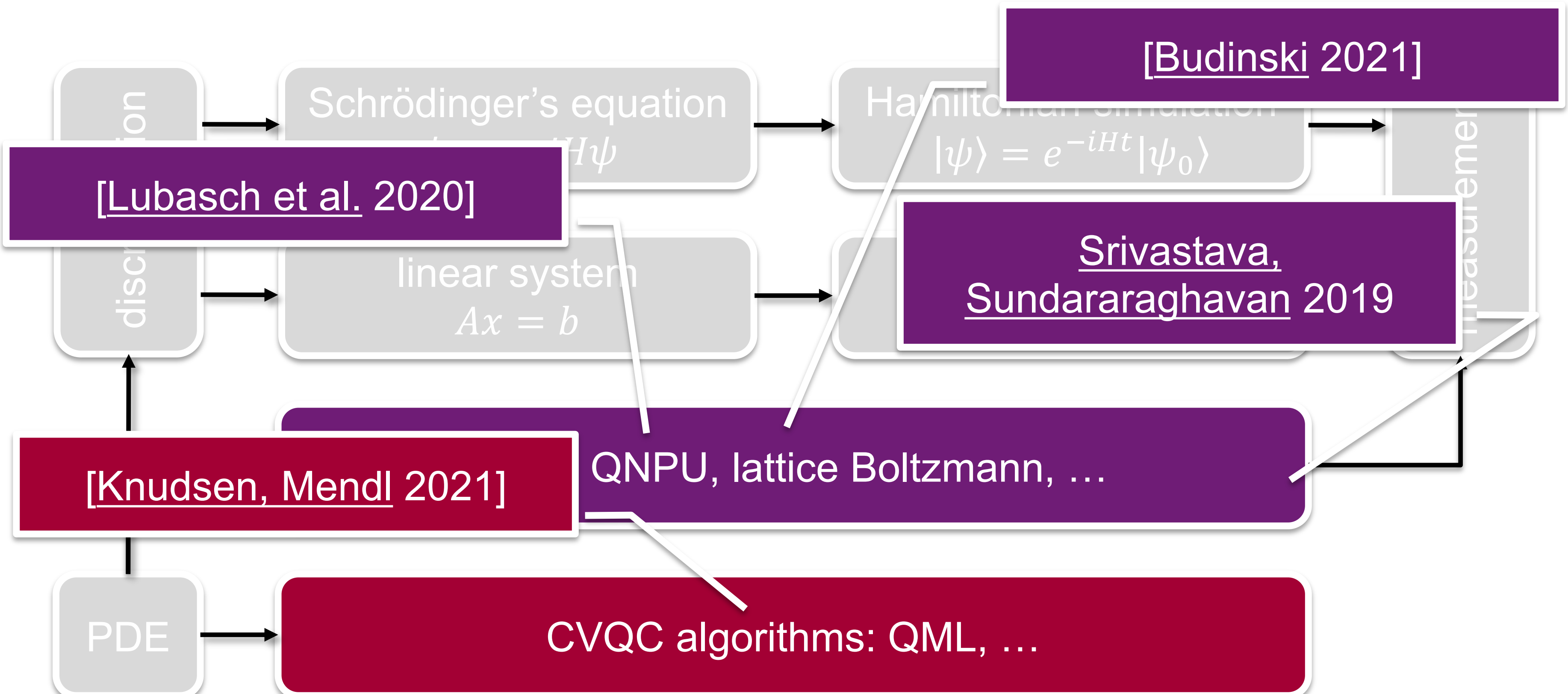
- Towards an implementable cost function

$$\langle\Phi|\Phi\rangle = \sum_{k,l} c_k^* c_l \langle 0|V^\dagger(\Theta)A_k^\dagger A_l V(\Theta)|0\rangle$$
$$\langle\Phi|b\rangle = \sum_{k,l} c_k^* c_l \langle 0|B^\dagger A_l V(\Theta)|0\rangle\langle 0|B^\dagger A_k V(\Theta)|0\rangle$$

- [Liu et al. 2021]:
  - Decomposition of the d-dimensional Poisson matrix (FDM) into $\mathcal{O}(\log N)$ terms consisting of identities and ½ spin operators $|1\rangle\langle 0|$ and $|0\rangle\langle 1|$
  - Difficulties to convergence the classical optimizer for 50-100 qubits
  - Fully connected measurement circuits



Quantum algorithms for solving PDEs



Inspired by A. Pesah's report "Quantum Algorithms for Solving Partial Differential Equations" 2020.

Summary and recommendations

- ODEs / transient PDEs (long term)
  - ‘smart’ time integrators that reduce the condition number (QLSA)
- Steady-state PDEs (near to mid term)
  - ‘smart’ discretization that reduce the condition number (QLSA)
  - problems that admit efficient matrix decompositions (V-QLSA)
- Service to QC
  - improve VQAs using classical CSE techniques