Quantum AI

A quantum computing perspective on many-body methods

Nicholas Rubin

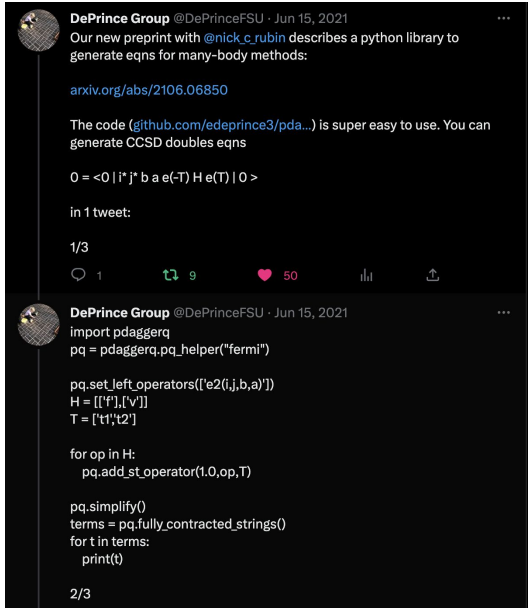
ENST “Workshop Automated tools for many-body theory”

June 7, 2023

Google

p†q: A tool for prototyping many-body methods for quantum chemistry

- automated equation and code generation for many-body quantum chemistry methods
- support for :
  - normal order with respect to the true vacuum or the fermi vacuum
  - CC-type objects (T, L, R) with up to four-body transitions
  - EE- / EA- / IP-type EOM-CC objects
  - Python einsum implementations (current) and C++ TiledArray implementations (coming soon)
  - Cavity QED EOM-CC



https://github.com/edeprince3/pdaggerq

N. C. Rubin and A. E. DePrince III, *Mol. Phys.* **119**, e1954709 (2021)

p†q: A tool for prototyping many-body methods for quantum chemistry

- automated equation and code generation for many-body quantum chemistry methods
- support for :
  - normal order with respect to the true vacuum or the fermi vacuum
  - CC-type objects (T, L, R) with up to four-body transitions
  - EE- / EA- / IP-type EOM-CC objects
  - Python einsum implementations (current) and C++ TiledArray implementations (coming soon)
  - Cavity QED EOM-CC

```python
# import pdaggerq
import pdaggerq

# import equation parser
from pdaggerq.parser import contracted_strings_to_tensor_terms

# grab pq_helper for fermio vacuum
pq = pdaggerq.pq_helper("fermi")

# ccsd doubles residual


# set bra
pq.set_left_operators([['e2(i,j,b,a)']])

# add similarity-transform hamiltonian
pq.add_st_operator(1.0,['f'],['t1', 't2'])
pq.add_st_operator(1.0,['v'],['t1', 't2'])

pq.simplify()

# grab list of fully-contracted strings, then print
terms = pq.fully_contracted_strings()
for my_term in terms:
    print(my_term)

# python code
terms = contracted_strings_to_tensor_terms(terms)
for my_term in terms:
    print("#\t", my_term)
    print("%s" % (my_term.einsum_string(update_val='doubles_res',
                                        output_variables=('a', 'b', 'i', 'j'))))
    print()


pq.clear()
```

```
(16:36 ~/) python ccsd.py
['-1.00000000000000', 'P(i,j)', 'f(k,j)', 't2(a,b,i,k)']
['+1.00000000000000', 'P(a,b)', 'f(a,c)', 't2(c,b,i,j)']
['-1.00000000000000', 'P(i,j)', 'f(k,c)', 't1(c,j)', 't2(a,b,i,k)']
['-1.00000000000000', 'P(a,b)', 'f(k,c)', 't1(a,k)', 't2(c,b,i,j)']
['+1.00000000000000', '<a,b||i,j>']

...

#            -1.0000 P(i,j)f(k,j)*t2(a,b,i,k)
contracted_intermediate = -1.000000000000000 * einsum('kj,abik->abij', f[o, o], t2)
doubles_res +=  1.00000 * contracted_intermediate + -1.00000 * einsum('abij->abji', contracted_intermediate)

#             1.0000 P(a,b)f(a,c)*t2(c,b,i,j)
contracted_intermediate =  1.000000000000000 * einsum('ac,cbij->abij', f[v, v], t2)
doubles_res +=  1.00000 * contracted_intermediate + -1.00000 * einsum('abij->baij', contracted_intermediate)

#            -1.0000 P(i,j)f(k,c)*t1(c,j)*t2(a,b,i,k)
contracted_intermediate = -1.000000000000000 * einsum('kc,cj,abik->abij', f[o, v], t1, t2, optimize=['einsum_path', (0, 1), (0, 1)])
doubles_res +=  1.00000 * contracted_intermediate + -1.00000 * einsum('abij->abji', contracted_intermediate)

#            -1.0000 P(a,b)f(k,c)*t1(a,k)*t2(c,b,i,j)
contracted_intermediate = -1.000000000000000 * einsum('kc,ak,cbij->abij', f[o, v], t1, t2, optimize=['einsum_path', (0, 1), (0, 1)])
doubles_res +=  1.00000 * contracted_intermediate + -1.00000 * einsum('abij->baij', contracted_intermediate)

#             1.0000 <a,b||i,j>
doubles_res +=  1.000000000000000 * einsum('abij->abij', g[v, v, o, o])

...
```

https://github.com/edeprince3/pdaggerq

N. C. Rubin and A. E. DePrince III, *Mol. Phys.* **119**, e1954709 (2021)

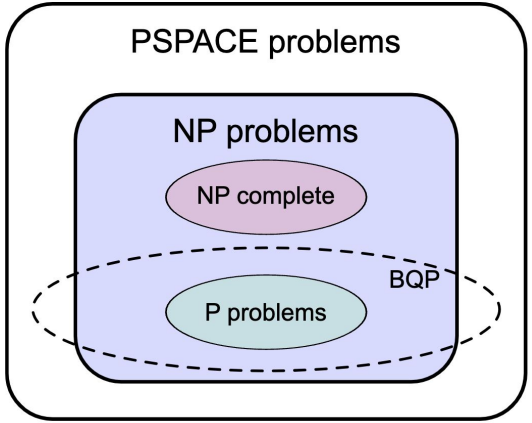
Qubits and gates, briefly

Any 2-state quantum system is a qubit, $|\psi\rangle = a_0 |0\rangle + a_1 |1\rangle$

For 2 qubits, $|\psi\rangle = a_{00} |00\rangle + a_{01} |01\rangle + a_{10} |10\rangle + a_{11} |11\rangle$

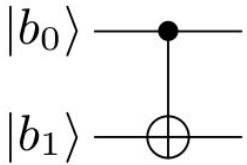
N qubit systems requires $O(2^N)$ classical bits to represent

Information manipulated by controlled Hamiltonian evolutions



For instance, evolve 2 qubits under $H = (Z_0 - I_0)\otimes(I_1 - X_1)$ for time, $t = \pi/4$

$$e^{-iHt} : |b_0\rangle |b_1\rangle \mapsto |b_0\rangle |b_0 \oplus b_1\rangle$$

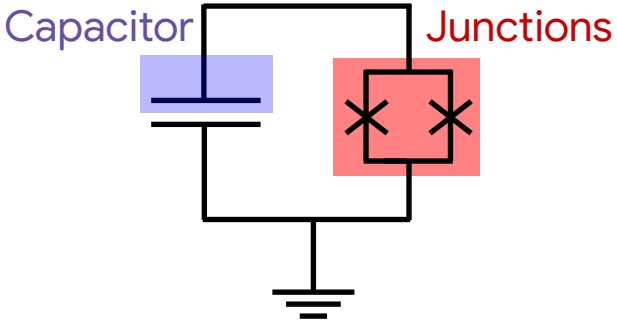


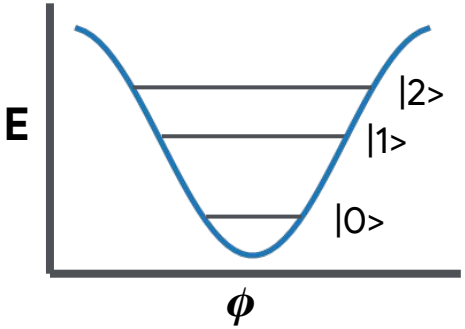
CNOT + single qubit rotations “universal” for all quantum dynamics / circuits

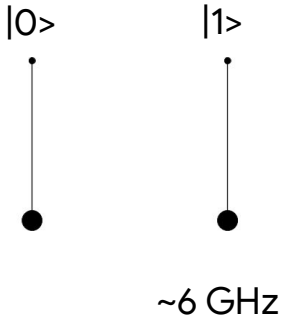
Quantum AI

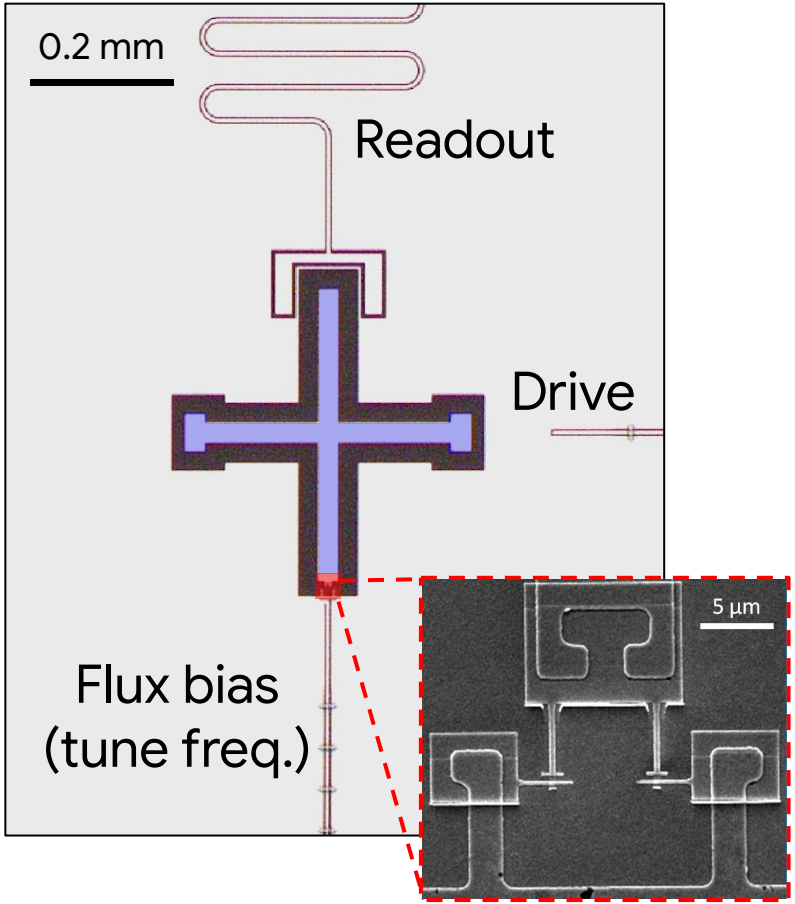
“Transmon” qubit

Nonlinear superconducting oscillator

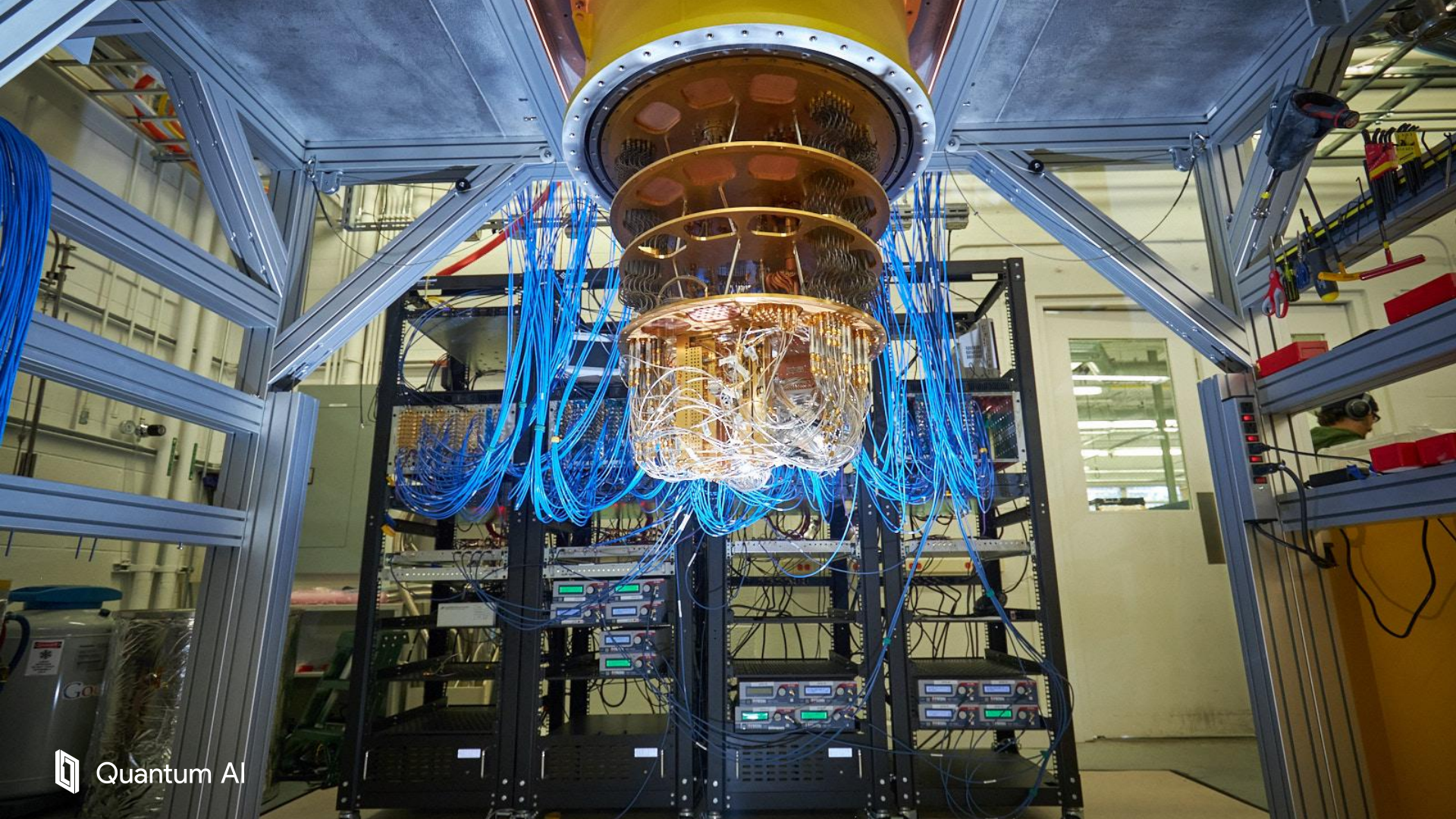




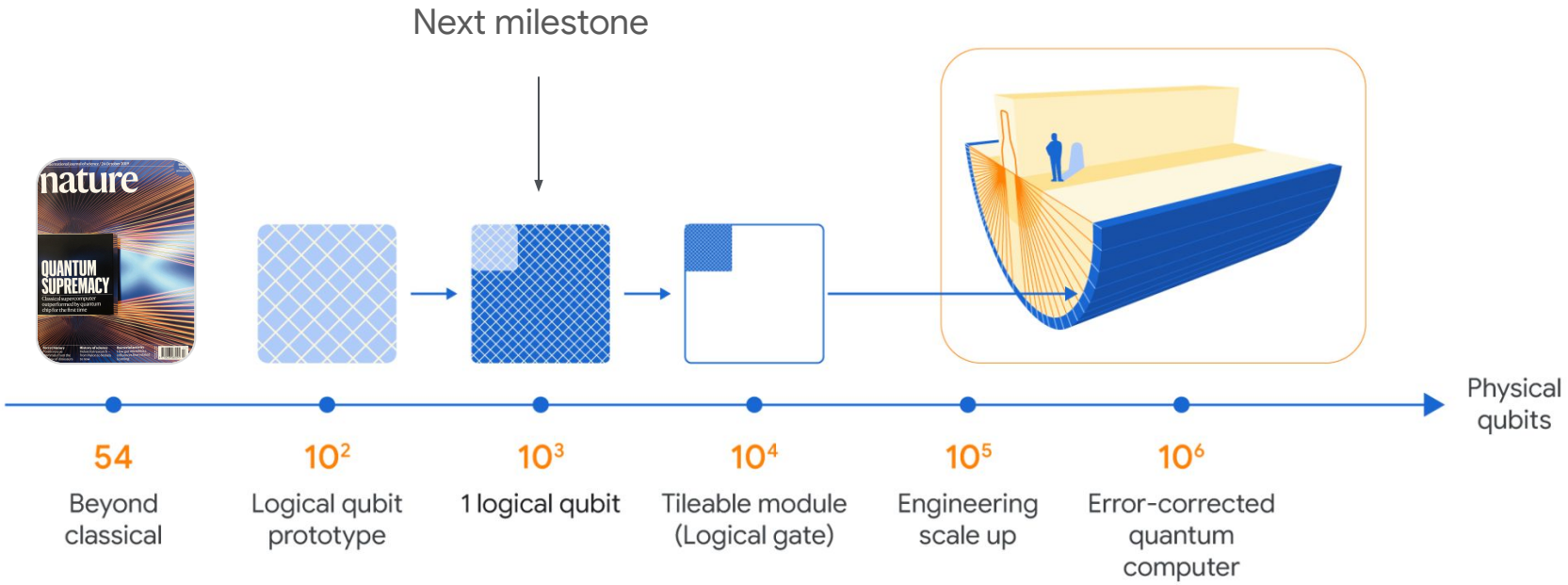




Quantum AI


Quantum AI

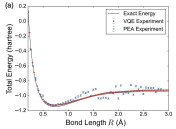
Google's roadmap to fault-tolerant quantum computing



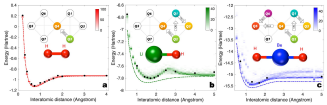
Are we on a path to qchem simulation advantage?

We are in the age of noisy intermediate scale (NISQ) quantum devices
We can run circuits on ~50 qubits but errors severely limit circuit size

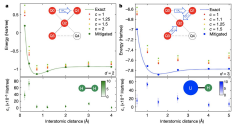
2014



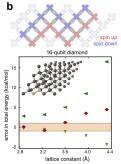
Colless, ..., McClean et al. (2017)



Hempel et al. (2018)



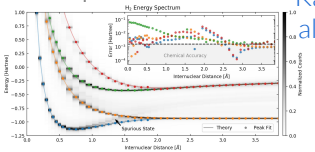
Arute et al. (2020)



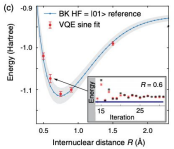
O'Brien et al. (2022),

Now

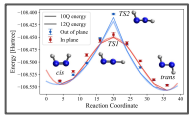
O'Malley, Babbush, et al. (2016)



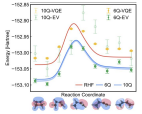
Kandala et al. (2017),



Kandala et al. (2019),



Huggins et al. (2021),

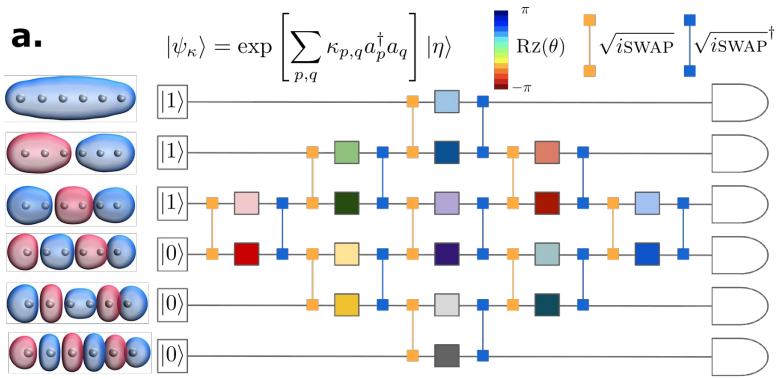


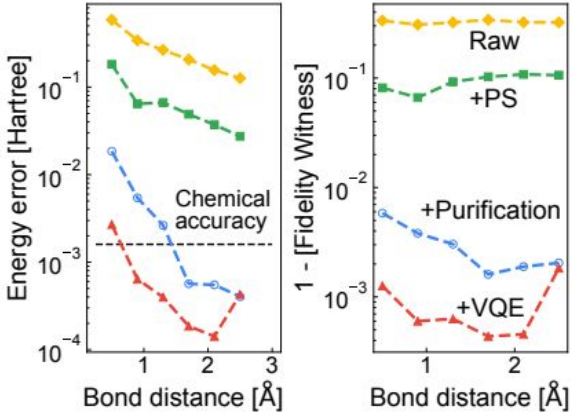
Experimental explorations are alive and well
Experiments are a playground for validating error mitigation and provable results

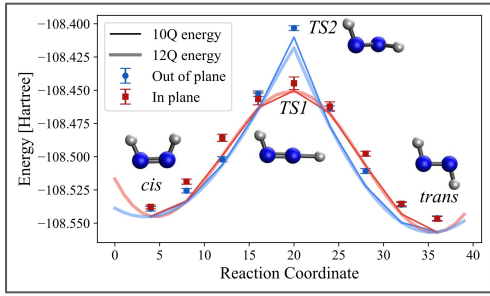
Google AI Quantum

Hartree-Fock on a superconducting qubit quantum computer

- Largest variational quantum algorithm to date
- Primitive for quantum computing algorithms (Givens rotation)
- Non-interacting fermions:
  - *Classically simulable :(*
  - *Free gradient estimation :)*
  - *McWeeny purification :)*







Quantum AI

Science 369 (6507), 1084-1089, 2020

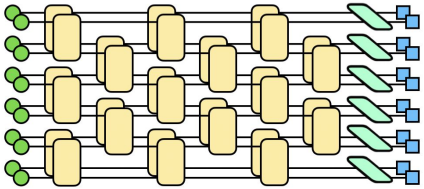
Rigorous results in NISQ

Error mitigation

Doubling in space: Virtual distillation

Phys. Rev. X 11, 041036 (2021)

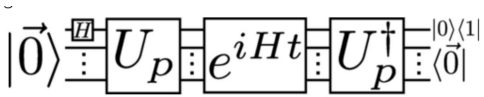
Circuit reps ∝ $1/f^4$



Doubling in time: Verified phase estimation

PRX Quantum **2**, 020317 (2021)

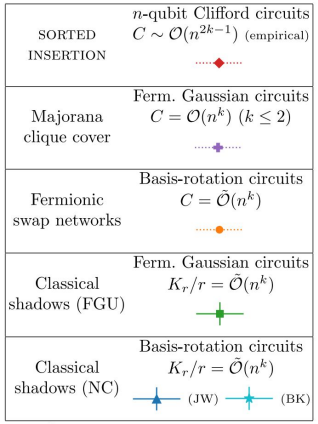
Circuit reps ∝ $1/f^2$



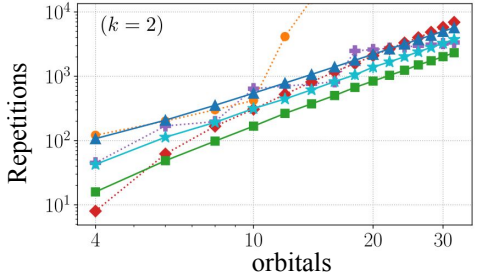
Measurement complexity

Optimal deterministic measurement complexity 1-RDM, 2-RDM, k-RDM

Science 369 1084-1089 (2021), Phys. Rev. X **10**, 031064 (2020), *Quantum*, *4*, 276.



$$M = \mathcal{O}\left[\binom{n}{k}\frac{k^{3/2}\log n}{\varepsilon^2}\right]$$

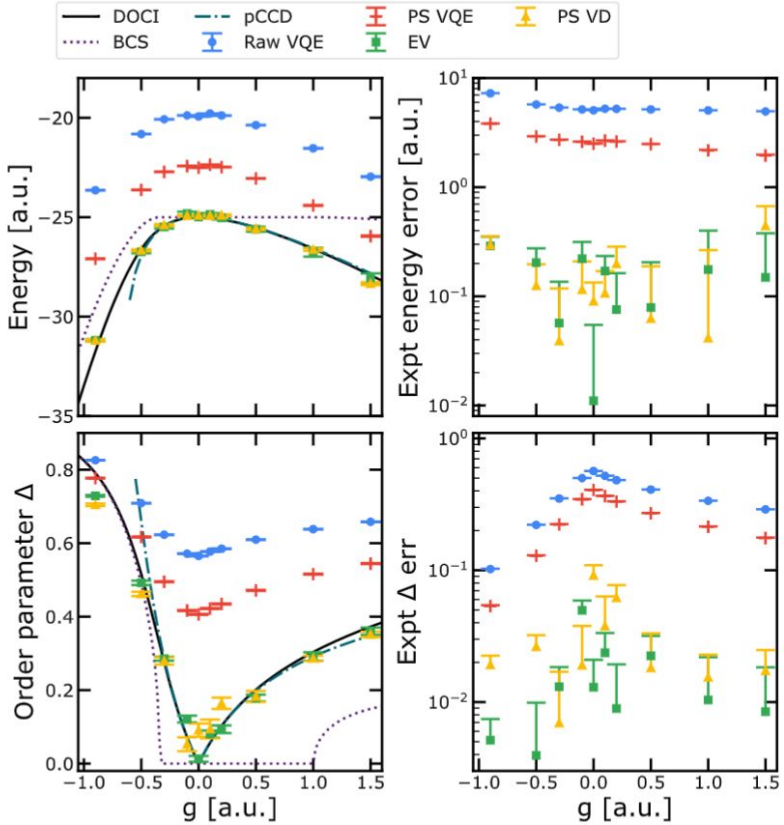


Quantum AI

The Richardson-Gaudin model (10 sites)

$$H = \sum_{j,\alpha} j n_{j,\alpha} + g \sum_{j,k} c^\dagger_{j\uparrow} c^\dagger_{j\downarrow} c_{k\downarrow} c_{k\uparrow}$$

- Qualitatively accurate for Δ (superconducting order parameter)
- EV ~ PS-VD in performance.
- Mean improvement 50x over Raw VQE
- Error stable across range of g values, unlike classical methods (pccd, BCS).



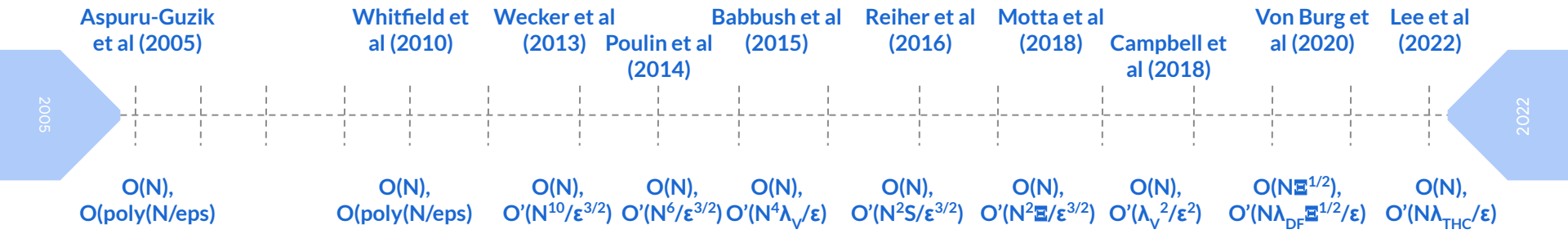
Quantum AI

Summary of NISQ Progress

- VQE is very popular but progress in chemistry has been slow and the approach is very difficult to scale
- The number of measurements required quickly becomes problematically large
- Most compelling ansatz require circuits far too deep
- Theoretical basis for error mitigation can though needs to be verified in practice
- Different strategies (QC-QMC) are needed [Lee et al. Nature 603 (2022)]

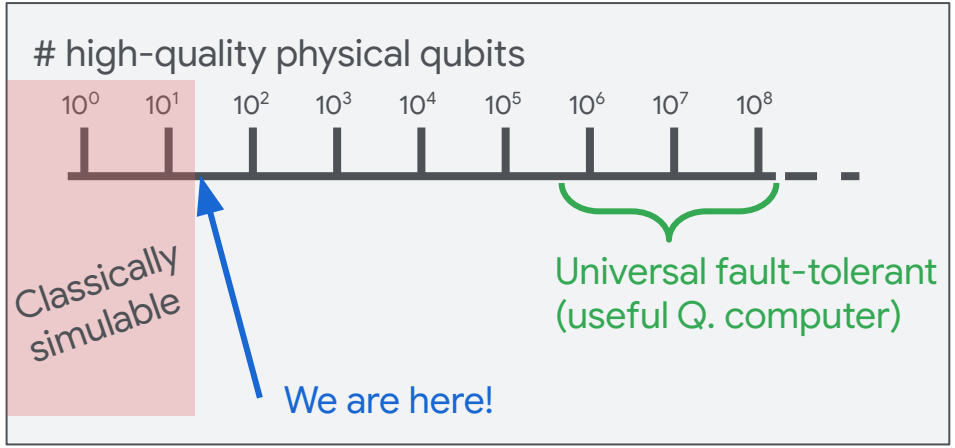
Molecule	# Qubits	Year
H_2	2	2014
H_2O	5	2020
BeH_2	6	2017
N_2H_2	10*	2020
H_{12}	12*	2020

Are we on a path to qchem simulation advantage?



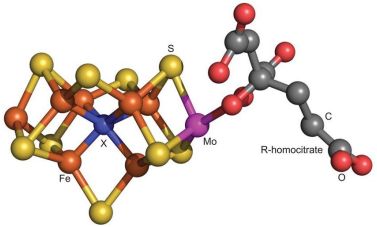
Theoretical progress in QPE going strong
Scaling in chemically relevant parameters are now close to optimal

Time for an honest assessment of progress?



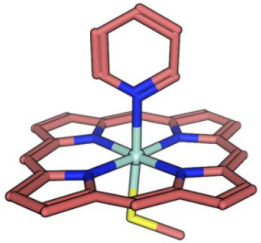
Quantum AI

Quantum simulation advantage?



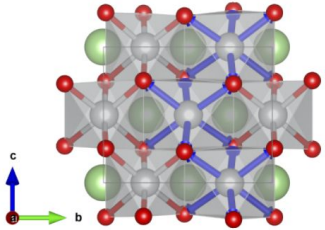
FeMoCo (fertilizer catalyst)
PRX Quantum 2 (3), 030305

4 M qubits
4 Days of runtime



P450 (drug anti-target)
PNAS 119 (38), e2203533119

4.6 M qubits
3 Days of runtime



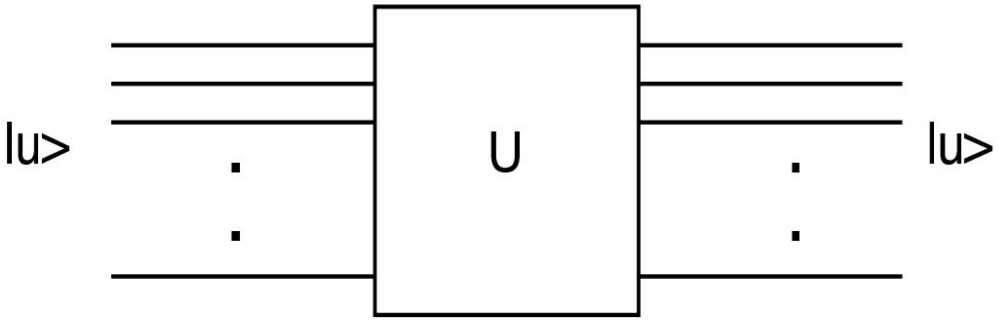
$LiNiO_2$ (battery cathode)
arXiv:2302.05531

? M qubits
? Days of runtime

Simulating Quantum Mechanics on a Quantum Computer

Phase Estimation Algorithm

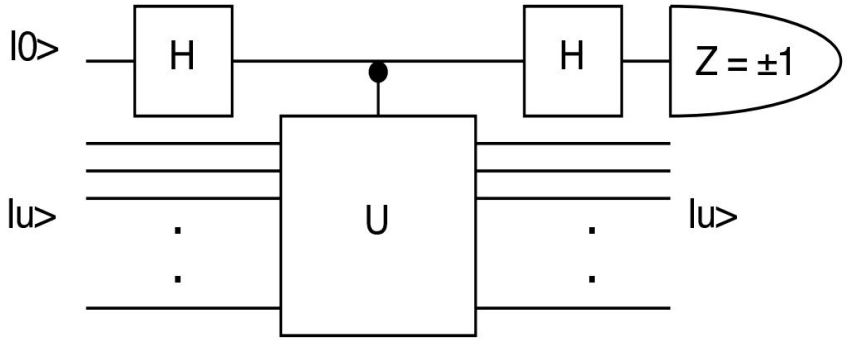
Time-evolution of eigenstate encodes the eigenvalues in phase information



Simulating Quantum Mechanics on a Quantum Computer

Phase Estimation Algorithm

Time-evolution of eigenstate encodes the eigenvalues in phase information



$$\langle \hat{Z} \rangle = \langle u|(\hat{U} + \hat{U}^\dagger)|u\rangle/2 = \frac{1}{2}\left(e^{2\pi i\phi} + e^{-2\pi i\phi}\right) = \cos(2\pi\phi)$$

Error-corrected quantum chemistry simulation

Science 309:5741 (2005), 1704-1707

1. Prepare an ansatz wavefunction ψ with "reasonable" support on the ground state

$$H\,|k\rangle = E_k\,|k\rangle \qquad\qquad |\langle\psi|0\rangle|^2 = \text{not-too-small}$$

2. Form quantum circuit $U = e^{-i\,f(H)}$ that encodes Hamiltonian spectrum in its eigenvalues

e.g., for Trotter:

$$f(H) = H = \sum_\ell H_\ell \qquad\qquad U \approx \left(\prod_\ell e^{-iH_\ell/r}\right)^r$$

3. Application of U to ψ accumulates phases $f(E)$ encoding the spectrum

$$U\,|\psi\rangle = \sum_k \underbrace{\langle k|\psi\rangle}_{a_k} e^{-if(E_k)}\,|k\rangle$$

4. Phase estimation gives E_0 *with error* ϵ and probability $|a_0|^2$ using $\frac{1}{\epsilon}\left\|\frac{\partial f(E)}{\partial E}\right\|^{-1}$ queries to U

Error-corrected quantum chemistry simulation

Science 309:5741 (2005), 1704-1707, PRX 8, 041015 (2018), PRL 121, 010501 (2018)

1. Prepare an ansatz wavefunction ψ with "reasonable" support on the ground state

$$H\,|k\rangle = E_k\,|k\rangle \qquad\qquad |\langle\psi|0\rangle|^2 = \text{not-too-small}$$

2. Form quantum circuit $U = e^{-i\,f(H)}$ that encodes Hamiltonian spectrum in its eigenvalues

e.g., for **Qubitization**:

$$f(H) = \arccos(H/\lambda) \qquad\qquad U = e^{i\arccos(H/\lambda)}$$

3. Application of U to ψ accumulates phases $f(E)$ encoding the spectrum

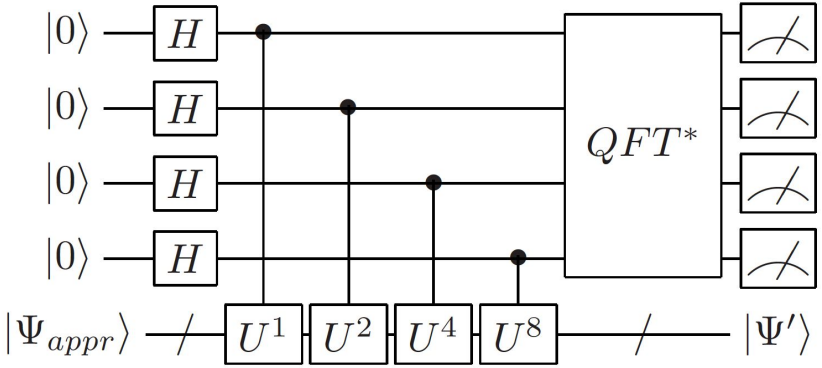
$$U\,|\psi\rangle = \sum_k \underbrace{\langle k|\psi\rangle}_{a_k} e^{-if(E_k)}\,|k\rangle$$

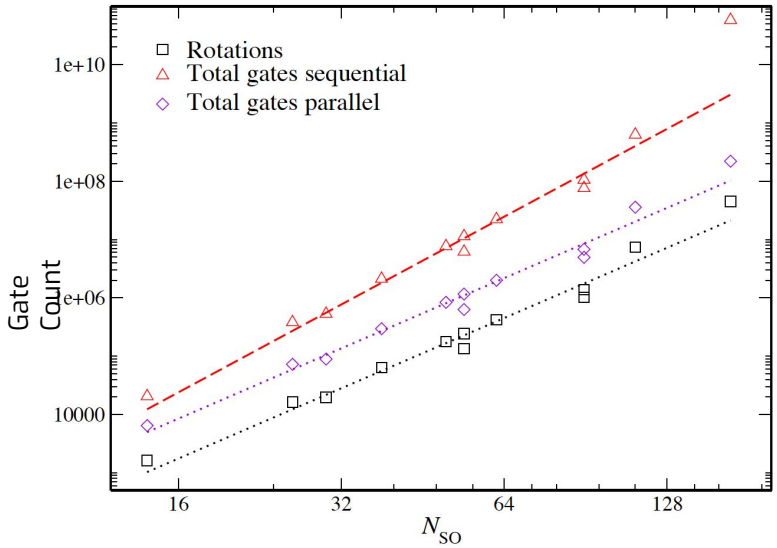
4. Phase estimation gives E_0 *with error* ϵ and probability $|a_0|^2$ using $\mathcal{O}\left(\frac{\lambda}{\epsilon}\right)$ queries to U

Simulating Quantum Mechanics on a Quantum Computer

Phase Estimation Algorithm

Time-evolution of eigenstate encodes the eigenvalues in phase information





$$T = \frac{\pi}{\epsilon} \approx 4000 E_h^{-1}$$

Very long coherence times required for eigenstate preparation

Wecker, Bauer, et. Al. Phys. Rev. A 90 022305 (2014), Whitfield, Biamonte, Aspuru-Guzik. Mol. Phys. 109 735 (2010)

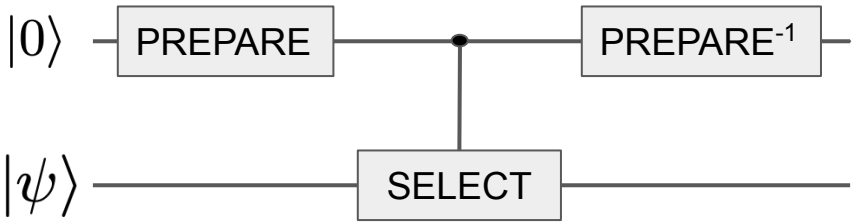
Block encoding via LCU

$$H = \sum_{\ell=1}^{L} \omega_\ell U_\ell \qquad \lambda = \sum_{\ell=1}^{L} |\omega_\ell|$$

A linear combinations of unitaries representation

$$\text{SELECT}\,|\ell\rangle\,|\psi\rangle \mapsto |\ell\rangle\,U_\ell\,|\psi\rangle$$

$$\text{PREPARE}\,|0\rangle^{\otimes \log L} \mapsto \sum_{\ell=0}^{L-1} \sqrt{\frac{w_\ell}{\lambda}}\,|\ell\rangle$$



Quantum AI

Block encoding H

$$U = \begin{bmatrix} \mathcal{H} & \sqrt{I-\mathcal{H}^2} \\ \sqrt{I-\mathcal{H}^2} & -\mathcal{H} \end{bmatrix}$$

$$= Z \otimes \mathcal{H} + X \otimes \sqrt{I-\mathcal{H}^2}$$

2D subspaces

$$\mathcal{H} = \sum_\lambda \lambda |\lambda\rangle\langle\lambda|$$

$$U|0\rangle|\lambda\rangle = \lambda|0\rangle|\lambda\rangle + \sqrt{1-\lambda^2}|1\rangle|\lambda\rangle$$

$$U|1\rangle|\lambda\rangle = -\lambda|1\rangle|\lambda\rangle + \sqrt{1-\lambda^2}|0\rangle|\lambda\rangle$$

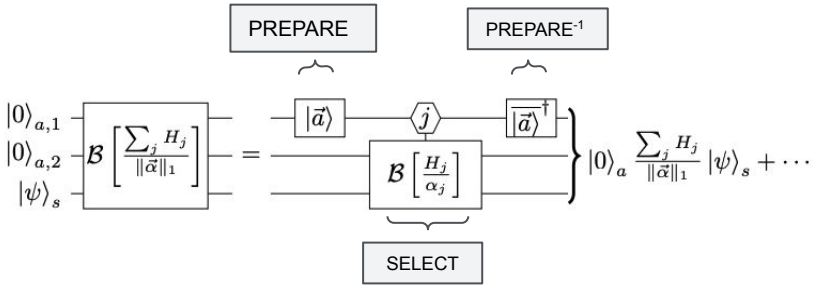
$$U = e^{i\cos^{-1}(\mathcal{H})}$$

block encodings arithmetic

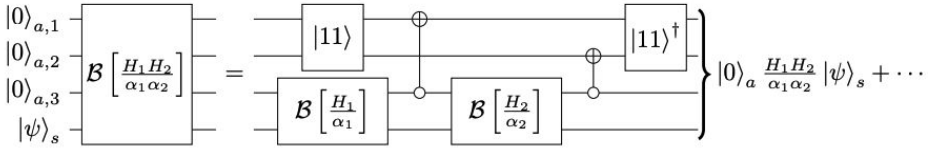
$$\mathcal{B}[H/\alpha] = \begin{pmatrix} H/\alpha & \cdots \\ \vdots & \ddots \end{pmatrix}$$

Given block encodings for $H_1, H_2, \ldots, H_m$ basic block encoding arithmetic allows you to generate a unitary encoding for any H in the algebra generated by $H_1, H_2, \ldots, H_m$

Adding two block encodings



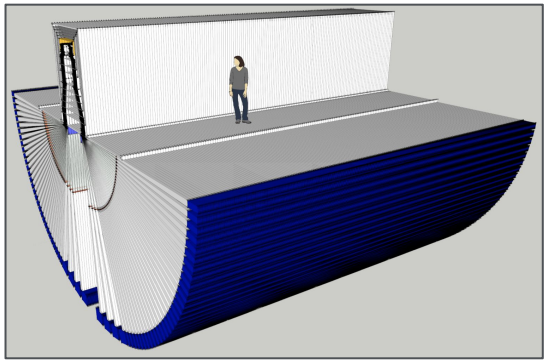
Multiplying two block encodings



Phys. Rev. Research **3**, 033055

Space-time considerations are different within error-correction

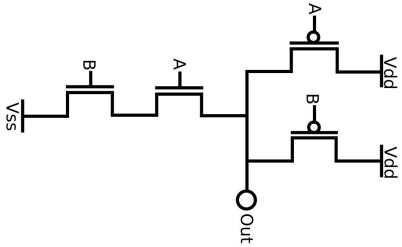
Quantum AI



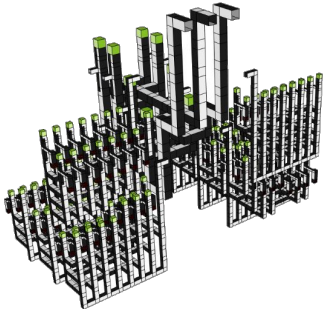
What we have in mind is a device with >99.95% fidelity physical gates and ~1MM physical qubits

~1k "physical qubits" per "logical qubit"

- Certain necessary gates (e.g. Toffolis) are very slow, require hundreds of *logical qubits*
- Gives a huge "constant factor slowdown" in spacetime versus classical computers



classical NAND gate (CMOS)
$<10^{-9}$ "transistorseconds"

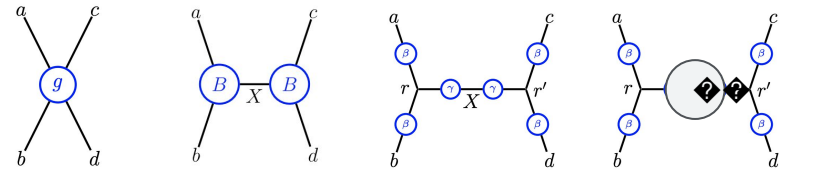


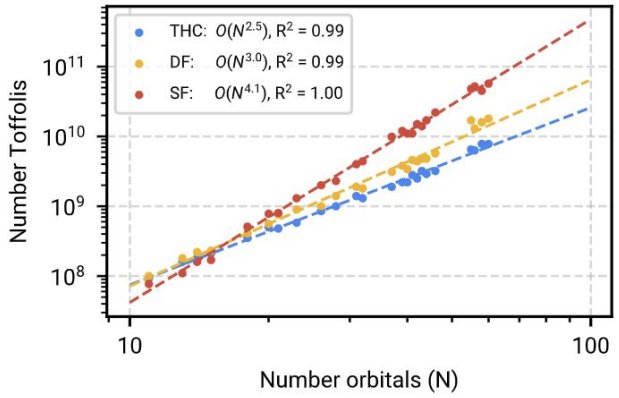
"quantum NAND" gate
(distillation of Toffoli state)
>10 "qubitseconds"

Cost of qubitization: block encodings

The complexity of building SELECT and PREPARE largely depends on the Hamiltonian factorization we use. The Coulomb kernel can be factorized in many different ways

$$H = \sum_{abcd} h_{abcd} a_a^\dagger a_b^\dagger a_d a_c = \sum_{\ell=0}^{L-1} \omega_\ell U_\ell$$





Qubits	$\tilde{O}(N^2)$	$\tilde{O}(N^{3/2})$	$\tilde{O}(N\sqrt{\Xi})$	$\tilde{O}(N)$
Toffoli	$\tilde{O}(N^2\lambda/\epsilon)$	$\tilde{O}(N^{3/2}\lambda/\epsilon)$	$\tilde{O}(N\lambda\sqrt{\Xi}/\epsilon)$	$\tilde{O}(N\lambda/\epsilon)$
Qubits	$\tilde{O}(N_k^{3/2}N^2)$	$\tilde{O}(N_k N^{3/2})$	$\tilde{O}(\sqrt{N_k}N\sqrt{\Xi})$	$\tilde{O}(N_k N)$
Toffoli	$\tilde{O}(N_k^{3/2}N^2\lambda_{\text{sparse}}/\epsilon)$	$\tilde{O}(N_k N^{3/2}\lambda_{\text{SF}}/\epsilon)$	$\tilde{O}(\sqrt{N_k}N\sqrt{\Xi}\lambda_{\text{DF}}/\epsilon)$	$\tilde{O}(N_k N\lambda_{\text{DF}}/\epsilon)$

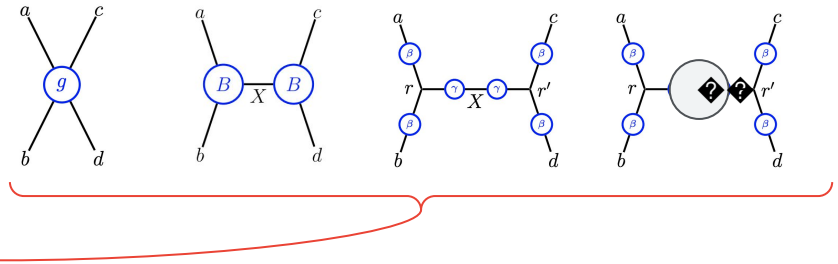
Google

Goings, ..., Rubin *PNAS* 119 (38) **2022** , Rubin et al. arXiv:2302:05521

Cost of qubitization: block encodings

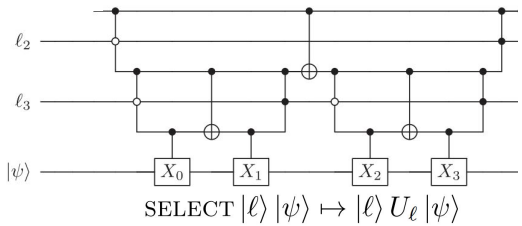
The complexity of building SELECT and PREPARE largely depends on the Hamiltonian factorization we use. The Coulomb kernel can be factorized in many different ways

$$H = \sum_{abcd} h_{abcd} a_a^\dagger a_b^\dagger a_d a_c = \sum_{\ell=0}^{L-1} \omega_\ell U_\ell$$

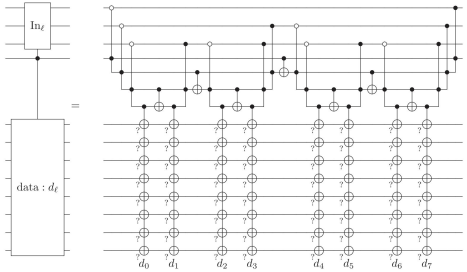


Important primitives to construct block encodings

For loop on a seg-tree **O(L) Toffoli**



Q-ROM **O(L) Toffoli + log(L) ancilla or O((Lβ)$^{0.5}$) Toffoli + O((Lβ)$^{0.5}$) ancilla**



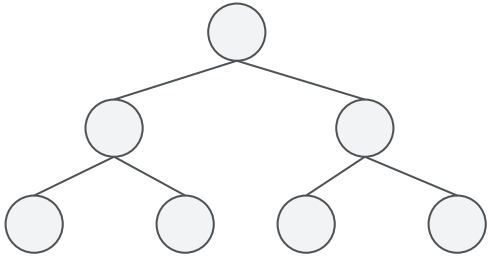
State preparation: **PREPARE O(L$^{1/2}$) cost**

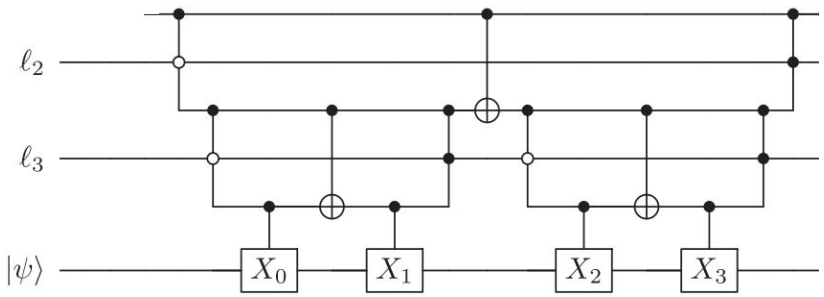
$$\sqrt{\frac{1}{L}} \sum_{\ell=0}^{L-1} |\ell\rangle |0\rangle \xrightarrow{\text{QROM}} \sum_{\ell=0}^{L-1} \sqrt{\frac{\omega_\ell}{\lambda}} |\ell\rangle |\mathrm{junk}_\ell\rangle$$

Interlude: Circuit gadgets: for loop

Main idea: perform loop by doing depth-first-search on a binary tree

$$\ell = b_{\ell p-1} \ldots b_{\ell 0}$$





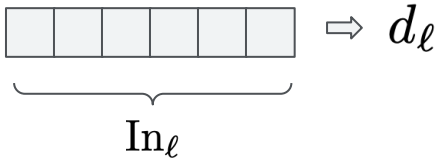
$$\text{SELECT}\ |\ell\rangle\ |\psi\rangle \mapsto |\ell\rangle\ U_\ell\ |\psi\rangle$$

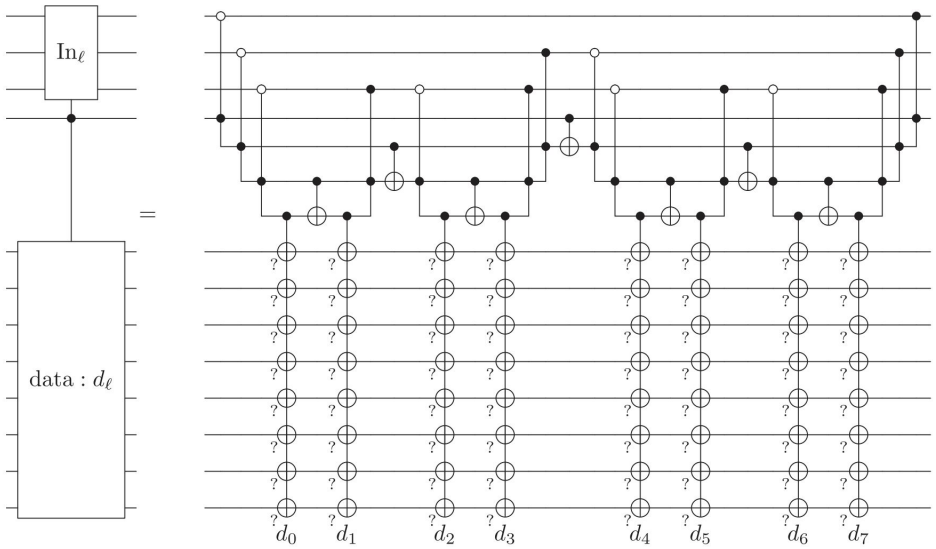
O(L) Toffoli complexity to construct tree and O(log(L)) qubits for control

Interlude: Circuit gadgets: Data lookup + multiplex

Main idea: read-only-memory circuit using unary iteration

$$D|\mathrm{In}_\ell\rangle|0_0 \ldots 0_d\rangle = |\mathrm{In}_\ell\rangle|d_\ell\rangle$$



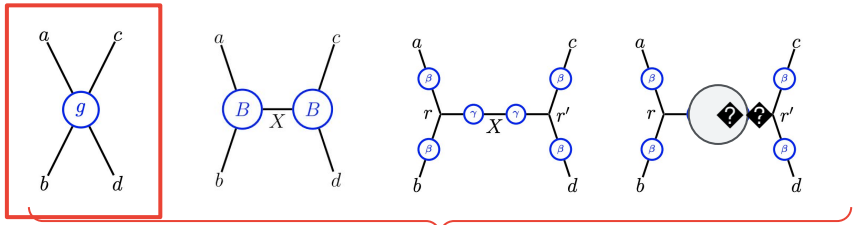


O(L) Toffoli + log(L) ancilla
or
$O((L\beta)^{0.5})$ Toffoli + $O((L\beta)^{0.5})$ ancilla

Cost of qubitization: block encodings

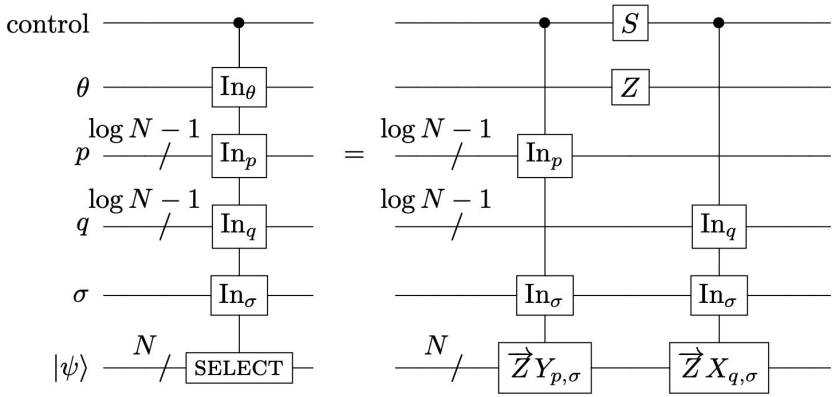
The complexity of building SELECT and PREPARE largely depends on the Hamiltonian factorization we use. The Coulomb kernel can be factorized in many different ways

$$H = \sum_{abcd} h_{abcd} a_a^\dagger a_b^\dagger a_d a_c = \sum_{\ell=0}^{L-1} \omega_\ell U_\ell$$



$$V' = \frac{1}{8} \sum_{\alpha,\beta\in\{\uparrow,\downarrow\}} \sum_{p,q,r,s=1}^{N/2} V_{pqrs} Q_{pq\alpha} Q_{rs\beta}$$

$$Q_{pq\sigma} = \begin{cases} X_{p,\sigma}\vec{Z}X_{q,\sigma}, & p < q, \\ Y_{p,\sigma}\vec{Z}Y_{q,\sigma}, & p > q, \\ -Z_{p,\sigma}, & p = q. \end{cases}$$



O(N)

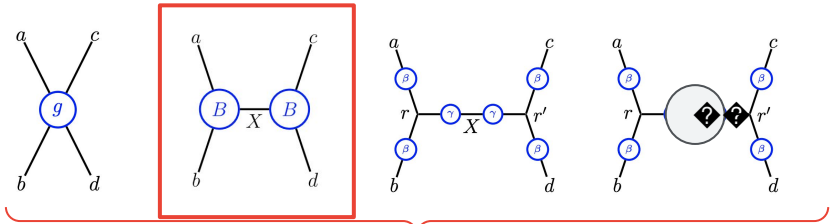
PRX Quantum **2**, 030305 (2021)

Google

Cost of qubitization: block encodings

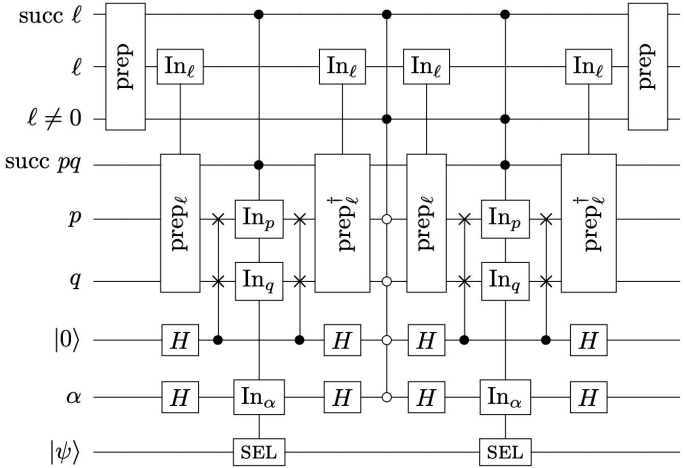
The complexity of building SELECT and PREPARE largely depends on the Hamiltonian factorization we use. The Coulomb kernel can be factorized in many different ways

$$H = \sum_{abcd} h_{abcd} a_a^\dagger a_b^\dagger a_d a_c = \sum_{\ell=0}^{L-1} \omega_\ell U_\ell$$



$$\frac{1}{8}\sum_{\ell=1}^{L}\left(\sum_{\sigma\in\{\uparrow,\downarrow\}}\sum_{p,q=1}^{N/2} W_{pq}^{(\ell)} Q_{pq\sigma}\right)^2$$

$$Q_{pq\sigma} = \begin{cases} X_{p,\sigma}\vec{Z}X_{q,\sigma}, & p<q, \\ Y_{p,\sigma}\vec{Z}Y_{q,\sigma}, & p>q, \\ -Z_{p,\sigma}, & p=q. \end{cases}$$



O(N^{3/2})

O(N)

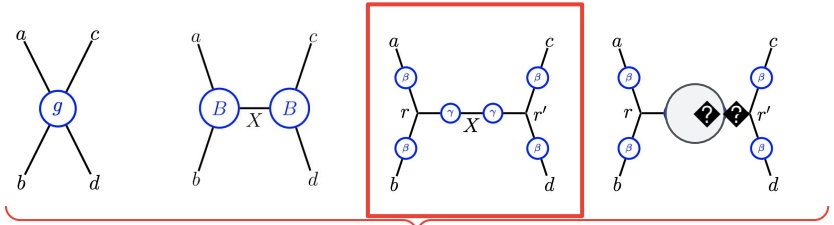
PRX Quantum **2**, 030305 (2021)

Google

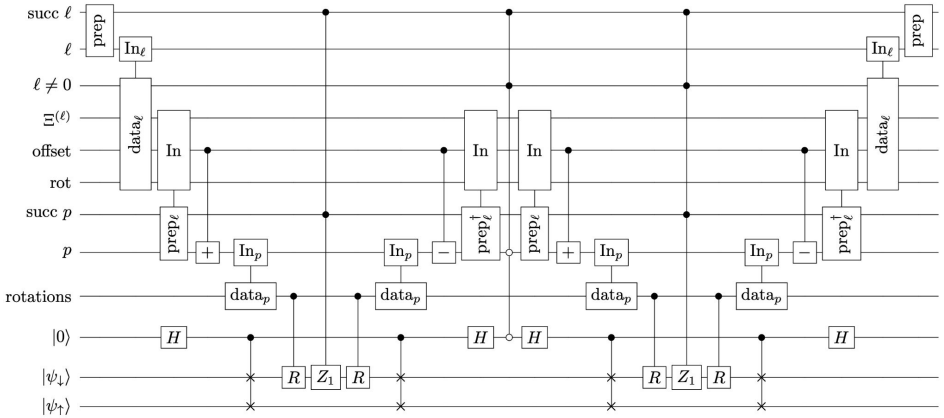
Cost of qubitization: block encodings

The complexity of building SELECT and PREPARE largely depends on the Hamiltonian factorization we use. The Coulomb kernel can be factorized in many different ways

$$H = \sum_{abcd} h_{abcd} a_a^\dagger a_b^\dagger a_d a_c = \sum_{\ell=0}^{L-1} \omega_\ell U_\ell$$



$$\frac{1}{8} \sum_{\ell=1}^{L} U_\ell \left(\sum_{\sigma \in \{\uparrow,\downarrow\}} \sum_{p=1}^{N/2} f_p^{(\ell)} \left(\mathbb{1} - Z_{p,\sigma} \right) \right)^2 U_\ell^\dagger$$



$O(N\Xi^{1/2})$

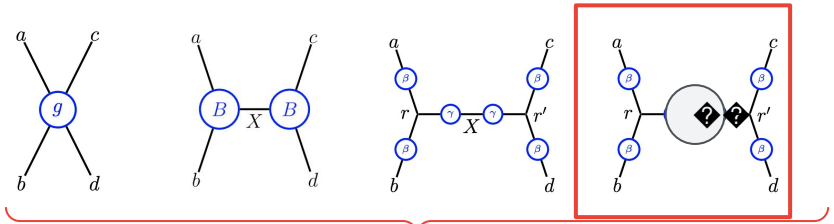
PRX Quantum **2**, 030305 (2021)

Google

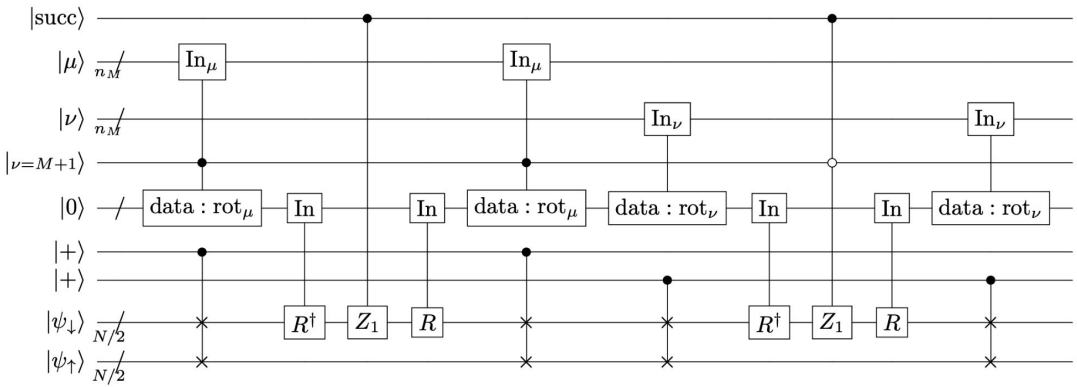
Cost of qubitization: block encodings

The complexity of building SELECT and PREPARE largely depends on the Hamiltonian factorization we use. The Coulomb kernel can be factorized in many different ways

$$H = \sum_{abcd} h_{abcd} a_a^\dagger a_b^\dagger a_d a_c = \sum_{\ell=0}^{L-1} \omega_\ell U_\ell$$



$$\frac{1}{8} \sum_{\alpha,\beta \in \{\uparrow,\downarrow\}} \sum_{\mu,\nu=1}^{M} \zeta_{\mu\nu} U_\mu^\dagger Z_{1,\alpha} U_\mu U_\nu^\dagger Z_{1,\beta} U_\nu$$



O(N)

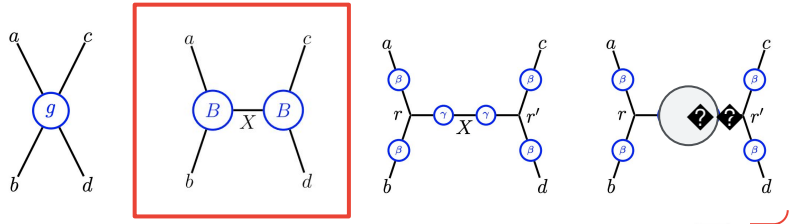
PRX Quantum **2**, 030305 (2021)

Google

Coulomb tensor factorizations symmetry-adapted

Each tensor factorization know must be constrained to the appropriate momentum conservation sum-rule.

$$H_2 = \frac{1}{2}\sum_{\sigma,\tau}\sum_{\mathbf{Q},\mathbf{k},\mathbf{k}'}\sum_{pqrs} V_{p\mathbf{k},q(\mathbf{k}\ominus\mathbf{Q}),r(\mathbf{k}'\ominus\mathbf{Q}),s\mathbf{k}'} a^\dagger_{p\mathbf{k}\sigma} a_{q(\mathbf{k}\ominus\mathbf{Q})\sigma} a^\dagger_{r(\mathbf{k}'\ominus\mathbf{Q})\tau} a_{s\mathbf{k}'\tau}$$

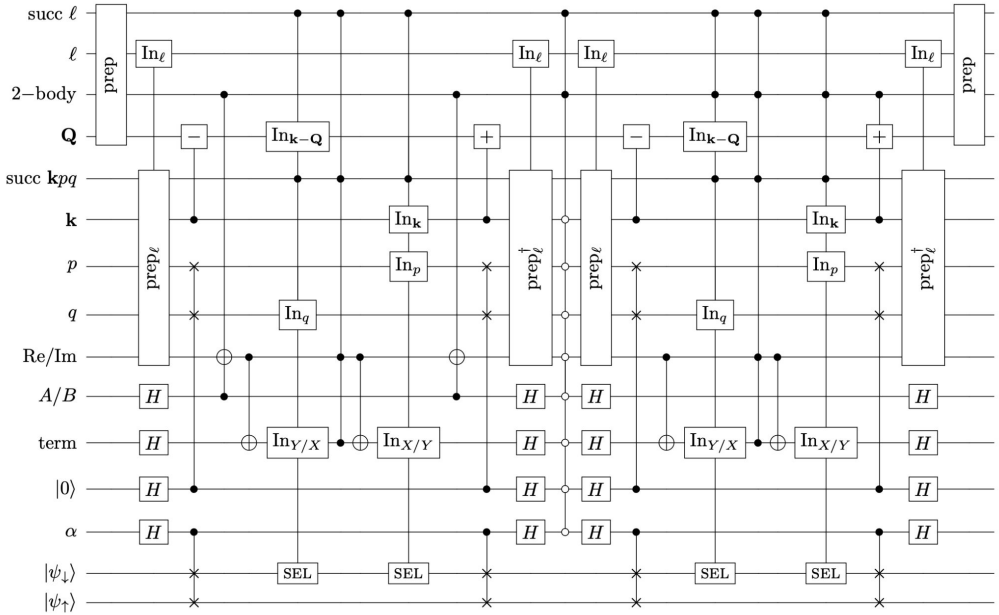


$$\hat{H}_2' = \frac{1}{2}\sum_{\mathbf{Q}}^{N_k}\sum_{n}^{M}\left(\hat{A}_n^2(\mathbf{Q}) + \hat{B}_n^2(\mathbf{Q})\right)$$

A and B terms have familiar Majorana fermion operator representations!

$$\hat{A}_n(\mathbf{Q}\neq 0) = \sum_{\sigma\in\{\uparrow,\downarrow\}}\sum_{\mathbf{k}}^{N_k}\sum_{pq}^{N/2}\left(\frac{i\mathrm{Re}[L_{p\mathbf{k}q(\mathbf{k}\ominus\mathbf{Q}),n}]}{4}\left(\vec{Z}X_{p\mathbf{k}\sigma}\vec{Z}Y_{q(\mathbf{k}\ominus\mathbf{Q})\sigma} - \vec{Z}Y_{p\mathbf{k}\sigma}\vec{Z}X_{q(\mathbf{k}\ominus\mathbf{Q})\sigma}\right) + \frac{i\mathrm{Im}[L_{p\mathbf{k}q(\mathbf{k}\ominus\mathbf{Q}),n}]}{4}\left(\vec{Z}X_{p\mathbf{k}\sigma}\vec{Z}X_{q(\mathbf{k}\ominus\mathbf{Q})\sigma} + \vec{Z}Y_{p\mathbf{k}\sigma}\vec{Z}Y_{q(\mathbf{k}\ominus\mathbf{Q})\sigma}\right)\right)$$

$O(N_k^2N^2)$ vs $O(N_kN^2)$



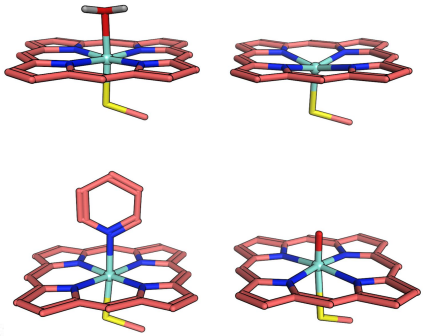
Assessing quantum/classical boundary for P450

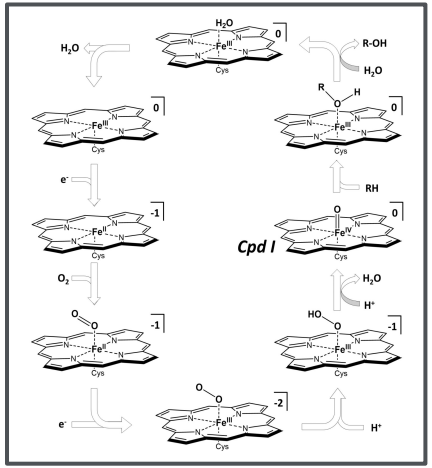
Quantum AI

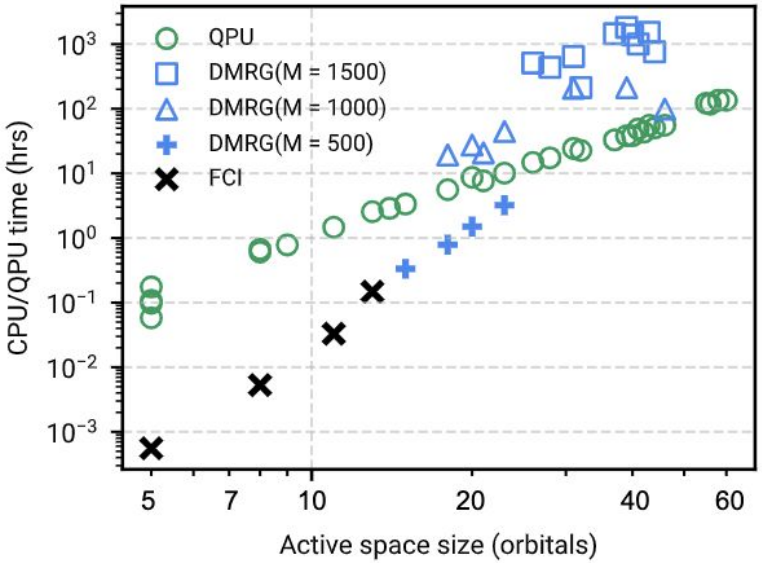
PNAS 119 (38), 2203533119 (2022)

Critical to articulate more specific industry-relevant use cases

P450 is strongly correlated iron-porphyrin / drug anti-target







We observe onset of quantum advantage for active space sizes near 80 qubits

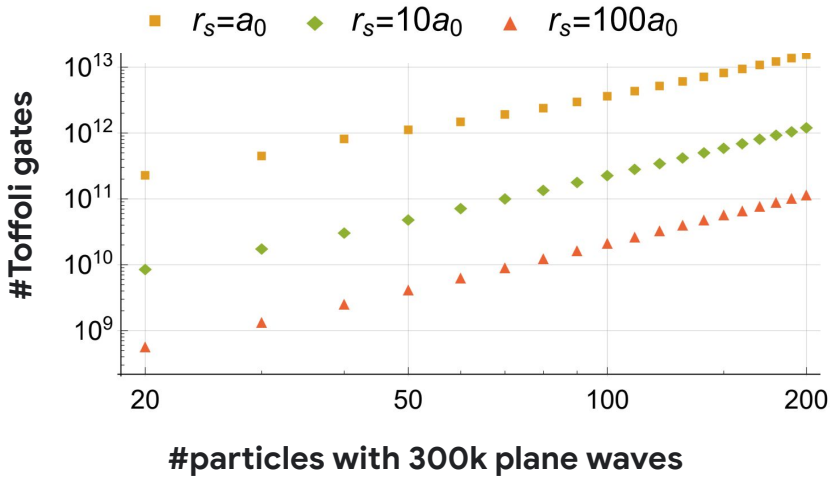
Calculations would require a few million physical qubits assuming 1e-4 error rates in surface code

The future of chemistry is 1st quantized!

Quantum AI

~100X more plane waves than molecular orbitals needed for target precision
In 2nd quantization #qubits = #orbitals so it is critical to use molecular orbitals

In 1st quantization #qubits = (#particles) log (#orbitals) but algorithmically challenging
Nature QI 5, 92 gives 1st quantized plane wave algorithm scaling as $(\#\text{particles})^{8/3}(\#\text{orbitals})^{1/3}$



PRX Quantum 2, 040332 (2021) compiles first quantized chem algorithms to fault-tolerant gates

Enables accurate simulation of realistic materials!

Non-Born-Oppenheimer quantum dynamics is now possible, and with very little overhead!

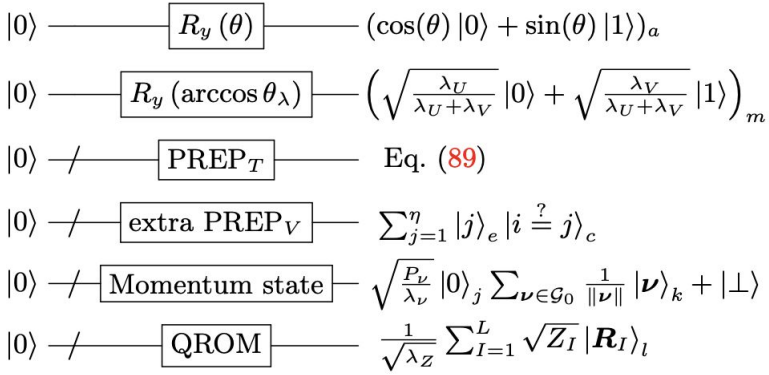
The future of chemistry is 1st quantized!

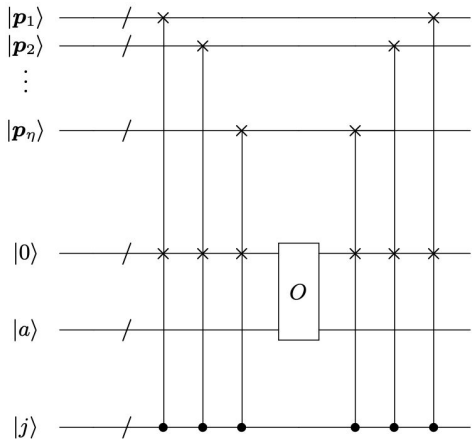
Quantum AI

$$T = \sum_{i=1}^{\eta} \sum_{p \in \mathcal{G}} \frac{\|\boldsymbol{G}_p\|^2}{2} |\boldsymbol{p}\rangle \langle \boldsymbol{p}|_i$$

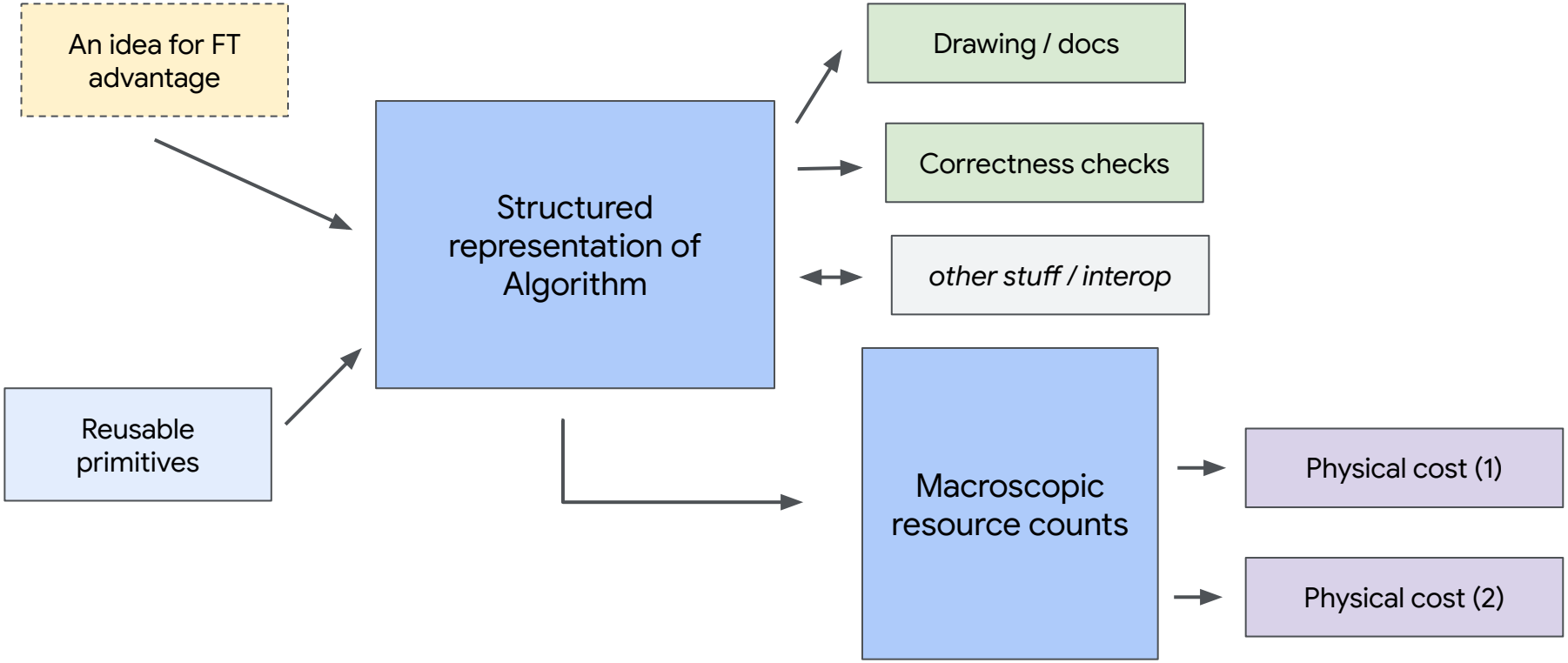
$$U = -\frac{4\pi}{\Omega} \sum_{i=1}^{\eta} \sum_{q \in \mathcal{G}} \sum_{\substack{\nu \in \mathcal{G}_0 \\ (\boldsymbol{q}-\boldsymbol{\nu}) \in \mathcal{G}}} \frac{\sum_{I=1}^{L} Z_I e^{i\boldsymbol{G}_\nu \cdot \boldsymbol{R}_I}}{\|\boldsymbol{G}_\nu\|^2} |\boldsymbol{q}-\boldsymbol{\nu}\rangle \langle \boldsymbol{q}|_i$$

$$V = \frac{2\pi}{\Omega} \sum_{i \neq j}^{\eta} \sum_{p,q \in \mathcal{G}} \sum_{\substack{\nu \in \mathcal{G}_0 \\ (\boldsymbol{p}+\boldsymbol{\nu}) \in \mathcal{G} \\ (\boldsymbol{q}-\boldsymbol{\nu}) \in \mathcal{G}}} \frac{1}{\|\boldsymbol{G}_\nu\|^2} |\boldsymbol{p}+\boldsymbol{\nu}\rangle \langle \boldsymbol{p}|_i |\boldsymbol{q}-\boldsymbol{\nu}\rangle \langle \boldsymbol{q}|_j$$





We need new software tools for this work

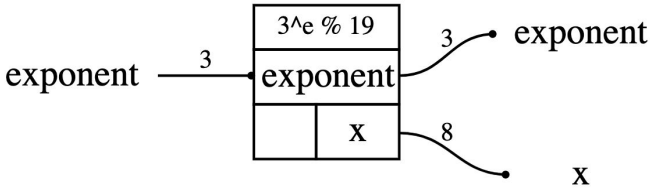


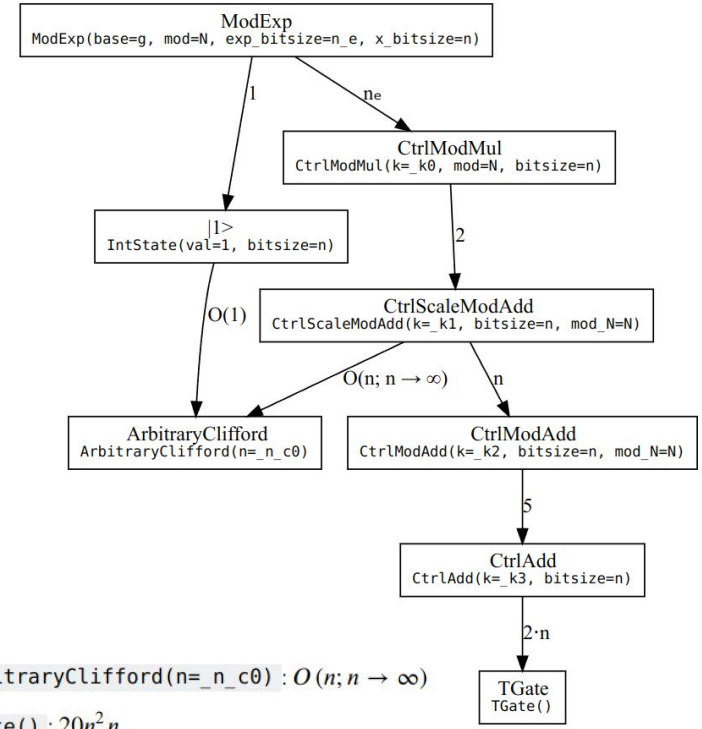
Bloqs: Structured Representation of Algos

We can't execute these algorithms (yet), so no requirement to code everything "all the way down"

Allow users to freely annotate known quantities and write protocols to query properties for complex algorithms.

E.g.: drawing, testing classical-reversible subroutines, counting resources etc.





`ArbitraryClifford(n=_n_c0)` : $O(n; n \to \infty)$

`TGate()` : $20n^2 n_e$

Cirq-FT: Cirq for Fault Tolerant algorithms

Make it easier to write Fault Tolerant algorithms in Cirq using qubit registers, quantum memory management etc.

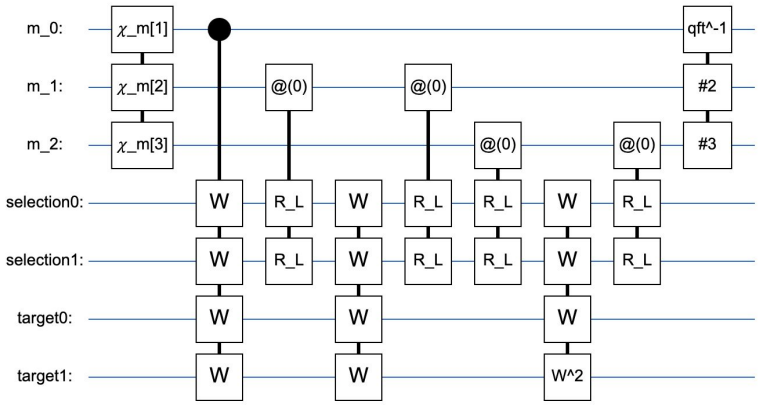
```python
@frozen
class MultiTargetCSwap(GateWithRegisters):
    bitsize: int

    @cached_property
    def registers(self) -> Registers:
        return Registers.build(ctrl=1, x=self.bitsize, y=self.bitsize)

    def decompose_from_registers(self,
        ctrl: Sequence[cirq.Qid],
        x: Sequence[cirq.Qid],
        y: Sequence[cirq.Qid],
    ) -> cirq.OP_TREE:
        yield [cirq.CSWAP(*ctrl, t_x, t_y) for t_x, t_y in zip(x, y)]
```

Reusable library of primitives and Bi-Directional Interop with Bloqs

Efficient resource counting for large systems.



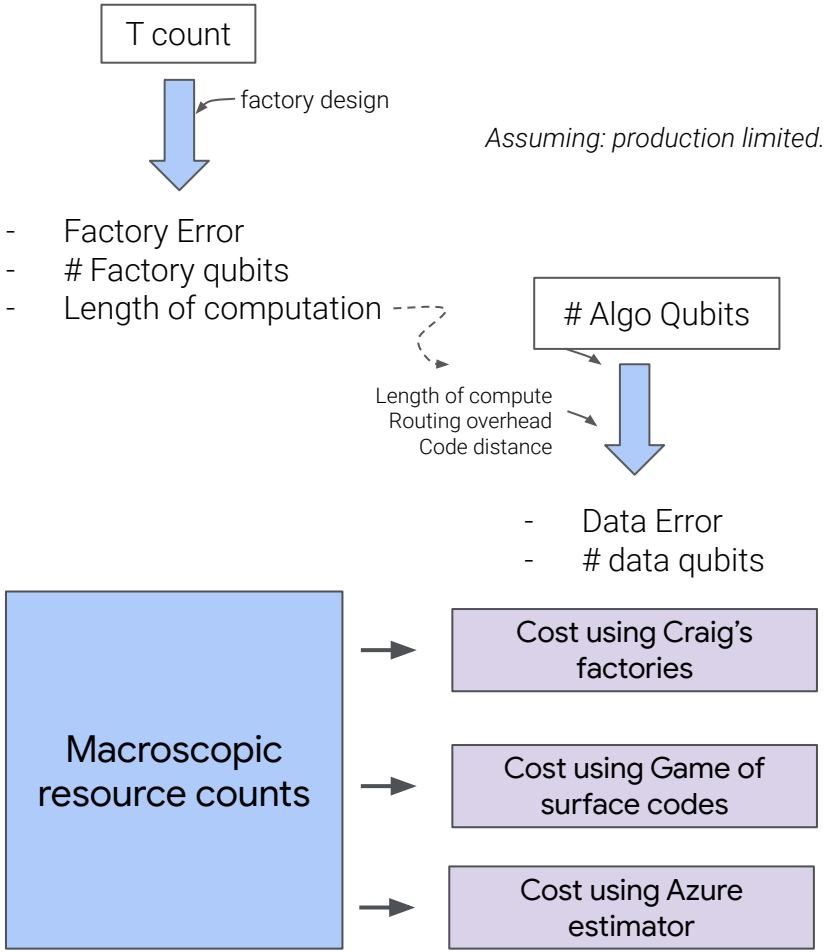
Heisenberg-Limited Phase Estimation of Qubitized Quantum Walk (arxiv:1805.03662)

Physical Costing

Formulas exist for estimating wallclock time, # physical qubits given T-gate count, code distance, etc.

They can be inaccessible or indiscriminately applied

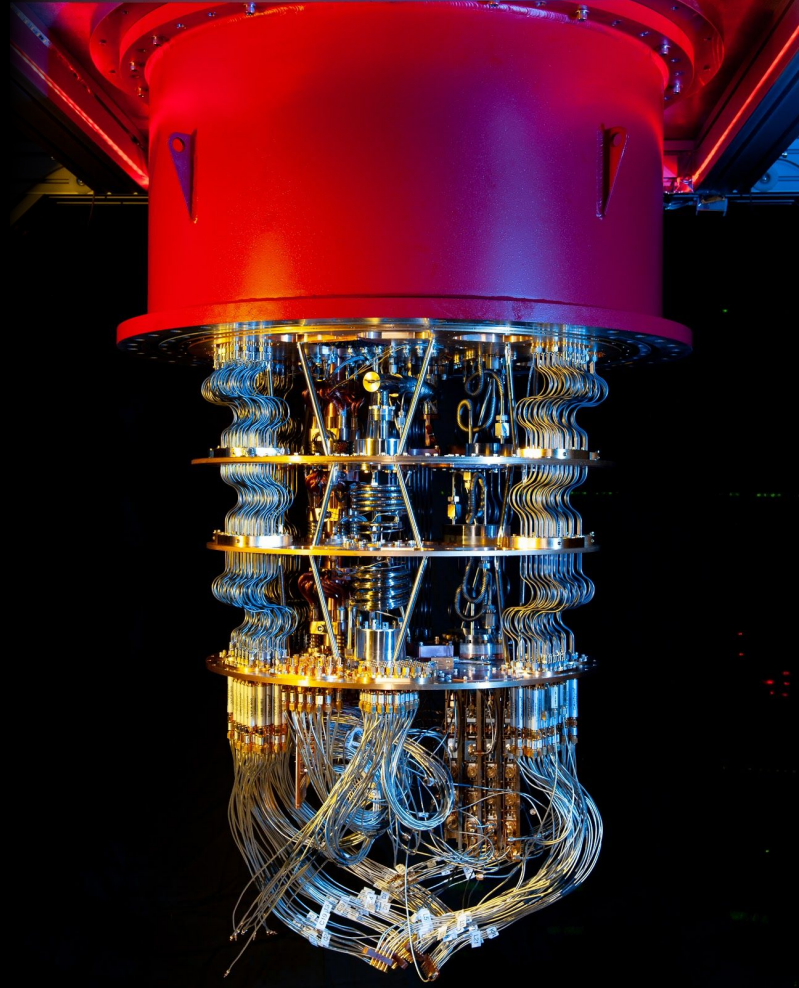
Goal: put a variety of them in one place with a common interface.



Thank you!

Quantum AI

- Google has PhD intern programs / student stay programs
- Interest in collaborating on FT algorithms applications in challenging domains
- Interest in quantum many-body methods development



Google