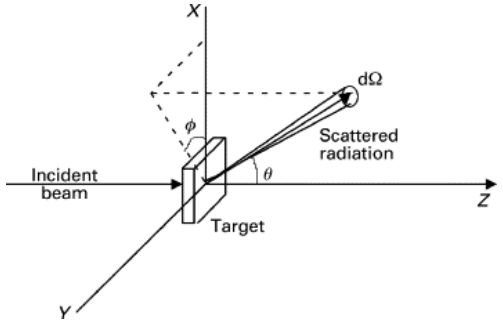
Optimising HEP Simulations on 3D Architecture: PrimitiveGates for BT

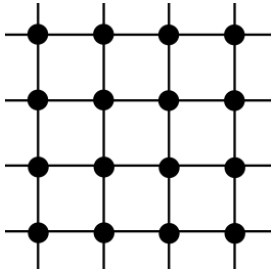
Andrea Maestri, Doga Kurkcuoglu (I typed it!), Hank Lamm,
Judah Unmuth-Yockey

September 16th 2022

HEP simulations on quantum computers

HEP simulations offer clear potential for quantum advantage.





- The group of interest in this case is $\mathbb{BT} \subset \mathbb{SU}(2)$
- 24 elements → quicosotetrit per link (qudit with d = 24).

- A map between generators of the group and the states of the qubits is required.

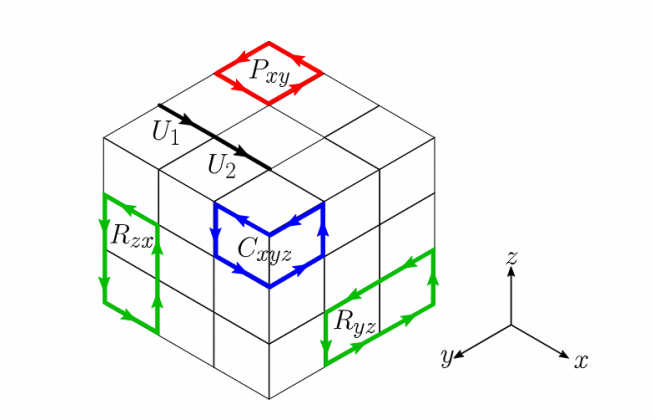
$$g = (-1)^m i^n j^o l^{p+2q} \rightarrow |mnopq\rangle \rightarrow |N\rangle$$

HEP simulations on quantum computers

In order to do that, an efficient implementation of the time evolution operator is required

$$U(t) \approx \left(\prod_{k=0} e^{-iH_k(t/N)} \right)^N$$

Where H_k are the non-commuting terms of $H = \sum_k H_k$

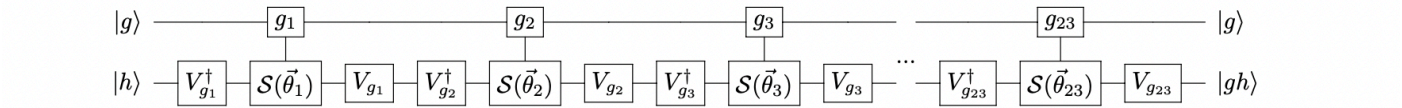


- Group theoretic operations as quantum circuits are required.
- Operations → Quantum Gates → Implementation

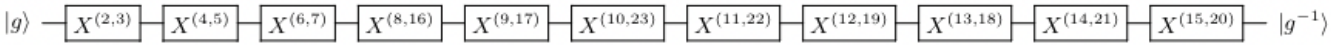
BT group's operations

The group operations can be implemented as quicosotetrit circuits:

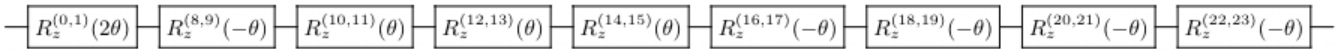
- Multiplication gate $\mathbb{U}_x$: $\mathbb{U}_x |g\rangle |h\rangle = |g\rangle |gh\rangle$



- Inversion gate $\mathbb{U}_{-1}$: $\mathbb{U}_{-1} |g\rangle = |g^{-1}\rangle$



- Trace gate $\mathbb{U}_{tr}$: $\mathbb{U}_{tr}(\theta) |g\rangle = e^{i\theta ReTr(g)} |g\rangle$



- Fourier transform gate $\mathbb{U}_F$: $\mathbb{U}_F \sum_{g \in G} f(g) |g\rangle = \sum_{\rho \in G^*} f^*(\rho)_{ij} |\rho, i, j\rangle$

SNAP & Displacement Decomposition

It can be shown that SNAP and Displacement gates are universal on a qudit

There is a problem with this!

- The cost of the simulation diverges rapidly
- Very far from being able to simulate $\mathbb{BT}$

- **Cost per link and trotter step [1]**
- **At least 4 links**
- **Grows exponentially with the number of steps**

Gate	$c\mathcal{S}(\vec{\theta})$	$\mathcal{S}(\vec{\theta})$	$\mathcal{D}(\alpha)$
$\mathcal{U}_{-1}$	0	24	25
$\mathcal{U}_{\times}$	23	575	575
$\mathcal{U}_{Tr}$	0	1	0
$\mathcal{U}_{FT}$	0	24	25
$e^{-iH_I\delta t}$	$598d - 506$	$15215.5d - 12771.5$	$15225d - 12775$

Pulse Optimisation, can we do better?

- Optimising pulses for a quicosotetrit is expensive using the standard libraries and methods.

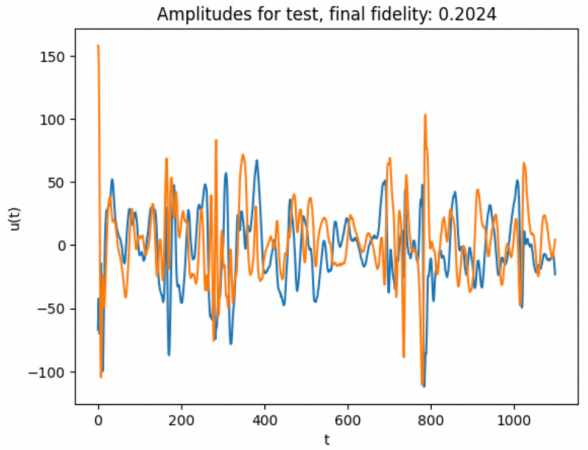
- Hamiltonian of the system

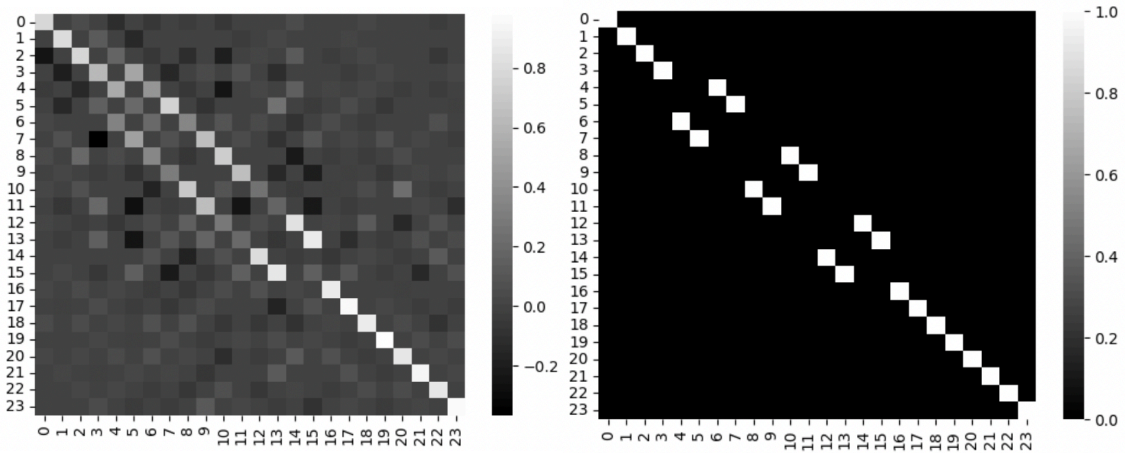
$$H = H_0 + \sum_{i=0}^{N} u_i(t) H_i$$

- Infidelity function

$$\hat{f} = 1 - |Tr(U_t^{\dagger} U_{opt})|$$

Simplified problem (best result)

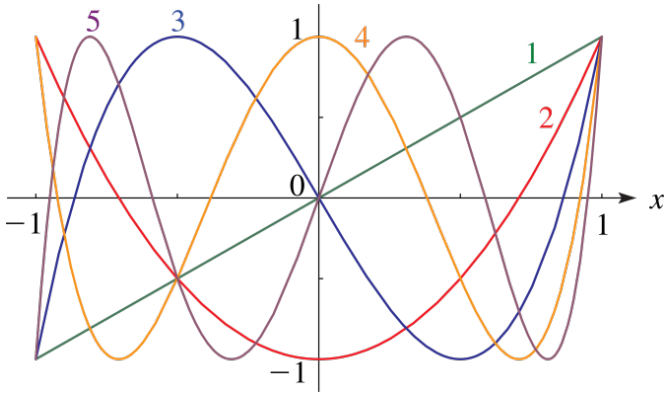




Chebyshev decomposition for the pulses

Analytical expression instead of approximation?

Pulses into Chebyshev polynomials: $u_i(t) \approx \sum_{k=0}^{n} c_k T_k(t)$



Adiabatic evolution of the system:

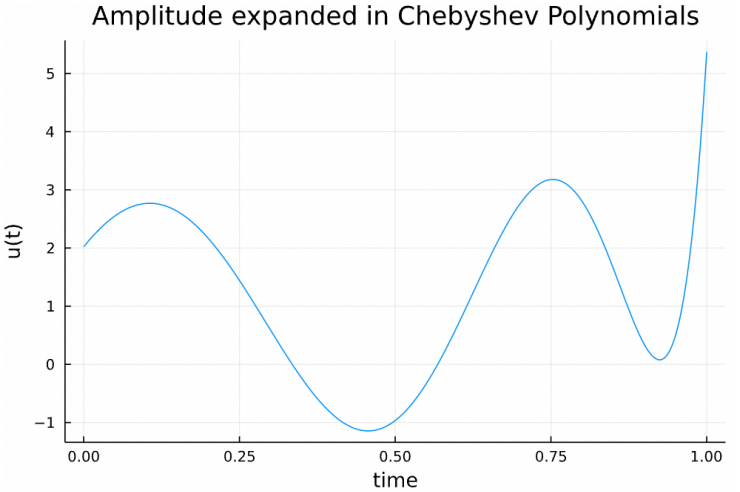
$$U(t) \approx \prod_{k=0}^{N} e^{-iH(t_k)\delta t} = \prod_{k=0}^{N} e^{-i\{H_0 + \sum_{i=0}^{N} u_i(t_k) H_i\}\delta t}$$

Time scale refined but less parameters to optimise

$$E(t) = |u(t) - P(t)| \leq \frac{1}{2^n (n+1)!} \max_{-1 \leq x \leq 1} |u^{(n+1)}(x)|$$

Preliminary Results: Hadamard gate

Good news: It works!



- Smoother signal then Qutip's optimisation.
- Speed up between 10 and 100 compared to Qutip.
- Shorter pulses compared to S&D decomposition.

$$\hat{f} \approx 10^{-16}$$

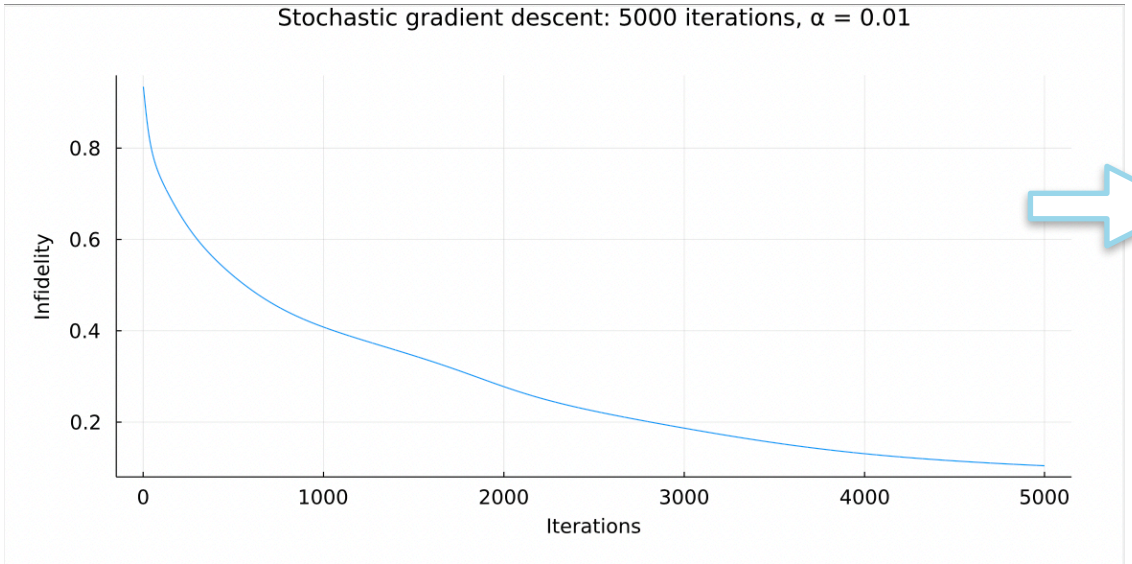
However, reducing the number of parameters might not be enough for higher dimensional problems!

- Analytical gradient was computed
- Different optimisation algorithms were tested

Gradually increasing Fock space's size N

Stochastic gradient descent

Fock space of size 5



- Search optimisation becomes more difficult than regular approaches
- Computing the gradient is extremely expensive.

The cost function is more complicated to optimise compared to piecewise approaches!

Brief analysis of the optimisation problem, $\hat{f}_{target} = 0.001$

$$U(t) \approx \prod_{k=0}^{N} e^{-iH(t_k)\delta t} = \prod_{k=0}^{N} e^{-i\{H_0 + \sum_{i=0}^{N} u_i(t_k)H_i\}\delta t}$$

Each parameter is in every term of the product

A fine time scale for the problem might be necessary but will make the optimisation more complicated

	Steps = 400	Steps = 600	Steps = 1000
N = 2	(4', 86,✓)	(4', 27, ✓)	(28', 26, ✓)
N = 3	(4', 68, ✓)	(4', 48, ✓)	(9', 69, ✓)
N = 4	(13', 179, ✓)	(19', 207, ✓)	(33', 210, ✓)
N = 5	(34', 323, ✓)	(62', 392, ✓)	(1h, 234, 0.006)
N = 6	(45', 322, ✓)	(3h, 834, ✓)	(1h, 181, 0.07)

Optimal combination of parameters for different N?

(T, iterations, $\hat{f}$)[1]

[1] all the simulations were run with the following specs: Apple M1 chip 8-core CPU with 4 performance cores and 4 efficiency cores

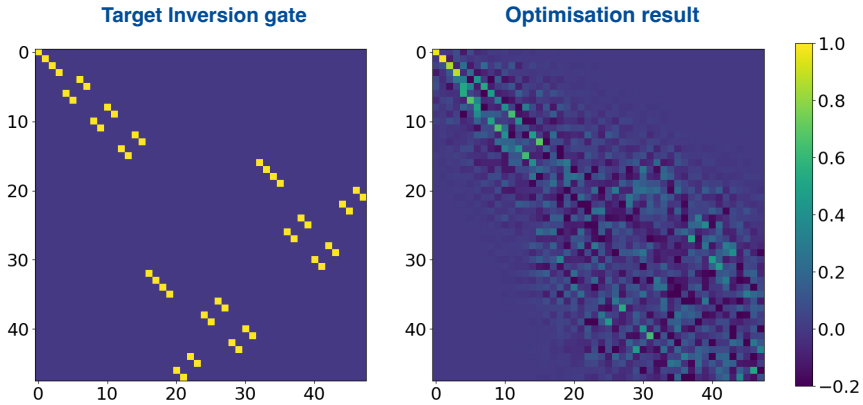
About the original HEP problem: Inversion gate

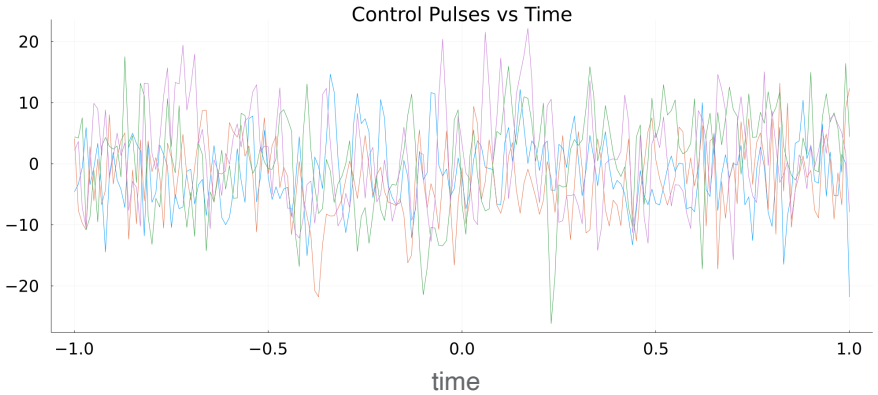
The best gradient descent algorithm → GRAPE

Qutip's result, Infidelity = 0.65

Chebyshev approximation result, Infidelity = 0.55

(New results are currently running)





The cost function is more complicated, but still "better" results.

More study has to be done to understand pros and drawbacks of this approach.

Orthogonality of Chebyshev polynomials?

What can be done next:

- Genetic algorithms for optimisation
- Hybrid approaches
- Adding more constraints to make the pulses easier to implement.
- Explore possible applications of the pulse optimisation

Thanks for listening!