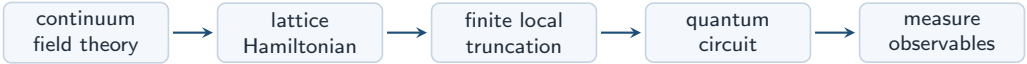
# Utilising Qudits for Fault-Tolerant Quantum Simulation

*arXiv:2604.26792*

Sam Godwood HEP Annual Meeting 22/05/26

**Supervisors:**
Constantinos Andreopoulos Gabriel N. Perdue Doğa Murat Kürkçüoğlu

# Quantum Simulation




$$H_{\text{lattice}} \longrightarrow U(t) = \exp(-iHt)$$

- Lattice field theory discretises continuum fields into coupled local degrees of freedom.
- Classical cost grows quickly with volume, cutoff, and real-time evolution.
- Real-time Hamiltonian dynamics a natural quantum target, but likely only with fault tolerance.

**NISQ**

hardware-native gates / pulses
parameterised rotations directly implemented
cost: depth, width, shots

**Fault tolerant**

protected logical gates
parameterised rotations $\rightarrow$ finite gate set
cost: non-Clifford gates

**Fault-tolerant resource reduction $\approx$ fewer non-Clifford operations.**

# Why Qudits?

Truncating local bosonic or gauge-field degrees of freedom naturally gives $d$-dimensional Hilbert spaces.

## Qubit vs. Qudit Encodings


**Qudit:** one logical $d$-level system.

**Qubits:** binary register with $\lceil \log_2 d \rceil$ qubits.

Same $d$-dimensional space, but different logical primitives.

## From NISQ savings to FTQC cost

- Qudits can reduce registers, gates, or depth vs. qubits in NISQ simulations, see e.g. *arXiv:2310.12110*, *arXiv:2203.15541*...
- Fault tolerance changes the accounting: continuous rotations must be synthesised from a finite logical gate set.

**Compare: non-Clifford count**

**continuous rotations → finite logical gate set → non-Clifford resources**

# Comparison setup

## Physical target

- discretised real scalar field in field amplitude basis

$$\phi_x = \sum_{\phi} \phi \, |\phi\rangle\langle\phi|, \qquad U_x(t) = \exp(-it\,\phi_x^2)$$

- same structure as e.g. quadratic electric-energy terms

## Logical operations

qubits: Clifford + $SU(2)$ rotations
qudits: Clifford + embedded two-level $SU(2)$ rotations

## Two simulation regimes

- **Product formula:**
  directly implement pieces of $U(t)$.
- **LCU / block encoding:**
  encode $\phi_x^2$ as a weighted sum of unitaries; time evolution built from the block encoding.
- **Comparison:**
  same local operator, two encodings, non-Clifford cost.

Product formulas: *arXiv:1912.08854*
LCU / block encodings: *arXiv:1610.06546*

# Regime I: Product Formula

Compare one onsite step: $U_x(t) = \exp(-it\,\phi_x^2)$.

## Qubit route

Binary register on $\lceil \log_2 d \rceil$ qubits.

$$\phi_x^2 \;=\; \alpha I + \sum_m \alpha_m Z^{(m)} + \sum_{m<\ell} \alpha_{m\ell}\, Z^{(m)} Z^{(\ell)}$$

single- and pairwise-$Z$ phases

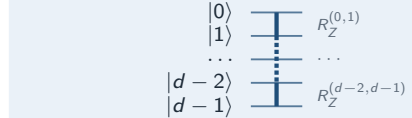
$$\boxed{q_0}\ \boxed{q_1}\ \cdots\ \boxed{q_{b-1}}$$

$$\#\ \text{rotations} \;=\; O((\log d)^2)$$

## Qudit route

$$U_x(t) = \sum_{n=0}^{d-1} e^{-it\lambda_n^2}\, |n\rangle\langle n|$$

phase tuned by neighbouring $SU(2)$ rotations




$$\#\ \text{rotations} = d - 1 = O(d)$$

**Qubits exploit binary structure, qudits tune level phases directly...**

# Regime I: Product Formula

Product formulas give continuous rotations; fault tolerance requires synthesising them into a discrete logical gate set.

continuous rotations $\longrightarrow$ finite logical gate set $\longrightarrow$ non-Clifford resources

**Qubit route**

$$\#\text{synthesised rotations} = \Theta((\log d)^2)$$

**Qudit route**

$$\#\text{synthesised rotations} = \Theta(d)$$

**Qubit encoding is asymptotically cheaper.**

# Regime II: LCU / block encoding

## LCU idea

$$H = \sum_j h_j U_j, \qquad \lambda = \sum_j |h_j|,$$

$$\mathrm{PREP}\,|0\rangle = \sum_j \sqrt{|h_j|/\lambda}\,|j\rangle, \quad \mathrm{SELECT} = \sum_j |j\rangle\langle j| \otimes U_j$$

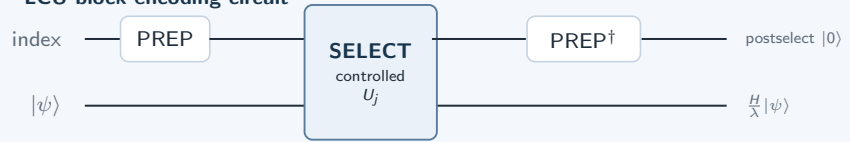
PREP chooses a term; SELECT applies the corresponding unitary; together block-encodes $H/\lambda$.

## Why qudits look promising

- Operator expands naturally in qudit Pauli $Z$ basis (Clifford)

$$\phi_x^2 = \beta_0 I + \sum_{r=1}^{d-1} |\beta_r|\,\widetilde{U}_r, \qquad \widetilde{U}_r = \exp(i\theta_r) Z_d^r$$

**LCU block-encoding circuit**



# Regime II: LCU / block encoding

## Qudit LCU

- SELECT essentially free: controlled-$Z_d^r$ is Clifford.
- **But PREP** must build a weighted superposition over all $d-1$ levels.
- Each weight requires one embedded two-level $SU(2)$ rotation.

**# rotations per call** $\sim O(d)$

## Qubit LCU

- Less natural Pauli structure for $\phi_x^2$ with qubits
- But binary registers are powerful *computational* objects.
- Block encoding built with *reversible arithmetic* on bit pairs, inspired by: *arXiv:2507.22814* & *arXiv:2105.12767*.

**# rotations per call** $\sim O(\log d)$

**Qubit block encoding is asymptotically cheaper.**

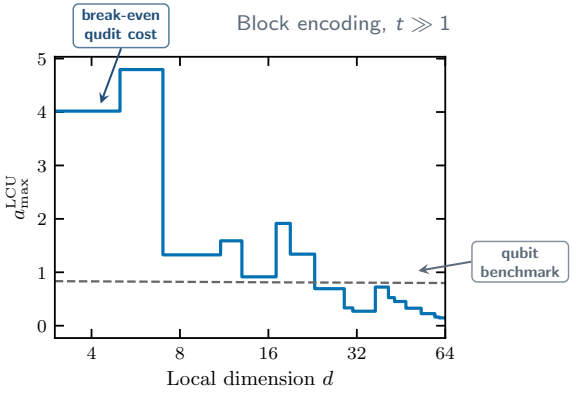
# Where can qudits win? I: compiler targets

## Qudit synthesis cost

- compile / synthesise $SU(2)$ rotations into finite logical qudit gate set
- approximating to precision $\varepsilon$ costs $a \log(1/\varepsilon)$ non-Cliffords
- $a$ unknown for $d > 3$

## How to read the target

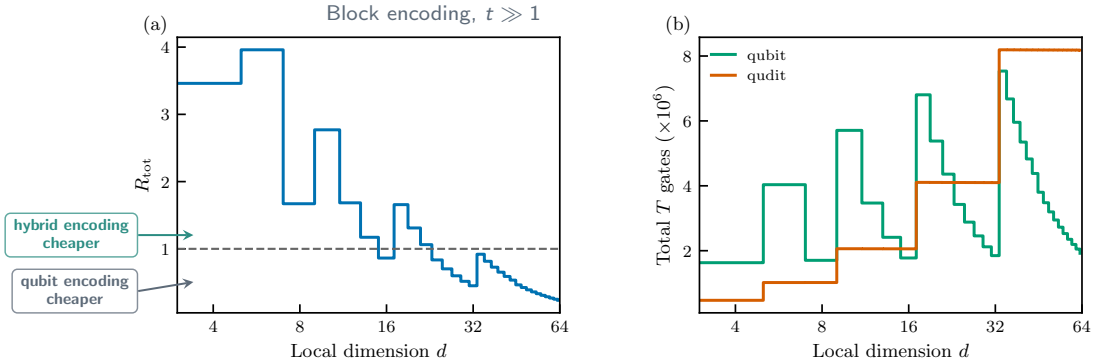
- curve gives the largest allowable synthesis prefactor $a_{\max}$
- qudits cheaper when realised compiler prefactor satisfies $a < a_{\max}$




**Finite-$d$ qudit advantage is plausible only when the required qudit synthesis cost lies within achievable compiler performance.**

# Where can qudits win? II: hybrid construction

- Hybrid algorithm: allow switching between qubits and qudits; potentially feasible via code switching *arXiv:1512.06132*




**Hybrid cheaper:** $d \in [3, 13] \cup [17, 21]$

**Hybrid algorithm gives constant-factor savings at low dimensions.**

# Takeaway and outlook

## Takeaway

- Qudits are natural for $d > 2$ local Hilbert spaces.
- But finding regimes of usefulness is non-trivial.
- We provide compiler targets and potential hybrid constructions to allow for qudit advantage.

## Outlook

- Tight synthesis bounds for single-qudit rotations.
- Explicit qudit FT compilers and gate sets.
- Feasibility of switching between qudit and qubits.
- More qudit-native Hamiltonians, e.g. $\mathbb{Z}_N$ lattice gauge theories.

# BACKUP

# Why does non-Clifford count matter?

- Each non-Clifford gate requires magic-state distillation
- A single non-Clifford gate can cost $\sim 10^3$ times more than a Clifford gate (see *arXiv:1905.06903*)
- Reducing non-Clifford count lowers the overall resource overhead.
- Number-theoretic approaches guarantee approximation of $SU(2)$ unitaries using $O(\log(1/\varepsilon))$ non-Clifford gates.
- But concrete compilers can be difficult and expensive in practice e.g. *arXiv:2503.20203*