Quantum Circuit and Tensor Network Implementation of the 2D Acoustic Wave Equation

Seismic Forward Modeling via Computational Basis Shifts

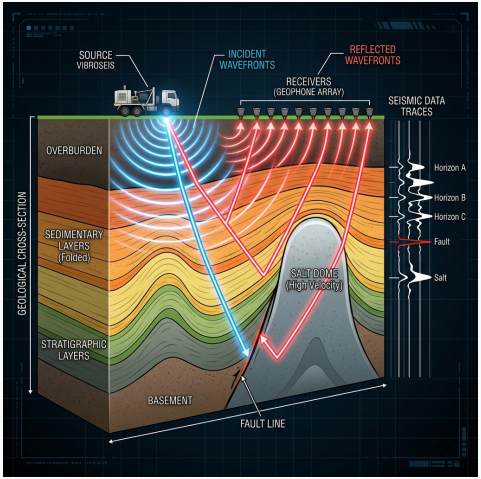
Tamas Nemeth[1] and Gábor Vattay[2]

[1]WaveRunr LLC, Palo Alto, California

[2]Eötvös Loránd University, Budapest, Hungary

GPUday Scientific Workshop – May 2026

Industrial Context: RTM and FWI in Geophysics



Seismic reflection survey mapping subsurface layers and high-contrast bodies (salt domes).

Reverse Time Migration (RTM)

- Uses the **two-way wave equation** to propagate source wavefields forward and receiver wavefields backward in time.
- Crucial for imaging beneath complex, high-velocity structures like **salt domes**.

Full Waveform Inversion (FWI)

- An iterative data-fitting method using the entire seismic waveform (phase and amplitude) to reconstruct high-resolution velocity models.
- Requires **thousands of forward simulations**, making it the primary HPC bottleneck in the energy sector.

Overview & Core Contribution

The Problem

Simulating seismic wavefields (e.g., for RTM and FWI) on classical computers scales exponentially with grid sizes, hitting memory and computational bottlenecks on modern architectures.

Our Proposed Solution

A unified framework linking **quantum computing paradigms** and their **classical tensor network equivalents**:

- A **quantum circuit-based formulation** for explicit Finite-Difference Time-Domain (FDTD) wave simulation without the overhead of Quantum Fourier Transforms (QFT).
- Mapping this quantum architecture onto a **Tensor Train / Matrix Product State (MPS)** representation.
- Demonstration of the MPS solver on the complex **2D Marmousi model**.

Introduction: The Wave Simulation Challenge

- **Seismic Forward Modeling** is the engine behind Reverse Time Migration (RTM) and Full Waveform Inversion (FWI).
- **The Computational Wall**: Massive multi-dimensional grids demand high-performance computing resources, pushing CPUs/GPUs to architectural limits.
- **The Quantum Paradigm**: Qubits offer exponential memory scaling (2^N states in N qubits).
- **Near-term Limits**: Direct Quantum algorithms (e.g. LCU) are too deep for noisy intermediate-scale quantum (NISQ) devices.
- **The Bridge**: Tensor Trains (TT) / Matrix Product States (MPS) compress quantum states to run deterministically on classical hardware.

2D Acoustic Wave Equation & FDTD Discretization

Continuous Wave Equation:

$$\frac{\partial^2 u(x,z,t)}{\partial t^2} = c^2(x,z)\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial z^2}\right)$$

- $u(x,z,t)$: Scalar acoustic wavefield
- $c(x,z)$: Subsurface velocity model
- Square grid: $N_x = N_z = 2^n$ with spacing Δh
- Temporal step Δt governed by CFL stability

FDTD Central Difference Scheme:

$$u_{i,j}^{k+1} = 2u_{i,j}^{k} - u_{i,j}^{k-1} + \\ + \lambda^2 c_{i,j}^2 \left(u_{i+1,j}^{k} + u_{i-1,j}^{k} + u_{i,j+1}^{k} + u_{i,j-1}^{k} - 4u_{i,j}^{k}\right)$$

- i, j: Spatial grid coordinates
- k: Discrete time step
- $\lambda = \frac{\Delta t}{\Delta h}$: Mesh ratio

Quantum Register Encoding

State Representation

We map the $2^n \times 2^n$ grid directly to the computational basis of a $2n$-qubit register. The wavefield state at time step k is represented by:

$$|\psi^k\rangle = \sum_{x=0}^{2^n-1} \sum_{z=0}^{2^n-1} u_{x,z}^k |x\rangle|z\rangle$$

where $|x\rangle = |x_{n-1} \dots x_0\rangle$ and $|z\rangle = |z_{n-1} \dots z_0\rangle$ are binary-encoded coordinate states.

Velocity Operator

The squared velocity field $c^2(x, z)$ maps to a diagonal operator $\hat{C}^2$:

$$\hat{C}^2|x\rangle|z\rangle = c^2(x, z)|x\rangle|z\rangle$$

Implemented via Quantum Read-Only Memory (QROM) or Walsh series expansions.

Quantum Shift Operators

- **Avoiding QFT Overhead**: Standard QFT-based translation requires high gate depth and routing overhead. We shift strictly in the **computational basis** using incrementer/decrementer circuits.
- **Forward Shift Operator** $\hat{S}_x^+$ maps $|x\rangle \to |(x+1) \bmod 2^n\rangle$ (cyclic ripple-carry adder with input fixed to 1).
- The bit-level transformation on the m-th qubit:

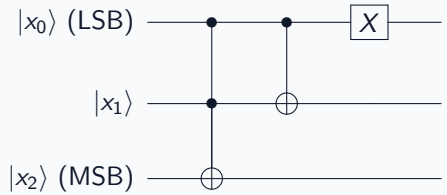
$$x_m \leftarrow x_m \oplus \left(\prod_{l=0}^{m-1} x_l \right)$$

  which translates into a cascaded sequence of multi-controlled Pauli-X (Toffoli) gates.
- **Backward Shift Operator** $\hat{S}_x^-$ is the adjoint $(\hat{S}_x^+)^\dagger$, implemented by reversing the gate order.

3-Qubit Quantum Incrementer Circuit

Circuit Construction ($\hat{S}^+$)

Executing Toffoli gates sequentially from the most significant bit (MSB) to the least significant bit (LSB) minimizes the gate depth:



Discrete Laplacian Operator

Using the 1D shift operators, the 2D discrete Laplacian $\hat{L}$ is defined as:

$$\hat{L} = \hat{S}_x^+ \otimes \hat{I}_z + \hat{S}_x^- \otimes \hat{I}_z + \hat{I}_x \otimes \hat{S}_z^+ + \hat{I}_x \otimes \hat{S}_z^- - 4\hat{I}$$

Wavefield Evolution via Linear Combination of Unitaries

Temporal Stepping Relation

The wavefield state update is given by:

$$|\psi^{k+1}\rangle = \hat{A}|\psi^{k}\rangle - |\psi^{k-1}\rangle$$

where $\hat{A} = 2\hat{I} + \lambda^2 \hat{C}^2 \hat{L}$ is a generalized non-unitary transition operator.

Block Encoding with LCU

- $\hat{A}$ is a sum of unitary operations scaled by coefficients.
- We can block-encode $\hat{A}$ into a larger unitary matrix.
- Applying it probabilistically on the register is achieved by preparing ancillae, running multiplexed select operations, and measuring the ancilla.
- Temporal stencil is maintained via two physical registers: $|\psi^{k}\rangle$ and $|\psi^{k-1}\rangle$.

The Bridge: Classical Tensor Train Emulation

Why Tensor Trains / Matrix Product States?

- Fault-tolerant quantum hardware is not yet available at the scale needed for seismic exploration.
- **Mathematical Isomorphism**: Tensor Train (TT) is the mathematical equivalent to Matrix Product States (MPS) in quantum many-body physics.
- Same spatial entanglement scaling principles are shared between quantum registers and tensor networks.

Physical Intuition: Compression of Wavefields

- Quantum entanglement $\leftrightarrow$ spatial correlation and wavefront complexity.
- In smoothly varying subsurface velocity models, the wavefield maintains a low entanglement signature.
- This allows the 2^{2n}-dimensional state vector to be compressed into a low-rank tensor network, bypassing the exponential memory wall of classical computing.

Tensor Network Representation

- The $N_x \times N_z = 2^n \times 2^n$ discrete field is reshaped into a $2n$-site spin-1/2 chain.
- The state $|\psi^k\rangle$ is decomposed into a Matrix Product State (MPS):

$$|\psi^k\rangle = \sum_{s_1,\ldots,s_{2n}\in\{0,1\}} A^{(1)}_{s_1} A^{(2)}_{s_2} \ldots A^{(2n)}_{s_{2n}} |s_1, \ldots, s_{2n}\rangle$$

- Bulk tensors $A^{(m)}_{s_m}$ have dimensions up to $\chi \times \chi \times 2$, where χ **is the bond dimension**.
- χ bounds the amount of entanglement entropy (spatial correlation complexity) stored in the wavefield.
- Truncation of χ removes high-frequency/scattering components, acting as a physical compression filter.

Shift Operators as Matrix Product Operators

- The cyclic shifts $\hat{S}_x^{\pm}$ and $\hat{S}_z^{\pm}$ are formulated as Matrix Product Operators (MPOs) with **bond dimension** $\chi = 2$.
- The virtual bond index $\alpha_m \in \{0, 1\}$ propagates the "carry bit" from LSB ($m = 0$) to MSB ($m = n - 1$).
- Local bulk core MPO tensor $W^{[m]}$:

$$W^{[m]} = \begin{pmatrix} \hat{I} & 0 \\ \hat{\sigma}^+ & \hat{\sigma}^- \end{pmatrix}$$

  where $\hat{I}$ = identity, $\hat{\sigma}^+ = |1\rangle\langle 0|$, and $\hat{\sigma}^- = |0\rangle\langle 1|$.
- Boundary tensors feed and terminate the carry bit:

$$W^{[0]} = \begin{pmatrix} \hat{\sigma}^+ & \hat{\sigma}^- \end{pmatrix}, \quad W^{[n-1]} = \begin{pmatrix} \hat{I} \\ \hat{\sigma}^+ \end{pmatrix}$$

MPS Wavefield Evolution and Compression

MPS Time-Stepping Scheme

The discrete temporal evolution of the wavefield is expressed as:

$$|\psi^{k+1}\rangle = 2|\psi^{k}\rangle - |\psi^{k-1}\rangle + \lambda^2 \hat{C}^2 \left(\hat{S}_x^+ + \hat{S}_x^- + \hat{S}_z^+ + \hat{S}_z^- - 4\hat{I} \right) |\psi^{k}\rangle$$

where operators $\hat{S}_x^{\pm}, \hat{S}_z^{\pm}$, and $\hat{C}^2$ are applied as MPOs to the MPS state $|\psi^{k}\rangle$.

Crucial SVD/QR Compression Step

- Applying MPOs and adding states multiplies the bond dimension χ.
- Without truncation, χ would grow exponentially with the number of steps.
- **SVD Compression**: After each step, a left-to-right QR sweep followed by a right-to-left SVD sweep is performed.
- Singular values below a cutoff threshold $\lambda_i < \epsilon \lambda_{\text{max}}$ are discarded, and the bond dimension is capped at χ_{max} to ensure numerical tractability.

Implementation: The SimpleMPS Python Class

```python
class SimpleMPS:
    def __init__(self, tensors):
        self.tensors = tensors
        self.n = len(tensors)

    def compress(self, max_chi, cut):
        # Left-to-right QR sweep to orthogonalize
        for i in range(self.n - 1, 0, -1):
            T = self.tensors[i]
            L, R, P = T.shape
            mat = T.transpose(0, 1, 2).reshape(L, R * P)
            Q, R_mat = np.linalg.qr(mat.T)
            self.tensors[i] = Q.T.reshape(-1, R, P)
            self.tensors[i-1] = np.tensordot(self.tensors[i-1], R_mat.T, axes=(1, 0)).transpose
                (0, 2, 1)

        # Right-to-left SVD sweep to truncate bond dimension
        for i in range(self.n - 1):
            T = self.tensors[i]
            L, R, P = T.shape
            U, S, Vh = np.linalg.svd(T.transpose(0, 2, 1).reshape(L * P, R), full_matrices=False)
            keep = min(len(S), max_chi)
            keep = max(min(keep, np.sum(S > cut * S[0])), 1)
            self.tensors[i] = U[:, :keep].reshape(L, P, keep).transpose(0, 2, 1)
```

Implementation: Operator Application & Time Step

```python
def apply_mpo(mpo, mps):
    new_T = []
    for W, A in zip(mpo, mps.tensors):
        T = np.tensordot(W, A, axes=(3, 2)).transpose(0, 3, 1, 4, 2)
        new_T.append(T.reshape(T.shape[0]*T.shape[1], T.shape[2]*T.shape[3], T.shape[4]))
    return SimpleMPS(new_T)

# In the simulation loop:
# 1. Apply shifts to get laplacian terms
u_xp, u_xm = apply_mpo(Sx_p, psi), apply_mpo(Sx_m, psi)
u_yp, u_ym = apply_mpo(Sy_p, psi), apply_mpo(Sy_m, psi)

# 2. Add MPS terms with weights [1.0, 1.0, 1.0, 1.0, -4.0]
lap_mps = add_multiple_mps_exact([u_xp, u_xm, u_yp, u_ym, psi], [1., 1., 1., 1., -4.])
lap_mps.compress(max_bond, cutoff)

# 3. Apply velocity operator C^2 and sponge boundaries, then step forward:
# nxt = (2 * psi - psi_prev_damped + dt^2 * C^2 * laplacian) / (1 + 0.5 * dt * gamma)
```

Numerical Results: Marmousi Wavefield Simulation

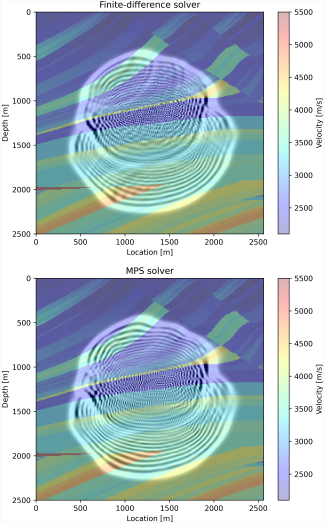
Simulation Parameters:

- **Grid Dimensions**: 256×256 grid points ($n = 8$ bits per coordinate).
- **Grid Spacing**: $\Delta x = \Delta z = 10$ meters (total physical domain: 2.56×2.56 km).
- **Time Step**: $\Delta t = 1$ ms.
- **Source**: Impulsive pressure source in the center of the grid $(1280\text{ m}, 1280\text{ m})$.
- **Propagation**: 300 steps.

MPS Parameters:

- **Max Bond Dimension**: $\chi_{\max} = 128$.
- **SVD Cutoff**: $\epsilon = 10^{-7}$.
- **Boundary Conditions**: Sponge PML (perfectly matched layer) dampening implemented directly as diagonal MPOs.

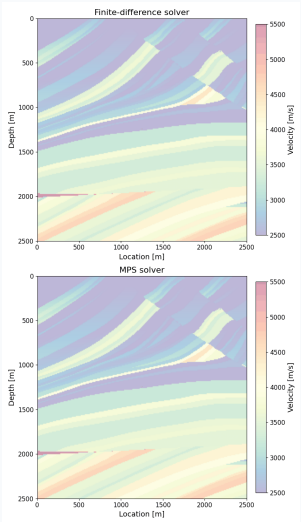
Results: FDTD vs. Tensor Train (MPS)



Wavefield Comparison Details

- **Upper Wavefield**: Explicit 2D FDTD solver (classical reference).
- **Lower Wavefield**: Matrix Product State (MPS) / Tensor Train solver.
- **Grid Setup**: 256×256 subset of the Marmousi velocity model, spatial spacing $\Delta h = 10$ m.
- **MPS Configuration**: $\chi_{\max} = 128$, cutoff $\epsilon = 10^{-7}$, run for 300 steps.
- **Findings**: The MPS solver reproduces the wavefronts with high fidelity. Slight discrepancies are due to bond dimension truncation filtering fine scattering details.

Results: Wavefield Propagation Movie



Comparison of Wave Equation Representations

Method	Equation / Mathematical Formulation
FDTD	$u_{i,j}^{k+1} = 2u_{i,j}^{k} - u_{i,j}^{k-1} + \lambda^2 c_{i,j}^2 \left(u_{i+1,j}^{k} + u_{i-1,j}^{k} + u_{i,j+1}^{k} + u_{i,j-1}^{k} - 4u_{i,j}^{k}\right)$
Quantum	$\lvert\psi^{k+1}\rangle = \left[2\hat{I} + \lambda^2 \hat{C}^2 (\hat{S}_x^+ \otimes \hat{I}_z + \hat{S}_x^- \otimes \hat{I}_z + \hat{I}_x \otimes \hat{S}_z^+ + \hat{I}_x \otimes \hat{S}_z^- - 4\hat{I})\right] \lvert\psi^{k}\rangle - \lvert\psi^{k-1}\rangle$
MPS	$\lvert\psi^{k+1}\rangle = 2\lvert\psi^{k}\rangle - \lvert\psi^{k-1}\rangle + \lambda^2 \hat{C}^2 \left(\hat{S}_x^+ + \hat{S}_x^- + \hat{S}_z^+ + \hat{S}_z^- - 4\hat{I}\right) \lvert\psi^{k}\rangle$

Table: All formulations compute the next time-step from the current and previous states, but differ in state representation (floats in RAM vs. entangled qubits vs. compressed tensor cores).

Discussion & Future Outlook

- **Feasibility Demonstrated**: The MPS representation accurately produces 2D seismic wavefields on the complex Marmousi model.
- **Memory Compression**: Low-entanglement regions of the wavefield are heavily compressed, showing potential for large-scale grids.
- **Parameter Tuning**: The trade-off between bond dimension χ and numerical accuracy (SVD cutoff ϵ) controls the balance between physical dispersion errors and computational speed.
- **Scaling to 3D**: The framework naturally generalizes to 3D using standard MPO extensions:

$$\hat{S}_y^{\pm} = \hat{I}_{2^n} \otimes \hat{S}^{\pm} \otimes \hat{I}_{2^n}$$

- **Hardware Acceleration**: Future work will map the MPS core multiplications directly to GPU architectures.

Thank you! Questions?