# Fast time evolution of matrix product states using the QR decomposition

Jakob Unfried, Johannes Hauschild and Frank Pollmann

Technical University Munich

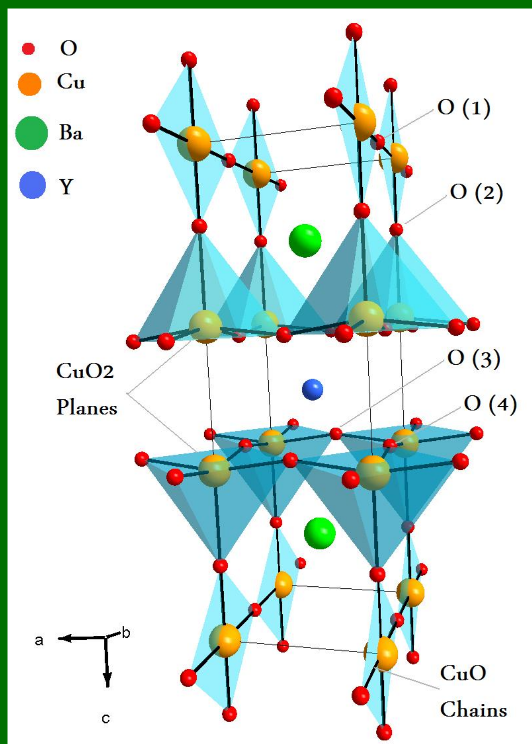
Munich Center for Quantum Science and Technology (MCQST)

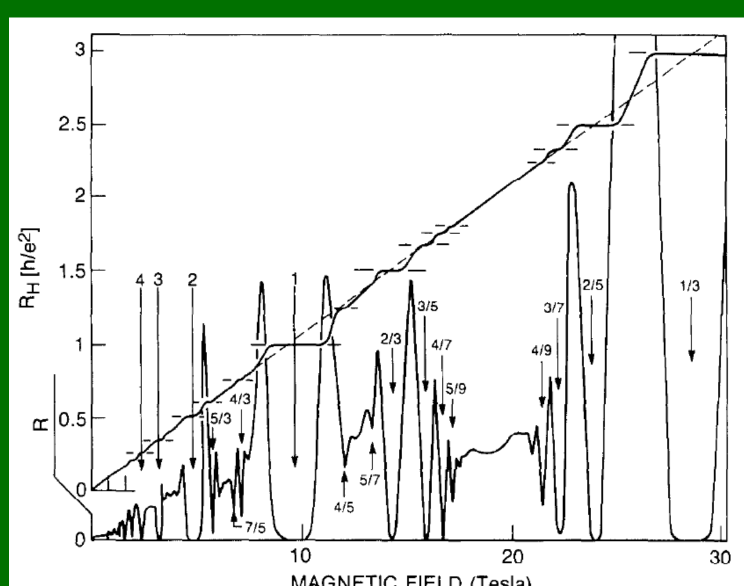
arXiv: 2212.09782

# Outline

- Matrix Product States
- Time Evolving Block Decimation (TEBD)
- QR based truncation
- Benchmark

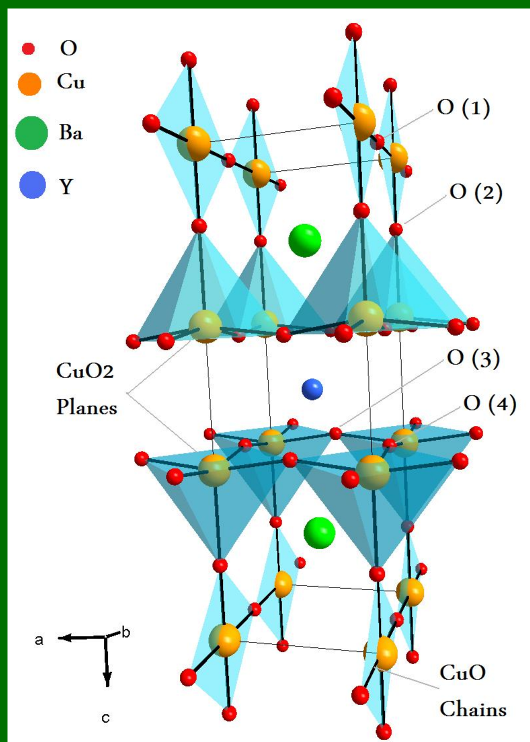
# Motivation

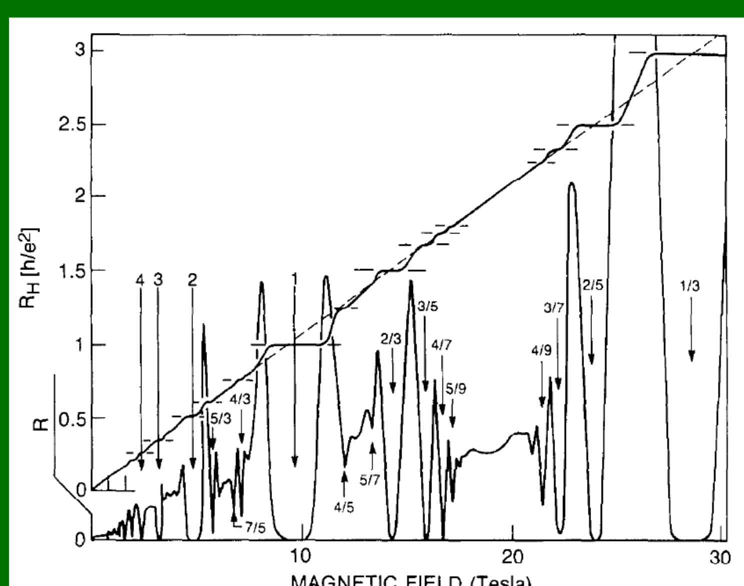



Model

# Motivation



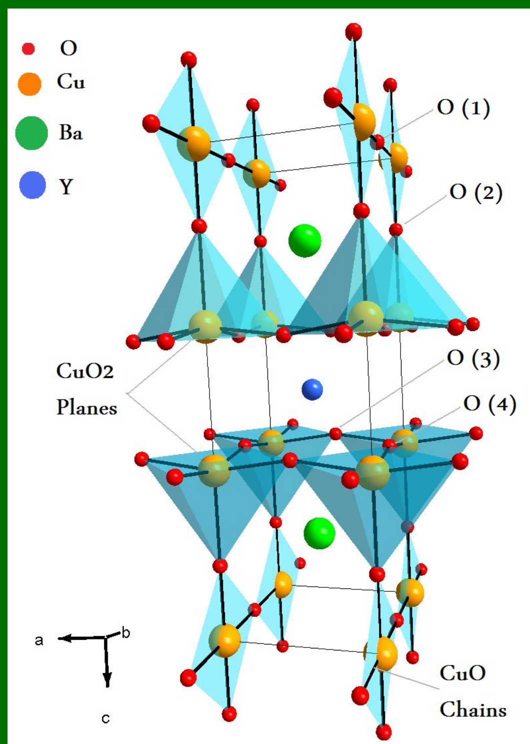

Model

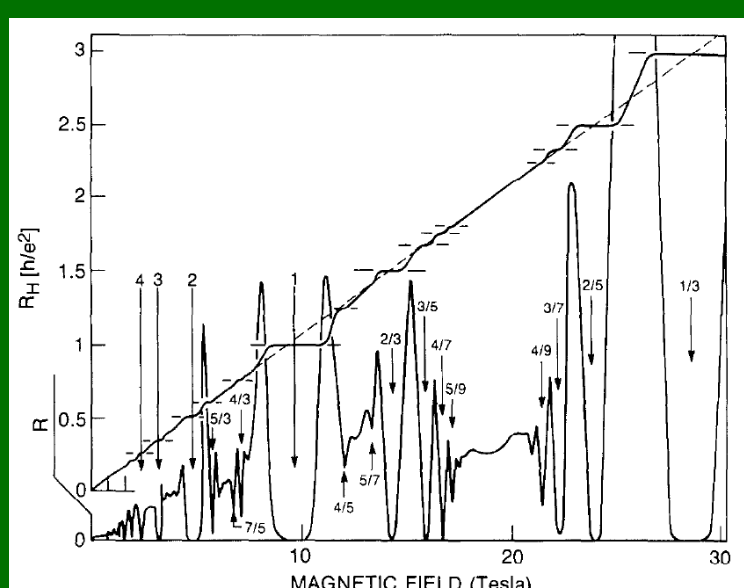
Numerically Solve

$$\hat{H}\,|\psi_0\rangle = E_0\,|\psi_0\rangle$$

$$i\frac{\partial}{\partial t}\,|\psi(t)\rangle = \hat{H}\,|\psi(t)\rangle$$

# Motivation




Model

Numerically Solve

$$\hat{H}\,|\psi_0\rangle = E_0\,|\psi_0\rangle$$

$$i\frac{\partial}{\partial t}\,|\psi(t)\rangle = \hat{H}\,|\psi(t)\rangle$$

$$|\psi\rangle \in \mathscr{H} = \bigotimes_{i=0}^{N} \mathbb{C}^d$$

challenge: $\dim \mathscr{H} = d^N$

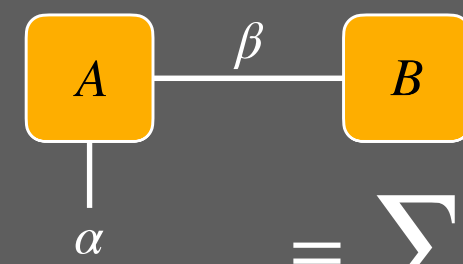
# Matrix Product States

challenge: $\dim \mathscr{H} = d^N$

# Matrix Product States

challenge: $\dim \mathscr{H} = d^N$

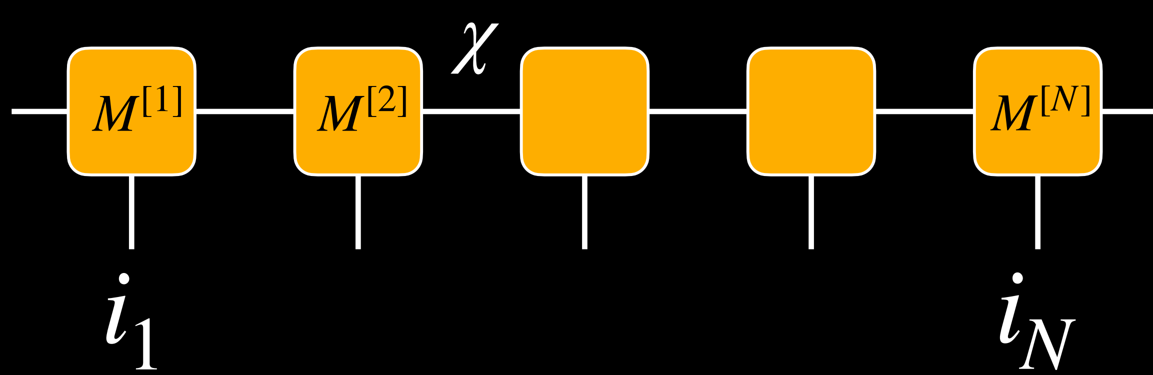
## Graphical Notation




$$= \sum_\beta A_{\alpha\beta} B_\beta$$

Variational Ansatz

$$|\psi\rangle = \sum_{i_1,\ldots i_N}$$

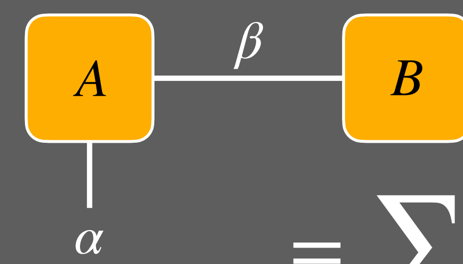



$$|i_1,\ldots,i_N\rangle$$

# Matrix Product States

challenge: $\dim \mathscr{H} = d^N$

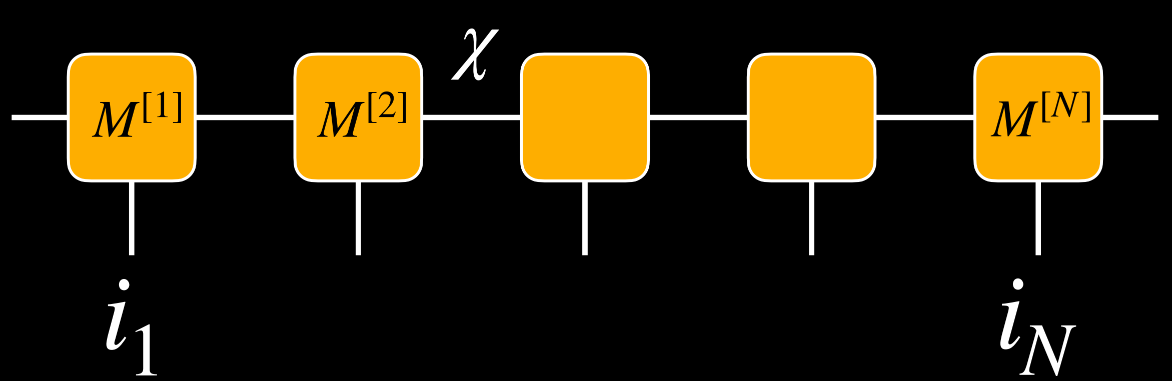
### Graphical Notation




$= \sum_\beta A_{\alpha\beta} B_\beta$

Variational Ansatz

$|\psi\rangle = \sum_{i_1,\ldots i_N}$




$|i_1, \ldots, i_N\rangle$

**Area Law**

For groundstates of gapped, local Hamiltonians

$S_{\mathrm{vN}}(A) \propto \partial A$

[Hastings 2007]

Can keep $\chi$ fixed *independent* of $L$

$\rightarrow \mathcal{O}(Ld\chi^2) \ll \mathcal{O}(d^L)$ parameters

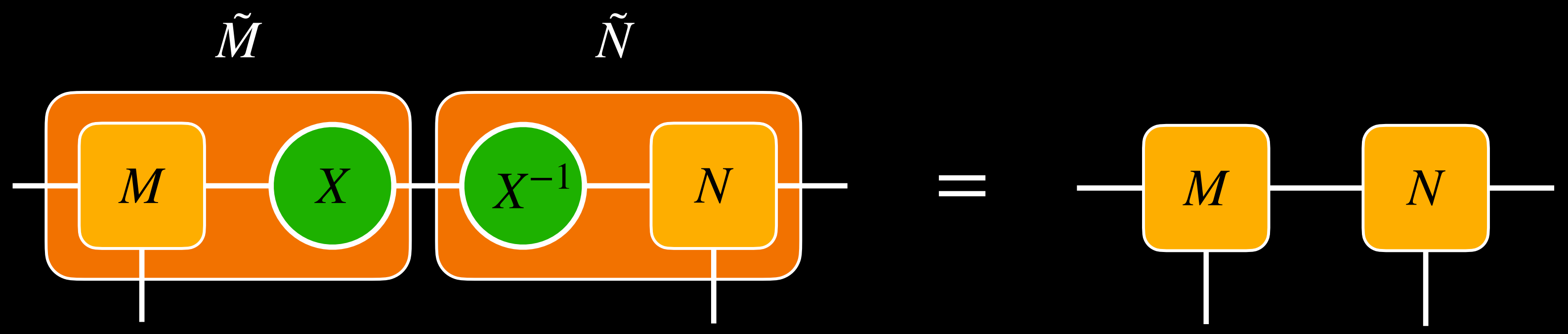
# MPS Crash Course

github.com/tenpy

# MPS Crash Course

github.com/tenpy

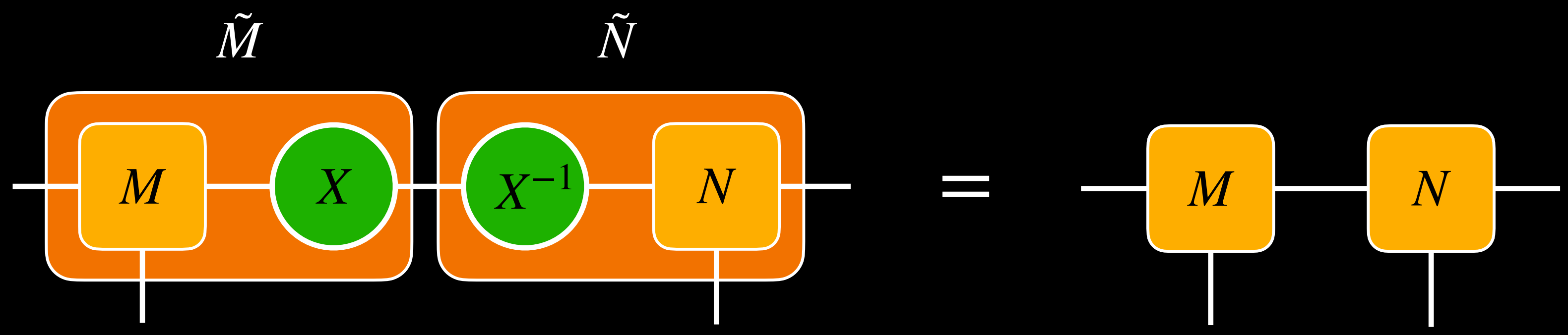
## Gauge Freedom



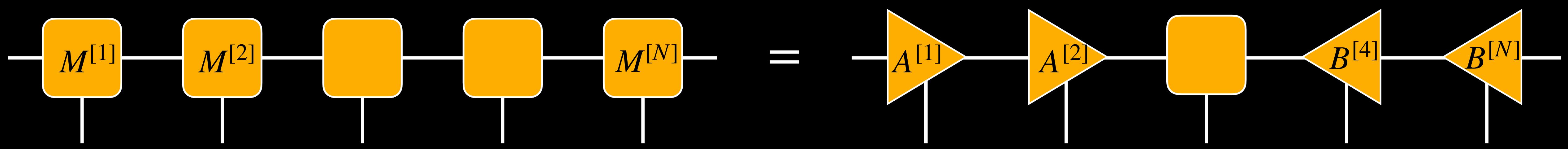
# MPS Crash Course

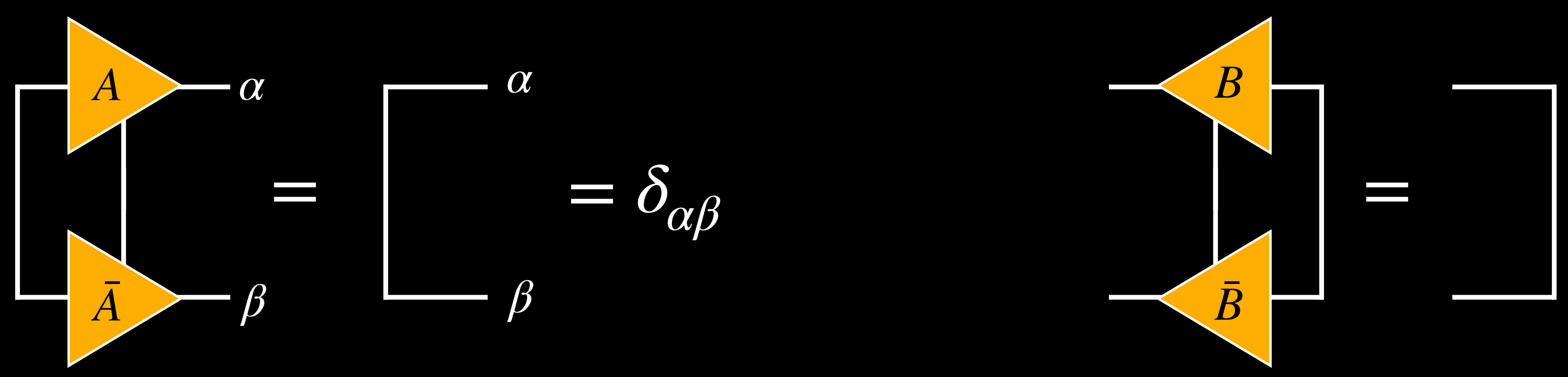
TeNPy
github.com/tenpy

## Gauge Freedom



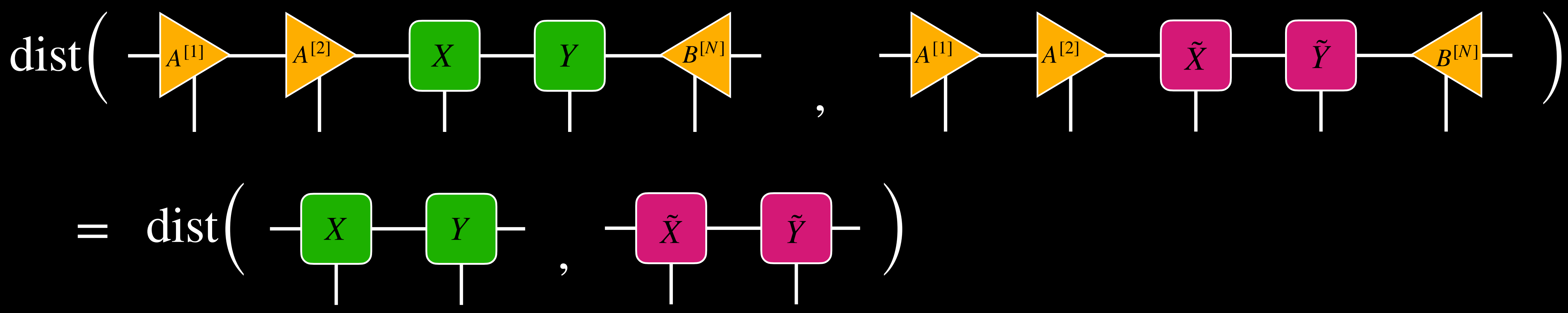

## → Isometric Form



# MPS Crash Course

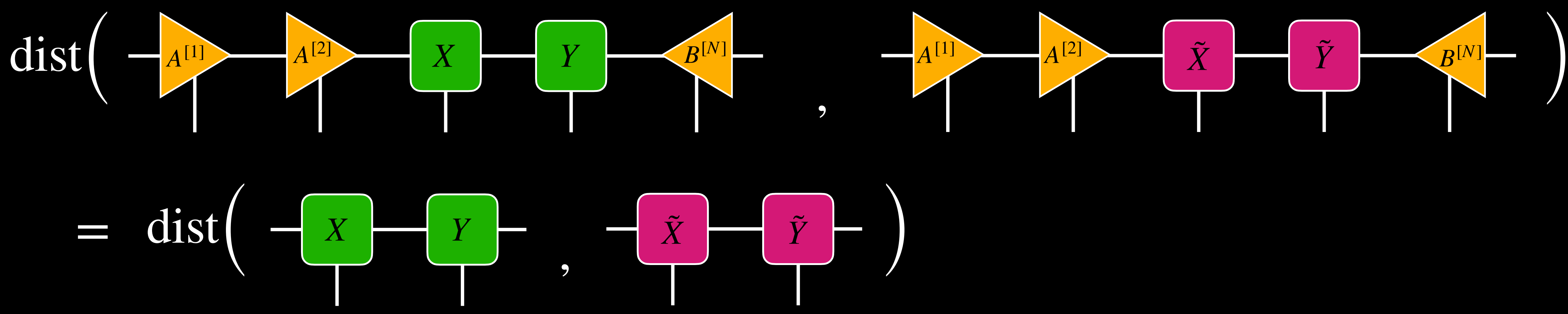
## Isometric Form




github.com/tenpy

# MPS Crash Course

## Isometric Form



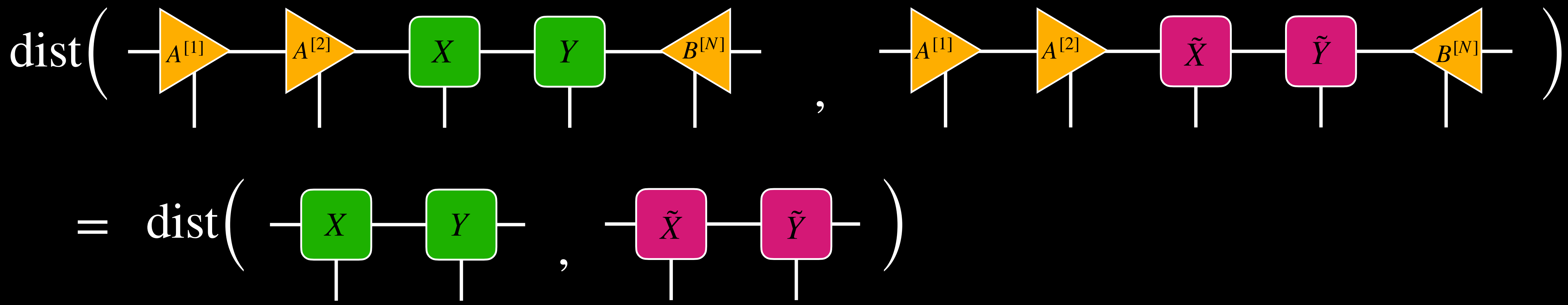

## Matrix Product States

- Groundstates (DMRG, iDMRG, VUMPS)
- Dynamics (TEBD, TDVP)
- Open/Thermal Systems (MPDO, purification, ...)
- Excitations (tangent space, spectral functions)
- 1d chains, 2d lattices, ...

github.com/tenpy

# MPS Crash Course

## Isometric Form

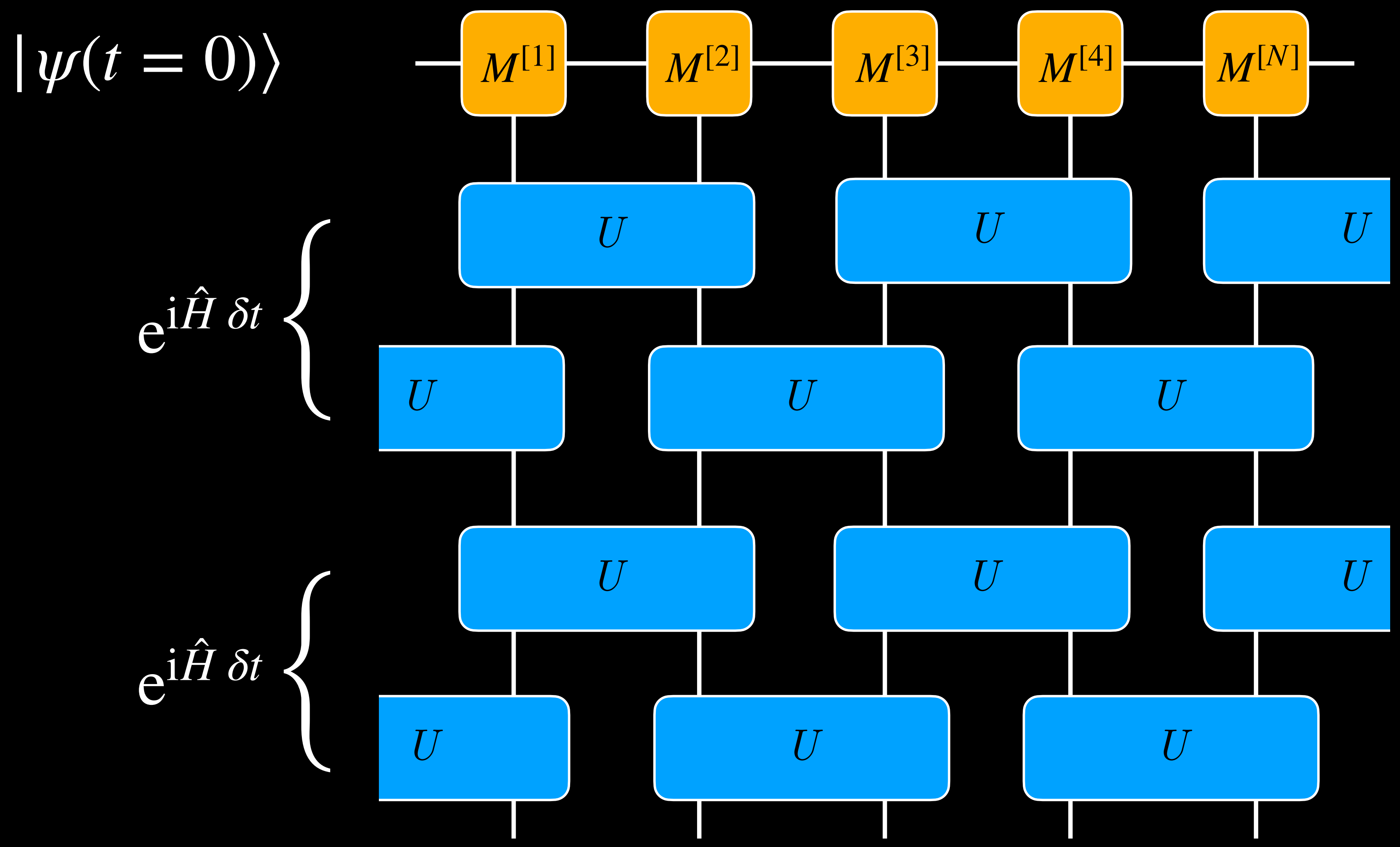



## Matrix Product States

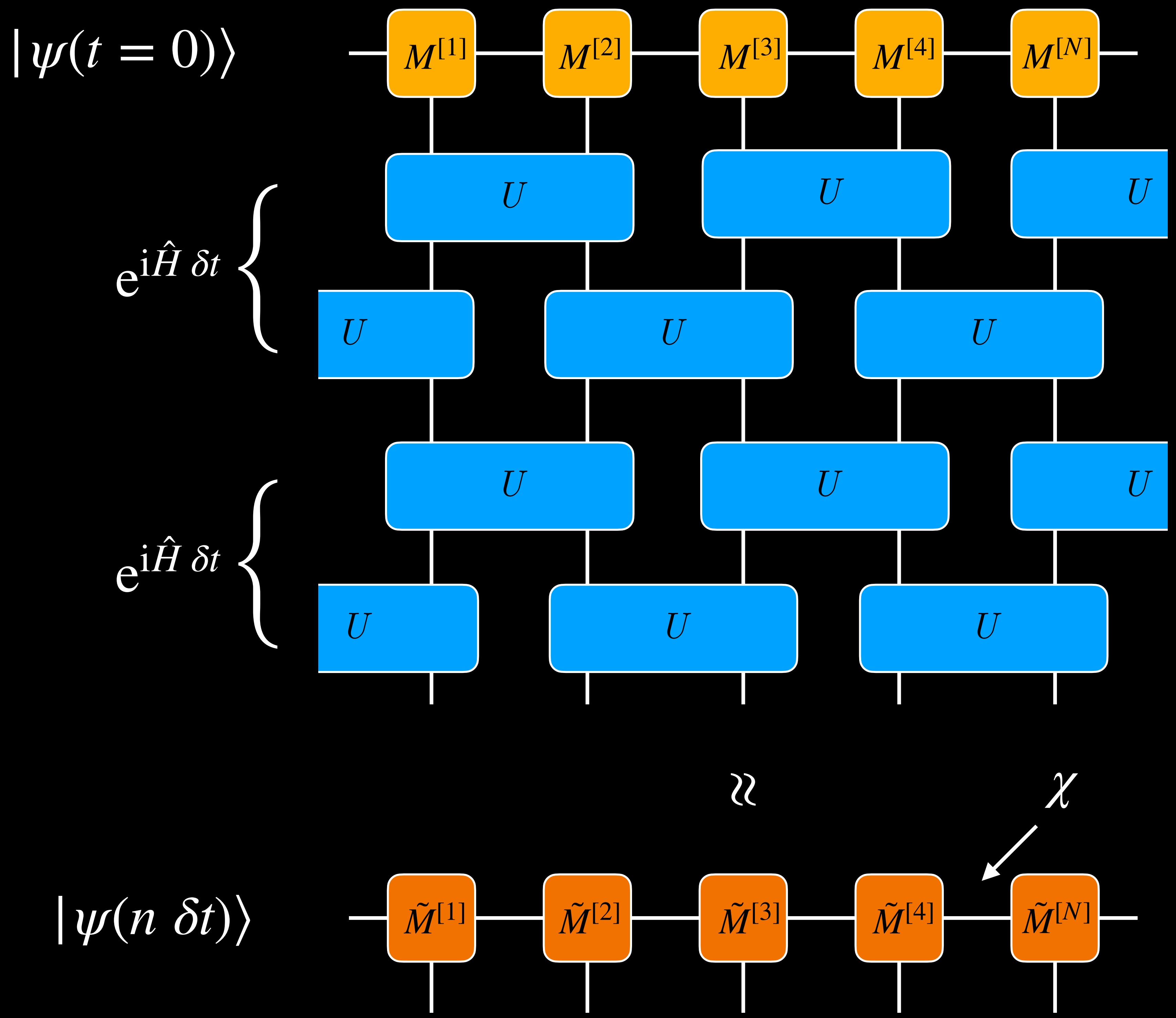
- Groundstates (DMRG, iDMRG, VUMPS)
- Dynamics (**TEBD**, TDVP)
- Open/Thermal Systems (MPDO, purification, ...)
- Excitations (tangent space, spectral functions)
- 1d chains, 2d lattices, ...

TeNPy
github.com/tenpy

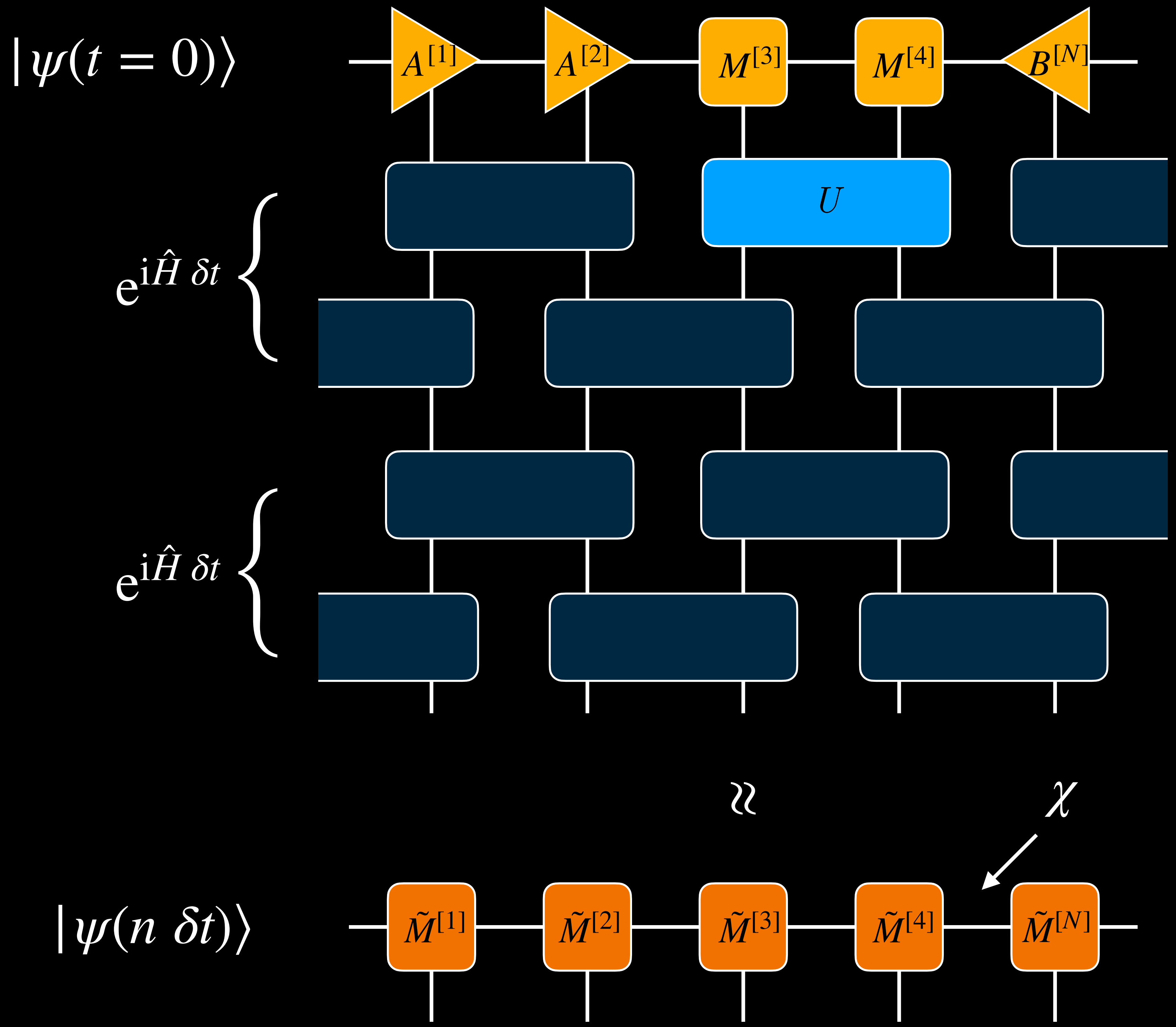
# Time Evolving Block Decimation (TEBD)



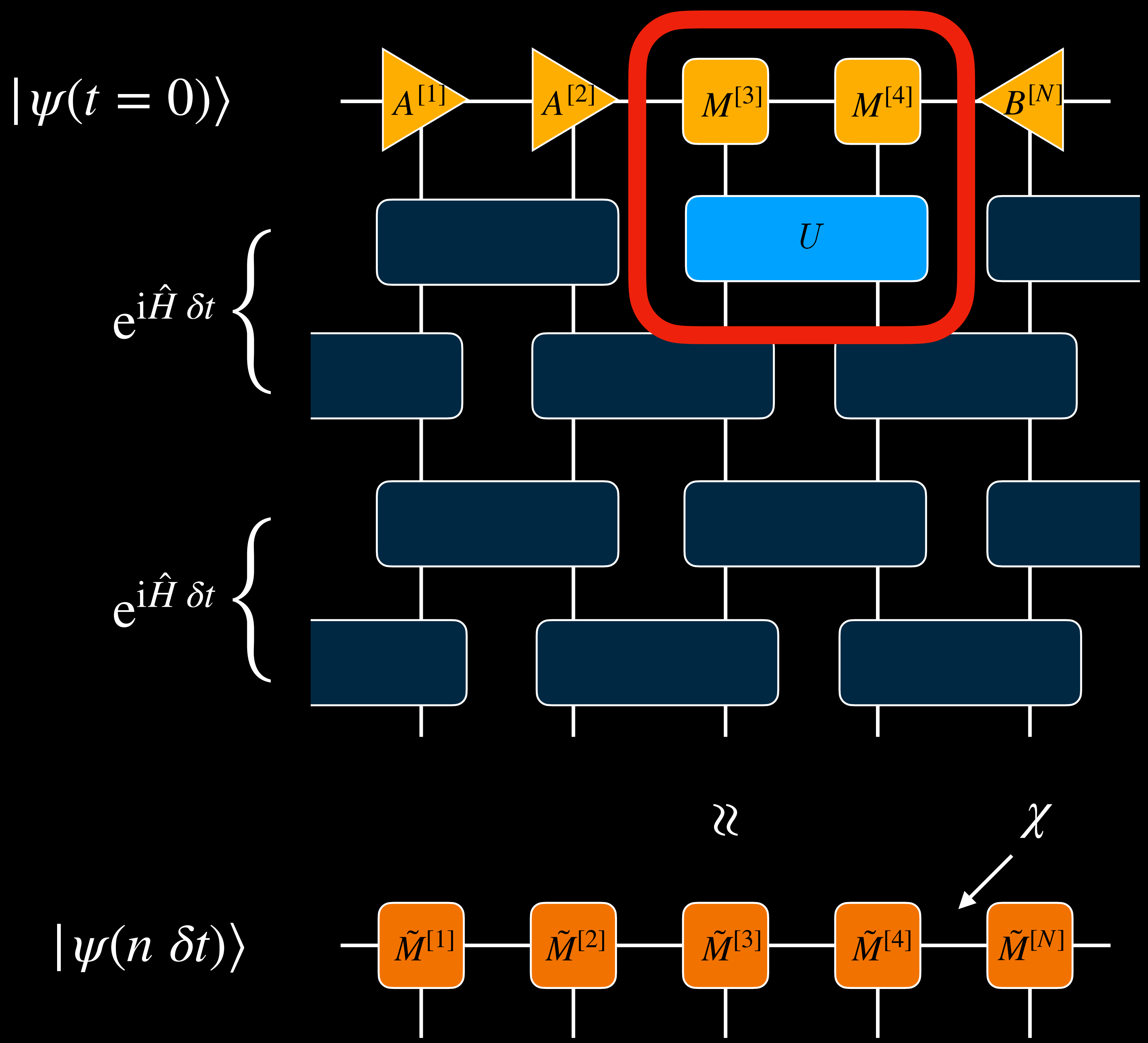
# Time Evolving Block Decimation (TEBD)



# Time Evolving Block Decimation (TEBD)

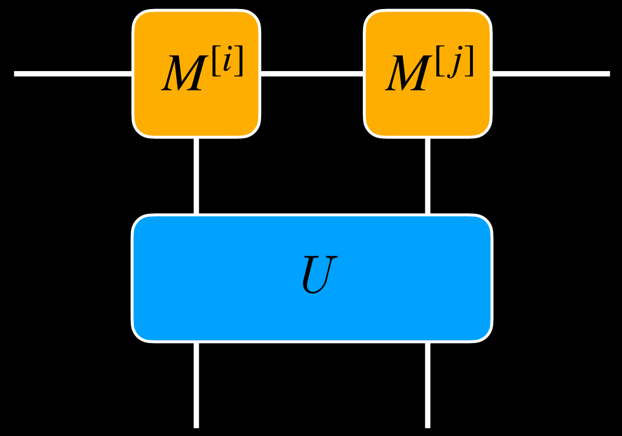


# Time Evolving Block Decimation (TEBD)



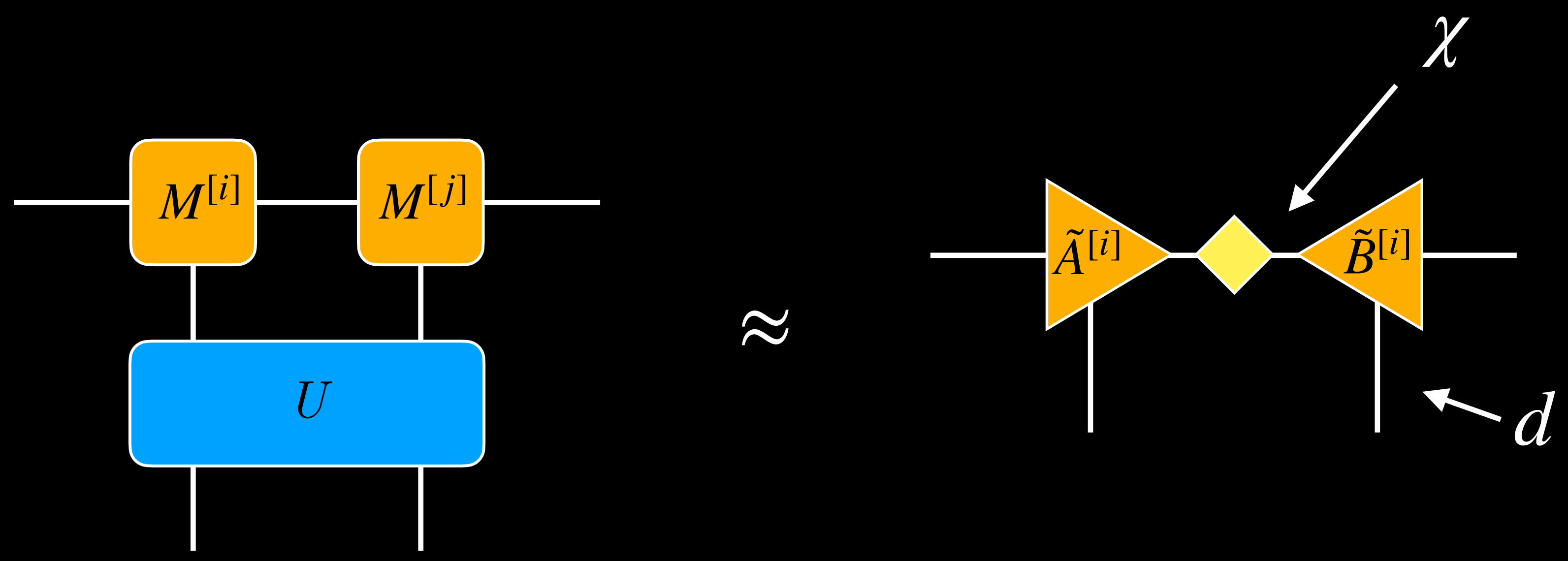
# Time Evolving Block Decimation (TEBD)

SVD-based Truncation:



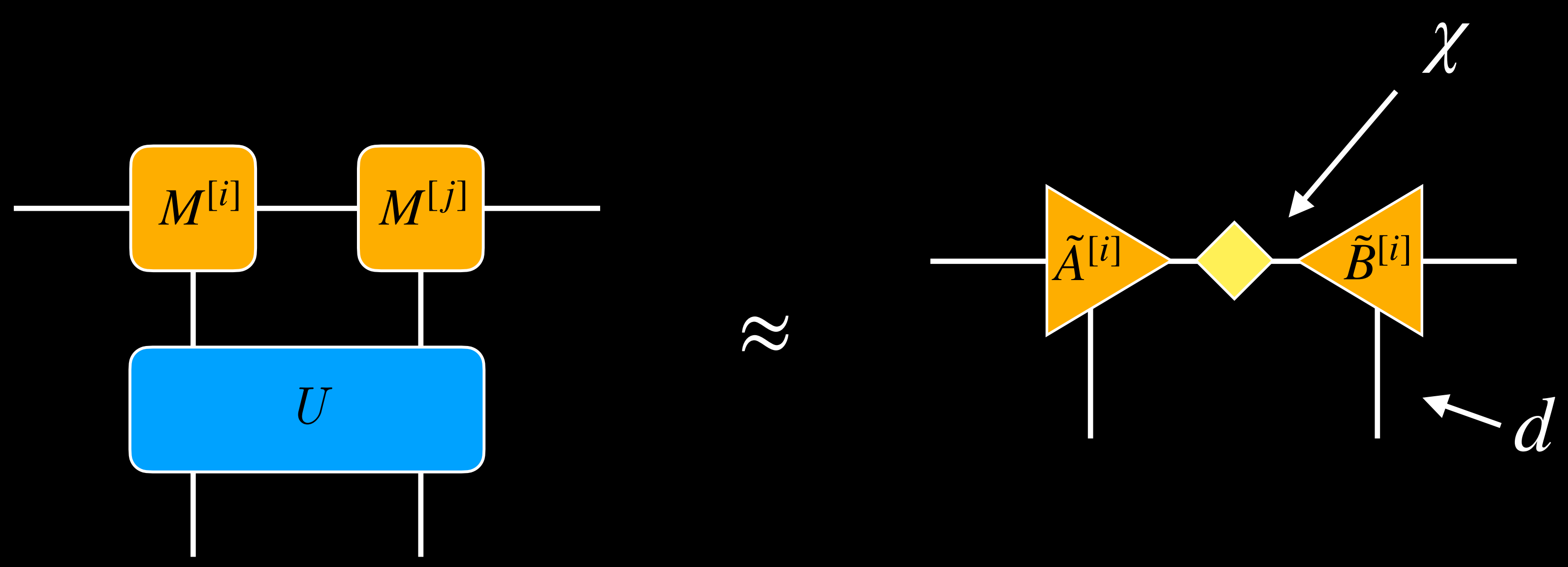
# Time Evolving Block Decimation (TEBD)

SVD-based Truncation:



# Time Evolving Block Decimation (TEBD)

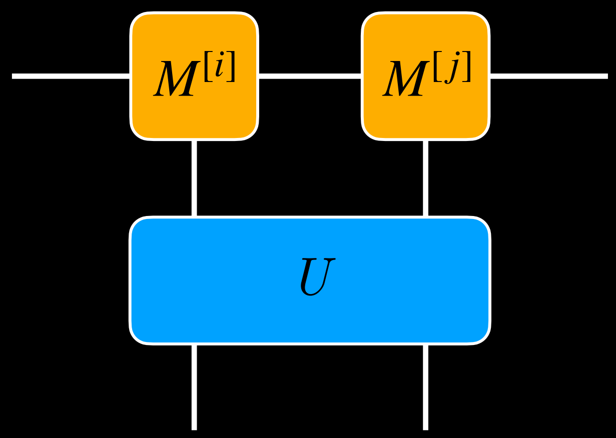
SVD-based Truncation:




- Truncated SVD: Keep only $\chi$ singular values and vectors
- Provably optimal low-rank approximation
- Cost to compute $\mathcal{O}(d^3\chi^3)$

# Time Evolving Block Decimation (TEBD)

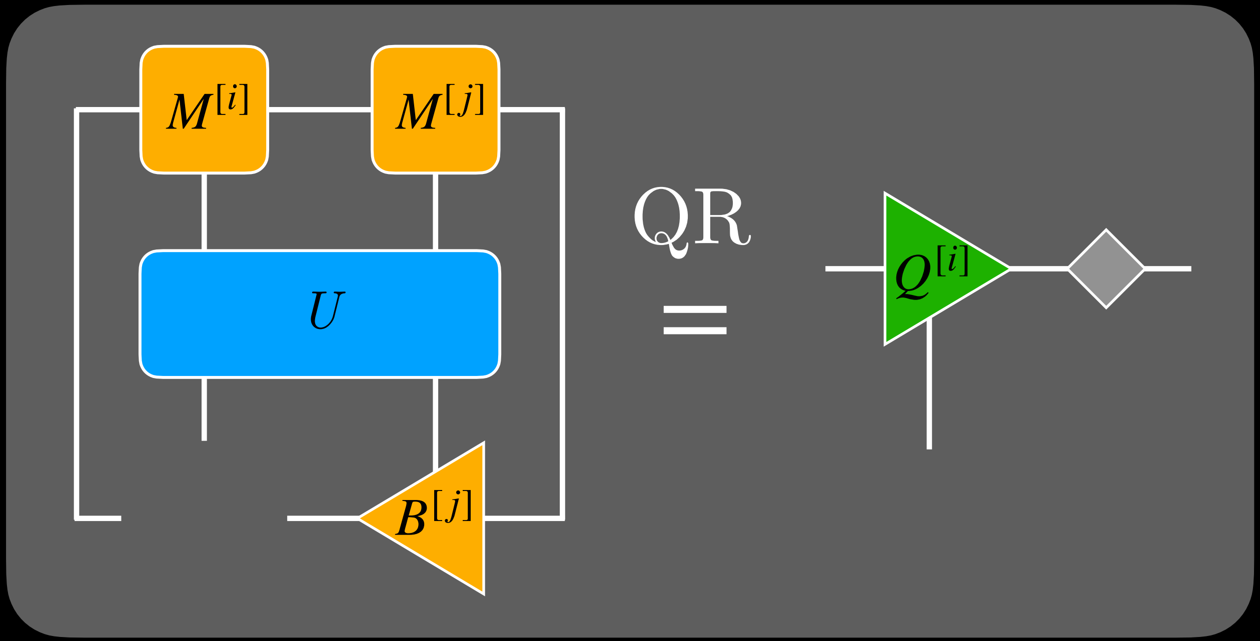
QR-based Truncation:

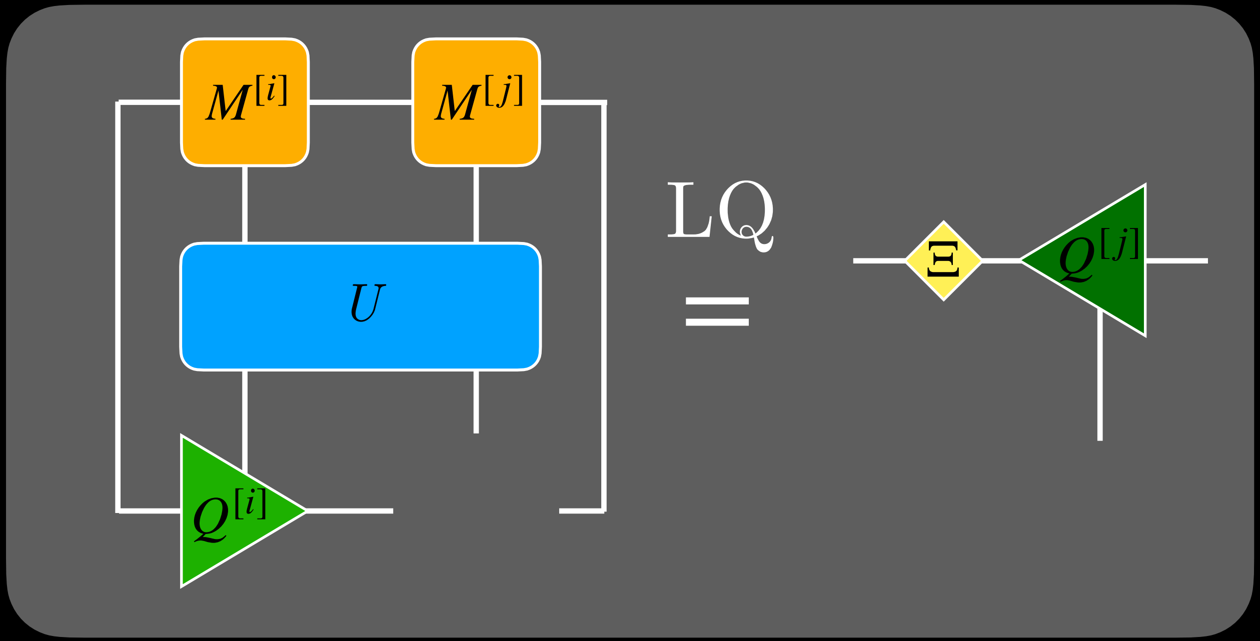


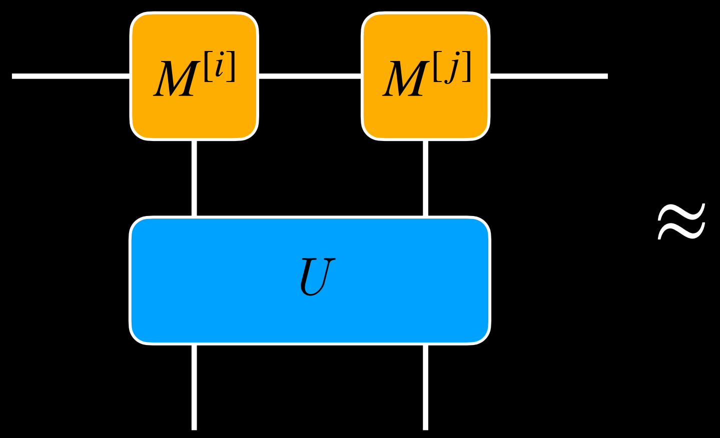

$\approx$

# Time Evolving Block Decimation (TEBD)

QR-based Truncation:

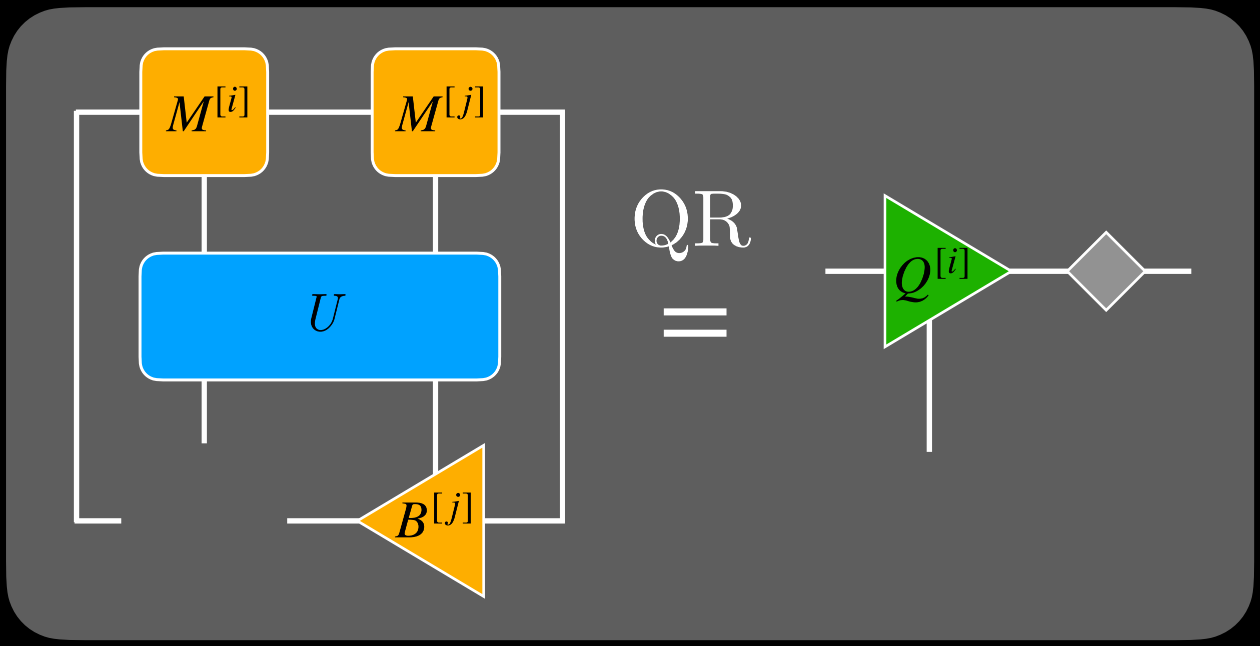


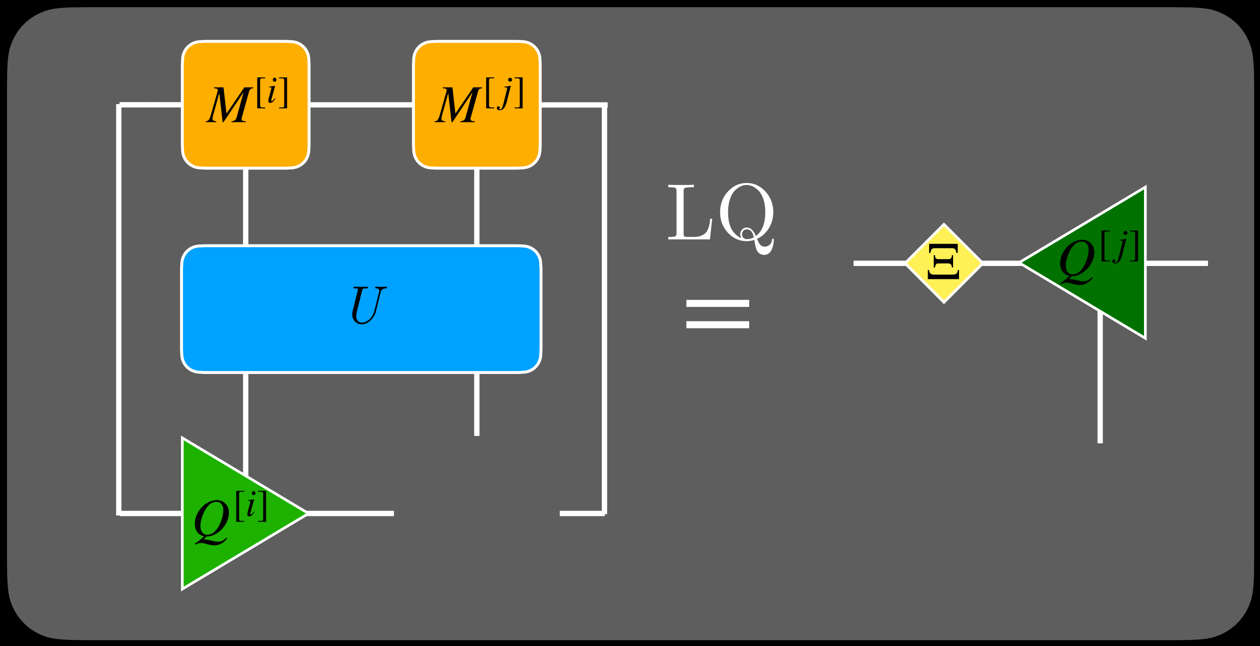



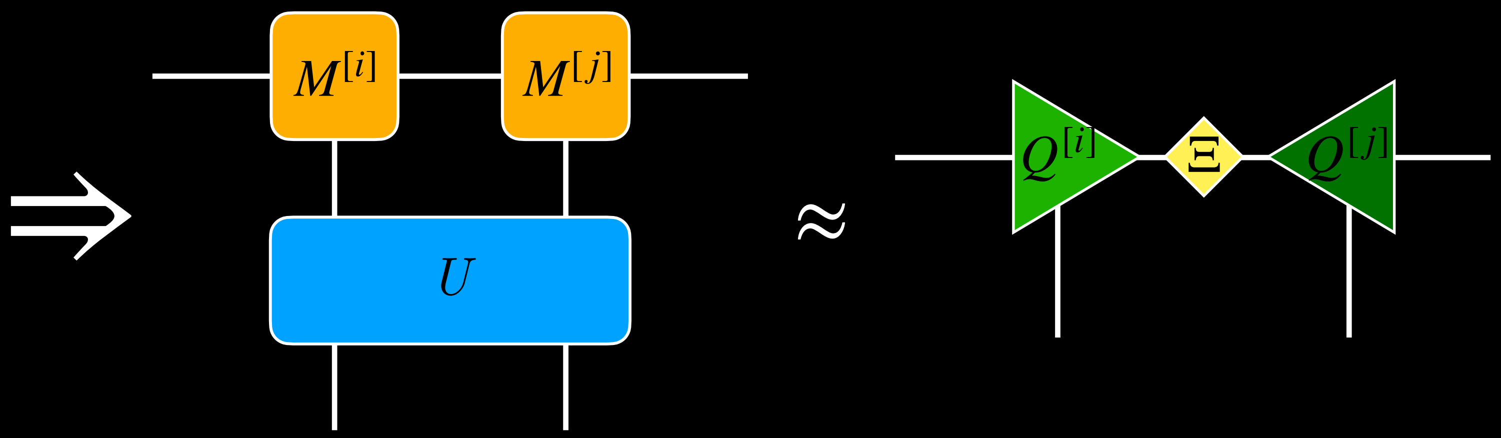



# Time Evolving Block Decimation (TEBD)

QR-based Truncation:

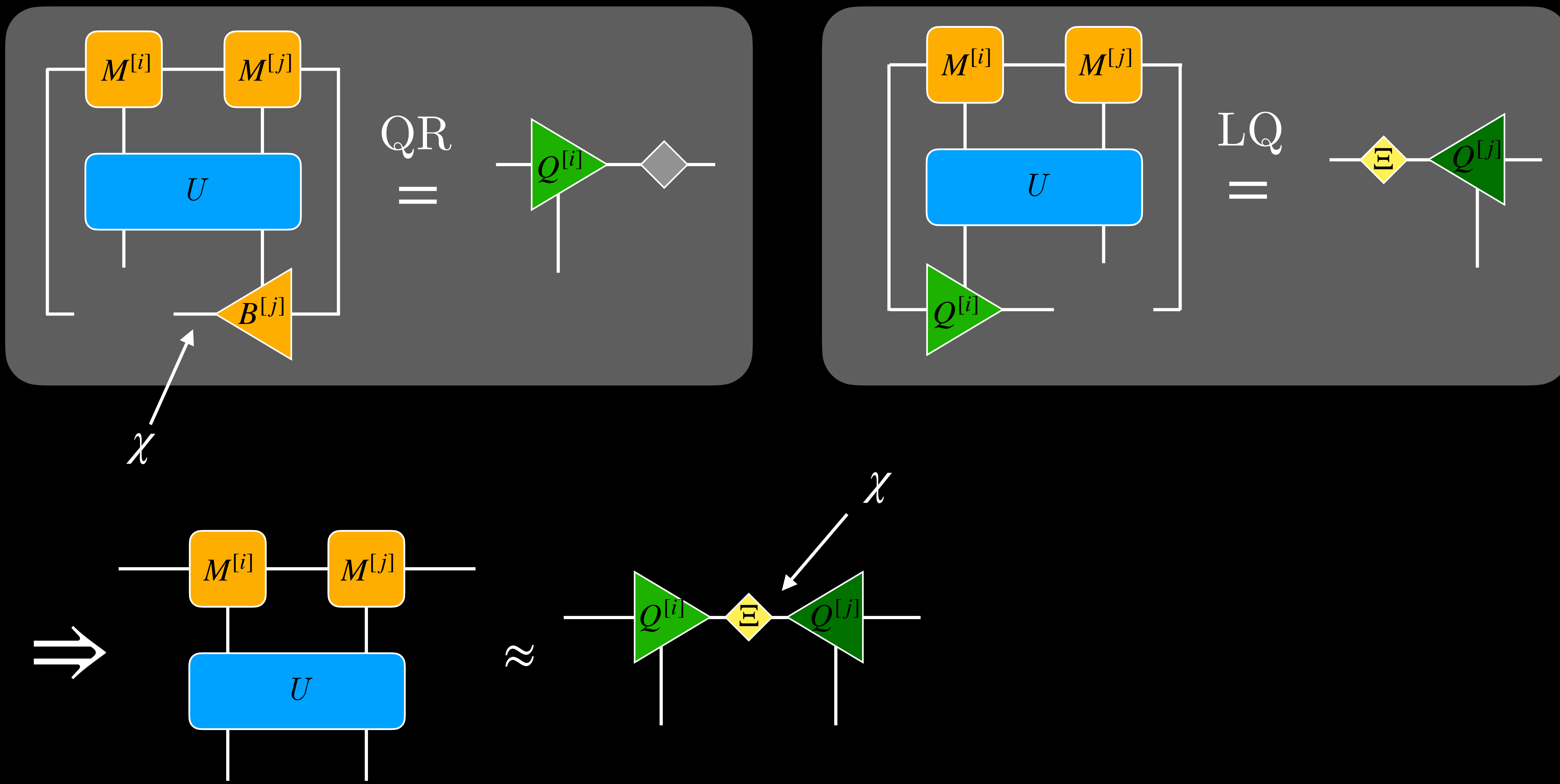


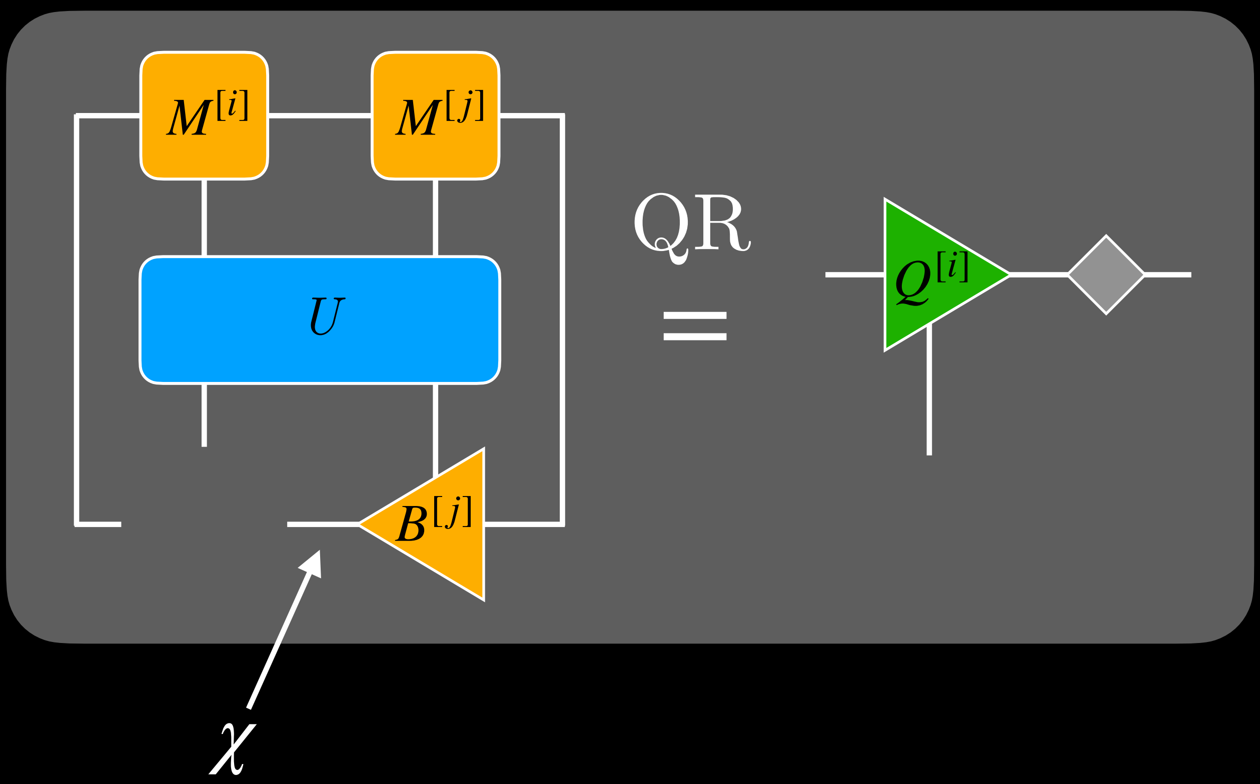
# Time Evolving Block Decimation (TEBD)

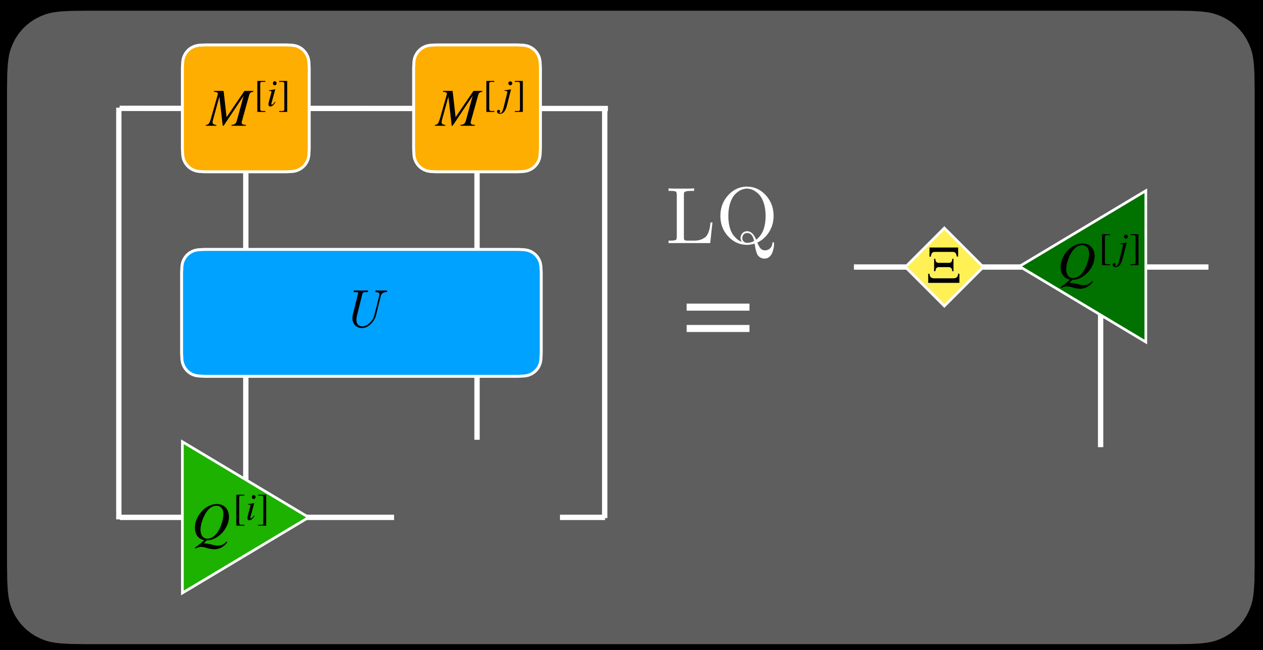
## QR-based Truncation:

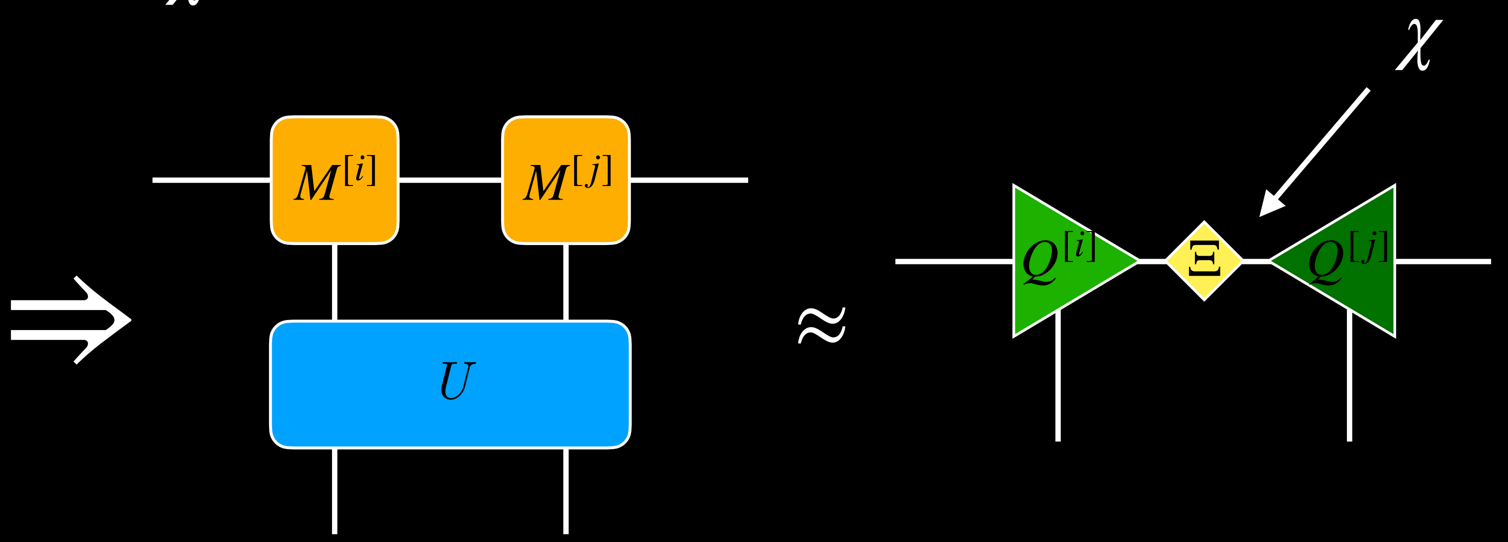


# Time Evolving Block Decimation (TEBD)

QR-based Truncation:




- Empirically: same accuracy
- Cost to compute $\mathcal{O}(d^2\chi^3)$
- $\chi$ can grow (not shown)

# Benchmark

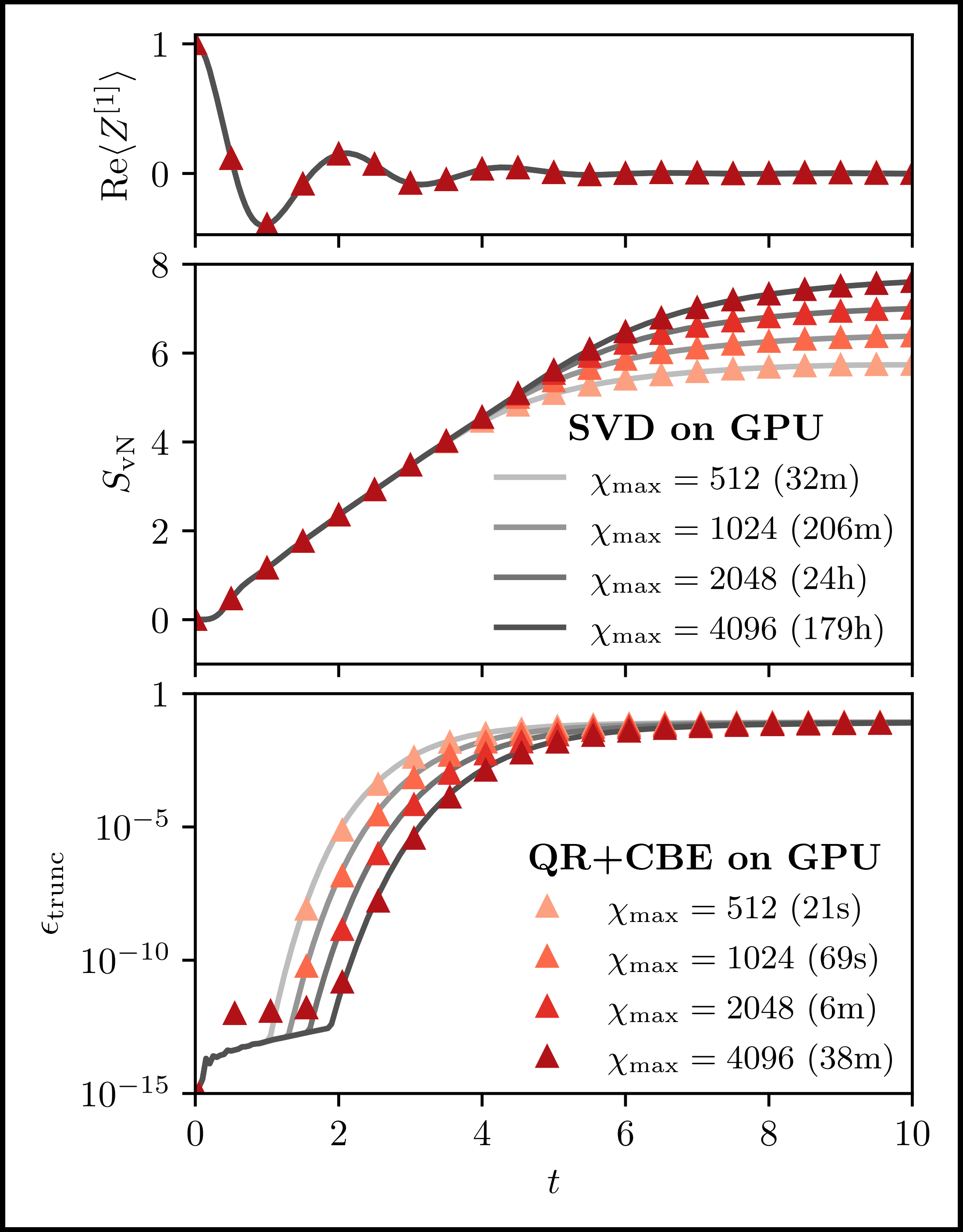
## Quantum Clock Model: quench $g = 0 \;\rightarrow\; g = 2$

$$H = -\sum_{<i,j>} (Z_i Z_j^\dagger + \mathrm{h.c.}) - g \sum_i (X_i + \mathrm{h.c.})$$

# Benchmark

Quantum Clock Model: quench $g = 0 \;\rightarrow\; g = 2$

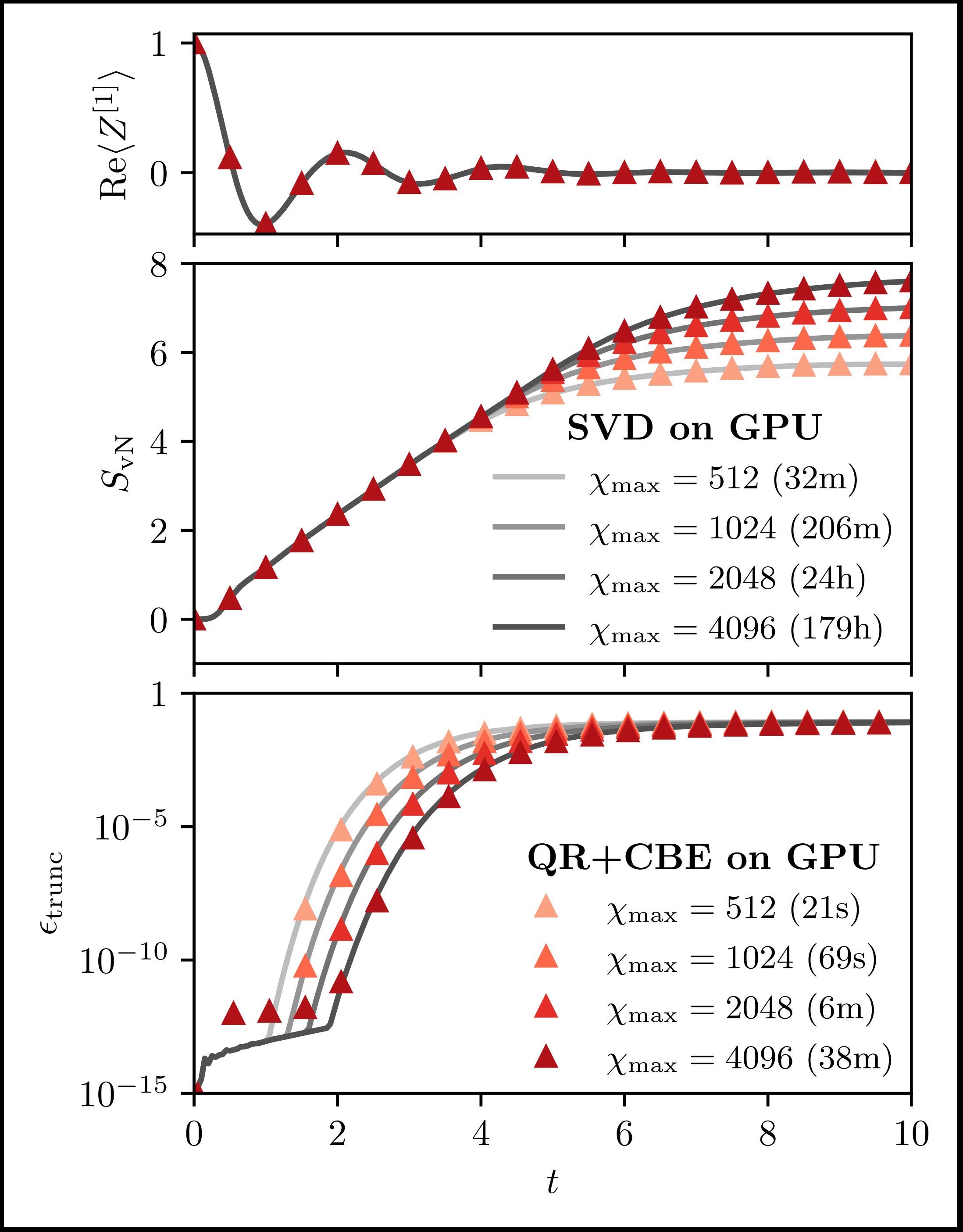
$$H = -\sum_{<i,j>} (Z_i Z_j^\dagger + \mathrm{h.c.}) - g \sum_i (X_i + \mathrm{h.c.})$$



# Benchmark

Quantum Clock Model: quench $g = 0 \rightarrow g = 2$

$$H = -\sum_{<i,j>} (Z_i Z_j^\dagger + \mathrm{h.c.}) - g \sum_i (X_i + \mathrm{h.c.})$$

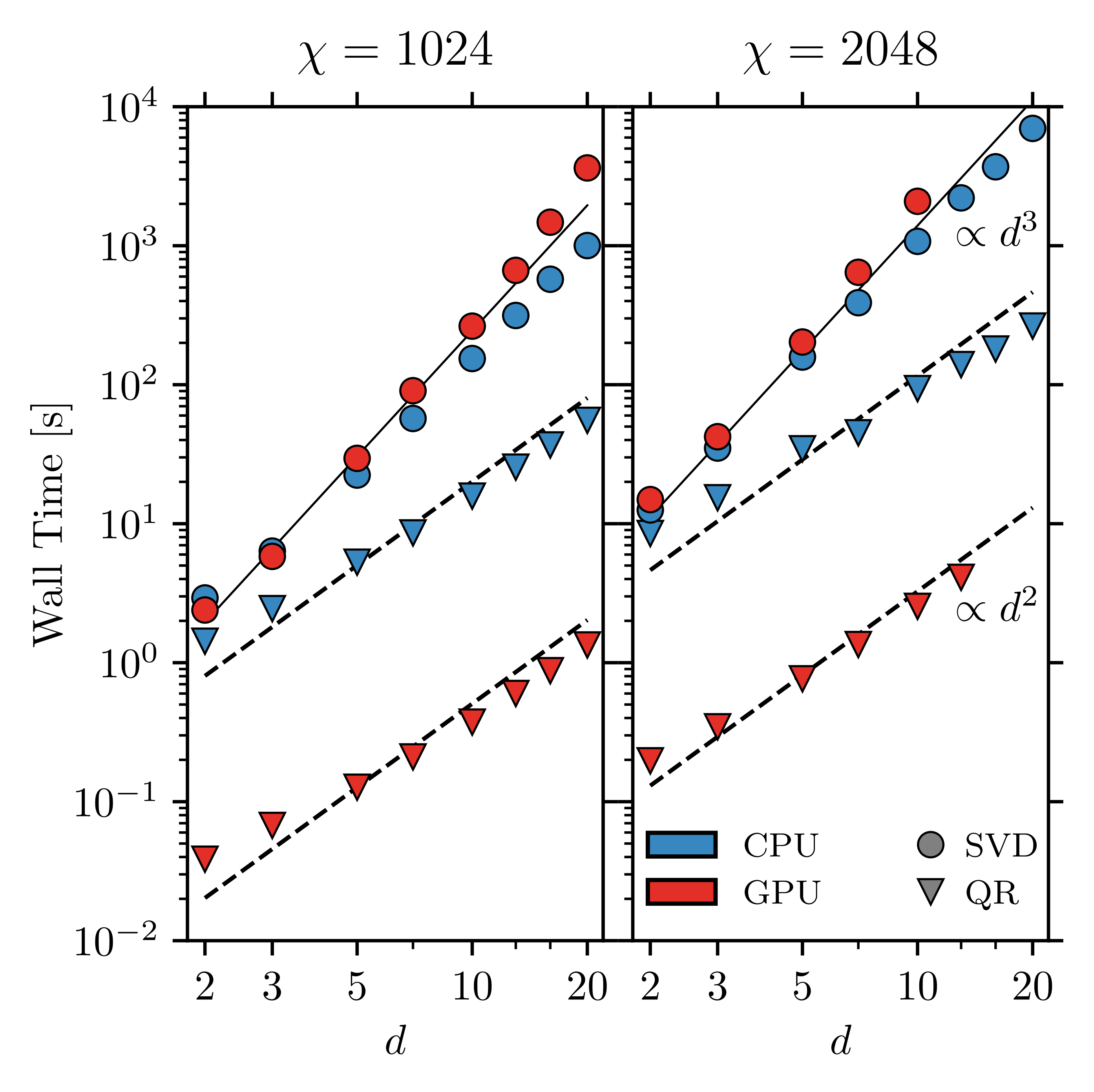



- Excellent agreement of observables with SVD based TEBD
- Much faster on GPU

# Benchmark

## Quantum Clock Model: quench $g = 0 \rightarrow g = 2$

$$H = -\sum_{<i,j>} (Z_i Z_j^\dagger + \mathrm{h.c.}) - g \sum_i (X_i + \mathrm{h.c.})$$

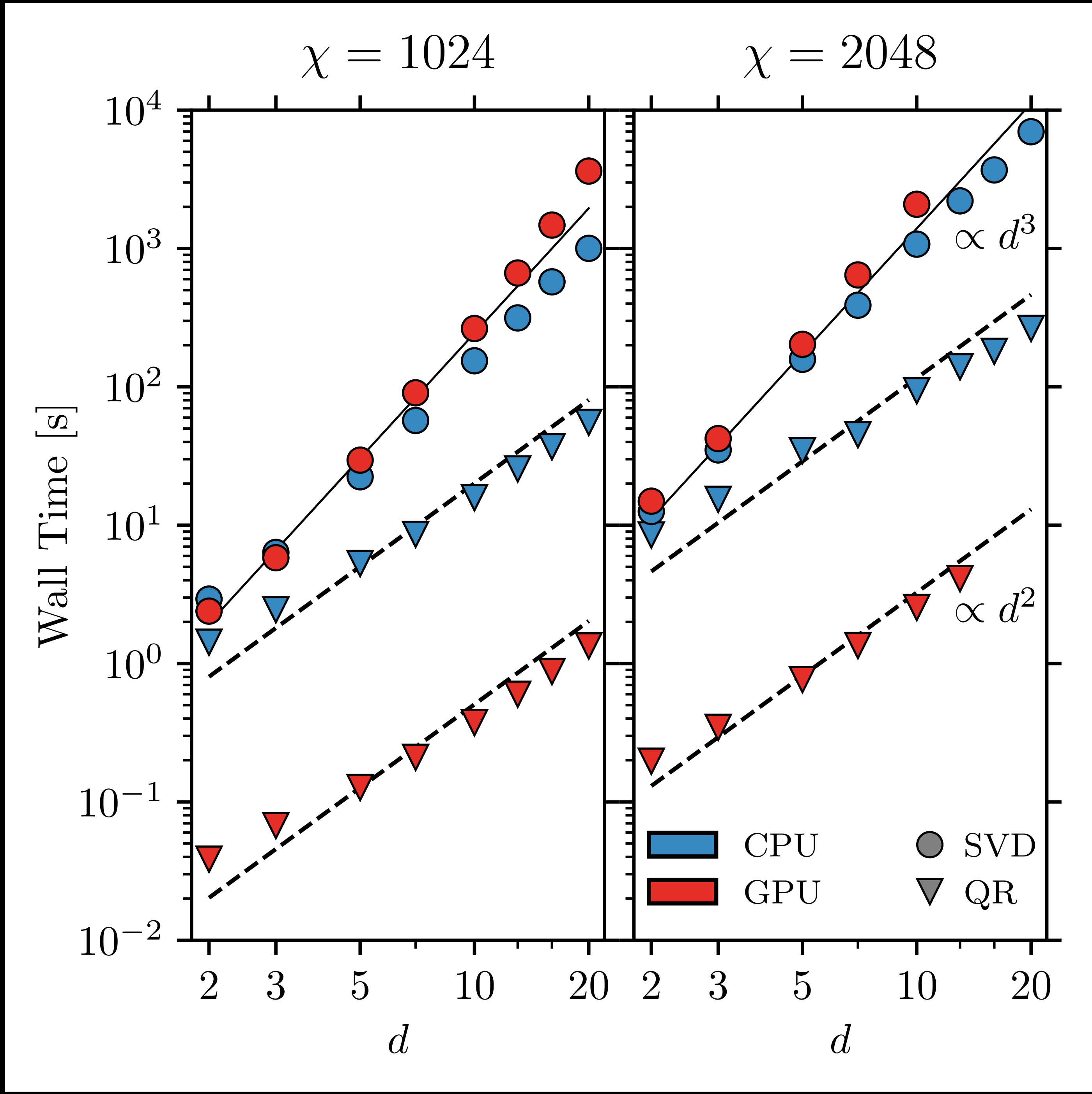


# Benchmark

Quantum Clock Model: quench $g = 0 \rightarrow g = 2$

$$H = -\sum_{<i,j>} (Z_i Z_j^\dagger + \mathrm{h.c.}) - g \sum_i (X_i + \mathrm{h.c.})$$

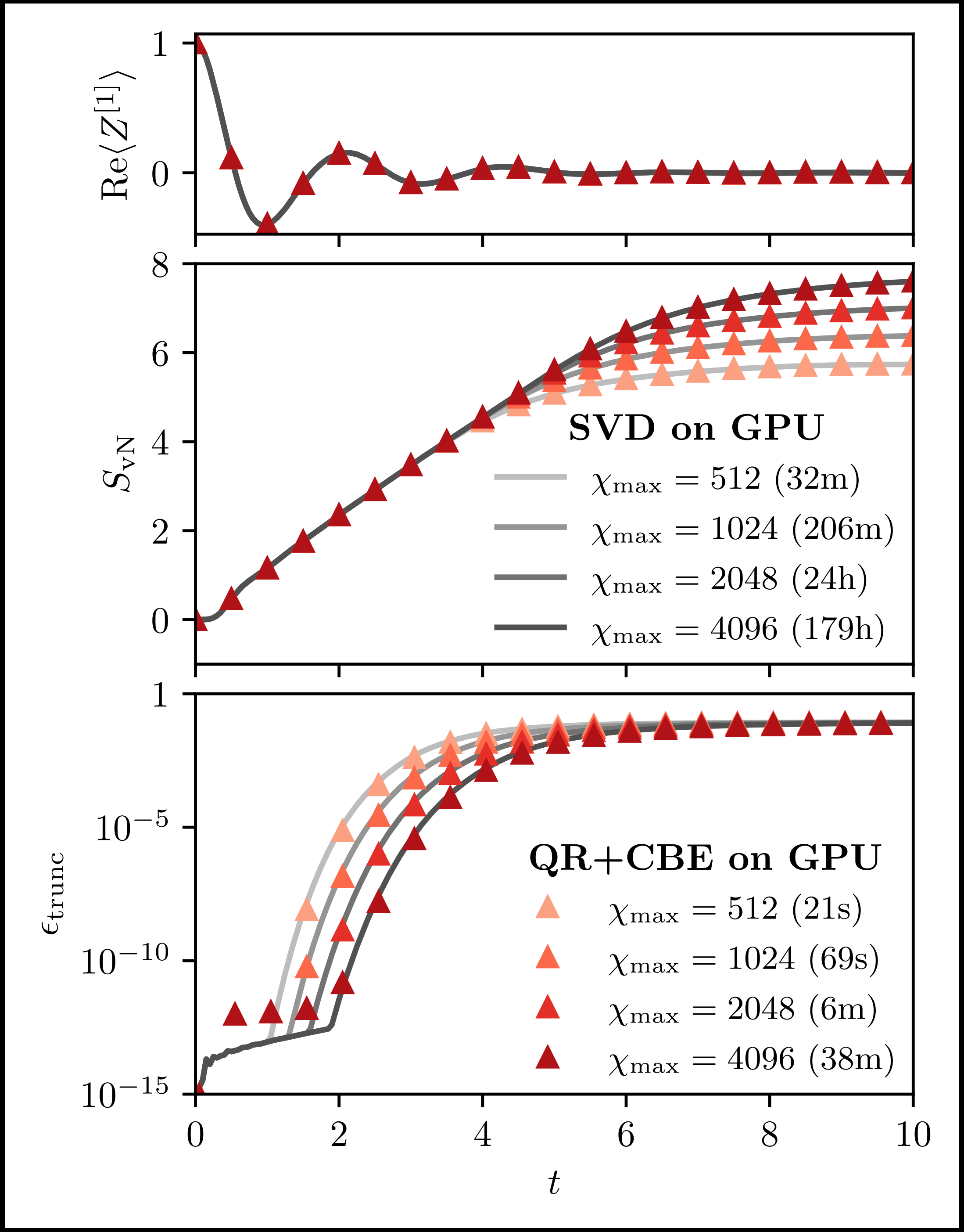
- Improved scaling $d^2$ instead of $d^3$
- CPU: speedup $\sim 10$ at larger $d$
- GPU: great speedup at all $d$, up to **750x**

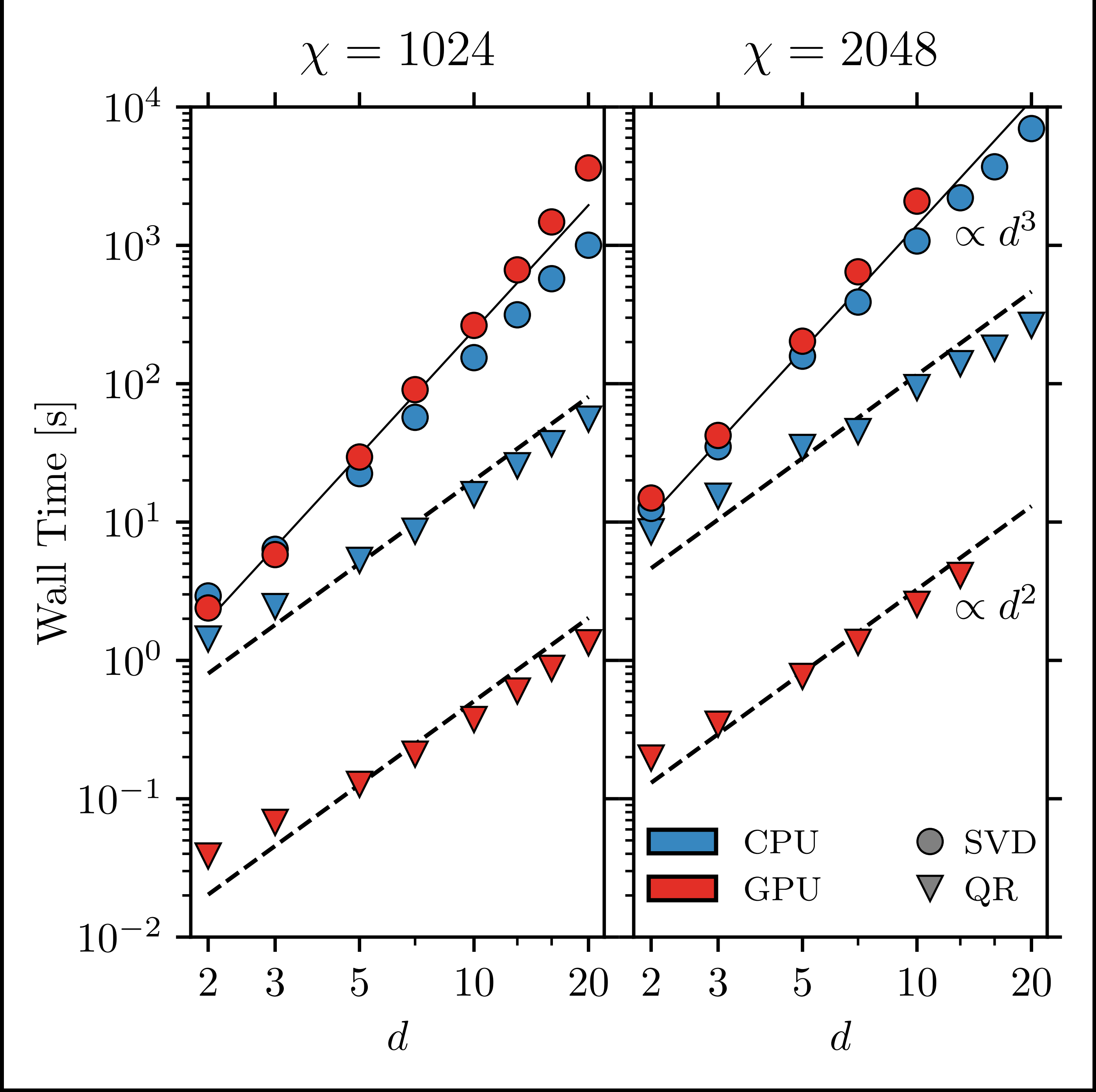


# Benchmark

Quantum Clock Model: quench $g = 0 \rightarrow g = 2$

$$H = -\sum_{<i,j>} (Z_i Z_j^\dagger + \mathrm{h.c.}) - g \sum_i (X_i + \mathrm{h.c.})$$



# Conclusion

arXiv: 2212.09782

- QR based truncation step for time-evolution of matrix product states as alternative to SVD based truncation
- Improved scaling $\mathcal{O}(d^3\chi^3) \rightarrow \mathcal{O}(d^2\chi^3)$ with dimension $d$ of local Hilbert space
- Unlike SVD, QR is accelerated on GPUs
  - Speedups of up to **750x**