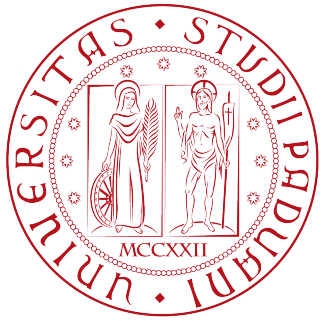



University of Cyprus

# Solving QUBO problems with tensor networks

Rocco Barač
*(with Ilaria Siloi, Marco Tesoro and Simone Montangero)*



Funded by the European Union

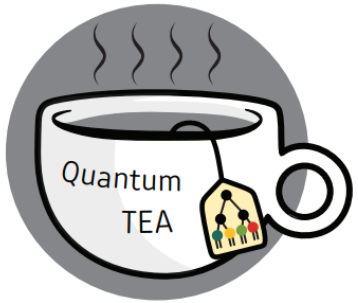
AQTIVATE

The *AQTIVATE* project receives funding from the European Union's HORIZON MSCA Doctoral Networks programme, under Grant Agreement No. 101072344.

# Introduction

- Optimization problems are everywhere
- Recent interest in quantum-inspired methods
- Solving optimization problems with tensor networks
- QUBO problems as ground-state search of spin glass Hamiltonians
- We solve Sherrington-Kirkpatrick (SK) Hamiltonian problems
  - Solve via DMRG
  - Benchmark against state-of-the-art classical solvers

- Starting point: Quantum TEA





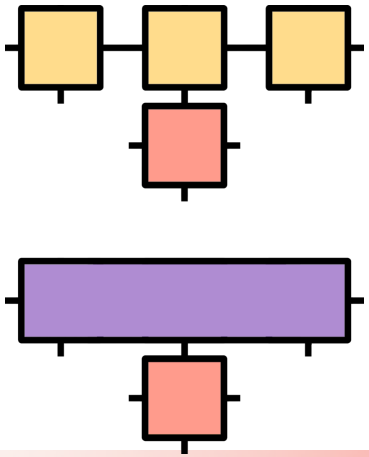


- Tensor networks are a classical way of simulating systems at low entanglement
- QUBO solver integrated in the Quantum TEA leaves library

https://www.quantumtea.it

# Tensor Network Introduction

- Quantum many-body physics
- Many systems do not have analytical solutions
- Exponential Hilbert space growth problem – storing a state vector of 30 qubits already requires ~ 17 GB of memory
- Numerical solutions using tensor networks (TNs) – time evolution and statics (such as ground state search)



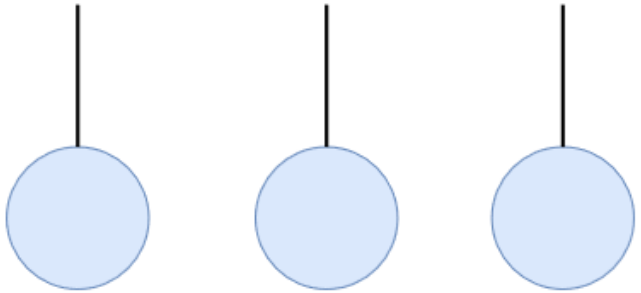
Tensor networks example

Image source:
https://en.wikipedia.org/wiki/Tensor_network

# Tensor networks basics

- Mathematical framework for working with tensors
- Used in quantum many-body physics as a class of variational wave functions
- Easier to work with due to their intuitive diagrammatic notation
- Bond dimension: parameter controlling the specific expressivity of a tensor network (TN)

$$|\psi_{MF}\rangle = |\psi_1\rangle \otimes |\psi_2\rangle \otimes |\psi_3\rangle$$



$$T^{s_1 s_2 s_3 s_4 s_5 s_6} = \sum_{\{\alpha\}} A^{s_1}_{\alpha_1} A^{s_2}_{\alpha_1 \alpha_2} A^{s_3}_{\alpha_2 \alpha_3} A^{s_4}_{\alpha_3 \alpha_4} A^{s_5}_{\alpha_4 \alpha_5} A^{s_6}_{\alpha_5}$$

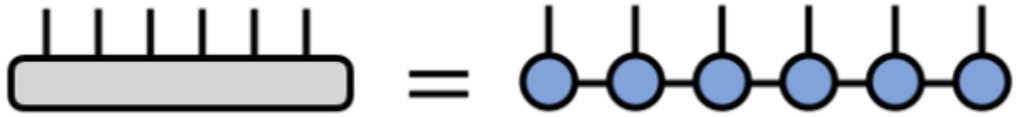


Image source: https://tensornetwork.org/mps/

# Tensor network notation basics

- Low order tensors:

vector $v_j$




matrix $M_{ij}$




3-index tensor $T_{ijk}$






Source of images: https://tensornetwork.org/diagrams/

# Tensor network notation basics

- Contractions examples:




$$= \sum_j M_{ij} v_j$$



$$= \sum_k T_{ijkl} V_{km}$$



$$= A_{ij} B_{jk} = AB$$



$$= \sum_{\alpha_1,\alpha_2,\alpha_3} A^{s_1}_{\alpha_1} B^{s_2}_{\alpha_1\alpha_2} C^{s_3}_{\alpha_2\alpha_3} D^{s_4}_{\alpha_3}$$



$$= A_{ij} B_{ji} = \mathrm{Tr}[AB]$$



Source of images: https://tensornetwork.org/diagrams/

# Optimization problems

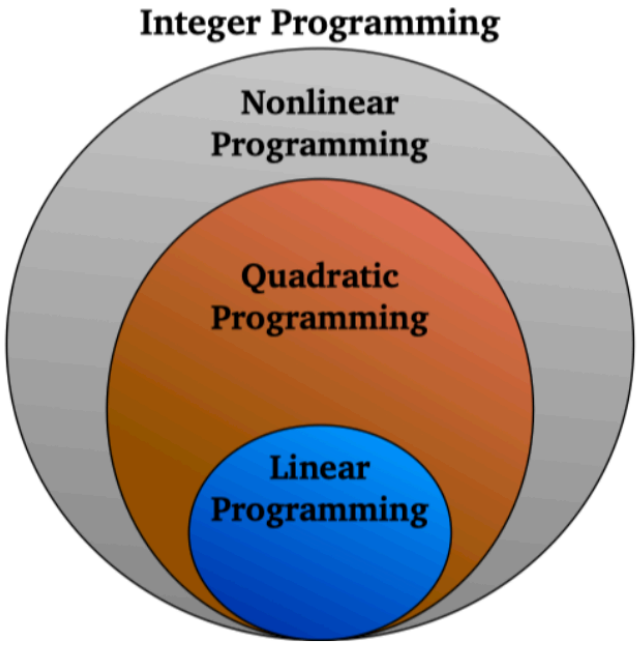
- We want to minimize a cost function
- Solving an optimization problem = finding the solution that minimizes the cost function (possibly subject to constraints)

$$f : \mathscr{D} \subseteq \mathbb{R}^n \longrightarrow \mathbb{R}$$ cost function

$$\Omega \subseteq \mathscr{D}$$ constraint set or solution space

$$\vec{x}_{opt} \in \Omega \mid f(\vec{x}_{opt}) \leq f(\vec{x}) \quad \forall \vec{x} \in \Omega$$ global minimizer

# Optimization problems




mathematical programming hierearchy
-> progressively more general

- Linear programming

Find $x$
that minimizes $c^\top x$
such that $Ax \leq b$
and $x \geq 0$

- Quadratic programming

Find $x$
that minimizes $\frac{1}{2}\, x^\top Q x + c^\top x$
such that $Ax \leq b$

# QUBO problems

$$\text{minimize} \quad f(\vec{x}) = \vec{x}^{\top} \boldsymbol{Q} \vec{x} = \sum_{i=1}^{n} \sum_{j=1}^{n} Q_{ij} x_i x_j$$

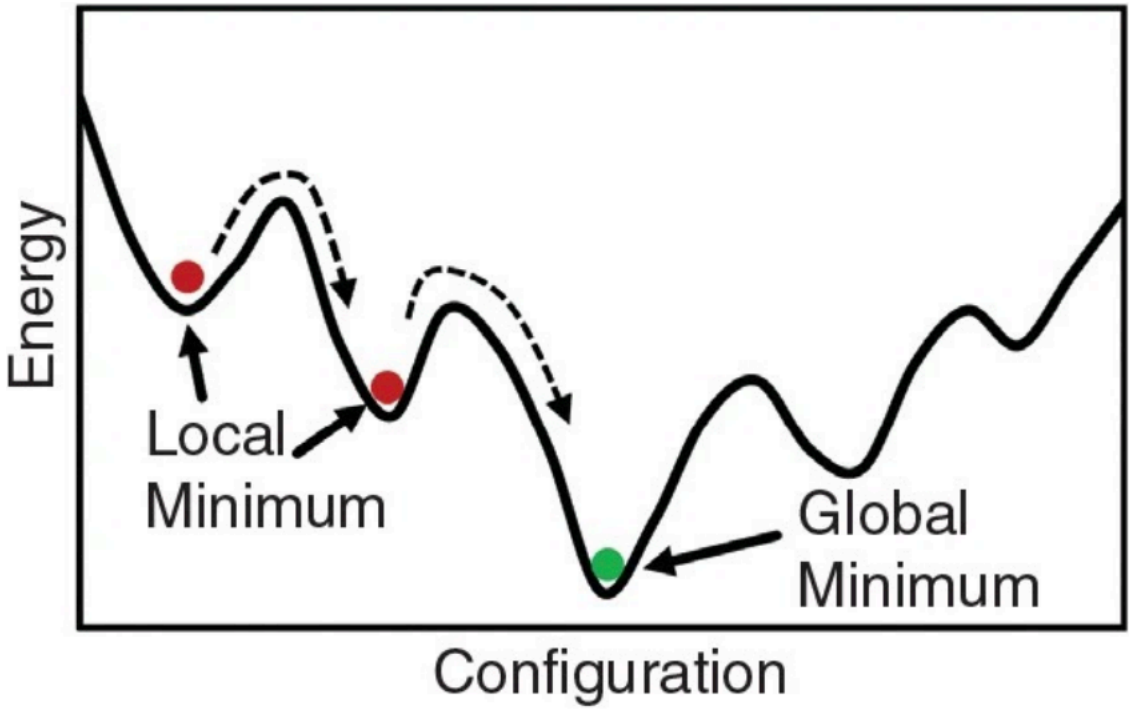
$$\boldsymbol{Q} \in \mathbb{R}^{n \times n}$$

$$\vec{x} \in \mathbb{B}^n \quad \text{where} \quad \mathbb{B} = \{0,1\}$$

- Subset of quadratic programming problems
- Solution is a bit string (discrete optimization)
- NP-hard *P. M. Pardalos et al., Operations Research Letters, 11 119-123 (1992)*
- Structure of the Q matrix determines the difficulty of the problem

# QUBO to spin glass

- Spin glass
  - Disordered spin system
  - All to all interactions
  - Frustration
  - Many local energy minima



# QUBO to spin glass

- QUBO problems can be transformed into spin glass problems
- Optimal solution ⟷ spin configuration of ground state

$$f(\vec{x}) = \vec{x}^{\top}\boldsymbol{Q}\vec{x} = \sum_{i=1}^{n}\sum_{j=1}^{n} Q_{ij}x_i x_j \longrightarrow H_{classical}(\{s_i\}) = \sum_{i<j} J_{ij}s_i s_j + \sum_i g_i s_i + const. \longrightarrow H_{quantum} = \sum_{i<j} J_{ij}\sigma_i^z\sigma_j^z + \sum_i g_i\sigma_i^z$$

$$x_i^2 = x_i$$

$$x_i = \frac{s_i + 1}{2}$$

$$s_i \rightarrow \sigma_i^z$$

| | | |
|---|---|---|
| Bit string | ⟷ | Spin configuration |
| Cost function | | System energy |

# Sherrington-Kirkpatrick (SK) model

- Spin couplings are drawn from a bimodal ($\pm 1$) distribution
- Local terms are 0
- Generate ensembles of random problems
- Ground state search via tensor networks vs. QUBO solver (Gurobi)

$$J_{ij} = \pm 1$$

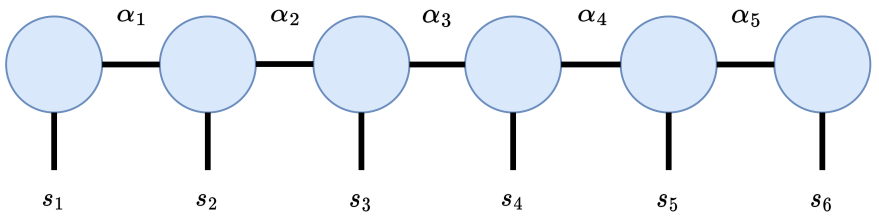
$$H_{SK} = \sum_{i<j} J_{ij}\sigma_i^z\sigma_j^z \longrightarrow f(\vec{x}) = \vec{x}^{\top}\boldsymbol{Q}\vec{x} = \sum_{i=1}^{n}\sum_{j=1}^{n} Q_{ij}x_i x_j$$

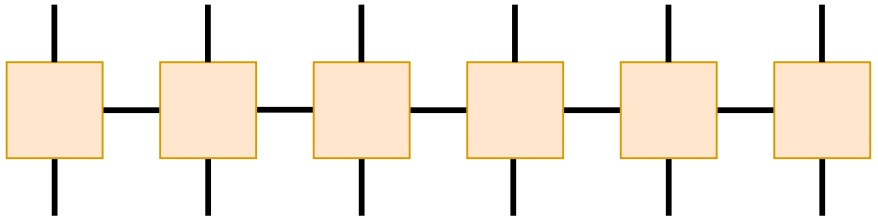
Back to QUBO for Gurobi

$$\sigma_i = 2x_i - 1$$

# Solving QUBO with tensor networks

- State as MPS
- Hamiltonian as MPO
- Exploring a superposition of solutions






DMRG → Measure spin configuration → Convert spin configuration to bit string

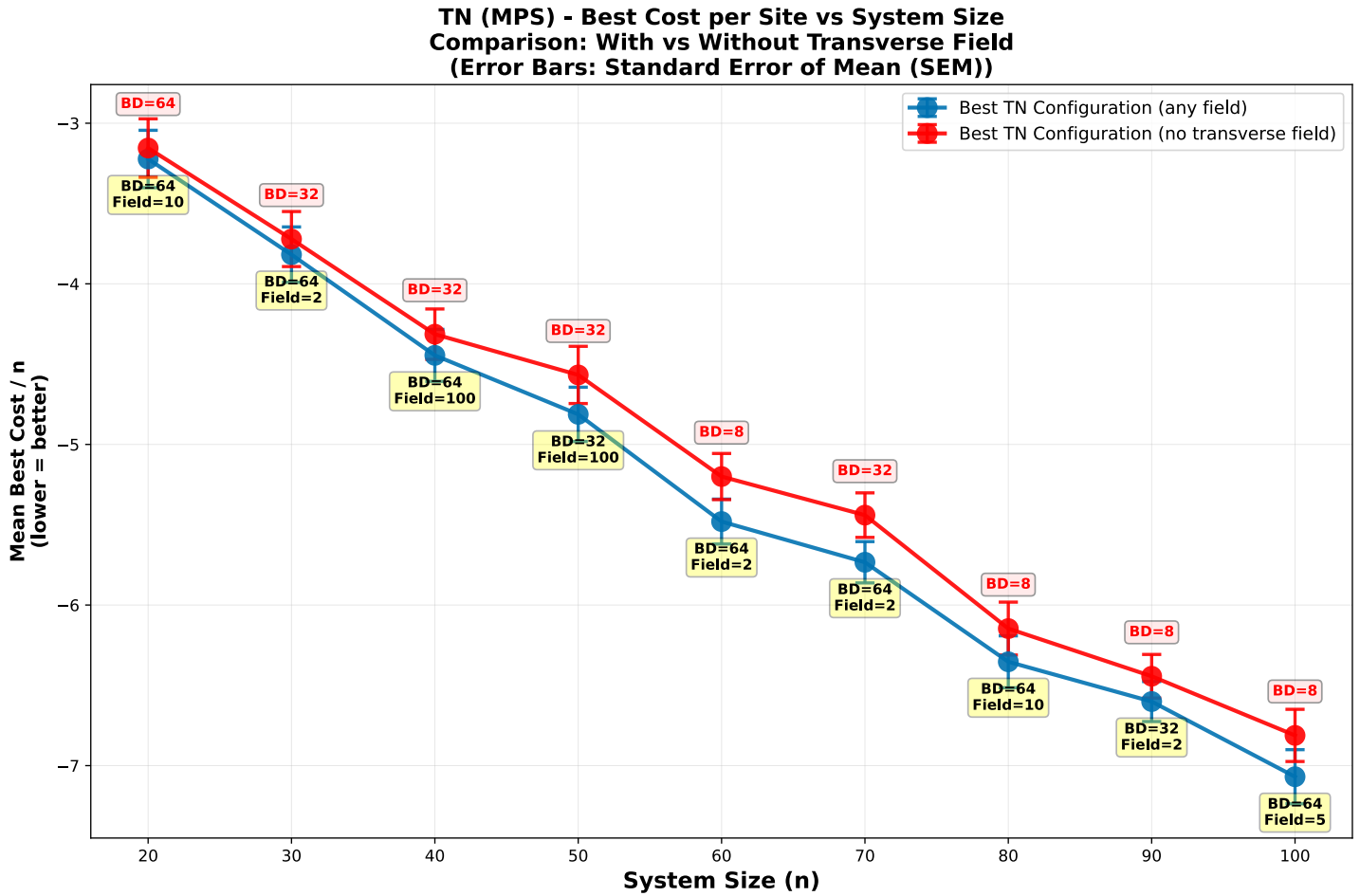
$$s_i = \langle \Psi_{GS} | \sigma_i^z | \Psi_{GS} \rangle$$

$$x_i = \frac{s_i + 1}{2}$$

# Solving QUBO with tensor networks

- Adding transverse fields to introduce randomness
  - Helps escape local minima

$$H = \sum_{i<j} J_{ij}\sigma_i^z\sigma_j^z + \sum_i g_i\sigma_i^x$$



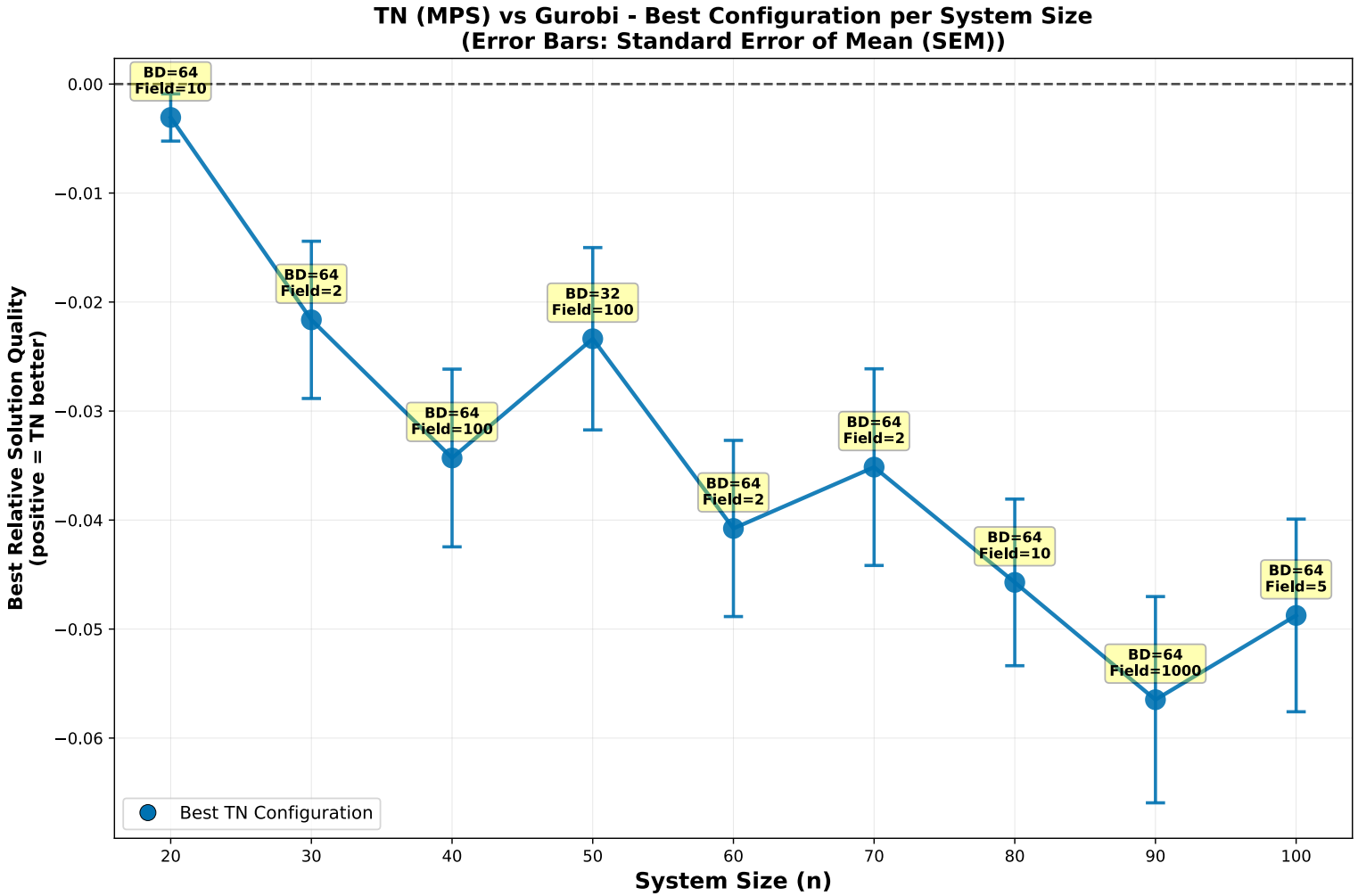

AQTIVATE

# Solving the SK model Hamiltonian with Gurobi

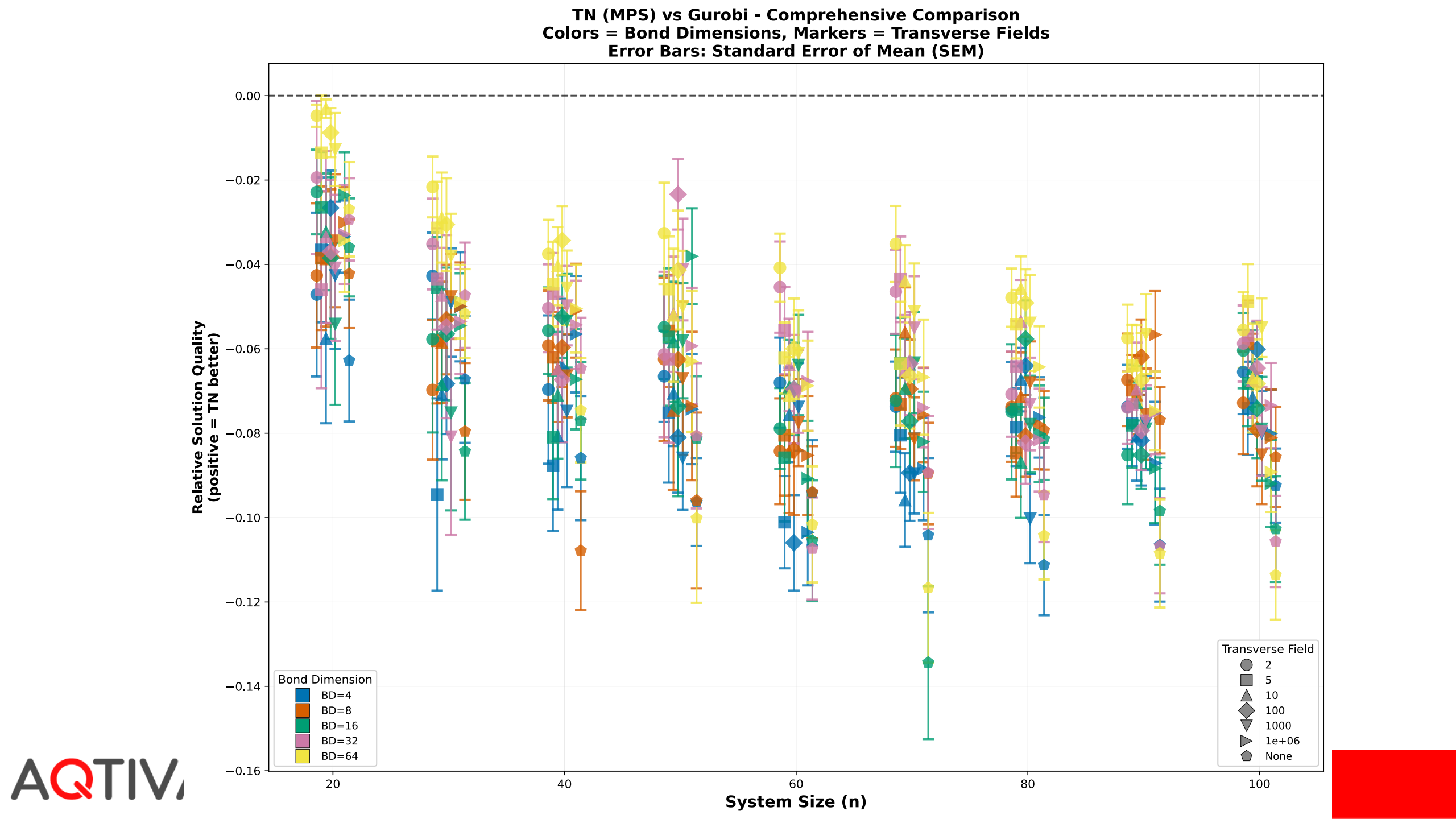
- Convert to QUBO
- Relaxation - “relaxes” the constraints by replacing the binary variables with an interval
- Branch and bound
  - Creates branches by fixing some of the variables to binary values
  - Computes the lower bound of the cost function of each branch using the relaxed problem
  - Decides which branch to continue exploring

# DMRG results

$$y = \frac{f_{\text{gurobi}} - f_{\text{tn}}}{0.5 \cdot \left| f_{\text{gurobi}} + f_{\text{tn}} \right|}$$




AQTIVATE


TN (MPS) vs Gurobi - Comprehensive Comparison
Colors = Bond Dimensions, Markers = Transverse Fields
Error Bars: Standard Error of Mean (SEM)
Relative Solution Quality
(positive = TN better)
0.00
−0.02
−0.04
−0.06
−0.08
−0.10
−0.12
−0.14
−0.16
20
40
60
80
100
System Size (n)
Bond Dimension
BD=4
BD=8
BD=16
BD=32
BD=64
Transverse Field
2
5
10
100
1000
1e+06
None

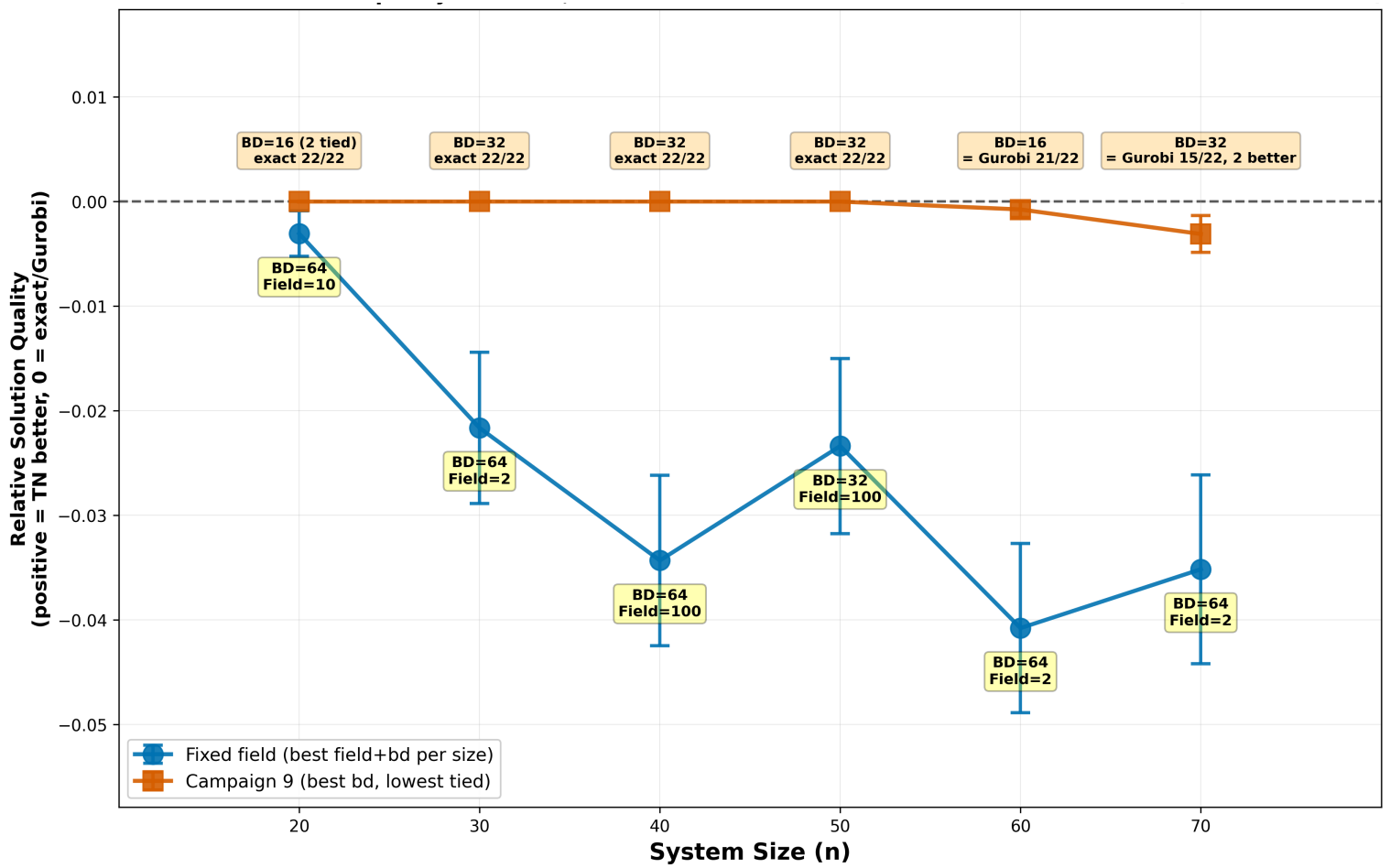
# Summary of results

- DMRG is always worse than Gurobi
- Increasing the bond-dimension doesn't help much
  - There is no consistent best transverse field ratio
  - No transverse field was never best

# Next steps

- Testing with adaptive transverse fields (converge -> change field -> converge -> change field -> …)
- Preliminary results show improvement but there is still no clear conclusion
- The adaptive algorithm highly depends on the chosen transverse fields

# Preliminary adaptive-field algorithm results




*Gurobi reference = 23 h `preq` incumbent, proven optimal on N=60: 19/22, N=70: 0/22 seeds (all seeds elsewhere). Where it is unproven, '= Gurobi' is not exactness and quality > 0 means the run beat the incumbent.*

AQTIVATE

Thank you

# Configuring Gurobi for SK

- Gurobi has configurable parameters that allow us to modify the algorithm used to search for the solution
- PreQLinearize=1 dramatically improved the memory-scaling issue with system size, allowing us to run the simulations for longer

$$\text{(QUBO)} \quad \min_x \; x^\top Q x \;=\; \sum_{i=1}^{n}\sum_{j=1}^{n} q_{ij}\, x_i x_j$$

$$x_i \in \{0,1\} \quad \forall i$$

PreQLinearize=1: $z_{ij} = x_i x_j$ for each $q_{ij} \neq 0,\ i < j$

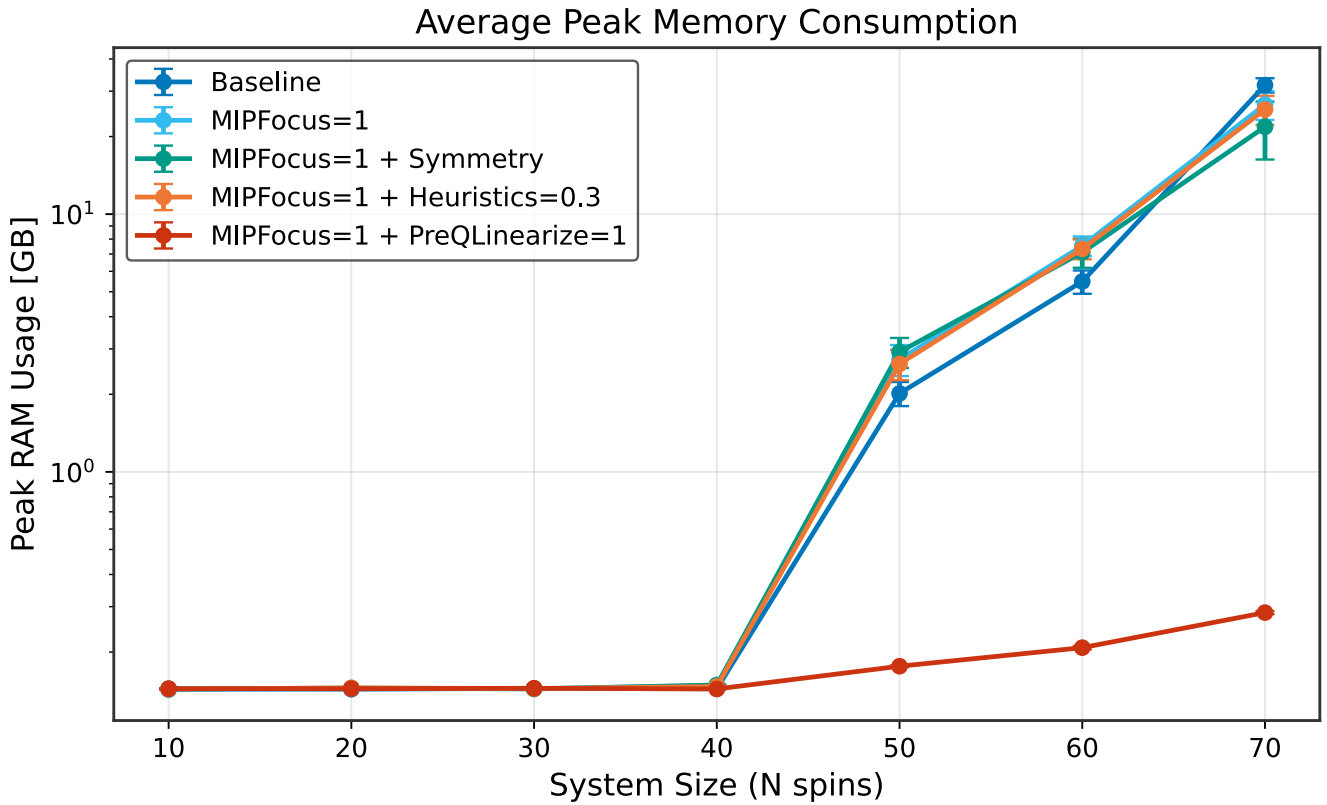
Replace $x_i x_j \to z_{ij}$ in objective

$$z_{ij} \le x_i$$
$$z_{ij} \le x_j$$
$$z_{ij} \ge x_i + x_j - 1$$
$$z_{ij} \in \{0,1\}$$

$$\min_{x,\,z} \quad \sum_{i=1}^{n} q_{ii}\, x_i \;+\; \sum_{1 \le i < j \le n} (q_{ij} + q_{ji})\, z_{ij}$$

# Configuring Gurobi for SK

- PreQLinearize=1 outperformed others in memory consumption



QUBO BRANCH-AND-BOUND

# Branch and Bound with Convex Quadratic Relaxation

minimize $f(x) = x^TAx = -x_1 - 2x_2 - x_3 + 4x_1x_2 + 4x_2x_3$ subject to $x_i \in \{0, 1\}$

z* = −2 OPTIMAL | CONVEX RELAXATION (P = A+5I) | PRUNING BY BOUND | 8 TOTAL NODES



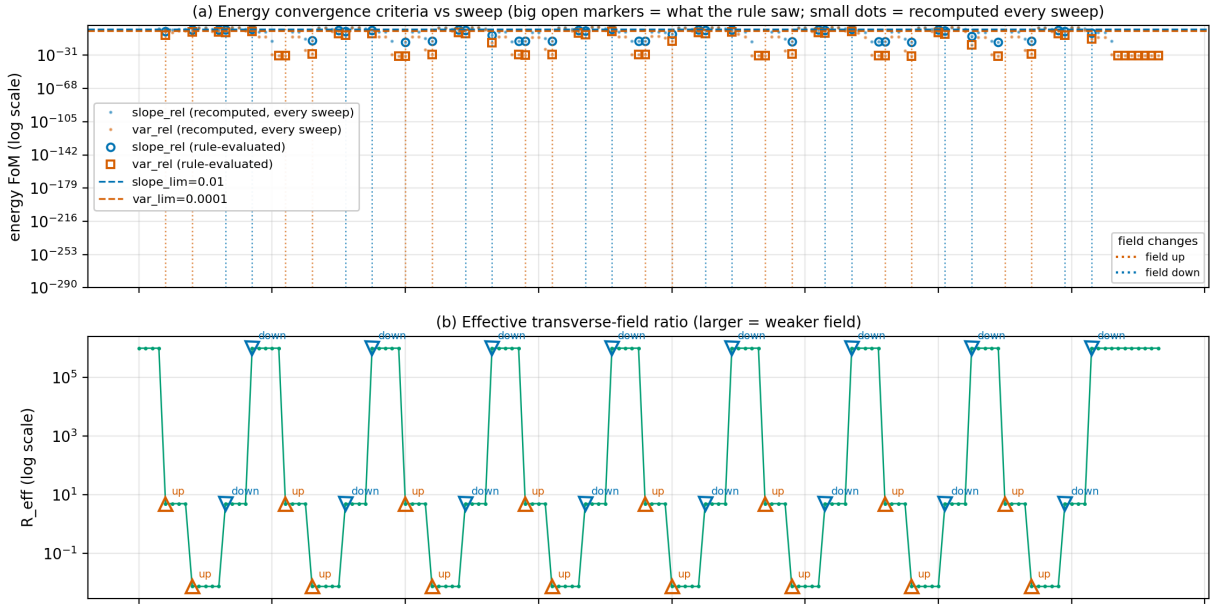

**Legend:** Branch node | Optimal integer | Integer (pruned by incumbent) | Pruned by bound (LB ≥ z*)

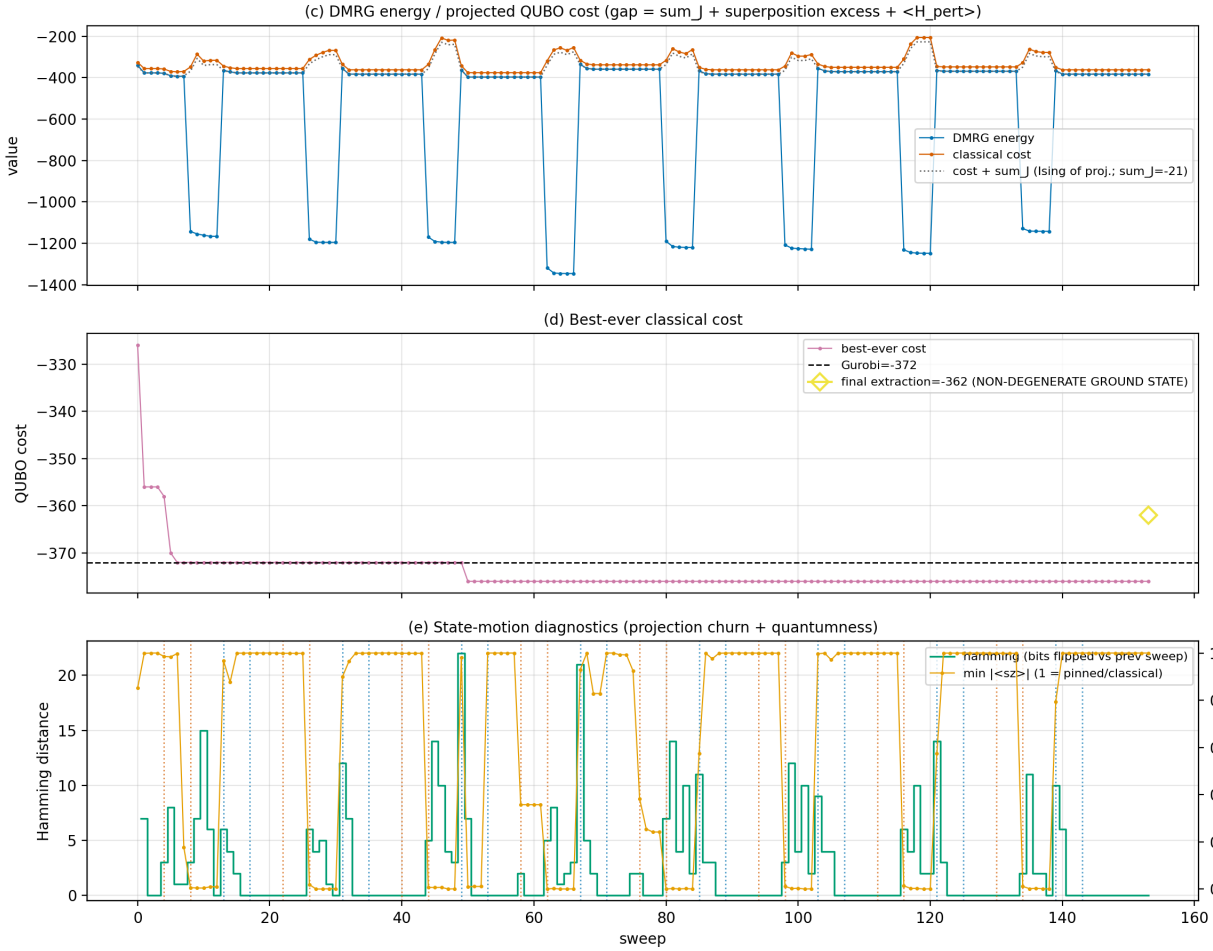
Relaxation: $g(x) = x^TPx - 5\Sigma x_i$ with $P = A + 5I \succ 0$

*For binary x: g(x) = f(x). For fractional x, g(x) gives a valid lower bound.*

8 nodes | 3 branch | 3 pruned by bound (1b, 2a, 2b) | optimal value −2 at (0,1,0) and (1,0,1)

# Preliminary results single-run evolution




AQTIVATE