

$$\mathcal{H}|\psi\rangle = E|\psi\rangle$$



Introduction to Exact Diagonalization

Andreas Läuchli

Institute for Theoretical Physics

University of Innsbruck, Austria

andreas.laeuchli@uibk.ac.at

<http://laeuchli-lab.uibk.ac.at/group-page>



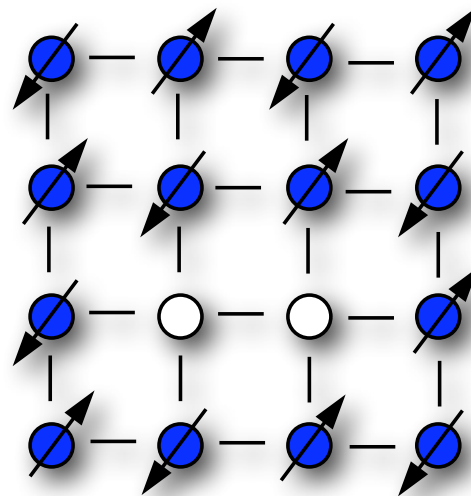
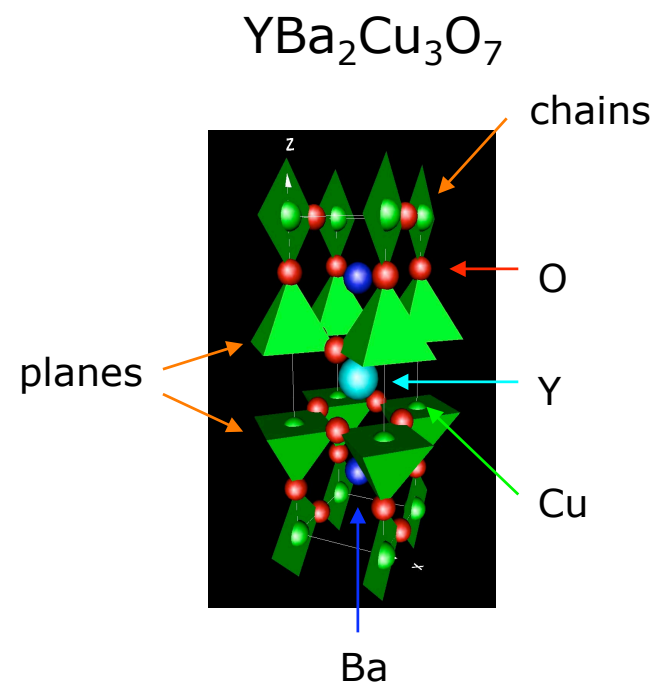
Quantum Many Body Systems

Strongly correlated electrons in solids

● High Tc superconductors & Quantum Magnets

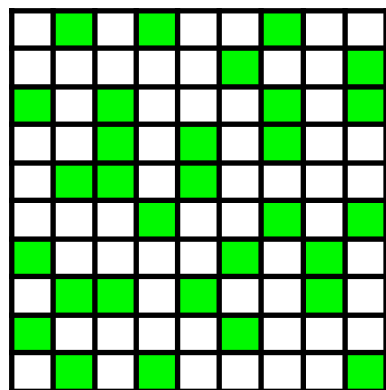


Bednorz & Müller, 1987

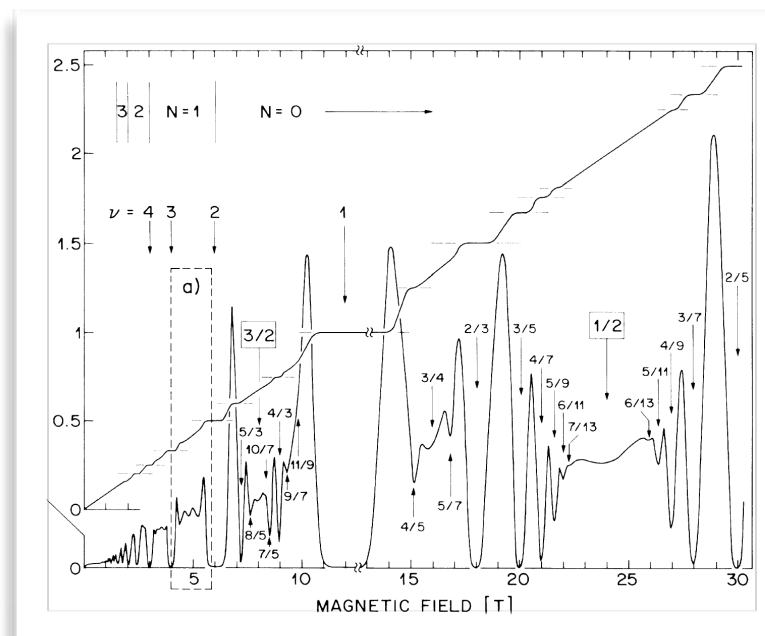


● Fractional Quantum Hall Effect

$$\nu = 1/3$$

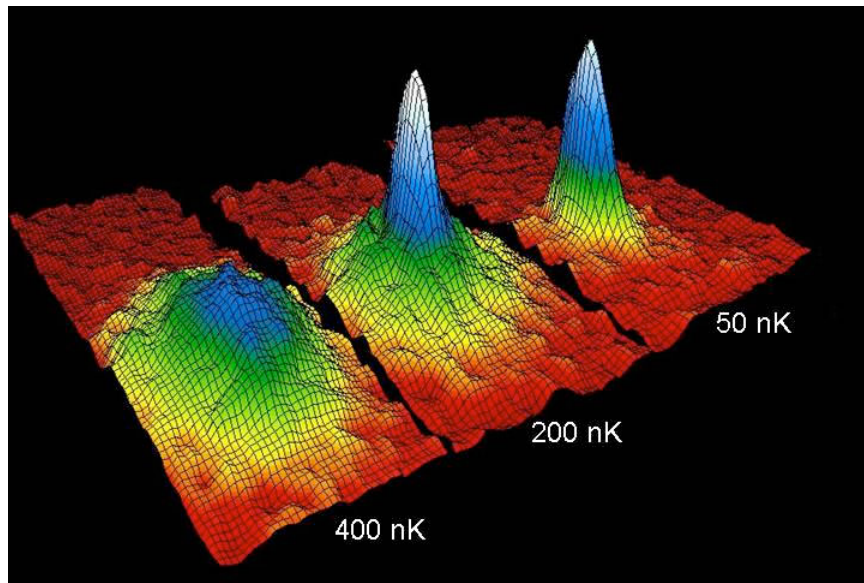


Störmer, Tsui, Laughlin, 1998



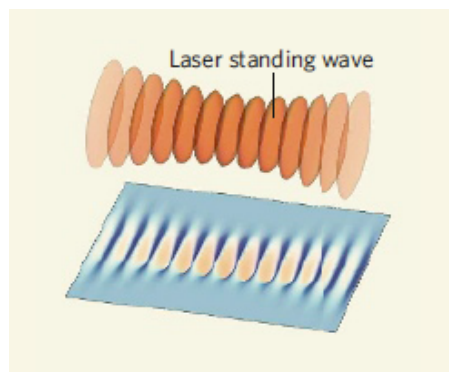
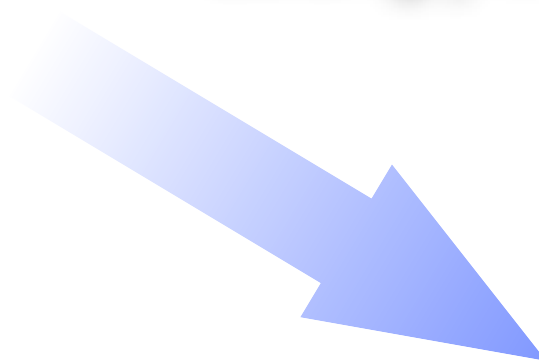
Quantum Many Body Systems

Ultracold atomic gases



Cornell, Ketterle, Wiemann, 2001

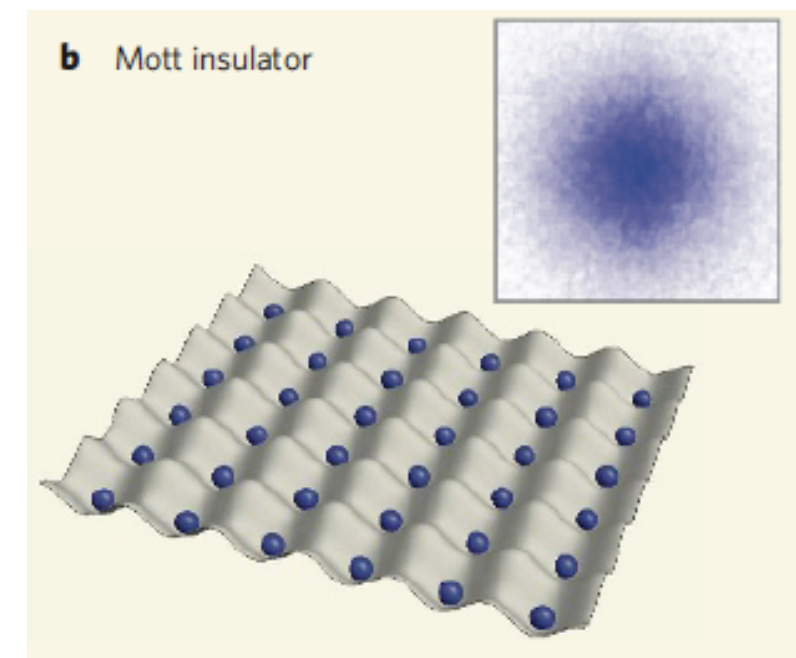
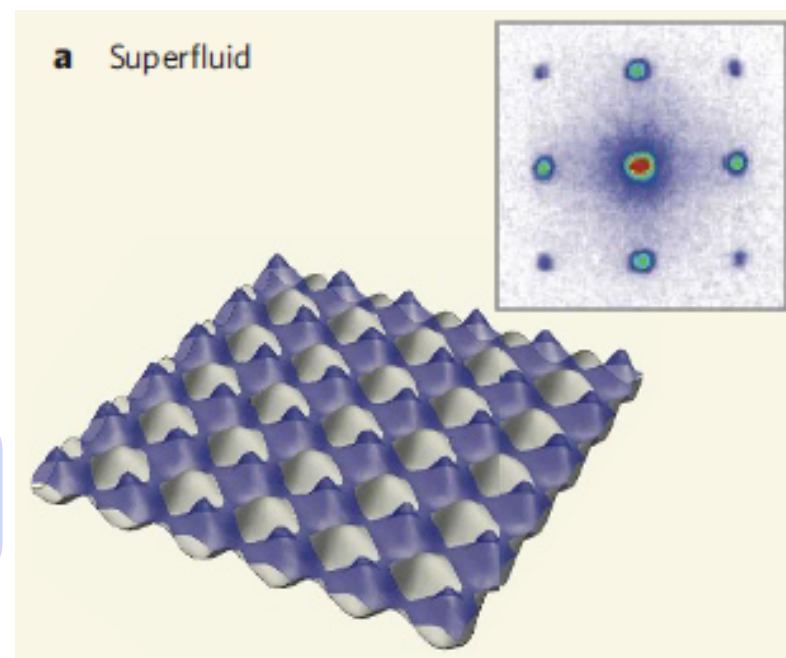
From **weakly interacting** Bose gases
to **strongly interacting** gases in optical lattices



$$H = -J \sum_{\langle i,j \rangle} (b_i^\dagger b_j + h.c.) + \frac{U}{2} \sum_i n_i (n_i - 1)$$

D. Jaksch et al., PRL (1998)

M. Greiner et al., Nature (2002)



Exact Diagonalization: Main Idea

- Solve the Schrödinger equation of a quantum many body system numerically

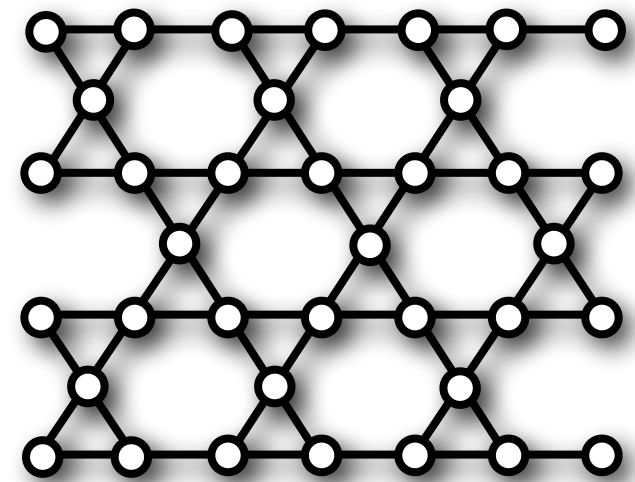
$$\mathcal{H}|\psi\rangle = E|\psi\rangle$$

- Sparse matrix, but for quantum many body systems the vector space dimension grows exponentially!
- Some people will tell you that's all there is.
- But if you want to get a maximum of physical information out of a finite system there is a lot more to do and the reward is a powerful:

Quantum Mechanics Toolbox

Hilbert space sizes

- The Hilbert space of a quantum many body system grows exponentially in general
- For N spin $1/2$ particles, the complete Hilbert space has $\text{dim}=2^N$ states
- 10 spins $\text{dim}=1'024$
- 20 spins $\text{dim}=1'048'576$
- 30 spins $\text{dim}=1'073'741'824$
- 40 spins $\text{dim}=1'099'511'627'776$
- 50 spins $\text{dim}=1'125'899'906'842'624 \dots$
- The quantum mechanical wave function is a vector in this Hilbert (vector) space and we would like to know the ground state and a few other low lying eigenstates



○ $|\uparrow\rangle$ or $|\downarrow\rangle$

Exact Diagonalization: Applications

- **Quantum Magnets**: nature of novel phases, critical points in 1D, dynamical correlation functions in 1D & 2D
- **Fermionic models (Hubbard/t-J)**: gaps, pairing properties, correlation exponents, etc
- **Fractional Quantum Hall states**: energy gaps, overlap with model states, entanglement spectra
- **Quantum dimer models or other constrained models (anyon chains, ...)**
- **Full Configuration Interaction in Quantum Chemistry, Nuclear structure**
- **Quantum Field Theory**

Exact Diagonalization: Present Day Limits

- Spin $S=1/2$ models:
 - 50 spins square lattice, 48 sites triangular, 42 sites Honeycomb lattice
 - 48 sites kagome lattice
 - 64 spins or more in elevated magnetization sectors
 - up to ~500 billion basis states (5×10^{11} states)
- Fractional quantum hall effect
 - different filling fractions ν , up to 16-20 electrons
 - up to 3.5 billion basis states
- Hubbard models (~ Full CI in Quantum Chemistry)
 - 20 sites square lattice at half filling, 21 sites triangular lattice
 - 24 sites honeycomb lattice
 - up to 160 billion basis states

low-lying eigenvalues, not full diagonalization

Structure of an Exact Diagonalization code

Ingredients

- Hilbert space
 - Basis representation, Lookup techniques
 - Symmetries
- Hamiltonian Matrix
 - Sparse Matrix representation (memory/disk)
 - Matrix recalculation on the fly (matrix-free)
- Linear Algebra : Eigensolver / Time propagation
 - LAPACK full diagonalization
 - Lanczos type diagonalization (needs only $|v\rangle = H|u\rangle$ operations)
 - More exotic eigensolver techniques, real oder imaginary-time propagation,
- Observables
 - Static quantities (multipoint correlation functions, correlation density matrices,...)
 - Dynamic observables (spectral functions, density of states,...)
 - Real-time evolution

Hilbert Space

Basis representation

- States of the Hilbert space need to be represented in the computer.
- Choose a representation which makes it simple to act with the Hamiltonian or other operators on the states, and to localize a given state in the basis
- Simple example: ensemble of $S=1/2$ sites in binary coding

$$|\uparrow \uparrow \downarrow \uparrow\rangle \rightarrow [1 \ 1 \ 0 \ 1]_2 = 13$$

detection of up or down spin can be done with bit-test.

transverse exchange $S^+ S^- + S^- S^+$ can be performed by an XOR operation:

$$\underbrace{[1 \ 1 \ 0 \ 1]_2}_{\text{initial config}} \text{ XOR } \underbrace{[0 \ 1 \ 1 \ 0]_2}_{\text{bit 1 at the two sites coupled}} = \underbrace{[1 \ 0 \ 1 \ 1]_2}_{\text{final config}}$$

- For $S=1$, one bit is obviously not sufficient. Use ternary representation or simply occupy two bits to label the 3 states.

Basis representation

- For t-J models at low doping it is useful to factorize hole positions and spin configurations on the occupied sites.
- For Hubbard models one can factorize the Hilbert space in up and down electron configurations.
- For constrained models - such as dimer models - the efficient generation of all basis states requires some thought.
- One of the key challenges for a fast ED code is to find the index of the new configuration in the list of all configurations (index f in $H_{i,f}$).
- Let us look at the example of $S=1/2$ spins at fixed S^z

Basis lookup procedures (Lin tables)

- One of the key problems for a fast ED code is to find the index of the new configuration in the list of all configurations (index f in $H_{i,f}$).

$$[1\ 0\ 1\ 1]_2 = 11_{10}$$

- But is 11 the index of this configuration in a list of all $S^z=1$ states ? **no !**
- Use Lin tables to map from binary number to index in list of allowed states:
(generalization of this idea works for arbitrary number of additive quantum numbers)
- Two tables with $2^{(N/2)}$ [=sqrt(2^N)] entries, one for MSBs and one for LSBs

[0 0]	=	X
[0 1]	=	0
[1 0]	=	1
[1 1]	=	2

MSB

[0 0]	=	X
[0 1]	=	0
[1 0]	=	1
[1 1]	=	0

LSB

$$\begin{aligned}\text{Ind}([0\ 1\ 1\ 1]) &= 0 + 0 = 0 \\ \text{Ind}([1\ 0\ 1\ 1]) &= 1 + 0 = 1 \\ \text{Ind}([1\ 1\ 0\ 1]) &= 2 + 0 = 2 \\ \text{Ind}([1\ 1\ 1\ 0]) &= 2 + 1 = 3\end{aligned}$$

Basis lookup procedures (Lin tables)

- Lookup can therefore be done with two direct memory reads. This is a time and memory efficient approach (at least in many interesting cases).
- An alternative procedure is to build a hash list [const access time] or to perform a binary search [log access time].
- This becomes somewhat more involved when using spatial symmetries...

Symmetries

- Consider a XXZ spin model on a lattice. What are the symmetries of the problem ?

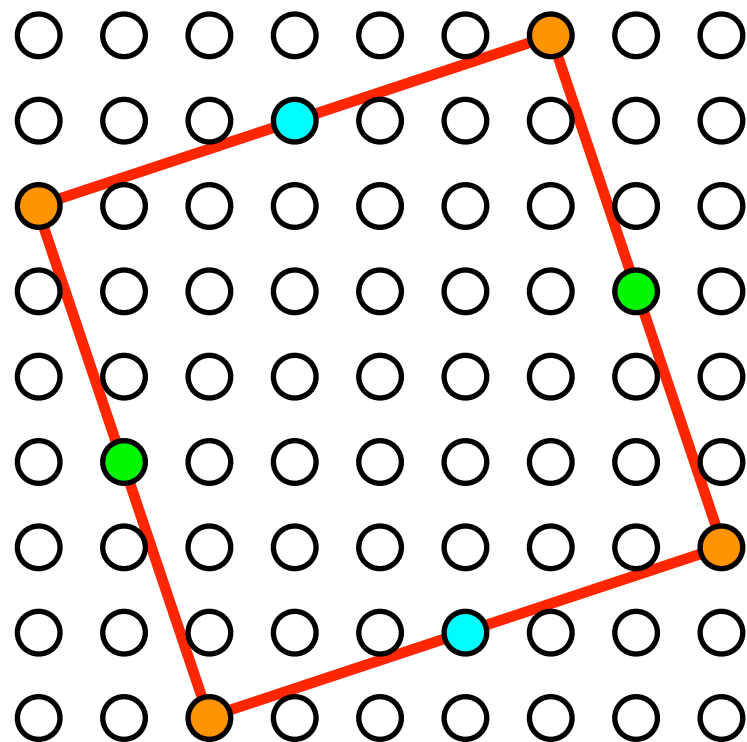
$$H = \sum_{i,j} J_{i,j}^{xy} (S_i^x S_j^x + S_i^y S_j^y) + J_{i,j}^z S_i^z S_j^z$$

- The Hamiltonian conserves total S^z , we can therefore work within a given S^z sector. This is easily implemented while constructing the basis, as we discussed before.
- The Hamiltonian is invariant under the space group, typically a few hundred elements. (in many cases = Translations x Pointgroup). Needs some technology to implement...
- At the Heisenberg point, the total spin is also conserved. It is however very difficult to combine the $SU(2)$ symmetry with the lattice symmetries in a computationally useful way (non-sparse and computationally expensive matrices).
- At $S^z=0$ one can use the spin-flip (particle-hole) symmetry which distinguishes even and odd spin sectors at the Heisenberg point. Simple to implement.

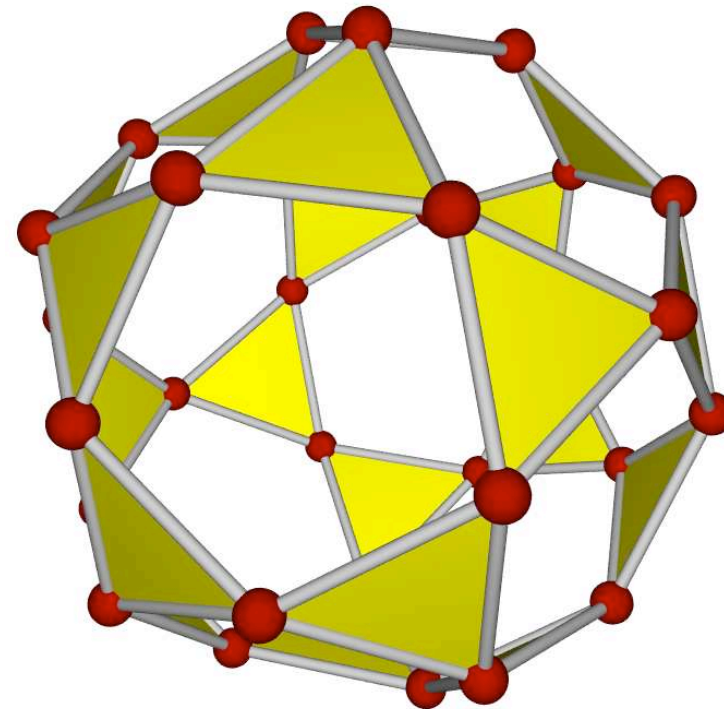
Spatial Symmetries

- Spatial symmetries are important for reduction of Hilbert space
- Symmetry resolved eigenstates teach us a lot about the physics at work, dispersion of excitations, symmetry breaking tendencies, topological degeneracy, ... \Rightarrow more about this in the second lecture

40 sites square lattice
 $T \otimes PG = 40 \times 4$ elements



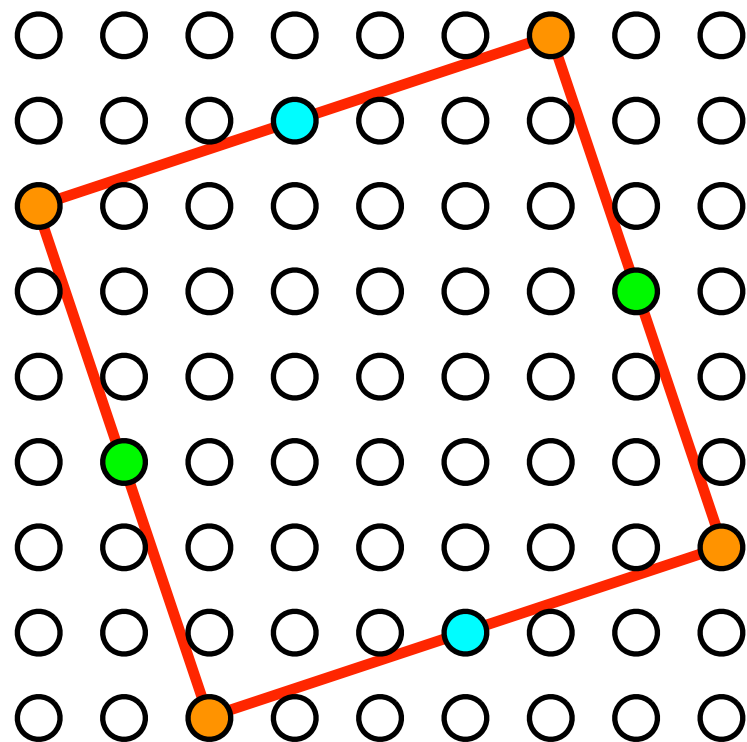
Icosidodecahedron (30 vertices)
 I_h : 120 elements



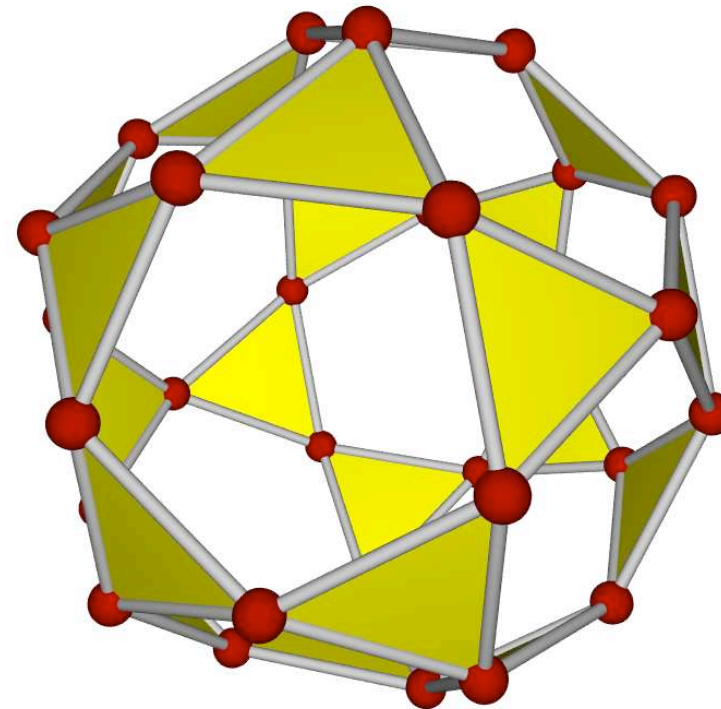
Spatial Symmetries

- Symmetries are sometimes not easily visible, use graph theoretical tools to determine symmetry group [nauty, grape].
- In an ED code a spatial symmetry operation is a site permutation operation.
(could become more complicated with spin-orbit interactions and multiorbital sites)

40 sites square lattice
 $T \otimes PG = 40 \times 4$ elements



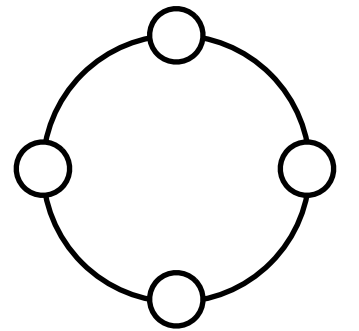
Icosidodecahedron (30 vertices)
 I_h : 120 elements



Spatial Symmetries: Building the basis

- Build a list of all allowed states satisfying the “diagonal” constraints, like particle number, total S^z , ...
- for each state we apply all symmetry operations and keep the state as a **representative** if it has the smallest integer representation among all generated states in the orbit.

Example: 4 site ring with cyclic translation T , $S^z=1$ sector



$$T^0([0 \ 1 \ 1 \ 1]) \rightarrow [0 \ 1 \ 1 \ 1]$$

$$T^1([0 \ 1 \ 1 \ 1]) \rightarrow [1 \ 0 \ 1 \ 1]$$

$$T^2([0 \ 1 \ 1 \ 1]) \rightarrow [1 \ 1 \ 0 \ 1]$$

$$T^3([0 \ 1 \ 1 \ 1]) \rightarrow [1 \ 1 \ 1 \ 0]$$

keep state

$$T^0([1 \ 0 \ 1 \ 1]) \rightarrow [1 \ 0 \ 1 \ 1]$$

$$T^1([1 \ 0 \ 1 \ 1]) \rightarrow [1 \ 1 \ 0 \ 1]$$

$$T^2([1 \ 0 \ 1 \ 1]) \rightarrow [1 \ 1 \ 1 \ 0]$$

$$T^3([1 \ 0 \ 1 \ 1]) \rightarrow [0 \ 1 \ 1 \ 1]$$

discard state

...

Spatial Symmetries: Building the basis

- For one-dimensional representations χ of the spatial symmetry group:

- “Bloch” state
$$|\tilde{r}\rangle = \frac{1}{\mathcal{N}\sqrt{|G|}} \sum_{g \in G} \chi(g) |g(r)\rangle$$

- Norm of the state is given as:
$$\mathcal{N} = \sqrt{\sum_{g \in G, g(r)=r} \chi(g)}$$

- The norm (and therefore the state itself) can vanish if it has a nontrivial stabilizer combined with a nontrivial representation χ .

- Example: 4 site $S=1/2$ ring with cyclic translations:

	$K = 0$	$K = \pm\pi/2$	$K = \pi$	$2^4=16$
$S^z=2$	$ 1\ 1\ 1\ 1\rangle, \mathcal{N} = 2$			1+1
$S^z=1$	$ 0\ 1\ 1\ 1\rangle, \mathcal{N} = 1$	$ 0\ 1\ 1\ 1\rangle, \mathcal{N} = 1$	$ 0\ 1\ 1\ 1\rangle, \mathcal{N} = 1$	4+4
$S^z=0$	$ 0\ 1\ 0\ 1\rangle, \mathcal{N} = \sqrt{2}$		$ 0\ 1\ 0\ 1\rangle, \mathcal{N} = \sqrt{2}$	2
	$ 0\ 0\ 1\ 1\rangle, \mathcal{N} = 1$	$ 0\ 0\ 1\ 1\rangle, \mathcal{N} = 1$	$ 0\ 0\ 1\ 1\rangle, \mathcal{N} = 1$	4

The Hamiltonian Matrix

The Hamiltonian Matrix

- Now that we have a list of representatives and their norms, can we calculate the matrix elements of the Hamiltonian ? $\langle \tilde{s} | H | \tilde{r} \rangle = ?$
- Let us look at an elementary, non-branching term in the Hamiltonian:

$$h^\alpha |r\rangle = h^\alpha(r) |s\rangle$$

- We can now calculate the matrix element $\langle \tilde{s} | h^\alpha | \tilde{r} \rangle$ without double expanding the Bloch states:

$$\langle \tilde{s} | h^\alpha | \tilde{r} \rangle = \frac{\mathcal{N}_s}{\mathcal{N}_r} \chi(g^*) h^\alpha(r)$$

- key algorithmic problem: given a possibly non-representative $|s\rangle$, how do we find the associated representative $|\tilde{s}\rangle$, as well as a symmetry element g^* relating $|s\rangle$ to $|\tilde{s}\rangle$?

The Hamiltonian Matrix

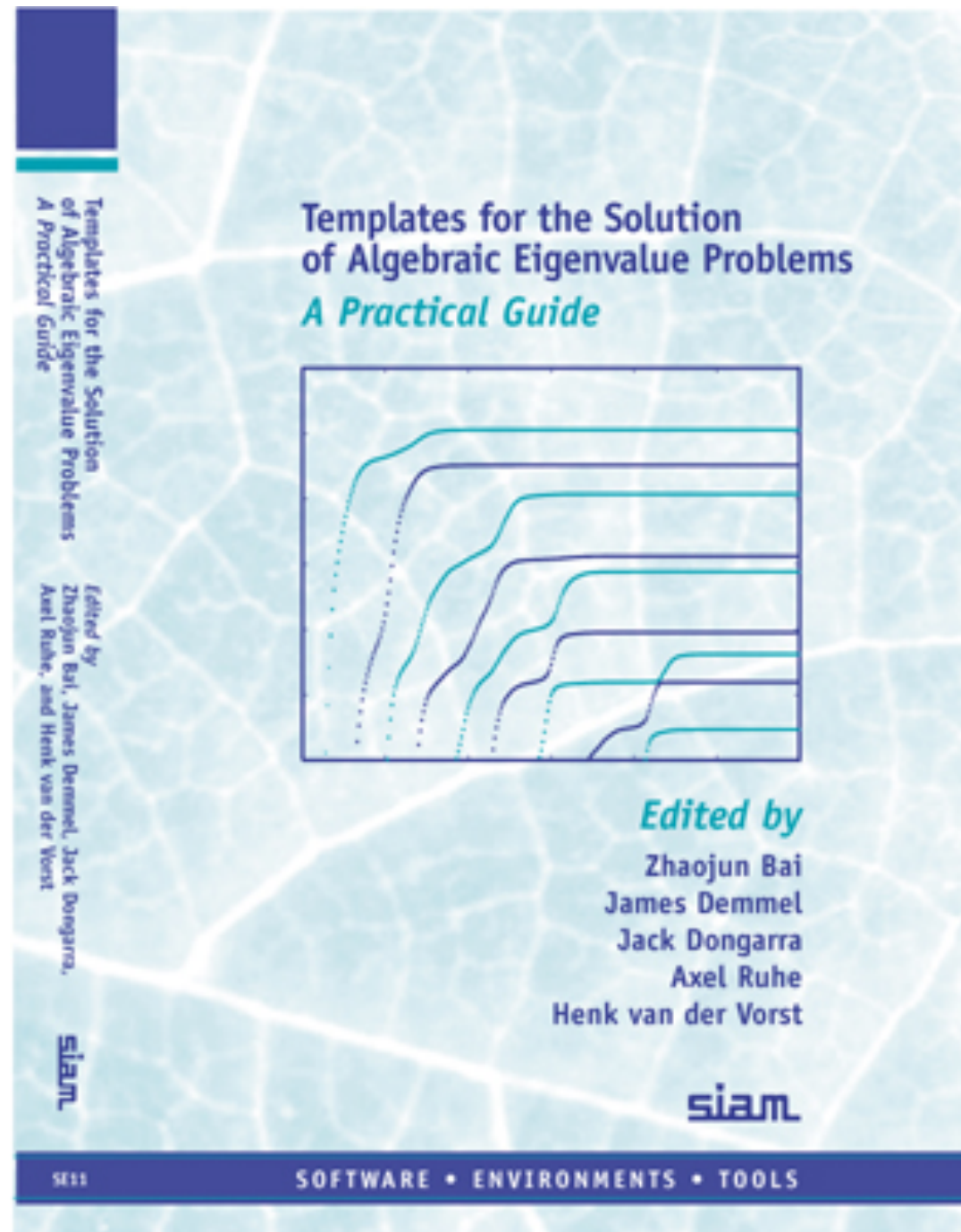
- key algorithmic problem: given a possibly non-representative $|s\rangle$, how do we find the associated representative $|\tilde{s}\rangle$, as well as a symmetry element g^* relating $|s\rangle$ to $|\tilde{s}\rangle$?
 - Brute force: loop over all symmetry operations applied on $|s\rangle$ and retain $|\tilde{s}\rangle$ and g^* . This is however often not efficient (many hundred symmetries).
 - Prepare a lookup list, relating each allowed configuration with the index of its representative, and also the associated group element linking the two. Gives fast implementation, but needs a list of the size of the non spatially-symmetrized Hilbert space.
 - For specific lattices and models (Hubbard models) clever tricks exist which factorize the symmetry group into a sublattice conserving subgroup times a sublattice exchange. They give $|\tilde{s}\rangle$ fast, then a hash or binary search is needed to locate $|\tilde{s}\rangle$ in the list of representatives in order to get its index.

Hamiltonian Matrix Storage

- Different possibilities exist:
 - Store hamiltonian matrix elements in memory in a sparse matrix format
Fast matrix vector multiplies, but obviously limited by available memory.
 - Store hamiltonian matrix elements on disk in a sparse matrix format.
In principle possible due to the vast disk space available, but I/O speed is much slower than main memory access times. Difficult to parallelize.
 - Recalculate the hamiltonian matrix elements in each iterations “on the fly”.
Needed for the cutting edge simulations, where the whole memory is used by the Lanczos vectors. Can be parallelized on most architectures.

The Linear Algebra Backend

The Reference:



● Online book at: <http://www.cs.utk.edu/~dongarra/etemplates/index.html>

Linear Algebra:

The most popular: Lanczos Algorithm

● Lanczos Algorithm (C. Lanczos, 1950)

Three vector recursion

$$|\phi'\rangle = H|\phi_n\rangle - \beta_n|\phi_{n-1}\rangle,$$

$$\alpha_n = \langle\phi_n|\phi'\rangle,$$

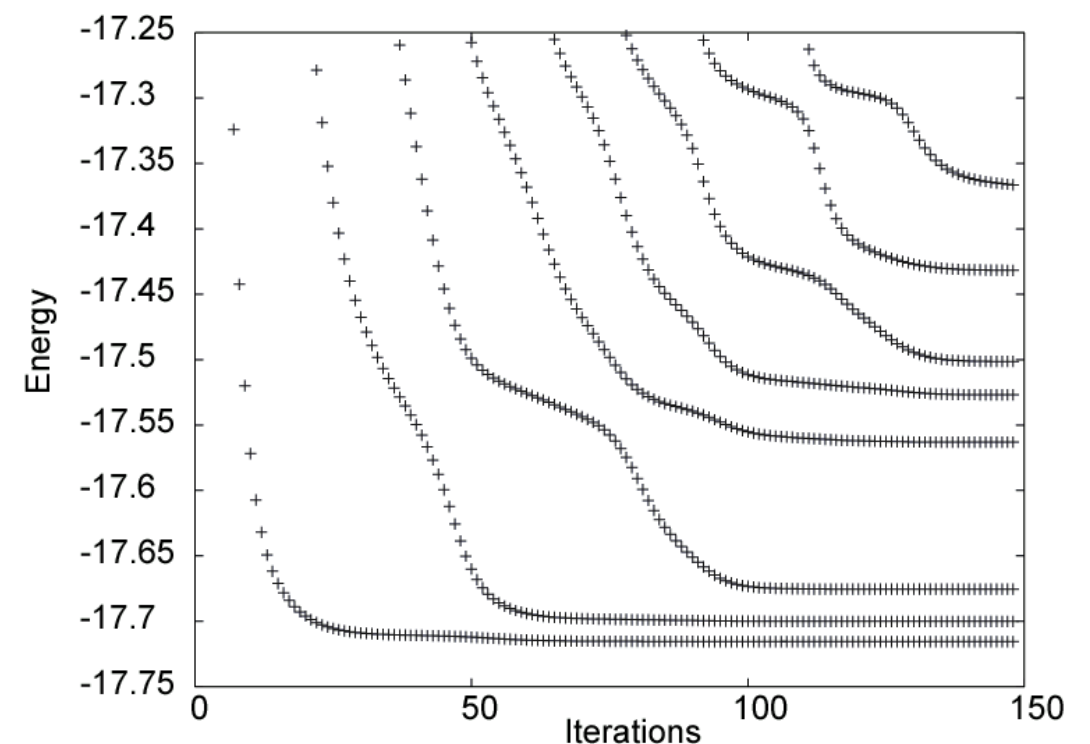
$$|\phi''\rangle = |\phi'\rangle - \alpha_n|\phi_n\rangle,$$

$$\beta_{n+1} = ||\phi''|| = \sqrt{\langle\phi''|\phi''\rangle},$$

$$|\phi_{n+1}\rangle = |\phi''\rangle/\beta_{n+1},$$

$$\tilde{H}_N = \begin{bmatrix} \alpha_0 & \beta_1 & 0 & \dots\dots\dots & 0 \\ \beta_1 & \alpha_1 & \beta_2 & 0 & \dots\dots & 0 \\ 0 & \beta_2 & \alpha_2 & \beta_3 & 0 & 0 \\ & & \ddots & \ddots & \ddots & \\ 0 & \dots & 0 & \beta_{N-2} & \alpha_{N-2} & \beta_{N-1} \\ 0 & \dots\dots\dots & 0 & \beta_{N-1} & \alpha_{N-1} \end{bmatrix}$$

- Eigenvalues of H_N converge rapidly towards eigenvalues of H .
- Once desired eigenvalue is converged, restart recursion and assemble the eigenvector.



very quick convergence for extremal eigenvalues !

Linear Algebra: Lanczos Algorithm

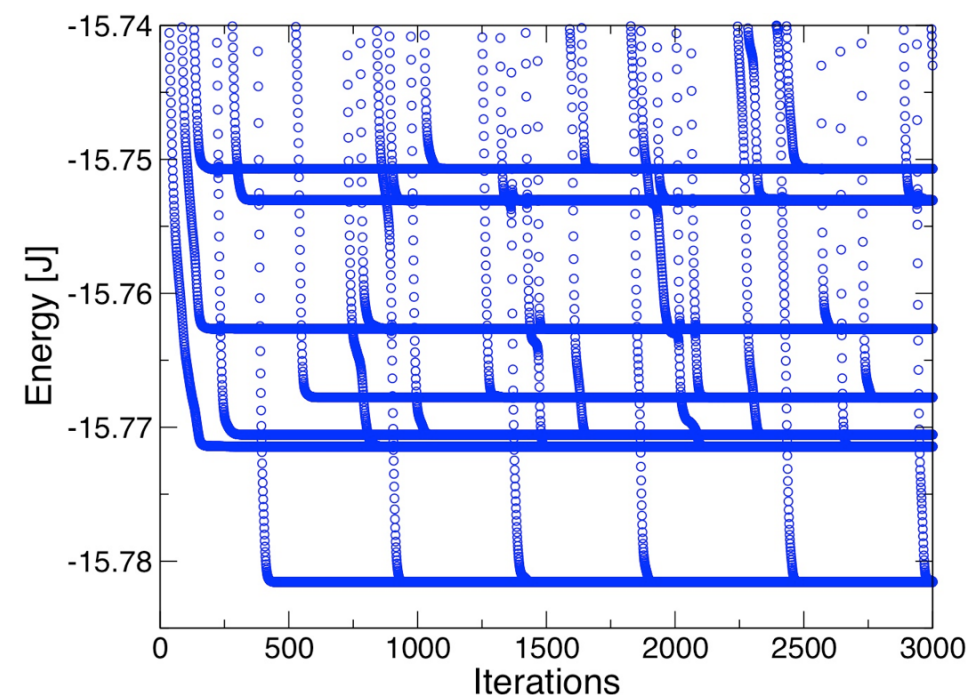
- The same algorithm according to the book:

ALGORITHM 4.6: Lanczos Method for HEP

- (1) *start with $r = v$, starting vector*
- (2) $\beta_0 = \|r\|_2$
- (3) **for** $j = 1, 2, \dots$, **until** convergence,
- (4) $v_j = r / \beta_{j-1}$
- (5) *operate $r = Av_j$*
- (6) $r = r - v_{j-1} \beta_{j-1}$
- (7) $\alpha_j = v_j^* r$
- (8) $r = r - v_j \alpha_j$
- (9) *reorthogonalize if necessary*
- (10) $\beta_j = \|r\|_2$
- (11) *compute approximate eigenvalues $T_j = S \Theta^{(j)} S^*$*
- (12) *test bounds for convergence*
- (13) **end for**
- (14) *compute approximate eigenvectors $X = V_j S$*

Linear Algebra: Lanczos Algorithm

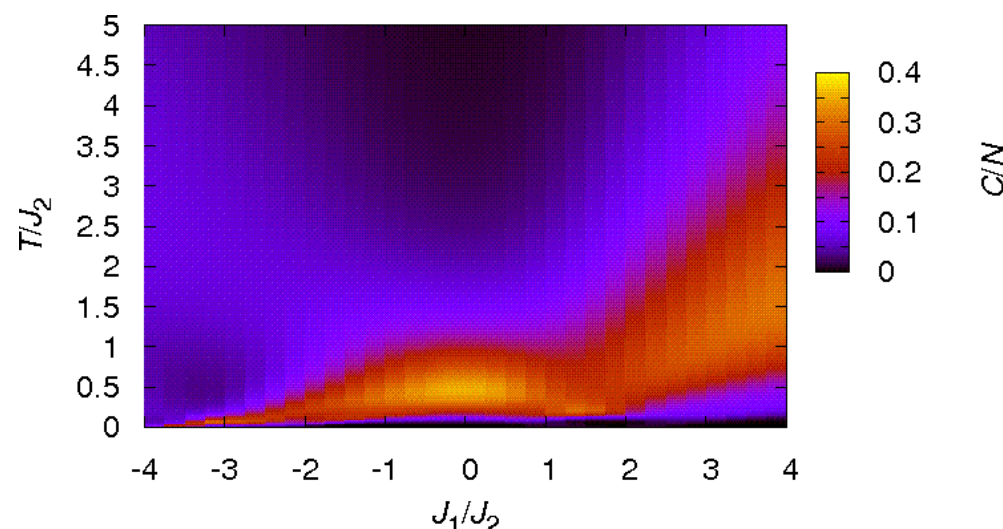
- Once the ground state has converged, the vectors in the recursion tend to lose their orthogonality. As a consequence fake new eigenvalues show up in the approximate spectrum. These can be removed by heuristic techniques



- Degeneracies of eigenvalues can **not** be resolved by construction. For this task one would need a band lanczos or the (Jacobi-)Davidson technique. However multiply degenerate eigenvalues are converged.
- Checkpointing is useful when performing large-scale simulations.

Full Diagonalization: Thermodynamics

- Lapack / Householder complete diagonalization of the spectrum.
- Calculate partition function and all the thermodynamic quantities you want, often the only pedestrian method available for frustrated systems.
- Symmetries are also very important, because the computational requirements scale as $O(D^3)$, where D is the dimension of the block Hilbert space. Typical D 's for a workstation are a few 1'000, up to a few 100'000 on supercomputers.



F. Heidrich-Meisner, A. Honecker, T. Vekua,
Phys. Rev. B 74, 020403(R) (2006).

- 'Finite-T Lanczos' and 'Quantum Typicality' methods are interesting!

$$\mathcal{H}|\psi\rangle = E|\psi\rangle$$



Introduction to Exact Diagonalization II

Andreas Läuchli

Institute for Theoretical Physics

University of Innsbruck, Austria

andreas.laeuchli@uibk.ac.at

<http://laeuchli-lab.uibk.ac.at/group-page>



Observables

Observables

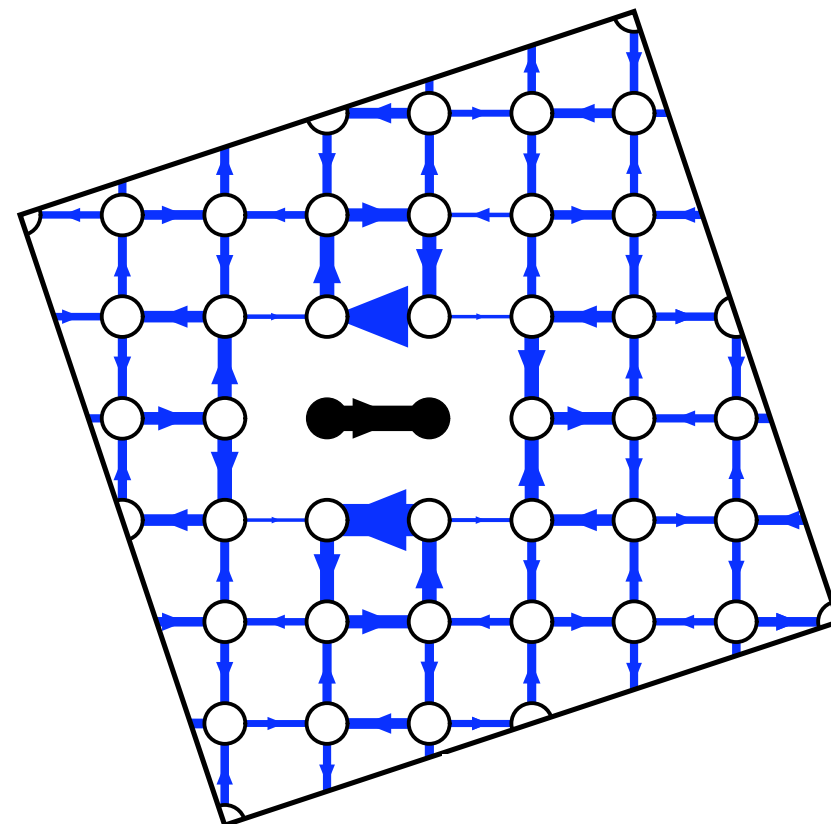
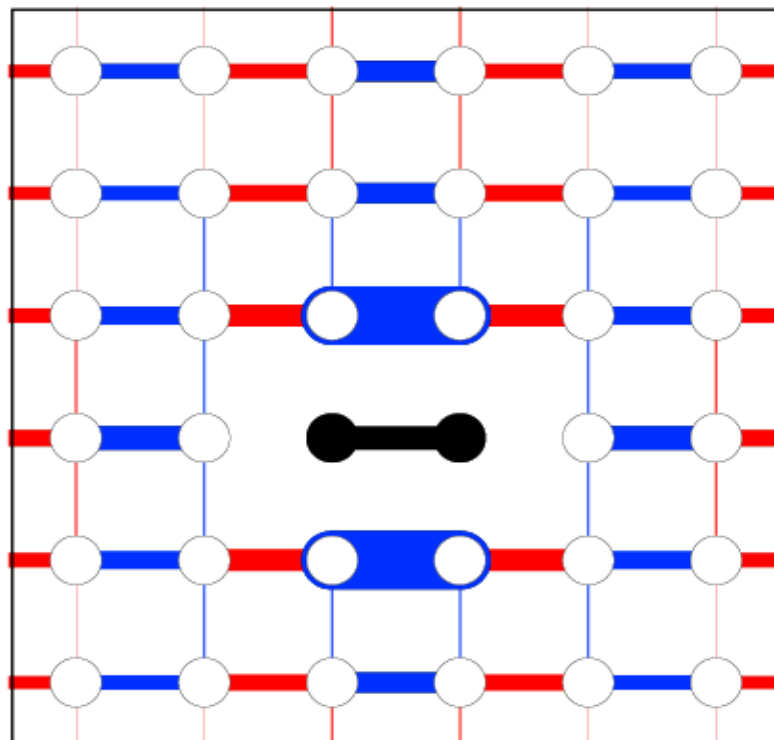
- In principle one can calculate any correlation function, since one has access to the full many body wave functions. When using spatial symmetries, the correlation functions need to be properly symmetrized too.

- Complicated correlation functions occur in frustrated systems:

$$\langle (\mathbf{S}_i \cdot \mathbf{S}_j)(\mathbf{S}_k \cdot \mathbf{S}_l) \rangle - \langle \mathbf{S}_i \cdot \mathbf{S}_j \rangle \langle \mathbf{S}_k \cdot \mathbf{S}_l \rangle \quad \langle (\mathbf{S}_i \wedge \mathbf{S}_j)^z (\mathbf{S}_k \wedge \mathbf{S}_l)^z \rangle - \langle (\mathbf{S}_i \wedge \mathbf{S}_j)^z \rangle \langle (\mathbf{S}_k \wedge \mathbf{S}_l)^z \rangle$$

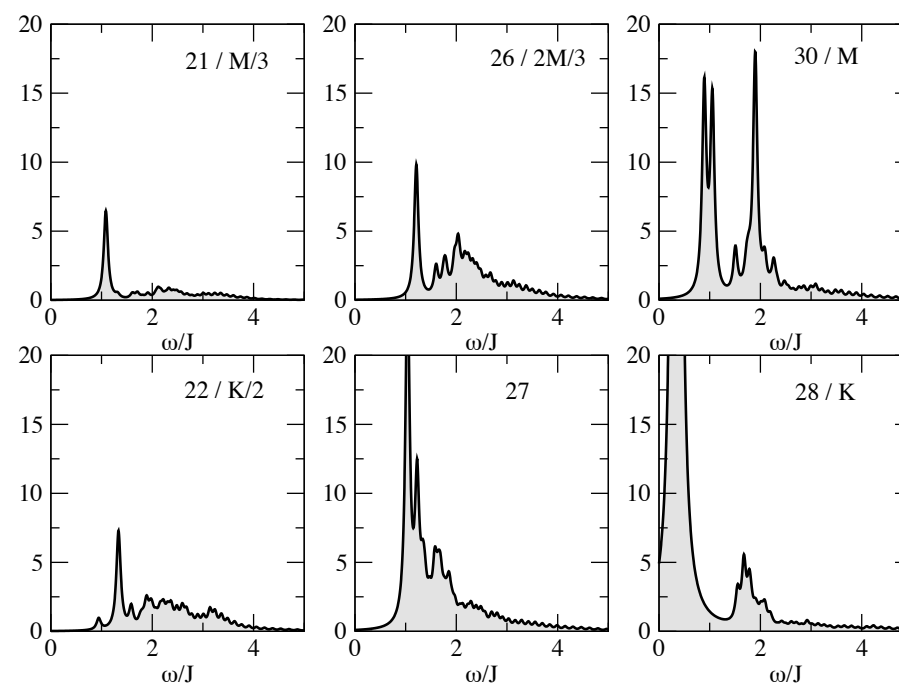
Dimer-dimer correlations

Spin current correlations



Frequency Dynamics

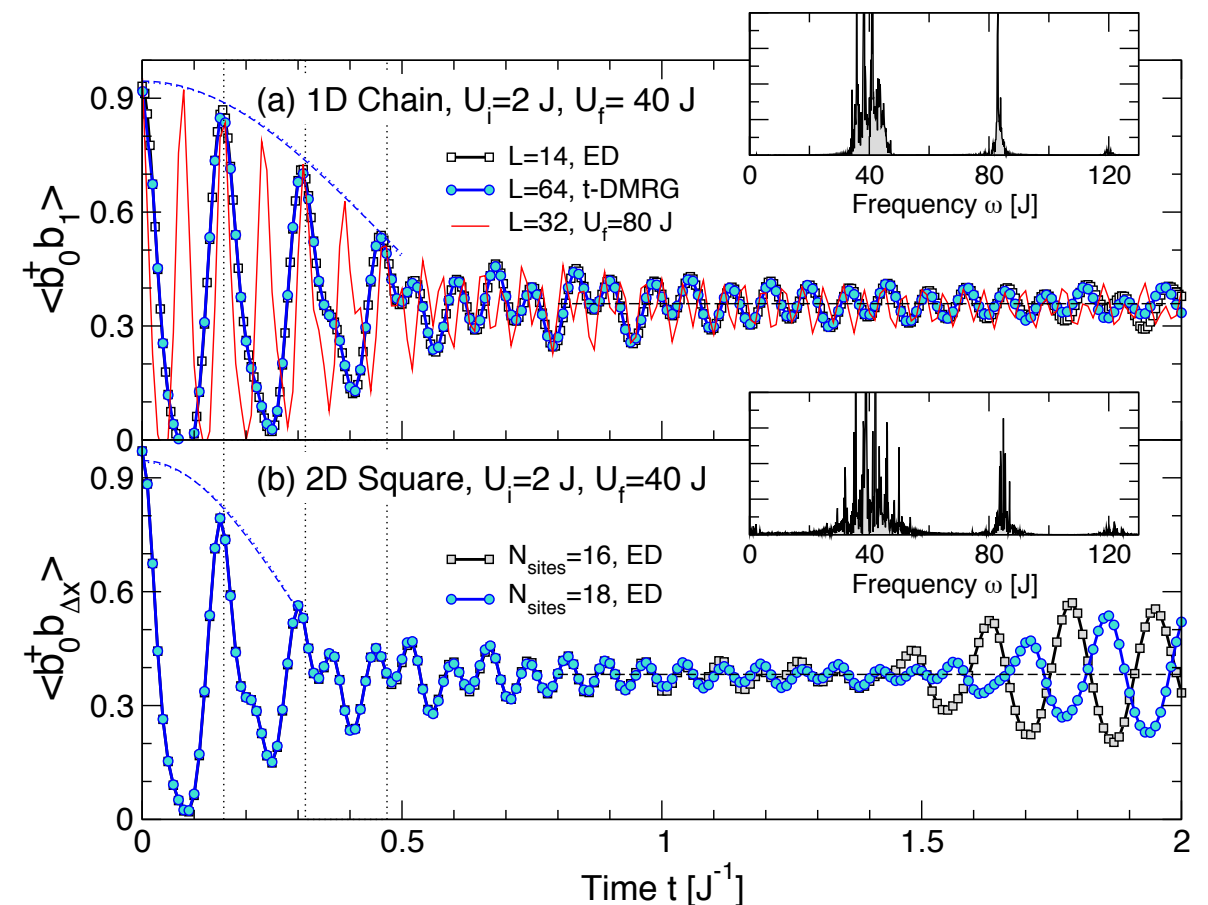
- $G_A(\omega + i\eta) = \langle \psi | A^\dagger \frac{1}{E_0 + \omega + i\eta - H} A | \psi \rangle \quad A = S^\alpha(\mathbf{q}), c_{\mathbf{k}}, \dots$
- Generate Krylov space of $A|\psi\rangle$
Use continued fraction to invert $(E_0 + \omega + i\eta - H)$
- Triangular Lattice Spin Dynamics in zero field



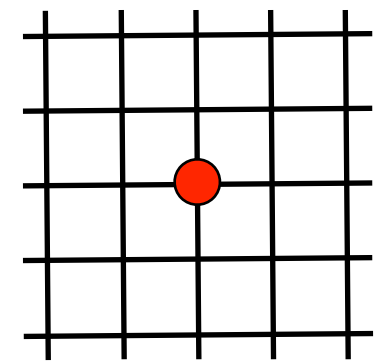
AML unpublished

Exact Diagonalization Real-Time Dynamics

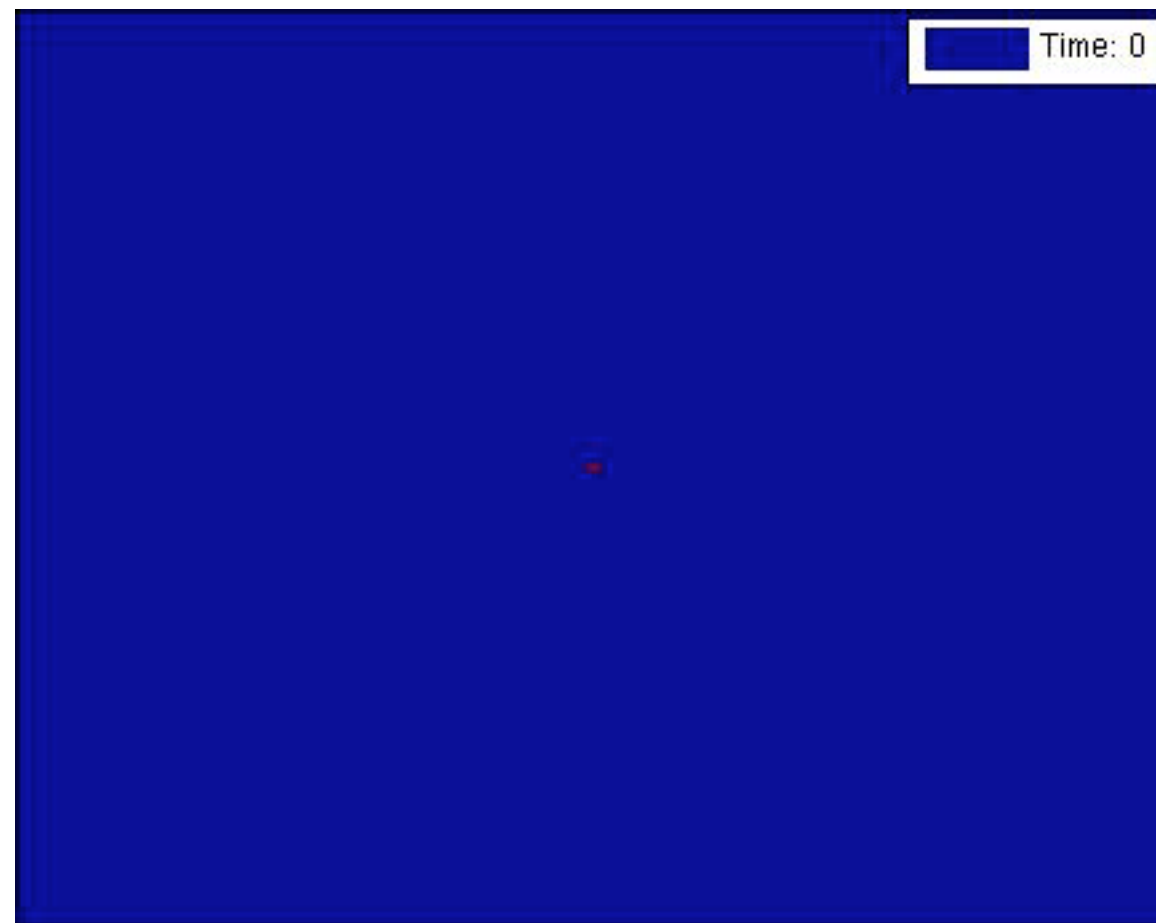
- It is expensive to obtain the full propagator $\exp[-itH]$
- Krylov methods exist to approximate the propagator for a given state $|\psi(0)\rangle$
One can get the time propagated state $|\psi(t)\rangle$ with only $|v\rangle = H|u\rangle$ operations.
- Example: time evolution of a strongly correlated quantum systems after an abrupt change in the parameters in the Hamiltonian. Revivals and Relaxation.



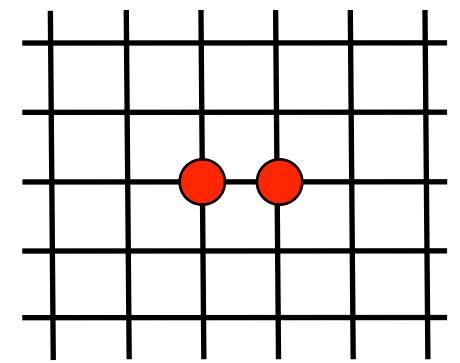
Two-dimensional uniform square lattice



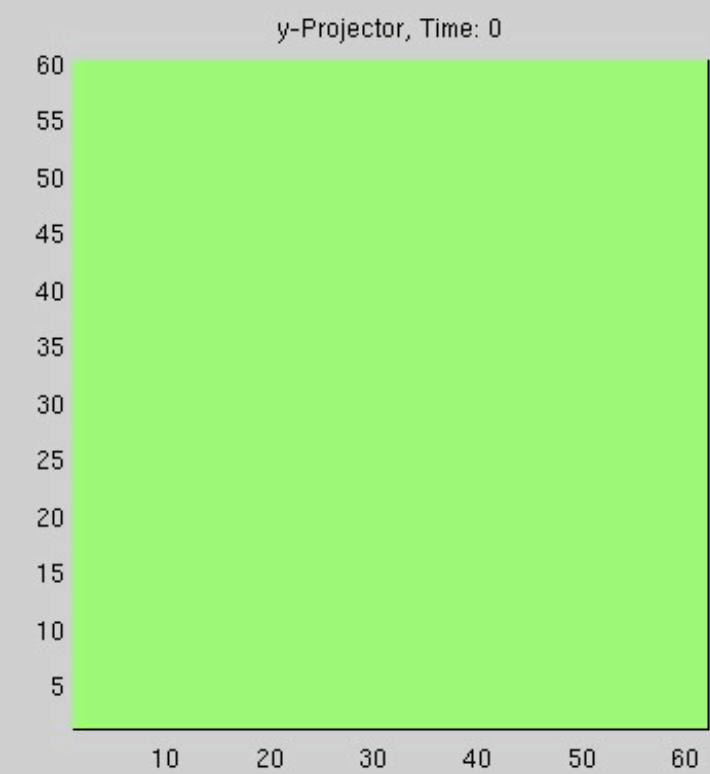
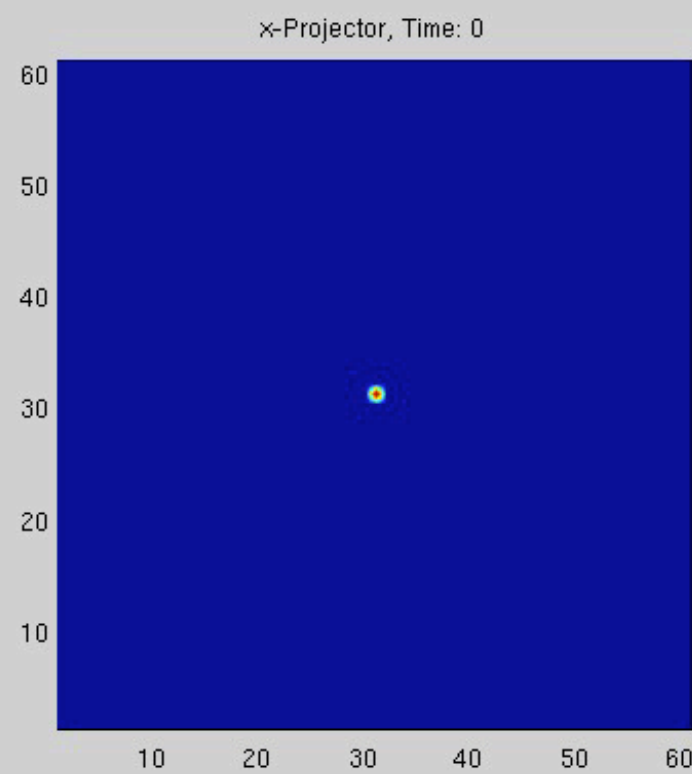
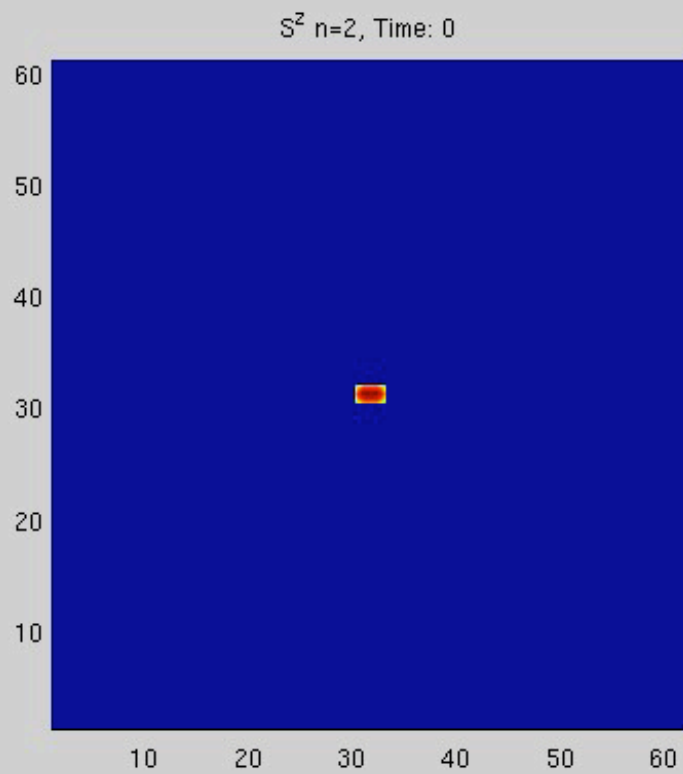
- Single particle spreading (on a $\sim 190 \times 190$ lattice)



Two-dimensional square lattice



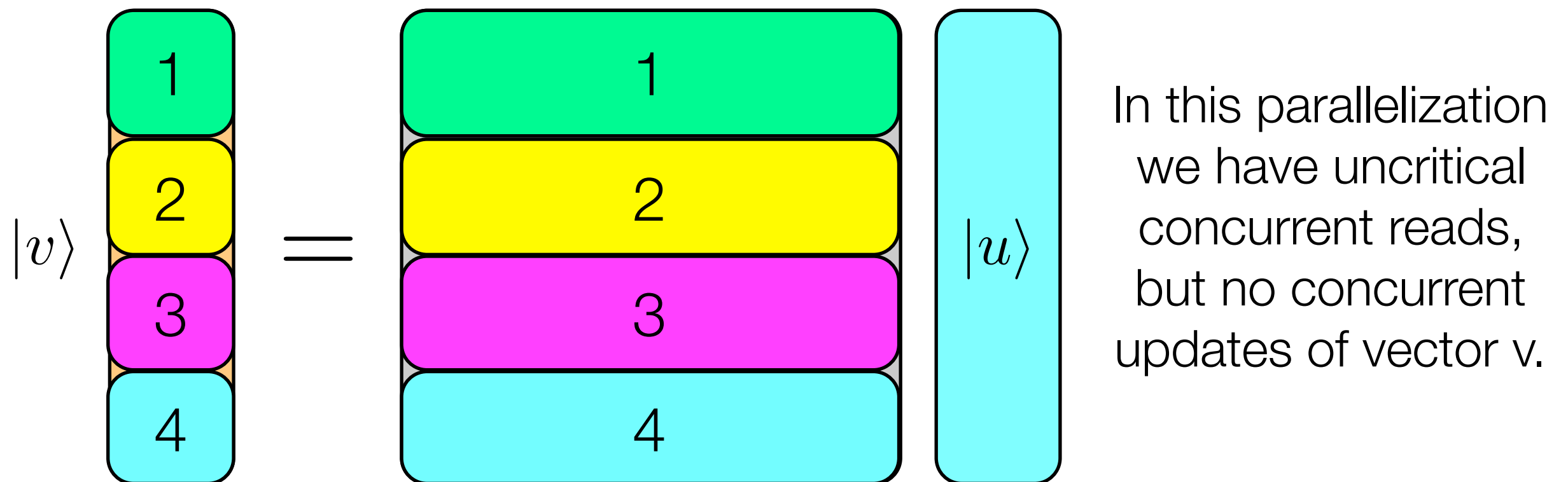
● two neighboring particles (aligned in x direction) on a 60x60 lattice



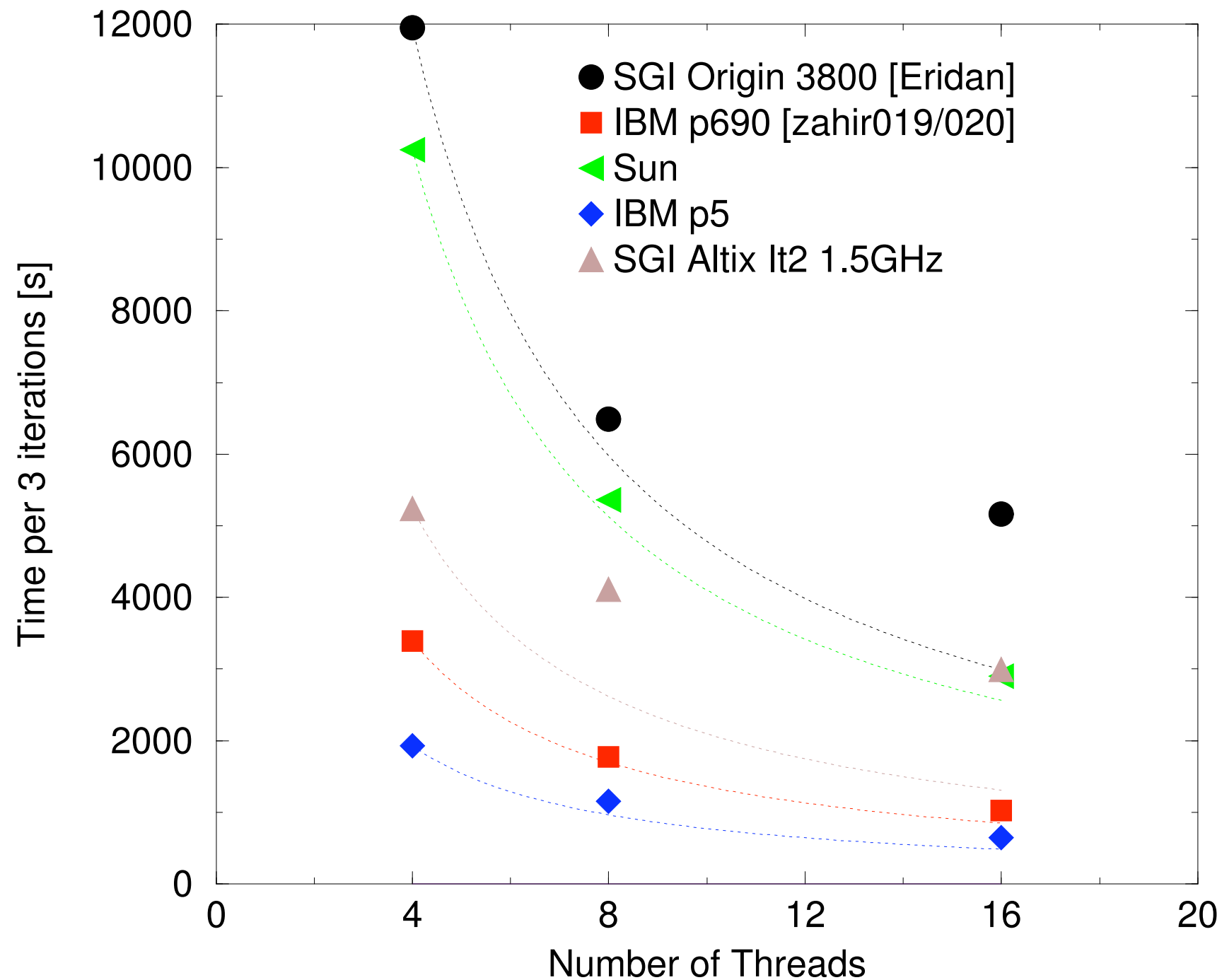
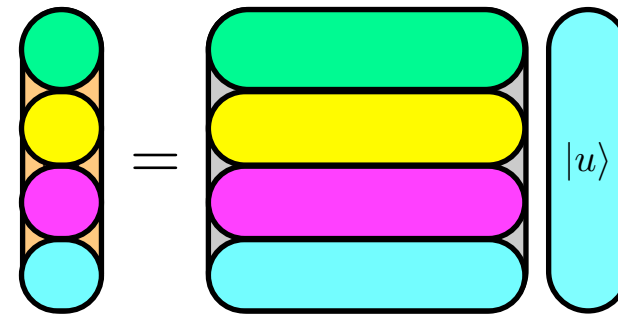
Parallelization Strategies

Parallelization: Shared memory nodes

- In the Lanczos algorithm the heaviest part is the elementary matrix-vector multiplication.
- In a matrix-free formulation this part can easily be parallelized using OpenMP pragmas in the code, even on your multi-core workstation.
Choose the right strategy between pull and push !



Parallelization: Shared memory nodes

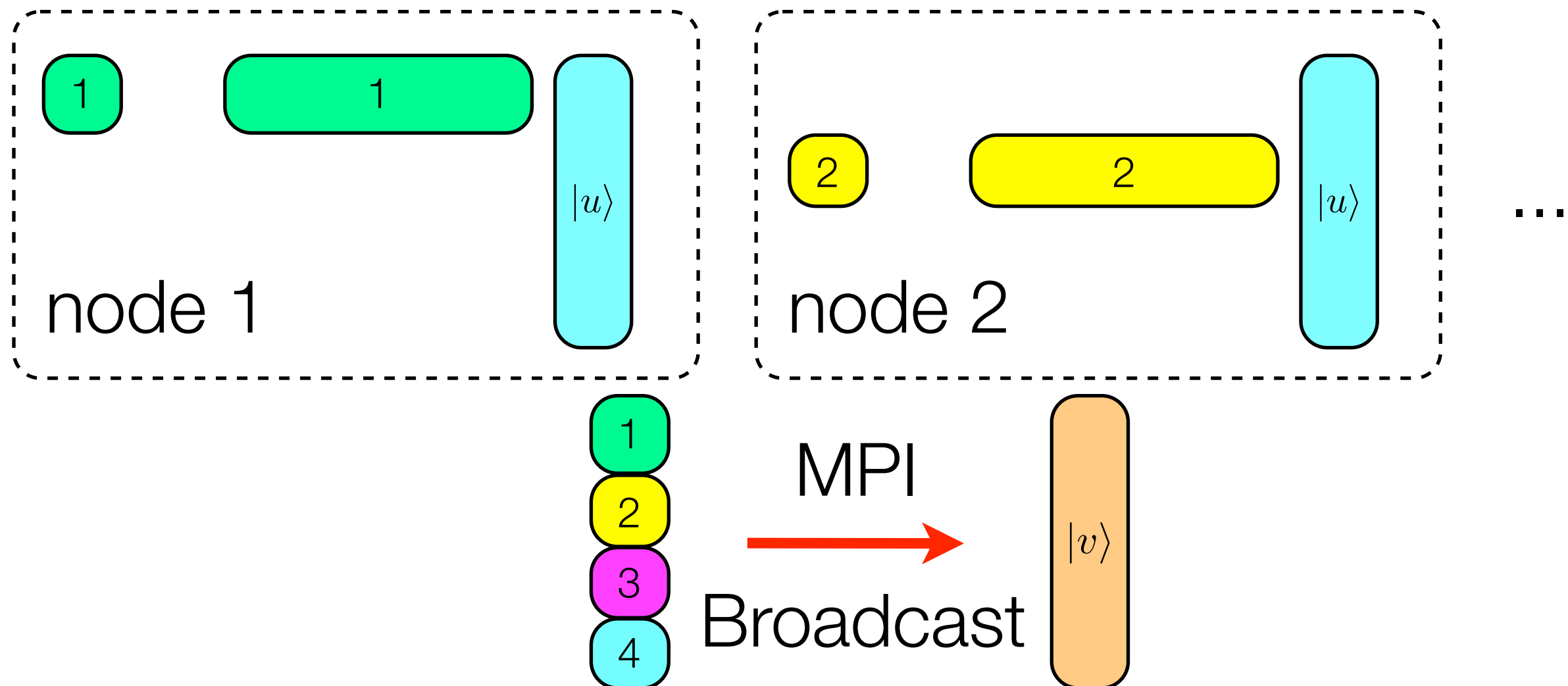


● scales well up to a few ten threads on “memory uniform” SMP machines.

Parallelization:

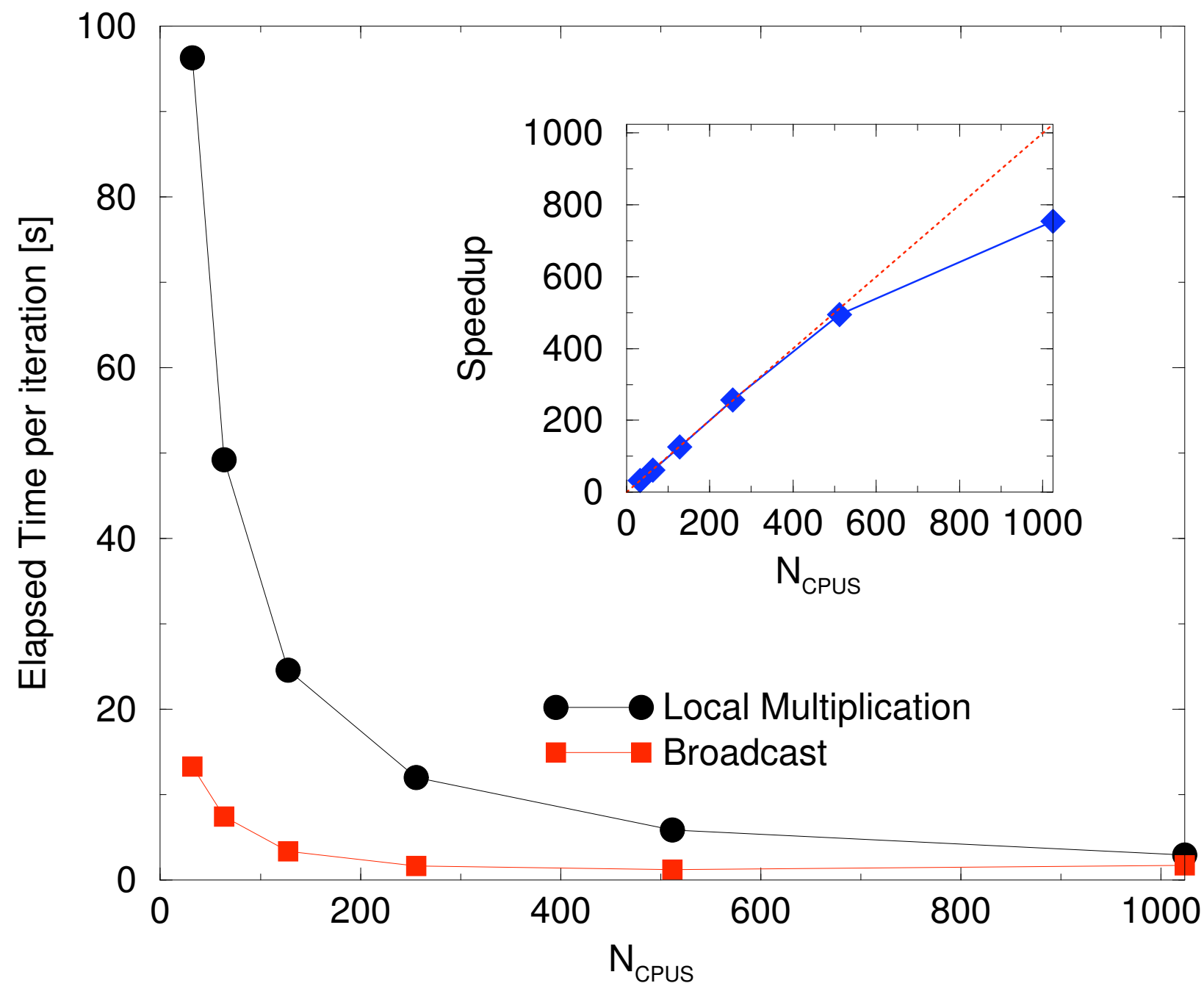
Distributed memory nodes

- For some classes of problems the Hilbert space size is not too big, but the vast number of matrix elements is a challenge.
[ED in momentum space formulation & Quantum Hall problems]
- These problems can be OpenMP parallelized, but are also suitable for large scale Message passing parallelization.



Parallelization: Distributed memory nodes

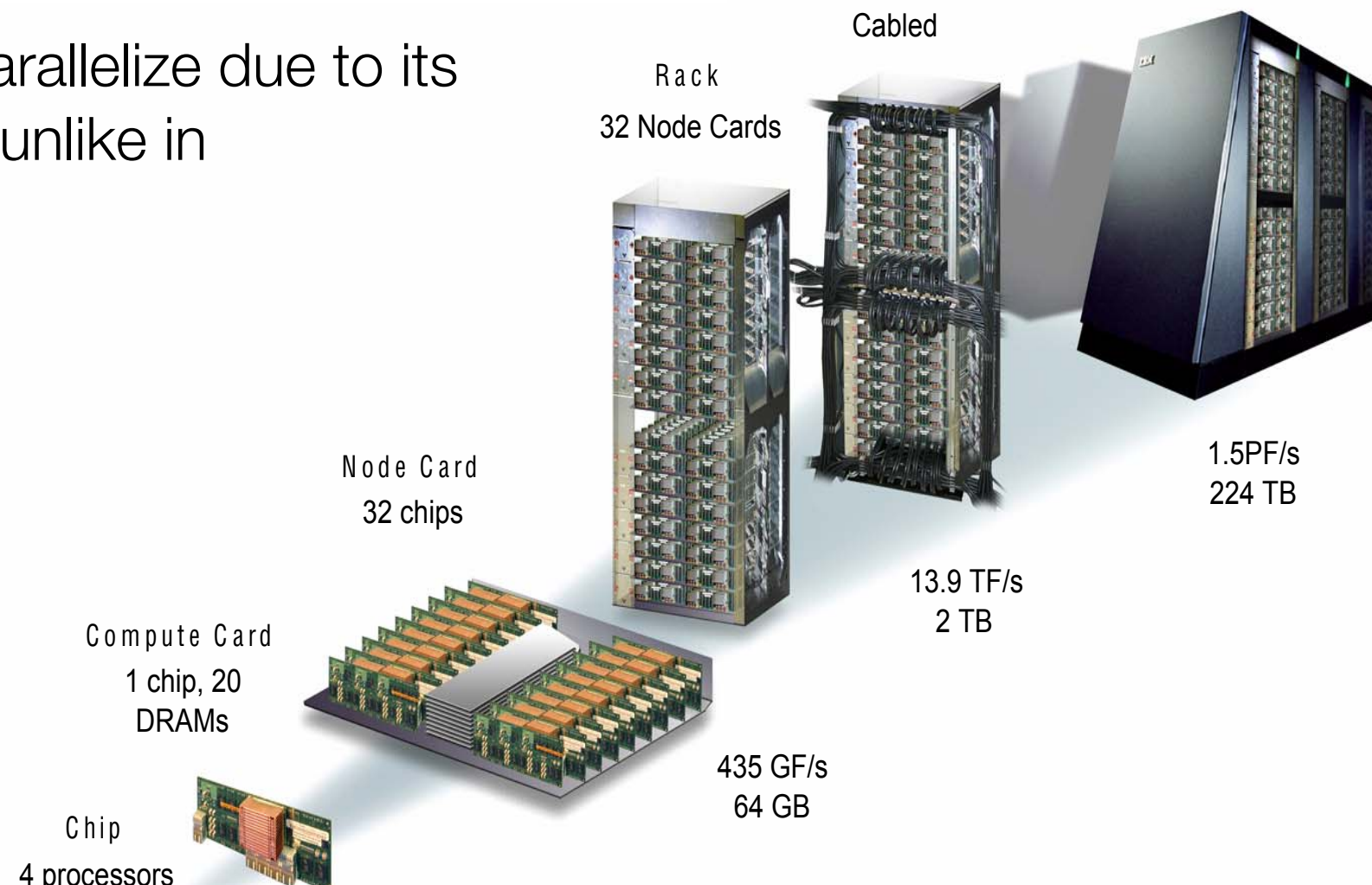
- Strong scaling example RG-ED: matrix dimension 10 million
performed on a 1024 node Cray XT-3 machine: speedup of ≈ 800 on 1024 procs



Parallelization:

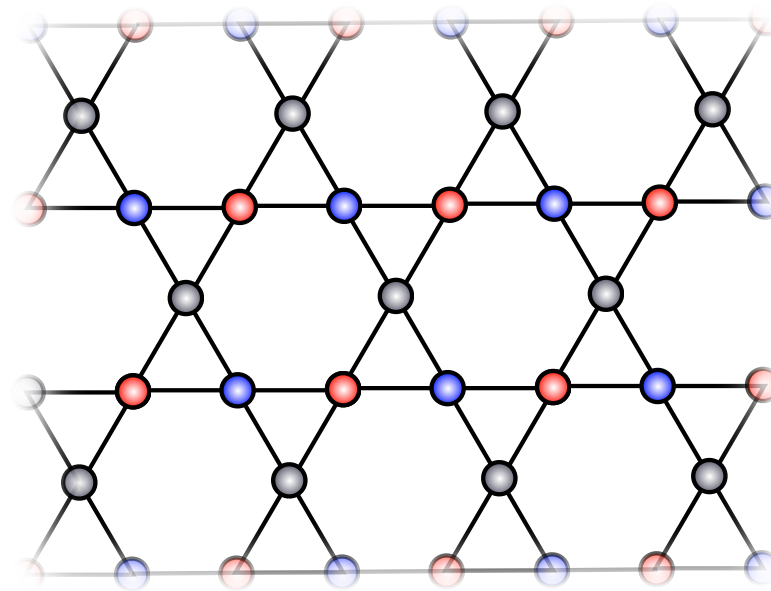
How to harness the petaflop computers ?

- Cutting edge petaflop systems have a huge number of core, but only a moderate amount of node-local memory.
- Next generation ED codes need to be developed in order to attack e.g. the huge Hilbert space of a 48 site kagome antiferromagnet.
- Problem remains difficult to parallelize due to its all-to-all structure. No locality unlike in PDE solvers.



MPI Parallel Kagome ED Code: Technical Aspects

- Three-sublattice stable symmetry implementation for fast lookups

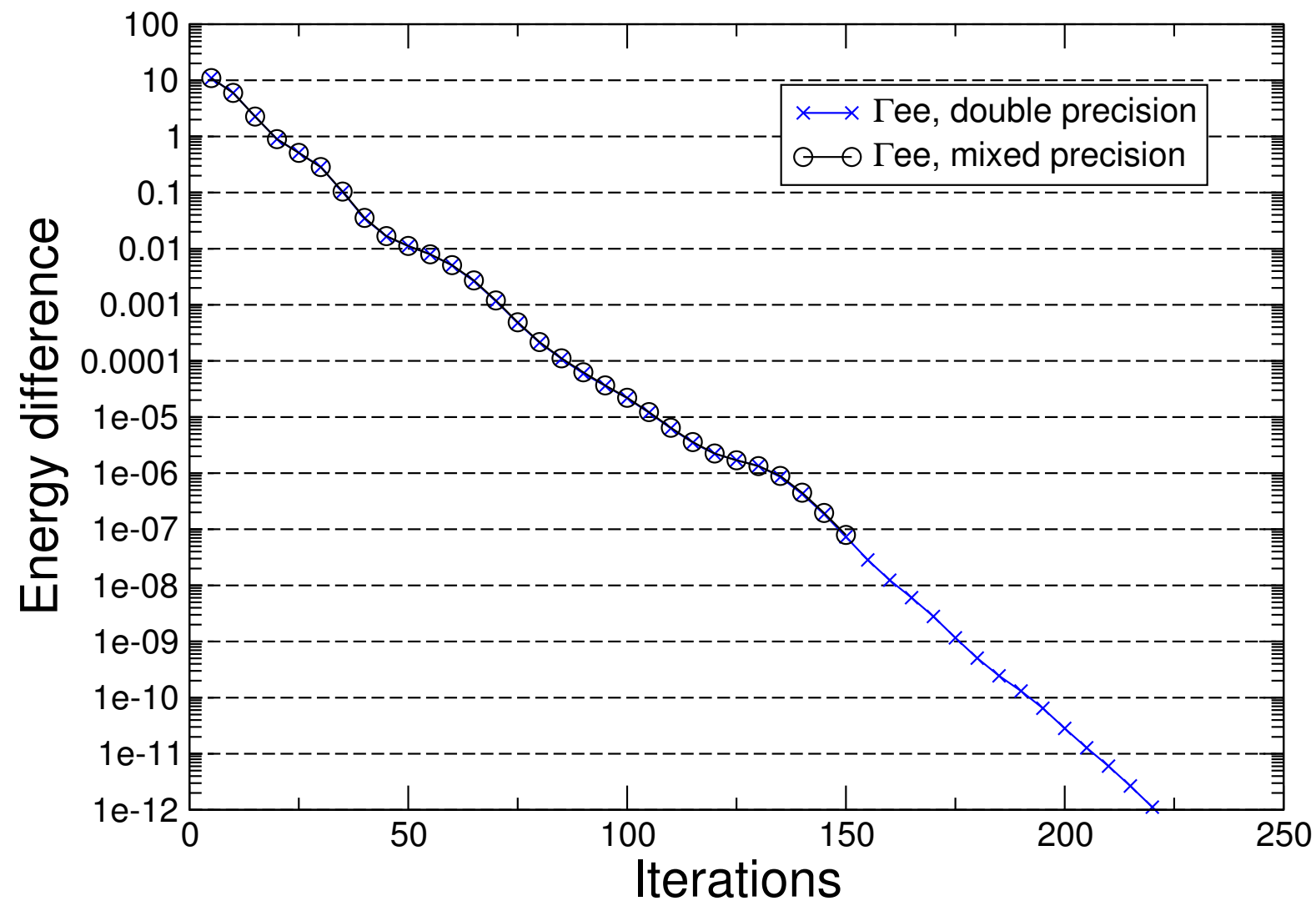


- MPI protocol based implementation for distributed memory architectures
- Performance (memory requirements up to 12 Terabytes)

Lattice	size of Hilbert space	number of tasks (architecture)	time per iteration
kagome $N_s = 42$	19,223,570,420	1,024 (Intel Xeon Infiniband)	74 seconds
kagome $N_s = 48$	251,936,333,376	1,600 (Intel Xeon NUMAlink5)	1,450 seconds
kagome $N_s = 48$	251,936,333,376	3,072 (Intel Xeon Infiniband)	650 seconds
kagome $N_s = 48$	251,936,333,376	16,384 (BlueGene/P)	520 seconds

Convergence

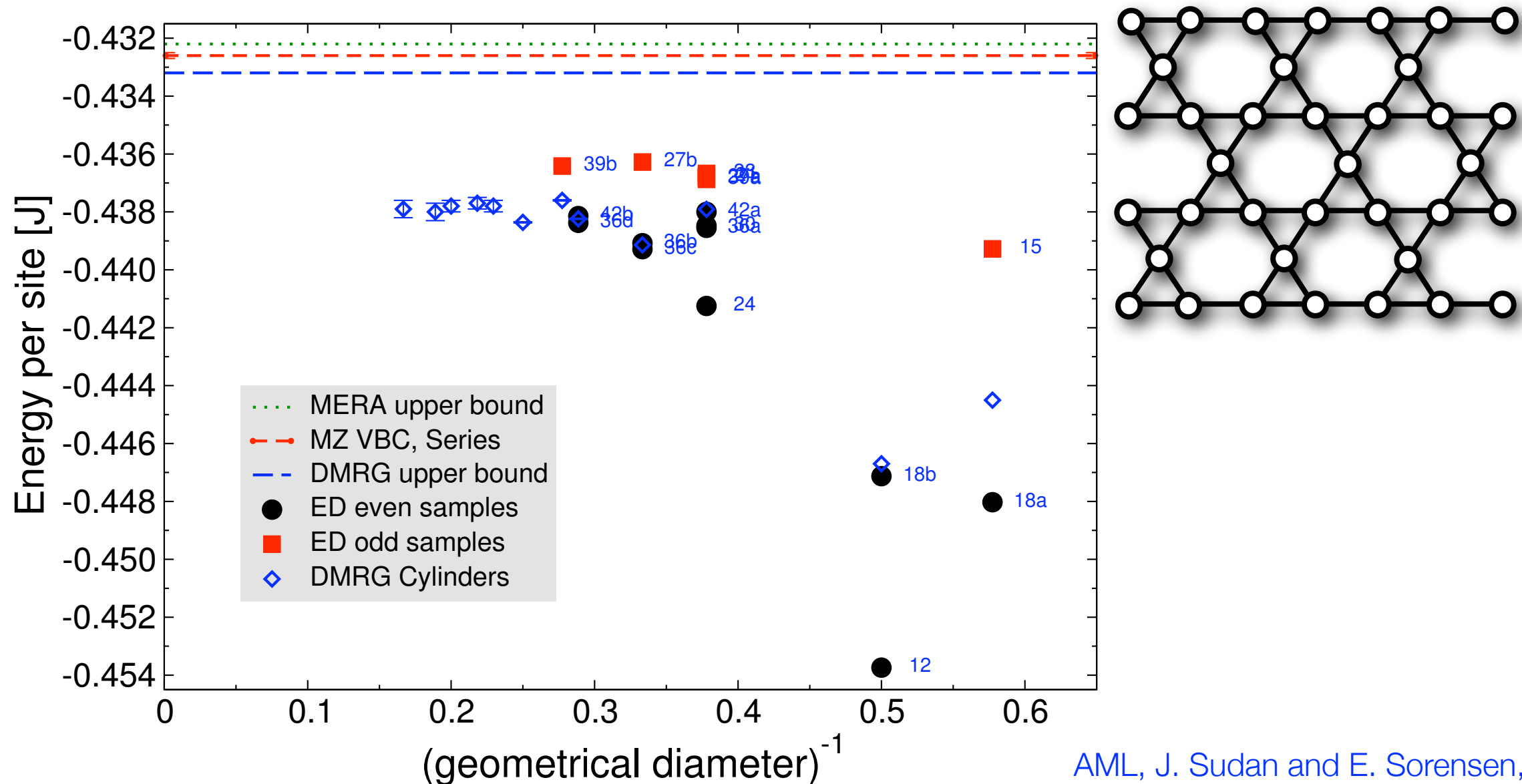
- Convergence for such large Hilbert spaces ? Finite precision arithmetic ?
Seems ok



- Upper end of spectrum converges to known energy of the ferromagnetic state !
ok !

MPI parallel ED for the kagome lattice without spatial symmetries

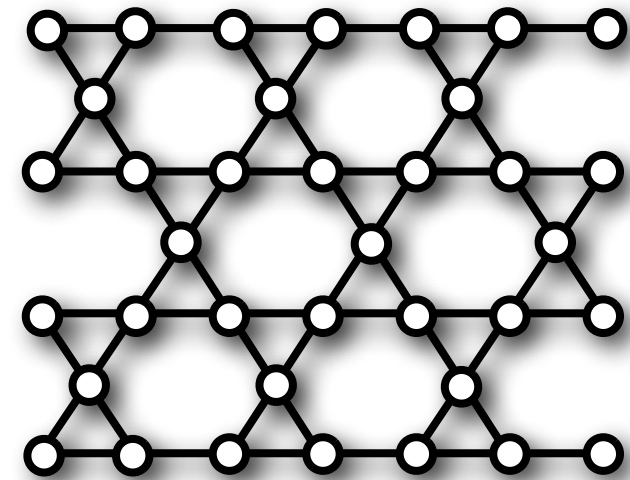
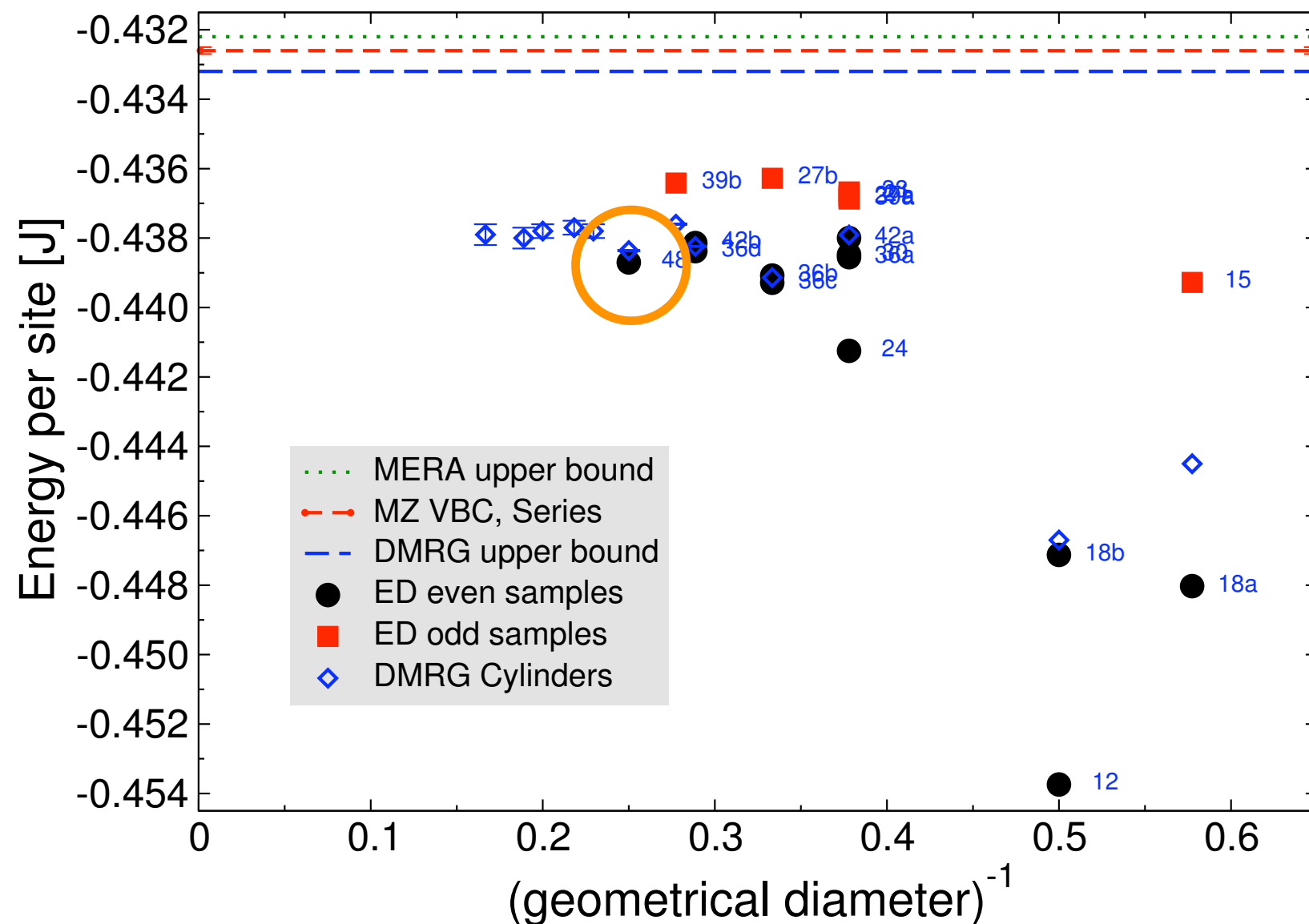
- ED Energy per site as a function of diameter (N up to 42 sites, dim up to 270×10^9).



- MPI implementation (with R. Johanni, MPG RZG):
180 seconds per iteration on 2048 cores on an 4x QDR IB Xeon cluster

MPI parallel ED for the kagome lattice now including spatial symmetries

- A new data point for **N=48** sites (251'936'333'376 states in GS sector)



- MPI implementation (with R. Johanni, MPG RZG):
520 seconds per iteration on 16'384 cores on the BlueGene/P @ MPG RZG

Exact Diagonalization Literature

- N. Laflorencie & D. Poilblanc,
"Simulations of pure and doped low-dimensional spin-1/2 gapped systems"
[Lecture Notes in Physics 645, 227 \(2004\).](#)
- R.M. Noack & S. Manmana,
"Diagonalization- and Numerical Renormalization-Group-Based Methods for Interacting Quantum Systems",
[AIP Conf. Proc. 789, 93 \(2005\).](#)
- A. Weisse, H. Fehske
"Exact Diagonalization Techniques"
[Lecture Notes in Physics 739, 529 \(2008\).](#)
- A. Läuchli
"Numerical Simulations of Frustrated Systems"
[in "Highly Frustrated Magnetism", Eds. Lacroix, Mendels, Mila, \(2011\).](#)
available upon e-mail request.

Thank you !