

## Numerical method

### Eigenvalue problem definition

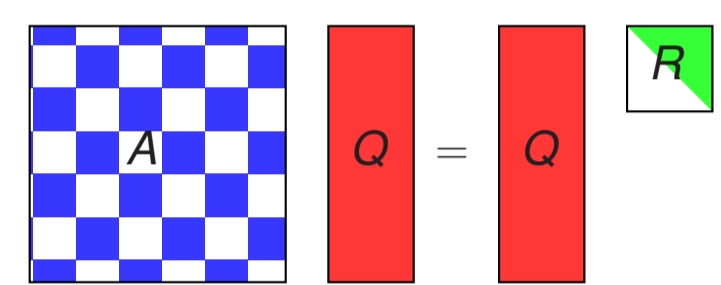
Calculate a small number of extremal eigenpairs  $(\lambda_i, v_i)$  for a sparse, large matrix  $A \in \mathbb{C}^{n \times n}$ :

$$Av_i = \lambda_i v_i, \quad i = 1, \dots, l.$$

With an orthonormal basis  $Q = (q_1, \dots, q_l)$  for the invariant subspace  $\mathcal{V} = \text{span}\{v_1, \dots, v_l\}$  one obtains the more stable **block formulation**:

$$\begin{cases} AQ - QR = 0, \\ -\frac{1}{2}Q^*Q + \frac{1}{2}I = 0. \end{cases}$$

→ Partial Schur decomposition with  $r_{ij} = \lambda_j$ :



### Block correction equation

In each step of a block Jacobi-Davidson algorithm one calculates correction vectors  $\Delta q_1, \dots, \Delta q_l$ :

$$(I - \tilde{Q}\tilde{Q}^*)(A - \tilde{\lambda}_i I)(I - \tilde{Q}\tilde{Q}^*)\Delta q_i \approx -(A\tilde{q}_i - \tilde{Q}\tilde{r}_i),$$

projection onto  $\tilde{Q}$

Here  $(\tilde{Q}, \tilde{R})$  is the current approximation with  $\tilde{\lambda}_i = r_{ii}$  and the column vectors  $\tilde{r}_i$  of  $\tilde{R}$ .

### Comparison to the single-vector JDQR

The single-vector JDQR correction equation is  $(I - \tilde{Q}\tilde{Q}^*)(A - \tilde{\lambda}_i I)(I - \tilde{Q}\tilde{Q}^*)\Delta q_i \approx -(I - Q_k Q_k^*)(A\tilde{q}_i - \tilde{\lambda}_i \tilde{q}_i)$  with converged Schur vectors  $Q_k$  and  $\tilde{Q} = (Q_k \tilde{q}_i)$ .  
→ Both RHS represent deflated residuals.  
→ We use a deflation with the complete block  $\tilde{Q}$ .

### Additional operations due to blocking

- **Blocking increases the number of operations** (but blocked operations are faster).
- Question: How large is the overhead?
- Approach to estimate possible performance gains:
  - Count sparse matrix-vector multiplications (spMVM).
  - Relate results to the performance of block spMVMs.
- For more than 20 eigenpairs blocking may be beneficial.

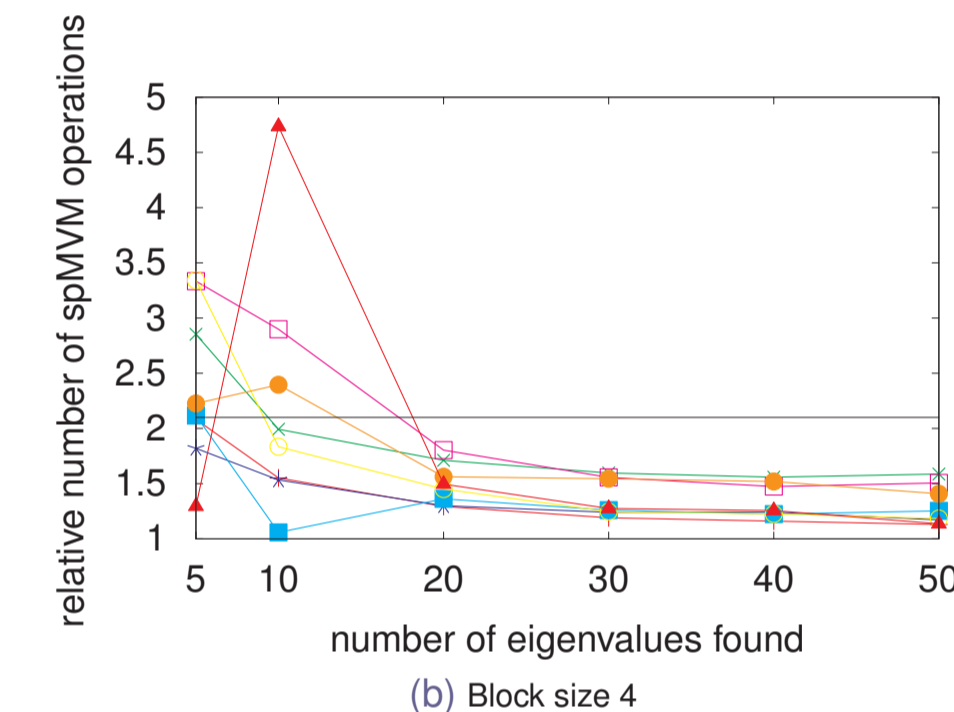
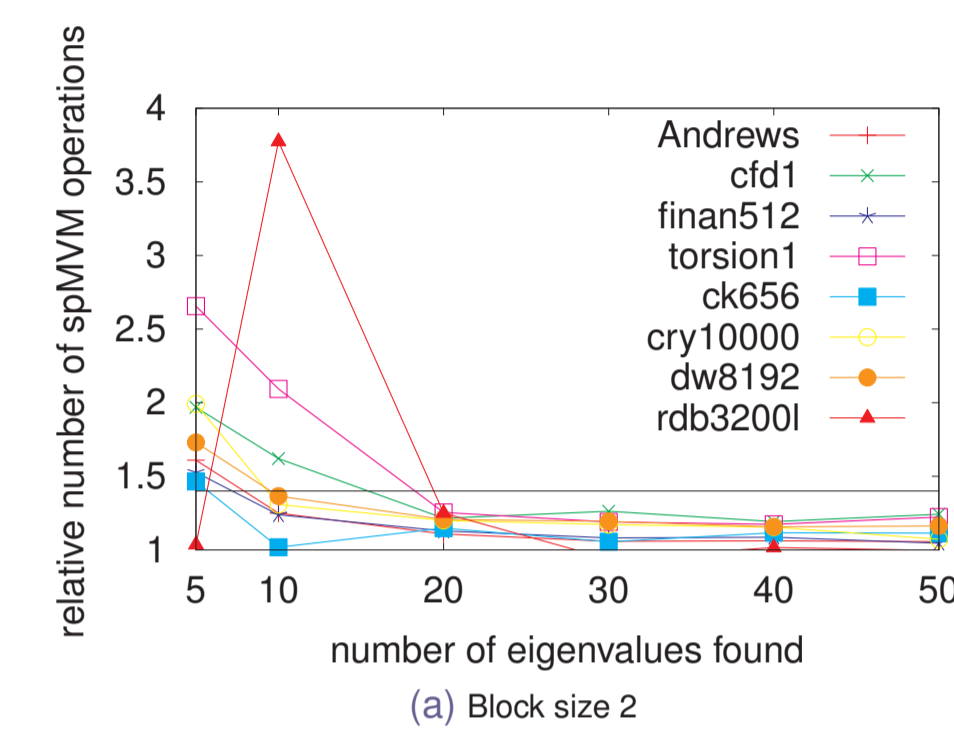


Figure 1: Number of spMVMs of block JDQR compared to single-vector JDQR. The black horizontal line indicates the spMVM block-speedup for a representative matrix (cf. Fig. 4)

## Performance engineering of the key operations

### Jacobi-Davidson operator

- Required in each inner iteration (iterative solver for the block correction equation)
- Calculate block vector  $Y$  for given  $X = (x_1, \dots, x_l)$  with  $Y_j \leftarrow (I - QQ^T)(A - \tau_j I)x_j$ .
- Shifted sparse matrix–multiple-vector multiplication (spMMVM) can be applied in one step

### Sparse matrix–multiple-vector multiplication

- The block vector storage scheme matters (see Fig. 4):  
→ Column-major scheme (standard) not beneficial  
→ **Row-major scheme significantly faster**
- Experiments with different sparse matrix formats (CRS vs. SELL-C- $\sigma$ )
- The Roofline performance model gives helpful insight.

### SELL-C- $\sigma$ format [2]

- parameterized “sliced ELLPACK”
- competitive on CPU, GPU and MIC alike
- $C$ : chunk size (zero padding)
- $\sigma$ : sorting scope (to reduce overhead)

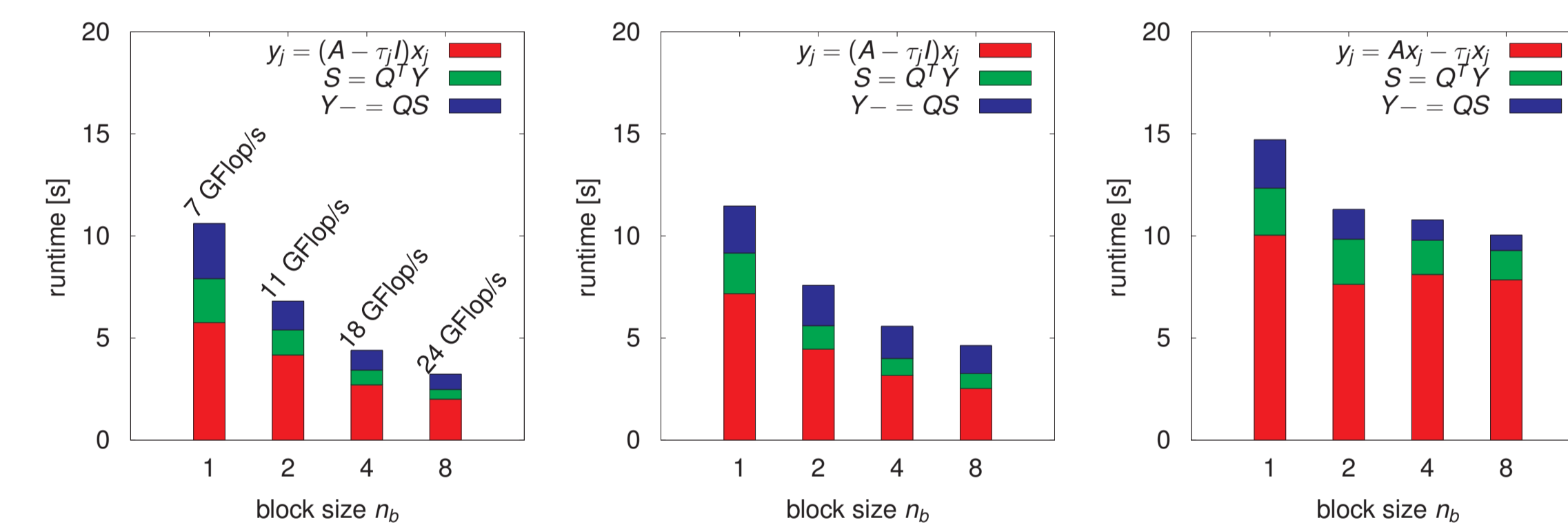
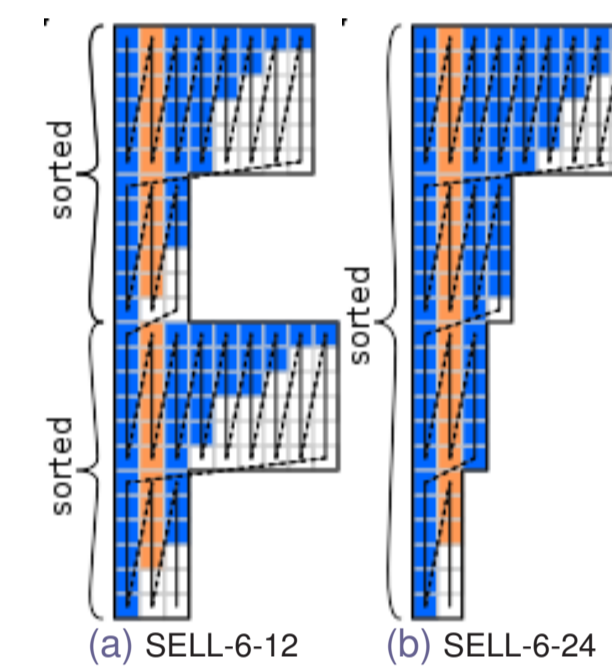


Figure 4: Single-socket performance of key operations (spMVM+projections) with the SIMD-friendly SELL-C- $\sigma$  matrix format [2] from GHOST (left) vs. the standard CRS format (center) and the Epetra CRS format (right). The latter package uses column-major ordering for block vectors and requires an additional copy operation of the entire input vector (‘import’).

### Block vector operations

- Block vectors are dense ‘tall skinny’ matrices.  
→ GEMM operations
- Performance is memory bound due to operands shape (and well predicted by the roofline model).
- Hand-optimized code faster than common BLAS libraries

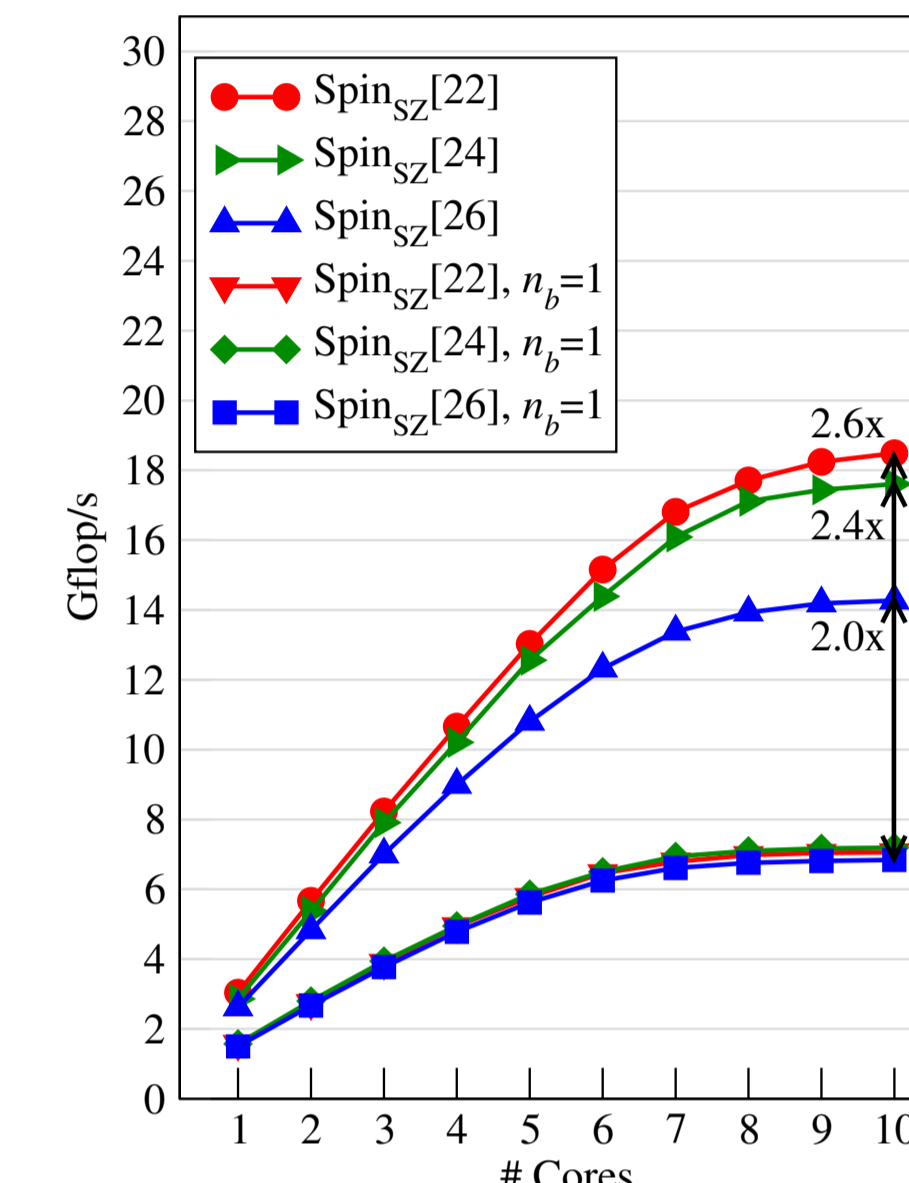


Figure 3: Intra-socket scaling of spMMVM for spin matrices of different sizes (Spin<sub>sz</sub>[22] ... Spin<sub>sz</sub>[26])

## Block performance in strong scaling experiments

### Setup

- Sparse matrices and vectors distributed on a cluster of 1-64 nodes (using MPI)
- Dual socket nodes with 10 cores per socket (using OpenMP parallelization)
- Intel Xeon E5-2660 v2 CPUs (‘Ivy bridge’) at 2.20 GHz

### Results

- Significant speedup of Jacobi-Davidson through blocking in contrast to the conclusion in [1]
- This holds for strong scaling tests on up to 1280 cores.
- Small block sizes (2 or 4) are beneficial.
- Further effects of blocking:
  - Total communication volume increases (spMVMs).
  - Message aggregation may improve the performance.
- See [3] for a detailed discussion.

### Future work

- Overlap communication and computation:
  - alleviate increased data traffic due to blocking
- Use accelerator hardware such as GPUs.

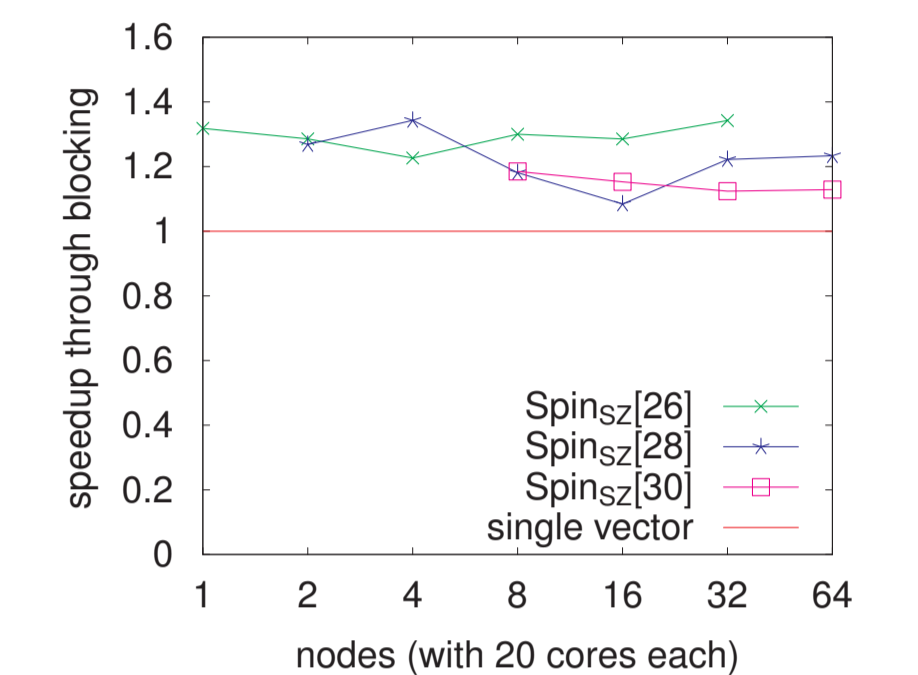


Figure 7: Relative performance gains with block size 2.

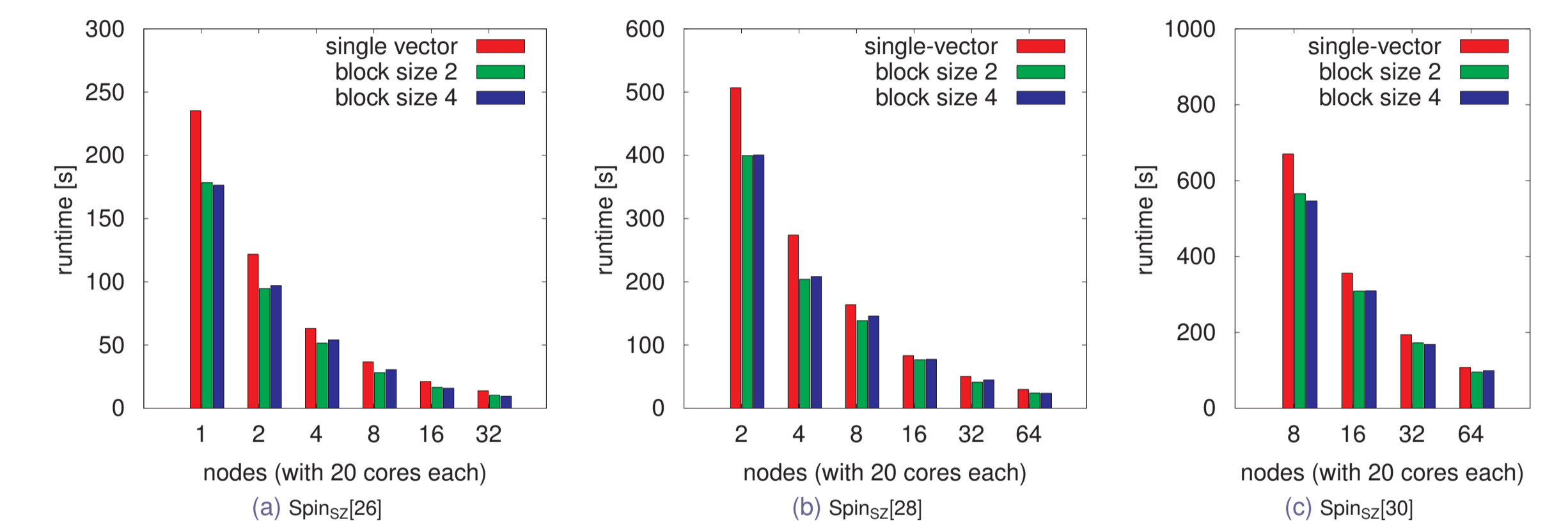


Figure 8: Strong scaling results for block sizes 2 and 4.

## Blocked linear solvers

### Requirements for blocked iterative solvers

- Concurrently solve  $l_b$  systems with different shifts.
- Group together similar operations of different systems.
- Employ faster block spMVM and block-vector operations.
- Reduce the number of MPI messages.
- Dynamic queue:
  - Remove converged systems.
  - Enqueue new systems.

### Blocked GMRES algorithm

- Standard restarted GMRES method (unpreconditioned)
- Single iteration:
  - 1: Apply operator to preceding basis vector  $(\tilde{v}_{k+1} \leftarrow (I - \tilde{Q}\tilde{Q}^*)(A - \tilde{\lambda}_k)\tilde{v}_k)$ ,
  - 2: orthogonalize  $\tilde{v}_{k+1}$  wrt. all previous basis vectors,
  - 3: perform local operations (Givens rotations, ...).
- Basis vectors stored as blocks in a ring buffer (Fig. 2)
- Individual systems can be restarted (when buffer is full).

### Blocked MINRES algorithm

- Standard MINRES method (unpreconditioned)
- Very similar to blocked GMRES:
  - Based on Lanczos recurrence instead of modified Gram-Schmidt (for orthogonalization).
  - Store subspace and update result at the end.

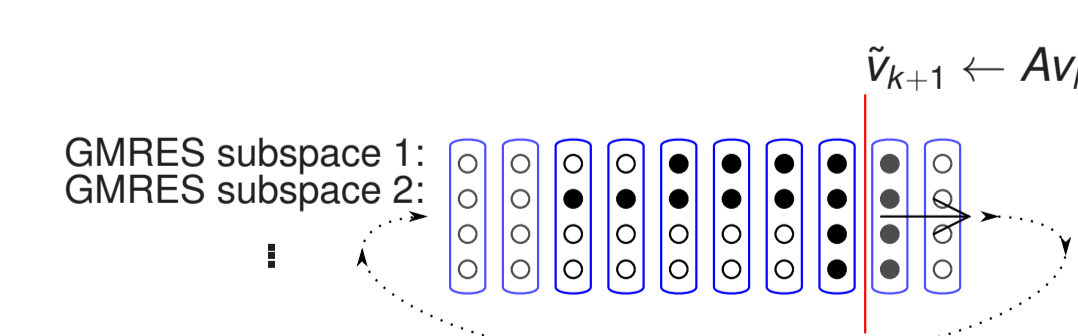


Figure 2: A partly filled ringbuffer for 4 systems. The data structure allows block operations while using different Krylov subspaces.

## Applications from quantum physics

### The Spin<sub>sz</sub>[L] matrices

- Generic benchmark problem from quantum physics
- Chain of  $L$  electron spins  $1/2$ , closed to a ring (Fig. 5)
- Computational representation of Hamilton operator in terms of bit patterns & bit swap/flip operations
- Find a few eigenvalues at the left end of the spectrum (lowest energy states)
- Symmetric matrix, can have lots of multiple eigenvalues
- Matrix dimension  $(L/2)$  grows exponentially with  $L$

name	number of rows	non-zero count
Spin <sub>sz</sub> [26]	$1.0 \cdot 10^7$	$1.5 \cdot 10^8$
Spin <sub>sz</sub> [28]	$4.0 \cdot 10^7$	$6.1 \cdot 10^8$
Spin <sub>sz</sub> [30]	$1.6 \cdot 10^8$	$2.6 \cdot 10^9$

Table 1: Dimension of the matrices used in the experiments.

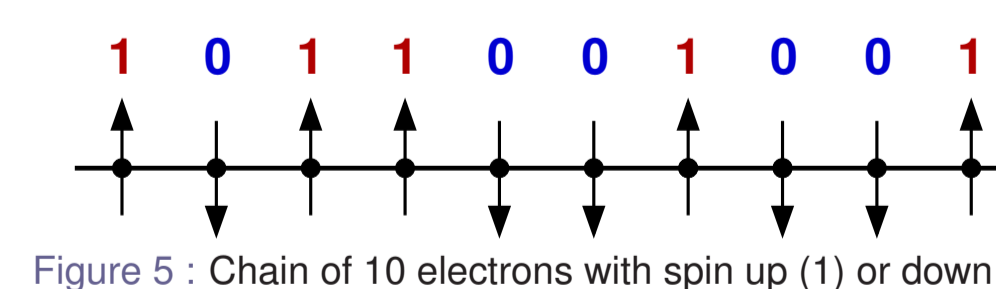


Figure 5: Chain of 10 electrons with spin up (1) or down (0).

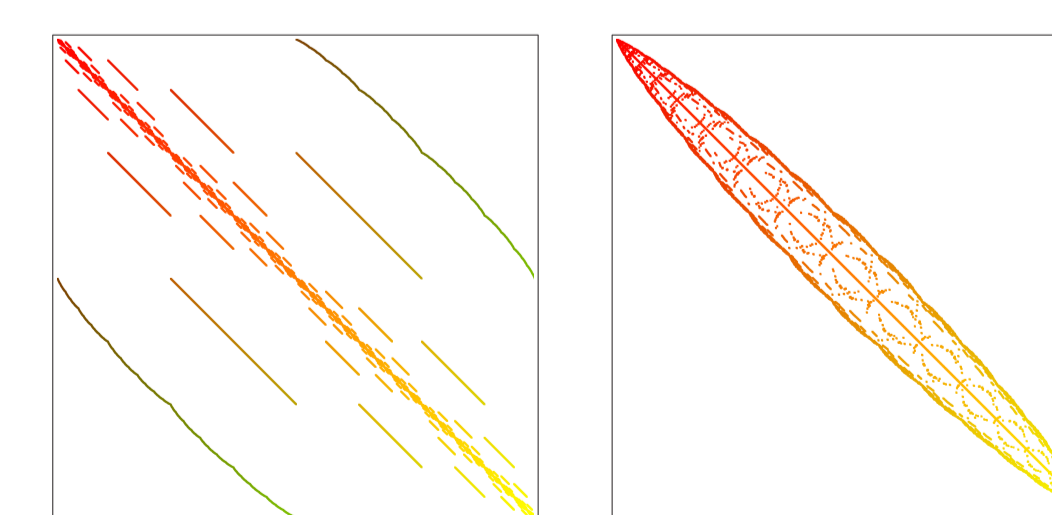


Figure 6: Typical sparsity structure of the Spin<sub>sz</sub>[L] matrices (here  $L = 20$ ). On the right a bandwidth-reducing reordering was applied.

## Software

<p><b>PHIST</b> (Pipelined Hybrid-parallel Iterative Solver Toolkit)</p> <ul style="list-style-type: none"> <li>it. lin. syst. solvers (e.g. Krylov methods)</li> <li>it. eigenvalue solvers (e.g. BJDQR, FEAST)</li> </ul>	<p><b>GHOST</b> (General Hybrid Optimized Sparse Toolkit)</p> <ul style="list-style-type: none"> <li>optimized kernels (e.g. <math>Y \leftarrow AX, C \leftarrow V^T W</math>) using MPI, GPI, OpenMP, Pthreads, CUDA etc.</li> </ul>	<p><b>PHYSICS</b> (quantum physics applications)</p> <ul style="list-style-type: none"> <li>e.g. graphene modelling, topological insulators, spin chains, ...</li> </ul>
<p>Abstract kernel interface, use alternatively</p> <p>‘stand-alone’ (Fortran+MPI+OpenMP)</p> <p><b>GHOST</b></p> <p>TPLs, e.g. Trilinos (Epetra or Tpetra)</p>		
<p>While the software from the ESSEX project has not been publicly released, users interested in early testing are welcome to contact the authors. Latest news and contact information:</p> <ul style="list-style-type: none"> <li>• <a href="http://blogs.fau.de/essex/">http://blogs.fau.de/essex/</a></li> <li>• GHOST: moritz.kreutzer@fau.de</li> <li>• PHIST: jonas.thies@dlr.de</li> <li>• PHYSICS: alvermann@physik.uni-greifswald.de</li> <li>• preprint on the topic available [3]</li> </ul>		

## References

- [1] Stathopoulos & McCombs. Nearly optimal preconditioned methods for Hermitian eigenproblems under limited memory. Part II: Seeking many eigenvalues. *SISC*, 29(5), 2007.
- [2] Kreutzer *et al.* A unified sparse matrix data format for modern processors with wide SIMD units. *SISC (accepted) arXiv:1307.6209*, 2014.
- [3] Röhrig-Zöllner *et al.* Increasing the performance of the Jacobi-Davidson method by blocking. *SISC (submitted) http://eelib.dlr.de/89980/*, 2014.