Performance of Block Jacobi-Davidson

M. Röhrig-Zöllner[1], J. Thies[1], M. Kreutzer[2], A. Alvermann[3], A. Pieper[3], A. Basermann[1], G. Hager[2], G. Wellein[2], and H. Fehske[3]

[1]DLR, Simulation and Software Technology,
[2]Erlangen Regional Computing Center, [3]Physics, University of Greifswald

Deutsches Zentrum für Luft- und Raumfahrt
German Aerospace Center

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG

Numerical method

Eigenvalue problem definition

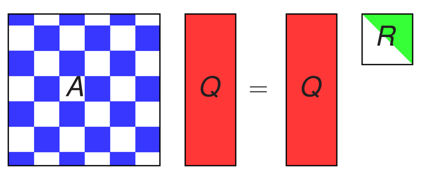
Calculate a small number of extremal eigenpairs (λ_i, v_i) for a sparse, large matrix $A \in \mathbb{C}^{n \times n}$:

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

With an orthonormal basis $Q = (q_1, \dots, q_l)$ for the invariant subspace $\mathcal{V} = \mathrm{span}\{v_1, \dots, v_l\}$ 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 $r_{i,i} = \lambda_i$:



Block correction equation

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

$$\underbrace{(I - \tilde{Q}\tilde{Q}^*)}_{\text{projection onto } Q^\perp} (A - \tilde{\lambda}_i I)(I - \tilde{Q}\tilde{Q}^*)\Delta q_i \approx -(A\tilde{q}_i - \tilde{Q}\tilde{r}_i).$$

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

Comparison to the single-vector JDQR

The single-vector JDQR correction equation is

$$(I - \bar{Q}\bar{Q}^*)(A - \tilde{\lambda}I)(I - \bar{Q}\bar{Q}^*)\Delta q \approx -(I - Q_k Q_k^*)(A\tilde{q} - \tilde{\lambda}\tilde{q})$$

with converged Schur vectors Q_k and $\bar{Q} = (Q_k \; \tilde{q})$.
→ 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.

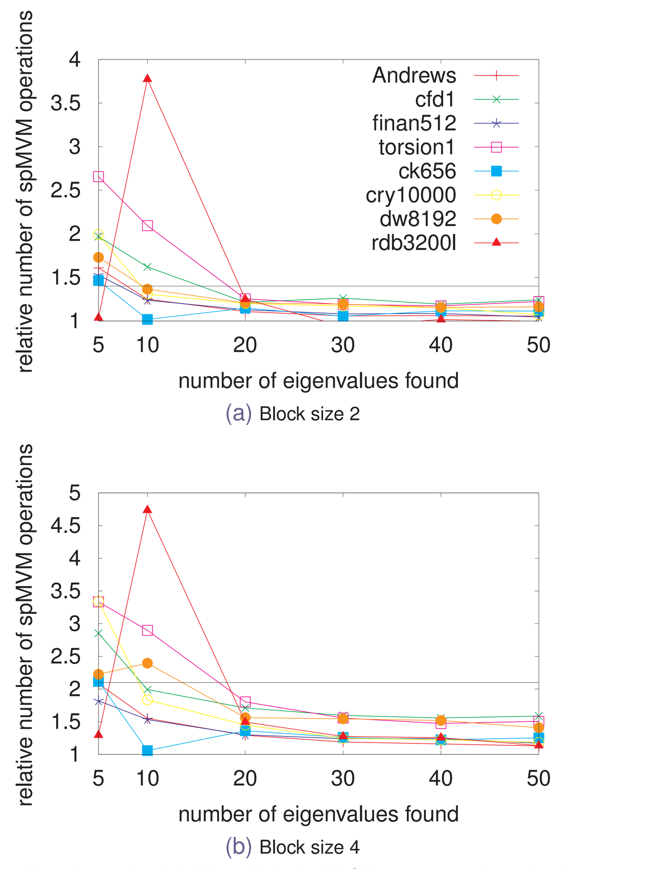


Figure 1 : Number of spMVMs of block JDQR compared to single-vector JDQR. The black horizontal line indicates the spMVM block-speedup for a representative matrix (cf. Fig. 8)

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 ($\tilde{v}_{k+1} \leftarrow (I - \tilde{Q}\tilde{Q}^*)(A - \tilde{\lambda}_j)v_k$),
  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.

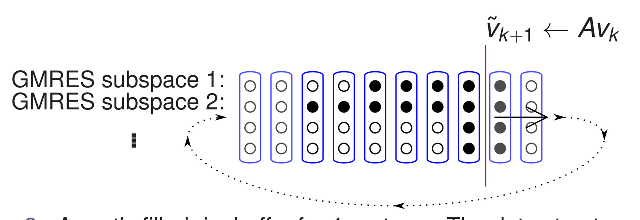


Figure 2 : A partly filled ringbuffer for 4 systems. The data structure allows block operations while using different Krylov subspaces.

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
- SELL-32-256 format for single-vector algorithm, SELL-8-32 for blocksize 2 and 4

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.

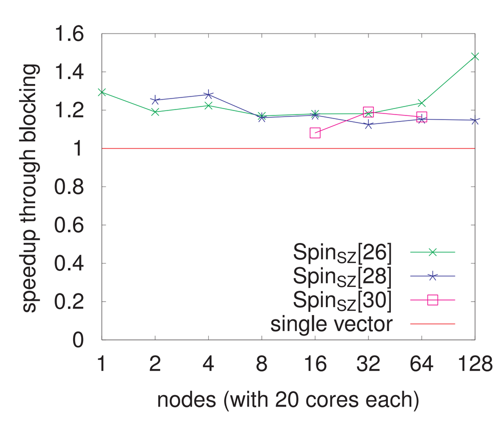


Figure 3 : Relative performance gains with block size 2.

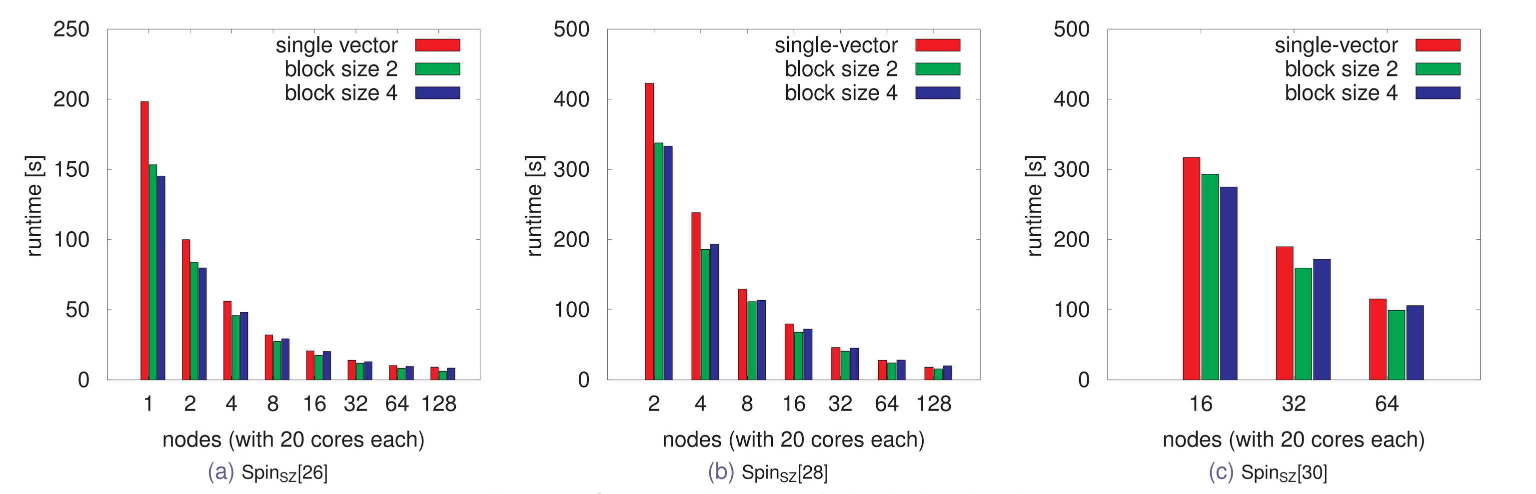


Figure 4 : Strong scaling results for block sizes 2 and 4.

Applications from quantum physics

The $\mathrm{Spin}_{SZ}[L]$ 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 $\binom{L}{L/2}$ grows exponentially with L

name	number of rows	non-zero count
$\mathrm{Spin}_{SZ}[26]$	$1.0 \cdot 10^7$	$1.5 \cdot 10^8$
$\mathrm{Spin}_{SZ}[28]$	$4.0 \cdot 10^7$	$6.1 \cdot 10^8$
$\mathrm{Spin}_{SZ}[30]$	$1.6 \cdot 10^8$	$2.6 \cdot 10^9$

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

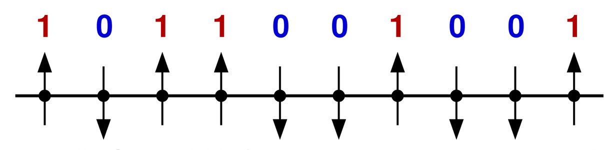


Figure 5 : Chain of 10 electrons with spin up (1) or down (0).

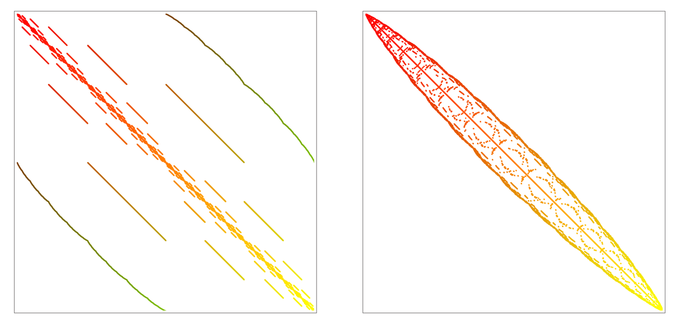


Figure 6 : Typical sparsity structure of the $\mathrm{Spin}_{SZ}[L]$ matrices (here $L = 20$). On the right a bandwidth-reducing preordering was applied.

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

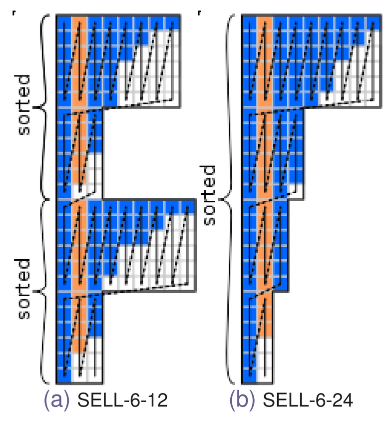
- The block vector storage scheme matters (see Fig. 8):

→ Column-major scheme (standard) not beneficial
→ Row-major scheme significantly faster

- Experiments with different sparse matrix formats (CRS vs. SELL-C-σ)
- The Roofline performance model gives helpful insight.

SELL-C-σ format [2]

- parameterized 'sliced ELLPACK'
- competitive on CPU, GPU and MIC alike
- C: chunk size (zero padding)
- σ: sorting scope (to reduce overhead)



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

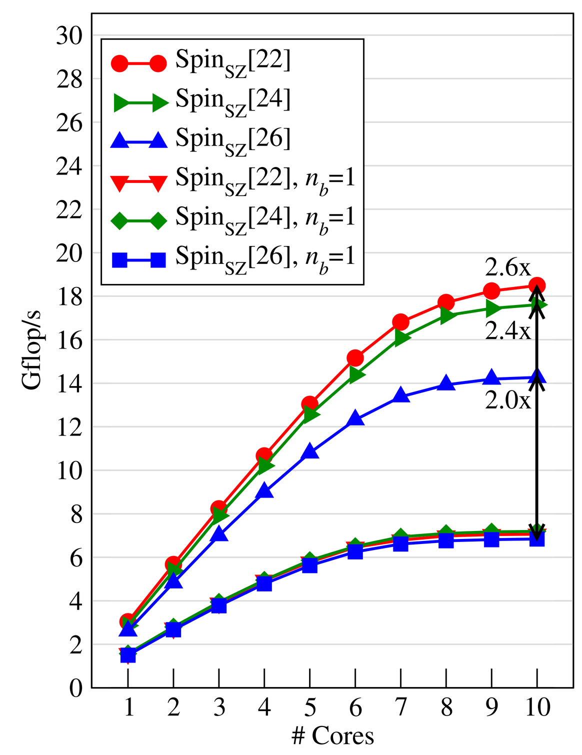


Figure 7 : Intra-socket scaling of spMMVM for spin matrices of different sizes ($\mathrm{Spin}_{SZ}[22] \dots \mathrm{Spin}_{SZ}[26]$)

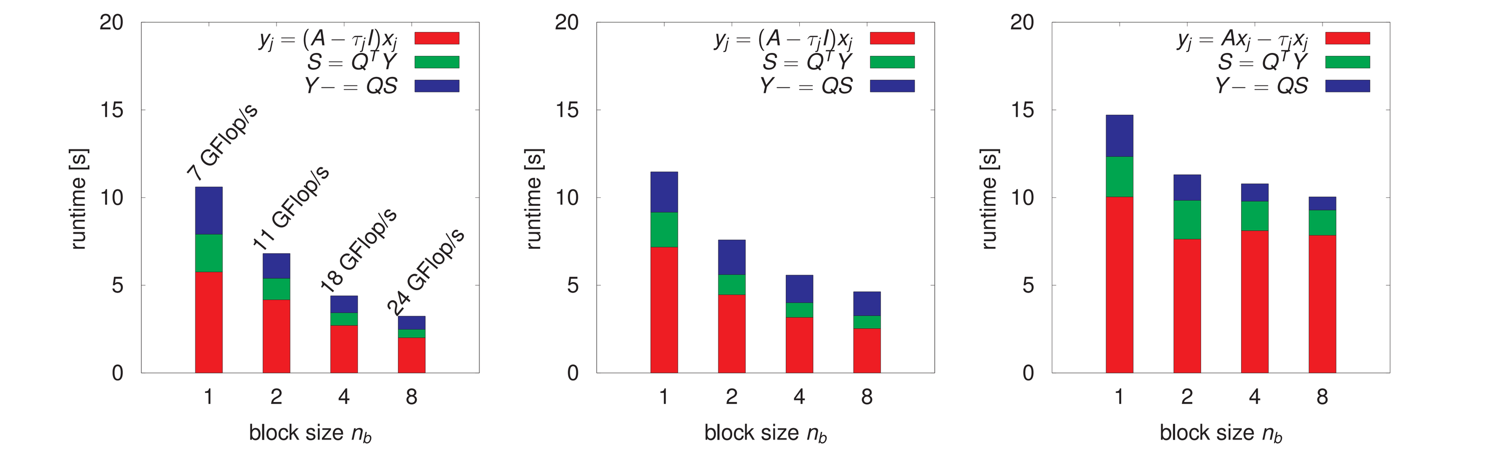


Figure 8 : Single-socket performance of key operations (spMMVM+projections) with the SIMD-friendly SELL-C-σ matrix format [2] from GHOST (left) vs. the standard CRS format (center) and the Epetra CRS format (right). The latter package uses column-major ordering for block vectors and requires an additional copy operation of the entire input vector ('import').

Software

PHIST (Pipelined Hybrid-parallel Iterative Solver Toolkit)	GHOST (General Hybrid Optimized Sparse Toolkit)	PHYSICS (quantum physics applications)
it. lin. syst. solvers (e.g. Krylov methods) it. eigenvalue solvers (e.g. BJDQR, FEAST)	optimized kernels (e.g. $Y \leftarrow AX, C \leftarrow V^T W$) using MPI, GPI, OpenMP, Pthreads, CUDA etc.	e.g. graphene modelling, topological insulators spin chains ...

Abstract kernel interface, use alternatively

- 'builtin' (Fortran+MPI+OpenMP)
- GHOST
- TPLs, e.g. Trilinos (Epetra or Tpetra)

The libraries developed in ESSEX are available under a BSD licence here:

- `https://bitbucket.org/essex/ghost`
- `https://bitbucket.org/essex/phist`
- `https://bitbucket.org/essex/physics`

Latest news and contact information:

- `http://blogs.fau.de/essex/`
- GHOST: `moritz.kreutzer@fau.de`
- PHIST: `jonas.thies@dlr.de`
- PHYSICS: `alvermann@physik.uni-greifswald.de`
- preprint on the topic available [3]

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://elib.dlr.de/89980/`, 2014.

ESSEX – Equipping Sparse Solvers for Exascale – funded by DFG priority program 1648 (SPPEXA) `http://blogs.fau.de/essex`