THE CHASE LIBRARY FOR LARGE HERMITIAN EIGENVALUE PROBLEMS

Parameter elimination and software architecture

March 1, 2019 | **E. Di Napoli**, J. Winkelmann, A. Schleife |

OUTLINE

Chebyshev Accelerated Subspace Iteration Eigensolver (ChASE): the main idea

ChASE: parameter selection

Some numerical results

OUTLINE

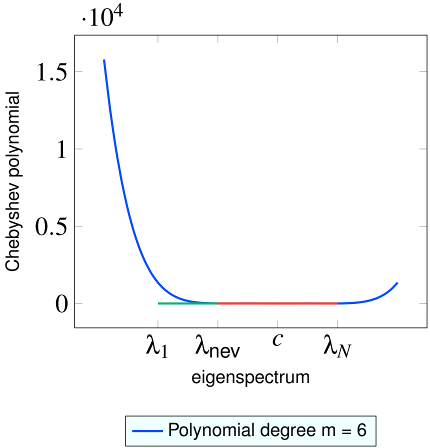
Chebyshev Accelerated Subspace Iteration Eigensolver (ChASE): the main idea

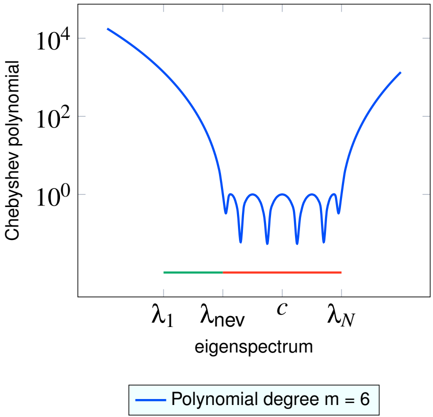
ChASE: parameter selection

Some numerical results

DIVIDE AND CONQUER

Chebyshev polynomials





JÜLICH Forschungszentrum

SUBSPACE ITERATION

Power Iteration: Given a generic vector $v = \sum_{i=1}^{n} s_i x_i$

$$v^m = A^m v = \sum_{i=1}^{n} s_i A^m x_i = \sum_{i=1}^{n} s_i \lambda_i^m x_i = s_1 x_1 + \sum_{i=2}^{n} s_i \left(\frac{\lambda_i}{\lambda_1}\right)^m x_i \sim \boxed{s_1 x_1}$$

Subspace iteration $+$ Chebyshev polynomials:

$$\begin{aligned} v^m = p_m(A)v &= \sum_{i=1}^{n} s_i p_m(A) x_i = \sum_{i=1}^{n} s_i p_m(\lambda_i) x_i \\ &\approx \sum_{i=1}^{\text{nev}} s_i C_m\left(\frac{\lambda_i - c}{e}\right) x_i + \sum_{j=\text{nev}+1}^{n} s_j x_j \\ \text{Reorthogonalization} &+ \text{Rayleigh} - \text{Ritz} \\ &\approx \sum_{i=1}^{\text{nev}} \left(s_i x_i + \sum_{j=\text{nev}+1}^{n} S_j^i \frac{1}{|\rho_j|^m} x_j \right) \\ &\sim \boxed{\sum_{i=1}^{\text{nev}} s_i x_i} \end{aligned}$$

THE CORE OF THE ALGORITHM: CHEBYSHEV FILTER

In practice

Three-terms recurrence relation

$$C_{m+1}(t) = 2xC_m(t) - C_{m-1}(t); \qquad m \in \mathbb{N}, \quad C_0(t) = 1, \quad C_1(t) = t$$

$$Z_m \doteq p_m(\tilde{A})\, Z_0 \quad \text{with} \quad \tilde{A} = A - cI_N \quad \text{and} \quad c = \frac{b_{\text{sup}} + \mu_{\text{nev}}}{2} \qquad e = \frac{b_{\text{sup}} - \mu_{\text{nev}}}{2}$$

FOR: $i = 1 \rightarrow \text{deg} - 1$

$$Z_{i+1} \leftarrow 2\frac{\sigma_{i+1}}{e}\, \tilde{A} \times Z_i - \sigma_{i+1}\sigma_i\, Z_{i-1}$$

`xHEMM`

END FOR.

OUTLINE

Chebyshev Accelerated Subspace Iteration Eigensolver (ChASE): the main idea

ChASE: parameter selection

Some numerical results

CHASE PARAMETERS

A typical iterative methods depends on a number of tunable parameters. ChASE is no exception.

General input parameters

- N – Size of eigenproblem
- nev – Number of desired eigenpairs
- nex – Size of search space augmentation
- tol – Required threshold tolerance

Filter parameters

- deg – Polynomial degree
- μ_1 – Estimate for lowest eigenvalue λ_1
- b_{sup} – Bound for largest eigenvalue λ_n
- $\mu_{\text{nev+nex}}$ – Estimate for eigenvalue bounding search space

ESTIMATING THE SPECTRAL PARAMETERS

Estimate of μ_1 and b_{sup} are obtained by the simple repeatition of few Lanczos steps[1]

1. Compute k Lanczos steps

$$AU = UT_k + f_k e_k^\top \qquad T_k = Z^H \tilde{\Lambda}_k Z \qquad \tilde{\Lambda}_k = \text{diag}[\tilde{\lambda}_1, \ldots, \tilde{\lambda}_k]$$

2. Compute upper bound

$$b_{\text{sup}} = \|f_k\|_2 + \max[\tilde{\lambda}_1, \ldots, \tilde{\lambda}_k]$$

3. Estimate lower eigenvalue

$$\mu_1 = \min[\tilde{\lambda}_1, \ldots, \tilde{\lambda}_k]$$

$k \sim 25$ is usually sufficient

[1] Based on work by Zhou and Li (2011)

ESTIMATING THE SPECTRAL PARAMETERS

Estimating $\mu_{\text{nev+nex}}$ and requires additional Laczos steps to build a spectral density[2].

1. Compute n_{vec} times k Lanczos steps

$$AU^{[j]} = U^{[j]}T_k^{[j]} + f_k^{[j]}e_k^\top \qquad T_k^{[j]} = (Z^{[j]})^H \tilde{\Lambda}_k^{[j]} Z^{[j]} \qquad \tilde{\Lambda}_k^{[j]} = \text{diag}[\tilde{\lambda}_1^{[j]}, \ldots, \tilde{\lambda}_k^{[j]}]$$

2. Compute the spectral density

$$\tilde{\phi}(t) = \frac{1}{n_{\text{vec}}} \sum_{j=1}^{n_{\text{vec}}} \sum_{i=0}^{k} |Z_{1,i}^{[j]}|^2 g_\sigma(t - \tilde{\lambda}_i^{[j]})$$

3. Find $\bar{t} = \mu_{\text{nev+nex}}$ such that

$$\int_{-\infty}^{\bar{t}} \tilde{\phi}(t)dt \approx \frac{\text{nev+nex}}{N}$$

.

Width of the Gaussian $\sigma = 0.25 * |b_{\text{sup}} - \mu_1|$
Number of random vectors $n_{\text{vec}} = 3 \div 5$

[2]Based on work by Lin et al. (2016)

POLYNOMIAL DEGREE OPTIMIZATION

Residuals vs convergence ratio

Definition

The **convergence ratio** for the eigenvector x_a corresponding to eigenvalue $\lambda_a \notin [\mu_{\text{nev+nex}}, b_{\text{sup}}]$ is defined as

$$|\rho_a|^{-1} = \min_{\pm} \left| \frac{\lambda_a - c}{e} \pm \sqrt{\left(\frac{\lambda_a - c}{e}\right)^2 - 1} \right|.$$

The further away λ_a is from the interval $[\mu_{\text{nev+nex}}, b_{\text{sup}}]$ the smaller is $|\rho_a|^{-1}$ and the faster the convergence to x_a is.

Residuals are a function of m and $|\rho|$

$$\text{Res}(w_a^m, \lambda_a) \quad = \text{Const} \times \left| \frac{1}{\rho_a} \right|^m \qquad 1 \leq a \leq \text{ nev+nex}.$$

"Const" is independent of m and ρ

JÜLICH Forschungszentrum

POLYNOMIAL DEGREE OPTIMIZATION

Tailoring polynomial degree for each eigenpair

1. Filter with an initial degree $m^{(0)}$ and compute residuals

$$\mathrm{Res}(w_a^{m_a^{(0)}}, \tilde{\lambda}_a) \sim \mathrm{Const} \times |\rho_a|^{-m_a^{(0)}} \qquad 1 \leq a \leq \text{ nev+nex.}$$

2. The residual of eigenpair after the next filtering step **would be**

$$\mathrm{Res}(w_a^{m_a^{(1)}}, \tilde{\lambda}_a) \approx \mathrm{Res}(w_a^{m_a^{(0)}}) \times |\rho_a|^{-m_a^{(1)}} \qquad 1 \leq a \leq \text{ nev+nex.}$$

3. Compute the optimal minimal degree such that $\mathrm{Res}(w_a^{m_a^{(0)}}, \tilde{\lambda}_a) \leq \text{tol}$

$$m_a^{(1)} \geq \ln \left| \frac{\mathrm{Res}(w_a^{m_a^{(0)}}, \tilde{\lambda}_a)}{\text{tol}} \right| (\ln|\rho_a|)^{-1} \qquad 1 \leq a \leq \text{ nev+nex.}$$

CHASE PSEUDOCODE (OPTIMIZED)

INPUT: Hermitian A, tol, deg — OPTIONAL: approx. vectors W, approx. values $\{\mu_1 \dots \mu_{\text{nev}}\}$.
OUTPUT: nev wanted eigenpairs (Λ, Y).

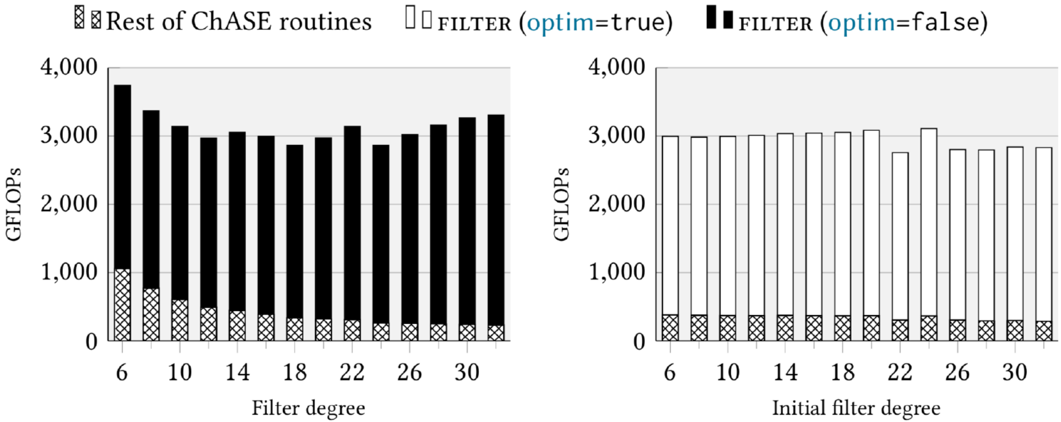
1. Lanczos DoS step. Computes **spectral estimates** μ_1, $\mu_{\text{nev+nex}}$, and $b_{\text{sup}} > \lambda_N$.
2. Set the initial vector of degrees $m_a =$deg

REPEAT UNTIL CONVERGENCE:

3. Chebyshev filter. **Filter** a block of vectors W with a vector of degrees m_a.
4. Re-orthogonalize $W = QR$ & compute the Rayleigh quotient $G = Q^\dagger AQ$.
5. Solve the reduced problem $GZ = Z\tilde{\Lambda}$ and compute the approximate Ritz pairs $(\tilde{\Lambda}, W \leftarrow QZ)$.
6. Compute and store Ritz vectors residuals $\text{Res}(w_a, \tilde{\lambda}_a)$ and check for convergence.
7. Deflate and lock the converged vectors in (Λ, Y).
8. Optimizer. **Compute vector** of polynomial degrees $m_a \geq \ln \left| \frac{\text{Res}(w_a, \tilde{\lambda}_a)}{\text{tol}} \right| / \ln |\rho_a|$.

END REPEAT

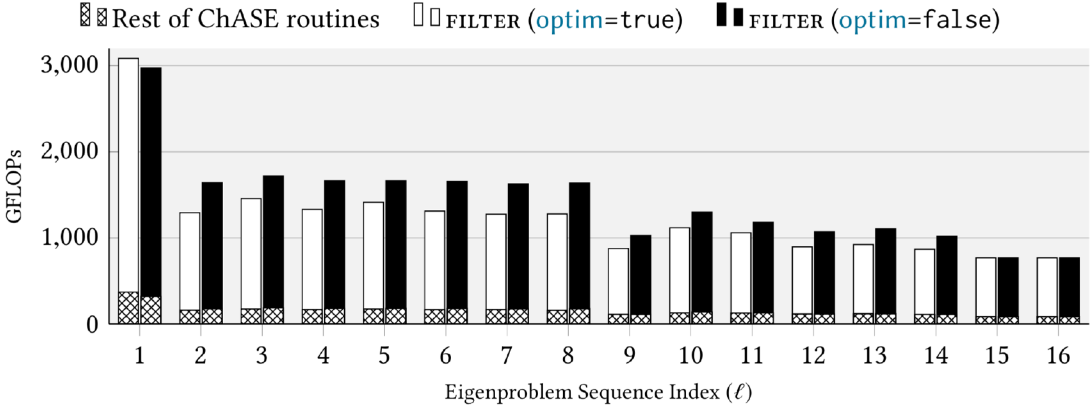
INDEPENDENCE FROM INITIAL DEGREE



(a) *Non* degree-optimized ChASE

(b) Degree-optimized ChASE

DEGREE OPTIMIZATION ⇒ FLOPS REDUCTION



JÜLICH Forschungszentrum

CHASE PARAMETERS

In ChASE, filter parameters have been practically eliminated.

General input parameters

- N – Size of eigenproblem
- nev – Number of desired eigenpairs
- nex – Size of search space augmentation
- tol – Required threshold tolerance

Filter parameters

- ~~deg – Polynomial degree~~
- ~~μ_1 – Estimate for lowest eigenvalue~~
- ~~b_{sup} – Bound for largest eigenvalue~~
- ~~$\mu_{\text{nev+nex}}$ – Estimate for eigenvalue bounding search space~~

OUTLINE

Chebyshev Accelerated Subspace Iteration Eigensolver (ChASE): the main idea

ChASE: parameter selection

Some numerical results

JÜLICH Forschungszentrum

CHASE LIBRARY RELEASED

ChASE is open source (BSD 2.0 license) and available at

`https://github.com/SimLabQuantumMaterials/ChASE`

`https://arxiv.org/abs/1805.10121` (To appear on ACM Transaction on Mathematical Software)

Highlights

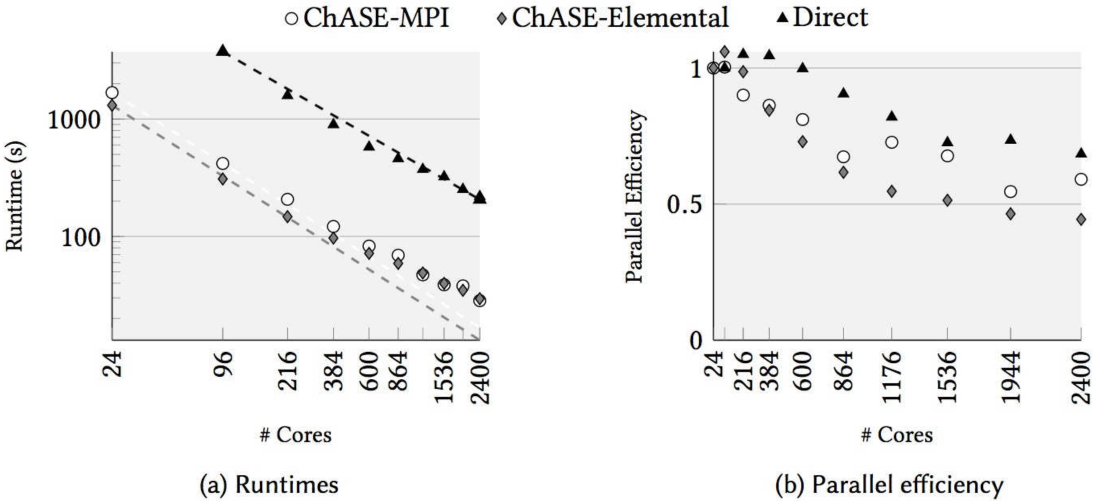
- Modern C++ interface: easy-to-integrate in application codes.
- Multiple parallel implementations: performance portability.
- Excellent strong- and weak-scale performance.

EXPERIMENTAL SETUP

Table: Matrices used in scaling experiments

	n	nev	nex	# Nodes	JURECA # Cores	JURECA $\frac{n^2}{\#\ \text{Cores}}$	BLUEWATERS # Cores	BLUEWATERS $\frac{n^2}{\#\ \text{Cores}}$
NaCl	3893	256	51	4	96	N/A	N/A	N/A
	9273	256	51	25	600	N/A	N/A	N/A
AuAg	13,379	972	194	25	600	N/A	N/A	N/A
BSE	22,360	100	20	9	216	2,314,674	64	7,812,025
	32,976	100	20	16	384	2,831,814	128	8,495,442
	47,349	100	20	36	864	2,594,823	288	7,784,471
	62,681	100	20	64	1536	2,557,882	512	7,673,648
	76,674	100	20	100	2400	2,449,542	800	7,348,628

STRONG SCALING



WEAK SCALING ON JURECA

Table: Weak scaling experimental results on JURECA

# Cores	Iterations		Matvecs		Runtime		
	ChASE-BLAS	ChASE-Elemental	ChASE-BLAS	ChASE-Elemental	ChASE-BLAS	ChASE-Elemental	Direct
216	11	11	19,990	20,192	25.1 s	26.0 s	81.5 s
384	10	9	16,778	16,100	23.7 s	24.0 s	141.2 s
864	17	11	23,424	27,506	39.8 s	45.2 s	211.1 s
1536	13	12	23,268	21,940	36.4 s	41.4 s	367.8 s
2400	10	13	22,614	21,720	38.4 s	40.8 s	380.1 s

Tests were performed on the JURECA cluster.

- 2 Intel Xeon E5-2680 v3 Haswell – Up to $0.96 \div 1.92$ TFLOPS DP/SP;
- 2 x NVIDIA K80 (four devices) – Up to $2.91 \div 8.74$ TFLOPS DP/SP.

WEAK SCALING WITH GPUS (BLUE WATERS)

Table: Weak scaling experiment results for ChASE-MPI on BLUEWATERS

# Cores	Iterations	Matvecs	Filter Runtime			Total Runtime		
			CPU	GPU	Speedup	CPU	GPU	Speedup
64	11	20,106	176.4 s	22.8 s	7.7	228.0 s	43.5 s	5.2
128	9	16,856	175.9 s	27.5 s	6.4	236.1 s	52.6 s	4.5
288	12	23,610	231.5 s	30.2 s	7.7	306.8 s	70.3 s	4.4
512	14	23,080	225.5 s	30.1 s	7.5	316.5 s	87.3 s	3.6
800	12	22,868	209.1 s	30.8 s	6.8	299.2 s	89.9 s	3.3

Tests were performed on the BLUE WATERS cluster.

- AMD 6276 Interlagos – Up to 156 GFLOPS DP;
- NVIDIA K20 – Up to 1310 GFLOPS DP.

CONCLUSIONS AND OUTLOOK

Conclusions

- $\surd$ Modern library based on C++ STL with clear separation b/w algorithm and implementation;
- $\surd$ ChASE is templates for SP, DP, Real and Complex;
- $\surd$ ChASE has a pure MPI and a distributed MPI+X implemetation for inner kernels;
- $\surd$ Eliminated dependence on Chebyshev filter's parameters;
- $\surd$ Minimized filter's FLOPs count.

Outlook

- $\rightarrow$ Refine ChASE node-level parallelism $\rightarrow$ multiple GPUs;
- $\rightarrow$ Add support for sparse matrices;
- $\rightarrow$ Modify filter to extend to generalized eigenproblems;
- $\rightarrow$ Eliminate dependence on nex.

THANK YOU

https://github.com/SimLabQuantumMaterials/ChASE

https://arxiv.org/abs/1805.10121 (To appear on ACM Transaction on Mathematical Software)

e.di.napoli@fz-juelich.de

http://www.fz-juelich.de/ias/jsc/slqm

QUANTUMATERIALS

JÜLICH Forschungszentrum