Eigensolvers for Symmetric Generalized Eigenproblems in SLEPc

Jose E. Roman

D. Sistemes Informàtics i Computació
Universitat Politècnica de València, Spain

RES Engineering Seminar – 15th October 2014, Barcelona

UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA

Outline

Overview of SLEPc

Eigenproblems

Large-scale eigenvalue problems are among the most demanding calculations in scientific computing

Example application areas:

- Dynamic structural analysis (e.g., civil engineering)
- Stability analysis (e.g., control engineering)
- Eigenfunction determination (e.g., electromagnetics)
- Bifurcation analysis (e.g., fluid dynamics)
- Information retrieval (e.g., latent semantic indexing)

SLEPc: Scalable Library for Eigenvalue Problem Computations

A general library for solving large-scale sparse eigenproblems on parallel computers

- For standard and generalized eigenproblems
- For real and complex arithmetic
- For Hermitian or non-Hermitian problems
- Also support for SVD, QEP, and more

$$Ax = \lambda x \qquad Ax = \lambda Bx \qquad Av_i = \sigma_i u_i \qquad (\lambda^2 M + \lambda C + K)x = 0$$

Authors: J. E. Roman, C. Campos, E. Romero, A. Tomas

`http://www.grycap.upv.es/slepc`

Current version: 3.5 (released July 2014)

PETSc/SLEPc Numerical Components

PETSc

Nonlinear Systems		
Line Search	Trust Region	Other

Time Steppers			
Euler	Backward Euler	Pseudo Time Step	Other

Krylov Subspace Methods							
GMRES	CG	CGS	Bi-CGStab	TFQMR	Richardson	Chebychev	Other

Preconditioners						
Additive Schwarz	Block Jacobi	Jacobi	ILU	ICC	LU	Other

Matrices					
Compressed Sparse Row	Block CSR	Symmetric Block CSR	Dense	CUSP	Other

Vectors	
Standard	CUSP

Index Sets			
Indices	Block	Stride	Other

PETSc/SLEPc Numerical Components

PETSc

Nonlinear Systems		
Line Search	Trust Region	Other

Time Steppers			
Euler	Backward Euler	Pseudo Time Step	Other

Krylov Subspace Methods							
GMRES	CG	CGS	Bi-CGStab	TFQMR	Richardson	Chebychev	Other

Preconditioners						
Additive Schwarz	Block Jacobi	Jacobi	ILU	ICC	LU	Other

Matrices					
Compressed Sparse Row	Block CSR	Symmetric Block CSR	Dense	CUSP	Other

Vectors	
Standard	CUSP

Index Sets			
Indices	Block	Stride	Other

SLEPc

Polynomial E-solver		
TOAR	Q-Arnoldi	Linear-ization

Nonlinear Eigensolver			
SLP	RII	N-Arnoldi	Interp.

SVD Solver			
Cross Product	Cyclic Matrix	Lanczos	Thick R. Lanczos

M. Function
Krylov

Linear Eigensolver					
Krylov-Schur	GD	JD	RQCG	CISS	Other

Spectral Transformation			
Shift	Shift-and-invert	Cayley	Preconditioner

BV **DS** **RG** **FN**

Use Case: Neutron Difusion Equation in Nuclear Eng.

Neutron power in nuclear reactor cores

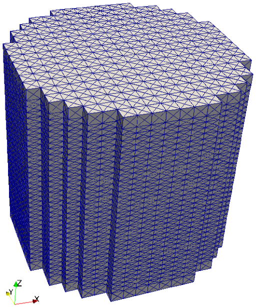
- Commercial reactors such as PWR
- Both steady state and transient
- Goal: assure safety

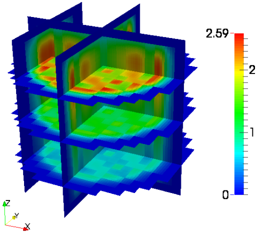
Lambda Modes Equation

$$\mathcal{L}\phi = \frac{1}{\lambda}\mathcal{M}\phi$$

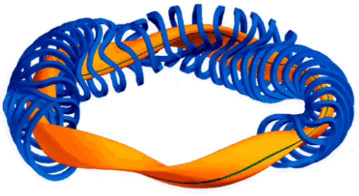
Current trends

- Complex geometries, unstructured meshes, FVM
- Coupled neutronic-thermalhydraulic calculations





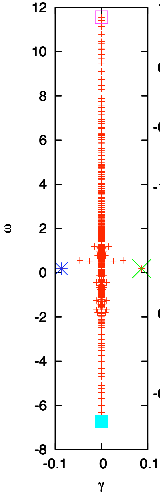
Use Case: Gyrokinetic Equations in Plasma Physics



Max-Planck-Institut für Plasmaphysik

Plasma turbulence in a tokamak determines its energy confinement

- GENE code
- Initial value solver



Knowledge of the spectrum of the linearized equation

$$Ax = \lambda x$$

- Complex, non-Hermitian, implicit A
- Sizes ranging from a few millions to a billion
- Estimate optimal timestep (largest eigenvalue); track sub-dominant instabilities (rightmost evals)

The SLEPc Project

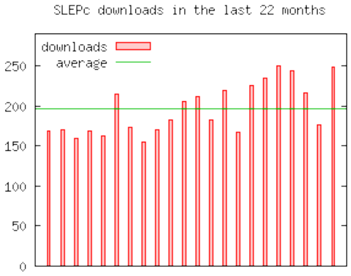
Project started around 2000; 14 releases since 2002

- Software distribution with GNU LGPL license
- User support via email
- Colaborations in cases of special interest

13,000 downloads (since 2003)
$\rightsquigarrow$ 200 per month (last year)

Main distributions (Linux, MacOSX, PyPI)

Supercomputing centers: BSC, NERSC, NCSA, Jülich



Applications

Google Scholar: 284 citations of main paper (ACM TOMS 2005)

Nuclear Engineering	6 %
Computational Electromagnetics, Electronics, Photonics	9 %
Plasma Physics	11 %
Astrophysics	1 %
Computational Physics, Materials Science, Electronic Structure	20 %
Acoustics	5 %
Computational Fluid Dynamics	12 %
Earth Sciences, Oceanology, Hydrology, Geophysics	4 %
Bioengineering, Computational Neuroscience	3 %
Structural Analysis, Mechanical Engineering	5 %
Information Retrieval, Machine Learning, Graph Algorithms	7 %
Visualization, Computer Graphics, Image Processing	3 %
PDE's, Numerical Methods	10 %
Dynamical Systems, Model Reduction, Inverse Problems	4 %

Definite Problems

Linear Eigenvalue Problems

Compute a few eigenpairs (λ, x) of

Standard Eigenproblem

$$Ax = \lambda x$$

Generalized Eigenproblem

$$Ax = \lambda Bx$$

where A, B can be real or complex, symmetric (Hermitian) or not

User can specify:

- Number of eigenpairs (`nev`), subspace dimension (`ncv`)
- Selected part of spectrum
- Tolerance, maximum number of iterations
- Many solvers: Krylov-Schur, GD, JD, RQCG, CISS, ...
- Solver options: extraction type, balancing, ...
- Initial guesses, deflation space

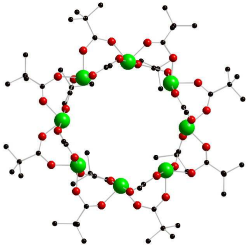
MAGPACK: Molecular Clusters

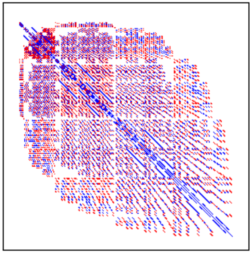
Analysis of high-nuclearity spin clusters

- Collaboration with ICMol (Valencia)
- Goal: study magnetic susceptibility and magnetization as well as inelastic neutron scattering spectra

Example: ring of 10 Ni(II) ions with antiferromagnetic interaction

- Dimension: 59,049
- Nonzero elements: ∼35 million
- Real, symmetric, indefinite
- 600 leftmost eigenvalues



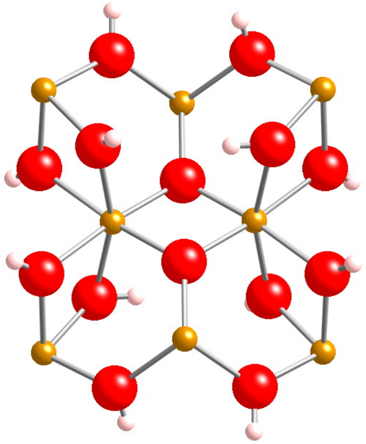


Molecular Clusters - Real Use Case: Fe8

Study: octanuclear iron, Fe8 (n=1,679,616)

Classical example of single molecule magnet, with interesting properties

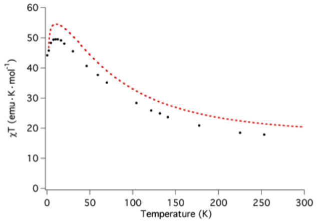
- Slow magnetic moment relaxation at low temperatures
- Strong axial anisotropy



S. Cardona-Serra, J.M. Clemente-Juan, E. Coronado, E. Ramos and J. E. Roman, "Parallel implementation of the MAGpack, the Fe8 example", in preparation.

Molecular Clusters - Results of Fe8 Simulations

Magnetic susceptibility

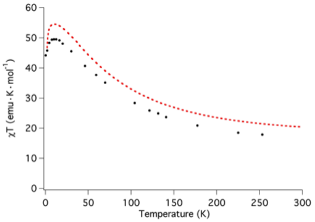


Good match against experimental data

- Results obtained with 2000 eigenvalues
- Computations in MareNostrum (during a visit of E. Ramos) with 4096 processors
- Low temperatures (<40K) computed with anisotropy (7 diagonalizations), high with isotropic analysis (21 decoupled matrices)

Molecular Clusters - Results of Fe8 Simulations

Magnetic susceptibility



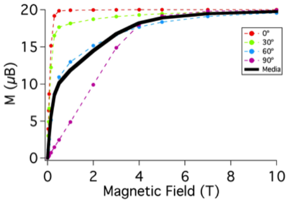
Good match against experimental data

- Results obtained with 2000 eigenvalues
- Computations in MareNostrum (during a visit of E. Ramos) with 4096 processors
- Low temperatures (<40K) computed with anisotropy (7 diagonalizations), high with isotropic analysis (21 decoupled matrices)

4 magnetic field orientations, 10 magnetic fields

- 2 diagonalizations per point (80 in total)
- Only 400 low-energy states
- Mostly computed in Tirant w/1024 procs
- About 8 hours each diagonalization

Magnetization at 2K



Spectral Transformation

The shift-and-invert transformation enables Lanczos methods to compute interior eigenvalues

$$\boxed{Ax = \lambda Bx} \quad \Longrightarrow \quad \boxed{(A - \sigma B)^{-1} B x = \theta x}$$

- Trivial mapping of eigenvalues: $\theta = (\lambda - \sigma)^{-1}$
- Eigenvectors are not modified
- Very fast convergence close to σ

Spectral Transformation

The shift-and-invert transformation enables Lanczos methods to compute interior eigenvalues

$$\boxed{Ax = \lambda Bx} \quad \Longrightarrow \quad \boxed{(A - \sigma B)^{-1} Bx = \theta x}$$

- Trivial mapping of eigenvalues: $\theta = (\lambda - \sigma)^{-1}$
- Eigenvectors are not modified
- Very fast convergence close to σ

Things to consider:

- Implicit inverse $(A - \sigma B)^{-1}$ via linear solves
- Direct linear solver for robustness
- Less effective for eigenvalues far away from σ

Spectrum Slicing

Indefinite (block-)triangular factorization:

$$A - \sigma B = LDL^T$$

By Sylvester's law of inertia, we get as a byproduct the number of eigenvalues on the left of σ

$$\nu(A - \sigma B) = \nu(D)$$

Spectrum Slicing

Indefinite (block-)triangular factorization:

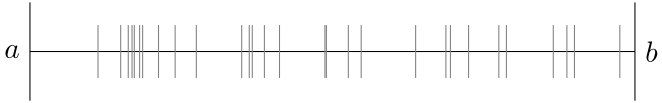
$$A - \sigma B = LDL^T$$

By Sylvester's law of inertia, we get as a byproduct the number of eigenvalues on the left of σ

$$\nu(A - \sigma B) = \nu(D)$$

Strategy:

- Multi-shift scheme that sweeps all the interval
- Compute eigenvalues by chunks
- Use inertia to validate sub-intervals



C. Campos and J. E. Roman, "Strategies for spectrum slicing based on restarted Lanczos methods", *Numer. Algorithms*, 60(2):279–295, 2012.

Spectrum Slicing

Indefinite (block-)triangular factorization:

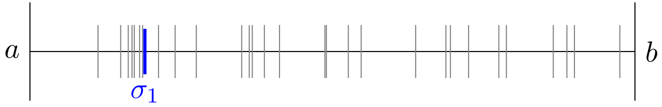
$$A - \sigma B = LDL^T$$

By Sylvester's law of inertia, we get as a byproduct the number of eigenvalues on the left of σ

$$\nu(A - \sigma B) = \nu(D)$$

Strategy:

- Multi-shift scheme that sweeps all the interval
- Compute eigenvalues by chunks
- Use inertia to validate sub-intervals



C. Campos and J. E. Roman, "Strategies for spectrum slicing based on restarted Lanczos methods", *Numer. Algorithms*, 60(2):279–295, 2012.

Spectrum Slicing

Indefinite (block-)triangular factorization:

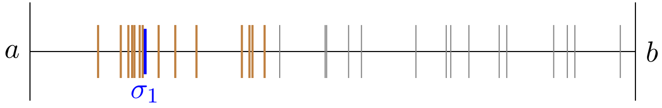
$$A - \sigma B = LDL^T$$

By Sylvester's law of inertia, we get as a byproduct the number of eigenvalues on the left of σ

$$\nu(A - \sigma B) = \nu(D)$$

Strategy:

- Multi-shift scheme that sweeps all the interval
- Compute eigenvalues by chunks
- Use inertia to validate sub-intervals



C. Campos and J. E. Roman, "Strategies for spectrum slicing based on restarted Lanczos methods", *Numer. Algorithms*, 60(2):279–295, 2012.

Spectrum Slicing

Indefinite (block-)triangular factorization:

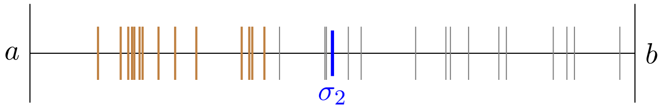
$$A - \sigma B = LDL^T$$

By Sylvester's law of inertia, we get as a byproduct the number of eigenvalues on the left of σ

$$\nu(A - \sigma B) = \nu(D)$$

Strategy:

- Multi-shift scheme that sweeps all the interval
- Compute eigenvalues by chunks
- Use inertia to validate sub-intervals



C. Campos and J. E. Roman, "Strategies for spectrum slicing based on restarted Lanczos methods", *Numer. Algorithms*, 60(2):279–295, 2012.

Spectrum Slicing

Indefinite (block-)triangular factorization:

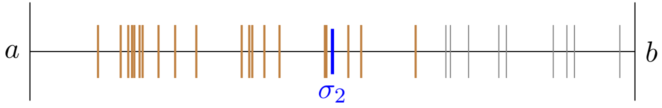
$$A - \sigma B = LDL^T$$

By Sylvester's law of inertia, we get as a byproduct the number of eigenvalues on the left of σ

$$\nu(A - \sigma B) = \nu(D)$$

Strategy:

- Multi-shift scheme that sweeps all the interval
- Compute eigenvalues by chunks
- Use inertia to validate sub-intervals



C. Campos and J. E. Roman, "Strategies for spectrum slicing based on restarted Lanczos methods", *Numer. Algorithms*, 60(2):279–295, 2012.

Spectrum Slicing

Indefinite (block-)triangular factorization:

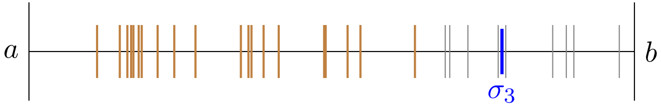
$$A - \sigma B = LDL^T$$

By Sylvester's law of inertia, we get as a byproduct the number of eigenvalues on the left of σ

$$\nu(A - \sigma B) = \nu(D)$$

Strategy:

- Multi-shift scheme that sweeps all the interval
- Compute eigenvalues by chunks
- Use inertia to validate sub-intervals



C. Campos and J. E. Roman, "Strategies for spectrum slicing based on restarted Lanczos methods", *Numer. Algorithms*, 60(2):279–295, 2012.

Spectrum Slicing

Indefinite (block-)triangular factorization:

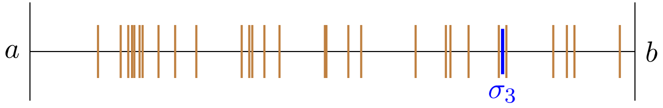
$$A - \sigma B = LDL^T$$

By Sylvester's law of inertia, we get as a byproduct the number of eigenvalues on the left of σ

$$\nu(A - \sigma B) = \nu(D)$$

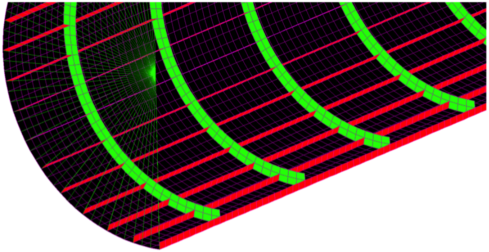
Strategy:

- Multi-shift scheme that sweeps all the interval
- Compute eigenvalues by chunks
- Use inertia to validate sub-intervals



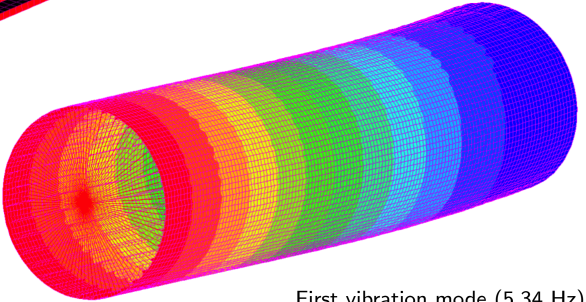
C. Campos and J. E. Roman, "Strategies for spectrum slicing based on restarted Lanczos methods", *Numer. Algorithms*, 60(2):279–295, 2012.

Test Case: Aircraft Fuselage



Simplified but realistic model: cylinder with skin, frames, and stringers

Parametric, “scalable”



First vibration mode (5.34 Hz)

SAMTECH

Test Case: Matrix Properties

Analysis of frequency range [0–60] Hz

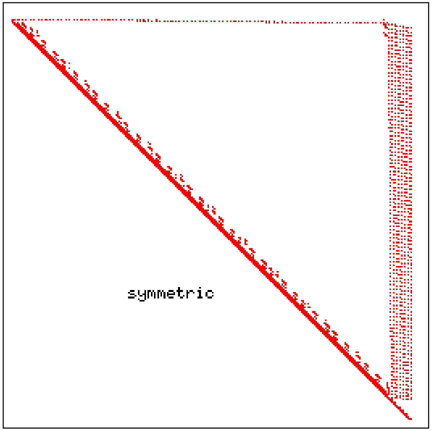
1 million dof's

- Dimension: 1,036,698
- Nonzeros: $\sim$29 million
- Eigenvalues in interval: 1989

2 million dof's

- Dimension: 2,141,646
- Nonzeros: $\sim$59 million
- Eigenvalues in interval: 2039

Maximum multiplicity: 2



B is singular

Evaluation: Parallel Performance

1 million, nev=120

p	Shft	Rest	Its	Time	Num.	Sym.	Tri.	Orth.
8	12	21	4,318	16,399	3,153	594	5,270	6,218
16	13	27	4,565	10,125	1,739	595	4,277	3,364
32	10	21	4,101	5,500	519	653	2,839	1,428
64	12	22	4,338	4,064	375	654	2,482	580
128	12	20	4,155	3,394	273	596	2,265	245

2 million, nev=120

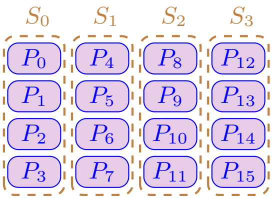
p	Shft	Rest	Its	Time	Num.	Sym.	Tri.	Orth.
32	13	24	4,873	14,429	2,012	1,691	6,536	3,902
64	12	22	4,472	9,120	922	1,697	4,886	1,263
128	12	23	4,501	7,817	707	1,692	4,709	668

CaesarAugusta (256 JS20 nodes, 2 PowerPC 970+, 4 GB/node, Myrinet)

Multi-Communicator Spectrum Slicing

p processes, ℓ partitions with p/ℓ processes each

- Each group processes a subinterval, with a parallel linear solver
- Goal: improve scalability
- Needs more memory: store A, B redundantly



Computational work will likely differ in each subinterval

- A priori knowledge of eigenvalue distribution can be used to improve load balance
- User can set subinterval boundaries

Indefinite Problems

Quadratic Eigenvalue Problem

QEP:
$$(\lambda^2 M + \lambda C + K)x = 0$$

where M, C, K are real symmetric (or complex Hermitian)

Solution strategies:

1. Symmetric linearization with pseudo-Lanczos [Parlett,Chen1990]

$$\begin{bmatrix} 0 & -K \\ -K & -C \end{bmatrix} - \lambda \begin{bmatrix} -K & 0 \\ 0 & M \end{bmatrix}$$

2. Disregard symmetry, use Arnoldi solver
   - Memory-efficient variant: Q-Arnoldi [Meerbergen 2008]
   - Stabilized variant: TOAR [Su et al. 2008]
3. Combine both

C.Campos and J. E. Roman, “Restarted Q-Arnoldi-type methods exploiting symmetry in quadratic eigenvalue problems”, submitted.

Pseudo-Lanczos

Adapt B-Lanczos to the case of indefinite $\langle \cdot, \cdot \rangle_B$

```
for $j = 1, 2, \ldots, k$
    $u = Sv_j$
    $t_{1:j,j} = V_j^* Bu$
    $u = u - V_j \Omega_j^{-1} t_{1:j,j}$
    $t_{j+1,j} = \sqrt{|u^* Bu|}$
    $\omega_{j+1} = \mathrm{sign}(u^* Bu)$
    $v_{j+1} = u\omega_{j+1} / t_{j+1,j}$
end
```

$S := B^{-1} A$

Orthogonalization:

- Full B-orthogonalization (twice)
- Must insert Ω_j^{-1} in the projector used in Gram-Schmidt

Computes Lanczos relation $SV_k = V_k \Omega_k^{-1} T_k + t_{k+1,k} \omega_{k+1}^{-1} v_{k+1} e_k^*$

- Lanczos vectors V_k satisfy $V_k^* B V_k = \Omega_k$

Quadratic Arnoldi

Consider the linearization of the QEP

$$A - \lambda B = \left[\begin{array}{cc} 0 & N \\ -K & -C \end{array} \right] - \lambda \left[\begin{array}{cc} N & 0 \\ 0 & M \end{array} \right]$$

with either $N = I$ or $N = -K$. The eigenvector is $[x^*, \lambda x^*]^*$

Quadratic Arnoldi

Consider the linearization of the QEP

$$A - \lambda B = \begin{bmatrix} 0 & N \\ -K & -C \end{bmatrix} - \lambda \begin{bmatrix} N & 0 \\ 0 & M \end{bmatrix}$$

with either $N = I$ or $N = -K$. The eigenvector is $[x^*, \lambda x^*]^*$

Q-Arnoldi implicitly computes an Arnoldi relation for $S = B^{-1}A$

$$\begin{bmatrix} 0 & I \\ -M^{-1}K & -M^{-1}C \end{bmatrix} \begin{bmatrix} V_k^0 \\ V_k^1 \end{bmatrix} = \begin{bmatrix} V_k^0 & v^0 \\ V_k^1 & v^1 \end{bmatrix} \underline{H}_k$$

From the 1st row block: $$V_k^1 = [V_k^0, v^0]\underline{H}_k$$

Hence, only $\{V_k^0, \underline{H}_k, v\}$ needs to be stored

Q-Arnoldi / Q-Lanczos

For each $j = 1, \dots, k$

1. Expand: $w^0 = v^1, \quad w^1 = -M^{-1}(Kv^0 + Cv^1)$
2. Gram-Schmidt coefficients
   $t = (V^1_{j-1})^* w^1 = \underline{H}^*_{j-1} \left[V^0_{j-1} \; v^0 \right]^* w^1$
   $h_j = \begin{bmatrix} V^0_{j-1} & v^0 \\ V^1_{j-1} & v^1 \end{bmatrix}^* w = \begin{bmatrix} (V^0_{j-1})^* w^0 + t \\ v^{0*} w^0 + v^{1*} w^1 \end{bmatrix}$
3. Gram-Schmidt update
   $\tilde{w}^0 = w^0 - \left[V^0_{j-1} \quad v^0 \right] h_j$
   $\tilde{w}^1 = w^1 - \left[\left[V^0_{j-1} \quad v^0 \right] \underline{H}_{j-1} \quad v^1 \right] h_j$
4. Normalize
   $h_{j+1,j} = \sqrt{\|\tilde{w}^0\|^2 + \|\tilde{w}^1\|^2}$
   $v^0 = \tilde{w}^0 / h_{j+1,j}, \quad v^1 = \tilde{w}^1 / h_{j+1,j}$

Q-Arnoldi / Q-Lanczos

For each $j = 1, \ldots, k$

1. Expand: $w^0 = v^1, \quad w^1 = -M^{-1}(Kv^0 + Cv^1)$
2. Gram-Schmidt coefficients
$$t = (V_{j-1}^1)^* M w^1 = \underline{H}_{j-1}^* \left[V_{j-1}^0 \;\; v^0 \right]^* M w^1$$
$$h_j = \begin{bmatrix} V_{j-1}^0 & v^0 \\ V_{j-1}^1 & v^1 \end{bmatrix}^* B w = \begin{bmatrix} -(V_{j-1}^0)^* K w^0 + t \\ -v^{0*} K w^0 + v^{1*} M w^1 \end{bmatrix}, \quad h_j = \Omega_{j+1}^{-1} h_j$$
3. Gram-Schmidt update
$$\tilde{w}^0 = w^0 - \left[V_{j-1}^0 \quad v^0 \right] h_j$$
$$\tilde{w}^1 = w^1 - \left[\left[V_{j-1}^0 \quad v^0 \right] \underline{H}_{j-1} \quad v^1 \right] h_j$$
4. Normalize
$$h_{j+1,j} = \|\tilde{w}\|_B, \quad \omega_{j+1,j+1} = \mathrm{sgn}(\|\tilde{w}\|_B)$$
$$v^0 = \tilde{w}^0 / h_{j+1,j}, \quad v^1 = \tilde{w}^1 / h_{j+1,j}$$

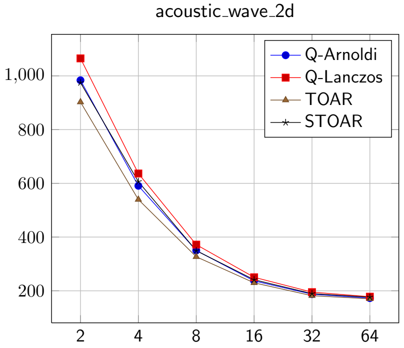
Some Experiments with SLEPc Implementation

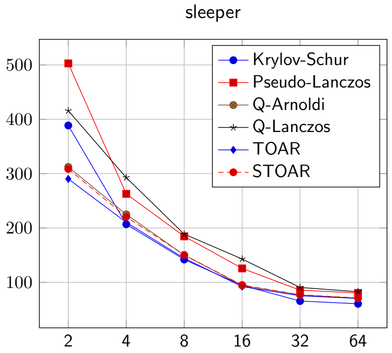
Maximum residual error for tolerance 10^{-9}, 20 eigenpairs with maximum basis size $k = 40$ (times for 64 processes)

name	method	nconv	its	$\max_i\{\|r_i\|\}$	time
acoustic_wave_2d	Q-Arnoldi	66	3	2.0e-10	172.8
$n = 6,247,500$	Q-Lanczos	66	2	1.8e-06	178.0
$\sigma = 0$	TOAR	66	3	1.4e-10	170.3
not scaled	STOAR	60	3	8.6e-10	176.3
	Krylov-Schur	40	2	1.8e-12	60.5
sleeper	Pseudo-Lanczos	37	2	5.2e-15	80.4
$n = 5,000,000$	Q-Arnoldi	41	2	7.0e-09	70.8
$\sigma = -0.9$	Q-Lanczos	46	2	0.017	82.6
not scaled	TOAR	41	2	4.2e-07	70.3
	STOAR	37	2	9.4e-04	71.2

Parallel Performance

Parallel computing times on MareNostrum III from Barcelona Supercomputing Center





Wrap Up

SLEPc is continually growing

- Increasing number of users, application areas
- New releases bring new features and optimizations

MareNostrum and friends very valuable for scalability analysis

- RES users can benefit immediately from new developments

Latest additions and things to come soon

- Nonlinear eigenproblems: polynomial, rational, general
- Contour integral for selecting a region of the complex plane
- Hybrid parallelism: multicore, GPU