Designing a nearly optimal eigensolver for symmetric problems

Andreas Stathopoulos

Computer Science Department and Computational Sciences Cluster
College of William and Mary

Premises and goals

Symmetry

↙ ↘

$Ax = b$
Optimal 3-term recurrence (CG)
Preconditioning (PCG)

$Ax = \lambda x$
All vectors required (Lanczos)
Preconditioning (JD, INVIT)

Goal: Use PCG to derive nearly optimal eigenmethods with smaller bases

Premises and goals

Symmetry

↙ ↘

$Ax = b$	$Ax = \lambda x$
Optimal 3-term recurrence (CG)	All vectors required (Lanczos)
Preconditioning (PCG)	Preconditioning (JD, INVIT)

Goal: Use PCG to derive nearly optimal eigenmethods with smaller bases

Near optimality: similar to a 3-term recurrence (PCG) for solving $(A - \lambda I)x = 0$

Two general approaches

Inner-outer iterations: (Jacobi-Davidson, RQI, INVIT, EIGIFP)

Exploit 3-term recurrences in the inner iteration
When to stop inner iteration?

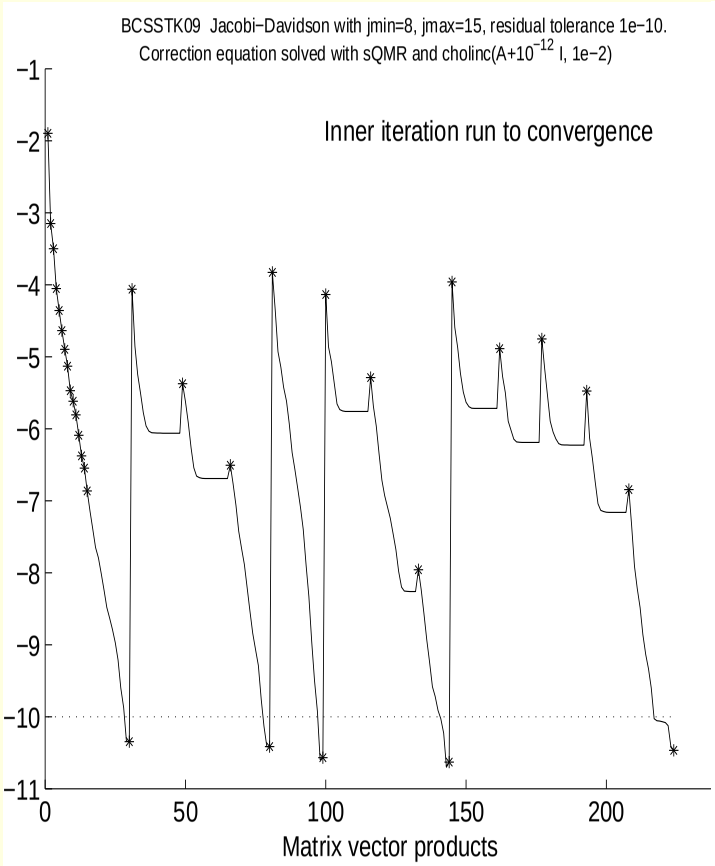
Restarting techniques: (JD with TR+p, LOBPCG, DACG)

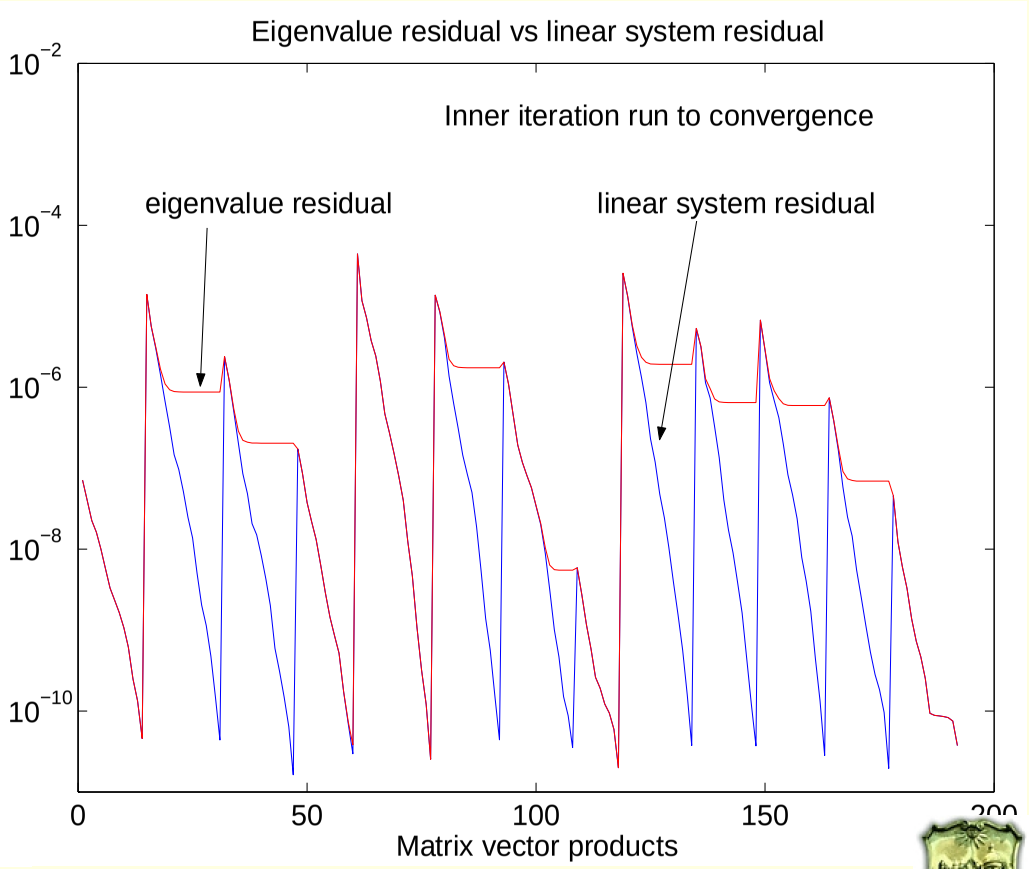
Maintain a small basis by keeping certain locally optimal directions
+ Can local optimality yield global optimality?

Stopping the inner JD iteration

$$(I - xx^T)(A - \eta I)(I - xx^T)\delta = r$$

exact λ unknown $\Rightarrow$ Full convergence of inner iteration wasteful





JDcg method [Notay, 2002]

At any iteration k of PCG on the correction equation

$$\text{Corrected Ritz vector: } y_k = x + \delta_k, \quad \theta_k = \frac{y_k^T A y_k}{y_k^T y_k}, \quad f = 1 + \|\delta_k\|^2$$

$$\text{Eigenvalue system residual: } r_k = A y_k - \theta_k y_k$$

$$\text{Linear system residual: } g_k = r - (I - xx^T)(A - \eta I)(I - xx^T)\delta_k$$

Based on quantities from PCG recurrence, Notay observed:

$$\|r_k\|^2 = \underbrace{\|g_k\|^2/f}_{\downarrow\ 0} + \underbrace{\|\delta_k\|^2(\theta_k - \eta + \beta_k)^2}_{\downarrow\ \text{Inverse iteration } r}$$

Stop when g_k starts to reduce at a faster rate then r_k

JDqmr: an extension based on sQMR ([Freund, Nachtigal, 1994])

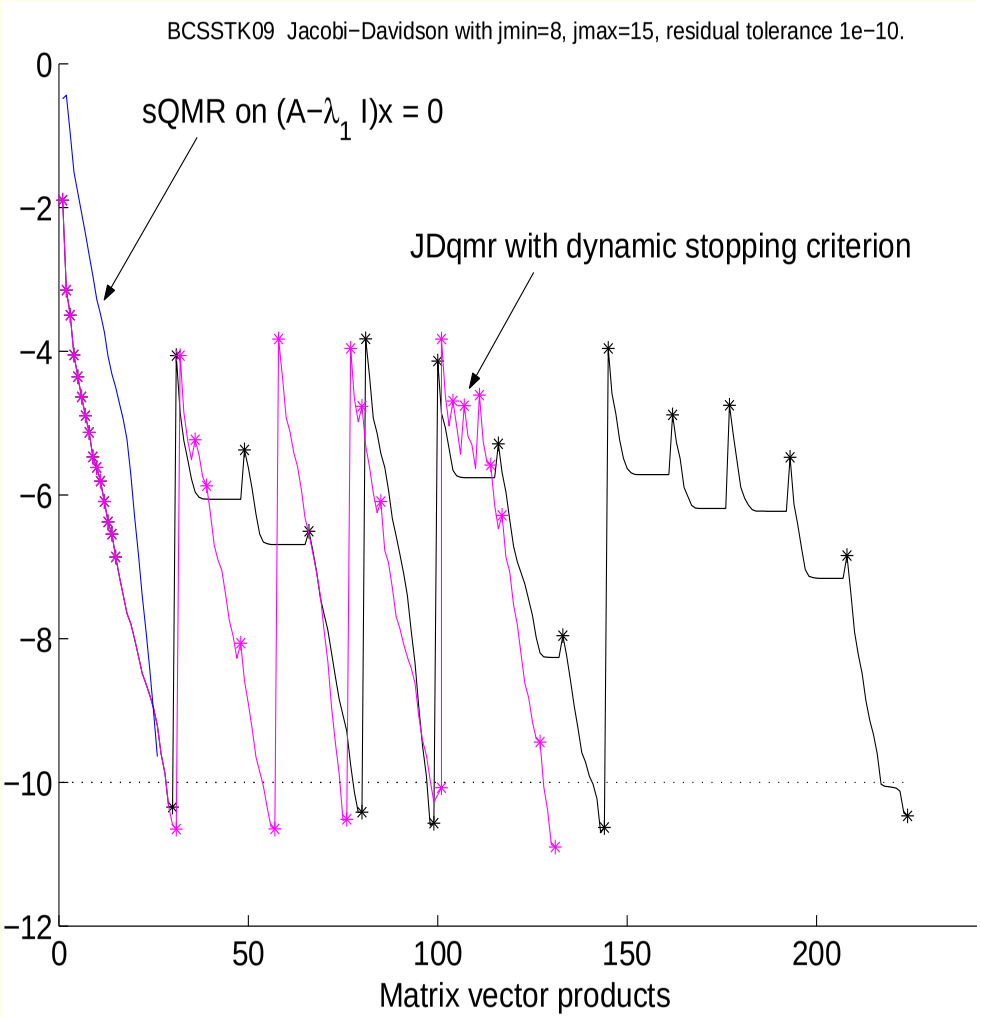
JDcg problems

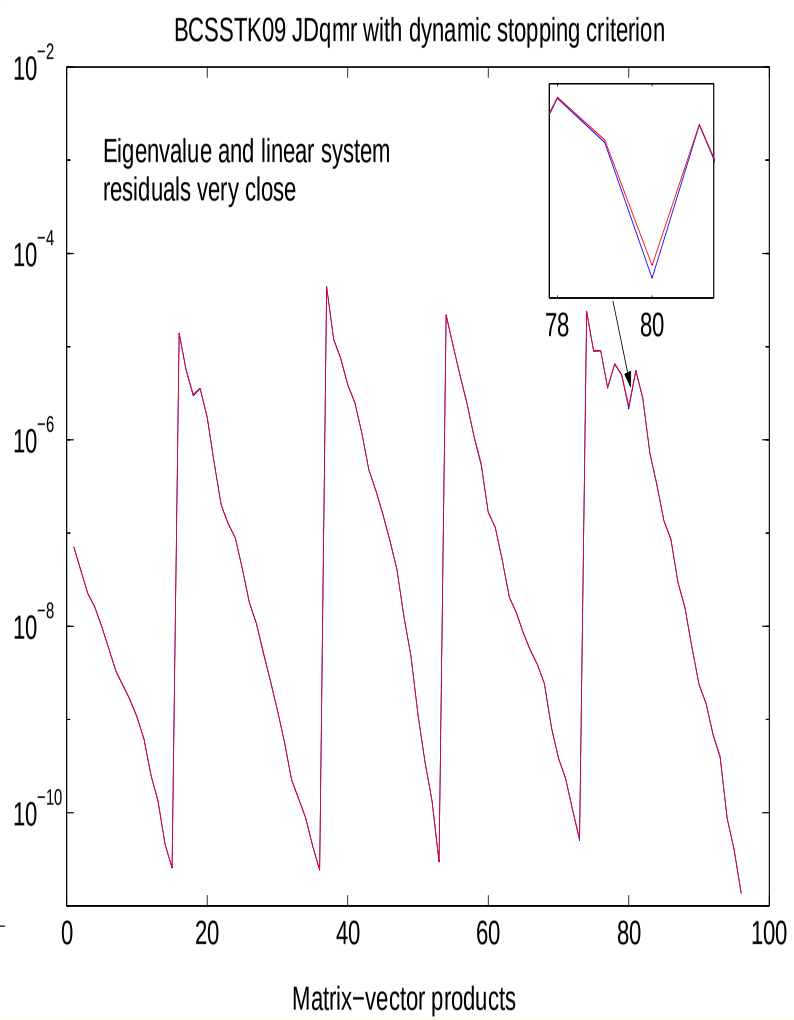
1. PCG requires SPD preconditioners
2. Not for interior eigenvalues
3. Residual convergence not smooth
4. Parameter dependent termination

JDqmr fixes

1. Can use indefinite preconditioners
2. Works for indefinite systems
3. Residual convergence smooth
4. Better stopping criteria

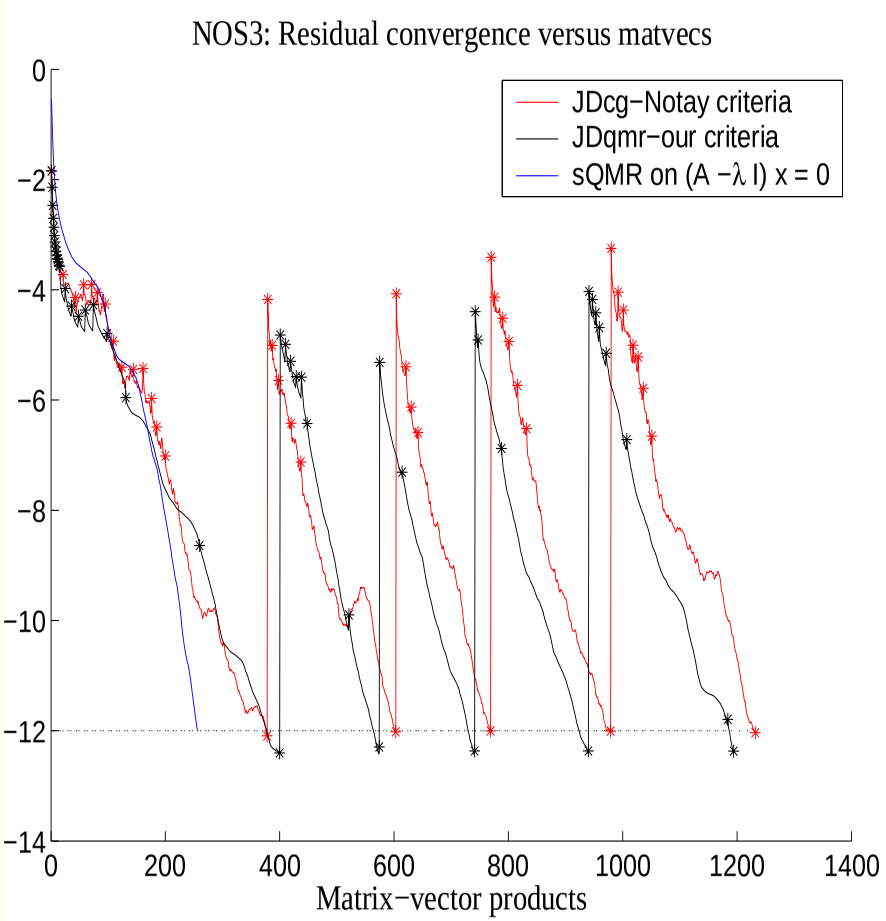
JDqmr reduces wasted iterations

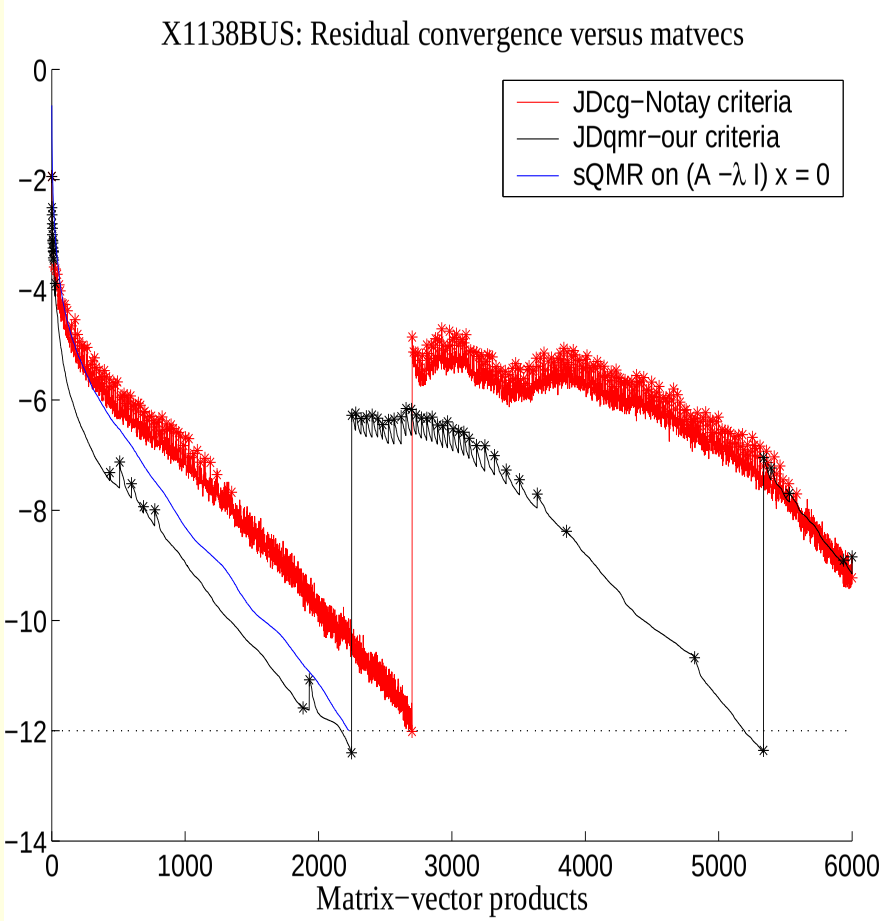




JDqmr performance versus JDcg and versus optimal sQMR

Harwell-Boeing matrices with no preconditioning: Near optimality

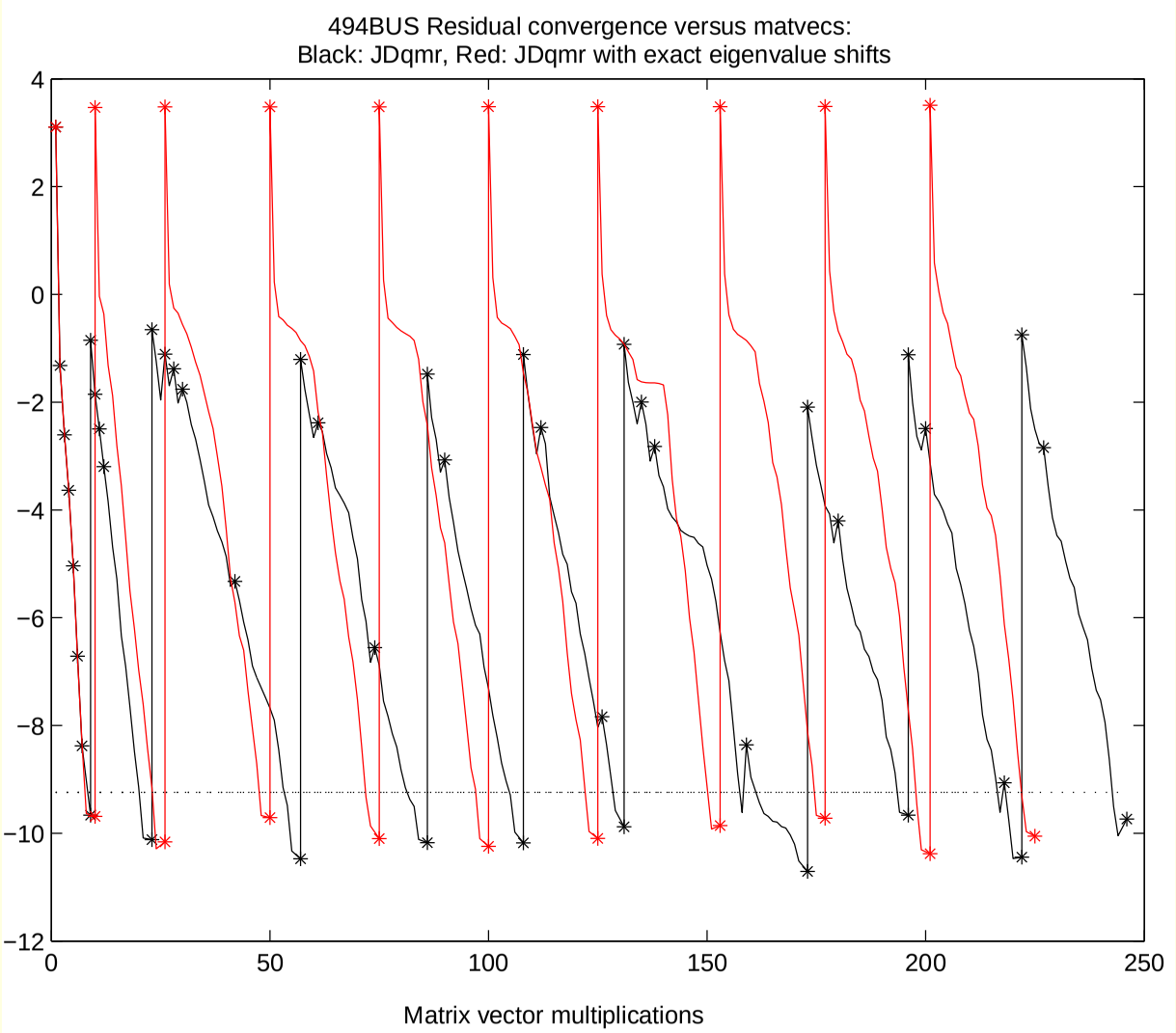




Near optimality. Yet, a linear system per eigenvalue?

Providing exact eigenvalues as shifts similar to letting the method find them

But QMR builds the space over and over for each eiganvalue.



Near optimality through restarting (?)

Restarting impairs convergence

Thick Restart JD(k,m) improves convergence with k. [Stathopoulos et al., 1998]
Yet, TR(m/2,m) better than TR(m-2,m)!

Unlike CG, no 3-term recurrence for eigenvectors

CG often used as a minimization method for the Rayleigh Quotient

[Bradbury & Fletcher, 1966]
[Bergamaschi et al., 1997]
[Pfrommer et al., 1999]
[Edelman et al.,1999]

Yet, a local optimality possible through the span of the 3-term CG vectors

Local optimality approaches

$$x_1^{i+1} = \text{Rayleigh\_Ritz}\left(x_1^{i-1},\ x_1^{i},\ M^{-1}(Ax_1^{i} - \lambda_1^{i} x_1^{i}) \right)$$

Local optimality approaches

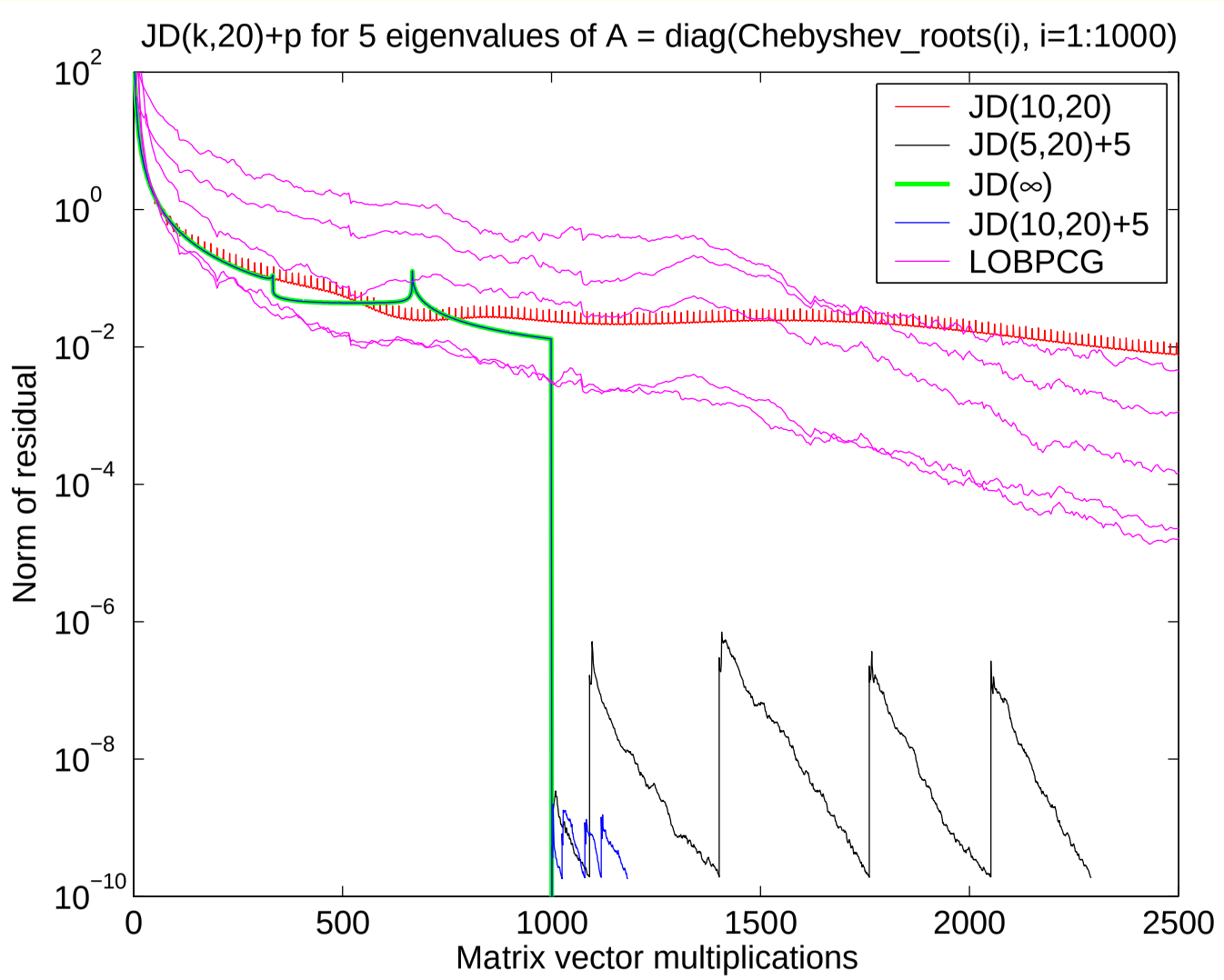
$$x_1^{i+1} = \text{Rayleigh\_Ritz}\left(x_1^{i-1},\ x_1^i,\ M^{-1}(Ax_1^i - \lambda_1^i x_1^i) \right)$$

1. Use it as a single vector method [D'yakonov 1983, Knyazev, 1991]
2. Restart a Davidson basis with $[x_1^{i-1},\ x_1^i]$ [Murray et al., 1992]
3. JD(k,m)+p: Restart with $[x_1^{i-1}, \ldots, x_p^{i-1},\ x_1^i, \ldots, x_k^i]$ [Stathopoulos 1999, 1998]
4. LOBPCG: $X^{i+1} = \text{RR}([X^{i-1}, X^i, M^{-1}R^i])$ [Knyazev, 2001]

All methods achieve near global optimality — constants differ!

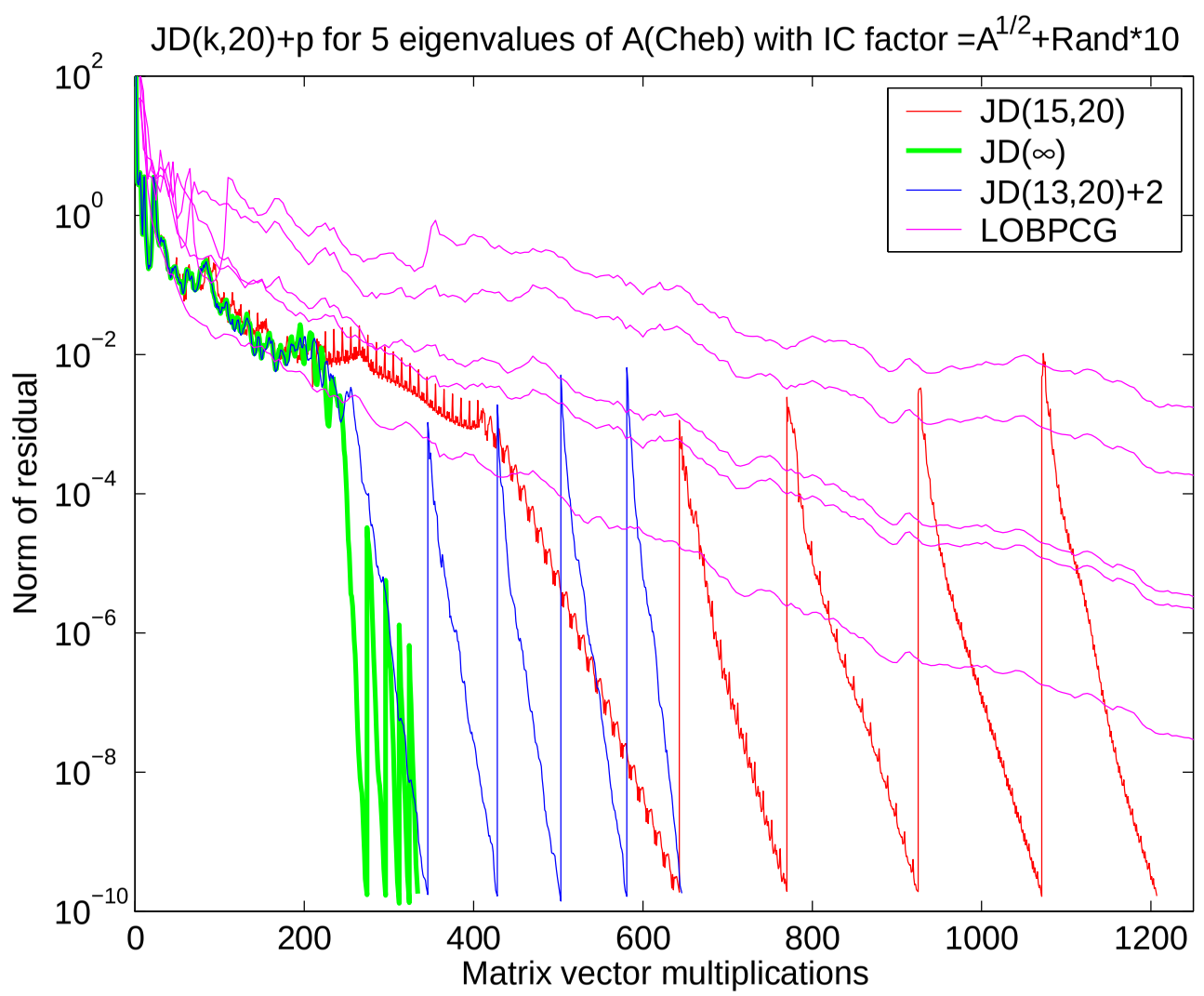
Thick Restart together with +p work miracles — (a) No preconditioning

Diagonal matrix with packed eigenvalues at both ends of spectrum



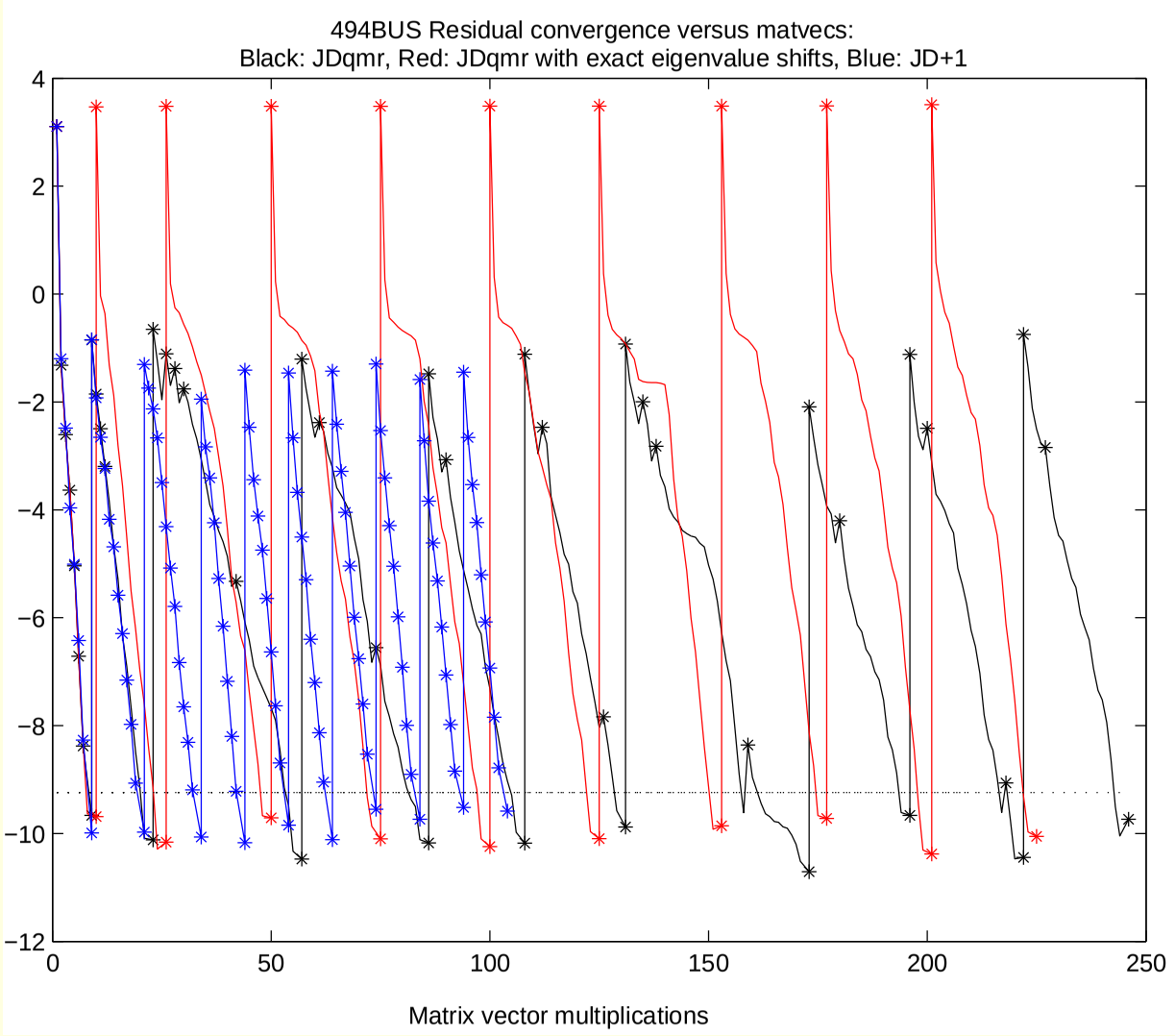
Thick Restart together with +p work miracles — (b) Preconditioned

$+p$ equally effective for small values of p — Acceleration important too!



For many eigenpairs

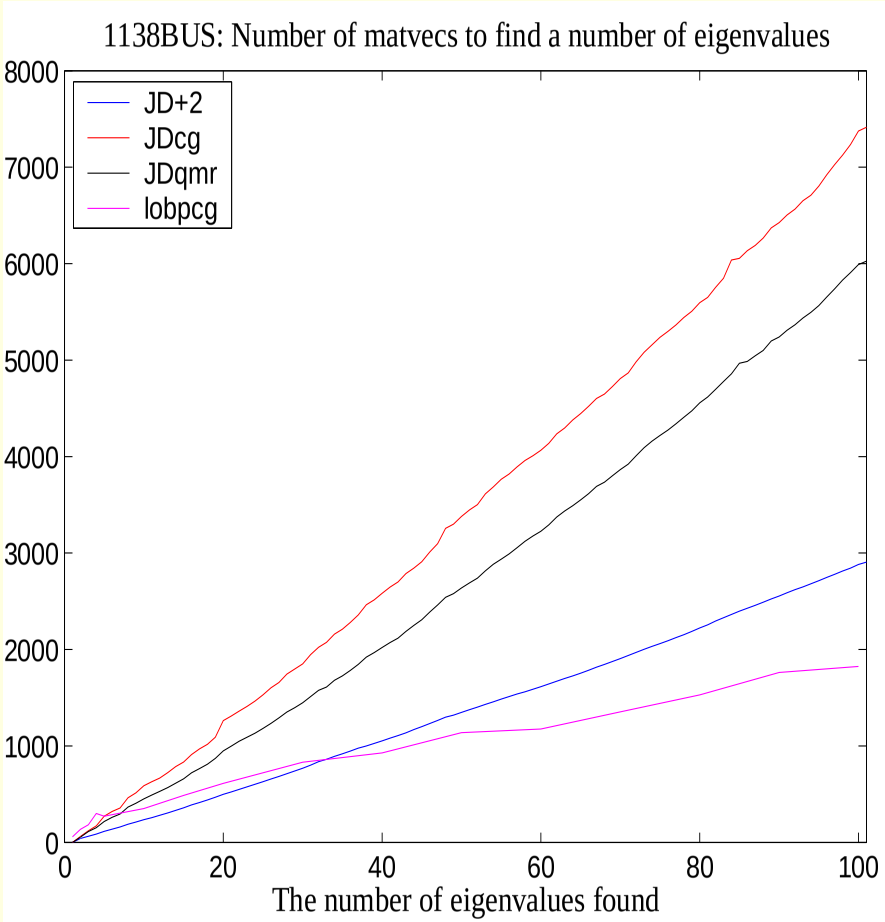
Lack of inner iteration accumulates better info for other pairs

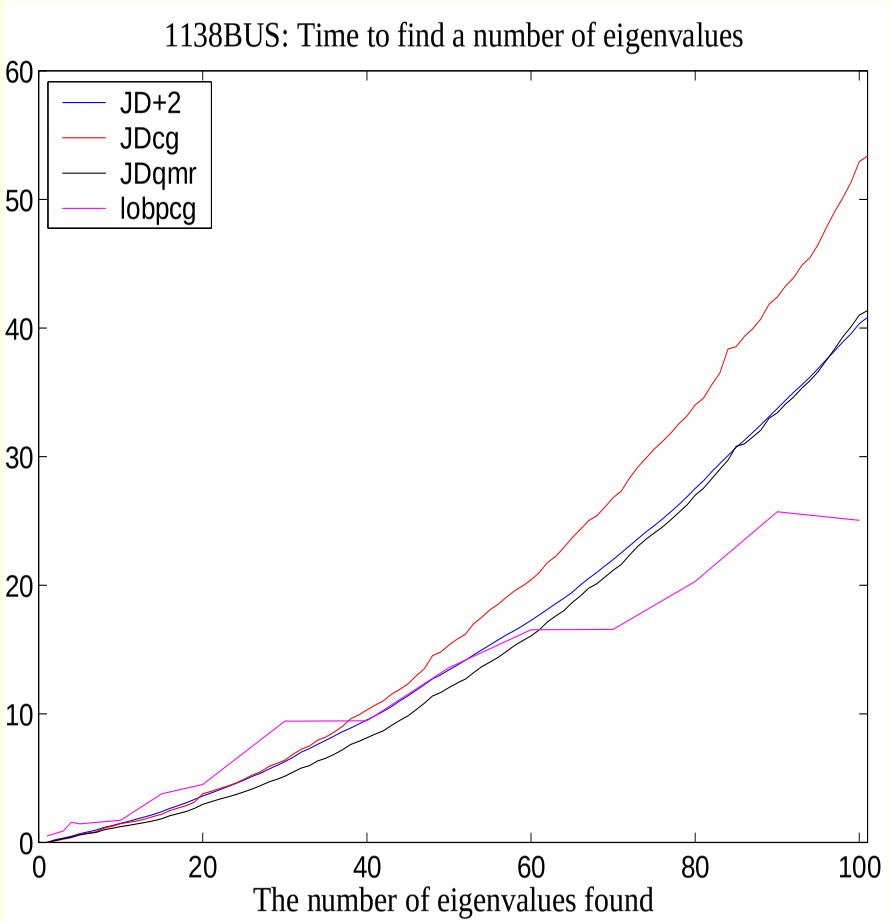


Scalability for large number of eigenpairs: 1138BUS

JD+2 winner in matvecs
despite constant basis size of 25

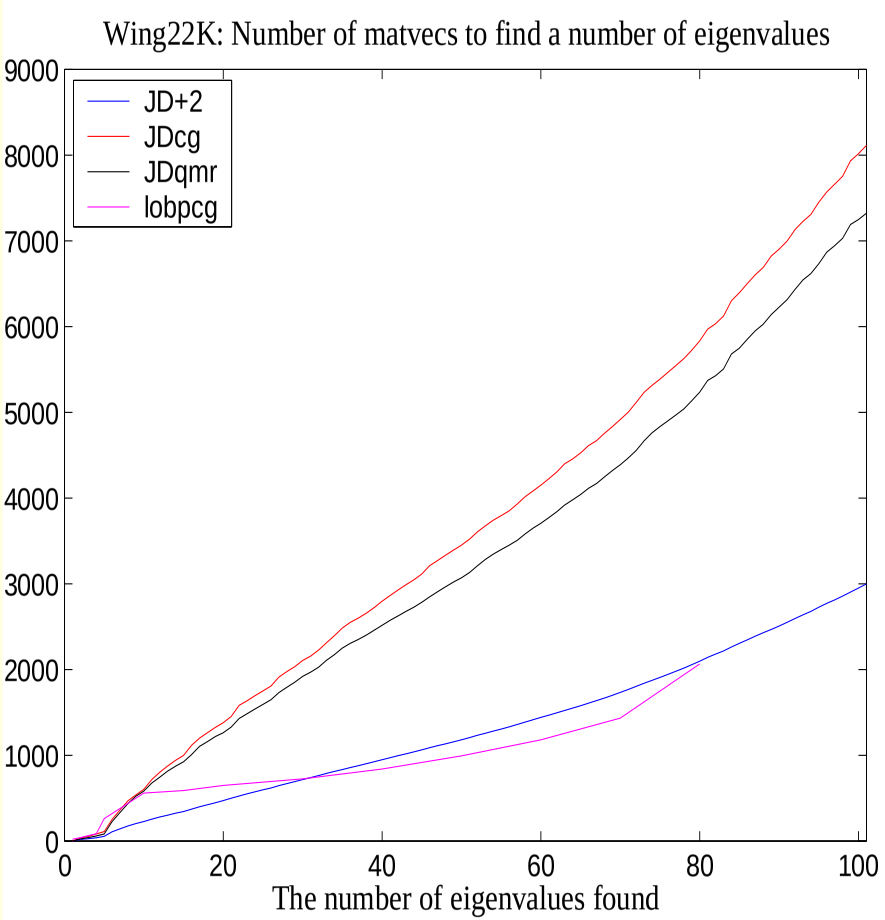
Blocking is cache efficient
LOBPCG basis size numEval

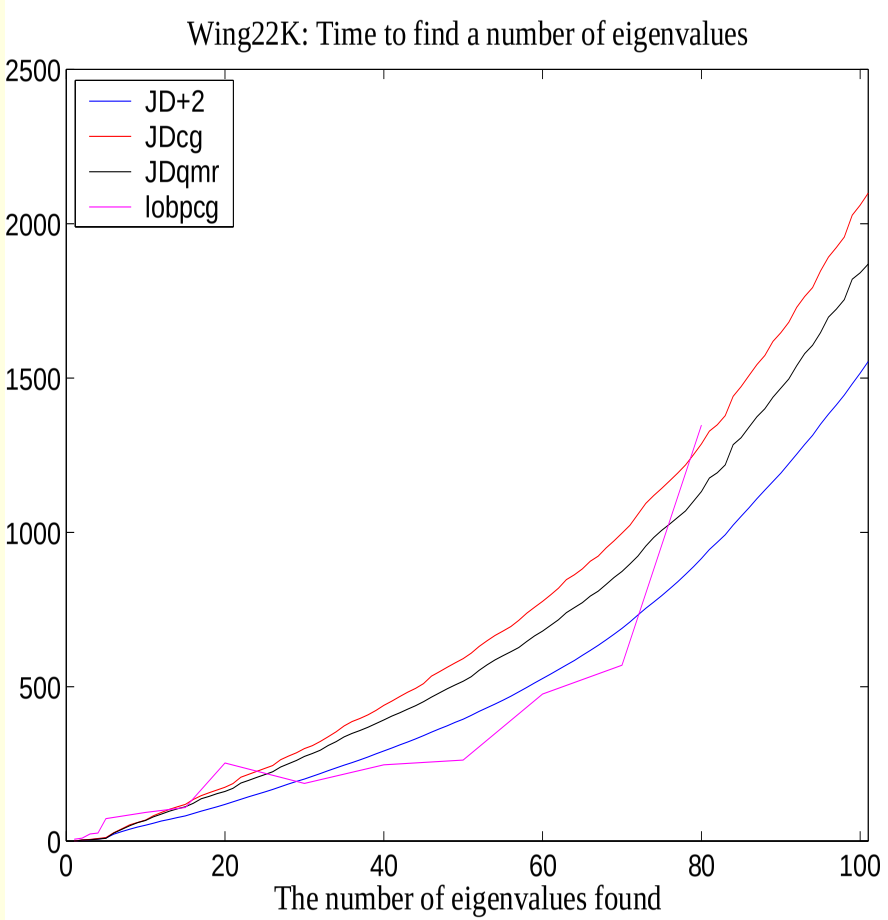




Scalability for large number of eigenpairs: Wing22K

JD+2 more stable and predictable LOBPCG stagnated for more than 90 pairs





JD+2 without preconditioning competitive to Lanczos

Five smallest eigenvalues of NASASRB

Matrix vector products

Method:	TRLAN		PARPACK	LOBPCG	JDqmr	JD+2
Basis size:	∞	1000	1000	15	30	30
1 eval:	11600		-	167400	80713	66167
5 eval:	-	50467	-	308728	257937	146248

Time:

Method	Time (sec)	Hardware	Software
TRLAN	8546	8 processors of T3E900	F90
LOBPCG	53773	1 Xeon processor, 3 GHz	MATLAB
JDqmr	42410	1 Xeon processor, 3 GHz	MATLAB
JD+2	11448	1 Xeon processor, 3 GHz	C

Interior eigenvalues close to σ

A frequently favored approach

find min eigenvalues of $(A - \sigma I)^2$

Simple: calls only canned eigenroutines

Interior eigenvalues close to σ

A frequently favored approach

find min eigenvalues of $(A - \sigma I)^2$

DO NOT USE!

- Much slower than a Krylov method on A
- Restrictive on which eigenvalues it finds

  eg., it cannot find values **only** on the right of σ

Alternative methods

Use a method such as JD+k, JDqmr or TRLAN, ARPACK, etc.

- After the Rayleigh Ritz, sort the wanted eigenvalues first
- For preconditioning, use the first eigenpair (JD+1, JDqmr)
- For restarting, pick the first k of them (TRLAN, JD)

Convergence not strictly monotonic but fast!

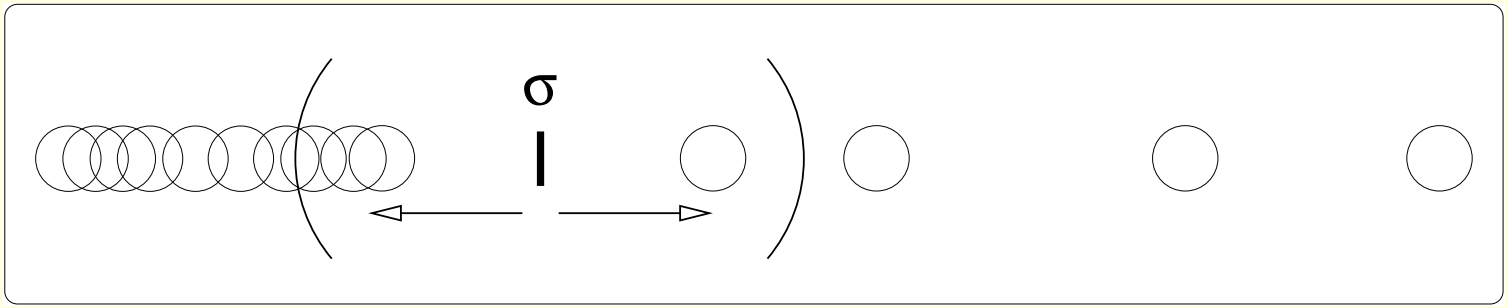
After m matvecs, works in a space of all polynomials of A of degree m

(Contrast with the space of polynomials of degree $m/2$ of $(A-\sigma I)^2$)

How to sort the Ritz values

Having obtained the Ritz values μ_i:

- if need eigenvalues larger than σ
  sort$(-(\mu_i - \sigma)^{-1})$
- if need eigenvalues smaller than σ
  sort$(\ (\mu_i - \sigma)^{-1})$
- if need eigenvalues closest to σ in an absolute sense
  sort$(\ (\mu_i - \sigma)^{2})$
- if need same number of eigenvalues on both sides of σ
  Find first the left side then the right



Some results from a Nano-project at NERSC

	JD iterations	Matvecs	Time	Found
SymmetrAround-.15, JD+1	975	976	1438.6	5L 5R
AbsAroundShift-.15, JD+1	990	991	1412.9	8L 2R
AbsAroundShift-.15,JDqmr	399	1305	1651.2	8L 2R
All 50 lower, JD+1	4999	5000	8431.2	41

Software suggestions

For **one** extreme eigenpair:

- **JDqmr is cheap and near optimal**
- both JD+1 and LOBPCG near optimal but not as fast as JDqmr

For **many** extreme eigenpairs:

- **JD(k,m)+1** always the smallest number of matvecs
- **JD(k,m)+1** usually the fastest

For **interior** eigenpairs:

- JD-type approaches far better than other approaches
- harmonic Ritz values not necessary