



Gradient-Type Subspace Iteration Methods for Symmetric Eigen-Problems

Yousef Saad

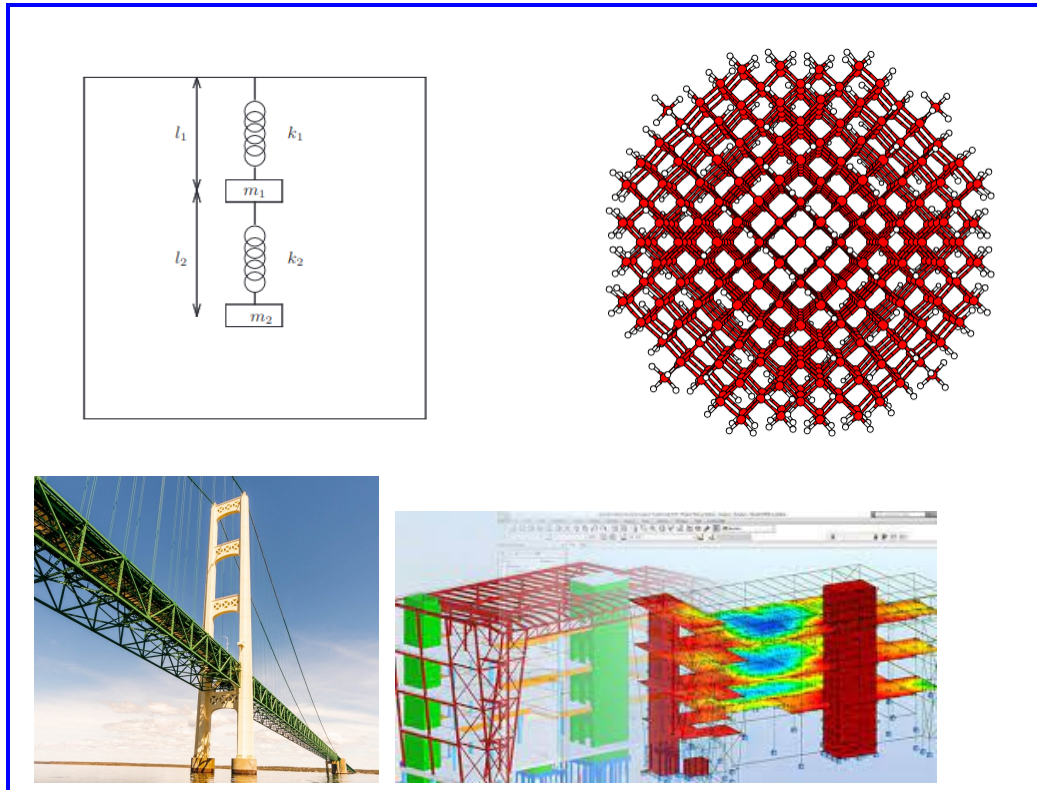
University of Minnesota

Foundations of Computational Mathematics

Paris, June 21-23, 2023

Introduction & Background

- Many applications require the computation of a few eigenvalues + associated eigenvectors of a matrix A



- Structural Engineering – (Goal: frequency response)
- Electronic structure calculations [Schrödinger equation..] – Quantum chemistry
- Stability analysis [e.g., electrical networks, mechanical system,..]
- ...

➤ What is really needed is an **invariant subspace** of some large matrix A , i.e., a **subspace** \mathcal{X} such that :

$$A\mathcal{X} \subseteq \mathcal{X} \quad \text{or} \quad AY = YC$$

Y = basis of subspace \mathcal{X} of dim m , $C \in \mathbb{R}^{m \times m}$

- Often ‘dominant’ invariant subspace needed [‘dimension reduction’]
- Smallest eigenvalues needed in, e.g., electronic structure

Problems:

- Approximate the subspace
- Update it, e.g., when data changes
- Estimate its dimension (inexpensively)
- Exploit the subspace for certain calculations [e.g., model reduction]
- Track subspace of a sequence of matrices
- Find approximate common invariant subspace to a set of matrices

Rayleigh-Ritz projection

Question: How to extract good approximations to eigenvalues/ eigenvectors from some subspace $X = \text{span}\{Q\}$ with $Q^H Q = I$

Answer: Projection method. Set approximate eigenvector $\tilde{u} = Qy$ + write:

$$Q^H (A - \tilde{\lambda}I)\tilde{u} = 0 \rightarrow Q^H A Q y = \tilde{\lambda} y$$

ALGORITHM : 1 $[X_{out}, R] = \text{Rayleigh-Ritz}(A, X)$

1. $[Q, \sim] = \text{qr}(X, 0)$ [Orthonormalize X into Q]
2. Compute $C = Q^H A Q$
3. $[Y, R] = \text{schur}(C)$ [Schur: $C = Y R Y^H$]
4. $X_{out} = QY$.

Subspace Iteration

Original idea: projection technique onto a subspace of the form $Y = A^k X$

➤ Also called just the:
“Power method”

➤ In practice: Replace A^k by suitable polynomial
[Chebyshev]

ALGORITHM : 2 $[X_{new}, D] = \text{Substl}(A, X)$

1. **Start:** Select an initial system $X = [x_1, \dots, x_m]$
and an initial polynomial C_k .
2. **Until convergence Do:**
3. Compute $\hat{X} = C_k(A)X$. *[Original: $\hat{X} = A^k X$]*
4. $[X_{new}, D] = \text{Rayleigh-Ritz}(A, \hat{X})$
5. If convergence satisfied Return.

 Else $X := X_{new}$ & select a new polynomial C'_k
6. **EndDo**

- $|\lambda_1| \geq |\lambda_2| \geq \dots \geq |\lambda_m| > |\lambda_{m+1}| \geq \dots$
- P = eigenprojector (associated with $\lambda_1, \dots, \lambda_m$)
- $\mathcal{L}_0 = \text{span}\{x_1, x_2, \dots, x_m\}$. Assume:
- $\{Px_i\}_{i=1, \dots, m}$ linearly independent.
- $\mathcal{P}_k = \perp$ projector onto $\mathcal{L}_k = \text{span}\{X_k\}$.

Assumptions:

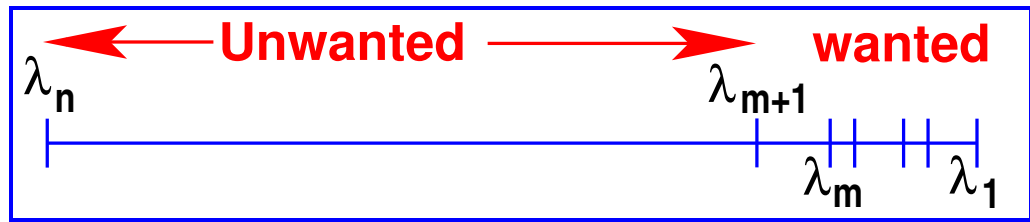
THEOREM: For each eigenvector u_i of A , $i = 1, \dots, m$, there exists a unique vector s_i in the subspace \mathcal{L}_0 such that $Ps_i = u_i$. Moreover, the following inequality is satisfied

$$\|(I - \mathcal{P}_k)u_i\|_2 \leq \|u_i - s_i\|_2 \left(\left| \frac{\lambda_{m+1}}{\lambda_i} \right| + \epsilon_k \right)^k,$$

where ϵ_k tends to zero as k tends to infinity.

Q: What Chebychev polynomial?

Typical scenario \rightarrow



Common thinking: shift and scale A to

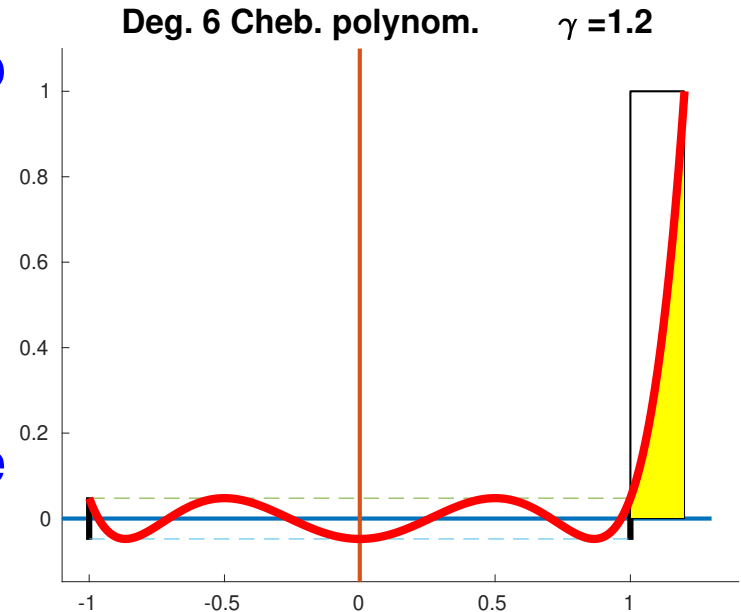
$B = (A - cI)/h$:

$$c = \frac{\lambda_{m+1} + \lambda_n}{2}, \quad h = \frac{\lambda_{m+1} - \lambda_n}{2}$$

Then: $p_k(t) = C_k(t)/C_k(\lambda_1)$

➤ Eigs of B in $[-1, 1]$ are now the 'unwanted' eigenvalues

➤ Polynomial 'optimal' in some sense for each λ_i , $i \leq m$ individually - but not for the invariant subspace as a whole.



Quick background: Krylov subspace methods

Principle: Projection methods on Krylov subspaces, i.e., on

$$K_m(A, v) = \text{span}\{v, Av, \dots, A^{m-1}v\}$$

- Arnoldi's method [$A^H \neq A$]
- Lanczos [$A^H = A$]

Krylov vs. subspace iteration

- From the perspective of computing invariant subspaces

Krylov-type methods

- + Fast
- + Optimal in a certain sense
- + Requires one starting vector
- Not easy to update
- Changes in A not allowed

Subspace iteration methods

- + Updates are easy
- + Geared toward subspaces [vs individual eigenvalues]
- + Tolerates changes in A
- Slower

Important note: both types of methods require only **matrix-vector products**. Can get superior convergence with shift-and-invert [replace A with $(A - \sigma I)^{-1}$ in Algorithms]. Issue: cost

Example: subspace iteration for Kohn-Sham equation

$$\left[-\frac{\nabla^2}{2} + V_{ion} + V_H + V_{xc} \right] \Psi(\mathbf{r}) = E \Psi(\mathbf{r})$$

With:

- Hartree potential (local)

$$\nabla^2 V_H = -4\pi\rho(\mathbf{r})$$

- V_{xc} depends on functional. For LDA:

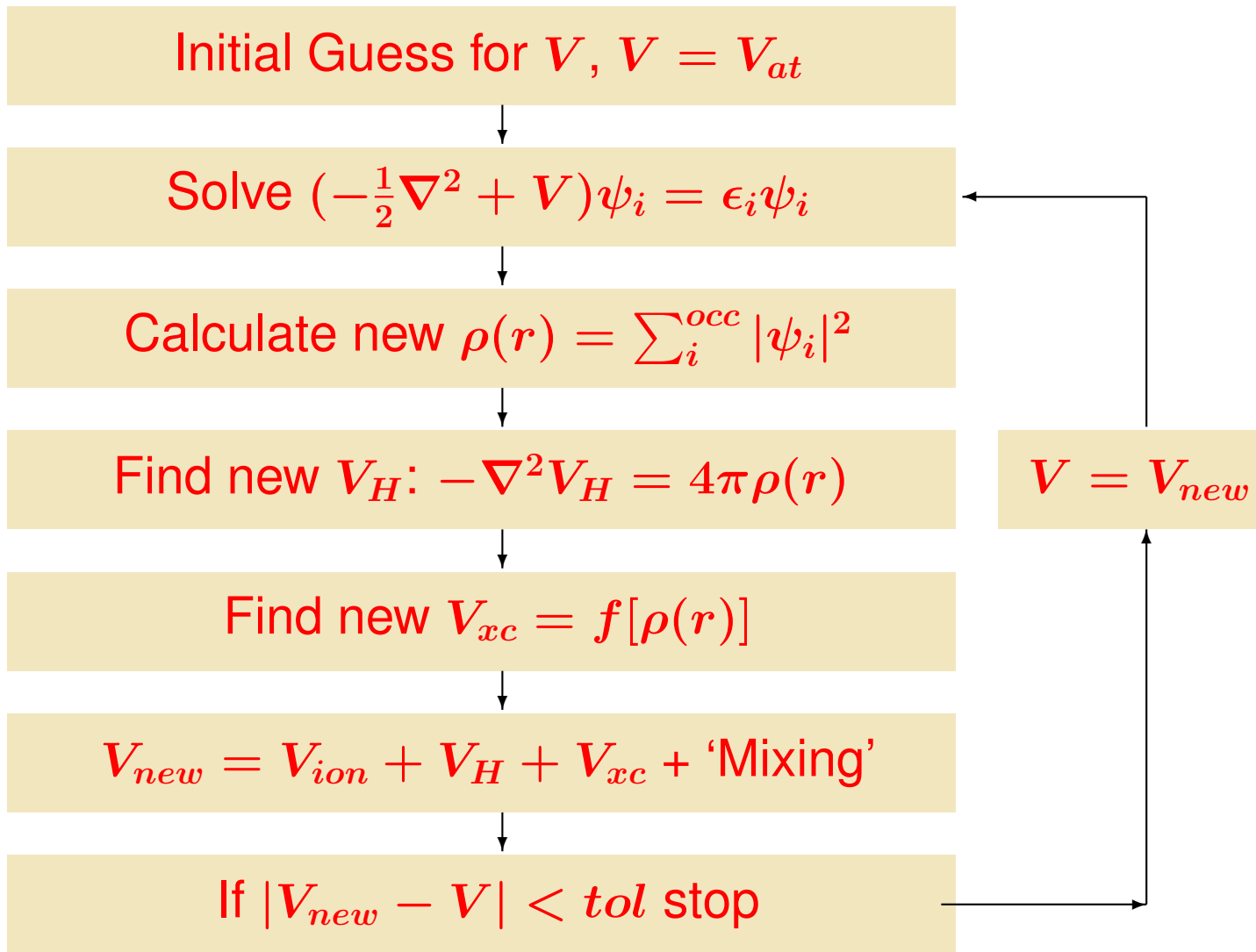
$$V_{xc} = f(\rho(\mathbf{r}))$$

- V_{ion} = nonlocal – does not explicitly depend on ρ

$$V_{ion} = V_{loc} + \sum_a P_a$$

- V_H and V_{xc} depend **nonlinearly** on eigenvectors:

$$\rho(\mathbf{r}) = \sum_{i=1}^{occup} |\psi_i(\mathbf{r})|^2$$

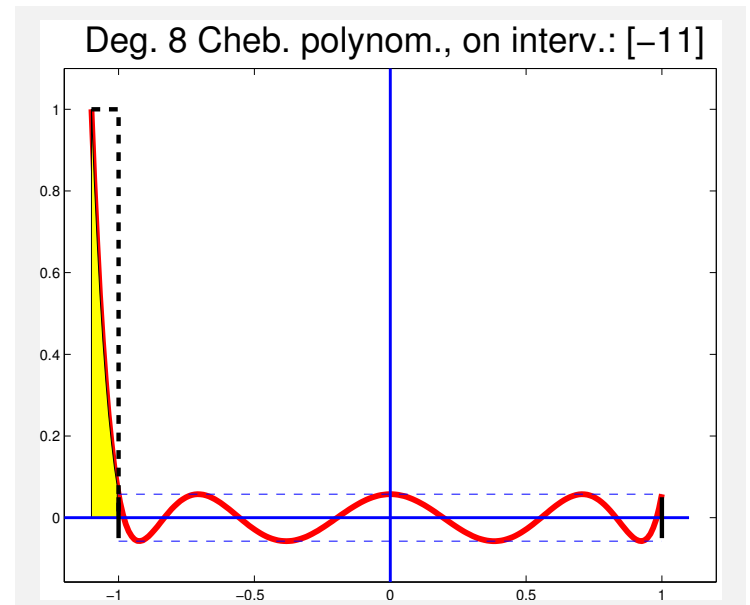


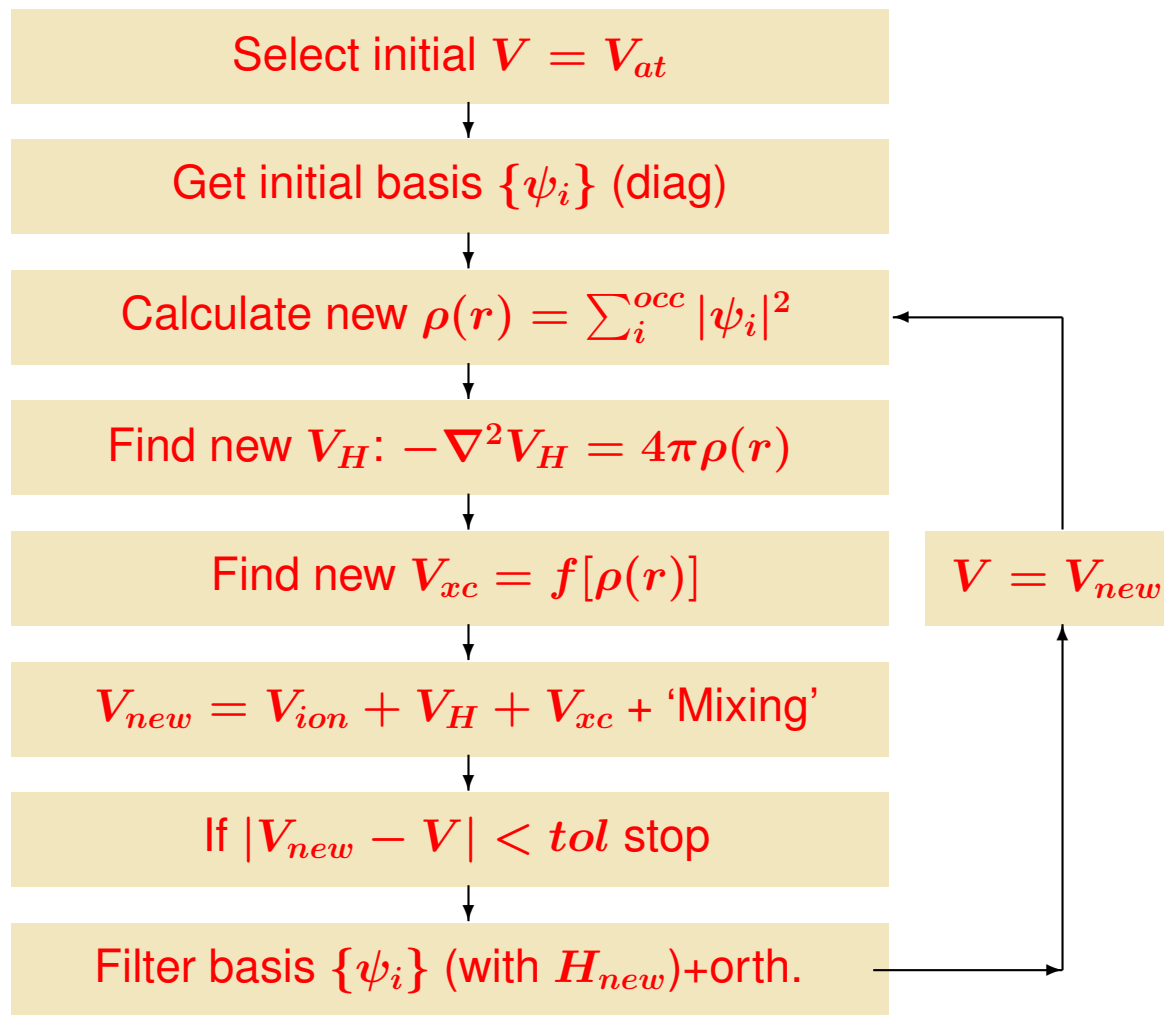
The subspace filtering viewpoint

Given a basis $[v_1, \dots, v_m]$,
'filter' each vector as

$$\hat{v}_i = P_k(A)v_i$$

- $p_k =$ Low deg. polynomial [Chebyshev]
- Filtering step not used to compute eigenvectors accurately
- SCF & diagonalization loops merged
- Another viewpoint: nonlinear form of subspace iteration





Yunkai Zhou, Y.S., Murilo L. Tiago, and James R. Chelikowsky, *Parallel Self-Consistent-Field Calculations with Chebyshev Filtered Subspace Iteration*, *Phy. Rev. E*, vol. 74, p. 066704 (2006)

$Si_{525}H_{276}$, Polynomial deg.
 == 8. Single proc.

method	# $A * x$	SCF	CPU(s.)
ChebSI	124761	11	5946.69
ARPACK	142047	10	62026.37
TRLan	145909	10	26852.84

$Si_{9041}H_{1860}$ # PEs = 48; $n_H = 2,992,832$. Degree $m = 8$

n_{state}	# $A * x$	# SCF	$\frac{total\ eV}{atom}$	1st CPU	total CPU
19015	4804488	18	-92.00412	102.12 h.	294.36 h.

The Grassmannian perspective

- Recall: Stiefel manifold ('compact' Stiefel manifold):

$$St(p, n) = \{Y \in \mathbb{R}^{n \times p} : Y^T Y = I\}.$$

- Grassmann manifold is the quotient manifold

$$G(p, n) = S(p, n) / O(p)$$

where: $O(p) \equiv$ orthogonal group of unitary $p \times p$ matrices.

- Each point on $G(p, n) \equiv$ a subspace of dimension p of \mathbb{R}^n
- Can be represented by a basis $V \in St(p, n)$.

Notation: $[V]$, [it does not matter which basis V of is used]

• A. Edelman, T. A. Arias, and S. T. Smith, *The geometry of algorithms with orthogonality constraints*, SIMAX, 20 (1999)

➤ Tangent space of the Grassmann manifold at $[Y]$ is the set of matrices $\Delta \in \mathbb{R}^{n \times p}$ s.t.:

$$Y^T \Delta = 0$$

➤ The EAS paper (above) considers minimizing

$$\phi(Y) = \frac{1}{2} \text{Tr} [Y^T A Y]$$

where $Y^T Y = I$ by a Newton approach

➤ The gradient of $\phi(Y)$ on the manifold at point $[Y]$ is

$$G = (I - Y Y^T) A Y$$

- For Newton: We need to solve $\text{Hess}\Delta = -G$ on manifold
- Notation: $\Pi = I - YY^T$, $C_Y = Y^T AY$
- Newton leads to Sylvester equation:

$$\Pi[A\Delta - \Delta C_Y] = -\Pi AY$$

- Solution: $\Delta = -Y + Z(Y^T Z)^{-1}$ where Z solves

$$AZ - ZC_Y = Y$$

A few other well-known references

1. P. -A. Absil, R. Mahony, R. Sepulchre and P. Van Dooren “A Grassmann-Rayleigh Quotient Iteration for Computing Invariant Subspaces”, *SIAM Review*, (2002)
2. P. A. Absil, R. Mahony and R. Sepulchre, *Riemannian Geometry of Grassmann Manifolds with a View on Algorithmic Computation*, *Acta Applicandae Mathematicae*, 80 (2004)
3. G. W. Stewart, “Error and perturbation bounds for subspaces associated with certain eigenvalue problems”, *SIAM Rev.*, 15 (1973)
4. J. W. Demmel, “Three methods for refining estimates of invariant subspaces”, *Computing* 38 (1987)
5. F. Chatelin, *Simultaneous Newton’s iterations for the eigenproblem*, *Proc. Oberwolfach Conference* (1984)
6. A. Sameh, J. Wisniewski, *The TraceMin algorithm*, 1982.

The Grassmannian perspective (continued)

- Problem with these 2nd-order methods: Need to solve multiple systems of equations or a Sylvester equation at each step
- Can we use Grassmannian perspective **without inversion**?
- Idea: Use a *gradient* - or *conjugate gradient* - approach

Recall: On $G(p, n)$, gradient of objective function ϕ at $[Y]$ is

$$G = \nabla \phi_Y = (I - YY^T)AY \equiv AY - YC_Y$$

with $C_Y = Y^T AY$.

Gradient approach

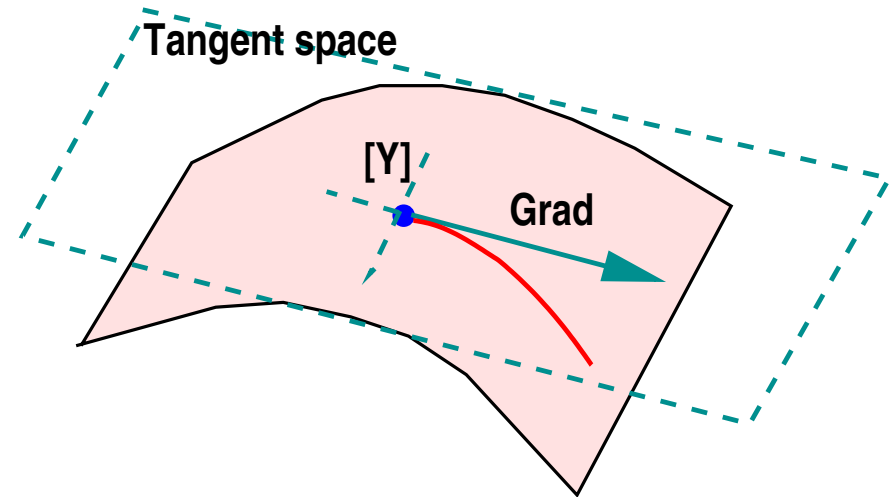
➤ Next iterate is of the following form
(μ to be determined)

$$\tilde{Y} = Y + \mu G,$$

➤ Direction of gradient will increase ϕ but iterates must stay on manifold

➤ Could follow a geodesic (EAS paper) ..

➤ Or follow a path along G but implicitly re-project each $Y + \mu G$ on manifold, i.e., consider $[Y + \mu G]$



➤ Can show

$$\phi(\tilde{Y}) = \phi(Y) + \mu \|G\|_F^2 + \frac{\mu^2}{2} \text{Tr} [AY]^T \Pi A \Pi [AY]$$

➤ ... and because $Y^T G = 0$ we have:

$$\tilde{Y}^T \tilde{Y} = [Y + \mu G]^T [Y + \mu G] = I + \mu^2 G^T G.$$

➤ Let: $G^T G = U D_\beta U^T \equiv$ spectral decomposition of $G^T G$

➤ Want: To orthonormalize \tilde{Y} without changing its span

➤ Sol: Right-multiply \tilde{Y} by $U D_\mu^{-1}$, i.e., define new Y as:

$$Y(\mu) = \tilde{Y} U D_\mu^{-1} = (Y + \mu G) U D_\mu^{-1}.$$

where:

$$D_\mu \equiv [I + \mu^2 D_\beta]^{1/2}$$

Set:

$$\begin{aligned} Y_u &= YU \\ \alpha_i &= (Y_u^T A Y_u)_{ii} \\ D_\alpha &= \text{Diag}(\alpha_i); \end{aligned}$$

$$\begin{aligned} G_u &= GU \\ \gamma_i &= (G_u^T A G_u)_{ii} \\ D_\gamma &= \text{Diag}(\gamma_i); \end{aligned}$$

Then:



$$\phi(Y(\mu)) = \frac{1}{2} \text{Tr} [I + \mu^2 D_\beta]^{-1} [D_\alpha + 2\mu D_\beta + \mu^2 D_\gamma]$$

This is a rational
function \rightarrow

$$\phi(Y(\mu)) = \frac{1}{2} \sum_{i=1}^m \frac{\alpha_i + 2\beta_i \mu + \gamma_i \mu^2}{1 + \beta_i \mu^2}$$

Derivative of
 $Y(\mu) \rightarrow$

$$\frac{dY(\mu)}{d\mu} = \sum_{i=1}^m \frac{\beta_i + (\gamma_i - \alpha_i \beta_i) \mu - \beta_i^2 \mu^2}{(1 + \beta_i \mu^2)^2}$$

- Each numerator is an inverted parabola:  then 
- Easy to devise procedures to optimize $\phi(Y(\mu))$

Z Careful in case β_i 's are small !

ALGORITHM : 3 Gradient Ascent algorithm

0. **Start:** Select initial Y such that $Y^T Y = I$.
1. Compute $G = AY - YC_Y$
2. **While** $\|G\|_F > tol$
3. Compute and Diagonalize $G^T G$ as $G^T G = U D_\beta U^T$
4. Compute D_α, D_γ
5. Call *get_mu* to approximately maximize $\phi(Y(\mu))$
6. Set $Y := (Y + \mu G)U[I + \mu^2 D_\beta]^{-1/2}$
7. Compute $G = AY - YC_Y$
8. **EndWhile**

Use of Conjugate Gradients [work in progress (!)]

- Can't use perspective of linear CG [obj. function not quadratic]
- Also we are maximizing a function [$\phi(Y)$]
- An approach based on a Polak-Ribiere formulation works quite well. New Conj. Direction P :

$$P_{new} = P + \beta G_{new} \quad \text{where} \quad \beta = \frac{\langle G_{new} - G, G_{new} \rangle}{\langle G, G \rangle}$$

- But we will also project new P on tangent space:

$$P_{new} \leftarrow (I - YY^T)P_{new}$$

- Since $Y_{new}^T P = 0$ formulas similar to Grad. case available [Slightly more expensive]

Conjugate Gradients – Polak-Ribiere

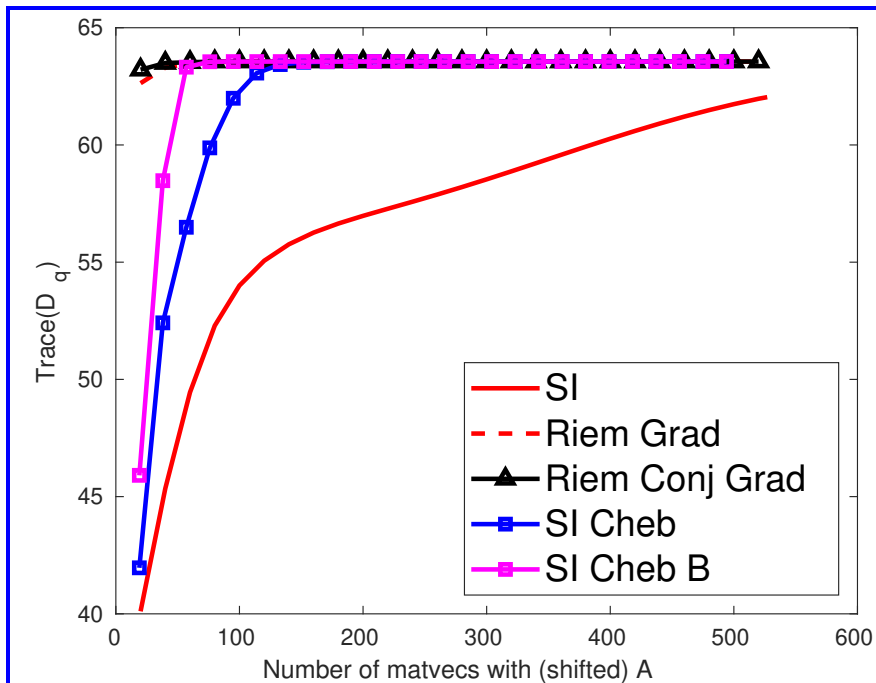
ALGORITHM : 4 Conjugate Gradient Ascent algorithm

0. **Start:** Select initial Y such that $Y^T Y = I$.
1. Compute $G = AY - Y C_Y$; Set $P := G$
2. **While** $\|G\|_F > tol$
3. Call `get_mu` to approximately maximize $\phi(Y(\mu))$
4. Set $[Y, R] = qr(Y + \mu P, 0)$ [Matlab]
5. Compute $G_{new} = AY - Y C_Y$
6. Compute $\beta = \frac{\langle G_{new} - G, G_{new} \rangle}{\langle G, G \rangle}$ and set:
7. $P_{new} := G_{new} + \beta P$ and $G := G_{new}$
8. $P_{new} := (I - Y Y^T) P_{new}$
9. **EndWhile**

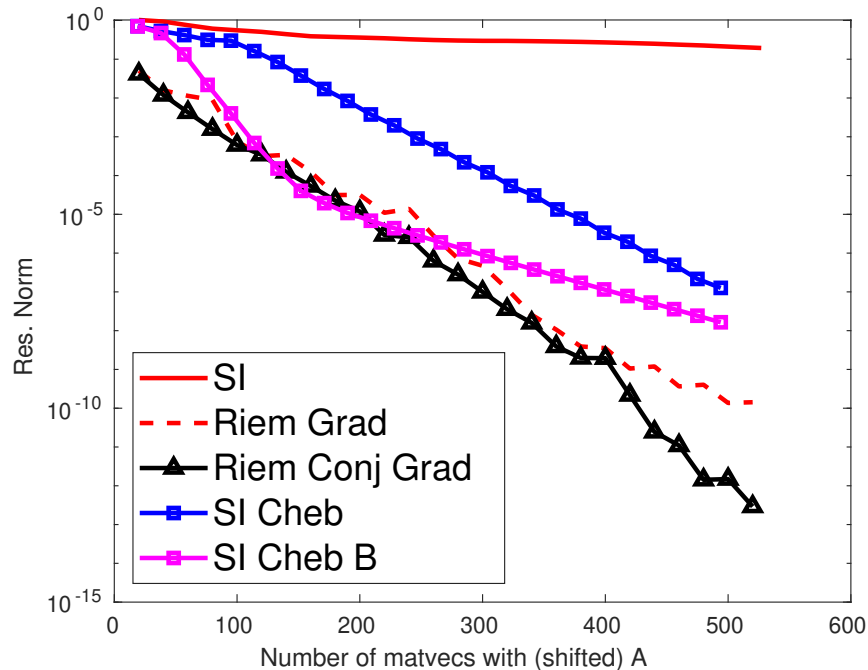
A few numerical tests. Laplacean example

- Small Finite Difference Laplacean on 35×40 grid ($n = 1,400$)
- All tests: $m = \text{Subsp. dim.} \equiv 8$
- For Standard Subspace iteration – we apply optimal shift so $A \rightarrow A - \sigma I$ [where $\sigma = (\lambda_n + \lambda_9)/2$]
- Tests: 1) Standard (shifted) subspace iteration (SI) 2) Riemann Gradient Descent 3) Chebyshev SI with Optimal pol. ; 4) Alternate Chebyshev SI 5) Riemman. Conj. Gradient

Small Laplacean [35×40 grid, $n = 1400$, $nnz = 6850$]



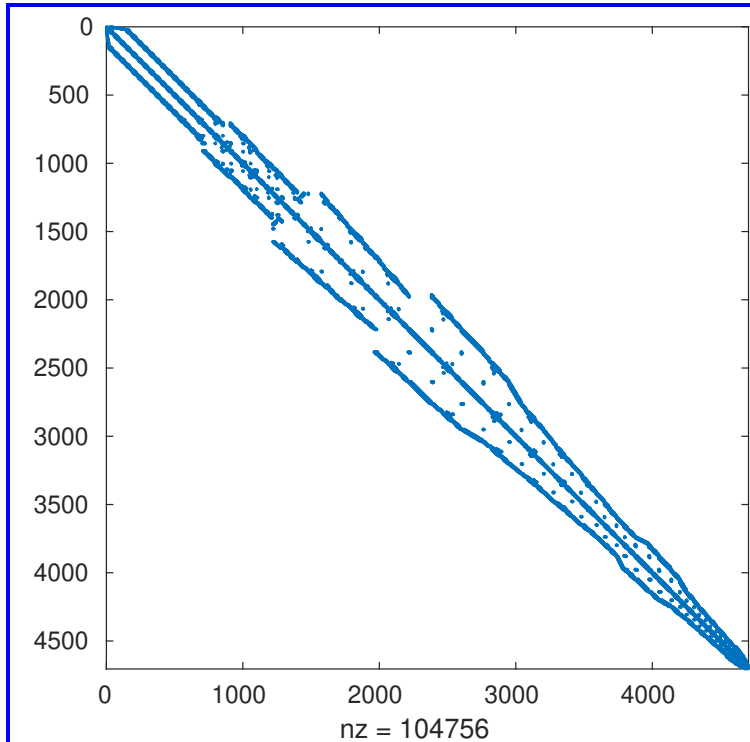
Trace of C_Y vs. its



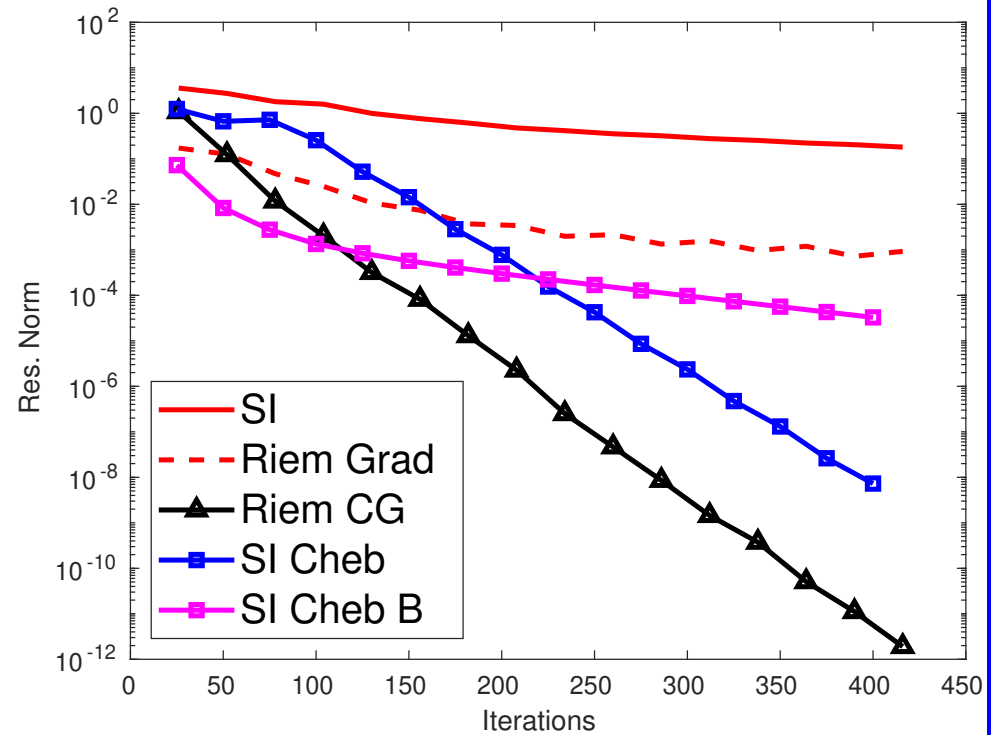
Invariance Meas. vs. its

Performance measures: 1) Trace; 2) Invariance $\|AY - YC_Y\|_1$

Matrix *nasa4704* [$n = 4,704, nnz = 104,756$]

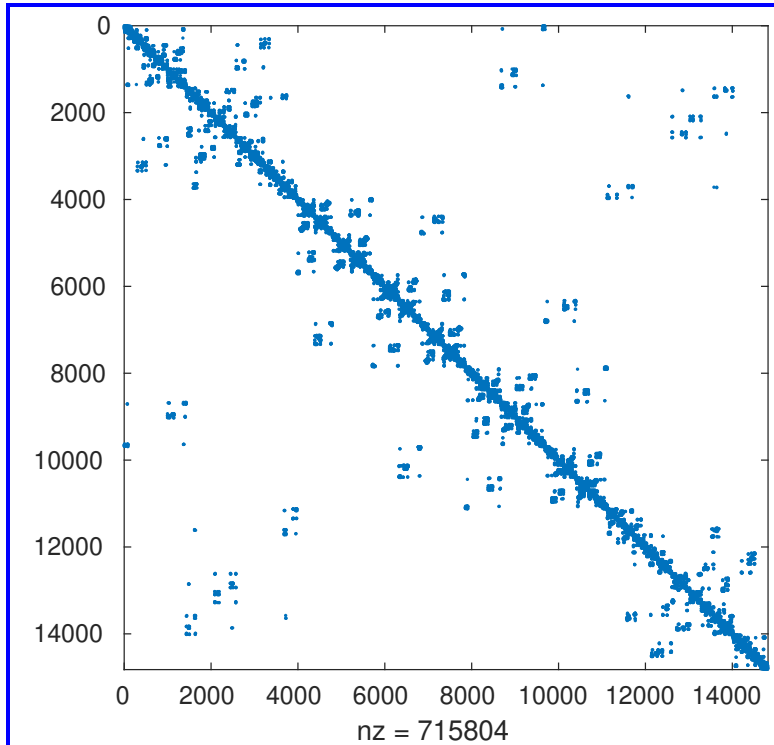


Matrix pattern

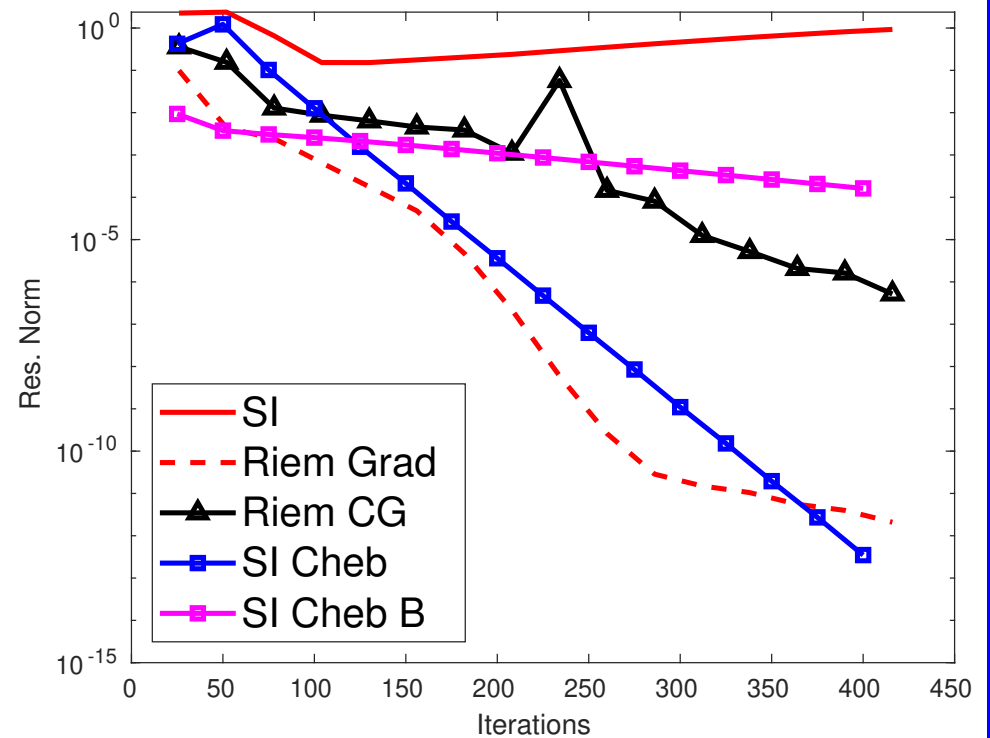


Invariance Meas. vs. its

Matrix Pre. Poisson [$n = 14,822, nnz = 715,804$]



Matrix pattern



Invariance Meas. vs. its

Concluding remarks

- Many tasks in applications deal with invariant subspaces
- Beneficial to explore algorithms that treat invariant subspaces as Grassmannian objects
- Krylov subspace methods not best choice for types of problems that arise in some applications ...
- ... but they are amazingly powerful for other tasks [e.g. Spectral densities]