

Algebraic Multigrid for the Singular Value Problem

UNIVERSITY OF
WATERLOO

uwaterloo.ca

Hans De Sterck

Department of Applied Mathematics
University of Waterloo

1. introduction

goal:

compute a few of the largest or smallest singular values of a rectangular matrix $A \in \mathbb{R}^{m \times n}$ and their associated singular vectors

introduction

- SVD of $A \in \mathbb{R}^{m \times n}$

$$A = U \Sigma V^t \quad m \geq n$$

$$U \in \mathbb{R}^{m \times m} \quad U^t U = I_m$$

$$V \in \mathbb{R}^{n \times n} \quad V^t V = I_n$$

$$\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_l \geq 0 \quad l = \min(m, n)$$

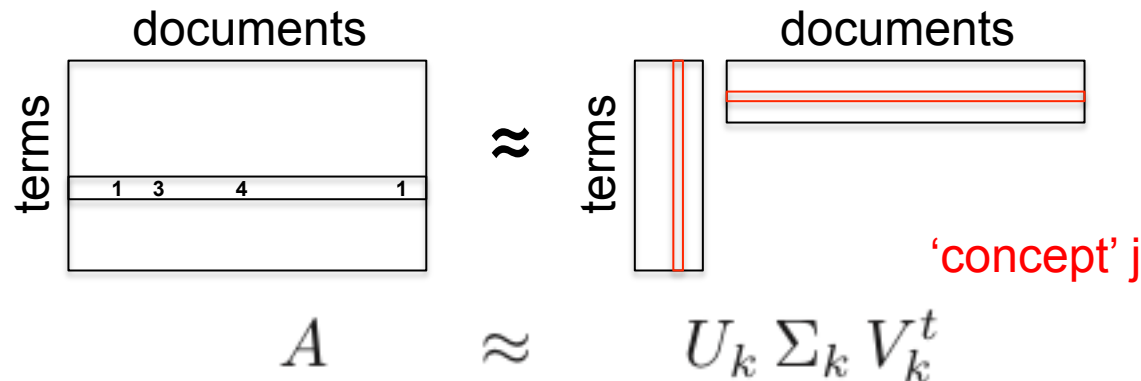
- for definiteness: we seek n_b dominant singular triplets (σ_j, u_j, v_j)

$$A v_j = \sigma_j u_j,$$

$$A^t u_j = \sigma_j v_j.$$

introduction

- why interest in dominant singular triplets?
 - the k dominant triplets give the best rank- k approximation to A
 - applications: principal component analysis
 - applications: term-document matrices



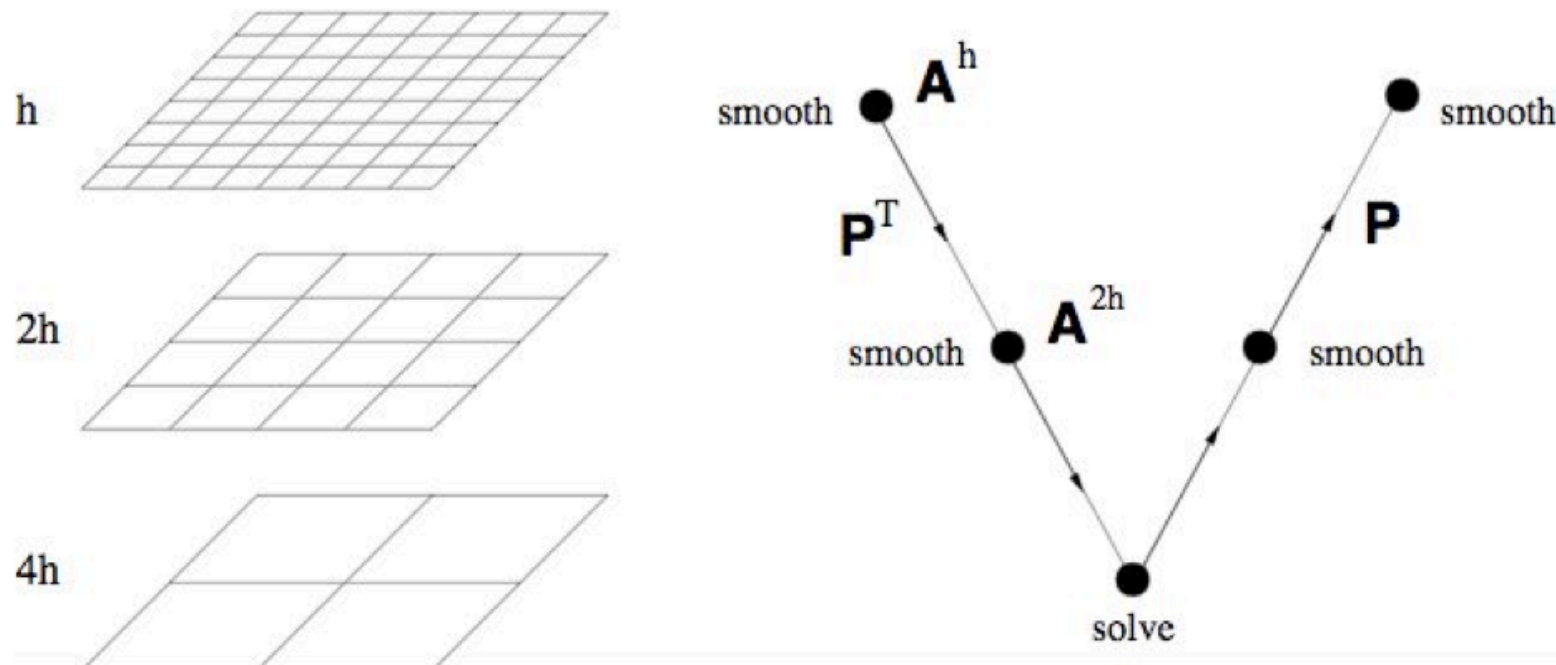
(note: nonnegative factorization is better)

introduction

- why consider (algebraic) multigrid (AMG) for dominant singular triplets?
 - for certain types of problems, multigrid may outperform other methods
 - because we can! ;-)

introduction

- algebraic multigrid V-cycle



introduction

$$A = U \Sigma V^t$$

- special case:
A symmetric positive definite (SPD)

$$A = V \Lambda V^t$$

$$A v_j = \lambda_j v_j$$

- our SVD approach will be applicable to SPD eigenproblem as a special case (or the other way around)

2. AMG for SPD eigenproblems

1) AMG for minimal eigenpairs by Borzi and Borzi

INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING
Int. J. Numer. Meth. Engng 2006; **65**:1186–1196
Published online 19 September 2005 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/nme.1478

Algebraic multigrid methods for solving generalized
eigenvalue problems

Alfio Borzi^{1,‡} and Giuseppe Borzi^{2,*†}

- use standard AMG interpolation to build P (for elliptic PDE)
- P contains slow-to-converge near-nullspace components in its range (including ‘small’ eigenvectors)

- additive correction formula:
$$v_j^{(i+1)} = v_j^{(i)} + P e_c$$

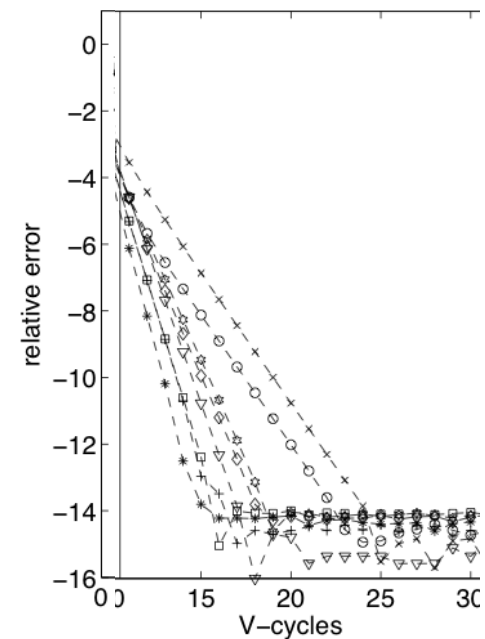
AMG for SPD eigenproblems

AMG for minimal eigenpairs by Borzi and Borzi

- use standard AMG interpolation to build P (for elliptic PDE)
- P contains slow-to-converge near-nullspace components in its range (including ‘small’ eigenvectors)
- additive correction formula:

$$v_j^{(i+1)} = v_j^{(i)} + P e_c$$

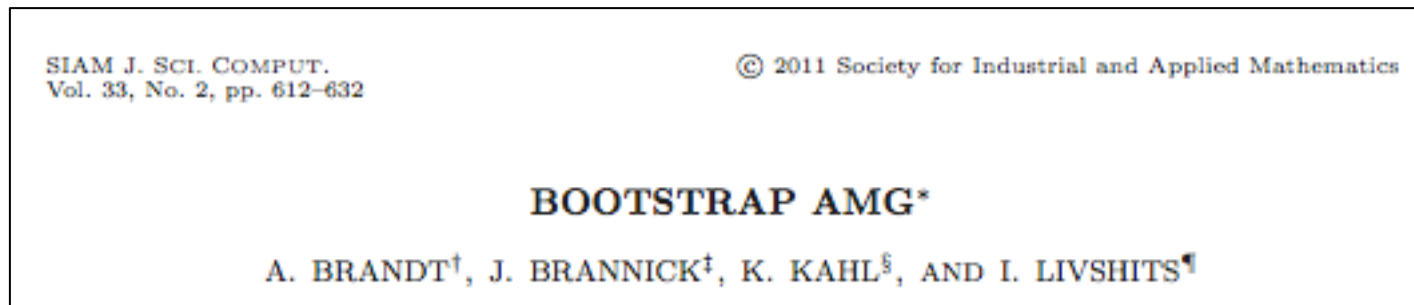
- **plus:** converges with high accuracy
- **minus:** not flexible, only works for small eigenvectors for ‘easy’ elliptic PDEs



AMG for SPD eigenproblems

2) adaptive 'bootstrap' AMG for minimal eigenpairs

-Brandt, Brannick, Kahl and Livshits



-Kushnir, Galun and Brandt



AMG for SPD eigenproblems

adaptive 'bootstrap' AMG for minimal eigenpairs

-Brandt, Brannick, Kahl and Livshits

-Kushnir, Galun and Brandt

- build P via bootstrap AMG (BAMG) approach
- P approximately fits all desired eigenvectors in its range ('exact interpolation scheme')
- multiplicative update formula: $v_j^{(i+1)} = P v_{c,j}$

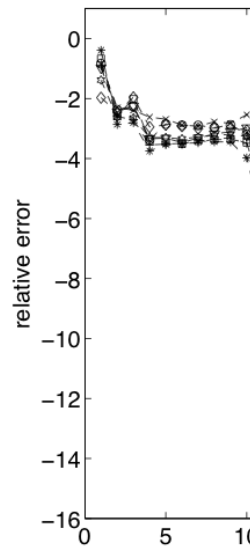
AMG for SPD eigenproblems

adaptive 'bootstrap' AMG for minimal eigenpairs

- build P via bootstrap AMG (BAMG) approach
- P approximately fits all desired eigenvectors in its range
- multiplicative update formula:

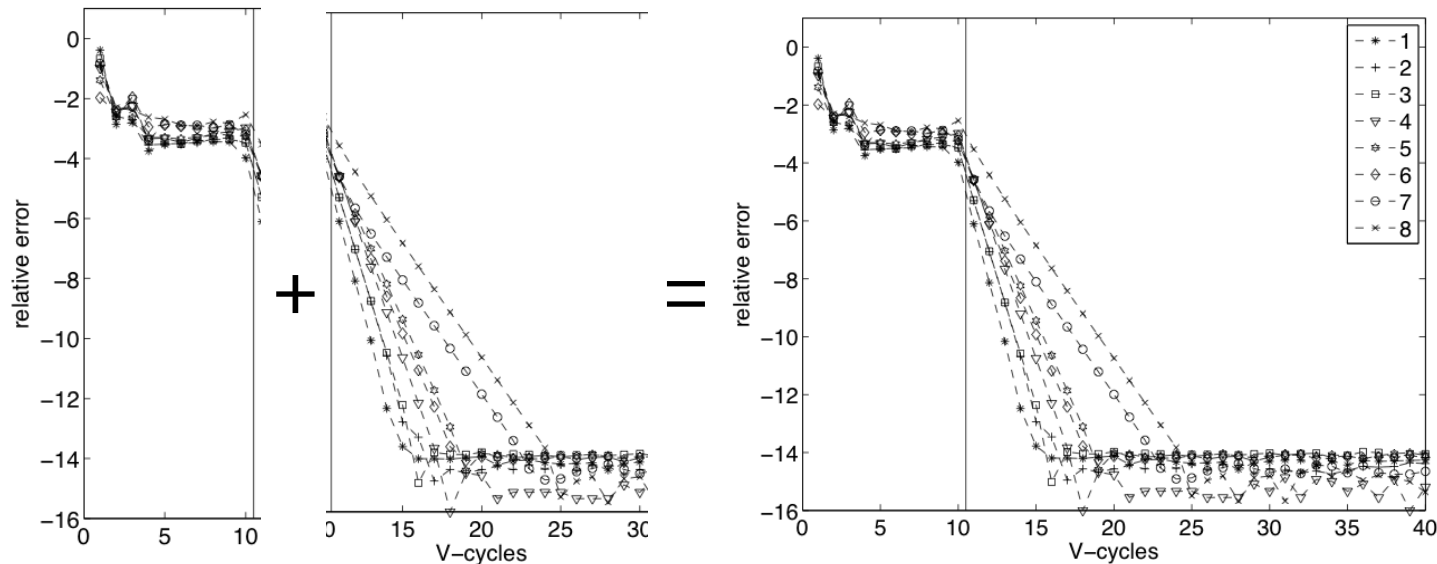
$$v_j^{(i+1)} = P v_{c,j}$$

- **plus:** flexible, adapts to eigenvectors sought
- **minus:** accuracy limited by accuracy by which the desired eigenvectors are collectively fitted by P



our approach

- combine 2) with 1): combine multiplicative (setup) phase (build P) with additive (solve) phase (like adaptive AMG for linear equation systems)
- extend to SVD computation (including dominant triplets)



UNIVERSITY OF
WATERLOO

3. multiplicative phase: coarse-level equations

- goals of the multiplicative (setup) phase:
 - find n_b tentative dominant triplets (σ_j, u_j, v_j)
 - determine interpolation operators P and Q that approximately contain the tentative singular vectors in their ranges collectively, on all levels

multiplicative phase: coarse-level equations

- assume we know triplet (σ, u, v) satisfying

$$\begin{aligned}Av &= \sigma u, & A &\in \mathbb{R}^{m \times n} \\ A^t u &= \sigma v.\end{aligned}$$

- assume P and Q have u and v exactly in their ranges:

$$\begin{aligned}u &= P u_c, \\ v &= Q v_c,\end{aligned} \quad \boxed{P} \quad \begin{aligned}P &\in \mathbb{R}^{m \times m_c} \\ Q &\in \mathbb{R}^{n \times n_c}\end{aligned}$$

- coarse equations:

$$\begin{aligned}P^t A Q v_c &= \sigma P^t B P u_c, \\ Q^t A^t P u_c &= \sigma Q^t C Q v_c,\end{aligned}$$

$$B = I_m \quad C = I_n$$

multiplicative phase: coarse-level equations

- assume we know triplet (σ, u, v)

$$\begin{aligned} A v &= \sigma u, & u &= P u_c, & P^t A Q v_c &= \sigma P^t B P u_c, \\ A^t u &= \sigma v. & v &= Q v_c, & Q^t A^t P u_c &= \sigma Q^t C Q v_c, \end{aligned}$$

- define coarse-level operators and equations

$$\begin{aligned} A_c &= P^t A Q, & A_c v_c &= \sigma B_c u_c, \\ B_c &= P^t B P, & A_c^t u_c &= \sigma C_c v_c. \\ C_c &= Q^t C Q, \end{aligned}$$

- coarse level will help: solving coarse equations (cheaper) gives exact answer in one step!
- do this approximately, and recursively (V-cycle)

4. an uncommon (new?) generalized SVD

- recall generalized symmetric eigenvalue problem for

$$A, B \in \mathbb{R}^{m \times m} \quad (B \text{ SPD})$$

$$A v = \lambda B v \quad A = B V \Lambda V^t \quad V^t B V = I_m$$

- we have to solve coarse-grid problem

$$A v = \sigma B u, \quad A \in \mathbb{R}^{m \times n}$$

$$A^t u = \sigma C v, \quad B \in \mathbb{R}^{m \times m} \quad C \in \mathbb{R}^{n \times n} \quad (B, C \text{ SPD})$$

- we have to generalize the SVD problem

$$A v = \sigma u, \quad A = U \Sigma V^t$$

$$A^t u = \sigma v.$$

an uncommon (new?) generalized SVD

$$\begin{aligned} A \in \mathbb{R}^{m \times n} \quad & A v = \sigma B u, \\ & A^t u = \sigma C v, \end{aligned} \quad (3.6)$$

DEFINITION 3.1 (Generalized singular value decomposition). *The generalized singular value decomposition of $A \in \mathbb{R}^{m \times n}$ with respect to $B \in \mathbb{R}^{m \times m}$ and $C \in \mathbb{R}^{n \times n}$, with B and C SPD, is given by*

$$A = B U \Sigma V^t C, \quad (3.7)$$

with $U \in \mathbb{R}^{m \times m}$, $V \in \mathbb{R}^{n \times n}$ and $\Sigma \in \mathbb{R}^{m \times n}$. The columns of U are called the left generalized singular vectors, and the columns of V are called the right generalized singular vectors. They satisfy the orthogonality relations $U^t B U = I_m = U B U^t$ and $V^t C V = I_n = V C V^t$. Matrix Σ has the $l = \min(m, n)$ real nonnegative generalized singular values $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_l \geq 0$ on its diagonal. Eqs. (3.6) are called the generalized singular value problem for matrix A with respect to matrices B and C .

- this appears to be uncommon in the literature

first way to compute the generalized SVD $A = BU\Sigma V^t C$,

THEOREM 3.2. *Generalized SVD (3.7) has the same existence and uniqueness properties as the standard SVD.*

Proof. This follows from a simple change of variables: with

$$\begin{aligned} T &= B^{1/2} U, \\ W &= C^{1/2} V, \\ D &= B^{-1/2} A C^{-1/2}, \end{aligned} \tag{3.8}$$

generalized SVD (3.7) can be rewritten as a standard SVD

$$D = T \Sigma W^t. \tag{3.9}$$

second way to compute the generalized SVD $A = B U \Sigma V^t C$,

THEOREM 3.3. *Let $A \in \mathbb{R}^{m \times n}$, $B \in \mathbb{R}^{m \times m}$ and $C \in \mathbb{R}^{n \times n}$, with B and C SPD. Let $l = \min(m, n)$. Then generalized eigenvalue problem*

$$\left(\begin{bmatrix} 0 & A \\ A^t & 0 \end{bmatrix} - \sigma \begin{bmatrix} B & 0 \\ 0 & C \end{bmatrix} \right) \begin{bmatrix} u \\ v \end{bmatrix} = 0, \quad (3.13)$$

has $m + n$ solution triplets (σ, u, v) with linearly independent eigenvectors $[u^t \ v^t]^t \neq 0$. There are l independent solutions with $\sigma_j \geq 0$ and vectors u_j and v_j satisfying orthogonality relations $u_j^t B u_i = \delta_{i,j}$ and $v_j^t C v_i = \delta_{i,j}$ ($j = 1, \dots, l$). The triplets (σ_j, u_j, v_j) are the generalized singular triplets of A with respect to B and C . Furthermore, there are l independent solutions $(-\sigma_j, u_j, -v_j)$. Finally, there are $\text{abs}(m - n) = m + n - 2l$ independent solutions with $\sigma = 0$ and either $u = 0$ or $v = 0$.

third way to compute the generalized
SVD $A = B U \Sigma V^t C$,

$$(A^t B^{-1} A) v = \sigma^2 C v,$$

$$(A C^{-1} A^t) u = \sigma^2 B u.$$

5. multiplicative phase: BAMG V-cycles

- find n_b tentative dominant triplets (σ_j, u_j, v_j)
- start from n_t random fine-level test vectors
- do relaxation on test vectors using the power method, to obtain first approximations for ‘large’ singular vectors:
 - start from random v
 - compute new u, σ via $A v = \sigma u,$
 - compute new v, σ via $A^t u = \sigma v.$
 - repeat
- determine P and Q to fit the n_t test vectors u and v collectively

$$u = P u_c,$$

$$v = Q v_c,$$

multiplicative phase: V-cycles

- downwards sweep of first V-cycle: create coarse grids and coarse-level operators P , Q , A_c , B_c , C_c for all levels, using relaxation on n_t initially random test vectors

n_t random test vectors

relax test vectors, coarsen, build P , Q , A_c , B_c , C_c

inject test vectors

relax test vectors, coarsen, build P , Q , A_c , B_c , C_c

inject test vectors

relax test vectors, coarsen, build P , Q , A_c , B_c , C_c

inject test vectors

$$A_c v_c = \sigma B_c u_c,$$

$$A_c^t u_c = \sigma C_c v_c.$$

relax: $A_c v_c = \sigma B_c u_c,$

$$A_c^t u_c = \sigma C_c v_c.$$

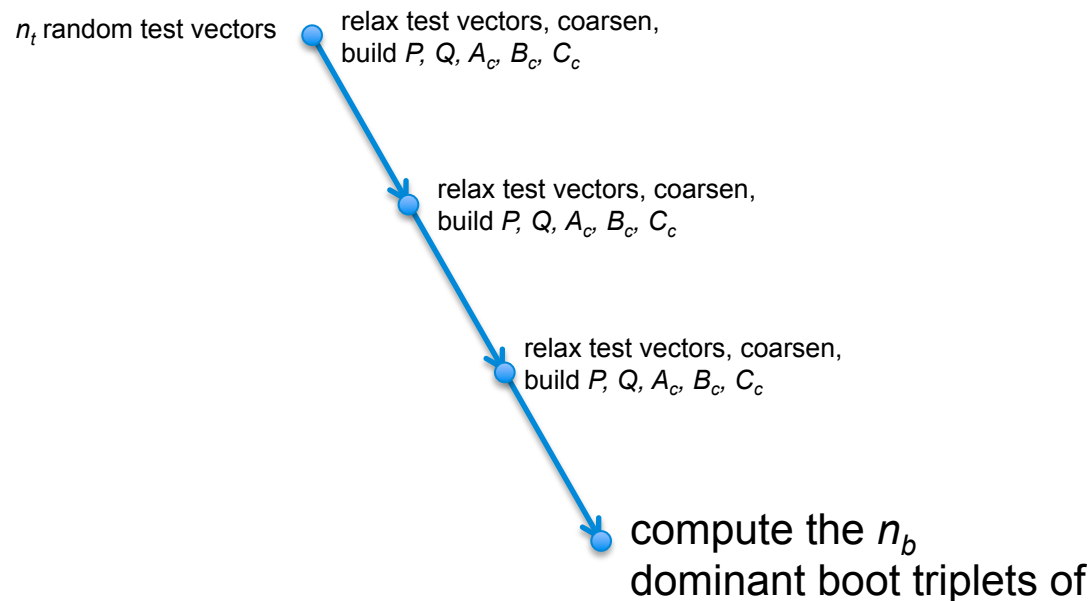
build A_c , B_c , C_c : $A_c = P^t A Q,$

$$B_c = P^t B P,$$

$$C_c = Q^t C Q,$$

multiplicative phase: V-cycles

- on the coarsest level: solve the generalized SVD problem, and select the n_b dominant triplets (σ_j, u_j, v_j) as the first (coarse) approximations of the dominant triplets sought (we call these ‘boot triplets’)

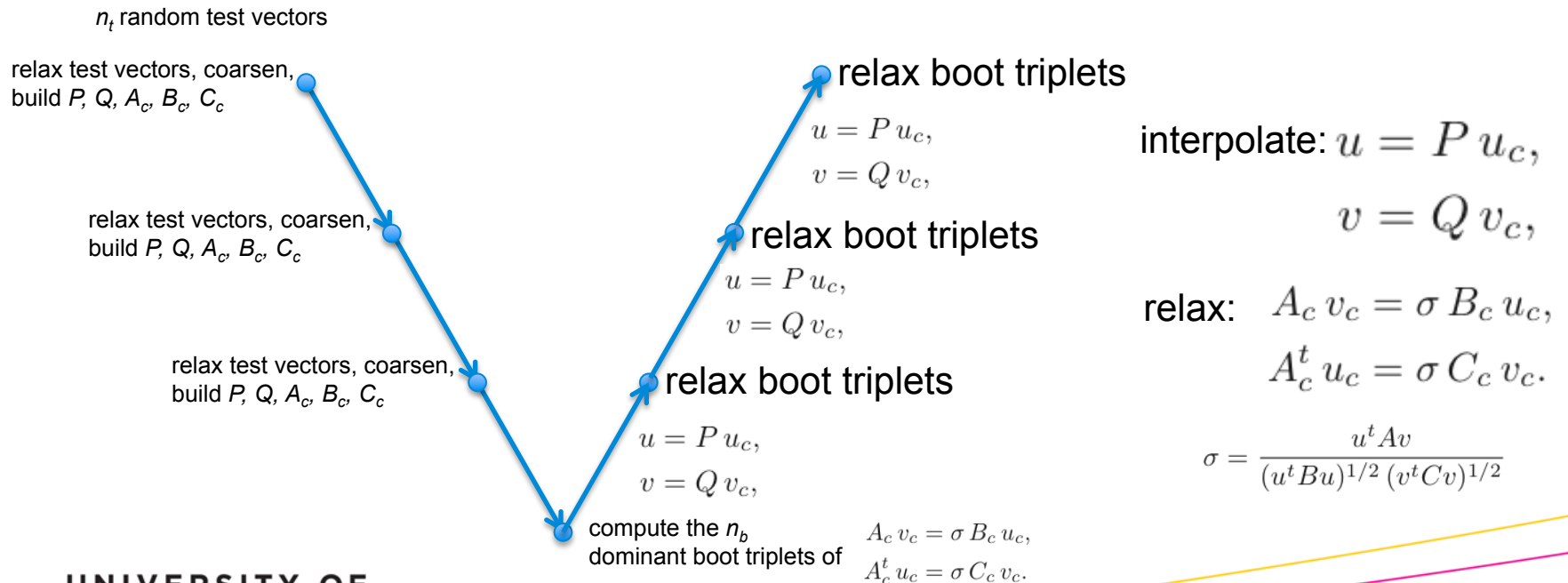


$$A_c v_c = \sigma B_c u_c,$$

$$A_c^t u_c = \sigma C_c v_c.$$

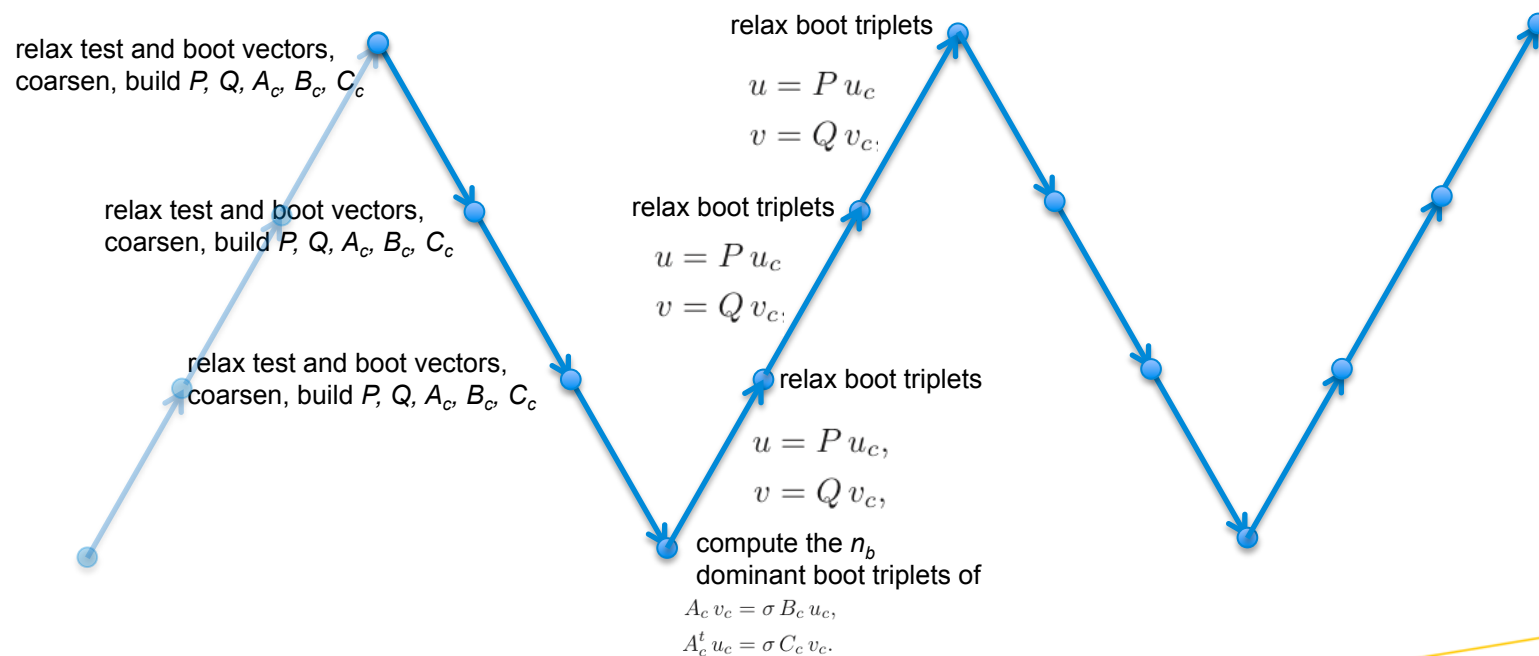
multiplicative phase: V-cycles

- upward sweep of first V-cycle: interpolate the n_b boot triplets up to finer levels, and relax (first fix σ and find u or v , then update σ via Rayleigh quotient formula), on each level



multiplicative phase: V-cycles

- repeat V-cycles until convergence stagnates (P and Q represent the boot vectors u and v collectively up to some accuracy)



6. multiplicative phase: relaxation

- test vectors: power method on

$$A v = \sigma B u,$$

$$A^t u = \sigma C v,$$

with inexact inversion of B and C (weighted Jacobi):

$$A^t u_j = C \bar{v}_j,$$

$$A v_j = B \bar{u}_j,$$

$$v_j = \bar{v}_j / (\bar{v}_j^t C \bar{v}_j)^{1/2}$$

$$u_j = \bar{u}_j / (\bar{u}_j^t B \bar{u}_j)^{1/2}.$$

$$\bar{v}_j^{(i+1)} = \bar{v}_j^{(i)} - \omega_J D_C^{-1} (C \bar{v}_j^{(i)} - A^t u_j)$$

multiplicative phase: relaxation

- boot vectors: block Gauss-Seidel (fix σ) on

$$A v = \sigma B u + \kappa,$$

$$A^t u = \sigma C v + \tau.$$

with inexact inversion of B and C (weighted Jacobi):

$$u_j^{(i+1)} = u_j^{(i)} - \omega_J D_B^{-1} (B u_j^{(i)} - (A v_j - \kappa) / \sigma_j).$$

- update σ using Rayleigh quotient formula

$$\sigma = \frac{u^t A v}{(u^t B u)^{1/2} (v^t C v)^{1/2}}$$

7. multiplicative phase: building P and Q

- coarsening: use standard (one-pass) AMG coarsening on AA^t for the u -variables, and on A^tA for the v -variables (correlations ...)
- in the future: coarsen directly using A
- try 'general' strength of connection formula

variable i is strongly influenced by variable j



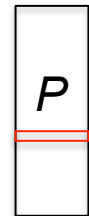
$$|n_{i,j}| \geq \theta \sum_k |n_{i,k}|$$

- interpolation stencils for F-points are formed by strongly influencing C-points (sparsity of P and Q)

multiplicative phase: building P and Q

- determine the weights in P and Q via least-squares fitting of the test (and boot) vectors (injected to the C-points)
- (for P) for each F-point i :

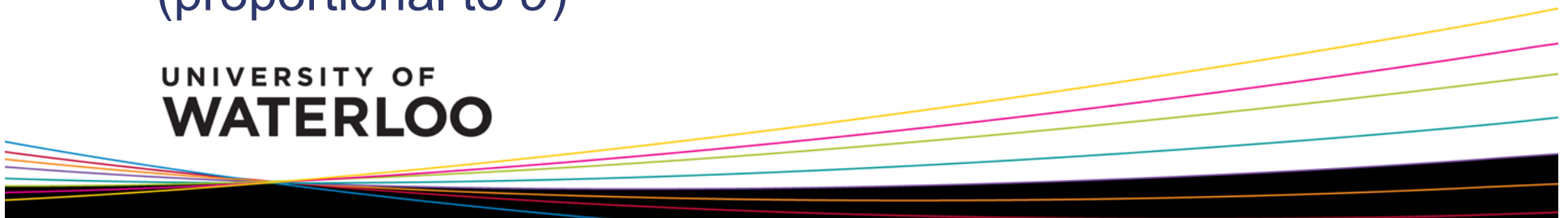
$$u_k^i = \sum_{j \in C_u^i} p_{i,j} u_{k,c}^j \quad (k = 1, \dots, n_f)$$



(one equation per test or booth vector k)

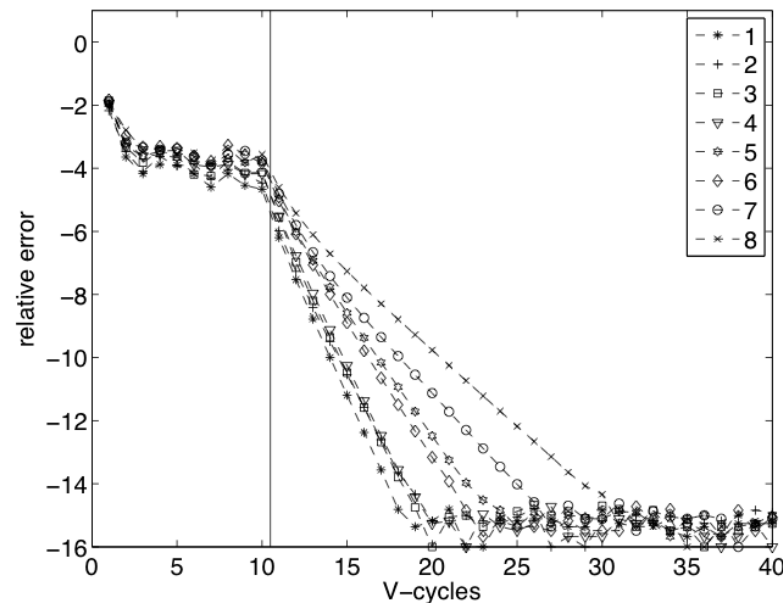
(over-determined LS system: more test+boot vectors than size of largest stencil)

- larger weight for boot vectors than for test vectors (proportional to σ)



OK, where are we...

- I have discussed how to do the first phase of the algorithm (multiplicative, find tentative triplets starting from random test vectors, build P and Q , bootstrap AMG)
- I will now discuss the second phase (additive V-cycles, use 'frozen' P and Q)



8. additive phase: V-cycles

- for each tentative boot triplet, keep σ fixed, improve u and v in additive-correction V-cycle
- coarse-level equations:

$$\begin{aligned} A v_j - \sigma_j B u_j &= \kappa_j, \\ A^t u_j - \sigma_j C v_j &= \tau_j, \end{aligned} \quad \Longrightarrow \quad \begin{aligned} A_c v_{j,c} - \sigma_j B_c u_{j,c} &= P^t r_j, \\ A_c^t u_{j,c} - \sigma_j C_c v_{j,c} &= Q^t s_j, \end{aligned}$$

- correction formula:
$$\begin{aligned} u_j^{(i+1)} &= u_j^{(i)} + P u_{j,c}, \\ v_j^{(i+1)} &= v_j^{(i)} + Q v_{j,c}, \end{aligned}$$
- P and Q from setup phase can be used: additive errors lie approximately in their ranges

9. additive phase: Ritz projection

- on the finest level, all boot triplets (including the σ s) are updated after each set of V-cycles
- seek new $u_j \in \mathcal{U}, v_j \in \mathcal{V}$ s.t

$$\langle u, A v_j - \sigma_j B u_j \rangle_B = 0 \quad \forall u \in \mathcal{U},$$

$$\langle v, A^t u_j - \sigma_j C v_j \rangle_C = 0 \quad \forall v \in \mathcal{V}.$$

- leads to very small generalized singular value problem

$$\langle y, \hat{U}^t A \hat{V} z_j - \sigma_j \hat{U}^t B \hat{U} y_j \rangle = 0 \quad \forall y \in \mathbb{R}^{m_c},$$

$$\langle z, \hat{V}^t A^t \hat{U} y_j - \sigma_j \hat{V}^t C \hat{V} z_j \rangle = 0 \quad \forall z \in \mathbb{R}^{n_c}.$$

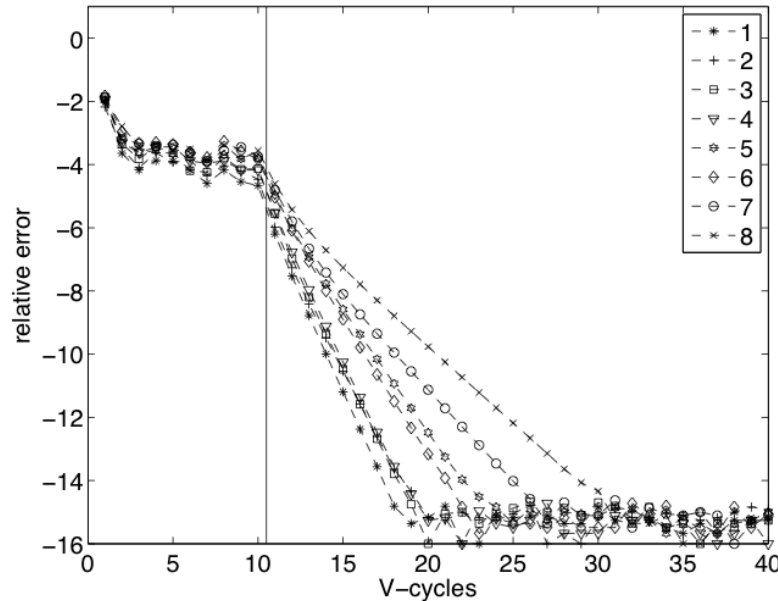
10. specializations and extensions

- square matrices (use A or A^t for coarsening)
- SPD matrices: only need A, B, P
- minimal singular triplets and eigenpairs:
 - algorithm is self-learning (adaptive), so we only need to change the relaxation and the coarsest-level solves of the multiplicative phase
 - use Kaczmarz relaxation on blocks of

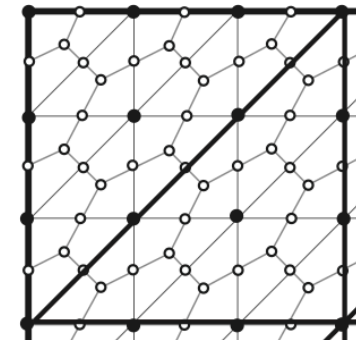
$$Av = \sigma Bu + \kappa,$$
$$A^t u = \sigma Cv + \tau.$$

11. numerical results

1) high-order finite volume element Laplacian on unit square (square, nonsymmetric, 961x961)

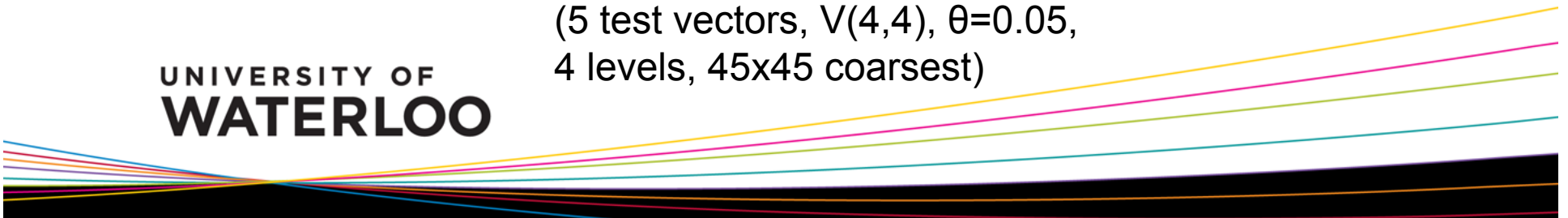


largest singular values



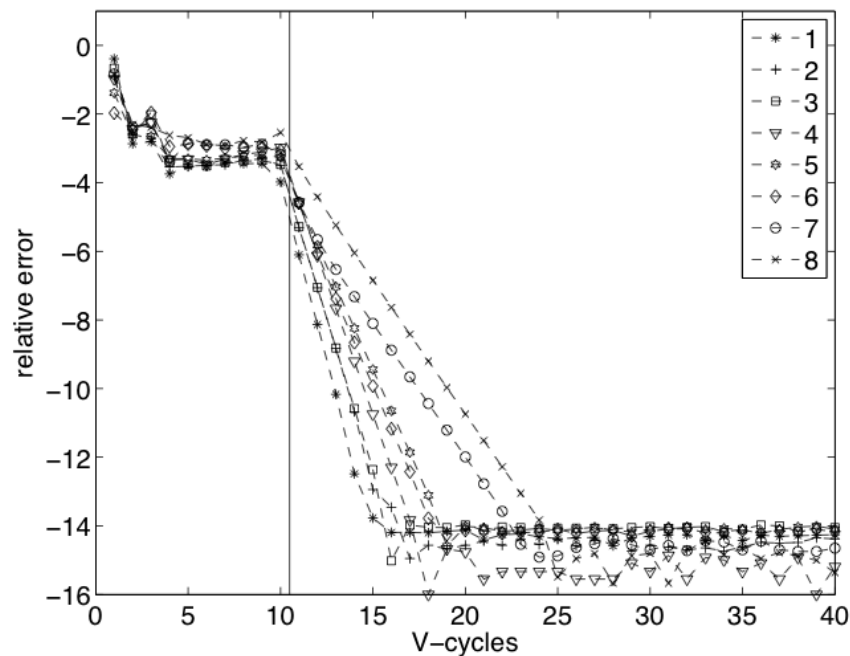
$$error = \frac{|\sigma_{exact} - \sigma_{approx}|}{\sigma_{exact}}$$

(5 test vectors, V(4,4), $\theta=0.05$,
4 levels, 45x45 coarsest)



numerical results

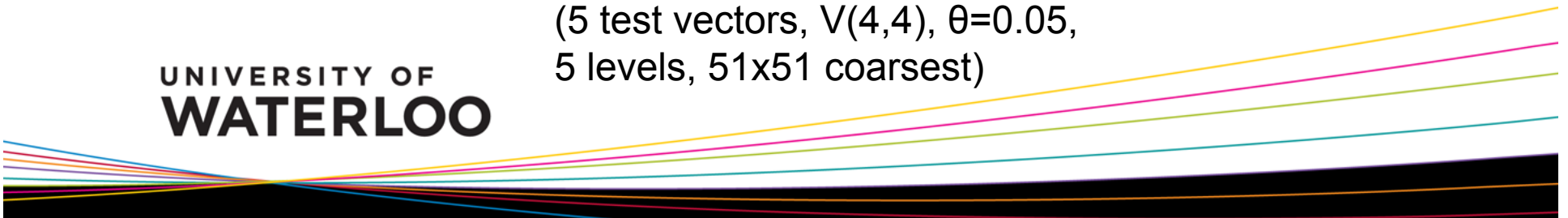
high-order finite volume element Laplacian on unit square (square, nonsymmetric, 961x961)



smallest singular values

(5 test vectors, $V(4,4)$, $\theta=0.05$,
5 levels, 51x51 coarsest)

UNIVERSITY OF
WATERLOO



numerical results

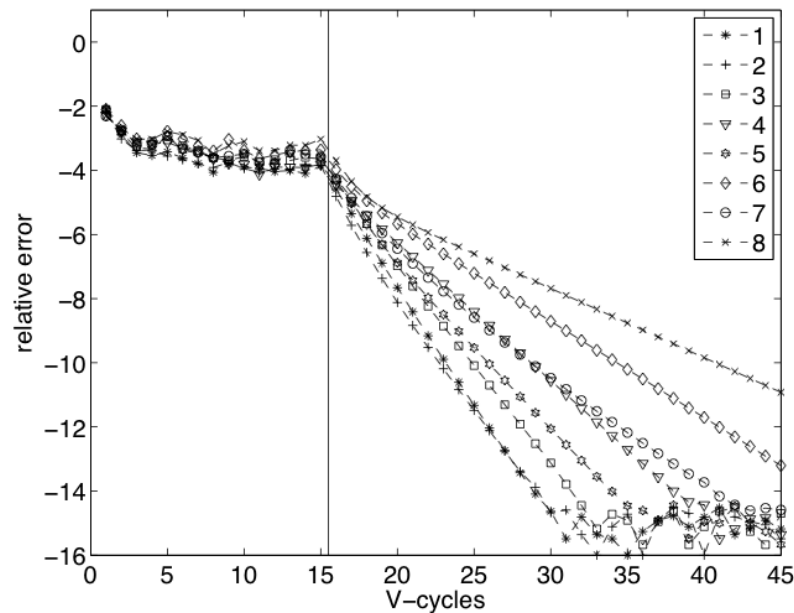
FVE lge	FVE sm	FD lge	FD sm	Graph lge	Graph sm	Term-Doc
7.9791546	0.01924183	7.9818877	0.01811231	13.509036	0.01000000	84.148337
7.9491729	0.04794913	7.9548012	0.04519876	13.352613	0.03456116	64.707532
7.9468326	0.04801773	7.9548012	0.04519876	13.350454	0.03901593	55.976437
7.9172573	0.07655365	7.9277148	0.07228521	12.472837	0.07966567	50.265499
7.8965349	0.09557904	7.9099298	0.09007021	12.416200	0.09490793	49.265360
7.8960066	0.09558103	7.9099298	0.09007021	11.874669	0.09918138	45.242034
7.8692955	0.12359047	7.8828433	0.11715666			44.400811
7.8616683	0.12415144	7.8828433	0.11715666			41.772394

TABLE 6.1

Singular values and eigenvalues sought for each problem (high-accuracy approximations).

numerical results

2) finite difference Laplacian on unit square (square, symmetric, 1024x1024)



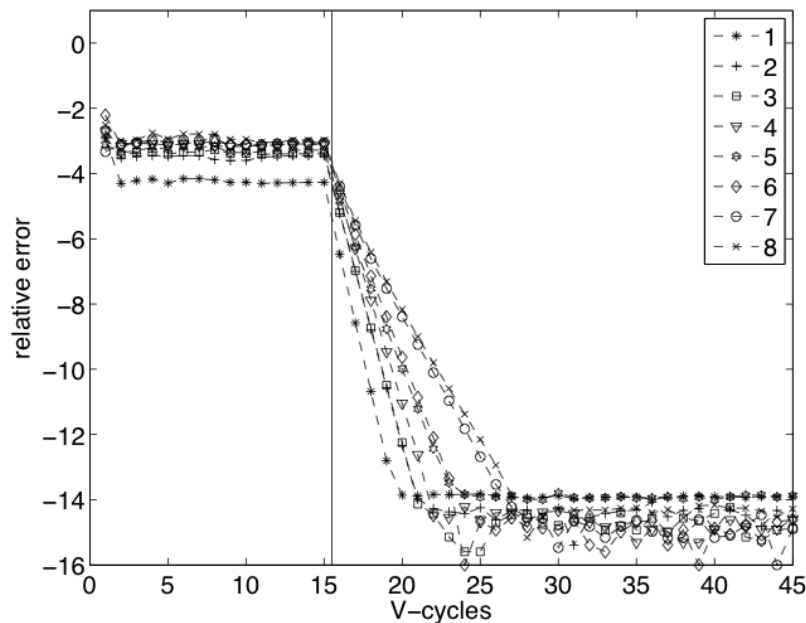
largest
eigenvalues

(6 test vectors, V(8,8) test and V(4,4) boot, $\theta=0.06$,
4 levels, 52x52 coarsest)



numerical results

finite difference Laplacian on unit square (square, symmetric, 1024x1024)



smallest
eigenvalues

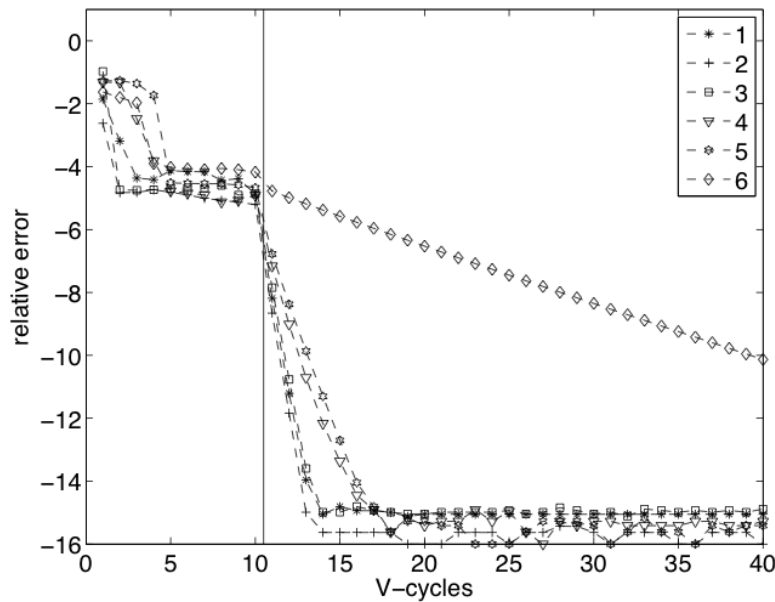
(6 test vectors, V(8,8) test and V(4,4) boot, $\theta=0.06$,
5 levels, 64x64 coarsest)

UNIVERSITY OF
WATERLOO

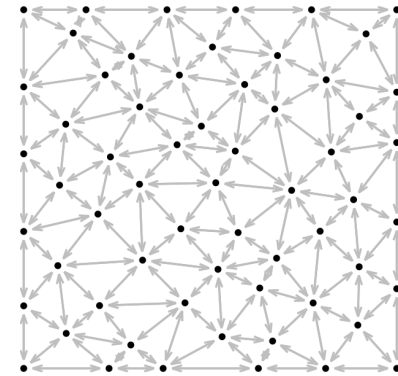


numerical results

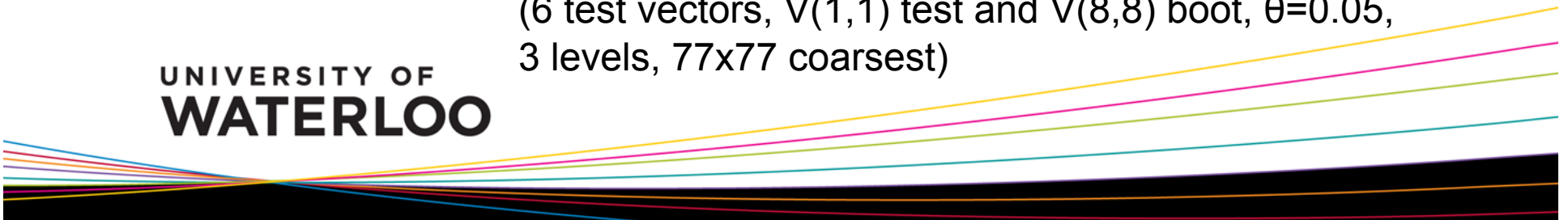
3) graph Laplacian on random triangular graph in unit square (square, symmetric, 1024x1024)



largest eigenvalues

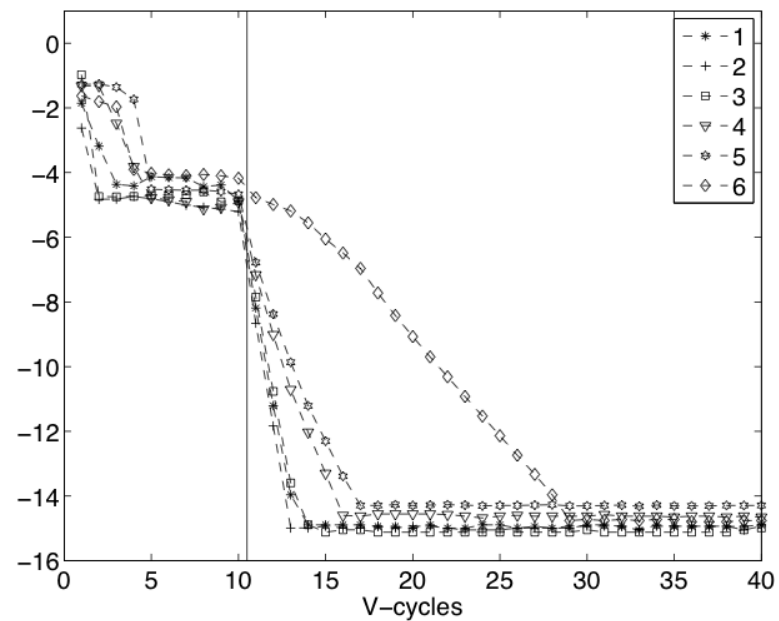
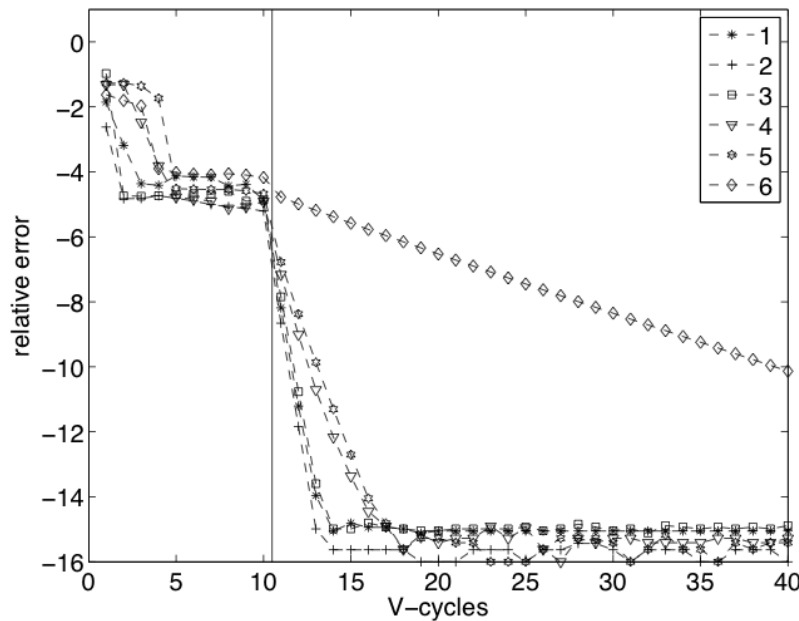


(6 test vectors, V(1,1) test and V(8,8) boot, $\theta=0.05$, 3 levels, 77x77 coarsest)



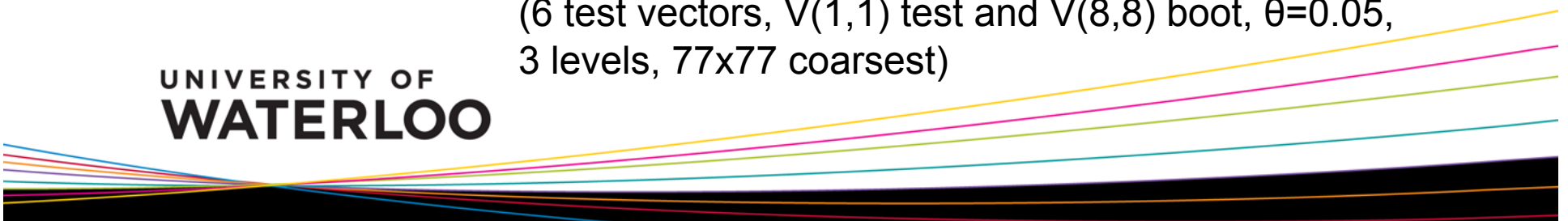
numerical results

graph Laplacian on random triangular graph in unit square (square, symmetric, 1024x1024)



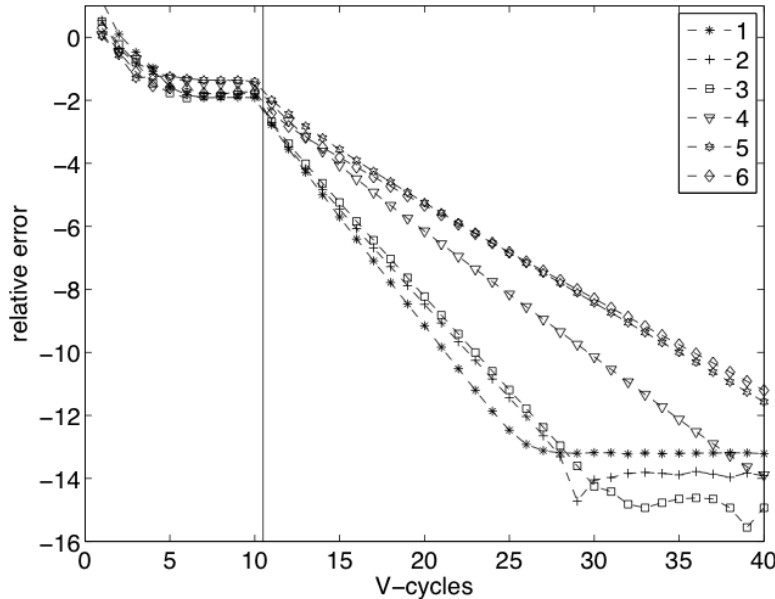
(6 test vectors, $V(1,1)$ test and $V(8,8)$ boot, $\theta=0.05$,
3 levels, 77x77 coarsest)

UNIVERSITY OF
WATERLOO



numerical results

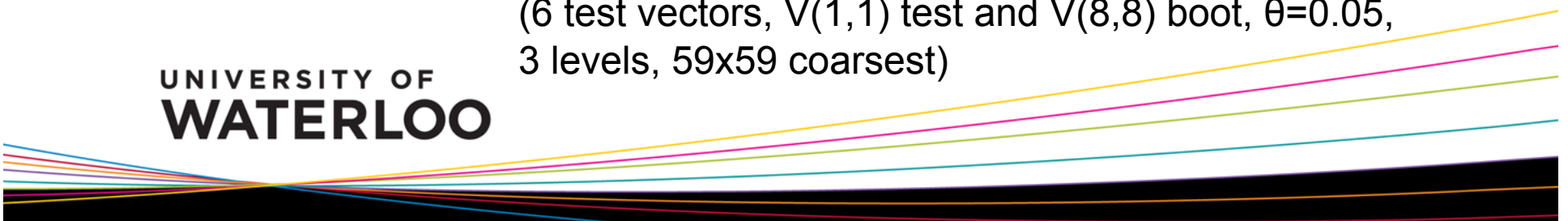
graph Laplacian on random triangular graph in unit square (square, symmetric, 1024x1024)



smallest
eigenvalues

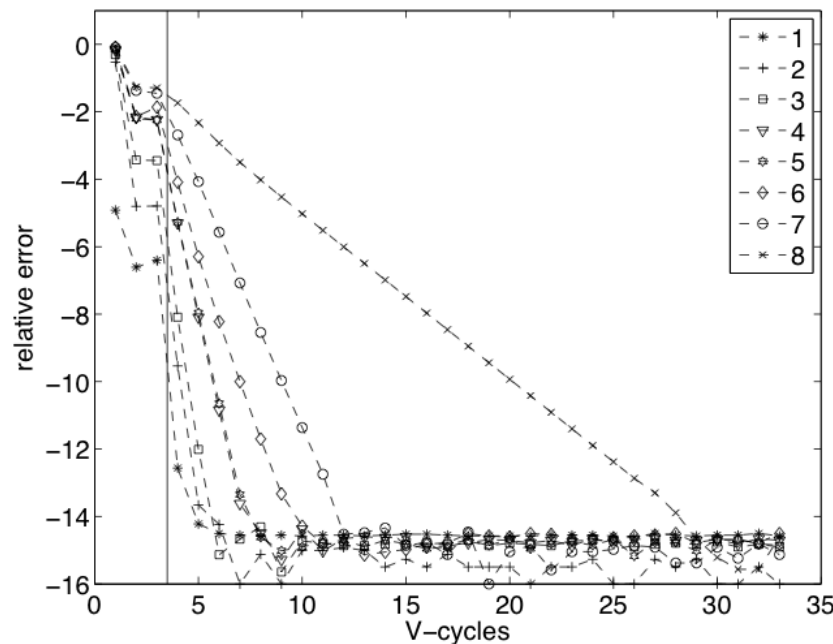
(6 test vectors, V(1,1) test and V(8,8) boot, $\theta=0.05$,
3 levels, 59x59 coarsest)

UNIVERSITY OF
WATERLOO



numerical results

4) Medline tem-document matrix (rectangular, 5735x1033)



largest
singular values

(14 test vectors, $V(1,1)$ test and $V(4,4)$ boot, $\theta=0.03$,
5 levels, 415x198 coarsest)



12. conclusions

- self-learning, collective AMG algorithm to compute a few dominant or minimal singular triplets (or eigenpairs)
- multiplicative setup phase, additive solve phase
- seems to work pretty well
- there are many parameters, and robustness needs to be improved (how many test vectors, relaxations, 'miss' some eigenvalues, ...)

conclusions

- improve coarsening (on A , compatible relaxation, general graph coarsening, small-world, others ...)
- improve multiplicative phase ('adaptive' approach instead of bootstrap?)
- improve additive phase robustness (for example, use LOBPCG or RQMG instead of V-cycle+Ritz)
- parallel?
- approach is quite general (self-learning), high accuracy, so seems promising

thank you
questions?

UNIVERSITY OF
WATERLOO

