Adaptive Algebraic Multigrid for Canonical Tensor Decomposition

UNIVERSITY OF WATERLOO

uwaterloo.ca

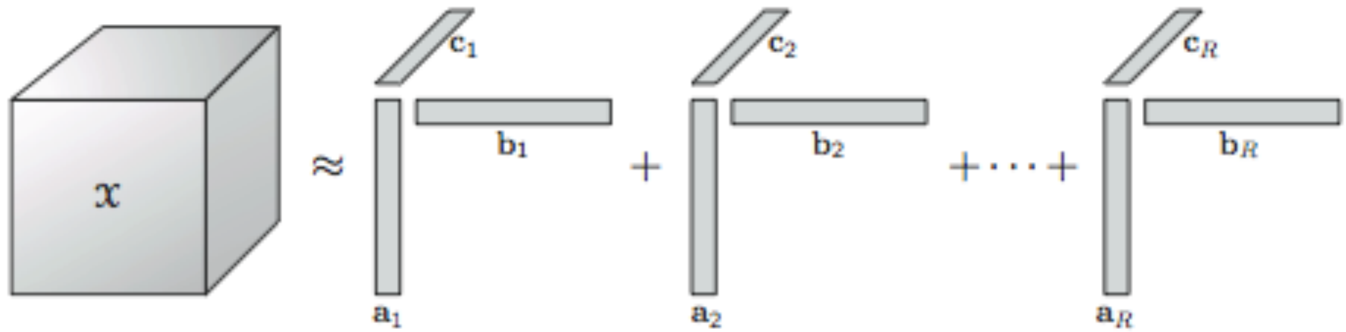
Hans De Sterck and Killian Miller

Department of Applied Mathematics
University of Waterloo, Canada

Weizmann Workshop 2013 on Multilevel Computational Methods and Optimization

(1) canonical tensor decomposition

- tensor = element of tensor product of real vector spaces (*N*-dimensional array)
- *N*=3:

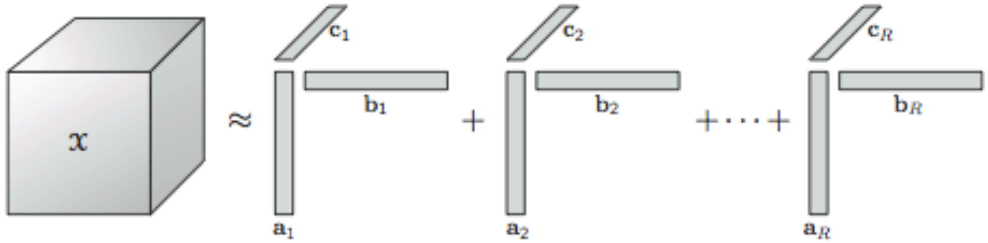


(from "Tensor Decompositions and Applications", Kolda and Bader, SIAM Rev., 2009 [1])

- canonical decomposition: decompose tensor in sum of *R* rank-one terms (approximately)

UNIVERSITY OF WATERLOO

canonical tensor decomposition



OPTIMIZATION PROBLEM

given tensor $\mathcal{T} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$, find rank-$R$ canonical tensor $\mathcal{A}_R \in \mathbb{R}^{I_1 \times \ldots \times I_N}$ that minimizes

$$f(\mathcal{A}_R) = \frac{1}{2} \|\mathcal{T} - \mathcal{A}_R\|_F^2.$$

FIRST-ORDER OPTIMALITY EQUATIONS

$$\nabla f(\mathcal{A}_R) = \mathbf{g}(\mathcal{A}_R) = 0.$$

(problem is non-convex, multiple (local) minima, solution may not exist (ill-posed), ... ; but smooth, and we assume there is a local minimum)

(de Silva and Lim, SIMAX, 2009)

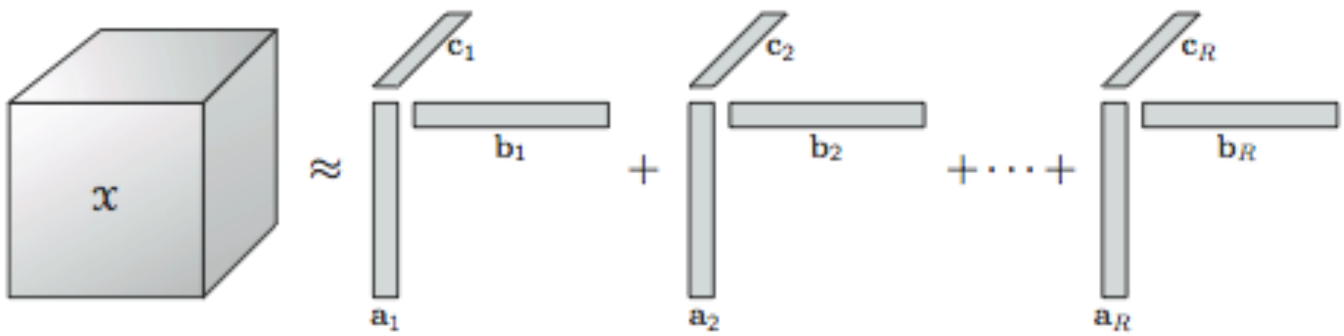
UNIVERSITY OF WATERLOO

link with singular value decomposition

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

$$A = U\,\Sigma\,V^t = \sigma_1\,u_1\,v_1^T + \ldots + \sigma_n\,u_n\,v_n^T$$

- canonical decomposition of tensor



a difference with the SVD

truncated SVD is best rank-*R* approximation:

$$A = \sigma_1 \, u_1 \, v_1^T + \ldots + \sigma_R \, u_R \, v_R^T + \sigma_{R+1} \, u_{R+1} \, v_{R+1}^T + \ldots + \sigma_n \, u_n \, v_n^T$$

$$\underset{B \text{ with rank } \leq R}{\arg\min} \; \|A - B\|_F = \sigma_1 \, u_1 \, v_1^T + \ldots + \sigma_R \, u_R \, v_R^T$$

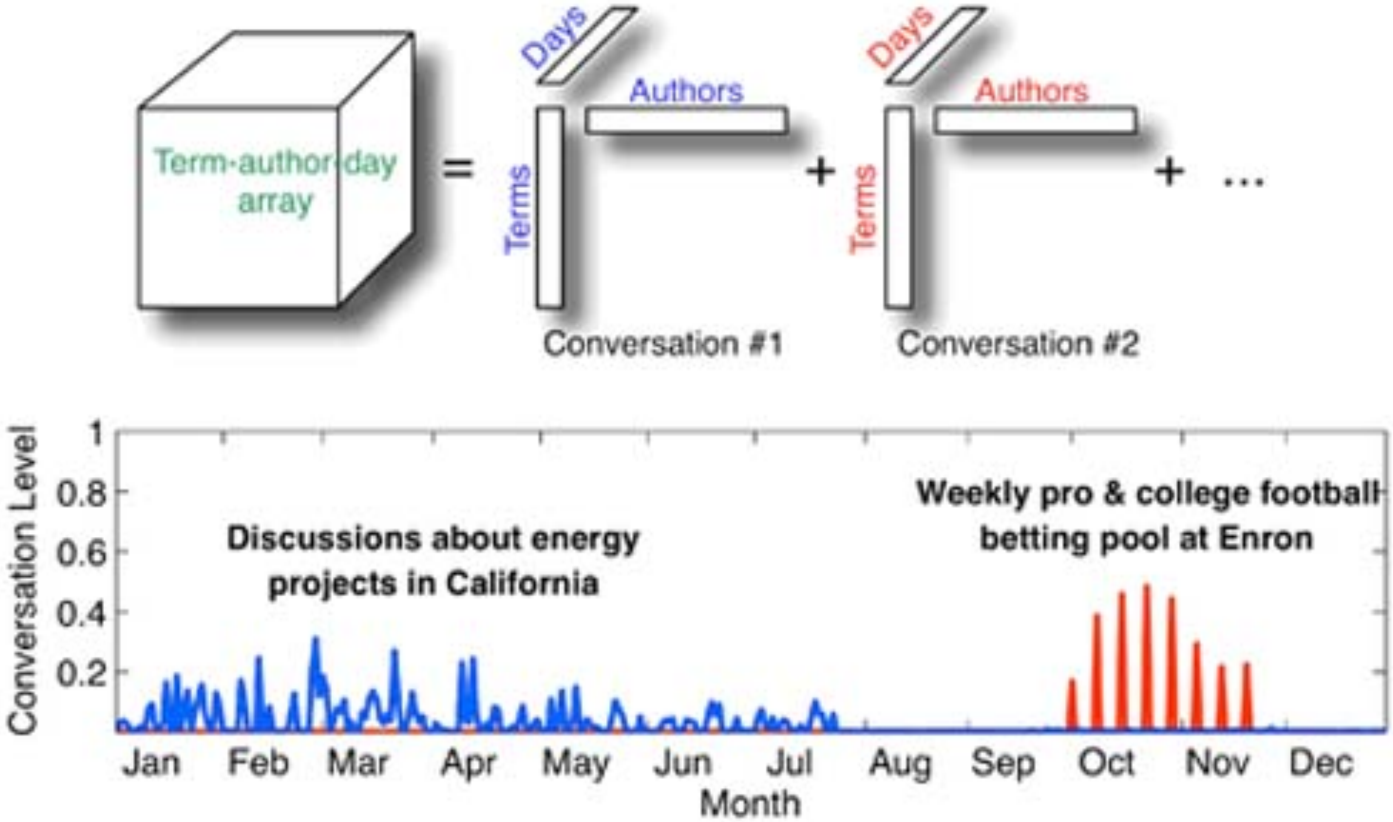
BUT best rank-*R* tensor cannot be obtained by truncation: different optimization problems for different *R*!

given tensor $\mathcal{T} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$, find rank-$R$ canonical tensor $\mathcal{A}_R \in \mathbb{R}^{I_1 \times \ldots \times I_N}$ that minimizes

$$f(\mathcal{A}_R) = \frac{1}{2} \|\mathcal{T} - \mathcal{A}_R\|_F^2.$$

tensor approximation applications

(1) “Discussion Tracking in Enron Email Using PARAFAC” by Bader, Berry and Browne (2008) (sparse, nonnegative)

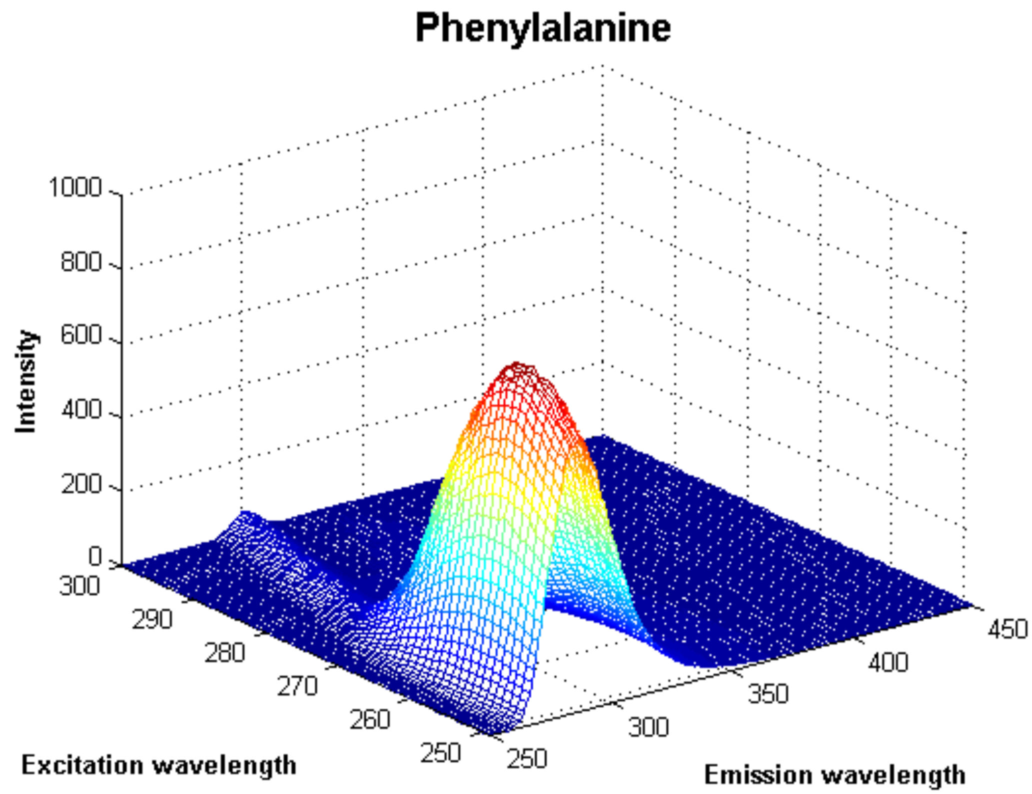


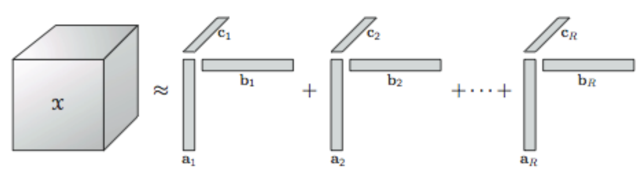
UNIVERSITY OF WATERLOO

tensor approximation applications

(2) chemometrics: analyze spectrofluorometer data (dense) (Bro et al., http://www.models.life.ku.dk/nwaydata1)

- 5 x 201 x 61 tensor: 5 samples (with different mixtures of three amino acids), 61 excitation wavelengths, 201 emission wavelengths
- goal: recover emission spectra of the three amino acids (to determine what was in each sample, and in which concentration)





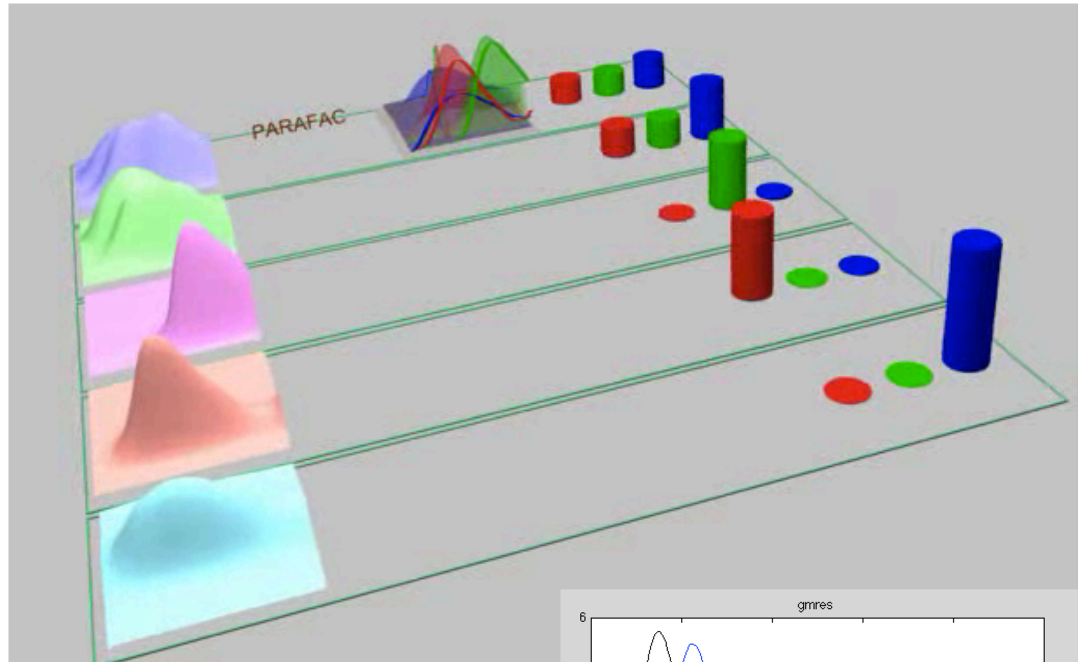
(from [1])

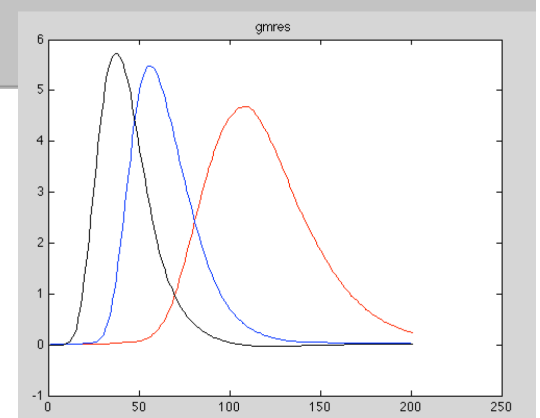
UNIVERSITY OF WATERLOO

tensor approximation applications

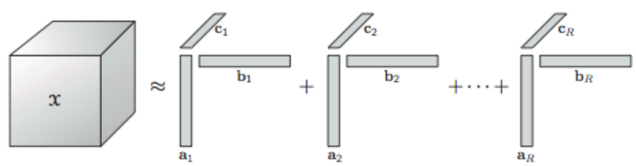
(2) chemometrics: analyze spectrofluorometer data (dense) (Bro et al.,

http://www.models.life.ku.dk/nwaydata1)





- 5 x 201 x 61 tensor: 5 samples (with different mixtures of three amino acids), 61 excitation wavelengths, 201 emission wavelengths
- goal: recover emission spectra of the three amino acids (to determine what was in each sample, and in which concentration)
- also: psychometrics, ..

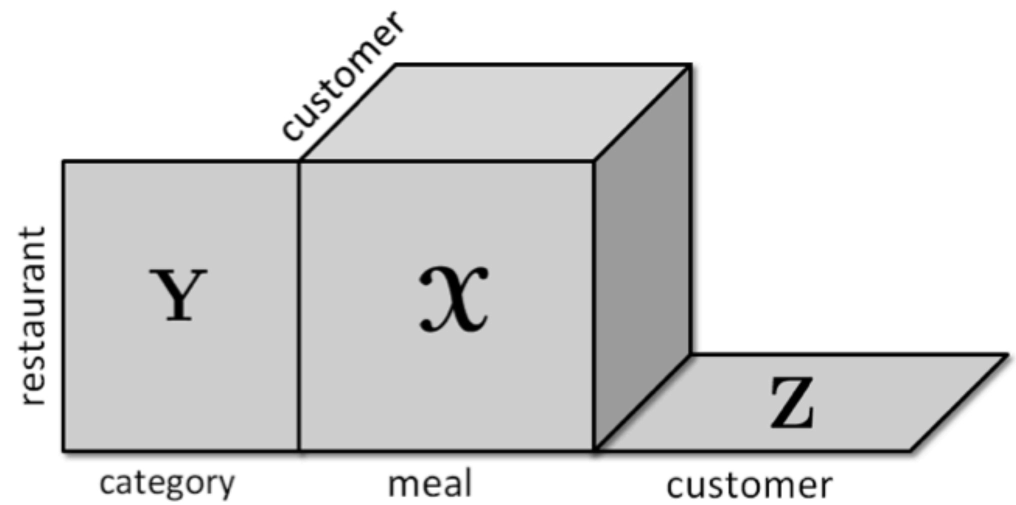


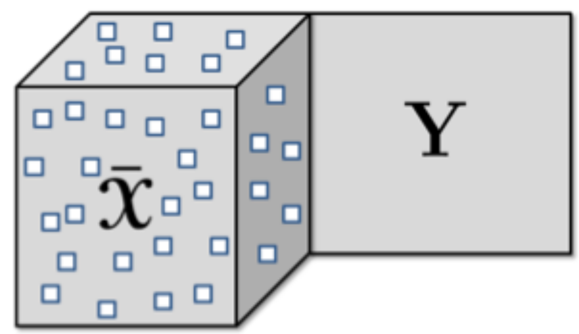
(from [1])

UNIVERSITY OF WATERLOO

tensor approximation applications

(3) “All-at-once Optimization for Coupled Matrix and Tensor Factorizations” by Acar, Kolda and Dunlavy (2011)





$$\left\| \mathcal{W} * \left(\mathcal{X} - [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}]\!] \right) \right\|^2 + \frac{1}{2} \left\| \mathbf{Y} - \mathbf{A}^{(n)} \mathbf{V}^{\mathsf{T}} \right\|^2$$

$$f(\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{V}) = \| \mathcal{X} - [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!] \|^2 + \left\| \mathbf{Y} - \mathbf{A}\mathbf{V}^{\mathsf{T}} \right\|^2$$

(2) ‘workhorse’ algorithm: alternating least squares (ALS)

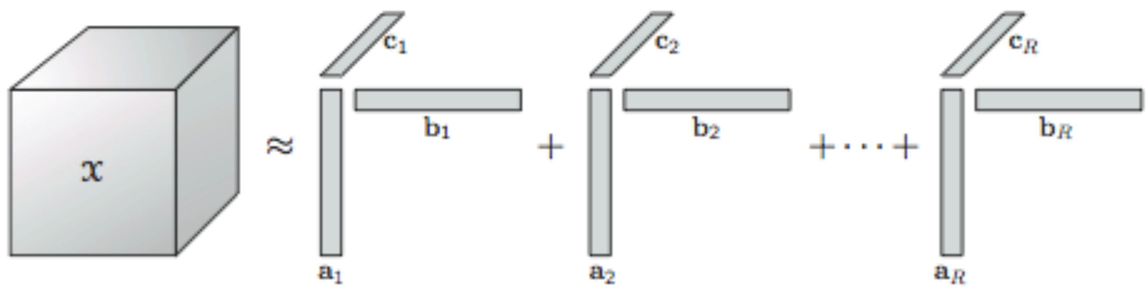
$$f(\mathcal{A}_R) = \frac{1}{2} \left\| \mathcal{T} - \sum_{r=1}^{R} a_r^{(1)} \circ a_r^{(2)} \circ a_r^{(3)} \right\|_F^2$$

(1) freeze all $a_r^{(2)}$, $a_r^{(3)}$, compute optimal $a_r^{(1)}$ via a least-squares solution (linear, overdetermined)

(2) freeze $a_r^{(1)}$, $a_r^{(3)}$, compute $a_r^{(2)}$

(3) freeze $a_r^{(1)}$, $a_r^{(2)}$, compute $a_r^{(3)}$

- repeat



(from [1])

alternating least squares (ALS)

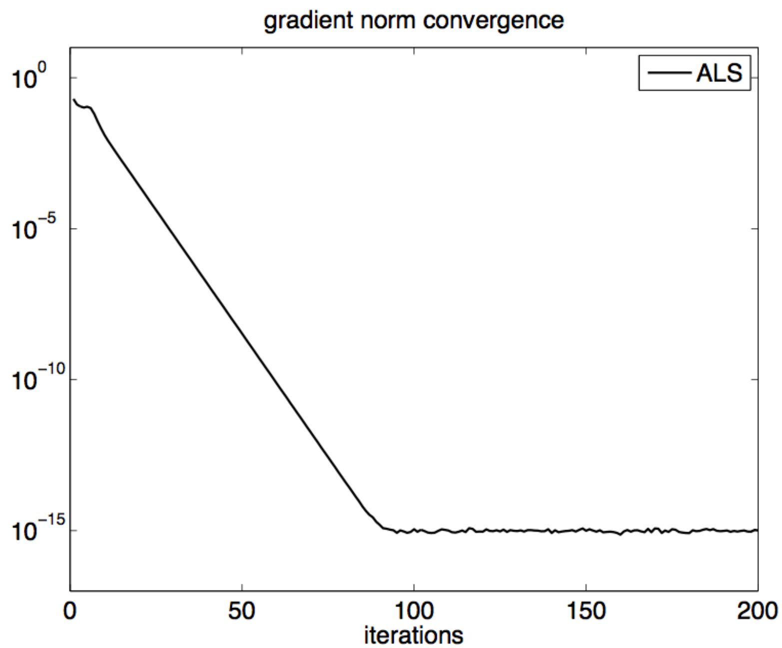
$$f(\mathcal{A}_R) = \frac{1}{2} \left\| \mathcal{T} - \sum_{r=1}^{R} a_r^{(1)} \circ a_r^{(2)} \circ a_r^{(3)} \right\|_F^2$$

- “simple iterative optimization method”
- ALS is block nonlinear Gauss-Seidel
- ALS is monotone
- ALS is sometimes fast, but can also be extremely slow (depending on problem and initial condition)

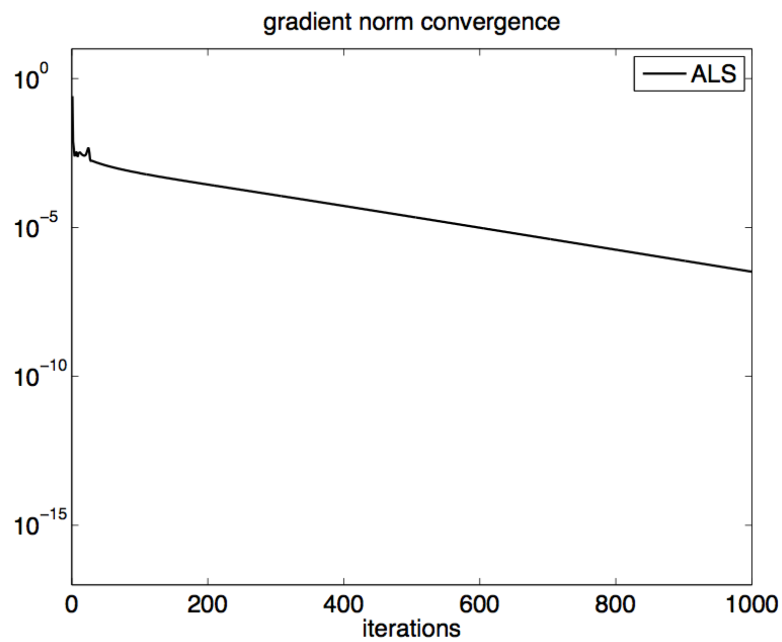
alternating least squares (ALS)

$$f(\mathcal{A}_R) = \frac{1}{2}\left\| \mathcal{T} - \sum_{r=1}^{R} a_r^{(1)} \circ a_r^{(2)} \circ a_r^{(3)} \right\|_F^2$$

fast case



slow case

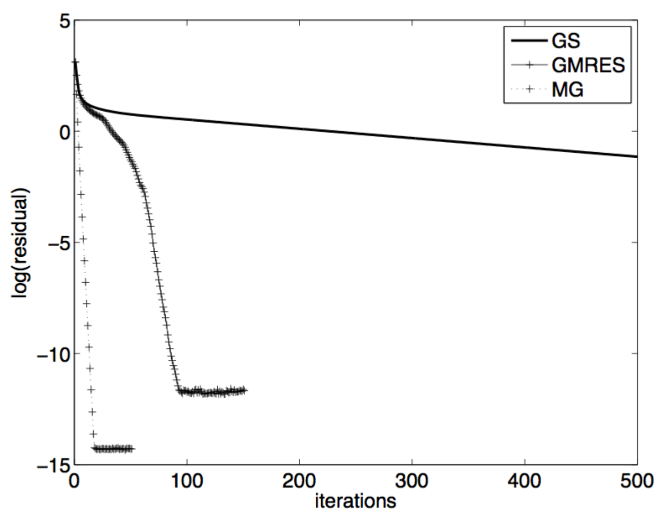


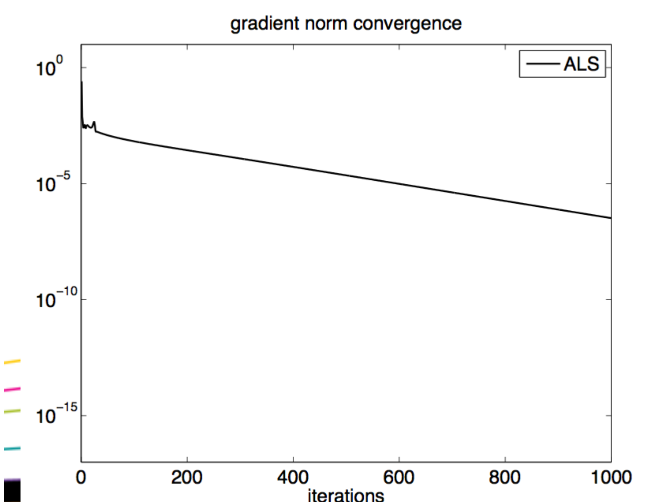
(we used Matlab with Tensor Toolbox (Bader and Kolda) and Poblano Toolbox (Dunlavy et al.) for all computations)

UNIVERSITY OF WATERLOO

convergence acceleration for ALS (nonlinear optimization)

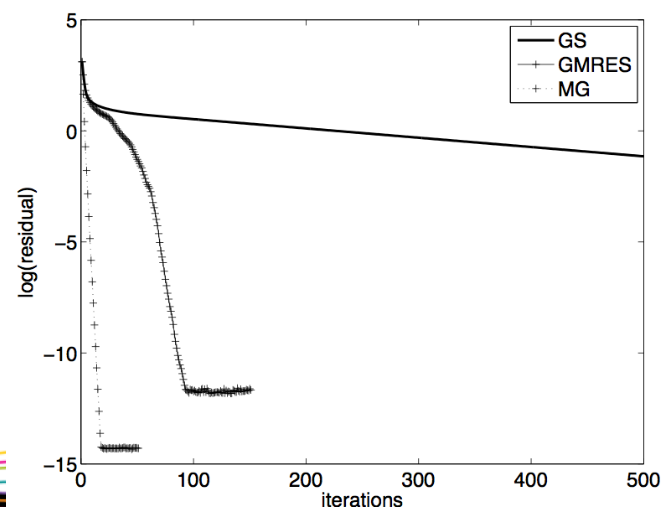
convergence acceleration for linear systems	convergence acceleration for nonlinear optimization		
P-CG	NCG, P?		
P-GMRES	?		
MG	? (MG/OPT)		

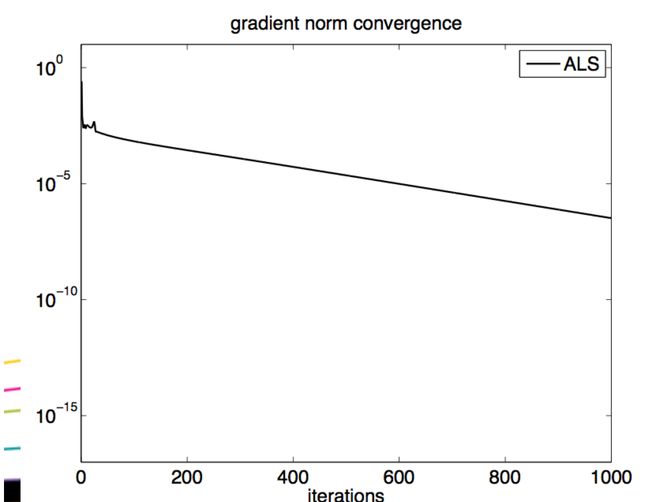




convergence acceleration for ALS (nonlinear optimization)

convergence acceleration for linear systems	convergence acceleration for nonlinear optimization	convergence acceleration for nonlinear systems	
P-CG	NCG, P?	NCG, P?	
P-GMRES	?	P-NGMRES (Washio and Oosterlee, Anderson)	
MG	? (MG/OPT)	FAS	

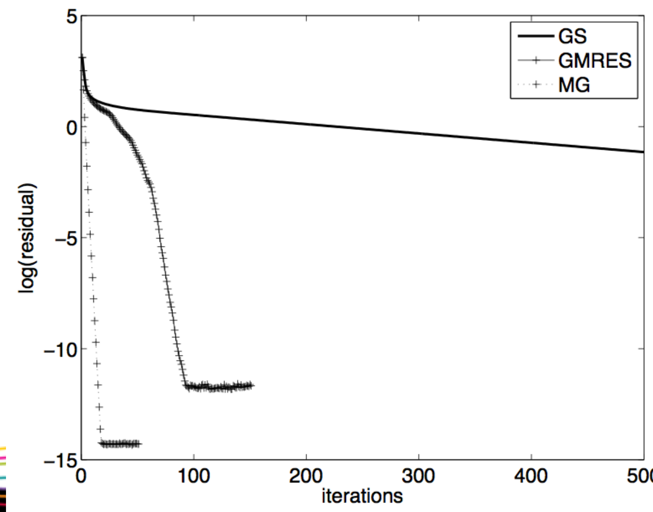


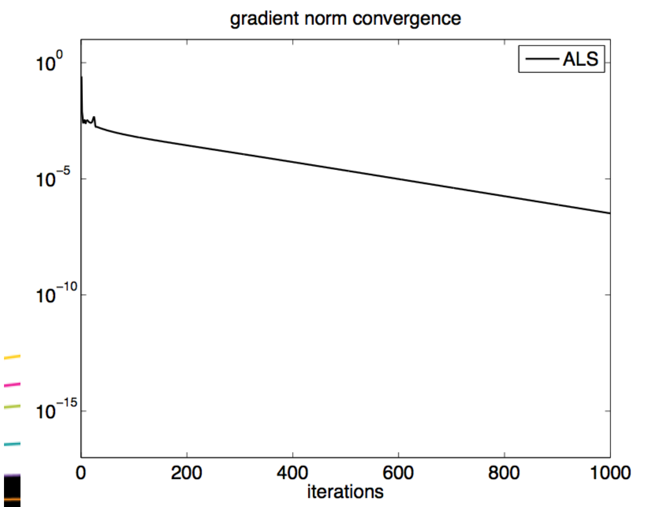


UNIVERSITY OF WATERLOO

convergence acceleration for ALS (nonlinear optimization)

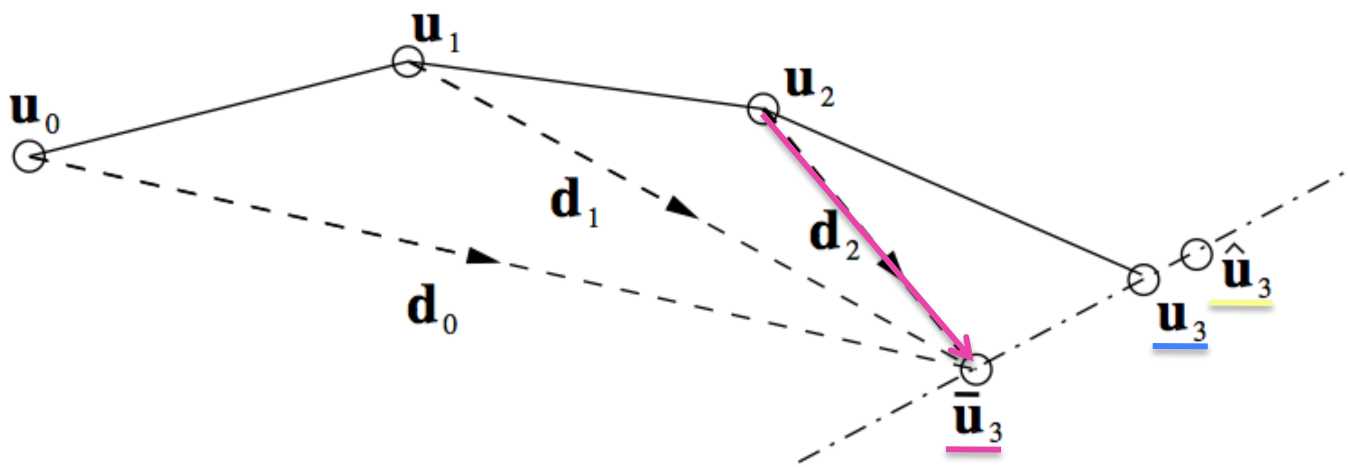
convergence acceleration for linear systems	convergence acceleration for nonlinear optimization	convergence acceleration for nonlinear systems	convergence acceleration for nonlinear optimization
P-CG	NCG, P?	NCG, P?	P-NCG
P-GMRES	?	P-NGMRES (Washio and Oosterlee, Anderson)	P-NGMRES for optimization
MG	?	FAS	adaptive AMG-FAS for optimization





UNIVERSITY OF WATERLOO

(3) nonlinear GMRES optimization method (N-GMRES)



Algorithm 1: N-GMRES optimization algorithm (window size w)

Input: w initial iterates $\mathbf{u}_0, \ldots, \mathbf{u}_{w-1}$.

$i = w - 1$

repeat

STEP I: *(generate preliminary iterate by one-step update process $M(.)$)* (ALS)

$\bar{\mathbf{u}}_{i+1} = M(\mathbf{u}_i)$

STEP II: *(generate accelerated iterate by nonlinear GMRES step)*

$\hat{\mathbf{u}}_{i+1} = \text{gmres}(\mathbf{u}_{i-w+1}, \ldots, \mathbf{u}_i; \bar{\mathbf{u}}_{i+1})$

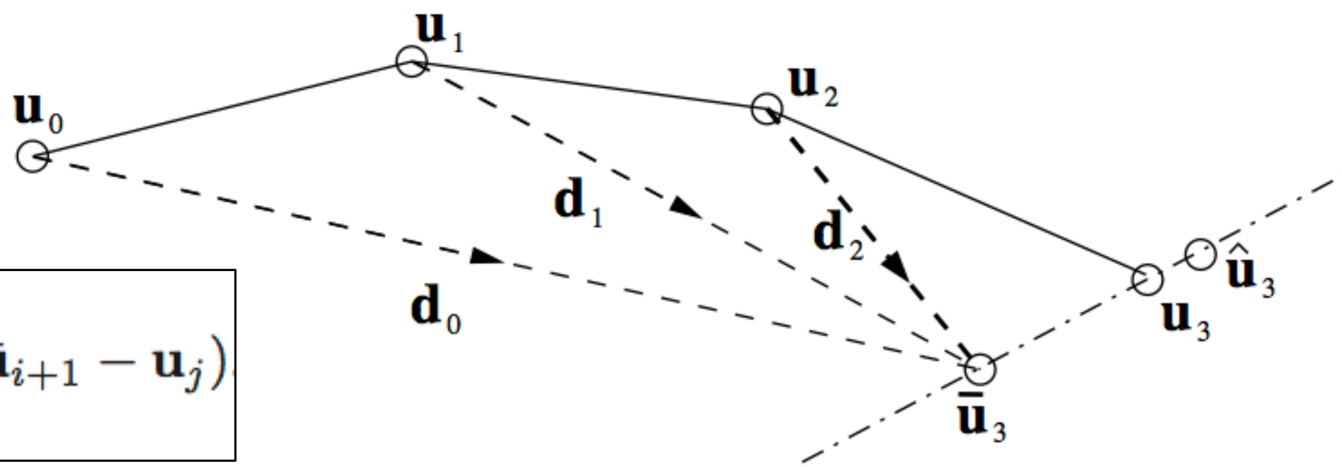
STEP III: *(generate new iterate by line search process)* (Moré-Thuente line search, satisfies Wolfe conditions)

$\mathbf{u}_{i+1} = \text{linesearch}(\bar{\mathbf{u}}_{i+1} + \beta(\hat{\mathbf{u}}_{i+1} - \bar{\mathbf{u}}_{i+1}))$

$i = i + 1$

until *convergence criterion satisfied*

step II: N-GMRES acceleration: $\nabla f(\mathcal{A}_R) = \mathbf{g}(\mathcal{A}_R) = 0$



$$\boxed{\hat{\mathbf{u}}_{i+1} = \bar{\mathbf{u}}_{i+1} + \sum_{j=0}^{i} \alpha_j \left(\bar{\mathbf{u}}_{i+1} - \mathbf{u}_j\right)}$$

$$\mathbf{g}(\hat{\mathbf{u}}_{i+1}) \approx \mathbf{g}(\bar{\mathbf{u}}_{i+1}) + \sum_{j=0}^{i} \left.\frac{\partial \mathbf{g}}{\partial \mathbf{u}}\right|_{\bar{\mathbf{u}}_{i+1}} \alpha_j \left(\bar{\mathbf{u}}_{i+1} - \mathbf{u}_j\right)$$

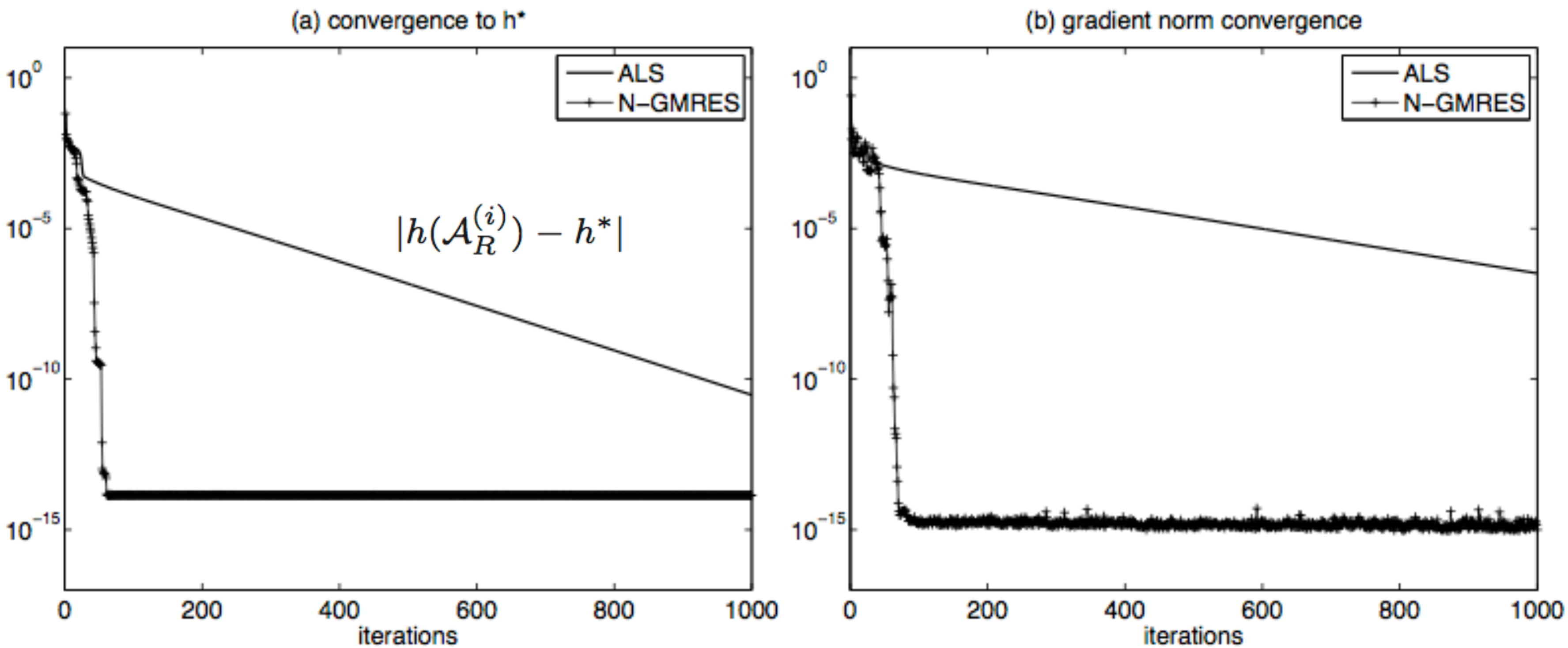
$$\approx \mathbf{g}(\bar{\mathbf{u}}_{i+1}) + \sum_{j=0}^{i} \alpha_j \left(\mathbf{g}(\bar{\mathbf{u}}_{i+1}) - \mathbf{g}(\mathbf{u}_j)\right)$$

find coefficients $(\alpha_0, \ldots, \alpha_i)$ that minimize

$$\left\|\mathbf{g}(\bar{\mathbf{u}}_{i+1}) + \sum_{j=0}^{i} \alpha_j \left(\mathbf{g}(\bar{\mathbf{u}}_{i+1}) - \mathbf{g}(\mathbf{u}_j)\right)\right\|_2.$$

numerical results for ALS-preconditioned N-GMRES applied to tensor problem

- dense test problem (from Tomasi and Bro; Acar et al.):
  - random rank-R tensor modified to obtain specific column collinearity, with added noise



$$h(\mathcal{A}_R^{(i)}) = \frac{\|\mathcal{T} - \mathcal{A}_R^{(i)}\|_F}{\|\mathcal{T}\|_F}$$

nonlinearly preconditioned N-GMRES for nonlinear optimization

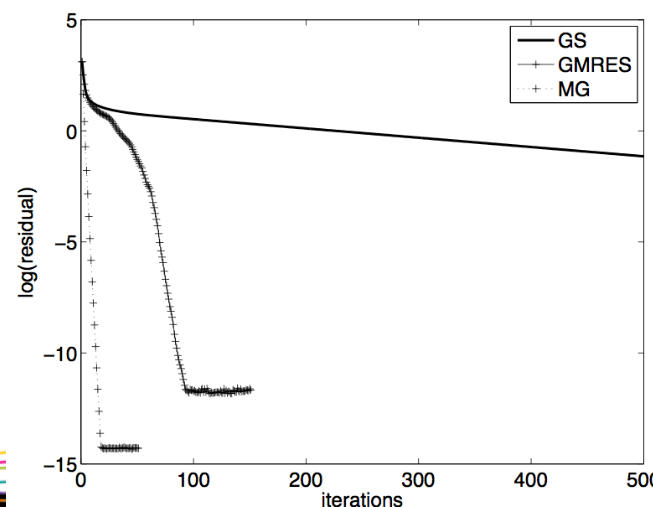
- works well for canonical tensor decomposition (N-GMRES accelerates ALS)
- also promising as a general way to accelerate 'simple' optimization methods

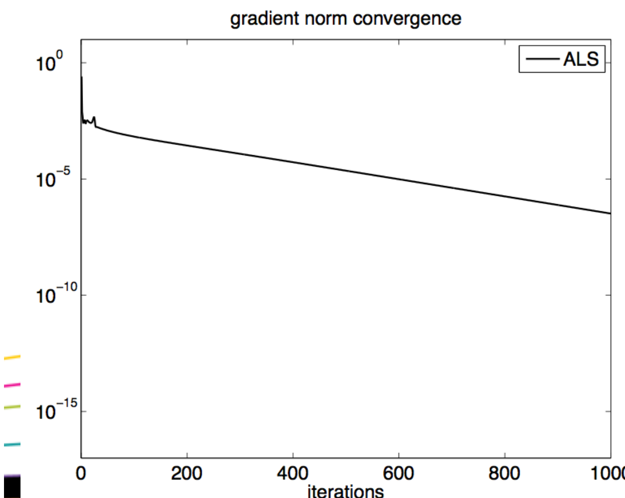
- Hans De Sterck, *'A Nonlinear GMRES Optimization Algorithm for Canonical Tensor Decomposition'*, SIAM J. Sci. Comp., 2012
- Hans De Sterck, *'Steepest Descent Preconditioning for Nonlinear GMRES Optimization'*, Numer. Linear Algebra Appl., 2012

UNIVERSITY OF WATERLOO

convergence acceleration for ALS (nonlinear optimization)

convergence acceleration for linear systems	convergence acceleration for nonlinear optimization	convergence acceleration for nonlinear systems	convergence acceleration for nonlinear optimization
P-CG	NCG, P?	NCG, P?	P-NCG
P-GMRES	?	P-NGMRES (Washio and Oosterlee, Anderson)	P-NGMRES for optimization
MG	?	FAS	adaptive AMG-FAS for optimization



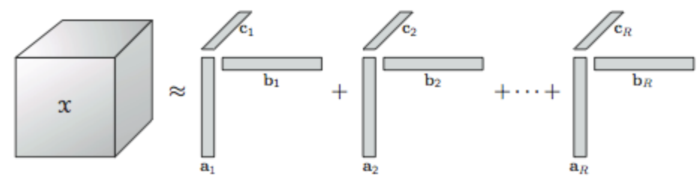


UNIVERSITY OF WATERLOO

(4) adaptive AMG-FAS for accelerating ALS

- with Killian Miller
- builds on: adaptive AMG for SVD (Hans De Sterck, 'A Self-learning Algebraic Multigrid Method for Extremal Singular Triplets and Eigenpairs', SIAM J. Sci. Comput. 34, 2012)

$$A = U\,\Sigma\,V^t = \sigma_1\,u_1\,v_1^T + \ldots + \sigma_n\,u_n\,v_n^T$$



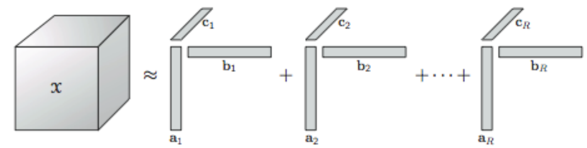
- SVD case: separate coarsening and interpolation for *u* and *v* variables

$$A \in \mathbb{R}^{m\times n} \qquad \begin{aligned} A\,v &= \sigma\,u, & u &= P\,u_c, & P^t\,A\,Q\,v_c &= \sigma\,P^t\,B\,P\,u_c,\\ A^t\,u &= \sigma\,v. & v &= Q\,v_c, & Q^t\,A^t\,P\,u_c &= \sigma\,Q^t\,C\,Q\,v_c, \end{aligned}$$

UNIVERSITY OF WATERLOO

adaptive AMG-FAS for accelerating ALS

$$A = U\,\Sigma\,V^t = \sigma_1\,u_1\,v_1^T + \ldots + \sigma_n\,u_n\,v_n^T$$



- SVD case: separate coarsening and interpolation for *u* and *v* variables

$$A \in \mathbb{R}^{m\times n} \qquad \begin{aligned} A\,v &= \sigma\,u, \\ A^t\,u &= \sigma\,v. \end{aligned} \qquad \begin{aligned} u &= P\,u_c, \\ v &= Q\,v_c, \end{aligned} \qquad \begin{aligned} P^t\,A\,Q\,v_c &= \sigma\,P^t\,B\,P\,u_c, \\ Q^t\,A^t\,P\,u_c &= \sigma\,Q^t\,C\,Q\,v_c, \end{aligned}$$

(note: collective interpolation of *u* singular vectors by *P*, ...)

- we don't know the nature of the slow-to-converge components in advance: determine *P*, *Q*, *R*, ... adaptively using Bootstrap AMG in a 'multiplicative' setup phase (Brandt/Brannick/Kahl/Livshits and Kushnir/Galun/Brandt)

- use FAS in an 'additive' solve phase (Brandt/McCormick/Ruge and Borzi/Borzi)

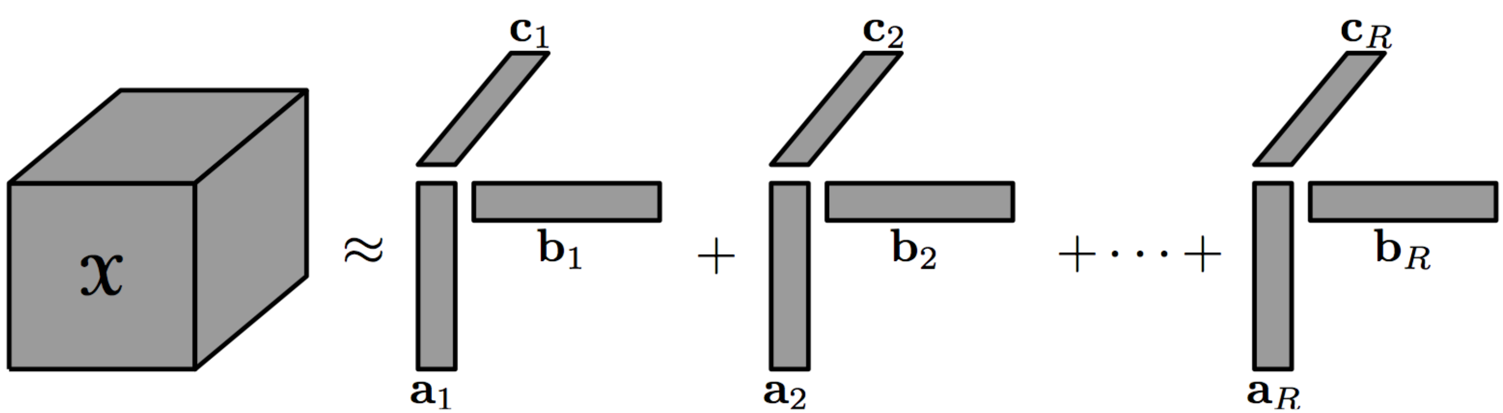
canonical tensor decomposition: notation

- Decompose a tensor as a sum of R rank-one terms
- For a third-order tensor:

$$\mathcal{X} \approx \sum_{r=1}^{R} \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$$

Factor matrices:

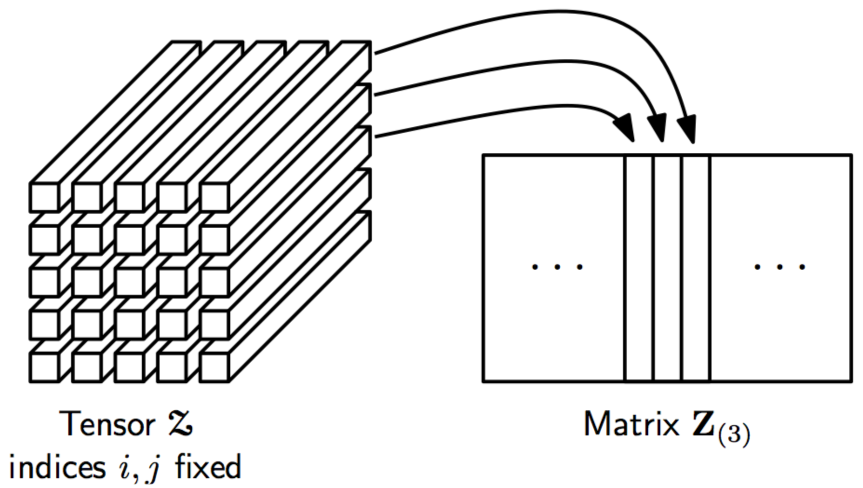
$$\mathbf{A} = [\mathbf{a}_1, \ldots, \mathbf{a}_R], \quad \mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_R], \quad \mathbf{C} = [\mathbf{c}_1, \ldots, \mathbf{c}_R]$$



some tensor machinery...

Mode-n Matricization: $\mathcal{Z} \to \mathbf{Z}_{(n)}$

- Transforms a tensor into a matrix by reordering tensor elements



Khatri-Rao product

- For $\mathbf{A} \in \mathbb{R}^{I \times J}$ and $\mathbf{B} \in \mathbb{R}^{K \times J}$

$$\mathbf{A} \odot \mathbf{B} = [\mathbf{a}_1 \otimes \mathbf{b}_1 \quad \mathbf{a}_2 \otimes \mathbf{b}_2 \quad \cdots \quad \mathbf{a}_J \otimes \mathbf{b}_J] \in \mathbb{R}^{IK \times J}$$

UNIVERSITY OF WATERLOO

(4a) multiplicative phase: coarse equations

$$\text{minimize} \quad f(\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}) := \frac{1}{2}\left\| \mathfrak{Z} - [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}]\!] \right\|^2$$

- Fine-level equations are the gradient equations of f:

$$\mathbf{Z}_{(n)}\left(\mathbf{A}^{(N)} \odot \cdots \odot \mathbf{A}^{(n+1)} \odot \mathbf{A}^{(n-1)} \odot \cdots \odot \mathbf{A}^{(1)}\right) = \mathbf{A}^{(n)}\mathbf{\Gamma}^{(n)}$$

- Assume that $\mathrm{range}(\mathbf{P}^{(n)})$ approximately contains $\mathbf{A}^{(n)}$, i.e.,

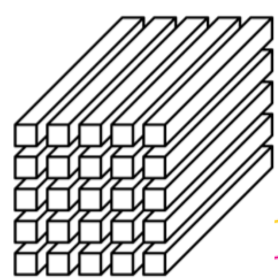
$$\mathbf{A}^{(n)} \approx \mathbf{P}^{(n)}\mathbf{A}_c^{(n)} \quad \text{for } n = 1, \ldots, N$$

- Coarse-level equations:

$$\mathbf{Z}^c_{(n)}\left(\mathbf{A}_c^{(N)} \odot \cdots \odot \mathbf{A}_c^{(n+1)} \odot \mathbf{A}_c^{(n-1)} \odot \cdots \odot \mathbf{A}_c^{(1)}\right) = \mathbf{B}^{(n)}\mathbf{A}_c^{(n)}\mathbf{\Gamma}_c^{(n)}$$

$$\mathbf{B}^{(n)} = \mathbf{P}^{(n)^{\mathrm{T}}}\mathbf{P}^{(n)} \ (\mathrm{SPD})$$

$$\mathfrak{Z}^c = \mathfrak{Z} \times_1 \mathbf{P}^{(1)^{\mathrm{T}}} \times_2 \mathbf{P}^{(2)^{\mathrm{T}}} \cdots \times_N \mathbf{P}^{(N)^{\mathrm{T}}}$$



Tensor $\mathfrak{Z}$

multiplicative phase: coarse equations

- Compute Cholesky factorization: $\mathbf{B}^{(n)} = \mathbf{L}^{(n)}\mathbf{L}^{(n)\mathrm{T}}$
- Let $\hat{\mathbf{A}}_c^{(n)} = \mathbf{L}^{(n)\mathrm{T}}\mathbf{A}_c^{(n)}$ for $n = 1, \ldots, N$
- Transformed coarse-level equations:

$$\boxed{\hat{\mathbf{Z}}^c_{(n)}\left(\hat{\mathbf{A}}_c^{(N)} \odot \cdots \odot \hat{\mathbf{A}}_c^{(n+1)} \odot \hat{\mathbf{A}}_c^{(n-1)} \odot \cdots \odot \hat{\mathbf{A}}_c^{(1)}\right) = \hat{\mathbf{A}}_c^{(n)}\hat{\mathbf{\Gamma}}_c^{(n)}}$$

$$\hat{\mathfrak{Z}}^c = \mathfrak{Z} \times_1 \hat{\mathbf{P}}^{(1)\mathrm{T}} \times_2 \hat{\mathbf{P}}^{(2)\mathrm{T}} \cdots \times_N \hat{\mathbf{P}}^{(N)\mathrm{T}}$$

$$\hat{\mathbf{P}}^{(n)} = \mathbf{P}^{(n)}\mathbf{L}^{(n)-\mathrm{T}}, \quad \hat{\mathbf{R}}^{(n)} = \hat{\mathbf{P}}^{(n)\mathrm{T}}$$

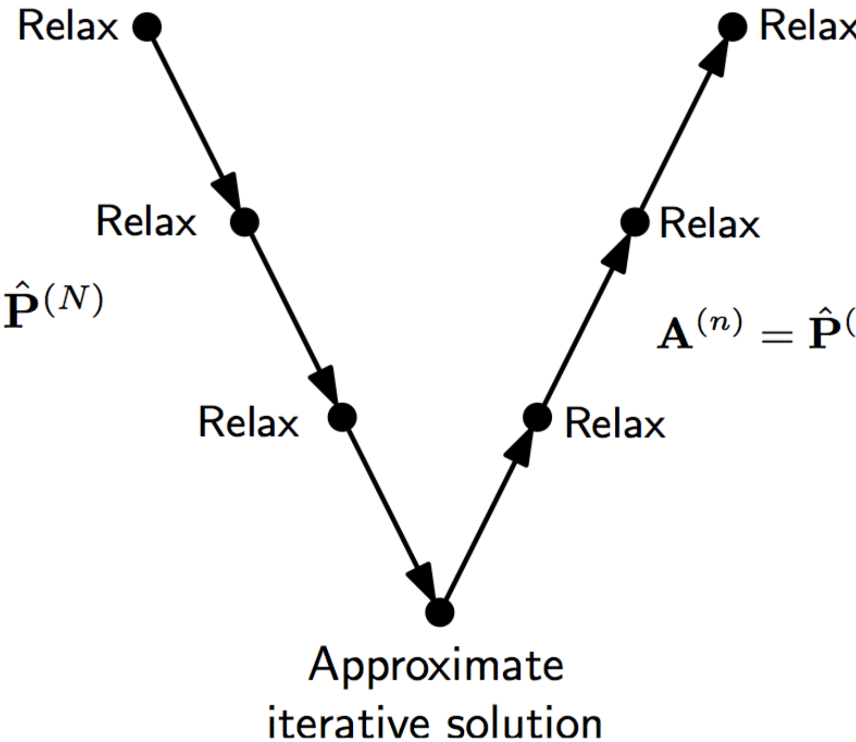
- Transformed coarse-level equations are gradient equations of

$$\hat{f}_c(\hat{\mathbf{A}}_c^{(1)}, \ldots, \hat{\mathbf{A}}_c^{(N)}) := \frac{1}{2}\left\|\hat{\mathfrak{Z}}^c - [\![\hat{\mathbf{A}}_c^{(1)}, \ldots, \hat{\mathbf{A}}_c^{(N)}]\!]\right\|^2$$

UNIVERSITY OF WATERLOO

multiplicative phase: bootstrap V-cycle

- Build $\hat{\mathbf{P}}^{(1)}, \ldots, \hat{\mathbf{P}}^{(N)}$
- Build $\hat{\boldsymbol{\mathfrak{z}}}^{c}$

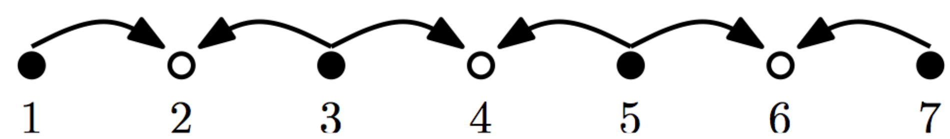


$$\mathbf{A}^{(n)} = \hat{\mathbf{P}}^{(n)} \hat{\mathbf{A}}_c^{(n)}, \; n = 1, \ldots, N$$

multiplicative phase: geometric coarsening

- Partition $\Omega_n = \{1, \dots, I_n\}$ into coarse points $\mathcal{C}_n$ and fine points $\mathcal{F}_n$
- Standard geometric coarsening:

$$\mathcal{C}_n = \{\text{odd points in } \Omega_n\}, \quad \mathcal{F}_n = \{\text{even points in } \Omega_n\}$$



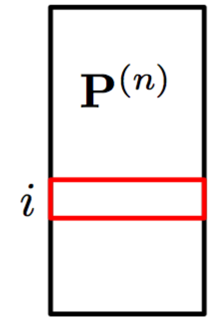
$$p_{ij}^{(n)} = \begin{cases} w_{ij}^{(n)}, & i \in \mathcal{F}_n,\ j \in \{i/2, i/2+1\} \\ 1, & i \in \mathcal{C}_n,\ j = (i+1)/2 \\ 0, & \text{otherwise,} \end{cases} \qquad \mathbf{P}^{(n)} = \begin{bmatrix} 1 & & & \\ * & * & & \\ & 1 & & \\ & * & * & \\ & & 1 & \\ & & * & * \\ & & & 1 \end{bmatrix}$$

multiplicative phase: coarse equations

- Determine $\mathbf{P}^{(n)}$ via weighted least squares fitting of the test/boot factors, *injected* to the coarse level
- Let $\mathbf{U} = [\mathbf{A}_{t,1}^{(n)}, \ldots, \mathbf{A}_{t,n_t}^{(n)} \mid \mathbf{A}_b^{(n)}]$
- For each point $i \in \mathcal{F}_n$

$$\mu_k u_{ik} = \sum_{j \in \mathcal{C}_n^i} \mu_k w_{ij}^{(n)} (\mathbf{u}_{k,c})_j$$

for $k = 1, \ldots, n_f = R(n_t + 1)$.



(one equation per test or boot factor)

- Choose n_t so system is overdetermined
- Weights μ_k proportional to reciprocal of gradient norm

(4b) additive phase: FAS

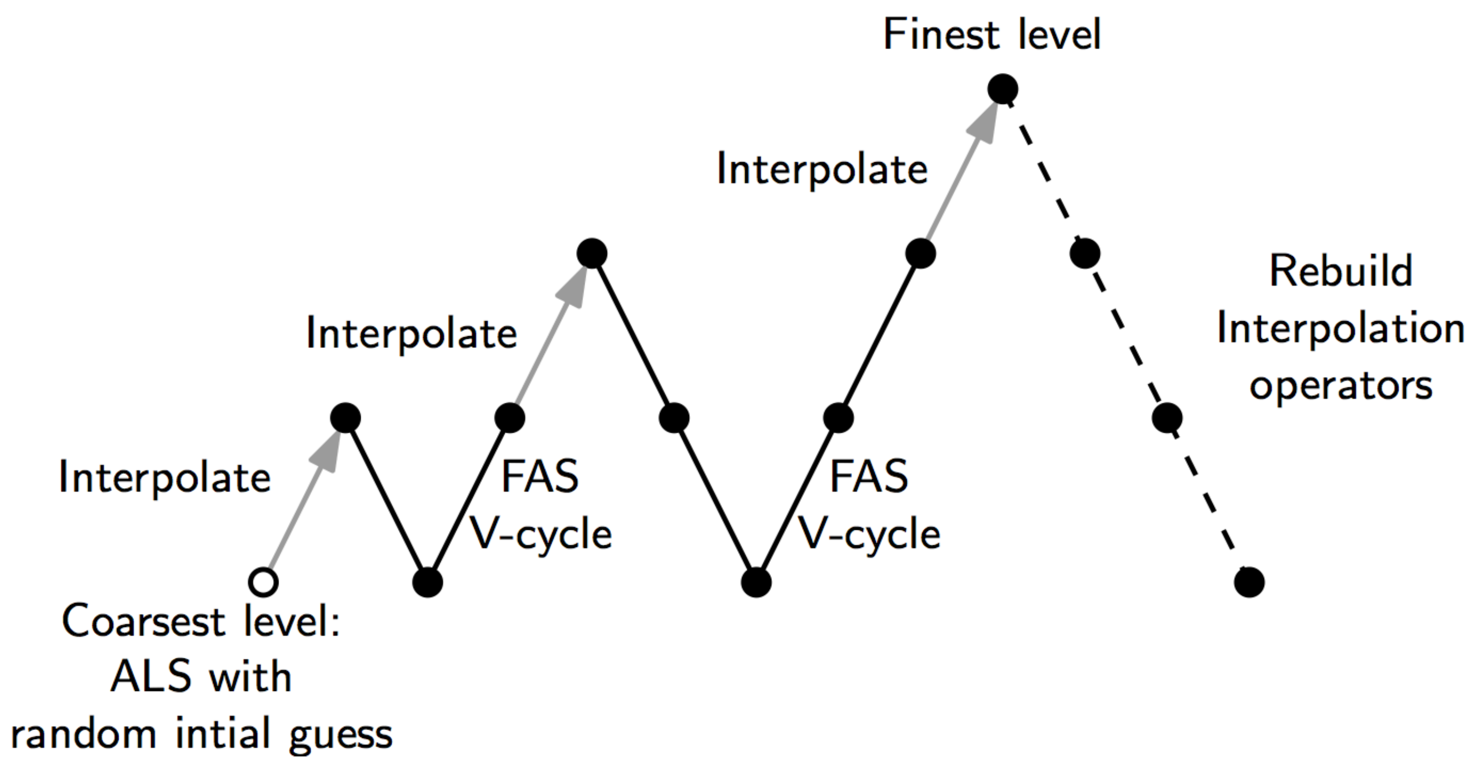
- we apply the FAS to the set of nonlinear equations

$$\mathbf{Z}_{(n)}\left(\mathbf{A}^{(N)} \odot \cdots \odot \mathbf{A}^{(n+1)} \odot \mathbf{A}^{(n-1)} \odot \cdots \odot \mathbf{A}^{(1)}\right) = \mathbf{A}^{(n)}\mathbf{\Gamma}^{(n)}$$

  using the coarse operators and interpolation matrices computed in the setup phase

- we apply block nonlinear Gauss-Seidel as the smoother on all levels

additive phase: FMG FAS

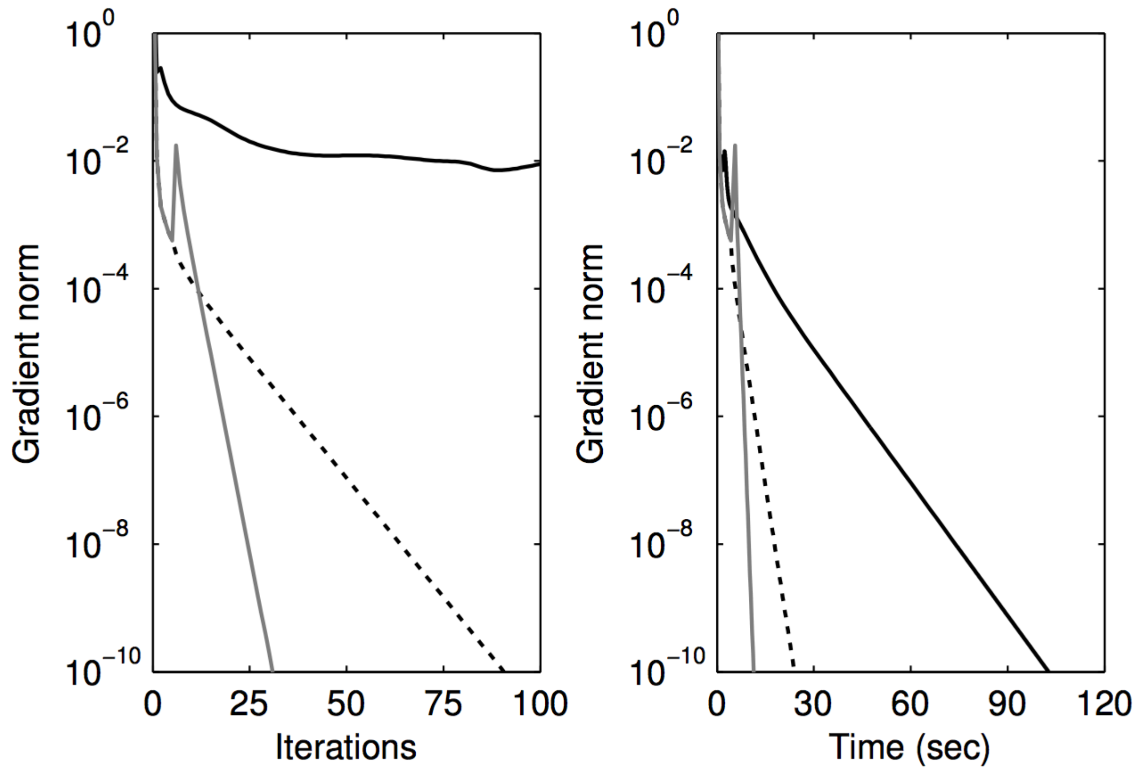


(4c) adaptive AMG numerical results

Finite Difference Laplacian Tensor

$$(N = 4,\ s = 20,\ R = 6)$$

Black: ALS, Dashed: Multilevel, Gray: Multilevel+FMG



UNIVERSITY OF WATERLOO

adaptive AMG numerical results

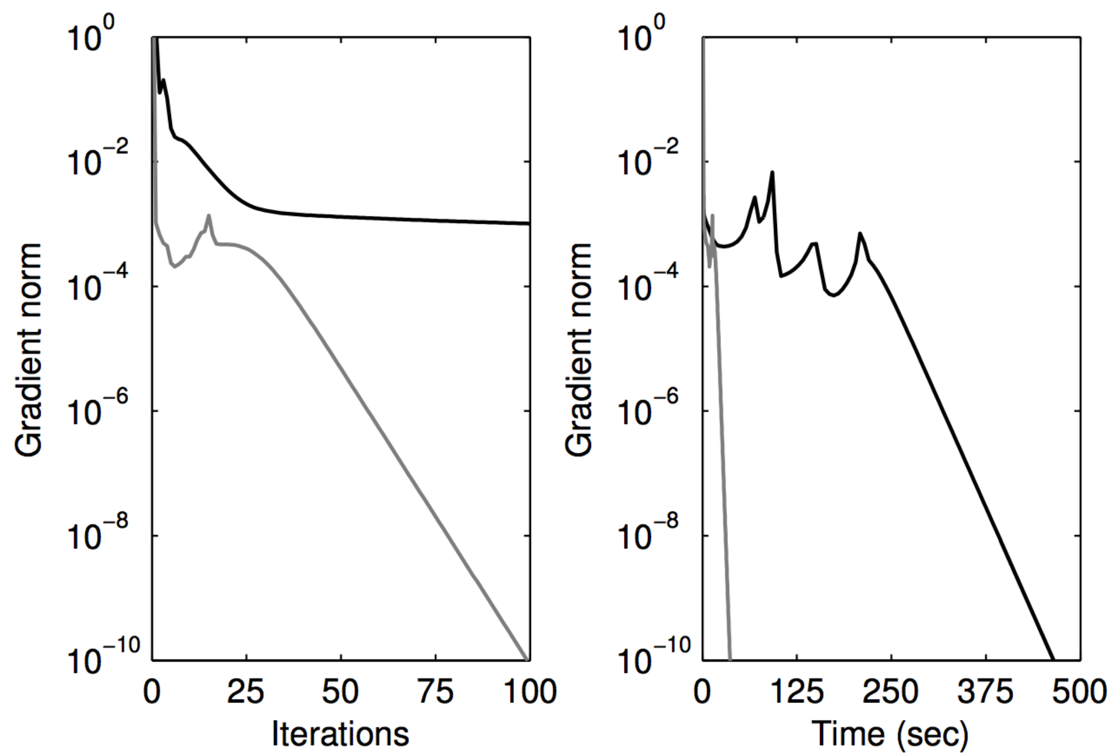
IJK Tensor

Third order tensor of size $s \times s \times s$ with elements

$$z_{ijk} = \left(i^2 + j^2 + k^2\right)^{-1/2} \quad \text{for } i, j, k = 1, \ldots, s.$$

$s = 100,\ R = 5$

Black: ALS, Gray: Multilevel+FMG



adaptive AMG numerical results

Third order tensor of size $s \times s \times s$ with elements

$$z_{ijk} = (i^2 + j^2 + k^2)^{-1/2} \quad \text{for } i, j, k = 1, \ldots, s.$$

	ALS			Multilevel + FMG			
problem parameters	it	time	ns	it	time	spd	ns
$s = 50,\ R = 2$	161	0.7	10	7	2.2	0.3	10
$s = 50,\ R = 3$	2435	11.8	10	15	2.7	4.7	10
$s = 50,\ R = 4$	4838	26.1	5	97	10.1	4.4	7
$s = 50,\ R = 5$				301	30.0		3
$s = 100,\ R = 2$	253	10.7	10	7	7.9	1.4	10
$s = 100,\ R = 3$	1695	80.2	9	9	8.1	10.1	10
$s = 100,\ R = 4$	3836	202.2	6	82	27.5	14.1	9
$s = 100,\ R = 5$	7854	455.2	2	192	62.0	9.0	4
$s = 200,\ R = 2$	274	90.3	10	7	48.8	1.9	10
$s = 200,\ R = 3$	1830	682.3	10	12	61.7	11.2	10
$s = 200,\ R = 4$	2998	1249.5	8	79	178.0	11.6	9
$s = 200,\ R = 5$	5686	2611.4	3	220	440.7	2.6	4

UNIVERSITY OF WATERLOO

adaptive AMG numerical results

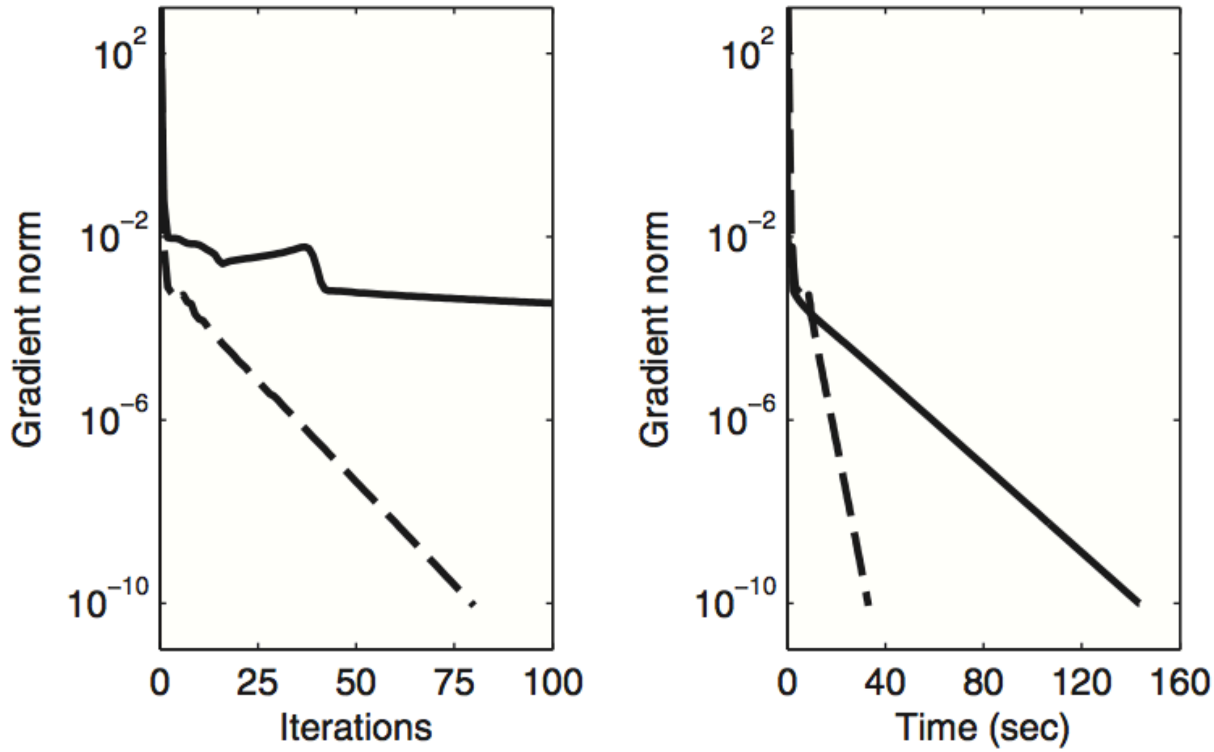


FIG. 6.6. *Random data problem. Convergence plot for test 5 in Table 6.5* ($s = 100$, $R = 5$). *The solid black line is ALS and the dashed line is the multilevel method without FMG.*

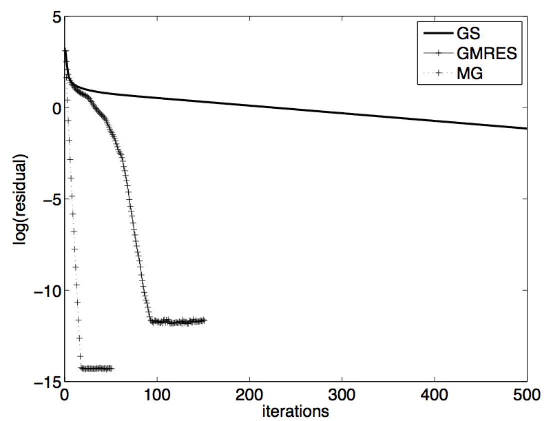
UNIVERSITY OF
WATERLOO

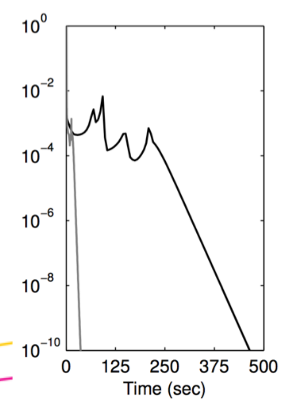
conclusions

convergence acceleration for linear systems	convergence acceleration for nonlinear optimization	convergence acceleration for nonlinear systems	convergence acceleration for nonlinear optimization
P-CG	NCG, P?	NCG, P?	P-NCG
P-GMRES	?	P-NGMRES (Washio and Oosterlee, Anderson)	P-NGMRES for optimization
MG	?	FAS	adaptive AMG-FAS for optimization

- Hans De Sterck, 'A Self-learning Algebraic Multigrid Method for Extremal Singular Triplets and Eigenpairs', SIAM J. Sci. Comput. 34, 2012

- Hans De Sterck and Killian Miller, 'An adaptive algebraic multigrid algorithm for low-rank canonical tensor decomposition', SIAM J. Sci. Comput. 35, 2013

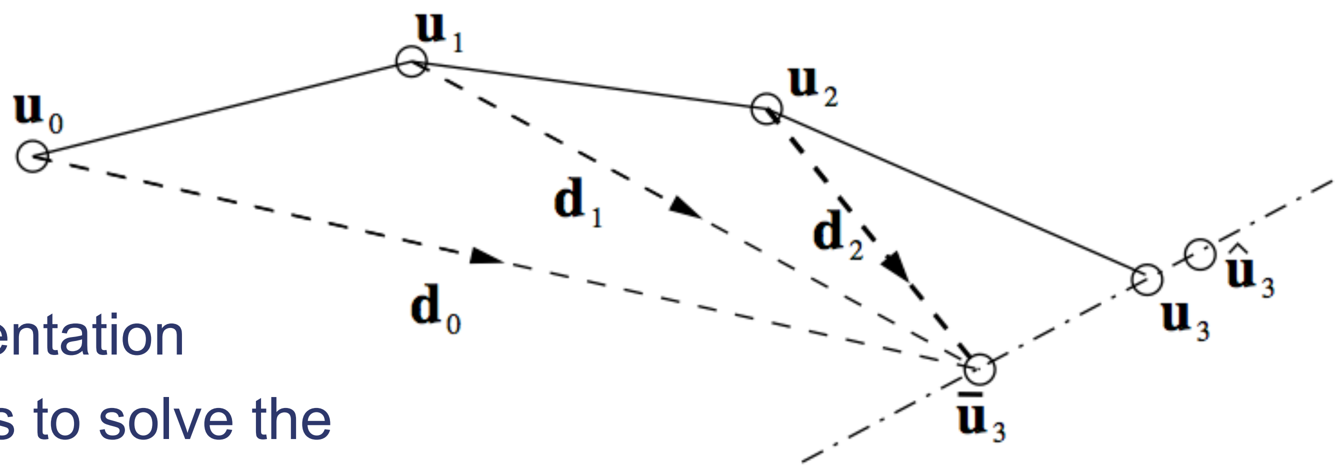




UNIVERSITY OF WATERLOO

thank you

step II: N-GMRES acceleration: $\nabla f(\mathcal{A}_R) = \mathbf{g}(\mathcal{A}_R) = 0$



- windowed implementation
- several possibilities to solve the small least-squares problems:
  - normal equations (with reuse of scalar products; cost *2nw*) (Washio and Oosterlee, 1997)
  - QR with factor updating (Walker and Ni, 2011)
  - SVD and rank-revealing QR (Fang and Saad, 2009)

$$\hat{\mathbf{u}}_{i+1} = \bar{\mathbf{u}}_{i+1} + \sum_{j=0}^{i} \alpha_j \left(\bar{\mathbf{u}}_{i+1} - \mathbf{u}_j\right)$$

find coefficients $(\alpha_0, \ldots, \alpha_i)$ that minimize

$$\left\|\mathbf{g}(\bar{\mathbf{u}}_{i+1}) + \sum_{j=0}^{i} \alpha_j \left(\mathbf{g}(\bar{\mathbf{u}}_{i+1}) - \mathbf{g}(\mathbf{u}_j)\right)\right\|_2.$$