# a residual-minimizing nonlinear optimization method applied to tensor approximation

UNIVERSITY OF WATERLOO

uwaterloo.ca

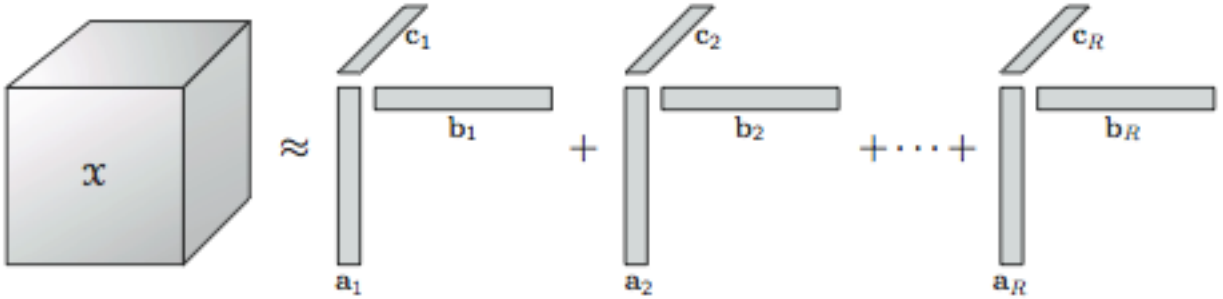
hans de sterck

sabbatical visitor, mpi-inf ag1 (up to august 2011)

department of applied mathematics
university of waterloo

# 1. introduction

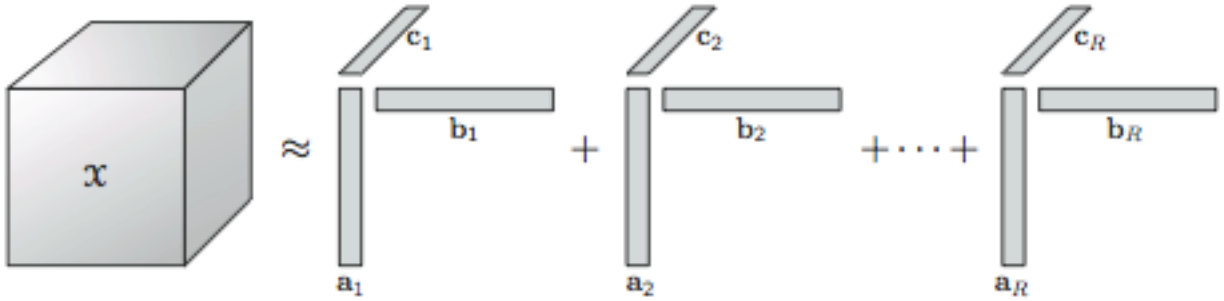
- tensor = *N*-dimensional array
- *N*=3:




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

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

# introduction




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

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, ... ; but smooth, and assume there is a local minimum)

(de Silva and Lim, SIMAX, 2009)

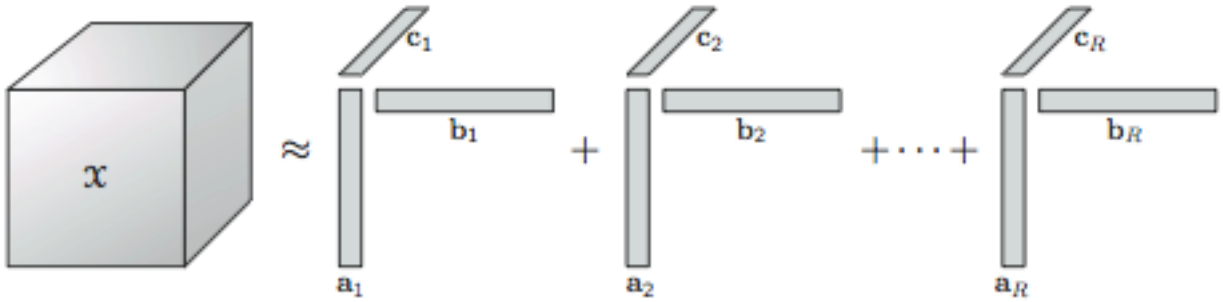
UNIVERSITY OF WATERLOO

# link with singular value decomposition

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

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

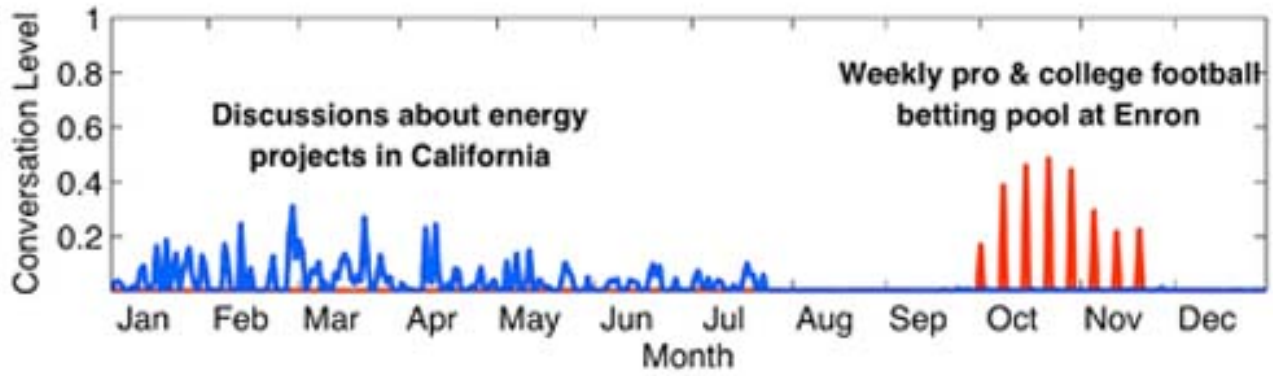
- canonical decomposition of tensor




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

# 2. tensor approximation applications

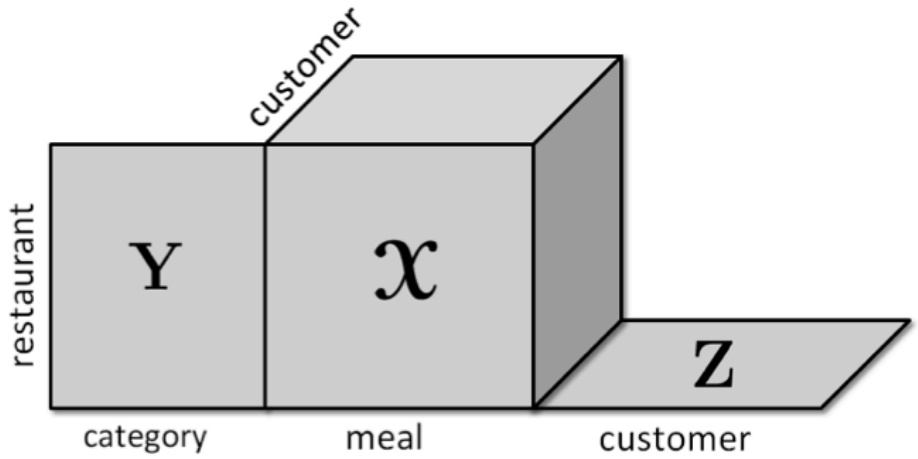
(1) “Discussion Tracking in Enron Email Using PARAFAC” by Bader, Berry and Browne (2008) (sparse, nonnegative)

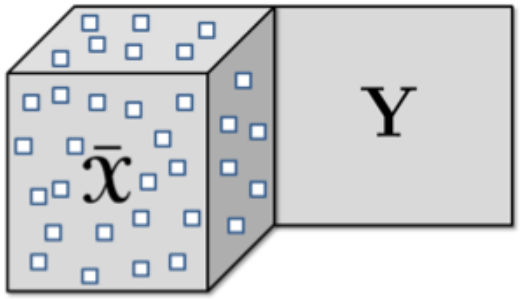




UNIVERSITY OF WATERLOO

# tensor approximation applications

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




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

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

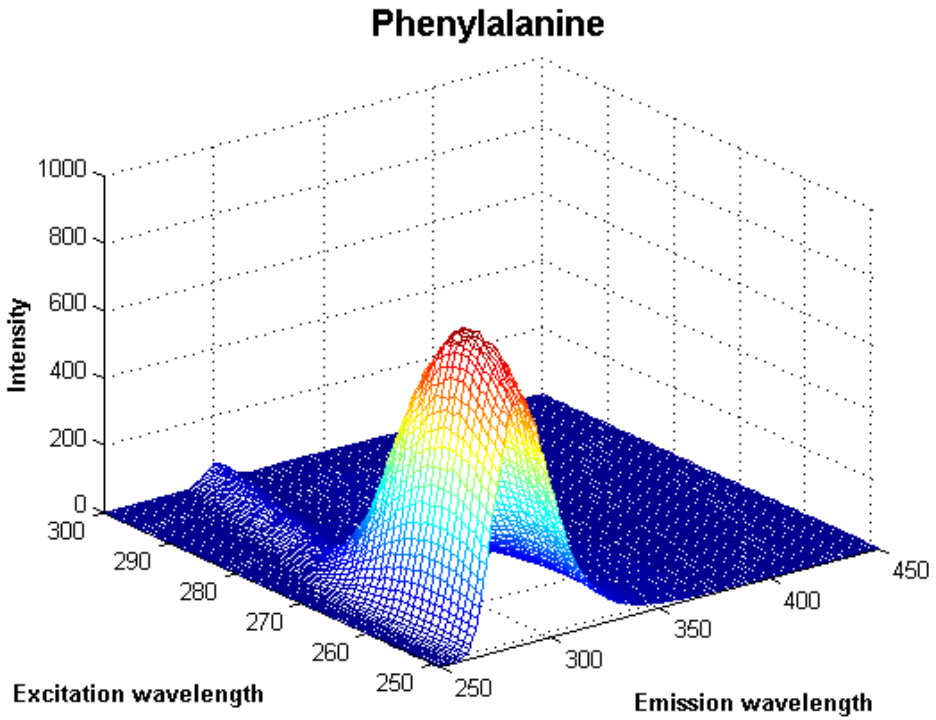
UNIVERSITY OF WATERLOO

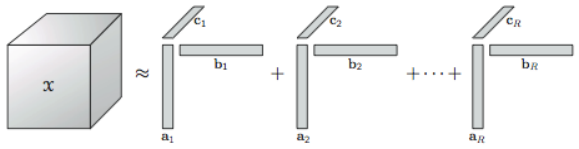
# tensor approximation applications

(3) 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

# 3. alternating least squares (ALS)

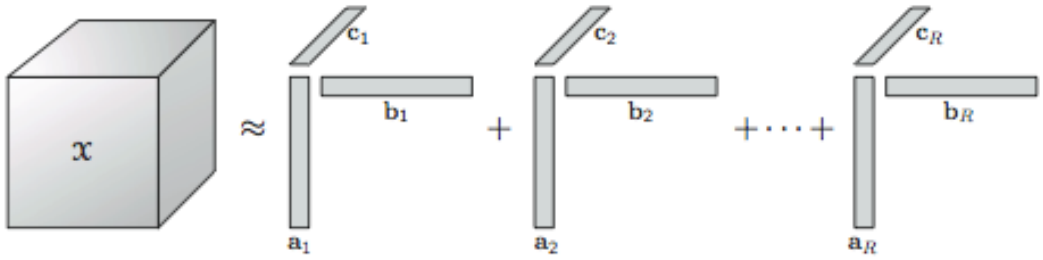
$$f(\mathcal{A}_R) = \frac{1}{2} \left\| \mathcal{T} - \sum_{r=1}^{R} a_r^{(1)} \circ a_r^{(2)} \circ a_r^{(3)} \right\|_F^2$$

(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)

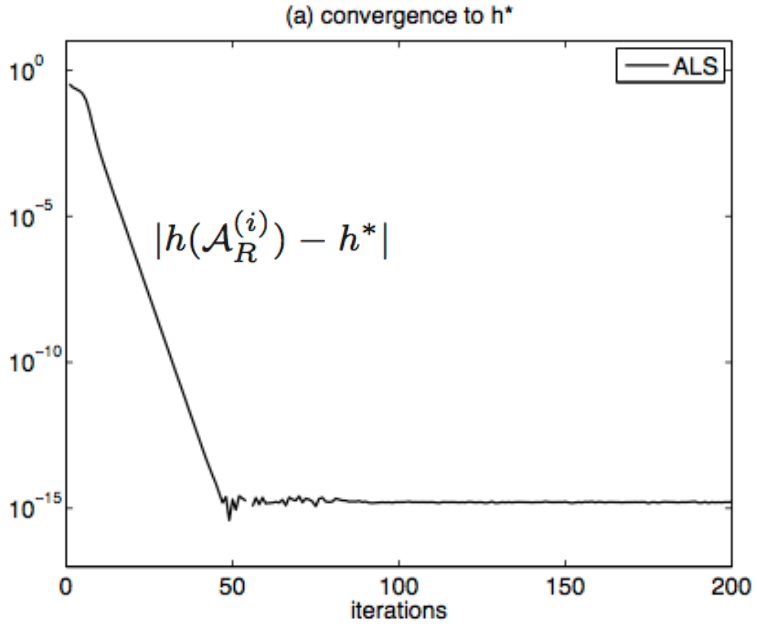
$$f(\mathcal{A}_R) = \frac{1}{2} \left\| \mathcal{T} - \sum_{r=1}^{R} a_r^{(1)} \circ a_r^{(2)} \circ a_r^{(3)} \right\|_F^2$$

- ALS is monotone
- ALS is sometimes fast, but can also be extremely slow (depending on problem and initial condition)

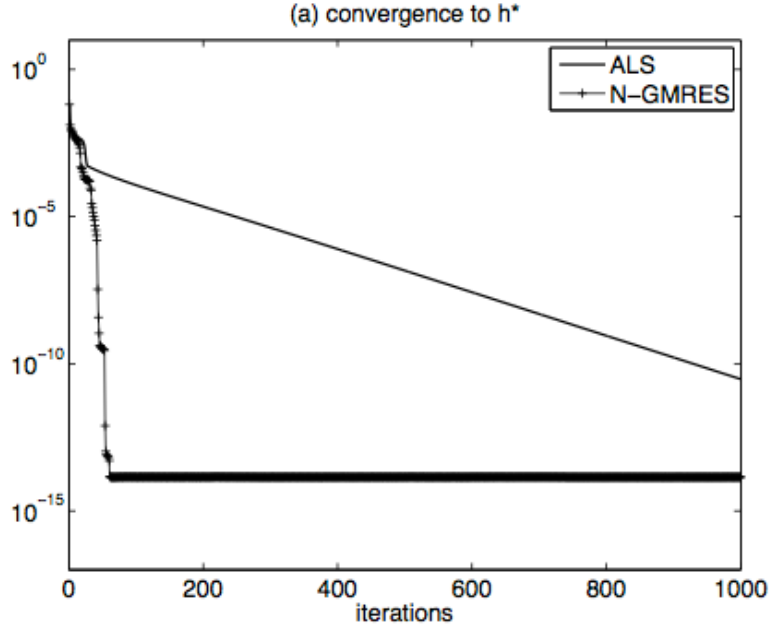
# alternating least squares (ALS)

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

## fast case




## slow case

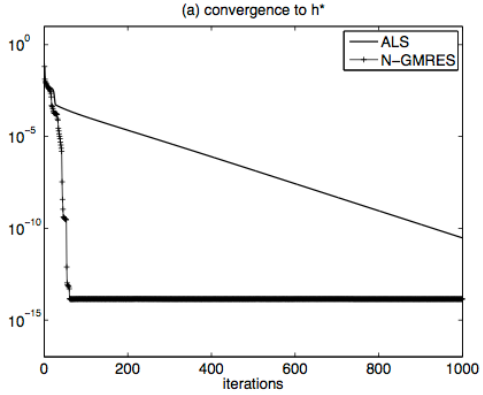



(we used Matlab with Tensor Toolbox (Bader and Kolda) and Poblano Toolbox (Dunlavy et al.) for all computations)

UNIVERSITY OF WATERLOO

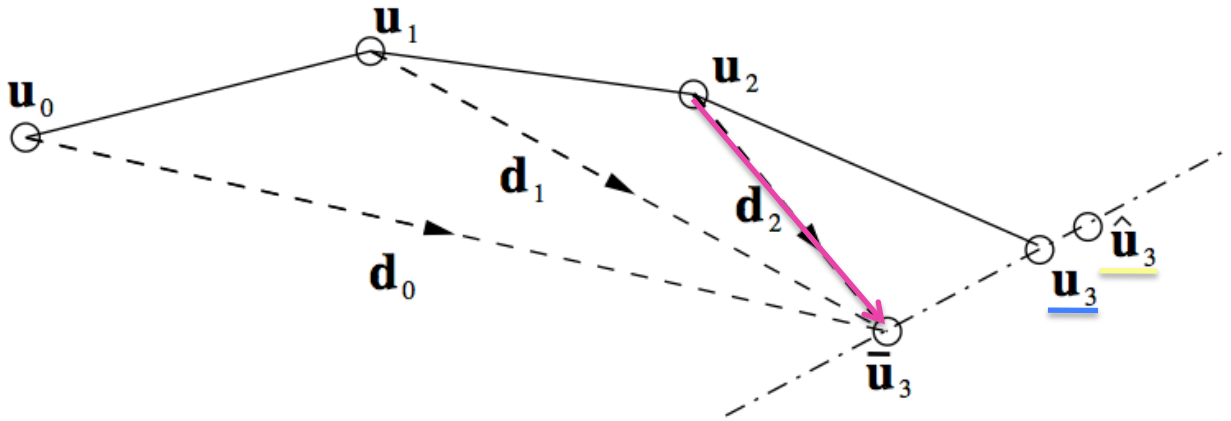
# alternating least squares (ALS)

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




- for linear systems $\mathbf{A}\,\mathbf{u} = \mathbf{b}$ , when a simple iterative method is slow, we accelerate it with
  - GMRES (generalized minimal residual method)
  - CG (conjugate gradient method), etc.
- for optimization problems, general approaches to accelerate simple iterative methods are uncommon (do not exist?)
- let's try to accelerate ALS for the tensor optimization problem
- issues: nonlinear, optimization context

# 4. nonlinear GMRES acceleration of ALS



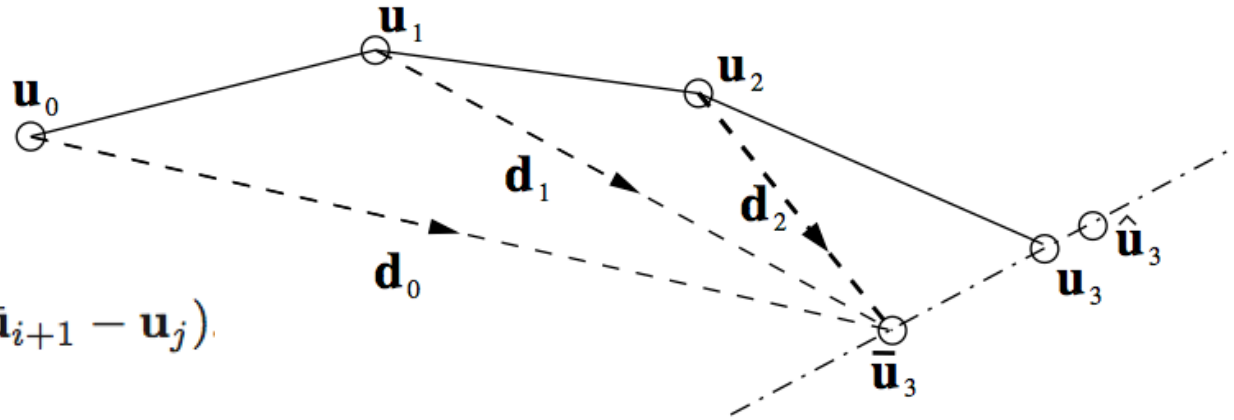

**Algorithm 1**: N-GMRES optimization algorithm (window size $w$)

**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 $M(.)$)*
    $\bar{\mathbf{u}}_{i+1} = M(\mathbf{u}_i)$
  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)*
    $\mathbf{u}_{i+1} = \text{linesearch}(\bar{\mathbf{u}}_{i+1} + \beta(\hat{\mathbf{u}}_{i+1} - \bar{\mathbf{u}}_{i+1}))$
  $i = i + 1$
**until** *convergence criterion satisfied*

(Moré-Thuente line search, satisfies Wolfe conditions)

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




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

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

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

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

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

UNIVERSITY OF WATERLOO

# history of nonlinear acceleration mechanism for nonlinear systems (step II)

STEP I: *(generate preliminary iterate by one-step update process $M(.)$)*
$\bar{\mathbf{u}}_{i+1} = M(\mathbf{u}_i)$
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)*
$\mathbf{u}_{i+1} = \text{linesearch}(\bar{\mathbf{u}}_{i+1} + \beta(\hat{\mathbf{u}}_{i+1} - \bar{\mathbf{u}}_{i+1}))$

$$\nabla f(\mathbf{u}) = \mathbf{g}(\mathbf{u}) = 0$$

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

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

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

- Washio and Oosterlee, ETNA, 1997
- GMRES, Saad and Schultz, 1986

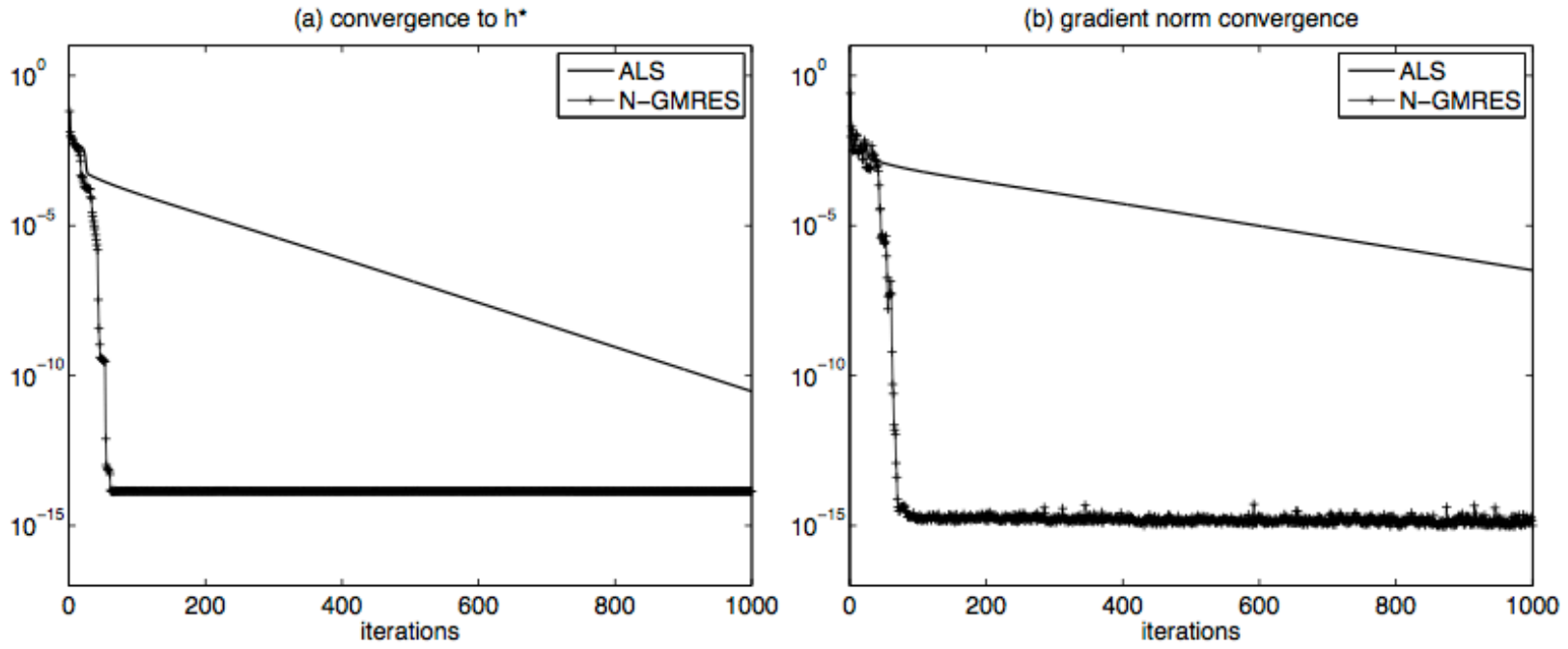
(also flexible GMRES, Saad, 1993)

- Anderson mixing, 1965; DIIS (direct inversion in the iterative subspace), Pulay, 1980
- can be interpreted as a specific Broyden-type multi-secant method for $\nabla f(\mathbf{u}) = \mathbf{g}(\mathbf{u}) = 0$ (see Fang and Saad, 2009; Walker and Ni, 2011)

- BUT: apparently not used systematically yet for optimization (or not common)
- this looks like a generally applicable continuous optimization method ...

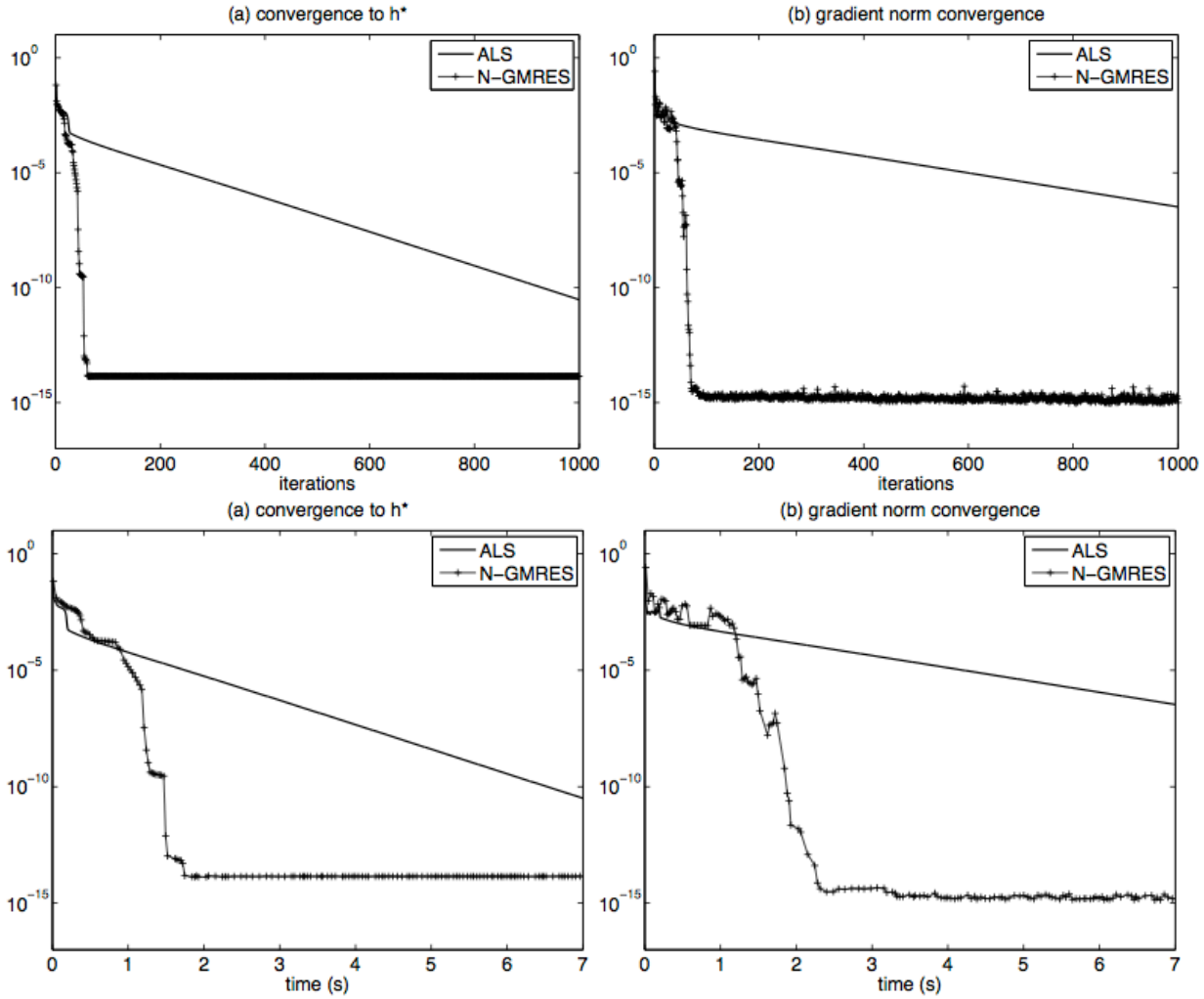
# 5. numerical results for ALS-preconditioned N-GMRES applied to tensor problem

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



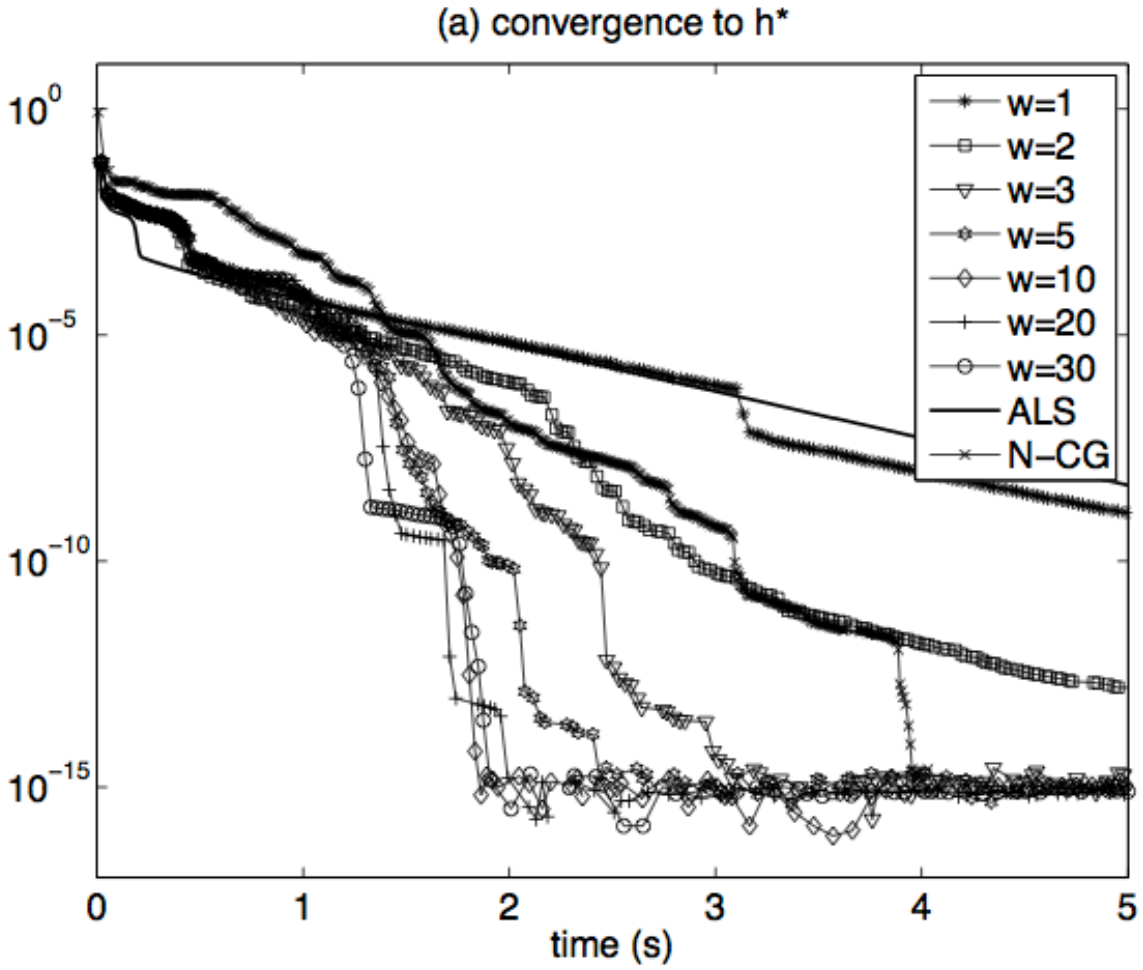

UNIVERSITY OF WATERLOO

# numerical results: dense test problem




UNIVERSITY OF WATERLOO

# dense test problem: optimal window size




UNIVERSITY OF WATERLOO

# 6. why does this work: linear case

GMRES for linear systems: $\mathbf{A}\,\mathbf{u} = \mathbf{b}$

- stationary iterative method $\mathbf{u}_{i+1} = \mathbf{u}_i + \mathbf{M}^{-1}\,\mathbf{r}_i$
  (preconditioning process)
- preconditioner $\mathbf{M}^{-1} \approx \mathbf{A}^{-1}$
- define residual and error:

$$\mathbf{r}_i = \mathbf{b} - \mathbf{A}\,\mathbf{u}_i \qquad \mathbf{e}_i = \mathbf{u} - \mathbf{u}_i \qquad \mathbf{A}\,\mathbf{e}_i = \mathbf{r}_i$$

- exact update equation: $\mathbf{u} = \mathbf{u}_i + \mathbf{e}_i = \mathbf{u}_i + \mathbf{A}^{-1}\,\mathbf{r}_i$
- approximate update equation: $\mathbf{u}_{i+1} = \mathbf{u}_i + \mathbf{M}^{-1}\,\mathbf{r}_i$

# comparing N-GMRES to GMRES

GMRES for linear systems: $\mathbf{A}\,\mathbf{u} = \mathbf{b}$

- stationary iterative method $\mathbf{u}_{i+1} = \mathbf{u}_i + \mathbf{M}^{-1}\,\mathbf{r}_i$
- generates residuals recursively:

$$\begin{aligned}\mathbf{r}_i &= \mathbf{b} - \mathbf{A}\,\mathbf{u}_i \\ &= (\mathbf{I} - \mathbf{A}\mathbf{M}^{-1})\,\mathbf{r}_{i-1} \\ &= (\mathbf{I} - \mathbf{A}\mathbf{M}^{-1})^i\,\mathbf{r}_0.\end{aligned}$$

- define Krylov space $K_{i+1}(\mathbf{A}\mathbf{M}^{-1}, \mathbf{r}_0)$

$$\begin{aligned}V_{1,i+1} &= span\{\mathbf{r}_0, \ldots, \mathbf{r}_i\}, \\ V_{2,i+1} &= span\{\mathbf{r}_0, \mathbf{A}\mathbf{M}^{-1}\,\mathbf{r}_0, (\mathbf{A}\mathbf{M}^{-1})^2\,\mathbf{r}_0\}, \ldots, (\mathbf{A}\mathbf{M}^{-1})^i\,\mathbf{r}_0\} \\ &= K_{i+1}(\mathbf{A}\mathbf{M}^{-1}, \mathbf{r}_0), \\ V_{3,i+1} &= span\{\mathbf{M}\,(\mathbf{u}_1 - \mathbf{u}_0), \mathbf{M}\,(\mathbf{u}_2 - \mathbf{u}_1), \ldots, \mathbf{M}\,(\mathbf{u}_{i+1} - \mathbf{u}_i)\}, \\ V_{4,i+1} &= span\{\mathbf{M}\,(\mathbf{u}_{i+1} - \mathbf{u}_0), \mathbf{M}\,(\mathbf{u}_{i+1} - \mathbf{u}_1), \ldots, \mathbf{M}\,(\mathbf{u}_{i+1} - \mathbf{u}_i)\}\end{aligned}$$

(Washio and Oosterlee, ETNA, 1997)

LEMMA 2.1. $V_{1,i+1} = V_{2,i+1} = V_{3,i+1} = V_{4,i+1}$

UNIVERSITY OF WATERLOO

# comparing N-GMRES to GMRES

(Washio and Oosterlee, ETNA, 1997)

GMRES for linear systems: $\mathbf{A}\,\mathbf{u} = \mathbf{b}$

- stationary iterative process $\mathbf{u}_{i+1} = \mathbf{u}_i + \mathbf{M}^{-1}\,\mathbf{r}_i$ generates preconditioned residuals that build Krylov space

$$
\begin{aligned}
V_{1,i+1} &= span\{\mathbf{r}_0, \ldots, \mathbf{r}_i\}, \\
V_{2,i+1} &= span\{\mathbf{r}_0, \mathbf{A}\mathbf{M}^{-1}\,\mathbf{r}_0, (\mathbf{A}\mathbf{M}^{-1})^2\,\mathbf{r}_0\}, \ldots, (\mathbf{A}\mathbf{M}^{-1})^i\,\mathbf{r}_0\} \\
&= K_{i+1}(\mathbf{A}\mathbf{M}^{-1}, \mathbf{r}_0),
\end{aligned}
$$

- GMRES: take optimal linear combination of residuals in Krylov space to minimize the residual $\|\hat{\mathbf{r}}_{i+1}\|_2$

UNIVERSITY OF WATERLOO

# comparing N-GMRES to GMRES

$$\mathbf{A}\,\mathbf{u} = \mathbf{b}$$

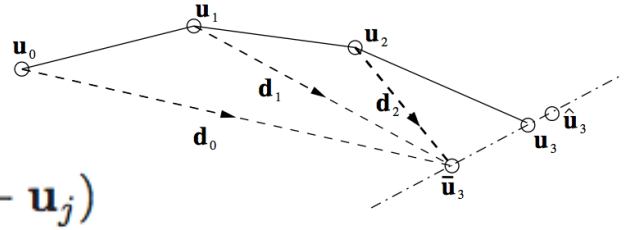
$$\mathbf{u}_{i+1} = \mathbf{u}_i + \mathbf{M}^{-1}\,\mathbf{r}_i$$

$$\begin{aligned}
V_{1,i+1} &= span\{\mathbf{r}_0, \ldots, \mathbf{r}_i\}, \\
V_{2,i+1} &= span\{\mathbf{r}_0, \mathbf{A}\mathbf{M}^{-1}\,\mathbf{r}_0, (\mathbf{A}\mathbf{M}^{-1})^2\,\mathbf{r}_0\}, \ldots, (\mathbf{A}\mathbf{M}^{-1})^i\,\mathbf{r}_0\} \\
&= K_{i+1}(\mathbf{A}\mathbf{M}^{-1}, \mathbf{r}_0), \\
V_{3,i+1} &= span\{\mathbf{M}\,(\mathbf{u}_1 - \mathbf{u}_0), \mathbf{M}\,(\mathbf{u}_2 - \mathbf{u}_1), \ldots, \mathbf{M}\,(\mathbf{u}_{i+1} - \mathbf{u}_i)\}, \\
V_{4,i+1} &= span\{\mathbf{M}\,(\mathbf{u}_{i+1} - \mathbf{u}_0), \mathbf{M}\,(\mathbf{u}_{i+1} - \mathbf{u}_1), \ldots, \mathbf{M}\,(\mathbf{u}_{i+1} - \mathbf{u}_i)\}
\end{aligned}$$

- GMRES: minimize $\|\hat{\mathbf{r}}_{i+1}\|_2$
- seek optimal approximation $\mathbf{M}\,(\hat{\mathbf{u}}_{i+1} - \mathbf{u}_i) = \sum_{j=0}^{i} \beta_j\,\mathbf{M}\,(\mathbf{u}_{i+1} - \mathbf{u}_j)$

$$\hat{\mathbf{u}}_{i+1} = \mathbf{u}_i + \sum_{j=0}^{i} \beta_j\,(\mathbf{u}_{i+1} - \mathbf{u}_j)$$

$$= \mathbf{u}_{i+1} - (\mathbf{u}_{i+1} - \mathbf{u}_i) + \sum_{j=0}^{i} \beta_j\,(\mathbf{u}_{i+1} - \mathbf{u}_j)$$




$$\hat{\mathbf{u}}_{i+1} = \mathbf{u}_{i+1} + \sum_{j=0} \alpha_j\,(\mathbf{u}_{i+1} - \mathbf{u}_j)$$ same as for N-GMRES

# convergence speed of GMRES

$$
\begin{aligned}
&\mathbf{A}\,\mathbf{u} = \mathbf{b}\\
&\mathbf{u}_{i+1} = \mathbf{u}_i + \mathbf{M}^{-1}\,\mathbf{r}_i\\
&\mathbf{r}_i = \mathbf{b} - \mathbf{A}\,\mathbf{u}_i\\
&\quad = (\mathbf{I} - \mathbf{A}\mathbf{M}^{-1})\,\mathbf{r}_{i-1}\\
&\quad = (\mathbf{I} - \mathbf{A}\mathbf{M}^{-1})^i\,\mathbf{r}_0.
\end{aligned}
$$

$$
\begin{aligned}
V_{1,i+1} &= span\{\mathbf{r}_0, \ldots, \mathbf{r}_i\},\\
V_{2,i+1} &= span\{\mathbf{r}_0, \mathbf{A}\mathbf{M}^{-1}\,\mathbf{r}_0, (\mathbf{A}\mathbf{M}^{-1})^2\,\mathbf{r}_0\}, \ldots, (\mathbf{A}\mathbf{M}^{-1})^i\,\mathbf{r}_0\}\\
&= K_{i+1}(\mathbf{A}\mathbf{M}^{-1}, \mathbf{r}_0),\\
V_{3,i+1} &= span\{\mathbf{M}\,(\mathbf{u}_1 - \mathbf{u}_0), \mathbf{M}\,(\mathbf{u}_2 - \mathbf{u}_1), \ldots, \mathbf{M}\,(\mathbf{u}_{i+1} - \mathbf{u}_i)\},\\
V_{4,i+1} &= span\{\mathbf{M}\,(\mathbf{u}_{i+1} - \mathbf{u}_0), \mathbf{M}\,(\mathbf{u}_{i+1} - \mathbf{u}_1), \ldots, \mathbf{M}\,(\mathbf{u}_{i+1} - \mathbf{u}_i)\}
\end{aligned}
$$

- GMRES: minimize $\|\hat{\mathbf{r}}_{i+1}\|_2$
- polynomial method: convergence determined by optimal polynomial (diagonalizable matrix, $A=V\Lambda V^{-1}$)

$$\|r_n\| \leq \inf_{p \in P_n} \|p_n(A)\| \leq \kappa_2(V) \inf_{p \in P_n} \max_{\lambda \in \sigma(A)} |p(\lambda)|$$

UNIVERSITY OF WATERLOO

# convergence speed of N-GMRES

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

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

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)*

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

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

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

- GMRES (linear case): convergence determined by optimal polynomial
- convergence speed of N-GMRES for optimization: open problem

# 7. general N-GMRES optimization method

general methods for nonlinear optimization (smooth, unconstrained)
("Numerical Optimization", Nocedal and Wright, 2006)

1. steepest descent with line search
2. Newton with line search
3. nonlinear conjugate gradient (N-CG) with line search
4. trust-region methods
5. quasi-Newton methods (includes Broyden–Fletcher–Goldfarb–Shanno (BFGS) and limited memory version L-BFGS)

6. N-GMRES as a general optimization method?

# general N-GMRES optimization method

- first question: what would be a general preconditioner?

OPTIMIZATION PROBLEM

$$\text{find } \mathbf{u}^* \text{ that minimizes } f(\mathbf{u})$$

FIRST-ORDER OPTIMALITY EQUATIONS

$$\nabla f(\mathbf{u}) = \mathbf{g}(\mathbf{u}) = 0$$

- idea: general N-GMRES preconditioner $\bar{\mathbf{u}}_{i+1} = M(\mathbf{u}_i)$
  = update in direction of steepest descent
  (or: use N-GMRES to accelerate steepest descent)

# 8. steepest-descent preconditioning

STEP I: *(generate preliminary iterate by one-step update process $M(.)$)*
$\bar{\mathbf{u}}_{i+1} = M(\mathbf{u}_i)$
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)*
$\mathbf{u}_{i+1} = \text{linesearch}(\bar{\mathbf{u}}_{i+1} + \beta(\hat{\mathbf{u}}_{i+1} - \bar{\mathbf{u}}_{i+1}))$

STEEPEST DESCENT PRECONDITIONING PROCESS:

$$\bar{\mathbf{u}}_{i+1} = \mathbf{u}_i - \beta \frac{\nabla f(\mathbf{u}_i)}{\|\nabla f(\mathbf{u}_i)\|} \qquad \text{with}$$

OPTION A: $\beta = \beta_{sdls}$,

OPTION B: $\beta = \beta_{sd} = \min(\, \delta \,,\, \|\nabla f(\mathbf{u}_i)\| \,)$

- option A: steepest descent with line search
- option B: steepest descent with predefined small step
- claim: steepest descent is the ‘natural’ preconditioner for N-GMRES

UNIVERSITY OF WATERLOO

# steepest-descent preconditioning

- claim: steepest descent is the 'natural' preconditioner for N-GMRES
- example: consider simple quadratic optimization problem

$$f(\mathbf{u}) = \frac{1}{2}\,\mathbf{u}^T A\mathbf{u} - \mathbf{b}^T\mathbf{u} \quad \text{where } A \text{ is SPD}$$

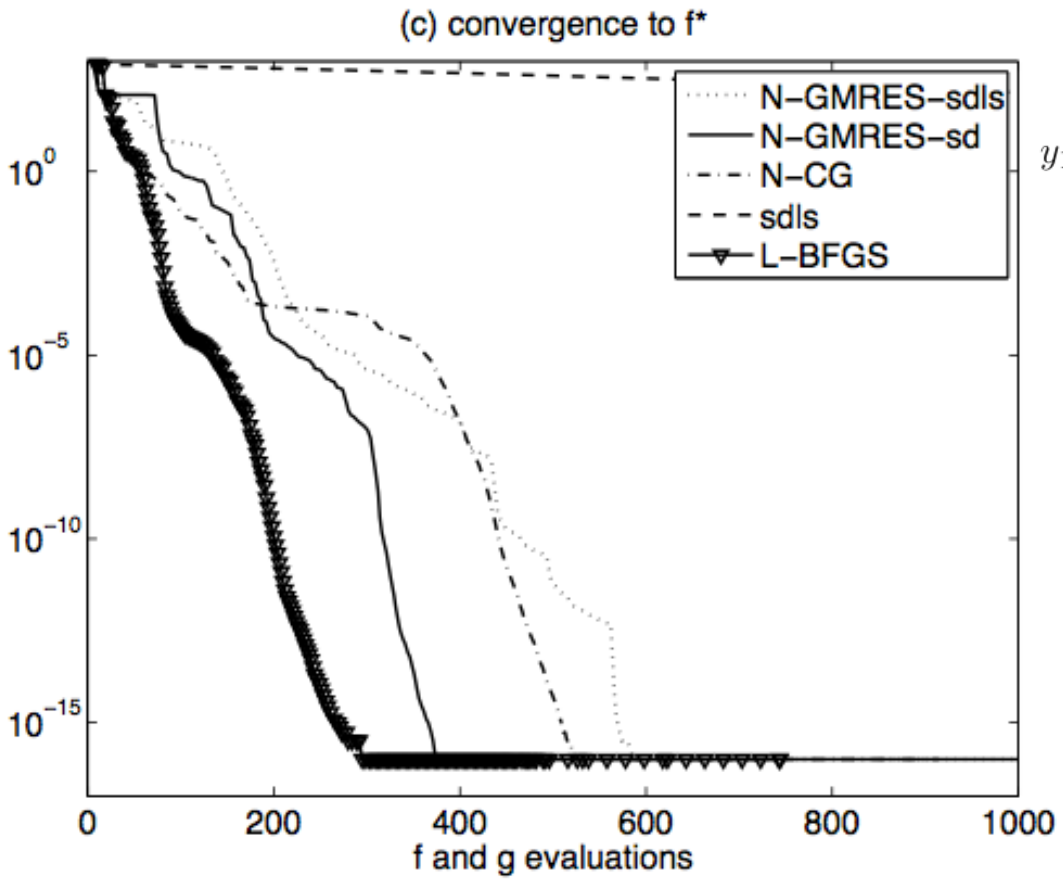
- we know $\nabla f(\mathbf{u}_i) = A\mathbf{u}_i - b = -\mathbf{r}_i$ so

$$\bar{\mathbf{u}}_{i+1} = \mathbf{u}_i - \beta\,\frac{\nabla f(\mathbf{u}_i)}{\|\nabla f(\mathbf{u}_i)\|} \quad \text{becomes} \quad \bar{\mathbf{u}}_{i+1} = \mathbf{u}_i + \beta\,\frac{\mathbf{r}_i}{\|\mathbf{r}_i\|}$$

- this gives the same residuals as $\mathbf{u}_{i+1} = \mathbf{u}_i + \mathbf{M}^{-1}\,\mathbf{r}_i$ with $\mathbf{M} = \mathbf{I}$ : steepest-descent N-GMRES preconditioner corresponds to identity preconditioner for linear GMRES

(and: small step is sufficient)

# 9. numerical results: steepest-descent preconditioning




$$f(\mathbf{u}) = \frac{1}{2}\,\mathbf{y}(\mathbf{u}-\mathbf{u}^*)^T D\,\mathbf{y}(\mathbf{u}-\mathbf{u}^*) + 1,$$

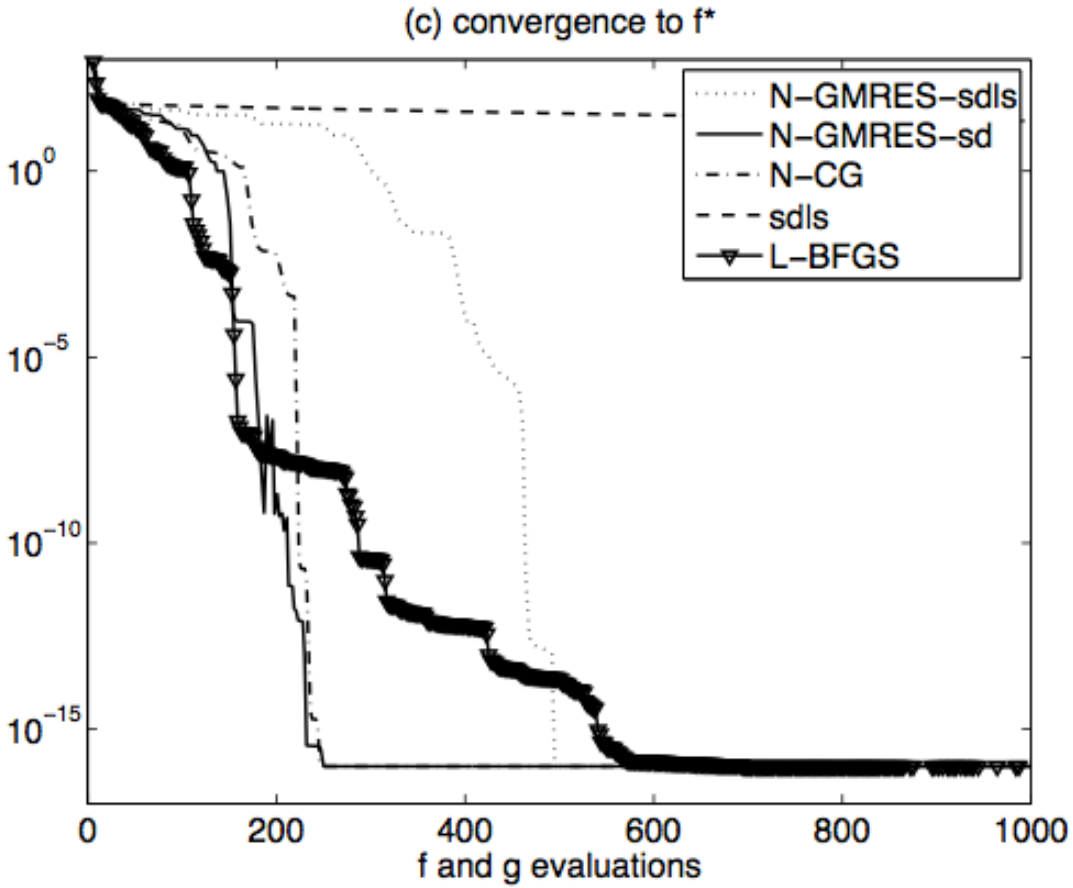
with $D = \mathrm{diag}(1, 2, \ldots, n)$ and $\mathbf{y}(\mathbf{x})$ given by

$y_1(\mathbf{x}) = x_1$ and $y_i(\mathbf{x}) = x_i - 10\,x_1^2$ $(i = 2, \ldots, n)$.

- steepest descent by itself is slow
- N-GMRES with steepest descent preconditioning is competitive with N-CG and L-BFGS
- option A slower than option B (small step)

UNIVERSITY OF WATERLOO

# numerical results: steepest-descent preconditioning




$$f(\mathbf{u}) = \frac{1}{2}\sum_{j=1}^{n} t_j^2(\mathbf{u}), \text{ with } n \text{ even and}$$

$$t_j = 10\,(u_{j+1} - u_j^2) \qquad (j \text{ odd}),$$

$$t_j = 1 - u_{j-1} \qquad (j \text{ even}).$$

- extended Rosenbrock function
- steepest descent by itself is slow
- N-GMRES with steepest descent preconditioning is competitive with N-CG and L-BFGS

UNIVERSITY OF WATERLOO

# 10. convergence of steepest-descent preconditioned N-GMRES optimization

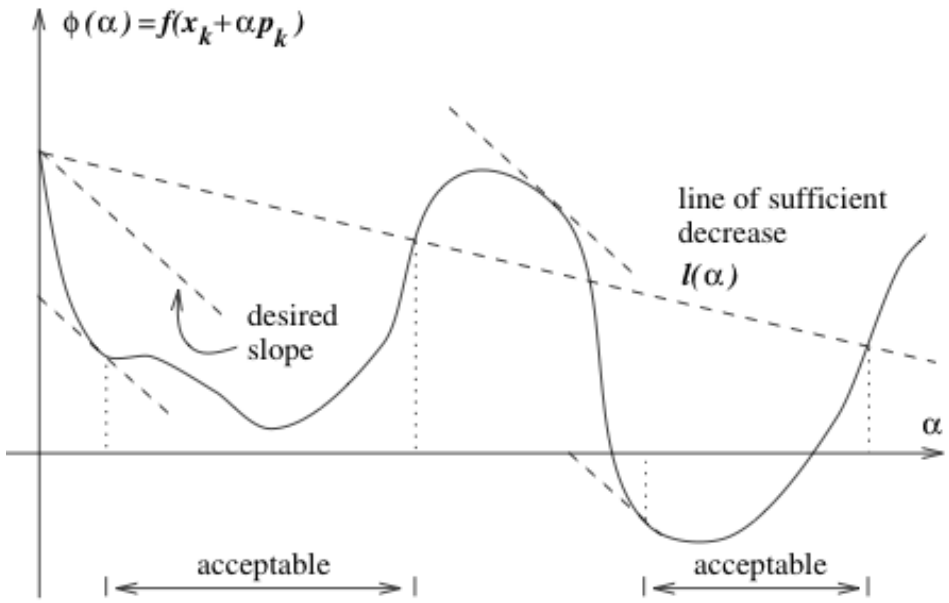
- assume line searches give solutions that satisfy Wolfe conditions:

SUFFICIENT DECREASE CONDITION:

$$f(\mathbf{u}_i + \beta_i \mathbf{p}_i) \leq f(\mathbf{u}_i) + c_1 \, \beta_i \, \nabla f(\mathbf{u}_i)^T \mathbf{p}_i,$$

CURVATURE CONDITION:

$$\nabla f(\mathbf{u}_i + \beta_i \mathbf{p}_i)^T \, \mathbf{p}_i \geq c_2 \, \nabla f(\mathbf{u}_i)^T \mathbf{p}_i,$$

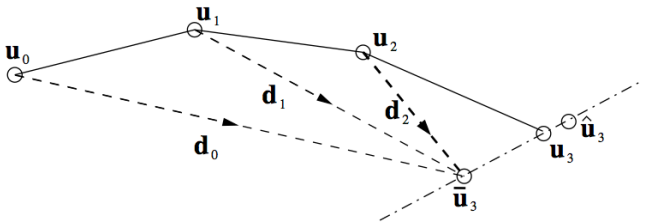



(Nocedal and Wright, 2006)

# convergence of steepest-descent preconditioned N-GMRES optimization

THEOREM 2.1 (Global convergence of N-GMRES optimization algorithm with steepest descent line search preconditioning). *Consider N-GMRES Optimization Algorithm 1 with steepest descent line search preconditioning (2.1) for Optimization Problem I, and assume that all line search solutions satisfy the Wolfe conditions, (2.11) and (2.12). Assume that objective function $f$ is bounded below in $\mathbb{R}^n$ and that $f$ is continuously differentiable in an open set $\mathcal{N}$ containing the level set $\mathcal{L} = \{\mathbf{u} : f(\mathbf{u}) \leq f(\mathbf{u}_0)\}$, where $\mathbf{u}_0$ is the starting point of the iteration. Assume also that the gradient $\nabla f$ is Lipschitz continuous on $\mathcal{N}$, that is, there exists a constant $L$ such that $\|\nabla f(\mathbf{u}) - \nabla f(\hat{\mathbf{u}})\| \leq L\|\mathbf{u} - \hat{\mathbf{u}}\|$ for all $\mathbf{u}, \hat{\mathbf{u}} \in \mathcal{N}$. Then the sequence of N-GMRES iterates $\{\mathbf{u}_0, \mathbf{u}_1, \ldots\}$ is convergent to a fixed point of Optimization Problem I in the sense that*

$$\lim_{i \to \infty} \|\nabla f(\mathbf{u}_i)\| = 0. \tag{2.13}$$

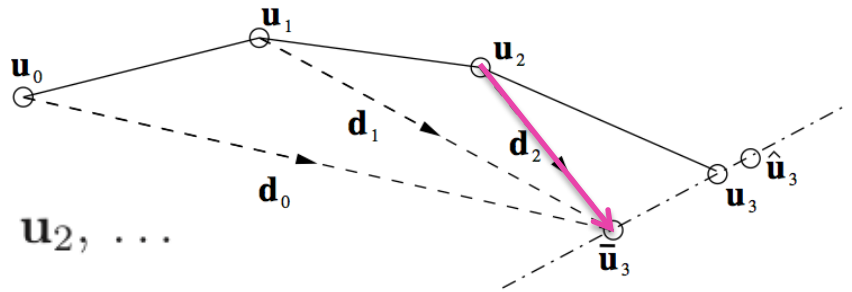



UNIVERSITY OF WATERLOO

STEP I: *(generate preliminary iterate by one-step update process $M(.)$)*
$\bar{\mathbf{u}}_{i+1} = M(\mathbf{u}_i)$
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)*
$\mathbf{u}_{i+1} = \text{linesearch}(\bar{\mathbf{u}}_{i+1} + \beta(\hat{\mathbf{u}}_{i+1} - \bar{\mathbf{u}}_{i+1}))$

# convergence of steepest-descent preconditioned N-GMRES optimization

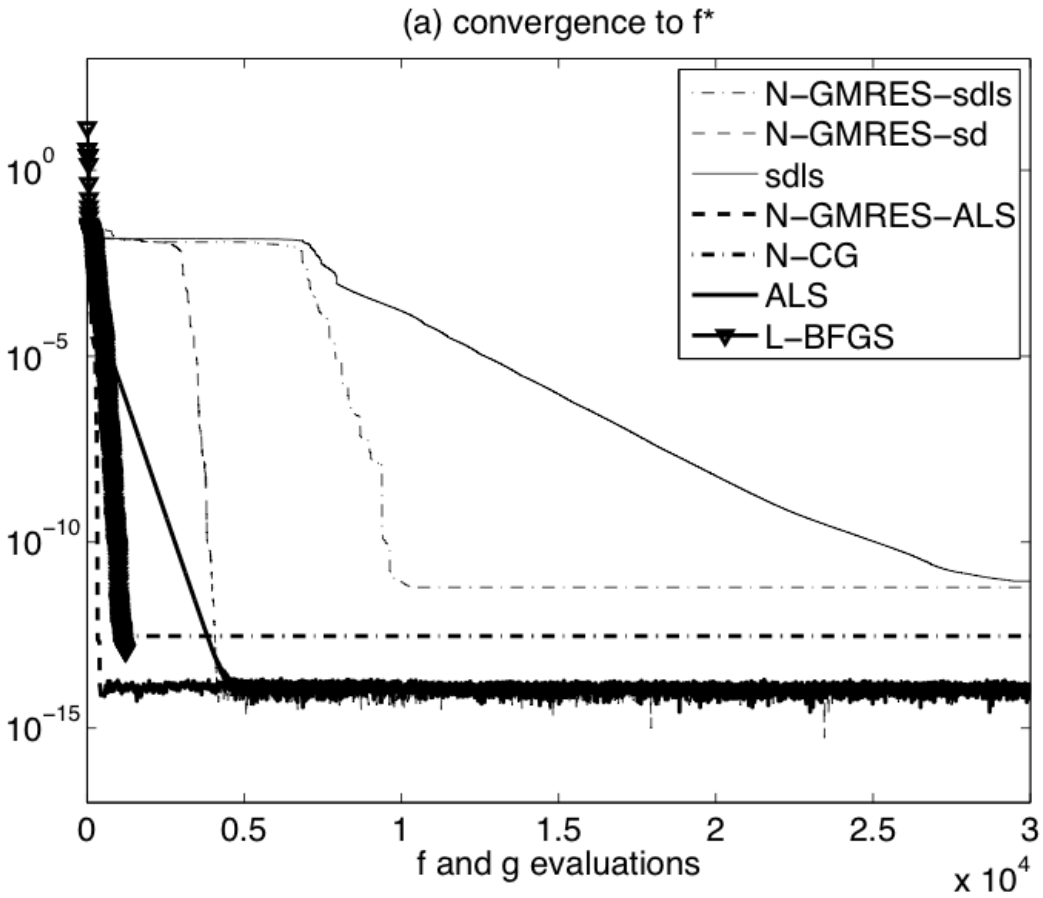
## sketch of (simple!) proof




- Consider the sequence $\{\mathbf{v}_0, \mathbf{v}_1, \ldots\}$ formed by the iterates $\mathbf{u}_0,\ \bar{\mathbf{u}}_1,\ \mathbf{u}_1,\ \bar{\mathbf{u}}_2,\ \mathbf{u}_2,\ \ldots$
- use Zoutendijk's theorem: $\sum_{i=0}^{\infty} \cos^2\theta_i \, \|\nabla f(\mathbf{v}_i)\|^2 < \infty$ with $\cos\theta_i = \dfrac{-\nabla f(\mathbf{v}_i)^T \mathbf{p}_i}{\|\nabla f(\mathbf{v}_i)\| \, \|\mathbf{p}_i\|}$ and thus $\lim_{i\to\infty} \cos^2\theta_i \, \|\nabla f(\mathbf{v}_i)\|^2 = 0$
- all $u_i$ are followed by a steepest descent step, so $\lim_{i\to\infty} \|\nabla f(\mathbf{u}_i)\| = 0.$
- global convergence to a stationary point for general *f(u)*

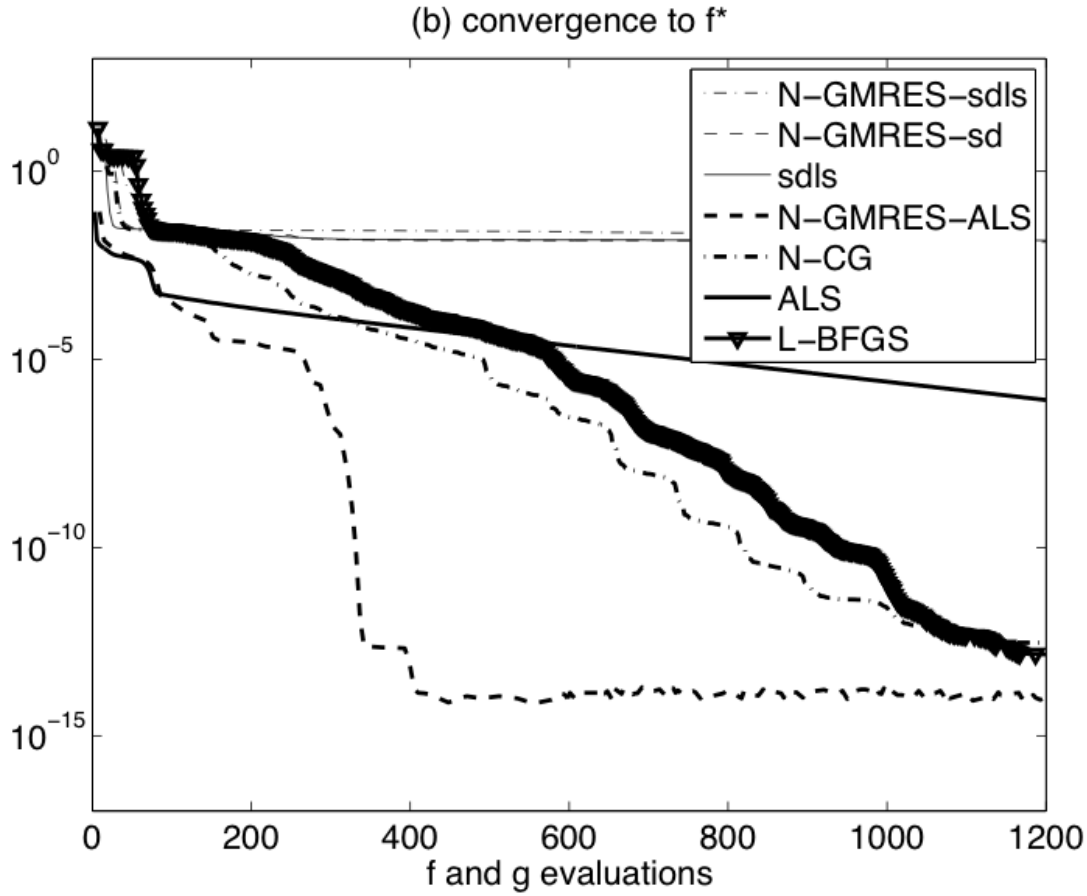
UNIVERSITY OF WATERLOO

# the power of N-GMRES optimization (tensor problem)




UNIVERSITY OF WATERLOO

# the power of N-GMRES optimization (tensor problem)




UNIVERSITY OF WATERLOO

- thank you
- questions?


- Hans De Sterck, *'A Nonlinear GMRES Optimization Algorithm for Canonical Tensor Decomposition'*, submitted, May 2011, *arXiv:1105.5331*
- Hans De Sterck, *'Steepest Descent Preconditioning for Nonlinear GMRES Optimization'*, submitted, June 2011, *arXiv:1106.4426*

# BACKUP SLIDES

# comparing N-GMRES to GMRES

- non-preconditioned GMRES for linear systems:

  $\mathbf{M} = \mathbf{I} \qquad \mathbf{u}_{i+1} = \mathbf{u}_i + \mathbf{M}^{-1}\mathbf{r}_i \quad \text{Krylov space } K_{i+1}(\mathbf{A}\mathbf{M}^{-1}, \mathbf{r}_0)$

- apply non-preconditioned GMRES to preconditioned linear system $\mathbf{A}\mathbf{M}^{-1}(\mathbf{M}\mathbf{u}) = \mathbf{b}$ or $(\mathbf{A}\mathbf{M}^{-1})\mathbf{y} = \mathbf{b}$
- preconditioner changes the spectrum of the operator such that (non-preconditioned) GMRES applied to the preconditioned operator converges better
- this alternative viewpoint of preconditioned GMRES leads to the same formulas as what we derived in the previous slides

# conjugate gradient (CG)

**Algorithm 5.2** (CG).

Given $x_0$;

Set $r_0 \leftarrow Ax_0 - b$, $p_0 \leftarrow -r_0$, $k \leftarrow 0$;

**while** $r_k \neq 0$

$$\alpha_k \leftarrow \frac{r_k^T r_k}{p_k^T A p_k};$$

$$x_{k+1} \leftarrow x_k + \alpha_k p_k;$$

$$r_{k+1} \leftarrow r_k + \alpha_k A p_k;$$

$$\beta_{k+1} \leftarrow \frac{r_{k+1}^T r_{k+1}}{r_k^T r_k};$$

$$p_{k+1} \leftarrow -r_{k+1} + \beta_{k+1} p_k;$$

$$k \leftarrow k + 1;$$

**end (while)**

(Nocedal and Wright, 2006)

# preconditioned conjugate gradient (PCG)

**Algorithm 5.3** (Preconditioned CG).

Given $x_0$, preconditioner $M$;
Set $r_0 \leftarrow Ax_0 - b$;
Solve $My_0 = r_0$ for $y_0$;
Set $p_0 = -y_0$, $k \leftarrow 0$;
**while** $r_k \neq 0$

$$\alpha_k \leftarrow \frac{r_k^T y_k}{p_k^T A p_k};$$
$$x_{k+1} \leftarrow x_k + \alpha_k p_k;$$
$$r_{k+1} \leftarrow r_k + \alpha_k A p_k;$$
$$\text{Solve } My_{k+1} = r_{k+1};$$
$$\beta_{k+1} \leftarrow \frac{r_{k+1}^T y_{k+1}}{r_k^T y_k};$$
$$p_{k+1} \leftarrow -y_{k+1} + \beta_{k+1} p_k;$$
$$k \leftarrow k + 1;$$

**end (while)**

(Nocedal and Wright, 2006)

# nonlinear conjugate gradient (N-CG)

**Algorithm 5.4** (FR).

Given $x_0$;

Evaluate $f_0 = f(x_0)$, $\nabla f_0 = \nabla f(x_0)$;

Set $p_0 \leftarrow -\nabla f_0$, $k \leftarrow 0$;

**while** $\nabla f_k \neq 0$

Compute $\alpha_k$ and set $x_{k+1} = x_k + \alpha_k p_k$;

Evaluate $\nabla f_{k+1}$;

$$\beta_{k+1}^{\mathrm{FR}} \leftarrow \frac{\nabla f_{k+1}^T \nabla f_{k+1}}{\nabla f_k^T \nabla f_k}; \tag{5.41a}$$

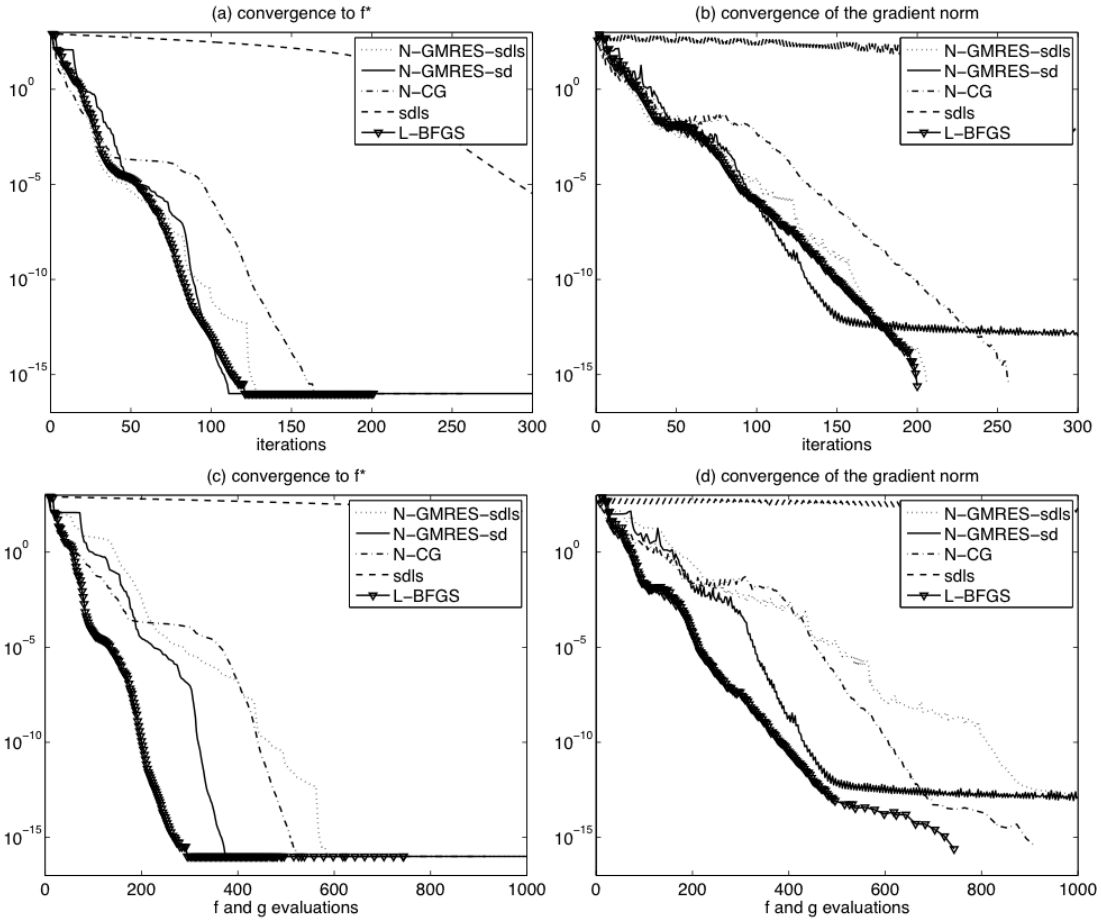
$$p_{k+1} \leftarrow -\nabla f_{k+1} + \beta_{k+1}^{\mathrm{FR}} p_k; \tag{5.41b}$$

$$k \leftarrow k + 1; \tag{5.41c}$$

**end (while)**

(Nocedal and Wright, 2006)

# 9. numerical results: steepest-descent preconditioning




$$f(\mathbf{u}) = \frac{1}{2}\mathbf{y}(\mathbf{u}-\mathbf{u}^*)^T D\,\mathbf{y}(\mathbf{u}-\mathbf{u}^*) + 1,$$

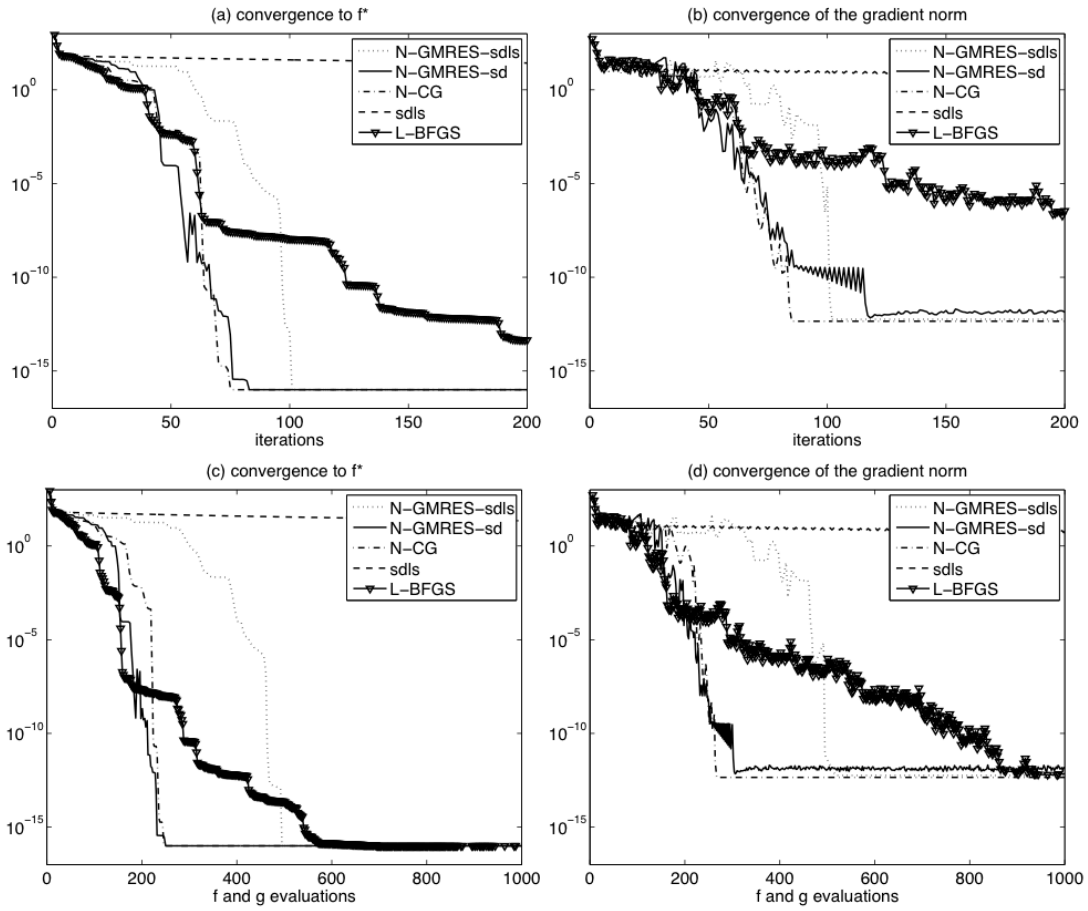
with $D = \mathrm{diag}(1, 2, \ldots, n)$ and $\mathbf{y}(\mathbf{x})$ given by

$y_1(\mathbf{x}) = x_1$ and $y_i(\mathbf{x}) = x_i - 10\,x_1^2$ $(i = 2, \ldots, n)$.

- steepest descent by itself is slow
- N-GMRES with steepest descent preconditioning is competitive with N-CG and L-BFGS
- option A slower than option B (small step)

UNIVERSITY OF WATERLOO

# numerical results: steepest-descent preconditioning




$$f(\mathbf{u}) = \frac{1}{2}\sum_{j=1}^{n} t_j^2(\mathbf{u}), \text{ with } n \text{ even and}$$

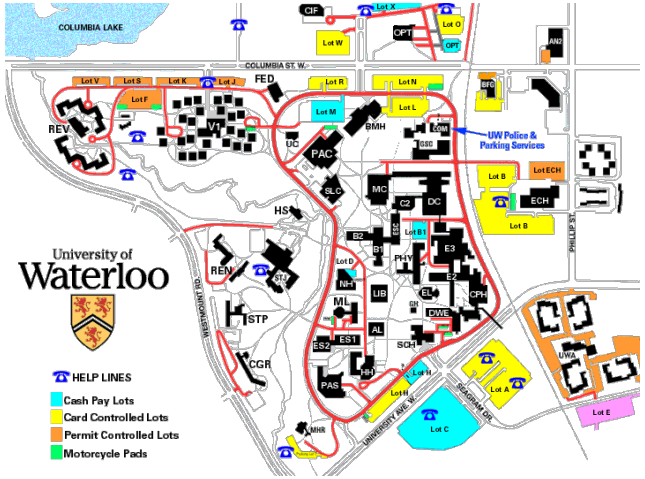
$$t_j = 10\,(u_{j+1} - u_j^2) \qquad (j \text{ odd}),$$

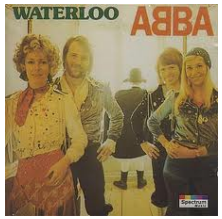
$$t_j = 1 - u_{j-1} \qquad (j \text{ even}).$$

- extended Rosenbrock function
- steepest descent by itself is slow
- N-GMRES with steepest descent preconditioning is competitive with N-CG and L-BFGS

UNIVERSITY OF WATERLOO

# Applied Mathematics Department, University of Waterloo, Canada


"Scalable Scientific Computing" research group

-2 postdocs

-5 PhD students

-Master's, undergraduate research students

Maplesoft
command the brilliance

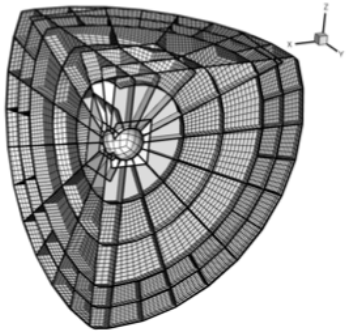
RIM
BlackBerry

Ads, Mobile, and ChromeOS

Google Canada

UNIVERSITY OF WATERLOO

# Scalable Scientific Computing group



- numerical PDEs
  - compressible fluid dynamics and MHD, space physics applications, HPC
  - GPU, finite volume element method, capillarity, ...
- numerical linear algebra, iterative methods
  - AMG for Markov chains
  - AMG for eigenproblems and SVD ➔ today's talk
  - 'graph applications', clustering (images), ...
- grid/cloud/hadoop/database, spin systems, inverse problems, ...

UNIVERSITY OF WATERLOO

# general N-GMRES optimization method

general methods for nonlinear optimization (smooth, unconstrained)
("Numerical Optimization", Nocedal and Wright, 2006)

1. steepest descent with line search
2. Newton with line search
3. nonlinear conjugate gradient (N-CG) with line search
4. trust-region methods
5. quasi-Newton methods (includes Broyden–Fletcher–Goldfarb–Shanno (BFGS) and limited memory version L-BFGS)

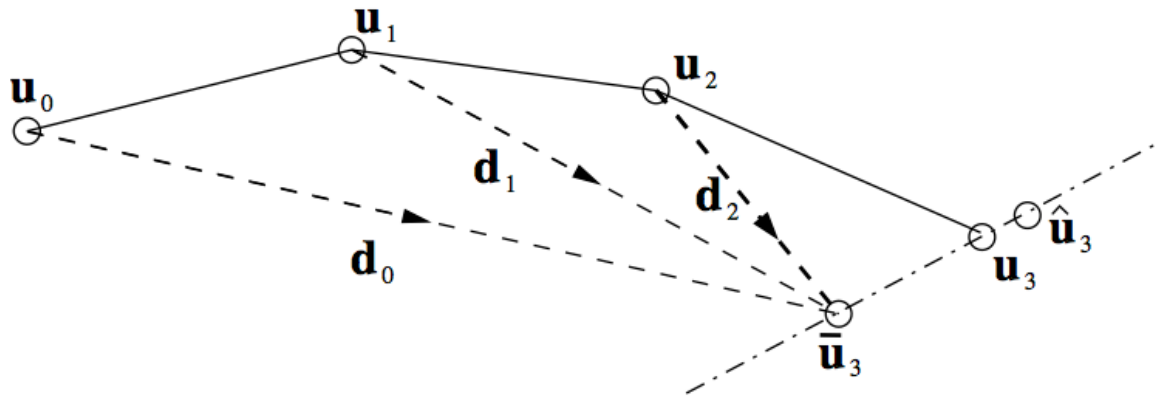
6. N-GMRES as a general optimization method

# 11. the power of N-GMRES optimization

- N-GMRES optimization method is a general, convergent method (steepest-descent preconditioning)
- its real power: N-GMRES optimization framework can employ sophisticated nonlinear preconditioners

STEP I: *(generate preliminary iterate by one-step update process $M(.)$)*
$\bar{\mathbf{u}}_{i+1} = M(\mathbf{u}_i)$
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)*
$\mathbf{u}_{i+1} = \text{linesearch}(\bar{\mathbf{u}}_{i+1} + \beta(\hat{\mathbf{u}}_{i+1} - \bar{\mathbf{u}}_{i+1}))$

# N-GMRES optimization algorithm to accelerate ALS




**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 $M(.)$)*
    $\bar{\mathbf{u}}_{i+1} = M(\mathbf{u}_i)$
  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)*
    $\mathbf{u}_{i+1} = \text{linesearch}(\bar{\mathbf{u}}_{i+1} + \beta(\hat{\mathbf{u}}_{i+1} - \bar{\mathbf{u}}_{i+1}))$
  $i = i + 1$
**until** *convergence criterion satisfied*

# differences with SVD

1. 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

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

# differences with SVD

2. SVD factor matrices are orthogonal

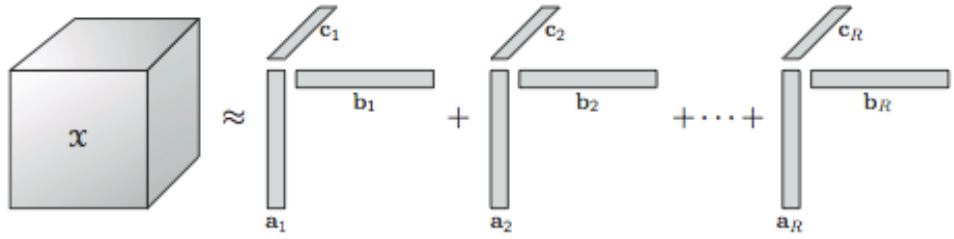
$$A = U \Sigma V^t \qquad U^t U = I_m \qquad V^t V = I_n$$

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

BUT best rank-*R* tensor factor matrices are not orthogonal

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

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



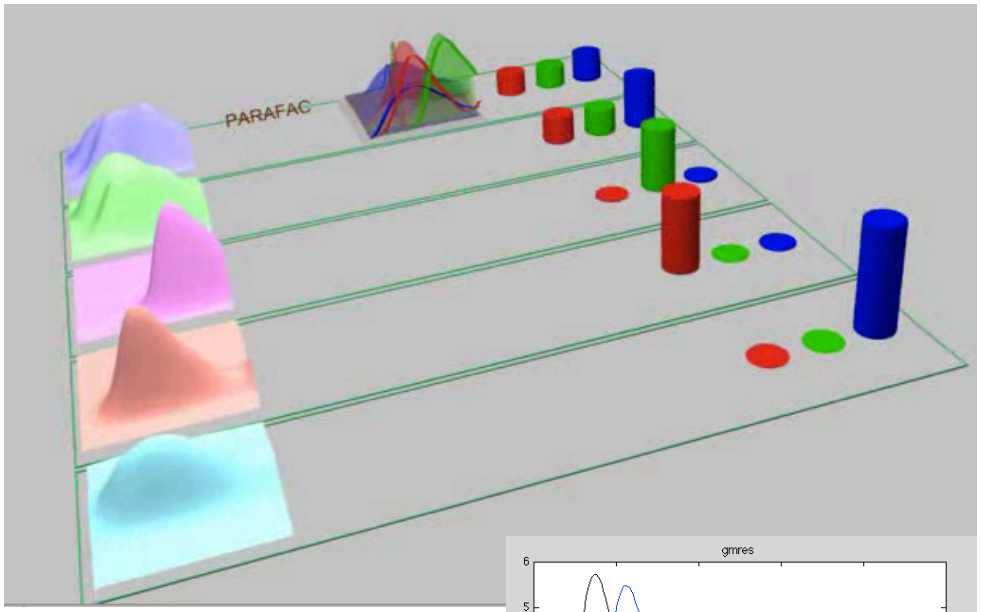

(from "Tensor Decompositions and Applications", Kolda and Bader, SIAM Rev., 2009 [1])

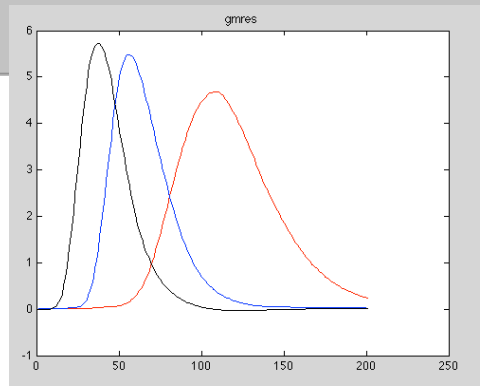
# tensor approximation applications

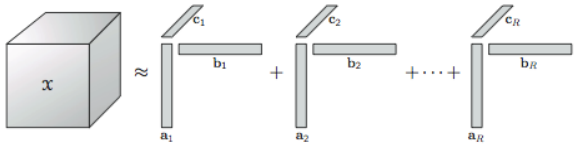
(3) chemometrics: analyze spectrofluorometer data (dense) (Bro et al.,

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)



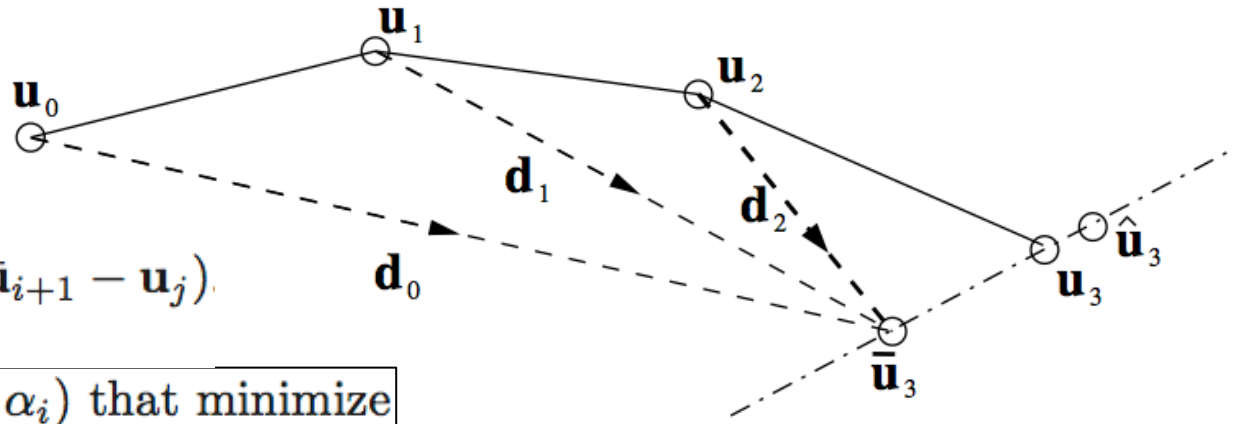




(from [1])

UNIVERSITY OF WATERLOO

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




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

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

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

$$\boldsymbol{\alpha} = (\alpha_0, \ldots, \alpha_i)^T,$$
$$\mathbf{p}_j = \mathbf{g}(\bar{\mathbf{u}}_{i+1}) - \mathbf{g}(\mathbf{u}_j),$$
$$\mathbf{P} = [\mathbf{p}_0 | \ldots | \mathbf{p}_j],$$

minimize $\|\mathbf{P}\,\boldsymbol{\alpha} + \mathbf{g}(\bar{\mathbf{u}}_{i+1})\|_2$

$$\mathbf{P}^T\mathbf{P}\,\boldsymbol{\alpha} = -\mathbf{P}^T\,\mathbf{g}(\bar{\mathbf{u}}_{i+1})$$

UNIVERSITY OF WATERLOO

# dense test problem: comparison

| $h^*$ accuracy $10^{-3}$ | | ALS | | N-GMRES | | N-CG | |
|---|---|---|---|---|---|---|---|
| problem parameters | | it | time | it | time | it | time |
| 1 | $s=20,\ c=0.5,\ R=3,\ l_1=1,\ l_2=1$ | 18 | **0.083** | 16 | 0.21 | 34 | 0.17 |
| 2 | $s=20,\ c=0.5,\ R=5,\ l_1=10,\ l_2=5$ | 9 | **0.083** | 8 | 0.17 | 64 | 0.51 |
| 3 | $s=20,\ c=0.9,\ R=3,\ l_1=0,\ l_2=0$ | 186 | 0.8 | 153 | 1.7 | 137 | **0.57** |
| 4 | $s=20,\ c=0.9,\ R=5,\ l_1=1,\ l_2=1$ | 19 | **0.15** | 13 | 0.34 | 195 | 1.4 |
| 5 | $s=50,\ c=0.5,\ R=3,\ l_1=1,\ l_2=1$ | 11 | **0.089** | 8 | 0.21 | 38 | 0.46 |
| 6 | $s=50,\ c=0.5,\ R=5,\ l_1=10,\ l_2=5$ | 10 | **0.15** | 9 | 0.3 | 50 | 0.97 |
| 7 | $s=50,\ c=0.9,\ R=3,\ l_1=0,\ l_2=0$ | 314 | 2.2 | 56 | **1.6** | 200 | 1.8 |
| 8 | $s=50,\ c=0.9,\ R=5,\ l_1=1,\ l_2=1$ | 15 | **0.2** | 10 | 0.43 | >1821 | >32 |
| 9 | $s=100,\ c=0.5,\ R=3,\ l_1=1,\ l_2=1$ | 9 | **0.31** | 9 | 1.1 | 71 | 5.7 |
| 10 | $s=100,\ c=0.5,\ R=5,\ l_1=10,\ l_2=5$ | 15 | **0.68** | 13 | 2.2 | 66 | 7.5 |
| 11 | $s=100,\ c=0.9,\ R=3,\ l_1=0,\ l_2=0$ | 178 | 5.9 | 30 | **3.9** | 340 | 23 |
| 12 | $s=100,\ c=0.9,\ R=5,\ l_1=1,\ l_2=1$ | 12 | **0.52** | 9 | 1.7 | 260 | 24 |

TABLE 3.1

(gradients, test case and N-CG from “A scalable optimization approach for fitting canonical tensor decompositions” by Acar, Dunlavy and Kolda, Chemometrics, 2011)

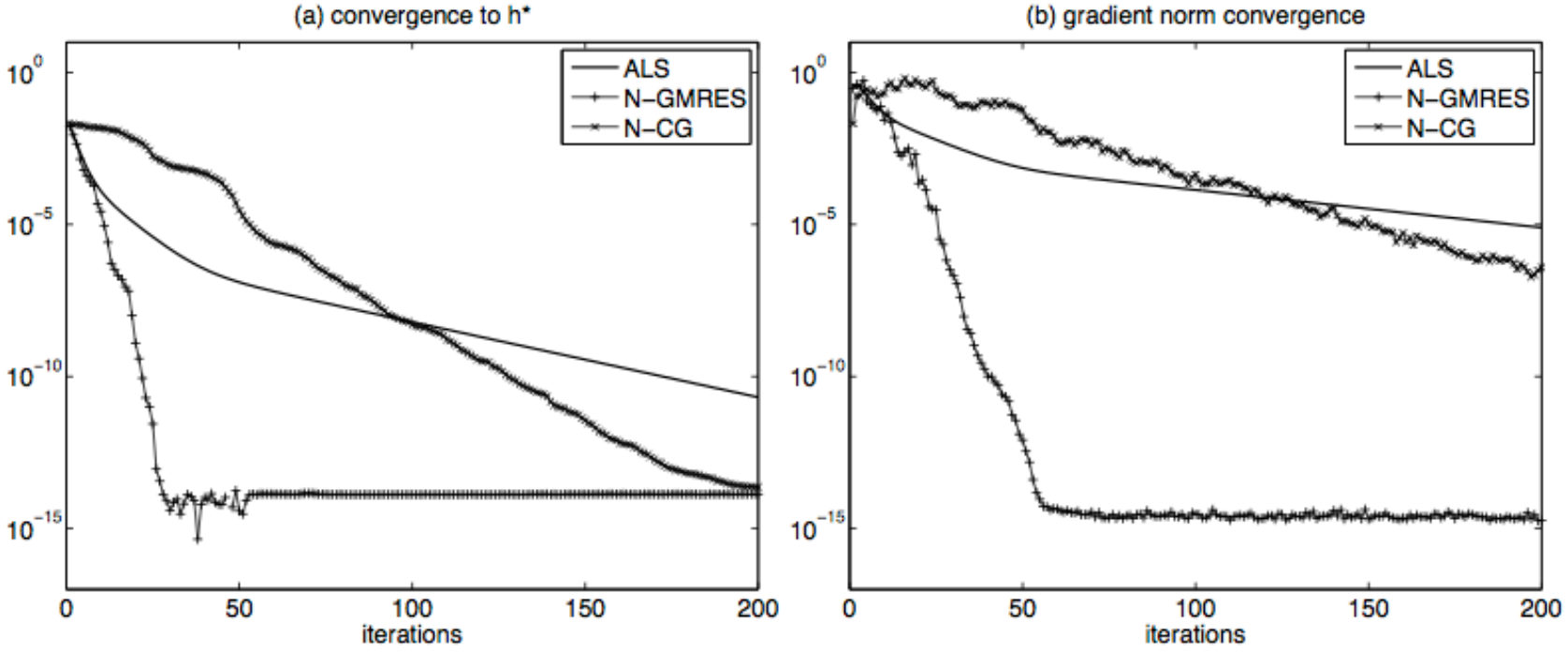
# dense test problem: comparison

| $h^*$ accuracy $10^{-10}$ | | ALS | | N-GMRES | | N-CG | |
|---|---|---|---|---|---|---|---|
| problem parameters | | it | time | it | time | it | time |
| 1 | $s=20,\ c=0.5,\ R=3,\ l_1=1,\ l_2=1$ | 37 | **0.16** | 22 | 0.3 | 52 | 0.24 |
| 2 | $s=20,\ c=0.5,\ R=5,\ l_1=10,\ l_2=5$ | 37 | **0.28** | 17 | 0.39 | 97 | 0.7 |
| 3 | $s=20,\ c=0.9,\ R=3,\ l_1=0,\ l_2=0$ | >1600 | >6.9 | 189 | **2.4** | >400 | >6.1 |
| 4 | $s=20,\ c=0.9,\ R=5,\ l_1=1,\ l_2=1$ | >1200 | >8.6 | 139 | **4.5** | 1100 | 6.8 |
| 5 | $s=50,\ c=0.5,\ R=3,\ l_1=1,\ l_2=1$ | 32 | **0.23** | 16 | 0.42 | 67 | 0.69 |
| 6 | $s=50,\ c=0.5,\ R=5,\ l_1=10,\ l_2=5$ | 36 | **0.44** | 17 | 0.67 | 89 | 1.6 |
| 7 | $s=50,\ c=0.9,\ R=3,\ l_1=0,\ l_2=0$ | >1200 | >8.5 | 104 | **3.5** | >553 | >7.6 |
| 8 | $s=50,\ c=0.9,\ R=5,\ l_1=1,\ l_2=1$ | 1252 | 14 | 171 | **10** | >1821 | >32 |
| 9 | $s=100,\ c=0.5,\ R=3,\ l_1=1,\ l_2=1$ | 31 | **1** | 16 | 2 | 136 | 9.6 |
| 10 | $s=100,\ c=0.5,\ R=5,\ l_1=10,\ l_2=5$ | 42 | **1.8** | 22 | 4.1 | 178 | 16 |
| 11 | $s=100,\ c=0.9,\ R=3,\ l_1=0,\ l_2=0$ | >800 | >27 | 99 | **17** | >748 | >60 |
| 12 | $s=100,\ c=0.9,\ R=5,\ l_1=1,\ l_2=1$ | 1218 | 51 | 112 | **26** | 880 | 72 |

TABLE 3.3

# numerical results: sparse test problem

- sparse test problem: d-dimensional finite difference Laplacian (2 d-way tensor)




UNIVERSITY OF WATERLOO

# sparse test problem: comparison

| $h^*$ accuracy $10^{-10}$ | | ALS | | N-GMRES | | N-CG | |
|---|---|---|---|---|---|---|---|
| problem parameters | | it | time | it | time | it | time |
| 1 | $N=4,\ s=8,\ R=6$ | >400 | >9.6 | 55 | **3.1** | 380 | 3.7 |
| 2 | $N=4,\ s=8,\ R=6$ | 242 | 5.8 | 26 | **1.5** | 327 | 3.5 |
| 3 | $N=4,\ s=16,\ R=3$ | >800 | >12 | 119 | 3.8 | 419 | **3.5** |
| 4 | $N=4,\ s=16,\ R=3$ | 724 | 11 | 84 | **2.7** | 375 | 3.2 |
| 5 | $N=6,\ s=4,\ R=2$ | 52 | 0.94 | 19 | **0.65** | 153 | 1.6 |
| 6 | $N=6,\ s=4,\ R=2$ | 51 | 0.95 | 18 | **0.67** | 386 | 3.3 |
| 7 | $N=6,\ s=8,\ R=5$ | 613 | 24 | 81 | **18** | 213 | 40 |
| 8 | $N=6,\ s=8,\ R=5$ | 127 | **5.1** | 31 | 6.8 | 262 | 46 |
| 9 | $N=8,\ s=4,\ R=2$ | 70 | 2 | 21 | **1.5** | 111 | 5.2 |
| 10 | $N=8,\ s=4,\ R=2$ | 72 | 2.1 | 24 | **1.8** | >280 | >19 |

TABLE 4.3

# 6. why does this work: GMRES

$$\mathbf{A}\,\mathbf{u} = \mathbf{b}$$

$$\mathbf{u}_{i+1} = \mathbf{u}_i + \mathbf{M}^{-1}\,\mathbf{r}_i$$

(Washio and Oosterlee, ETNA, 1997)

$$\begin{aligned}
V_{1,i+1} &= span\{\mathbf{r}_0, \ldots, \mathbf{r}_i\}, \\
V_{2,i+1} &= span\{\mathbf{r}_0, \mathbf{A}\mathbf{M}^{-1}\,\mathbf{r}_0, (\mathbf{A}\mathbf{M}^{-1})^2\,\mathbf{r}_0\}, \ldots, (\mathbf{A}\mathbf{M}^{-1})^i\,\mathbf{r}_0\} \\
&= K_{i+1}(\mathbf{A}\mathbf{M}^{-1}, \mathbf{r}_0), \\
V_{3,i+1} &= span\{\mathbf{M}\,(\mathbf{u}_1 - \mathbf{u}_0), \mathbf{M}\,(\mathbf{u}_2 - \mathbf{u}_1), \ldots, \mathbf{M}\,(\mathbf{u}_{i+1} - \mathbf{u}_i)\}, \\
V_{4,i+1} &= span\{\mathbf{M}\,(\mathbf{u}_{i+1} - \mathbf{u}_0), \mathbf{M}\,(\mathbf{u}_{i+1} - \mathbf{u}_1), \ldots, \mathbf{M}\,(\mathbf{u}_{i+1} - \mathbf{u}_i)\}
\end{aligned}$$

- N-GMRES step II reduces to preconditioned GMRES in the linear case $\hat{\mathbf{u}}_{i+1} = \bar{\mathbf{u}}_{i+1} + \sum_{j=0}^{i} \alpha_j\,(\bar{\mathbf{u}}_{i+1} - \mathbf{u}_j)$
- 'nonlinear Krylov space' $span\{(\mathbf{u}_{i+1} - \mathbf{u}_0), (\mathbf{u}_{i+1} - \mathbf{u}_1), \ldots, (\mathbf{u}_{i+1} - \mathbf{u}_i)\}$
- $\bar{\mathbf{u}}_{i+1} = M(\mathbf{u}_i)$ in step I is a nonlinear preconditioner for N-GMRES (ALS)

STEP I: *(generate preliminary iterate by one-step update process $M(.)$)*
$\bar{\mathbf{u}}_{i+1} = M(\mathbf{u}_i)$
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)*
$\mathbf{u}_{i+1} = \text{linesearch}(\bar{\mathbf{u}}_{i+1} + \beta(\hat{\mathbf{u}}_{i+1} - \bar{\mathbf{u}}_{i+1}))$

UNIVERSITY OF WATERLOO

# numerical results: steepest-descent preconditioning

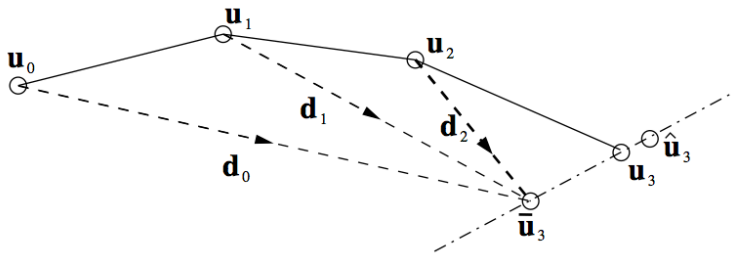
| problem | N-GMRES-sdls | N-GMRES-sd | N-CG | L-BFGS |
|---|---|---|---|---|
| D $n$=500 | 525 | 172 | 222 | 166 |
| D $n$=1000 | 445 | 211 | 223 | 170 |
| E $n$=100 | 294 | 259 | 243 | 358 |
| E $n$=200 | 317 | 243 | 240 | 394 |
| F $n$=200 | 140 | 102(1) | 102 | 92 |
| F $n$=500 | 206(1) | 175(1) | 135 | 118 |
| G $n$=100 | 1008(2) | 152 | 181 | 358 |
| G $n$=200 | 629(1) | 181 | 137 | 240 |

TABLE 3.2

- standard test problems, 10 random initial guesses
- N-GMRES with steepest descent preconditioning is competitive with N-CG and L-BFGS
- N-GMRES preconditioner option A (line search) slower than option B (small step)

# 12. conclusions

- we have proposed the 3-step preconditioned N-GMRES optimization algorithm as a general nonlinear optimization method (smooth *f(u)*, unconstrained) (uncommon approach, new in optimization?)
- steepest descent preconditioning is the natural 'default' preconditioner, it makes N-GMRES competitive with N-CG and L-BFGS, and we have proved global convergence



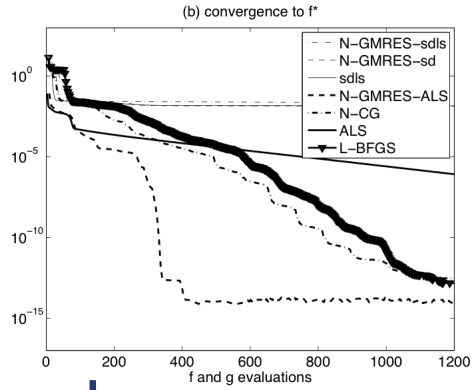

**Algorithm 1**: N-GMRES optimization algorithm (window size $w$)

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

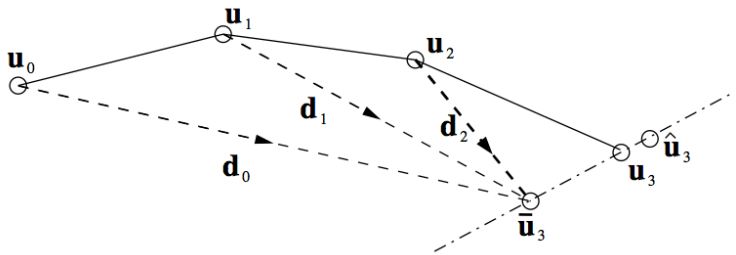
$i = w - 1$
**repeat**
  STEP I: *(generate preliminary iterate by one-step update process $M(.)$)*
    $\bar{\mathbf{u}}_{i+1} = M(\mathbf{u}_i)$
  STEP II: *(generate accelerated iterate by nonlinear GMRES step)*
    $\hat{\mathbf{u}}_{i+1} = \text{gmres}(\mathbf{u}_{i-w+1}, \ldots, \mathbf{u}_i; \bar{\mathbf{u}}_{i+1})$
  STEP III: *(generate new iterate by line search process)*
    $\mathbf{u}_{i+1} = \text{linesearch}(\bar{\mathbf{u}}_{i+1} + \beta(\hat{\mathbf{u}}_{i+1} - \bar{\mathbf{u}}_{i+1}))$
  $i = i + 1$
**until** *convergence criterion satisfied*

UNIVERSITY OF WATERLOO

# conclusions




- the real power of the N-GMRES optimization framework is that advanced nonlinear preconditioners can be used
- ALS-preconditioned N-GMRES optimization performs very well for tensor optimization problem




**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 $M(.)$)*
    $\bar{\mathbf{u}}_{i+1} = M(\mathbf{u}_i)$
  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)*
    $\mathbf{u}_{i+1} = \text{linesearch}(\bar{\mathbf{u}}_{i+1} + \beta(\hat{\mathbf{u}}_{i+1} - \bar{\mathbf{u}}_{i+1}))$
  $i = i + 1$
**until** *convergence criterion satisfied*

UNIVERSITY OF WATERLOO