Efficient Solution of Sequences of Linear Systems

Jurjen Duintjer Tebbens

Institute of Computer Science

Academy of Sciences of the Czech Republic

Miroslav Tůma

Institute of Computer Science

Academy of Sciences of the Czech Republic

August 14, 2009
Hong Kong Baptist University

Motivation: Example I

1. Solving systems of nonlinear equations

$$F(x) = 0$$

$$\Downarrow$$

Sequences of linear algebraic systems of the form

$$J(x_k)(x_{k+1} - x_k) = -F(x_k),\ J(x_k) \approx F'(x_k)$$

Motivation: Example I

1. Solving systems of nonlinear equations

$$F(x) = 0$$

$$\Downarrow$$

Sequences of linear algebraic systems of the form

$$J(x_k)(x_{k+1} - x_k) = -F(x_k),\ \ J(x_k) \approx F'(x_k)$$

solved until for some $k, k = 1, 2, \ldots$

$$\|F(x_k)\| < tol$$

$J(x_k)$ may change both structurally and numerically

Motivation: Examples II and III

2. Solving equations with a parabolic term

$$\frac{\partial u}{\partial t} - \Delta u = f$$

Motivation: Examples II and III

2. Solving equations with a parabolic term

$$\frac{\partial u}{\partial t} - \Delta u = f$$

$$\Downarrow$$

diagonal changes in the sequence of linear systems

Motivation: Examples II and III

2. Solving equations with a parabolic term

$$\frac{\partial u}{\partial t} - \Delta u = f$$

$$\Downarrow$$

diagonal changes in the sequence of linear systems

3. Nonlinear convection-diffusion problems

$$-\Delta u + u\nabla u = f$$

$$\Downarrow$$

more general sequences of linear systems, upwind discretizations, anisotropy: possibly more structural nonsymmetry

Motivation: Our goals

- The talk considers a general sequence of linear systems

$$A^{(i)}x = b^{(i)}, \quad i = 1, 2, \ldots$$

Motivation: Our goals

- The talk considers a general sequence of linear systems

$$A^{(i)}x = b^{(i)}, \quad i = 1, 2, \ldots$$

- Such sequences arise in numerous applications like CFD problems, operation research problems, Helmholtz equations, ...

Motivation: Our goals

- The talk considers a general sequence of linear systems

$$A^{(i)}x = b^{(i)}, \quad i = 1, 2, \ldots$$

- Such sequences arise in numerous applications like CFD problems, operation research problems, Helmholtz equations, ...
- The central question for efficient solution of *sequences* of linear systems is:

Motivation: Our goals

- The talk considers a general sequence of linear systems

$$A^{(i)}x = b^{(i)}, \quad i = 1, 2, \ldots$$

- Such sequences arise in numerous applications like CFD problems, operation research problems, Helmholtz equations, ...
- The central question for efficient solution of *sequences* of linear systems is:
- How can we share a part of the computational effort throughout the sequence ?

Outline

Outline

1. Our goal and a short summary of related work
2. The basic triangular updates
3. Triangular updates in matrix-free environment
4. An alternative strategy for matrix-free environment

Our goal and some related work

Our goal

We concentrate on sequences arising from Newton-type iterations to solve nonlinear equations,

$$J(x_k)(x_{k+1} - x_k) = -F(x_k), \quad k = 1, 2, \ldots$$

where $J(x_k)$ is the Jacobian of F evaluated at x_k.

Our goal and some related work

Our goal

We concentrate on sequences arising from Newton-type iterations to solve nonlinear equations,

$$J(x_k)(x_{k+1} - x_k) = -F(x_k), \quad k = 1, 2, \ldots$$

where $J(x_k)$ is the Jacobian of F evaluated at x_k.

- A black-box approximate preconditioner update designed for *nonsymmetric* linear systems solved by arbitrary iterative methods.

Our goal and some related work

Our goal

We concentrate on sequences arising from Newton-type iterations to solve nonlinear equations,

$$J(x_k)(x_{k+1} - x_k) = -F(x_k), \quad k = 1, 2, \ldots$$

where $J(x_k)$ is the Jacobian of F evaluated at x_k.

- A black-box approximate preconditioner update designed for *nonsymmetric* linear systems solved by arbitrary iterative methods.
- Its computation should be much cheaper than the computation of a new preconditioner.

Our goal and some related work

Our goal

We concentrate on sequences arising from Newton-type iterations to solve nonlinear equations,

$$J(x_k)(x_{k+1} - x_k) = -F(x_k), \quad k = 1, 2, \ldots$$

where $J(x_k)$ is the Jacobian of F evaluated at x_k.

- A black-box approximate preconditioner update designed for *nonsymmetric* linear systems solved by arbitrary iterative methods.
- Its computation should be much cheaper than the computation of a new preconditioner.
- Interested in its behaviour in matrix-free environment: effort to decrease counts of matvecs to compute the subsequent systems.

Our goal and some related work: II.

Short summary of related work

- Freezing approximate Jacobians (using the same approximate Jacobian) over a couple of subsequent systems and, in this way, skipping some evaluations of the approximate Jacobian (MNK: Shamanskii, 1967; Brent, 1973).

Our goal and some related work: II.

Short summary of related work

- Freezing approximate Jacobians (using the same approximate Jacobian) over a couple of subsequent systems and, in this way, skipping some evaluations of the approximate Jacobian (MNK: Shamanskii, 1967; Brent, 1973).
- Freezing preconditioners over a couple of subsequent systems (periodic recomputation) (MFNK: Knoll, McHugh, 1998; Knoll, Keyes, 2004). Naturally, a frozen preconditioner will deteriorate when the system matrix changes too much.

Our goal and some related work: II.

Short summary of related work

- Freezing approximate Jacobians (using the same approximate Jacobian) over a couple of subsequent systems and, in this way, skipping some evaluations of the approximate Jacobian (MNK: Shamanskii, 1967; Brent, 1973).
- Freezing preconditioners over a couple of subsequent systems (periodic recomputation) (MFNK: Knoll, McHugh, 1998; Knoll, Keyes, 2004). Naturally, a frozen preconditioner will deteriorate when the system matrix changes too much.
- Physics-based preconditioners (preconditioning by discretized simpler operators like scaled diffusion operators for convection-diffusion equations and/or using fast solvers; using other physics-based operator splittings; using symmetric parts of matrices) (only a selection from huge bibliography: Concus, Golub, 1973; Elman, Schultz, 1986; Brown, Saad, 1990; Knoll, McHugh, 1995; Knoll, Keyes, 2004)

Our goal and some related work: III.

- In Quasi-Newton methods the difference between system matrices is of small rank and preconditioners may be efficiently adapted with approximate small-rank preconditioner updates; see e.g. Bergamaschi, Bru, Martínez, Putti (2006); Nocedal, Morales, 2000.

Our goal and some related work: III.

- In Quasi-Newton methods the difference between system matrices is of small rank and preconditioners may be efficiently adapted with approximate small-rank preconditioner updates; see e.g. Bergamaschi, Bru, Martínez, Putti (2006); Nocedal, Morales, 2000.
- Further bunch of possible ways: Krylov subspace recycling, exact updates of LU-factorizations, etc ...

Our goal and some related work: III.

- In Quasi-Newton methods the difference between system matrices is of small rank and preconditioners may be efficiently adapted with approximate small-rank preconditioner updates; see e.g. Bergamaschi, Bru, Martínez, Putti (2006); Nocedal, Morales, 2000.
- Further bunch of possible ways: Krylov subspace recycling, exact updates of LU-factorizations, etc ...
- To enhance the power of a frozen preconditioner one may also use approximate preconditioner updates.

Our goal and some related work: III.

- In Quasi-Newton methods the difference between system matrices is of small rank and preconditioners may be efficiently adapted with approximate small-rank preconditioner updates; see e.g. Bergamaschi, Bru, Martínez, Putti (2006); Nocedal, Morales, 2000.
- Further bunch of possible ways: Krylov subspace recycling, exact updates of LU-factorizations, etc ...
- To enhance the power of a frozen preconditioner one may also use approximate preconditioner updates.
  - Approximate preconditioner updates of incomplete Cholesky factorizations, Meurant, 2001.
  - banded preconditioner updates were proposed for both symmetric positive definite approximate inverse and incomplete Cholesky preconditioners, Benzi, Bertaccini, 2003, 2004.
  - Approximate preconditioner updates based on approximate inverses are considered in Calgaro, Chehab, Saad, 2009.

Outline

The considered preconditioner updates: I.

Triangular updates from Duintjer Tebbens, T., 2007

Consider two systems

$$Ax = b, \qquad \text{and} \qquad A^+x^+ = b^+$$

preconditioned by M and M^+ respectively and let the difference matrix be defined as

$$B \equiv A - A^+.$$

The considered preconditioner updates: I.

Triangular updates from Duintjer Tebbens, T., 2007

Consider two systems

$$Ax = b, \qquad \text{and} \qquad A^+ x^+ = b^+$$

preconditioned by M and M^+ respectively and let the difference matrix be defined as

$$B \equiv A - A^+.$$

Define the standard splitting

$$B = L_B + D_B + U_B$$

The considered preconditioner updates: I.

Triangular updates from Duintjer Tebbens, T., 2007

Consider two systems

$$Ax = b, \qquad \text{and} \qquad A^+ x^+ = b^+$$

preconditioned by M and M^+ respectively and let the difference matrix be defined as

$$B \equiv A - A^+.$$

Define the standard splitting

$$B = L_B + D_B + U_B$$

and let M be factorized as

$$M = LDU \approx A.$$

The considered preconditioner updates: I.

Triangular updates from Duintjer Tebbens, T., 2007

Consider two systems

$$Ax = b, \qquad \text{and} \qquad A^+ x^+ = b^+$$

preconditioned by M and M^+ respectively and let the difference matrix be defined as

$$B \equiv A - A^+.$$

Define the standard splitting

$$B = L_B + D_B + U_B$$

and let M be factorized as

$$M = LDU \approx A.$$

We would like M^+ to be an update of M that is similarly powerful.

2. The considered preconditioner updates: II.

In particular, if $||A - M||$ is the accuracy of the preconditioner M for A, we will try to find an updated M^+ for A^+ with comparable accuracy,

2. The considered preconditioner updates: II.

In particular, if $||A - M||$ is the accuracy of the preconditioner M for A, we will try to find an updated M^+ for A^+ with comparable accuracy,

Lemma

Let $||A - LDU|| = \varepsilon||A|| < ||B||$. *Then* $M^+ = L(\overline{DU - B})$ *with* $\overline{DU - B} \approx DU - B$ *satisfies*

$$\begin{aligned} || & A^+ - M^+|| \leq \\ \leq & \frac{\|L\| \, \|DU - B - \overline{DU - B}\| + ||L - I|| \, \|B\| + \varepsilon||A||}{||B|| - \varepsilon||A||} \cdot ||A^+ - LDU||. \end{aligned}$$

2. The considered preconditioner updates: II.

In particular, if $||A - M||$ is the accuracy of the preconditioner M for A, we will try to find an updated M^+ for A^+ with comparable accuracy,

Lemma

Let $||A - LDU|| = \varepsilon||A|| < ||B||$. *Then* $M^+ = L(\overline{DU - B})$ *with* $\overline{DU - B} \approx DU - B$ *satisfies*

$$\begin{aligned} || \quad & A^+ - M^+|| \leq \\ \leq \quad & \frac{\|L\|\,\|DU - B - \overline{DU - B}\| + ||L - I||\,\|B\| + \varepsilon||A||}{||B|| - \varepsilon||A||} \cdot ||A^+ - LDU||. \end{aligned}$$

- $\overline{DU - B}$ should be close to $DU - B$. Similar lemma with $\overline{LD - B}$.
- $||L - I||$ should be small
- $||M^+ - A^+||$ can be even smaller than $||M - A||$.
- Possible to state some (weak) existence results.

The considered preconditioner updates: III.

We propose the preconditioner update of the form

$$M^+ \equiv (LD - L_B - D_B)U \quad \text{or} \quad M^+ \equiv L(DU - D_B - U_B).$$

That is, $\overline{DU - B} = DU - D_B - U_B$, $\overline{LD - B} = LD - L_B - D_B$.

The considered preconditioner updates: III.

We propose the preconditioner update of the form

$$M^+ \equiv (LD - L_B - D_B)U \quad \text{or} \quad M^+ \equiv L(DU - D_B - U_B).$$

That is, $\overline{DU - B} = DU - D_B - U_B$, $\overline{LD - B} = LD - L_B - D_B$.

Note that M^+ is for free and its application asks for one forward and one backward solve. Schematically,

type	initialization	solve step	memory
Recomp	$A^+ \approx L^+U^+$	solves with L^+, U^+	A^+, L^+, U^+
Update	—	solves with $L, U, triu(B)$	$A^+, triu(A), L, U$

The considered preconditioner updates: III.

We propose the preconditioner update of the form

$$M^+ \equiv (LD - L_B - D_B)U \quad \text{or} \quad M^+ \equiv L(DU - D_B - U_B).$$

That is, $\overline{DU - B} = DU - D_B - U_B$, $\overline{LD - B} = LD - L_B - D_B$.

Note that M^+ is for free and its application asks for one forward and one backward solve. Schematically,

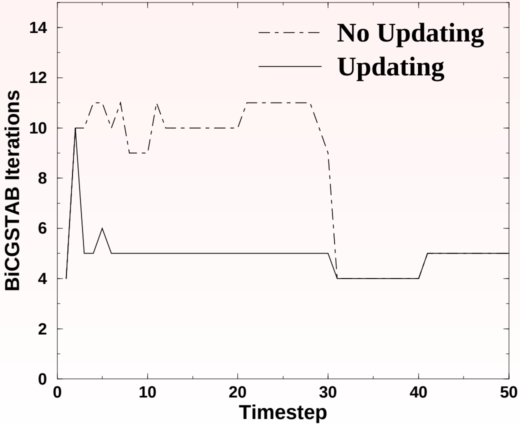
type	initialization	solve step	memory
Recomp	$A^+ \approx L^+U^+$	solves with L^+, U^+	A^+, L^+, U^+
Update	—	solves with $L, U, triu(B)$	$A^+, triu(A), L, U$

- This is the basic idea; more sophisticated improvements are possible
- Ideal for upwind/downwind modification but our experiments cover broader spectrum of problems

Some experimental results with the updates

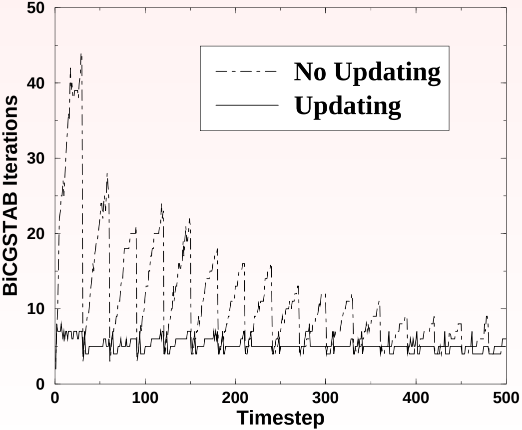
(Birken, Duintjer Tebens, Meister, T., 2007)

flow around a NACA0012 airfoil, angle of attack: 2^0,
4605 cells of FVM (n=18420), Mach 0.8



Some experimental results with the updates: II.

supersonic flow around a cylinder, FVM, n=83976, 3000 steps of implicit Euler method



Possible improvements

Gauss-Seidel type updates

$$L(DU + D_B + U_B)),\ (LD + D_L + L_B)U$$

$$\downarrow$$

$$L(L_C + D_C)D_C^{-1}(U_C + D_C),\ C = DU - B$$
$$(L_C + D_C)D_C^{-1}(U_C + D_C)U,\ C = LD - B$$

Possible improvements

Gauss-Seidel type updates

$$L(DU + D_B + U_B)), \; (LD + D_L + L_B)U$$

$$\downarrow$$

$$L(L_C + D_C)D_C^{-1}(U_C + D_C), \; C = DU - B$$
$$(L_C + D_C)D_C^{-1}(U_C + D_C)U, \; C = LD - B$$

- switching formulas based on norms $||I - L||$ and $||I - U||$

Possible improvements

Gauss-Seidel type updates

$$L(DU + D_B + U_B)), \ (LD + D_L + L_B)U$$

$$\downarrow$$

$$L(L_C + D_C)D_C^{-1}(U_C + D_C), \ C = DU - B$$
$$(L_C + D_C)D_C^{-1}(U_C + D_C)U, \ C = LD - B$$

- switching formulas based on norms $||I - L||$ and $||I - U||$
- implementation: sparse merge of triangular matrices

Possible improvements

Gauss-Seidel type updates

$$L(DU + D_B + U_B)),\ (LD + D_L + L_B)U$$

$$\downarrow$$

$$L(L_C + D_C)D_C^{-1}(U_C + D_C),\ C = DU - B$$
$$(L_C + D_C)D_C^{-1}(U_C + D_C)U,\ C = LD - B$$

- switching formulas based on norms $||I - L||$ and $||I - U||$
- implementation: sparse merge of triangular matrices
- possible parametrized relaxation of the updated.

Possible improvements

Gauss-Seidel type updates

$$L(DU + D_B + U_B)), \ (LD + D_L + L_B)U$$

$$\downarrow$$

$$L(L_C + D_C)D_C^{-1}(U_C + D_C), \ C = DU - B$$
$$(L_C + D_C)D_C^{-1}(U_C + D_C)U, \ C = LD - B$$

- switching formulas based on norms $||I - L||$ and $||I - U||$
- implementation: sparse merge of triangular matrices
- possible parametrized relaxation of the updated.
- some related theory based on decay properties/diagonal dominance and/or ILU(0) preconditioner.

Possible improvements

Gauss-Seidel type updates

$$L(DU + D_B + U_B)),\ (LD + D_L + L_B)U$$

$$\downarrow$$

$$L(L_C + D_C)D_C^{-1}(U_C + D_C),\ C = DU - B$$
$$(L_C + D_C)D_C^{-1}(U_C + D_C)U,\ C = LD - B$$

- switching formulas based on norms $||I - L||$ and $||I - U||$
- implementation: sparse merge of triangular matrices
- possible parametrized relaxation of the updated.
- some related theory based on decay properties/diagonal dominance and/or ILU(0) preconditioner.
- See Duintjer Tebbens, T., 2007a

Experimental results: driven cavity problem

Example: Driven cavity problem

$$\Delta\Delta u + R\left(\frac{\partial u}{\partial y}\frac{\partial \Delta u}{\partial x} - \frac{\partial u}{\partial x}\frac{\partial \Delta u}{\partial y}\right) = 0,$$

- 2D on unit square
- 13-point finite differences
- $u = 0$ on $\partial\Omega$ and $\partial u(0,y)/\partial x = 0$, $\partial u(1,y)/\partial x = 0$, $\partial u(x,0)/\partial x = 0$ and $\partial u(x,1)/\partial x = 1$
- modest Reynolds numbers R in order to avoid potential discretization problems

Experimental results: driven cavity problem: II.

Table: *Driven cavity problem with R=50, n=2500, nnz=31504, ILU*(0.01).

ILU(0.01), psize ≈ 47000					
Matrix	Recomp	Freeze	Str. upd.	Unstr. upd.	Unstr. upd.$_{fr}$
$A^{(0)}$	93	93	93	93	93
$A^{(1)}$	269	88	88	177	177
$A^{(2)}$	> 500	242	156	391	245
$A^{(3)}$	> 500	196	179	266	230
$A^{(4)}$	> 500	284	298	227	202
$A^{(5)}$	> 500	> 500	144	221	202
$A^{(6)}$	> 500	306	132	210	226
overall time	∞	∞	7 s	17 s	11 s

Experimental results: driven cavity problem: III.

Table: *Driven cavity problem with R=10, n=2500, nnz=31504, ILU(0.01).*

ILU(0.01), psize ≈ 47000					
Matrix	Recomp	Freeze	Str. upd.	Unstr. upd.	Unstr. upd.$_{fr}$
$A^{(0)}$	84	84	84	84	84
$A^{(1)}$	84	91	95	180	180
$A^{(2)}$	312	239	119	197	180
$A^{(3)}$	261	155	119	222	227
$A^{(4)}$	352	> 500	190	165	171
$A^{(5)}$	259	266	163	170	167
$A^{(6)}$	291	262	150	182	182
overall time	12 s	∞	7 s	16 s	10 s

Outline

Matrix-free updates and matvecs

- Krylov subspace methods do not require the system to be stored explicitly; a matrix-vector product (matvec) subroutine, based on a function evaluation, suffices.
- $\Rightarrow$ important reduction of storage and computational costs.

Matrix-free updates and matvecs

- Krylov subspace methods do not require the system to be stored explicitly; a matrix-vector product (matvec) subroutine, based on a function evaluation, suffices.
- $\Rightarrow$ important reduction of storage and computational costs.

- Standard example: Newton iteration of the form

$$J(x_k)(x_{k+1} - x_k) = -F(x_k), \quad k = 1, 2, \ldots$$

  where $J(x_k)$ is the Jacobian of F evaluated at x_k.
- Matvec with $J(x_k)$ is replaced by the standard difference approximation,

$$J(x_k) \cdot v \approx \frac{F(x_k + h\|x_k\|v) - F(x_k)}{h\|x_k\|},$$

  for some small h.

Matrix-free updates and matrix estimation

- In order to compute an incomplete factorization in matrix-free environment, the system matrix has to be *estimated by matvecs*

Matrix-free updates and matrix estimation

- In order to compute an incomplete factorization in matrix-free environment, the system matrix has to be *estimated by matvecs*
- We need to know its sparsity pattern

Matrix-free updates and matrix estimation

- In order to compute an incomplete factorization in matrix-free environment, the system matrix has to be *estimated by matvecs*
- We need to know its sparsity pattern
- Efficient estimation of a banded matrix

$$
\begin{pmatrix}
\spadesuit & * & & & & & & \\
\spadesuit & * & * & & & & & \\
& * & * & \spadesuit & & & & \\
& & * & \spadesuit & * & & & \\
& & & \spadesuit & * & * & & \\
& & & & * & * & \spadesuit & \\
& & & & & * & \spadesuit & * \\
& & & & & & \spadesuit & *
\end{pmatrix}
$$

Columns with "red spades" can be computed at the same time in one matvec since sparsity patterns of their rows do not overlap. Namely,

Matrix-free updates and matrix estimation

- In order to compute an incomplete factorization in matrix-free environment, the system matrix has to be *estimated by matvecs*
- We need to know its sparsity pattern
- Efficient estimation of a banded matrix

$$
\begin{pmatrix}
\spadesuit & * & & & & & & \\
\spadesuit & * & * & & & & & \\
& * & * & \spadesuit & & & & \\
& & * & \spadesuit & * & & & \\
& & & \spadesuit & * & * & & \\
& & & & * & * & \spadesuit & \\
& & & & & * & \spadesuit & * \\
& & & & & & \spadesuit & *
\end{pmatrix}
$$

Columns with “red spades” can be computed at the same time in one matvec since sparsity patterns of their rows do not overlap. Namely,

$A(e_1 + e_4 + e_7)$ computes entries in the columns 1, 4 and 7.

Matrix estimation: II.

How to approximate a matrix by small number of matvecs if we know matrix pattern:

Example 2: Efficient estimation of a general matrix

$$\begin{pmatrix} * & * & & * & & \\ * & * & & & * & \\ & * & * & & * & \\ * & & & * & * & \\ & & & * & * & * \\ & * & & & * & * \end{pmatrix}$$

Again, By one matvec can be computed the columns for which sparsity patterns of their rows do not overlap.

Matrix estimation: II.

How to approximate a matrix by small number of matvecs if we know matrix pattern:

Example 2: Efficient estimation of a general matrix

$$\begin{pmatrix} \spadesuit & * & & * & & \\ \spadesuit & * & & & * & \\ & * & \spadesuit & & * & \\ \spadesuit & & & * & * & \\ & & & * & * & \spadesuit \\ & * & & & * & \spadesuit \end{pmatrix}$$

Again, By one matvec can be computed the columns for which sparsity patterns of their rows do not overlap.

For example, $A(e_1 + e_3 + e_6)$ computes entries in the columns 1, 3 and 6.

How to approximate a matrix by small number of matvecs if we know matrix pattern:

Example 2: Efficient estimation of a general matrix

$$
\begin{pmatrix}
{\color{red}\spadesuit} & {\color{blue}\spadesuit} & & {\color{orange}\spadesuit} & & \\
{\color{red}\spadesuit} & {\color{blue}\spadesuit} & & & {\color{green}\spadesuit} & \\
 & {\color{blue}\spadesuit} & {\color{red}\spadesuit} & & {\color{green}\spadesuit} & \\
{\color{red}\spadesuit} & & & {\color{orange}\spadesuit} & {\color{green}\spadesuit} & \\
 & & & {\color{orange}\spadesuit} & {\color{green}\spadesuit} & {\color{red}\spadesuit} \\
 & {\color{blue}\spadesuit} & & & {\color{green}\spadesuit} & {\color{red}\spadesuit}
\end{pmatrix}
$$

Entries in A can be computed by four matvecs.

In each matvec we need to have structurally orthogonal columns.

Efficient matrix estimation: well established field

- Structurally orthogonal columns can be grouped

4. Matrix estimation: III.

Efficient matrix estimation: well established field

- Structurally orthogonal columns can be grouped
- Finding the minimum number of groups: combinatorially difficult problem (NP-hard)

4. Matrix estimation: III.

Efficient matrix estimation: well established field

- Structurally orthogonal columns can be grouped
- Finding the minimum number of groups: combinatorially difficult problem (NP-hard)
- Classical field: a (very restricted) selection of references: Curtis, Powell; Reid,1974; Coleman, Moré, 1983; Coleman, Moré, 1984; Coleman, Verma, 1998; Gebremedhin, Manne, Pothen, 2007.

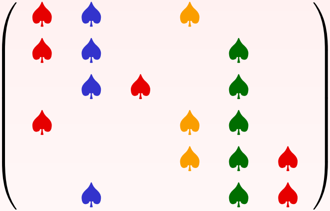
4. Matrix estimation: III.

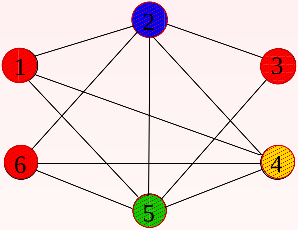
Efficient matrix estimation: well established field

- Structurally orthogonal columns can be grouped
- Finding the minimum number of groups: combinatorially difficult problem (NP-hard)
- Classical field: a (very restricted) selection of references: Curtis, Powell; Reid,1974; Coleman, Moré, 1983; Coleman, Moré, 1984; Coleman, Verma, 1998; Gebremedhin, Manne, Pothen, 2007.
  - extensions to SPD (Hessian) approximations
  - extensions to use both A and A^T in automatic differentiation
  - not only direct determination of resulting entries (substitution methods)

4. Matrix estimation: IV.

Efficient matrix estimation: graph coloring problem





- In the other words, columns which form an independent set in the graph of $A^T A$ (called intersection graph) can be grouped $\Rightarrow$ a graph coloring problem for the graph of $A^T A$.

Problem: Find a coloring of vertices of the graph of $A^T A$ $(G(A^T A))$ with minimum number of colors such that edges connect only vertices of different colors

General notes on matrix estimation

- The matrix estimation works even when the pattern is only approximate Cullum, T., 2006

General notes on matrix estimation

- The matrix estimation works even when the pattern is only approximate Cullum, T., 2006
- Recomputing the preconditioner requires for every linear system:

General notes on matrix estimation

- The matrix estimation works even when the pattern is only approximate Cullum, T., 2006
- Recomputing the preconditioner requires for every linear system:
  - Additional matvecs to estimate the current matrix

General notes on matrix estimation

- The matrix estimation works even when the pattern is only approximate Cullum, T., 2006
- Recomputing the preconditioner requires for every linear system:
  - Additional matvecs to estimate the current matrix
  - Rerunning the graph coloring algorithm if the nonzero pattern changes during the sequence. Not always necessary - in our experiments we do not use it.

General notes on matrix estimation

- The matrix estimation works even when the pattern is only approximate Cullum, T., 2006
- Recomputing the preconditioner requires for every linear system:
  - Additional matvecs to estimate the current matrix
  - Rerunning the graph coloring algorithm if the nonzero pattern changes during the sequence. Not always necessary - in our experiments we do not use it.

How about the preconditioner updates?

Updates and matrix-free environment

Recall the upper triangular update is of the form

$$M^+ = L(DU - D_B - U_B)$$

based on the splitting

$$L_B + D_B + U_B = B = A - A^+.$$

Updates and matrix-free environment

Recall the upper triangular update is of the form

$$M^+ = L(DU - D_B - U_B)$$

based on the splitting

$$L_B + D_B + U_B = B = A - A^+.$$

Thus the update needs some entries of A and A^+ and repeated estimation is necessary.

Updates and matrix-free environment

Recall the upper triangular update is of the form

$$M^+ = L(DU - D_B - U_B)$$

based on the splitting

$$L_B + D_B + U_B = B = A - A^+.$$

Thus the update needs some entries of A and A^+ and repeated estimation is necessary.

However:

- A has been estimated before to obtain the reference ILU-factorization
- Of A^+ we need estimate only the upper (lower) triangular part
- Can there be taken any advantage of the fact we estimate only the upper triangular part?

Updates in matrix-free environment

Example:

$$\begin{pmatrix} * & & & & * \\ & * & & & * \\ & & \ddots & & \vdots \\ & & & \ddots & * \\ * & * & * & * & * \end{pmatrix}$$

- estimating the whole matrix asks for n matvecs with all unit vectors;
- estimating the upper triangular part requires only 2 matvecs,

$$(1, \ldots, 1, 0)^T \qquad \text{and} \qquad (0, \ldots, 0, 1)^T.$$

The problem of estimating only the upper triangular part is an example of the *partial graph coloring problem*, Pothen et al., 2007.

Updates in matrix-free environment: II.

- Let us remind that the graph coloring algorithm for a matrix C works on the *intersection graph*
$$G(C^T C).$$

Updates in matrix-free environment: II.

- Let us remind that the graph coloring algorithm for a matrix C works on the *intersection graph*
$$G(C^T C).$$

- We intend to estimate only a matrix triangular part $triu(C)$.

Updates in matrix-free environment: II.

- Let us remind that the graph coloring algorithm for a matrix C works on the *intersection graph*

$$G(C^T C).$$

- We intend to estimate only a matrix triangular part $triu(C)$.

Lemma

The graph coloring algorithm for $triu(C)$ works on

$$G(triu(C)^T triu(C)) \cup G_K, \qquad \text{where}$$

$$G_K = \cup_{i=1}^{n} G_i, \quad G_i = (V_i, E_i) = (V, \{\{k, j\} |\ c_{ik} \neq 0 \wedge c_{ij} \neq 0 \wedge k < i \leq j$$

Updates in matrix-free environment: III.

- The estimates should be combined with an a priori sparsification
- significantly less matvecs may be needed to estimate $triu(C)$ than to estimate C.
- The partial matrix estimation depends on matrix reordering.

Summarizing,

type	initialization	solve step	memory
Recomp	$est(A^+)$, $A^+ \approx L^+U^+$	solves with L^+, U^+	L^+, U^+
Update	$est(triu(A^+))$	solves with $L, U, triu(B)$	$triu(A^+), triu(A),$

Updates in matrix-free environment and experiments

Example: Structural mechanics problem.

- A small strain metal viscoplasticity model for a rectangular plate of length 100, width 21.2 and height 9.62 cm with a hole in the middle;
- The discretization used 1 350 quadratic elements in most of the domain;
- Newton algorithm where every time-step contains an inner loop requires the solution of nonlinear systems
- We consider here a sequence of linear systems from a randomly chosen time-step in the middle of the simulation process;
- This sequence consists of 8 linear systems of dimension 4 936 with matrices containing about 315 000 nonzeros;
- We use restarted GMRES(40) preconditioned by ILUT.

Kindly provided by Karsten Quint (Universität Kassel).

Updates in matrix-free environment and experiments: II.

Number of function evaluations for different precondition strategies.

$\text{ILUT}(10^{-5}, 50), \text{Psize} \approx 812\,000$						
Matrix	Recompute		Freeze		Update	
	GMRES	estim	GMRES	estim	GMRES	estim
$A^{(0)}$	65	89	65	89	65	89
$A^{(1)}$	31	89	128	0	52	25
$A^{(2)}$	35	89	163	0	45	25
$A^{(3)}$	35	89	237	0	45	25
$A^{(4)}$	37	89	167	0	52	25
$A^{(5)}$	38	89	169	0	51	25
$A^{(6)}$	37	89	168	0	51	25
$A^{(7)}$	50	89	168	0	51	25
Total fevals	1 040		1 354		701	

Outline

An alternative strategy for matrix-free environment

An alternative strategy circumvents estimation of A^+:

Let the matvec be replaced with a function evaluation

$$A^+ \cdot v \quad \to \quad F^+(v), \quad F^+ : \mathbb{R}^n \to \mathbb{R}^n,$$

e.g. in Newton's method

$$J(x^+) \cdot v \quad \approx \quad \frac{F(x^+ + h\|x^+\|v) - F(x^+)}{h\|x^+\|} \equiv F^+(v).$$

An alternative strategy for matrix-free environment

An alternative strategy circumvents estimation of A^+:

Let the matvec be replaced with a function evaluation

$$A^+ \cdot v \quad \rightarrow \quad F^+(v), \quad F^+ : \mathbb{R}^n \rightarrow \mathbb{R}^n,$$

e.g. in Newton's method

$$J(x^+) \cdot v \quad \approx \quad \frac{F(x^+ + h\|x^+\|v) - F(x^+)}{h\|x^+\|} \equiv F^+(v).$$

We assume function components are well separable, i.e. we assume it is possible to compute the components $F_i^+ : \mathbb{R}^n \rightarrow \mathbb{R}$,

$$F_i^+(v) = e_i^T F^+(v)$$

at the cost of about one $n-$th of the full function evaluation $F^+(v)$.

An alternative strategy for matrix-free environment

An alternative strategy circumvents estimation of A^+:

Let the matvec be replaced with a function evaluation

$$A^+ \cdot v \quad \rightarrow \quad F^+(v), \quad F^+ : \mathbb{R}^n \rightarrow \mathbb{R}^n,$$

e.g. in Newton's method

$$J(x^+) \cdot v \quad \approx \quad \frac{F(x^+ + h\|x^+\|v) - F(x^+)}{h\|x^+\|} \equiv F^+(v).$$

We assume function components are well separable, i.e. we assume it is possible to compute the components $F_i^+ : \mathbb{R}^n \rightarrow \mathbb{R}$,

$$F_i^+(v) = e_i^T F^+(v)$$

at the cost of about one $n-$th of the full function evaluation $F^+(v)$.

Then the following strategy can be beneficial:

An alternative strategy for matrix-free environment: II.

- The forward solves with L in $M^+ = L(DU - D_B - U_B)$ are trivial.

- For the backward solves, use a mixed explicit-implicit strategy: Split

$$DU - D_B - U_B = DU - triu(A) + triu(A^+)$$

in the explicitly given part

$$X \equiv DU - triu(A)$$

and the implicit part $triu(A^+)$.

An alternative strategy for matrix-free environment: II.

- The forward solves with L in $M^+ = L(DU - D_B - U_B)$ are trivial.

- For the backward solves, use a mixed explicit-implicit strategy: Split

$$DU - D_B - U_B = DU - triu(A) + triu(A^+)$$

in the explicitly given part

$$X \equiv DU - triu(A)$$

and the implicit part $triu(A^+)$.

We then have to solve the upper triangular systems

$$\Big(X + triu(A^+)\Big)\, z = y,$$

yielding the standard backward substitution cycle:

An alternative strategy for matrix-free environment: III.

$$z_i = \frac{y_i - \sum_{j>i} x_{ij} z_j - \sum_{j>i} a^+_{ij} z_j}{x_{ii} + a^+_{ii}}, \qquad i = n, n-1, \dots, 1.$$

The sum $\sum_{j>i} a^+_{ij} z_j$ can be computed by the function evaluation

$$\sum_{j>i} a^+_{ij} z_j = e_i^T A^+ (0, \dots, 0, z_{i+1}, \dots, z_n)^T \approx F_i^+ \left((0, \dots, 0, z_{i+1}, \dots, z_n)^T \right).$$

The diagonal $\{a^+_{11}, \dots, a^+_{nn}\}$ can be found by computing

$$a^+_{ii} = F_i^+(e_i), \qquad 1 \le i \le n.$$

Summarizing, with this technique we can obtain the cost comparison:

type	initialization	solve step	memory
Recomp	$est(A^+)$, $A^+ \approx L^+ U^+$	solves with L^+, U^+	L^+, U^+
Update	$est(diag(A^+))$	solves with L, U, eval($\mathcal{F}$), eval($\mathcal{F}^+$)	L, U

Experimental results again

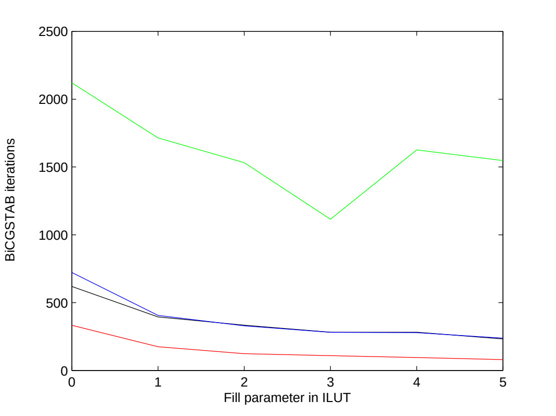
As an example consider a two-dimensional nonlinear convection-diffusion model problem: It has the form

$$-\Delta u + Ru\left(\frac{\partial u}{\partial x} + \frac{\partial u}{\partial y}\right) = 2000x(1-x)y(1-y), \tag{1}$$

on the unit square, discretized by 5-point finite differences on a uniform grid.

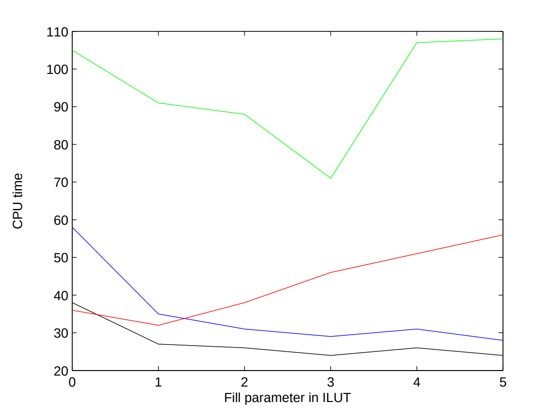
- The initial approximation is the discretization of $u_0(x, y) = 0$.
- We use here $R = 500$ and a 250×250 grid.
- We use a Newton-type method and solve the resulting 10 to 12 linear systems with BiCGSTAB with right preconditioning.
- We use a flexible stopping criterion.
- Fortran implementation (embedded in the UFO - software for testing nonlinear solvers).

Experimental results again: II.



Green: Freeze; Red: Recompute; Black: Update with partial estimation;
Blue: Update with implicit backward/forward solves.

3. Updates in matrix-free environment



Green: Freeze; Red: Recompute; Black: Update with partial estimation; Blue: Update with implicit backward/forward solves.

For more details see:

- Duintjer Tebbens J, Tůma M: *Preconditioner Updates for Solving Sequences of Linear Systems in Matrix-Free Environment*, submitted to NLAA in 2008.
- Birken Ph, Duintjer Tebbens J, Meister A, Tůma M: *Preconditioner Updates Applied to CFD Model Problems*, Applied Numerical Mathematics vol. 58, no. 11, pp.1628–1641, 2008.
- Duintjer Tebbens J, Tůma M: *Improving Triangular Preconditioner Updates for Nonsymmetric Linear Systems*, LNCS vol. 4818, pp. 737–744, 2007.
- Duintjer Tebbens J, Tůma M: *Efficient Preconditioning of Sequences of Nonsymmetric Linear Systems*, SIAM J. Sci. Comput., vol. 29, no. 5, pp. 1918–1941, 2007.

Thank you for your attention!

Supported by project number KJB100300703 of the grant agency of the Academy of Sciences of the Czech Republic.