New preconditioner updates in Newton-Krylov methods for nonlinear systems

Margherita Porcelli

ISTITUTO DI SCIENZA E TECNOLOGIE DELL'INFORMAZIONE "A. FAEDO"

Consiglio Nazionale delle Ricerche

Joint work with
Stefania Bellavia and Benedetta Morini
Dipartimento di Ingegneria Industriale, Università di Firenze

11th EUROPT Workshop on Advances in Continuous Optimization
Firenze - June 26-28, 2013

Newton-Krylov methods for nonlinear systems

$$F(x) = 0$$

$F : \mathbb{R}^n \to \mathbb{R}^n$ is continuously differentiable and J is the Jacobian matrix of F, n is large.

Given $x_0 \in \mathbb{R}^n$, a sequence $\{x_k\}$ is built:

- a Krylov method, e.g. GMRES, BiCGSTAB, is used to compute a step s_k that approximately solves

$$J(x_k)s = -F(x_k)$$

satisfying

$$\|J(x_k)s_k + F(x_k)\| \leq \eta_k \|F(x_k)\|, \quad \eta_k \in [0,1)$$

Then $x_{k+1} = x_k + s_k$,

Sequence of linear systems

$$J(x_k)s = -F(x_k), \quad k = 0, 1, \ldots$$

- By continuity, if $x_k \approx x_{k+1}$ then matrix $J(x_k)$ varies slowly from one step k to the next;
- Generally, $J(x_k)$ is not symmetric.
- Matrix-free setting: the action of the Jacobian times a vector can be provided by an operator or it can be approximated by finite differences:

$$J(x)v \approx \frac{F(x+\epsilon v) - F(x)}{\epsilon}$$

where ϵ is an appropriately chosen positive scalar.

Preconditioning & Matrix-free setting

- Recomputing an algebraic preconditioner at each nonlinear iterations requires forming the Jacobian matrices or approximating them and the resulting Newton-Krylov procedure is no longer matrix-free.
- Freezing the preconditioner slows the Krylov solver when the system matrix changes too much.
- A preconditioning strategy is considered to be matrix-free, or nearly matrix-free if
  - only a few full Jacobians are formed;
  - it needs matrices that are reduced in complexity with respect to the full Jacobians;
  - it takes advantage of matrix-vector product approximations by finite differences.

[Knoll, Keyes, 2004]

Preconditioning & Matrix-free setting

- Recomputing an algebraic preconditioner at each nonlinear iterations requires forming the Jacobian matrices or approximating them and the resulting Newton-Krylov procedure is no longer matrix-free.
- Freezing the preconditioner slows the Krylov solver when the system matrix changes too much.
- A preconditioning strategy is considered to be matrix-free, or nearly matrix-free if
  - only a few full Jacobians are formed;
  - it needs matrices that are reduced in complexity with respect to the full Jacobians;
  - it takes advantage of matrix-vector product approximations by finite differences.

[Knoll, Keyes, 2004]

Preconditioning & Matrix-free setting

- Recomputing an algebraic preconditioner at each nonlinear iterations requires forming the Jacobian matrices or approximating them and the resulting Newton-Krylov procedure is no longer matrix-free.
- Freezing the preconditioner slows the Krylov solver when the system matrix changes too much.
- A preconditioning strategy is considered to be matrix-free, or nearly matrix-free if
  - only a few full Jacobians are formed;
  - it needs matrices that are reduced in complexity with respect to the full Jacobians;
  - it takes advantage of matrix-vector product approximations by finite differences.

[Knoll, Keyes, 2004]

Techniques for preconditioning sequences of matrices

- Freezing the preconditioner computed for the first matrix of the sequence.
- Recomputing the preconditioner from scratch for each matrix (costly and pointless accurate).

$\Downarrow$

Updating an existing preconditioner computed for one matrix of the sequence (seed preconditioner) by inexpensive strategies.

Techniques for preconditioning sequences of matrices

- Freezing the preconditioner computed for the first matrix of the sequence.
- Recomputing the preconditioner from scratch for each matrix (costly and pointless accurate).

$$\Downarrow$$

Updating an existing preconditioner computed for one matrix of the sequence (seed preconditioner) by inexpensive strategies.

Existing updating approaches

Quasi-Newton Preconditioner:

- Broyden-type rank one updates for sequences arising in Newton methods [Bergamaschi, Bru, Martinez, Putti 2006]
- Limited memory Quasi-Newton updates (spd matrices) [Morales, Nocedal 2000].

Approximate preconditioner updated:

- Banded preconditioned updated for sequences of spd matrices [Benzi, Bertaccini 2003, Bertaccini 2004]
- AINV based preconditioners for sequences of unsymmetric linear systems [Bellavia, Bertaccini, Morini 2011]
- LDL^T based preconditioner updated for sequences of diagonally modified linear systems [Bellavia, De Simone, di Serafino, Morini 2010-2012]
- LDU based preconditioner for sequences of unsymmetric linear systems [Duintjer Tebbens, Tuma 2007, 2010]

Existing updating approaches

Quasi-Newton Preconditioner:

- Broyden-type rank one updates for sequences arising in Newton methods [Bergamaschi, Bru, Martinez, Putti 2006]
- Limited memory Quasi-Newton updates (spd matrices) [Morales, Nocedal 2000].

Approximate preconditioner updated:

- Banded preconditioned updated for sequences of spd matrices [Benzi, Bertaccini 2003, Bertaccini 2004]
- AINV based preconditioners for sequences of unsymmetric linear systems [Bellavia, Bertaccini, Morini 2011]
- LDL^T based preconditioner updated for sequences of diagonally modified linear systems [Bellavia, De Simone, di Serafino, Morini 2010-2012]
- LDU based preconditioner for sequences of unsymmetric linear systems [Duintjer Tebbens, Tuma 2007, 2010]

The Duintjer Tebbens & Tuma approach

- Let J_s be a specific Jacobian of the sequence and $P_s = LDU$ be an ILU factorization of J_s and let $J_k = J_s + (J_k - J_s)$.

The ideal updated preconditioner

$$P_k^I = L(D + L^{-1}(J_k - J_s)U^{-1})U$$

- Simple approximations to $L^{-1}(J_k - J_s)U^{-1}$: extracts triangular parts of the current J_k:

$$\begin{aligned} P_k &= L(\,DU + triu(J_k - J_s)\,) \\ P_k &= (\,LD + tril(J_k - J_s)\,)U \end{aligned}$$

- Key Assumption: L or $U \approx I$ (L^{-1} or U^{-1} neglected in P_k^I)

The Diagonally Updated ILU (DU_ILU)

We take the main diagonal of the matrix $J_k - J_s$:

$$P_k^I \simeq LDU + \underbrace{diag(J_k - J_s)}_{\Delta_k}$$

- The factorization of $LDU + \Delta_k$ is impractical: approximate factorization $LDU + \Delta_k \approx P_k = L_k D_k U_k$

$$D_k = D + \Delta_k, \quad L_k = eye(n), \quad off(L_k) = off(L)S_k$$
$$U_k = eye(n), \quad off(U_k) = S_k off(U)$$

$$S_k = diag(s_{11}^k, \dots, s_{nn}^k), \quad s_{ii}^k = \frac{|d_{ii}|}{|d_{ii}| + |\delta_{ii}^k|}, \quad i = 1, \dots, n,$$

Generalization to the unsymmetric case of the update preconditioning presented in [Bellavia, De Simone, di Serafino, Morini, 2010-2012]

Properties of DU_ILU

- The sparsity pattern of the factors of P_s is preserved
- The cost to form P_k is low;
- By the form of the diagonal scaling matrix S_k, P_k mimics the behaviour of $J_s + \Delta_k$

$$s_{ii}^k = \frac{|d_{ii}|}{|d_{ii}| + |\delta_{ii}^k|}, \quad i = 1, \dots, n,$$

- The conditioning of the matrices L_k and U_k is at least as good as the conditioning of L and U respectively.

The Banded Updated ILU (BU_ILU)

We take banded parts of the matrices $J_k - J_s, L^{-1}, U^{-1}$:

$$P_k^I \approx P_k = L \left[D + [L^{-1}]_\gamma [J_k - J_s]_\delta [U^{-1}]_\gamma \right]_\beta U$$

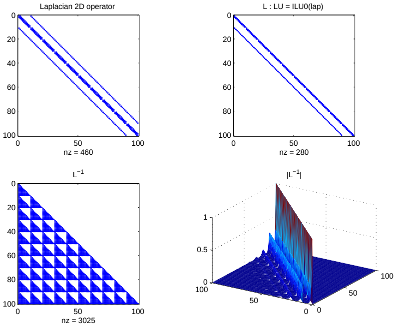
with $\delta, \gamma, \beta > 0$, small.

- We propose an algorithm for computing $[L^{-1}]_\gamma$ and $[U^{-1}]_\gamma$ exactly without the need of a complete inversion of L and U (suitable for parallel implementation).
- The application of P_k depends on β:
  - If $\beta = 0 \Rightarrow$ 1 diagonal and 2 triangular systems must be solved.
  - Else, the factorization of the banded middle factor in P_k is necessary.

$[A]_\gamma$ denotes the banded matrix obtained extracting from A the main diagonal and γ upper and lower diagonals.

Motivation: matrices with decaying inverses

- banded spd and indefinite matrices [Demko et al. 1984][Meurant 1992];
- nonsymmetric block tridiagonal banded matrices [Nabben 1999];
- matrices $f(A)$ with A symm and banded and f analytic [Benzi, Golub 99].



Typically, the rate of decay of the entries of A^{-1} is fast if A is diag. dominant.

Approximation analysis of DU_ILU and BU_ILU

The quality of DU_ILU and BU_ILU measured as

$$\|J_k - P_k\|$$

depends on two terms:

- $\|J_s - P_s\|$, i.e. the quality of the seed preconditioner;
- $\|J_k - J_s\|$, i.e. distance of the current Jacobian from the seed Jacobian

$$\Downarrow$$

$P_s \approx J_s$ and $\{J_k\}$ is slowly varying $\Rightarrow P_k \approx J_k$

[Bellavia, Morini, Porcelli 2013]

Approximation analysis of DU_ILU and BU_ILU

The quality of DU_ILU and BU_ILU measured as

$$\|J_k - P_k\|$$

depends on two terms:

- $\|J_s - P_s\|$, i.e. the quality of the seed preconditioner;
- $\|J_k - J_s\|$, i.e. distance of the current Jacobian from the seed Jacobian

$$\Downarrow$$

$P_s \approx J_s$ and $\{J_k\}$ is slowly varying $\Rightarrow P_k \approx J_k$

[Bellavia, Morini, Porcelli 2013]

Preconditioned Newton-Krylov method with linesearch: numerical comparison under Matlab

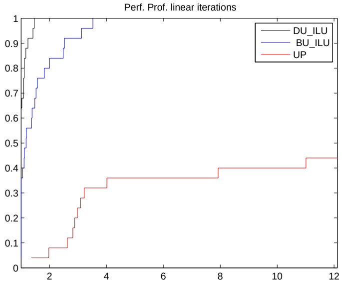
- 3 preconditioning strategies (DU_ILU , BU_ILU , TT) and the unpreconditioned (UP) case.
- Linear solver: BiCGSTAB, $\mathtt{LI}_{\mathtt{max}} = 400$
- Refresh: when the backtracking strategy fails in producing an acceptable step a new reference matrix and preconditioner are initialized.
- Finite difference approximation for the computation of :
  - $J_s (= J_0)$
  - J_k times a vector
  - $[J_k]_\delta$ (and then stored)

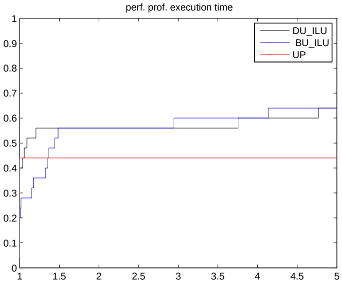
Numerical comparison under Matlab c.ed

- Test Problems
  - Nonlinear Convection-Diffusion (NCD), $Re = 750, 1000, 1250$
  - Flow in a Porous Medium (FPM),
  - CounterCurrent Reactor (CCR)
  - 2D Driven Cavity (2DC), $Re = 50, 100, 150$
- The dimension n of the nonlinear systems can be varied: $n = 4900, 8100, 10000, 15625, 22500$.
- The most time-consuming part of each updating strategy has been implemented as Fortran 90 mex-file with MATLAB interface.

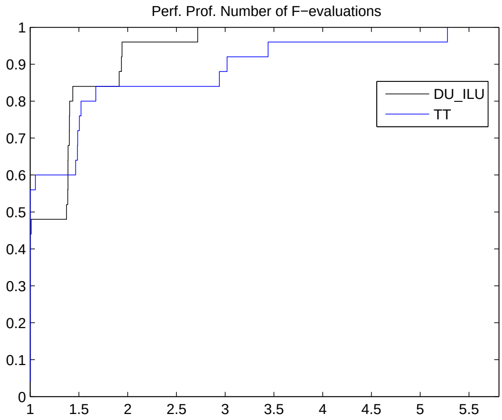
Comparison among DU_ILU , BU_ILU , UP

- Problems NCD, FPM, CCR: 25 runs
- BU_ILU strategy: $\delta = \beta = 0$, $\gamma = 20$





Comparison between DU_ILU and TT



Comparison between DU_ILU and BU_ILU

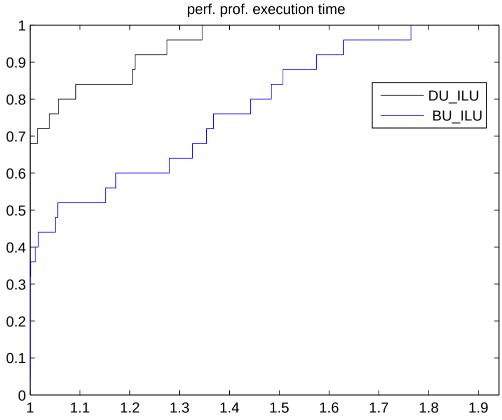


Figure: Performance profile in terms of execution time

2D Driven Cavity

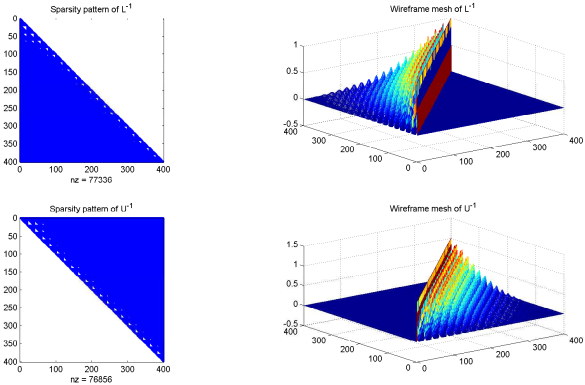


Figure: Sparsity pattern (on the left) and wireframe mesh (on the right) of the inverses of the L and U factors obtained from the ILU factorization of the Jacobian at the starting point ($n = 400$).

2D Driven Cavity c.ed

- The Newton-Krylov method with UP fails in all runs.
- The DU_ILU strategy is not applicable.
- The Newton-Krylov method with TT fails in 11 runs out of 15
  - Failures of preconditioned BiCGSTAB gave rise to poor descent directions and unsuccessful backtracking strategies
  - Preconditioner's refresh was not beneficial.
- The Newton-Krylov method with BU_ILU is successful in all runs and gives the best performance with $\gamma = 50$, $\delta = \beta = 1$: the tridiagonal part of J is used and a tridiagonal system has to be solved to apply the preconditioner.

2D Driven Cavity c.ed

- The Newton-Krylov method with UP fails in all runs.
- The DU_ILU strategy is not applicable.
- The Newton-Krylov method with TT fails in 11 runs out of 15
  - Failures of preconditioned BiCGSTAB gave rise to poor descent directions and unsuccessful backtracking strategies
  - Preconditioner's refresh was not beneficial.
- The Newton-Krylov method with BU_ILU is successful in all runs and gives the best performance with $\gamma = 50$, $\delta = \beta = 1$: the tridiagonal part of J is used and a tridiagonal system has to be solved to apply the preconditioner.

Final remarks

- The DU_ILU and BU_ILU preconditioners use matrices that are reduced in complexity with respect to the full J_k's.
- The DU_ILU and BU_ILU preconditioners can be used for the solution of general sequences of linear systems

$$A_k x = b_k, \quad k = 1, 2, \ldots$$

where $A_k \in \mathbb{R}^{n \times n}$ are large and sparse and $b_k \in \mathbb{R}^n$.

  - Their applicability comprises the symmetric case since the updates preserve the symmetry of the preconditioners.
  - They are nearly matrix-free whenever the function that performs the matrix-vector product is separable.

Final remarks

- The DU_ILU and BU_ILU preconditioners use matrices that are reduced in complexity with respect to the full J_k's.
- The DU_ILU and BU_ILU preconditioners can be used for the solution of general sequences of linear systems

$$A_k x = b_k, \quad k = 1, 2, \ldots$$

where $A_k \in \mathbb{R}^{n\times n}$ are large and sparse and $b_k \in \mathbb{R}^n$.

  - Their applicability comprises the symmetric case since the updates preserve the symmetry of the preconditioners.
  - They are nearly matrix-free whenever the function that performs the matrix-vector product is separable.

Thanks for your attention

S. Bellavia, B. Morini, M. Porcelli, *New updates of incomplete LU factorizations and applications to large nonlinear systems*, Optimization Methods and Software (2013)