Lobachevsky State University of Nizhni Novgorod

Faculty of Computational mathematics and cybernetics

Iterative Methods for Solving Linear Systems

Preconditioning Methods

Supported by Intel

K. A. Barkalov
Software Department

Contents

- Problem Statement
  - Concept of Preconditioning
  - Requirements to Preconditioners
  - Preconditioning Types
- Basic Preconditioners
  - Jacobi (J), Gauss-Seidel (GS)
  - SOR, SSOR, SGS
- Incomplete LU-factorization
  - General Pattern
  - ILU(0), factorization without filling
  - ILU(0), factorization with filling control
- Experimental Results

Problem statement

- Let us consider a system of n linear equations that looks as follows:

$$\begin{aligned} a_{11}x_1 &+ a_{12}x_2 &+ \ldots + a_{1n}x_n &= b_1 \\ a_{21}x_1 &+ a_{22}x_2 &+ \ldots + a_{2n}x_n &= b_2 \\ \ldots & & & \\ a_{n1}x_1 &+ a_{n2}x_2 &+ \ldots + a_{nn}x_n &= b_n \end{aligned}$$

- This system can be represented as a matrix

$$Ax=b$$

- $A=(a_{ij})$ is a $n\times n$ real matrix; A is a sparse matrix; b and x are vectors consisting of n elements; the exact system solution is x^*.
- *An iterative method* generates a sequence of vectors $x^{(s)}\in R^m$, $s=0,1,2,\ldots$, where $x^{(s)}$ is an approximate system solution.

Convergence of Iterative Methods

- Iterative method is convergent if

$$\forall\ x^{(0)} \in R^m \quad \lim_{s\to\infty} \left\| x^{(s)} - x^* \right\| = 0$$

- For iterative methods, the following is true

$$\left\| z^{(s+1)} \right\| \le \left(\varphi(\mu_A)\right)^s \left\| z^{(0)} \right\|$$

where $z^{(s)}=x^{(s)}-x^*$ is the next approximation error,

φ is a function, $\varphi\to 0$ when $\sigma\to\infty$.

$\mu_A=\lambda_{max}/\lambda_{min}$ is the condition number.

Example for the conjugate gradient method

$$\varphi(\mu_A) = \frac{\sqrt{\mu_A} - 1}{\sqrt{\mu_A} + 1}$$

Idea of Preconditioning

- $\mu_A \approx 1$ – the convergence rate is high (A is well-conditioned)
- $\mu_A >> 1$ – the convergence rate is low(A is ill-conditioned)
- The idea of preconditioning lies in converting an ill-conditioned system

$$Ax=b$$

to a well-conditioned one

$$M^{-1}Ax=M^{-1}b.$$

Here, M is a preconditioner.
- $M^{-1}A$ is not computed explicitly as $M^{-1}A$ is very likely to be a dense matrix
- Corrective steps allowing for preconditioning are added to the iterative method.

Requirements to the Preconditioner

1. M must be close to A ($M^{-1}A$ is well-conditioned)
2. M must be easy to compute;
3. M must allow for fast solution of systems such as

$$Mz = r$$

in relation to an unknown vector z.

Example 1.

Let $M=A$. Then requirements 1 and 2 are satisfied.

Requirement 3 is not satisfied. $Az=r$ is the same problem as the initial one.

Example 2.

Let $M=\mathrm{diag}(A)$. Then requirements 2 and 3 are satisfied.

Requirement 1 may not be satisfied

Preconditioning Types

$Ax=b$ *is* the initial system

1. Left preconditioning
   $M^{-1}Ax=M^{-1}b.$
2. Right preconditioning
   $AM^{-1}u=b$, where $x=M^{-1}u.$
3. Split preconditioning
   Let us present the preconditioner as $M = M_L M_R$
4. Then
   $M_L^{-1} A M_R^{-1} u = M_L^{-1} b$, where $x = M_R^{-1} u$

Basic Preconditioners

- Let us remember the basic iterative methods that include Jacobi, Seidel and over relaxation (SOR and SSOR) methods.
- All the methods above are particular cases of the simple iteration method.

$$x^{(s+1)} = Gx^{(s)} + c$$

  where $A=M-N$ and $G = M^{-1}N = M^{-1}(M - A) = E - M^{-1}A$

- Simple iteration method (*) for the system

$$(E - G)x = c$$

  that may be formulated as follows

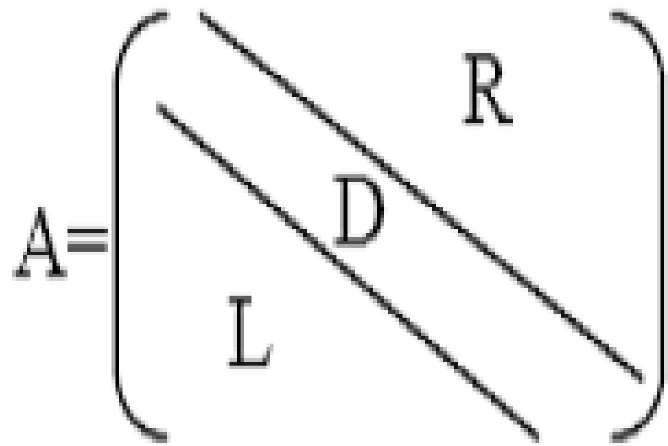
$$M^{-1}Ax = M^{-1}b$$

- As a result, the Jacobi, Seidel, SOR and SSOR methods are equivalent to the simple iteration method with a preconditioner.

Basic Preconditioners

Therefore, the following preconditioners are obtained:

1. $M_J = D$ (Jacobi)
   may be explicitly applied to the system - multiply by D^{-1}.
2. $M_{GS} = D + L$ (Gauss-Seidel)
3. $M_{SOR} = \frac{1}{\omega}(D + \omega L)$
4. $M_{SSOR} = \frac{1}{\omega(2-\omega)}(D + \omega L)D^{-1}(D + \omega R)$



SSOR-preconditioning

- How to select ω?
- Parameter selection for the preconditioner is not as critical as for the SSOR method: ω=1
- Symmetric Gauss-Seidel preconditioner

$$M_{SGS} = (D + L)D^{-1}(D + R)$$

- Use of preconditioner, i.e. system solution

$$M_{SGS} z = r$$

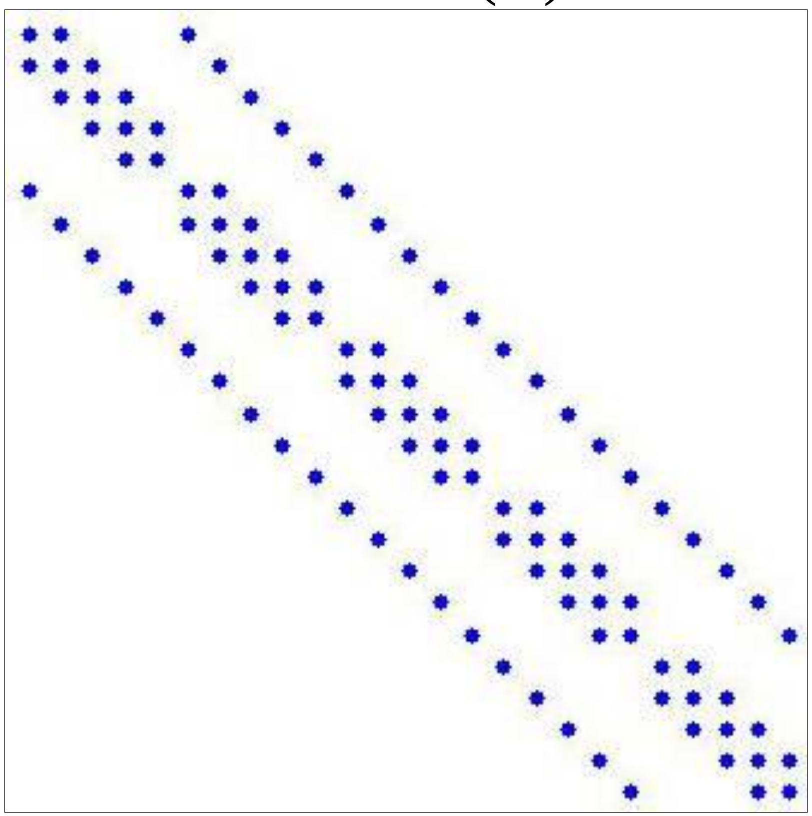
  is as complex as multiplying a matrix by vector.
- Upon the whole, M_{SGS} is better than M_J, but is still insufficient.

SGS-Preconditioning – Example

Numerical solution of Poisson Equation for a 5×5 grid

Matrix A: size n=25 (n^2 = 625), number of non-zeroes nz = 105,

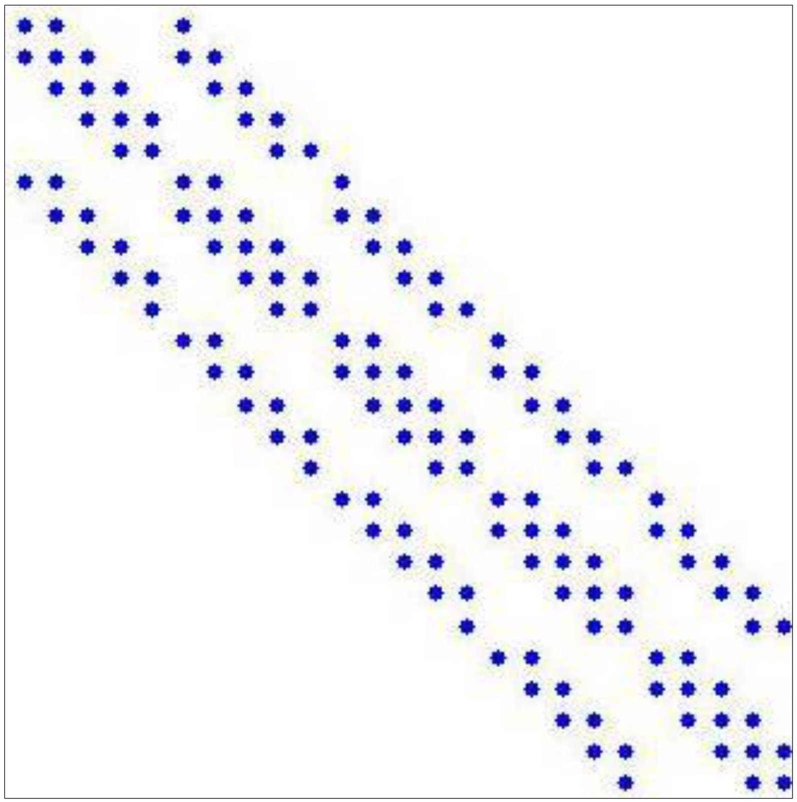
Condition number cond(A)= 20.7



SGS-Preconditioning – Example

Symmetric Gauss-Seidel preconditioner

$$M_{SGS} = (D + L) D^{-1} (D + R)$$



Initial system: cond(A)=20.7;
M_{SGS}: cond($M^{-1}A$)=5.1.

Poisson equation for a 40 × 40 grid
Matrix size 1600 × 1600.

Initial system:cond(A)=989;
M_{SGS}: cond($M^{-1}A$)=210.

ILU(0)-Preconditioning

- Let A be a sparse matrix
  $NZ(A)=\{(i,j): a_{ij}\neq 0\}$
- Let A factorization has been found in the form of

$$A=LU-R$$

  L and U are lower (with a single diagonal) and upper triangular matri ces;

  $NZ(L) \cup NZ(U) = NZ(A)$;

  $r_{ij} = 0$ for all $(i,j) \in NZ(A)$.

  Then ILU(0) is a preconditioner for $M=LU\approx A$.
- The requirements above do not provide for a unique determination of ILU(0).

ILU(0)-Preconditioning

- Constructive definition of ILU(0) : Perform LU–factorization of A zeroing all filling elements in L and U outside $NZ(A)$ at the same time.
- LU–factorization (method of Gaussian elimination)

for i=2,…, n do

for k=1,…, i–1 do

$a_{ik} = a_{ik}/a_{kk}$

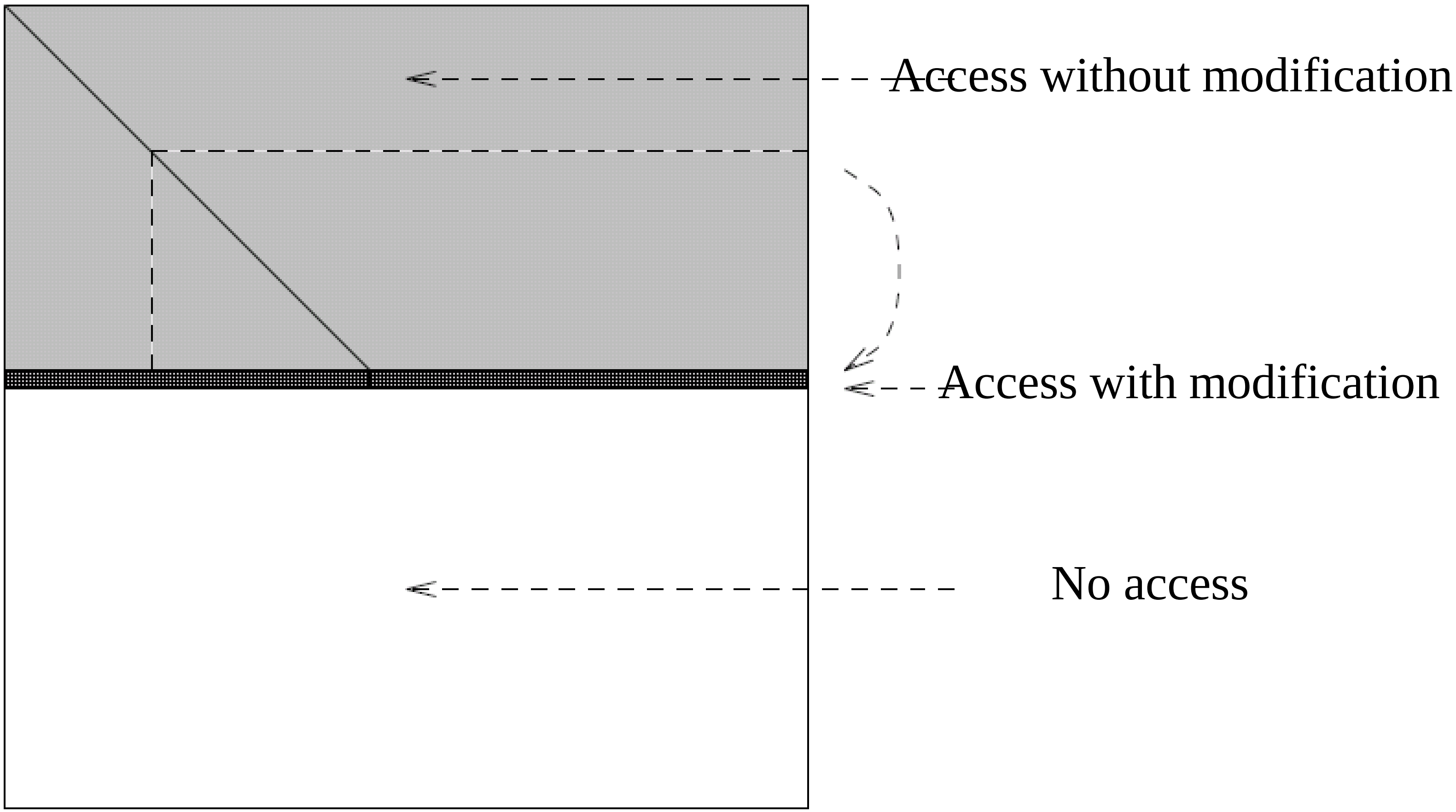
for j=k+1,…, n do

$a_{ij} = a_{ij} - a_{ik}*a_{kj}$

end i

end k

Memory State



Access to the matrix lines: effective for sparse matrices in CRS format

ILU(0)-Factorization - Algorithm

for i=2,…, n do

for k=1,…, i–1 and if $(i,k)\in NZ(A)$ do

$a_{ik} = a_{ik}/a_{kk}$

for j=k+1,…, n and if $(i,j)\in NZ(A)$ do

$a_{ij} = a_{ij} - a_{ik}*a_{kj}$

end i

end k

- If the matrix A is symmetric positive definite, then $ILU(0)$ transforms to $IC(0)$: this is incomplete Cholesky factorization.

ILU(0)-Factorization – Example 1

Let us consider factorization of the matrix A

$$A = \begin{bmatrix} 4 & -1 & -1 & 0 \\ -1 & 4 & 0 & -1 \\ -1 & 0 & 4 & -1 \\ 0 & -1 & -1 & 4 \end{bmatrix}$$

and perform a complete LU-factorization

$$A = LU = \begin{bmatrix} 1 & & & \\ -0.25 & 1 & & \\ -0.25 & -0.067 & 1 & \\ 0 & -0.0267 & -0.286 & 1 \end{bmatrix} \begin{bmatrix} 4 & -1 & -1 & 0 \\ & 3.75 & -0.25 & -1 \\ & & 3.733 & -1.067 \\ & & & 3.429 \end{bmatrix}$$

ILU(0)-Factorization – Example 1

Incomplete factorization $A \approx IL * IU$

$$IL = \begin{bmatrix} 1 & & & \\ -0.25 & 1 & & \\ -0.25 & 0 & 1 & \\ 0 & -0.267 & -0.267 & 1 \end{bmatrix} \qquad IU = \begin{bmatrix} 4 & -1 & -1 & 0 \\ & 3.75 & 0 & -1 \\ & & 3.75 & -1 \\ & & & 3.467 \end{bmatrix}$$

Incomplete factorization residue

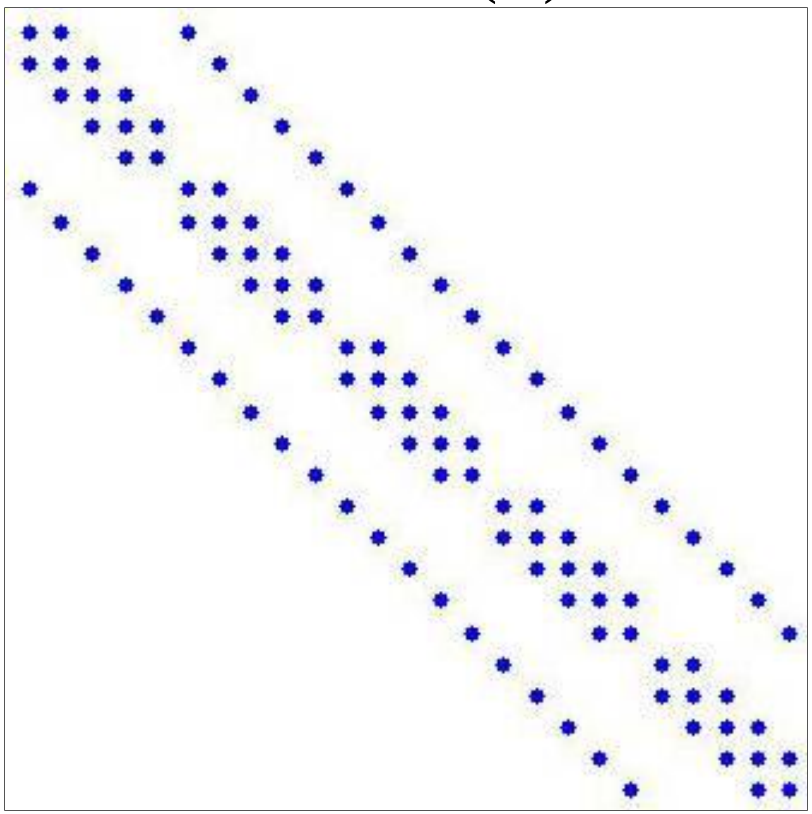
$$A - IL * IU = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -0.25 & 0 \\ 0 & -0.25 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

ILU(0)-factorization – Example 2

Numerical solution of Poisson equation for a 5×5 grid

Matrix A: size n=25 (n^2 = 625), number of nonzeroes nz = 105,

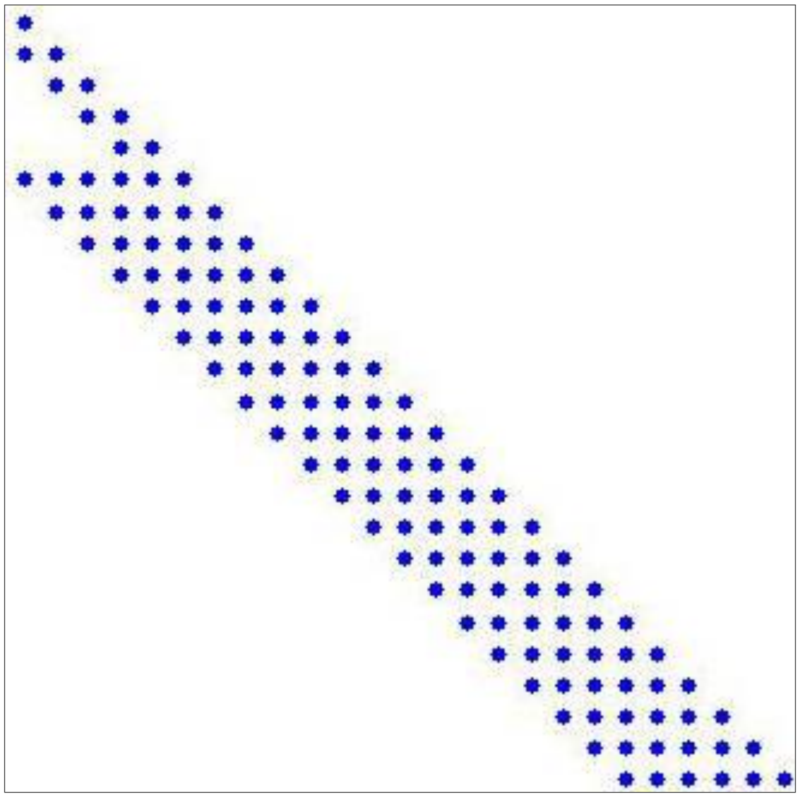
Condition number cond(A)= 20.7

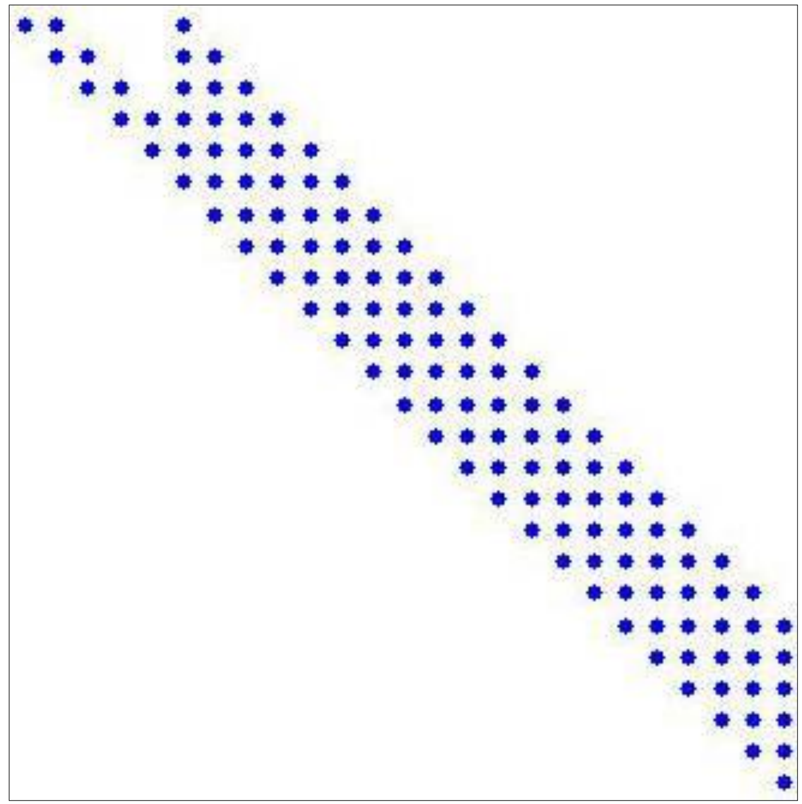


ILU(0)-Factorization – Example 2

Perform a complete LU-factorization

L U

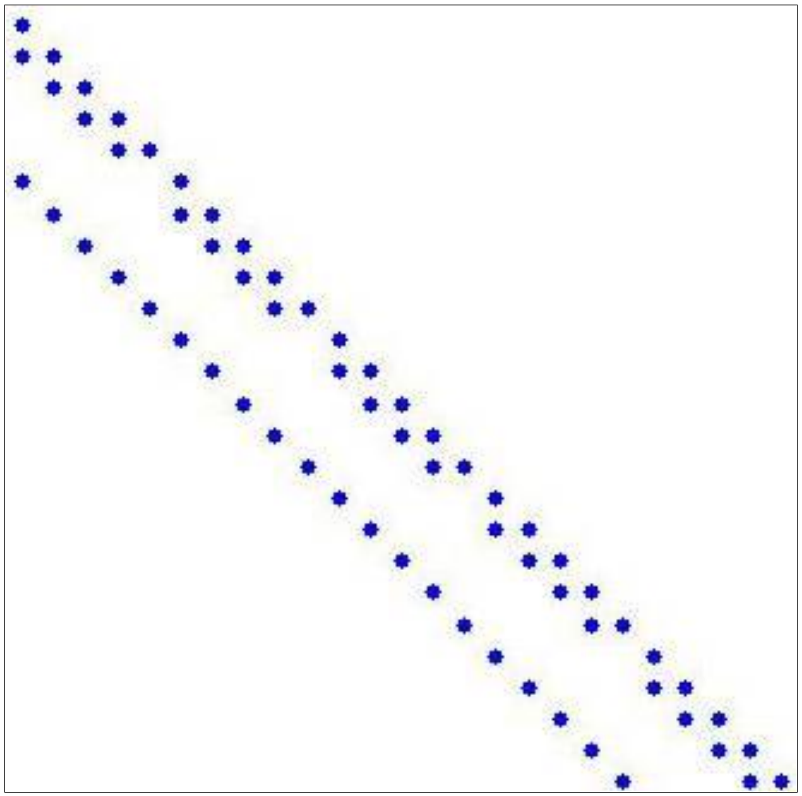


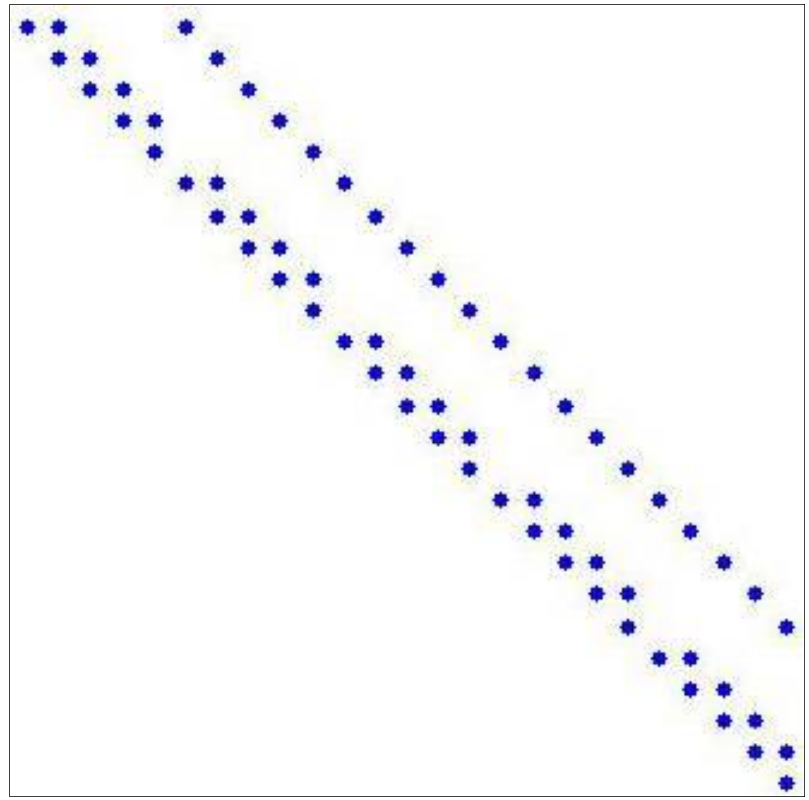


ILU(0)-Factorization – Example 2

Perform *ILU(0)*-factorization of $A \approx IL*IU$

IL *IU*

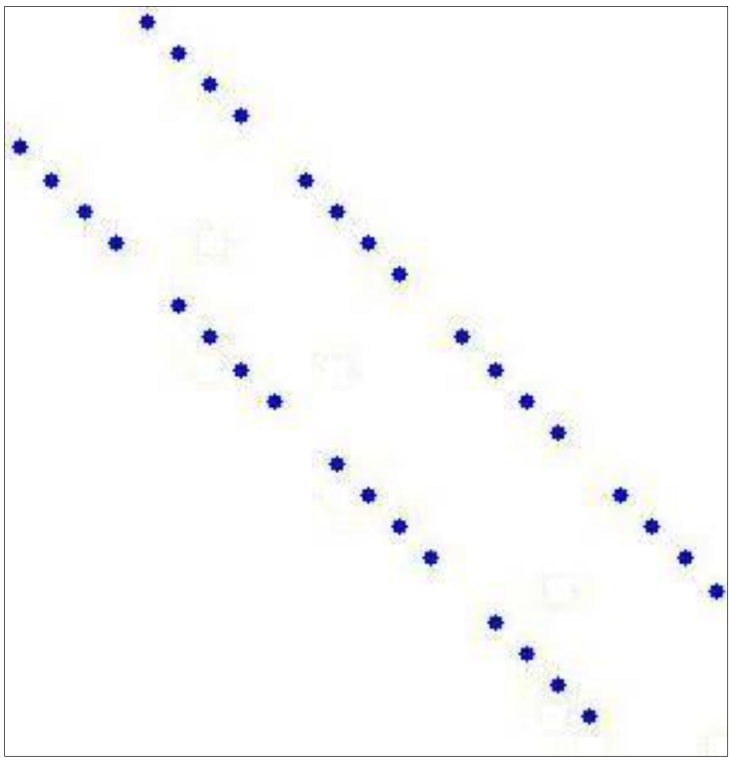




ILU(0)-Factorization – Example 2

ILU(0)-factorization residual

$$A - IL * IU$$



Initial system: cond(A)=20.7;

M_{SGS}: cond($M^{-1}A$)=5.1.

$M_{ILU(0)}$: cond($M^{-1}A$)=3.6.

Poisson equation for a 40 × 40 grid
Matrix size 1600 × 1600.

Initial system:cond(A)=989;

M_{SGS}: cond($M^{-1}A$)=210.

$M_{ILU(0)}$: cond($M^{-1}A$)=143.

Filling Control. ILU(p)-Factorization

- A more exact ILU-factorization can be obtained by allowing a certain degree of factor filling
  - for matrices with a regular structure *p* additional diagonals may be filled;
  - generalization for matrices with an irregular structure via the *filling level* concept.
- Initial l_{ij} filling level value

$$l_{ij} = \begin{cases} 0, \text{если } a_{ij} \neq 0 \text{ или } i = j, \\ \infty, \text{иначе.} \end{cases}$$

- At the *i*-th step of Gaussian elimination

$$l_{ij} = \min\{ l_{ij}, l_{ik} + l_{kj} + 1\}$$

ILU(p)-Factorization - Algorithm

- ILU(p) strategy is to zero all elements whose filling level exceeds *p*.

for $i = 2,\ldots,n$ do

  for $k = 1, \ldots, i-1$ and if $a_{ij} \neq 0$ do

    $a_{ik} = a_{ik} / a_{jj}$

    $a_{i*} = a_{i*} - a_{ik} * a_{i*}$

    update filling levels for a_{i*} :

    for *i*-th line: if $l_{ij} > p$ then $a_{ij} = 0$

$$l_{ij} = \min\{ l_{ij}, l_{ik} + l_{kj} + 1\}$$

  end *k*

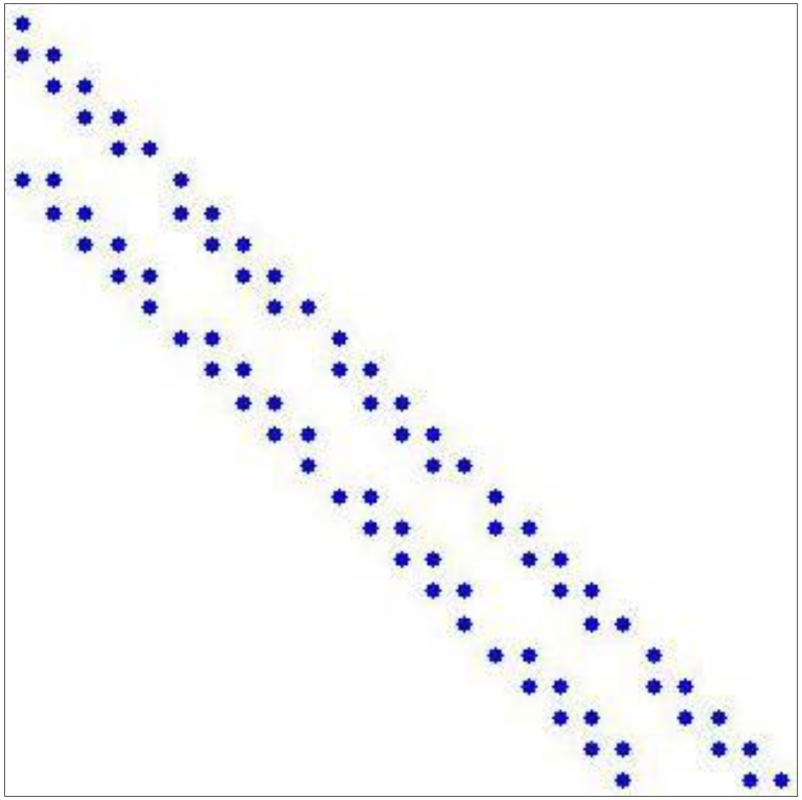
end *i*

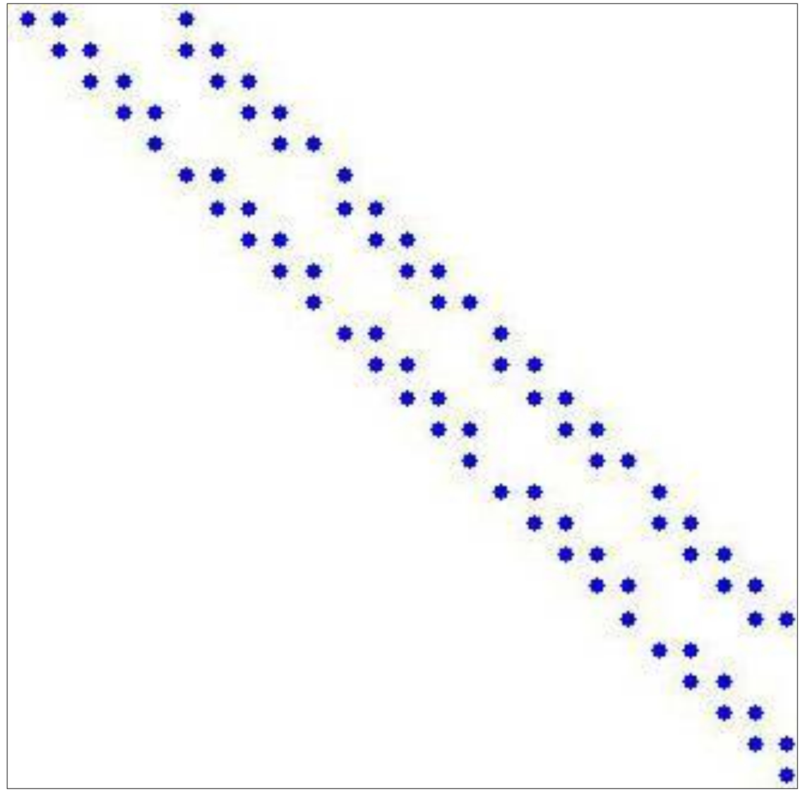
- The algorithm may be divided into two parts: symbolic (*L* and *U* patterns) and numeric (*L* and *U* values)

ILU(p)-Factorization - Example

Numerical solution of Poisson equation for a 5×5 grid

Perform *ILU(1)*-factorization of $A \approx IL * IU$

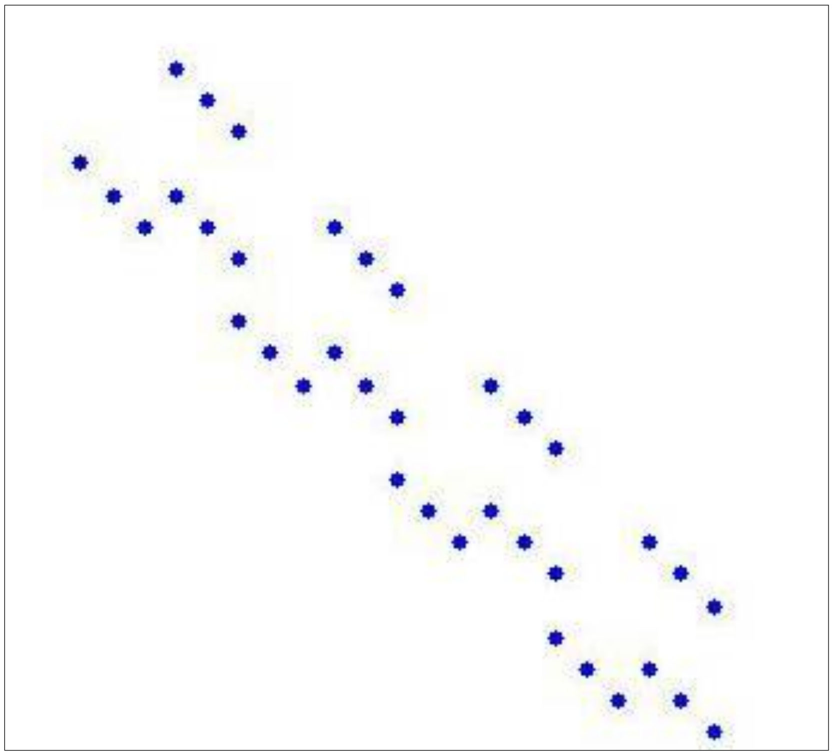
IL 

IU 

ILU(p)-Factorization - Example

ILU(1)-factorization residual

$A – IL * IU$



Initial system: cond(A)=20.7;
M_{SGS}: cond($M^{-1}A$)=5.1.
$M_{ILU(0)}$: cond($M^{-1}A$)=3.6.
$M_{ILU(1)}$: cond($M^{-1}A$)=1.5.

Poisson Equation for a 40 × 40 grid
Matrix size 1600 × 1600.

Initial system:cond(A)=989;
M_{SGS}: cond($M^{-1}A$)=210.
$M_{ILU(0)}$: cond($M^{-1}A$)=143.
$M_{ILU(0)}$: cond($M^{-1}A$)=54.

Conclusion

- As part of this lecture, we have reviewed the following:
  - Concept of preconditioning
  - Requirements to preconditioners
  - Preconditioning types
  - Basic preconditioners
    - Jacobi (J), Gauss-Seidel (GS)
    - SOR, SSOR, SGS
  - Partial LU-factorization
    - General Pattern
    - ILU(0), factorization without filling
    - ILU(0), factorization with filling control
  - Experimental results

References

1. James W. Demmel. Applied Numerical Linear Algebra. SIAM, 1997.
2. Gene H. Golub, Charles F. Van Loan. Matrix Computations. The John Hopkins University Press, 1996.
3. J. Dongarra et al. Templates for the solution of linear systems: building blocks for iterative methods. SIAM, 1994.
4. Y. Saad. Iterative Methods for Sparse Linear Systems. SIAM, 2003.

Internet Resources

5. Intel Math Kernel Library Reference Manual. [http://software.intel.com/sites/products/documentation/hpc/mkl/mklman.pdf].

Authors

- Konstantin Barkalov
  Candidate of Physical and Mathematical Sciences,
  Associate Professor, Software Department, CMC Faculty,
  Nizhny Novgorod State University
  barkalov@vmk.unn.ru

- Evgeni Kozinov (education software codes)