Lobachevsky State University of Nizhni Novgorod

Faculty of Computational mathematics and cybernetics

Iterative Methods for Solving Linear Systems

Basic Iterative Methods. Chebyshev's Aceleration

Supported by Intel

K.A. Barkalov,
Software Department

Contents

Problem Description

- Let us consider a system of n linear equations like

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

- As a matrix, the system may be represented as follows

$$Ax=b$$

- $A=(a_{ij})$ is a $n\times n$ real matrix; b and x are vectors consisting of n elements; x^* is the exact solution of the system
- *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.

Properties of Iterative Methods

- Iterative method is convergent if

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

- Iterative method stop criteria: accuracy and number of iterations.
  - Stop, if $\|x^{(s)} - x^{(s-1)}\| \le \varepsilon_1$, where $\varepsilon_2 = \|x^{(s)} - x^{(s-1)}\|$ is the attainable method accuracy.
  - Stop, if $\|r^{(s)}\| \le \varepsilon_1$, where $\varepsilon_2 = \|r^{(s)}\|$ is the attainable method accuracy.
  - Stop, if $s = N$, where $x^{(N)}$ is understood as an obtained solution. The maximum number of iterations N is predefined.
- From this on, let us suppose that A is a SPD matrix.

Fixed Point Iteration Method

- Solve the system $Ax=b$
- Convert it into $x=Gx+c$

  Ways of conversion are varied

  Let $A=M-N$, then $(M-N)x=b$, $Mx = Nx+b$

  $$x=M^{-1}Nx+M^{-1}b, \qquad \text{i.e. } G=M^{-1}N,\ c=M^{-1}b.$$
- Iterative process

  $$x^{(s+1)} = Gx^{(s)} + c$$
- Convergence
  - $\|G\|<1$ (necessary); $\rho(G) < 1$ (sufficient).
  - $\left\|x^{*} - x^{(s)}\right\| \le \left\|G\right\|^{s}\left\|x^{*} - x^{(0)}\right\|$

Fixed Point Iteration Method

- A special case is $$\frac{x^{(s+1)} - x^{(s)}}{\tau} + Ax^{(s)} = b$$

where $\tau \neq 0$ is the method parameter.

- Computation of the next approximation:

$$x^{(s+1)} = -\tau(Ax^{(s)} - b) + x^{(s)} = -\tau \cdot r^{(s)} + x^{(s)}$$

where $r^{(s)}$ is a residual of the *sth* approximation to the solution.

- Component-wise representation of the method

$$x_i^{(s+1)} = -\tau\left(\sum_{j=1}^{n} a_{ij} x_j^{(s)} - b_i\right) + x_i^{(s)}$$

- Complexity estimation for *L* iterations of the method

$$T_1 = L(2n^2 + 2n).$$

Fixed Point Iteration – Convergence

- If the A matrix is symmetric and positive determined, and $\tau \in (0, \lambda_{\max})$, the method converges to the exact system solution from any initial approximation.
- Best value of the parameter $$\tau_{opt} = \frac{2}{\lambda_{\min} + \lambda_{\max}}$$
- For fixed point iteration with the best parameter value, the following is true $$\left\| z^{(s+1)} \right\|_2 \le \left(\frac{\mu_A - 1}{\mu_A + 1} \right)^{s+1} \left\| z^{(0)} \right\|_2$$

where μ_A is the condition number of the matrix A,

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

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

Fixed Point Iteration – Parallel Algorithm

- Iterations are accomplished in a sequence
- Computations performed as part of a single iteration are parallelized by means of:
  - Basic computation according to the selected method that consists in multiplication of the matrix A by the vector $x^{(s)}$,
  - Additional computation (scalar multiplication and addition of vectors) that are less complex.
- Algorithms of parallel matrix multiplication by a vector may also be used

Fixed Point Iteration – Parallel Algorithm

- Estimated complexity of the parallel operation $Ax^{(s)}$ in case of horizontal band division of the matrix A is

$$2n^2/p + \delta$$

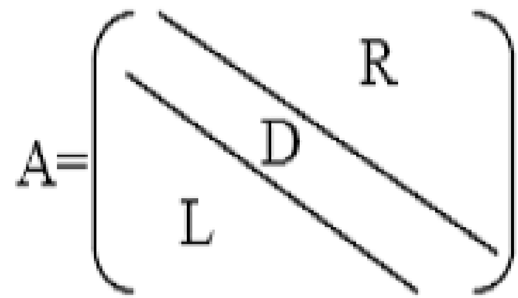
  where *n* is the vector length, *p* is the number of flows, δ – contingency
- Less complex computations are subject to single threading
- Total complexity estimation of the parallel fixed point iteration method is

$$T_p = L\left(\frac{2n^2}{p} + 2n + \delta\right)$$

  where *L* is the number of method iterations.

Jacobi and Seidel Methods

- Let us come round to solution of the system $Ax=b$ with the symmetric positive definite matrix A.
- Let us represent the matrix as $A=L+D+R$, where D, L, R – diagonal, strict lower triangular strict upper triangular parts of the matrix A.



- Jacobi method

$$Dx^{(s+1)} = (-L - R)x^{(s)} + b$$

- Seidel method

$$(D + L)x^{(s+1)} = (-R)x^{(s)} + b$$

Jacobi and Seidel Methods – Convergence

- Let us write down the methods in a component-wise manner
  - Jacobi method $$x_i^{(s+1)} = \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(s)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(s)} \right) \Big/ a_{ii}$$
  - Seidel method $$x_i^{(s+1)} = \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(s+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(s)} \right) \Big/ a_{ii}$$
- Method convergence
  - Jacobi method: A>0, strict diagonal dominance
  - Seidel method: A>0
- Transition matrices
  - Jacobi method: $G_{Jac} = -D^{-1}(L + R) = D^{-1}A - E$
  - Seidel method: $G_{GS} = -(D+L)^{-1} R = (D+L)^{-1} A - E$

Successive Over Relaxation Method (SOR)

- Successive over relaxation method (SOR) is written as

$$\frac{(D+\omega L)(x^{(s+1)}-x^{(s)})}{\omega}+Ax^{(s)}=b$$

where ω is the method parameter.

- Convergence: $\omega\in(0,2)$ (required), if $A>0$, then it is sufficient
- For numerical solution of mathematical physics problems

$$\omega_{opt}\approx 2-O(h)$$

- Required number of iterations when $\omega=\omega_{opt}$: $O(h^{-1})$
  when $\omega=1$ (SOR and Seidel method show the same): $O(h^{-2})$
- More exact estimation

$$\omega_{opt}=\frac{2}{1+\sqrt{1-\rho^2\left(D^{-1}(R+L)\right)}}$$

SOR – Algorithm

- With regard to $A–L=R+D$, let us put it into a more convenient form

$$Dx^{(s+1)} = -\omega Lx^{(s+1)} + (1-\omega)Dx^{(s)} - \omega Rx^{(s)} + \omega b$$

- New approximation components are computed as

$$a_{ii}x_i^{(s+1)} = -\omega\sum_{j=1}^{i-1} a_{ij}x_j^{(s+1)} + (1-\omega)a_{ii}x_i^{(s)} - \omega\sum_{j=i+1}^{n} a_{ij}x_j^{(s)} + \omega b_i$$

- Transition matrix $G_{SOR} = (D+\omega L)^{-1}((1-\omega)D - \omega R)$
  - Non-symmetric!
- Total complexity of a single iteration

$$t_1 = 2n^2 + n$$

- Performance of L iterations

$$T_1 = L(2n^2 + n).$$

SSOR – symmetric method

- A SSOR step consists of:

1. A SOR step that involves computation of $x^{(s+1/2)}$ in the normal order;
2. A SOR step that involves computation of $x^{(s+1)}$ in the reverse order.

- SSOR step in a matrix form

1. $(D+\omega L)x^{(s+1/2)} = (1-\omega)Dx^{(s)} - \omega R x^{(s)} + \omega b$
2. $(D+\omega U)x^{(s+1)} = (1-\omega)Dx^{(s+1/2)} - \omega L x^{(s+1/2)} + \omega b$

- Transition matrix

$$G_{SSOR} = (D+\omega U)^{-1}((1-\omega)D - \omega L)(D+\omega L)^{-1}((1-\omega)D - \omega R)$$

– usually more iterations than for SOR with ω_{opt}

– G_{SSOR} – symmetric, used for Chebyshev’s acceleration.

Chebyshev's Acceleration

- Having found approximations $x^{(0)}, x^{(1)}, \dots, x^{(m)}$
- Let us find $y^{(m)} = \sum_{i=0}^{m} \alpha_i x^{(i)}$ which is better than $x^{(m)}$
- Let us write the error $y^{(m)}$

$$y^{(m)} - x^* = \sum_{i=0}^{m} \alpha_i x^{(i)} - x^* = \sum_{i=0}^{m} \alpha_i (x^{(i)} - x^*) = \sum_{i=0}^{m} \alpha_i G^i (x^{(0)} - x^*) = p_m(G)(x^{(0)} - x^*),$$

where $p_m(G) = \sum_{i=0}^{m} \alpha_i G^i$ is a polynomial in the matrix G, $p_m(1) = \sum_{i=0}^{m} \alpha_i = 1$

- $\rho(p_m(G)) \to \min$ – the spectral radius is minimized.
- $p_m(G)$ can be obtained using Chebyshev's polynomials $T_m(x)$

Chebyshev's Acceleration

- $p_m(G) = \mu_m T_m(G/\rho)$, where $\mu_m \equiv 1 / T_m(1/\rho)$, $T_m(x)$ is a Chebyshev's polynomial, ρ is the spectral radius of the matrix *G*.
- Chebyshev's polynomials

  $$T_0(x) = 1 \quad T_1(x) = x \quad T_m(x) = 2xT_{m-1}(x) - T_{m-2}(x)$$
- Three-term relation enables only three vectors $y^{(m)}$, $y^{(m-1)}$, $y^{(m-2)}$ to be used, but not all vectors $x^{(m)}$, $0 \le i \le m$.
- The following relations may be derived

  $$y^{(m)} = \frac{2\mu_m}{\mu_{m-1}} \frac{G}{\rho} y^{(m-1)} - \frac{\mu_m}{\mu_{m-2}} y^{(m-2)} + \frac{2\mu_m}{\rho\mu_{m-1}} c \qquad \mu_m = \left(\frac{2}{\rho\mu_{m-1}} - \frac{1}{\mu_{m-2}} \right)^{-1}$$
- Requirements to *G*: $\lambda_i \in [-\rho, \rho]$
  - SOR is not applicable, but SSOR may be used

Chebyshev's Acceleration

- Therefore, for this method, Chebyshev's acceleration $x^{(s+1)} = Gx^{(s)} + c$ consists in:
  - Set $\mu_0 = 1,\ \mu_1 = \rho,\ y^{(0)} = x^{(0)}, y^{(1)} = Gx^{(0)} + c$.
  - Compute the following for m=2, 3, …

$$\mu_m = \left(\frac{2}{\rho\mu_{m-1}} - \frac{1}{\mu_{m-2}} \right)^{-1} \qquad y^{(m)} = \frac{2\mu_m}{\rho\mu_{m-1}}\left(Gy^{(m-1)} + c\right) - \frac{\mu_m}{\mu_{m-2}} y^{(m-2)}$$

- There is no need to expressly compute G and c; iteration will have two stages:

1) $\bar{y} = Gy^{(m-1)} + c$ 2) $y^{(m)} = \dfrac{2\mu_m}{\rho\mu_{m-1}}\bar{y} - \dfrac{\mu_m}{\mu_{m-2}} y^{(m-2)}$

Results – sparse matrices

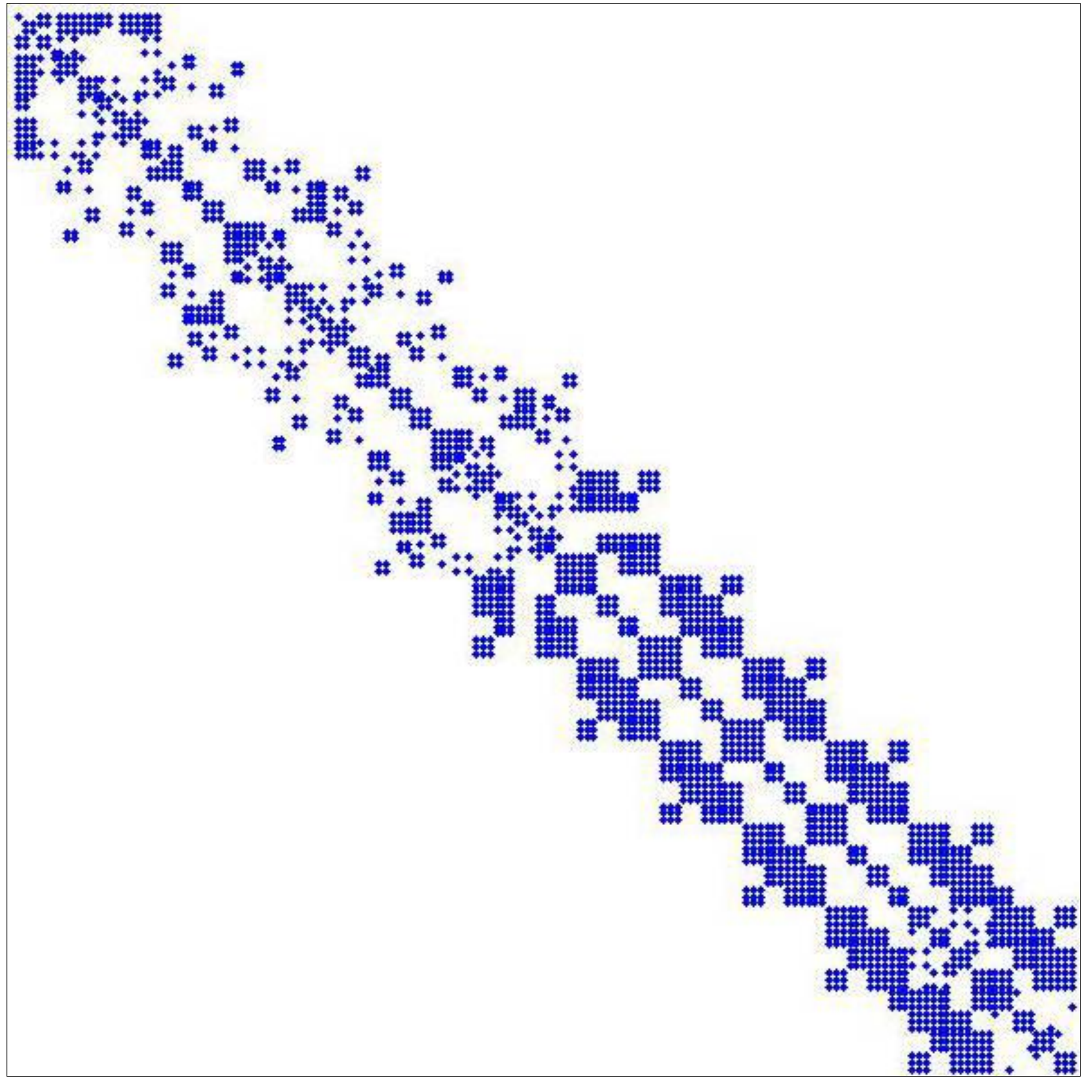
- The University of Florida Sparse Matrix Collection
  http://www.cise.ufl.edu/research/sparse/matrices/
- Parameters of the matrices involved

Name	n	nz	μ_A
mesh1em6	48	306	6.1
bcsstk04	132	3648	$2.3*10^6$
bcsstk05	153	2423	$1.4*10^4$
bcsstk09	1083	18437	$9.5*10^3$
chem97ZtZ	2541	7361	$2.5*10^2$

- All matrices are symmetric positive definite.

Results – Sparse Matrices

- Matrix portrait
  bcsstk05
- Exact solution
  $x^* = \{x_i = 1, 1 \le i \le n\}$
- Right-hand member
  $b = Ax^*$
- System
  $Ax=b$



Results – Sparse Matrices

- The best ω and ρ values are difficult to compute analytically, so they were determined experimentally.
- Method precision is $\varepsilon=10^{-6}$.
- The table shows the number of iterations s.

Problem name	SOR		SOR-Cheb		
	ω	s	ω	ρ	s
mesh1em6	1.9	146	1.9	0.9	35
bcsstk04	1.9	341	1.09	0.99	229
bcsstk05	1.87	986	1.0	0.998	251
bcsstk09	1.95	885	1.11	0.999	537
chem97ZtZ	1.9	144	1.9	0.9	125

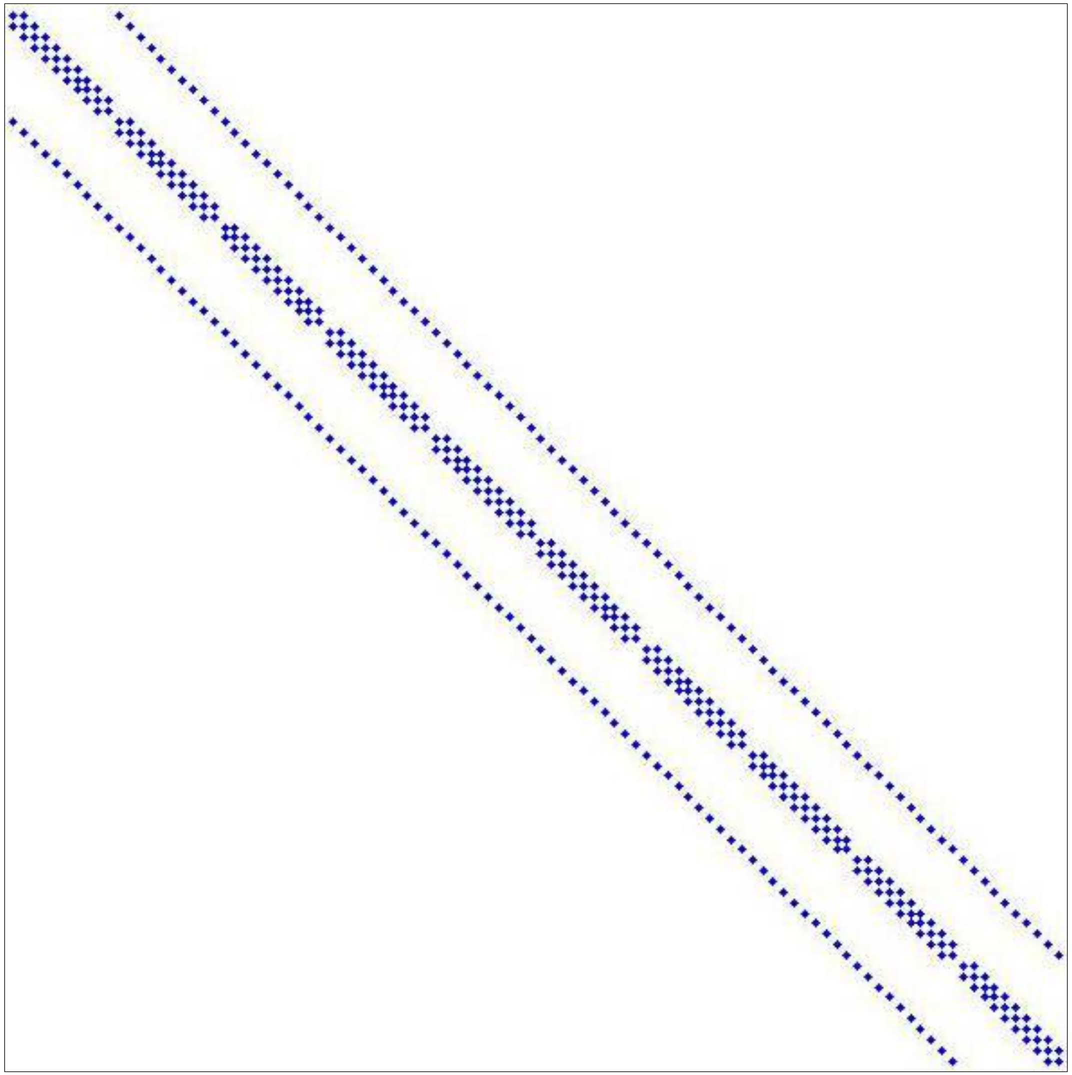
Results – Poisson's Equation

- Linear system appears as a result of PDE discretization
- Test problem with a foregone exact solution may be defined
- $\omega_{opt} = \dfrac{2}{1 + 2\sin(\pi h/2)}$ is known for this type of problems
- $\rho_{opt} = 1 - \dfrac{\pi h}{2}$ is known for this type of problems

- The linear system matrix has a five-diagonal portrait

Results – Poisson's Equation

- Linear system portrait for Poisson's equation



Results – Poisson's Equation

- Method parameters: ρ=0.99, ε=10^{-6}.

n	nz/n	ω	s		
			SOR	SSOR	SSOR-Cheb
10000	$4{,}9\cdot10^{-6}$	1.9397	286	342	53
22500	$9{,}8\cdot10^{-6}$	1.9592	428	512	65
40000	$3{,}1\cdot10^{-7}$	1.9692	569	682	72
62500	$1{,}2\cdot10^{-7}$	1.9753	711	852	123
90000	$6{,}1\cdot10^{-8}$	1.9793	853	1022	91
122500	$3{,}3\cdot10^{-8}$	1.9823	995	1192	85
160000	$1{,}9\cdot10^{-8}$	1.9845	1137	1362	97
202500	$1{,}2\cdot10^{-8}$	1.9862	1278	1532	143
250000	$7{,}9\cdot10^{-9}$	1.9875	1420	1702	276

SOR – Parallel Algorithm

- Iterations are performed in a sequence
- The next approximation components are also computed in a sequence
- Computation of specific components of the next approximation can be parallelized
  - Basic computation consist in calculation of $\sum_{j=1}^{i-1} a_{ij} x_j^{(s+1)}$ and $\sum_{j=i+1}^{n} a_{ij} x_j^{(s)}$
- For calculation purposes, known parallel summing algorithms will be used.

SOR – Parallel Algorithm

- Complexity estimation for parallel summing is

$$2n/p + \delta$$

n – sum length, *p* – number of flows, δ – contingency

- Complexity estimation for a single iteration is

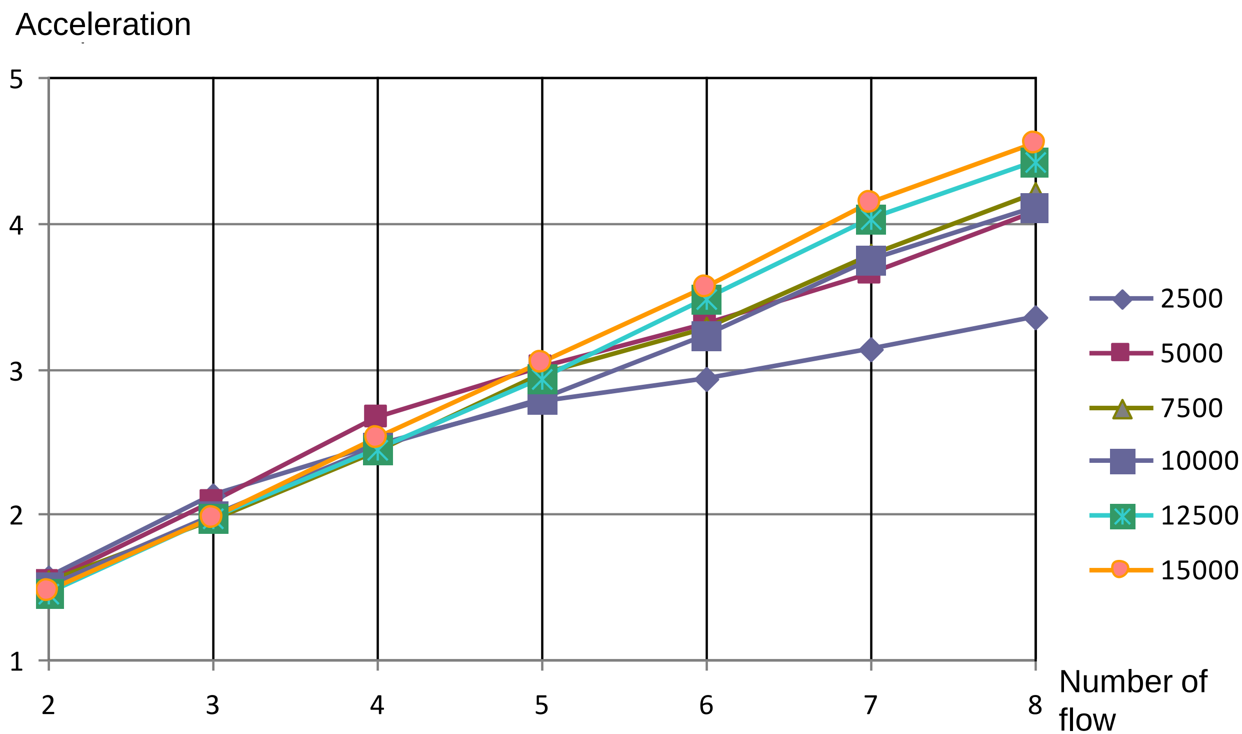
$$t_p = n(2n/p + \delta) + n$$

- Total parallel SOR complexity estimation

$$T_p = L(2n^2/p + n + \delta n)$$

Results – SOR, Dense Matrices

- Acceleration in relation to the single-flow version



Conclusion

- The lecture gives a review of the following:
  - Notion of iterative methods
  - Fixed point iteration method
    - Sequential algorithm and its properties
    - Ways of parallelizing
  - Jacobi and Seidel methods
  - SOR
    - Sequential algorithm and its properties
    - Parallel algorithm
  - Chebyshev's acceleration of iterative methods
  - Experimental results

References

1. David R. Kincaid and E. Ward Cheney. Numerical analysis : mathematics of scientific computing. Brooks/Cole Publishing Company, 1991.
2. Richard L. Burden, J. Douglas Faires. Numerical Analysis. Brooks Cole, 2000.
3. James W. Demmel. Applied Numerical Linear Algebra. SIAM, 1997.
4. Gene H. Golub, Charles F. Van Loan. Matrix Computations. The John Hopkins University Press, 1996.

Web Resources

5. J. Dongarra et al. Templates for the solution of linear systems: building blocks for iterative methods. SIAM, 1994.
6. Y. Saad. Iterative Methods for Sparse Linear Systems. SIAM, 2003.
7. 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

- Darya Rogova (education software codes)