The Ministry of Education and Science of the Russian Federation

Lobachevsky State University of Nizhni Novgorod

Computing Mathematics and Cybernetics faculty

The competitiveness enhancement program
of the Lobachevsky State University of Nizhni Novgorod
among the world's research and education centers

Strategic initiative
“Achieving leading positions in the field of supercomputer technology
and high-performance computing”

# ITERATIVE METHODS FOR SOLVING LINEAR SYSTEMS

## *Practice 4. Solving sparse linear systems using the preconditioned conjugate gradient method*

Nizhni Novgorod

2014

## OBJECTIVES

The purpose of this work is to demonstrate practical implementation of the conjugate gradient method for symmetric sparse matrices using preconditioning and study influence of the computation error on the solution accuracy.

## ABSTRACT

In the course of this laboratory work, we shall review one of iterative methods of solving linear system, i.e. the conjugate gradient method. Students will be proposed to implement this method for symmetric sparse matrices and study its convergence. We will also consider the use of ILU(0)-preconditioner to improve the method convergence. Then we will use multiple precision numbers to implement the method, analyze its convergence and compare the results to theoretical estimates. In the final part of the work, we will develop the parallel method version and evaluate its efficiency.

## GUIDELINES

The theoretical part of this work will describe the general scheme of solving the linear system $Ax = b$ with a symmetric positive definite matrix $A$ using the conjugate gradient method. In the course of laboratory work, formulas to compute iterative approximation of the method will be substantiated in the same way as the minimization problem solution. Then we shall introduce the definition of preconditioner and describe modification of the conjugate gradient method formulas with the *ILU*-preconditioner.

It is proposed to implement the conjugate gradient method for a symmetric sparse matrix stored in the *CRS* format. Sequential implementation will be based on the method computation formulas. Then the program will be modified to use the $ILU(0)$-preconditioner from the laboratory work on Preconditioning Using Incomplete *LU*-factorization to improve the method convergence.

Experimental results will be demonstrated for the matrices from the University of Florida Sparse Matrix Collection with size from 48 through 700,000 based on single and double precision floating point numbers. Experiments show that the number of method iterations exceeds theoretical estimates many times due to loss of accuracy in the course of arithmetic operations and accumulation of computational error. The use of a preconditioner for smaller matrices enabled reduction of the number of iterations by 5 to 9 times when double precision numbers were used or 7 to 15 times in case of single precision numbers. For one of the test matrices, the solution was obtained only with the help of a preconditioner.

To reduce the computational error in the course of arithmetic operations, it is proposed to use the MPFR free library [4] for multiple precision computations with correct rounding. For this purpose, the library is preassembled to run under Windows. Then, the consecutive program version is redesigned to enable the use of **`mpfr_t`** floating point numbers, all arithmetic operations are replaced by the corresponding library functions (sum of two numbers **`mpfr_add()`**, difference of two numbers **`mpfr_sub()`**,product of two numbers **`mpfr_mul()`**, quotient of two numbers **`mpfr_div()`** etc). Experimental results for test matrices show that the use of multiple precision variables ensures consistency with the theoretical estimates, as the number of method iterations is close to the matrix order.

Parallel implementation of the conjugate gradient method is based on OpenMP. As the conjugate gradient method iterations have interdependent data, we will propose to parallelize matrix-vector operations used to compute the method coefficients. For this purpose, the **`#pragma omp parallel for`** directive is applied to the cycles of corresponding functions. Experiments show that the selected scheme is inefficient as the overheads for threads are comparable to the corresponding computation load in terms of time.

## RECOMMENDATIONS FOR STUDENTS

A description of the conjugate gradient method can be found in many publications as it is one of the best known methods of solving linear systems. A brief method description including pseudocode algorithms can be found in [1]. A detailed description of the method including the respective conclusions, theoretical justification and application examples can be found in [2]. The method is reviewed in terms of matrix operations in [3].

## REFERENCES

1. J. Dongarra et al. Templates for the solution of linear systems: building blocks for iterative methods. SIAM, 1994.
2. Y. Saad. Iterative Methods for Sparse Linear Systems. SIAM, 2003.
3. Gene H. Golub, Charles F. Van Loan. Matrix Computations. The John Hopkins University Press, 1996.
4. Fousse L., Hanrot G., Lefèvre V., Pélissier P., Zimmermann P. MPFR: A multiple-precision binary floating-point library with correct rounding // ACM Trans. Math. Softw. – 2007. – Vol. 33, No. 2. – P. 1–14.

## PRACTICE

1. Apply the preconditioner *ILU(p)* to the conjugate gradient method. How will it influence the convergence rate?
2. Modify your program to implement the biconjugate gradient method. Compare convergence rates of the conjugate and biconjugate gradient methods.

## TEST

1. Can the CG method be considered as a direct method of solving linear systems?
   a. + Yes
   b. Yes, but only for well-conditioned matrices
   c. No
2. What problems are solvable using the conjugate gradient method?
   a. + Problems with a symmetric positive definite matrix
   b. Problems with a general matrix
   c. Problems with a symmetric matrix
3. What optimization problem does the solution for the system$Ax = b$ with a symmetric positive definite matrix correspond to?
   a. + $F(x) = \frac{1}{2}(Ax, x) - (b, x)$
   b. $F(x) = \frac{1}{2}(Ax, x) - b$
   c. $F(x) = Ax - b$
4. Which matrix-vector operation is not used for computation of the conjugate gradient method coefficients?
   a. Matrix multiplication by a vector
   b. Scalar product of two vectors
   c. +Product of two matrices
5. Which condition is the $\beta_i$ computation formula responsible for?
   a. Conjugacy of residual and direction vectors $r_i$ and $p_i$
   b. Conjugacy of residual vectors $r_i$ and $r_{i-1}$
   c. + Conjugacy of direction vectors $p_i$ and $p_{i-1}$
6. What computation formula appears in the method when the preconditioner is used?
   a. $r_{i+1} = M^{-1} z_{i+1}$
   b. + $z_{i+1} = M^{-1} r_{i+1}$
   c. $z_{i+1} = M^{-1} p_{i+1}$

7. Why may the number of method iterations exceed the theoretical estimate for double-precision floating point numbers?
    a. Computational error accumulates due to number representation.
    b. The matrix is ill-conditioned
    c. +Accuracy is lost and computational error accumulates due to number representation.
8. What will the variable be equal to after initialization by the **`mpfr_init()`** function?
    a. + NaN
    b. 0
    c. Plus infinity
9. What will rounding of 4.5 and 7.5 using **`MPFR_RNDN`** (rounding to the nearest) result in?
    a. + 4 and 7
    b. 5 and 8
    c. 4 and 8
10. Which operator is not subject to reduction in OpenMP?
    a. ^
    b. + /
    c. &&
11. What is the reason for low scalability of the implemented parallel version?
    a. Use of reduction
    b. +Low computation load on threads
    c. Atomic operations for shared variables.