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”

NUMERICAL METHODS FOR SOLVING DIFFERENTIAL EQUATIONS

Lecture 4. Solving Partial Differential Equations as Illustrated by the Dirichlet Problem Posed for Poisson’s Equation

Nizhni Novgorod

2014

OBJECTIVES

The purpose of this lecture is to study methods for numerical solution of partial differential equations and construction of corresponding parallel algorithms. As an example, this lecture describes the Dirichlet problem posed for Poisson's equation.

ABSTRACT

The lecture describes numerical solution of the Dirichlet problem posed for Poisson's equation. It defines the problem and derives a difference scheme to solve it. To solve a linear system resulting from the difference scheme, the SOR method will be used. This lecture describes a parallel numerical solution algorithm based on wavefront computational grid traversal. To optimize the said algorithm, perform computations per grid block as part of wavefront grid processing. The lecture describes a method of solving the Poisson's equation that combines the tridiagonal matrix algorithm and Fourier transform, and an approach to its parallelization. It gives experimental results for the described algorithms.

GUIDELINES

The lecture describes numerical solution of the Dirichlet problem posed for Poisson's equation. $\Delta u(x,y)=-f(x,y)$, $x\in(0,1)$, $y\in(0,1)$;$u(0,y)=\psi_1(y)$, $u(1,y)=\psi_2(y)$,$u(x,1)=\psi_3(x)$, $u(x,0)=\psi_4(x)$.This problem can be considered as a problem of determining deformation of an elastic membrane exposed to external effects whose edges are fixed in a predetermined manner, or as a problem of determining stationary temperature distribution in a square plate exposed to external heat sources whose edges are subject to sustainable preset temperature specifications. To find the numerical problem solution, introduce into the domain a uniform rectangular grid with n partitions in the x variable and m partitions in the y variable. Then, derive the second-order difference scheme that enables finding the solution $v(x,y)$ within the grid points.

The constructed difference scheme is a linear system with respect to an unknown vector v sized $(n-1)\cdot(m-1)$ with components

$$v = (v_{11}, \ \dots \ , v_{n-1,1}, v_{11}, \ \dots \ , v_{n-1,2}, \ \dots \ v_{1,m-1}, v_{2,m-1}, \ \dots \ , v_{n-1,m-1}) \ ,$$

where v_{ij} is the grid function value $v(x,y)$ in the point (x_i,y_j). The matrix constructed on the basis of a difference scheme is a block diagonal sparse matrix. Its main diagonal has numbers in the form of $A = -2\left(1/h^2 + 1/k^2\right)$, while other non-zeroes can be equal either to $1/h^2$ or $1/k^2$ (here, h and k are grid steps). The matrix is symmetric, nonsingular and negative definite. By multiplying the system equations by −1, the difference scheme is formulated as a linear system with a symmetric positive definite matrix. To solve it, one can use iterative methods. Now let us discuss dependence of iterative methods convergence on grid steps and matrix condition number.

To solve linear systems, this lecture proposes to use the SOR method. For this purposes, the SOR method computational formulas are introduced subject to the difference scheme peculiarities. The SOR method optimal parameter is known for the considered problem. To implement the SOR method, each next k^{th} approximation of v_{ij} is computed based on the last k-th approximation of $v_{i-1,j}$ и $v_{i,j-1}$ and the second last $(k-1)^{\text{th}}$ approximation of $v_{i+1,j}$ and $v_{i,j+1}$.

Then a parallel algorithm is constructed to perform only the same actions as the sequential version and ensure exactly the same solutions of the initial problem. Such an algorithm divides the method iteration into a sequence of steps, each of them being used for computation of points on the additional grid diagonal with the number determined by the stage number. The resulting scheme is called a *wavefront* scheme. However, the computational wavefront shows poor compliance with the processor cache utilization rules. The considered algorithm is based on row-wise data storage and diagonal-wise wavefront which leads to low cache utilization efficiency. A possible way to improve the situation proposed in this lecture is to use a procedure of processing

of a certain rectangular subregion (block) of the computation grid as a set of actions divided among the threads.

Now let us list experimental results of solving the test Dirichlet problem posed for Poisson's equation. For experimental purposes, the partition number for x and y was the same and varied from 250 through 1,500; the SOR method stop criterion was accuracy. Experimental results showed that implementation speed up demonstrated almost no increase for all grids if the number of threads exceeded 3. This can be explained by low complexity of operations performed by each thread in case of wavefront computations. In this case, overhead outweigh the benefits of parallelization. To confirm this, experiments with an increased computational load on threads took place. They showed that in this case speed up was subject to linear growth for large grids (n>500) and reached 5 for 8 threads. For smaller grids, speed up showed insignificant growth or even reduced if the number of threads exceeded 4.

At this moment, let us study the method of solving Dirichlet Problem posed for Poisson's equation based on Fourier transform. Let us consider an auxiliary single-dimension problem in eigenvalues for difference equations. It is known that this problem has a fully orthonormal system of eigenfunctions. If v_{ij} and f_{ij} are considered as single-dimension grid functions depending only on j, j=1,…,n–1, they may be expanded into the auxiliary problem eigenfunctions. In this case, to find the Fourier coefficients $c_k(i)$ of the function v_{ij} an auxiliary tridiagonal linear system has to be solved.

Thus, the proposed algorithm consists of three steps:

- Computation of the Fourier coefficients $f_k(i)$ of the right-hand part f_{ij}, for each $i = 1,\ldots, n-1$;
- Computation of the Fourier coefficients $c_k(i)$ as a solution to the auxiliary linear system using the tridiagonal matrix algorithm for each $k = 1,\ldots, n-1$;
- Restoration of the solution v_{ij} for each $i = 1,\ldots, n-1$ by Fourier transform.

Fourier coefficients are computed by means of fast Fourier transform which requires $O(n \log_2 n)$ operations to be computed n times for the first and the last stages. Complexity of the tridiagonal matrix algorithm is $O(n)$. Thus, the complexity of the tridiagonal matrix algorithm is $O(n^2 \log_2 n)$.

At each algorithm stage, computations are performed in loops with each iteration independent on other ones in terms of data. This is why, to construct a parallel algorithm it is proposed to parallelize the loop executed at each problem solution stage.

The disadvantage of such algorithms (both parallel and sequential) is the necessity to explicitly compute the eigenvalues and eigenfunctions of the single-dimension problem. Therefore, this approach is not applicable in the cases when the problem solution for eigenvalues is impossible to write explicitly.

The lecture lists experimental results for the test problem allowing for analytical solution. A FFT application implemented as part of the practice "Development, Optimization and Parallelization of Fast Fourier Transform as Applicable to Filtration Problem" is compared to the Intel MKL-based application and the FFTW-based one. The grid partition number varied from 128 through 8,192. Comparison of sequensial versions showed that for the implementation proposed in the laboratory work the runtime was 1.3 times longer than that of the FFTW implementation and 1.8 times longer that that of the Intel MKL implementation with 8,192 grid partitions. If the partition number is less or equal to 2,048, the applications have close runtimes. The speed up of the own parallel version implementation exceeds 5 for a grid whose partition number is greater than 4,000. Such parameters are ensured due to absence of data dependencies at each algorithm stage.

RECOMMENDATIONS FOR STUDENTS

See [2, 3] for a detailed description of numerical methods for solving differential equations. See [1] for the methods for solving linear systems and the tridiagonal matrix algorithm and the SOR method in particular.

REFERENCES

1. Gene H. Golub, Charles F. Van Loan. Matrix Computations. The John Hopkins University Press, 1996.
2. Hoffman J.D. Numerical Methods for Engineers and Scientists, 2nd Edition. New York: CRC Press, 2001.
3. Kincaid D.R., Cheney E.W. Numerical Analysis: Mathematics of Scientific Computing, 3rd Edition. Pacific Grove: Brooks Cole, 2001.

PRACTICE

1. Implement a sequential program for solving the test Dirichlet problem posed for the Poission's equation. Compare efficiency of your implementation with the experimental results mentioned in the lectures.
2. Implement a parallel program of solving the test Dirichlet problem posed for the Poission's equation using the wave and block wave computation scheme. Find the best block size for the latter. Evaluate efficiency of the implementations and make conclusions.
3. Do the laboratory work "Solving Sparse Linear Systems By Iterative Methods: Problem of Heat Diffusion in a Plate". Compare efficiency of your implementation with the experimental results mentioned in the lectures. How can you reduce the computation time?

TEST

1. What approximation order for interior grid points does the studied numerical solution of the Poisson's equation have? The grid step in the x variable is h, the step in the y variable is k.
    a. + $O(h^2 + k^2)$
    b. $O(h^2 + k^3)$
    c. $O(h^3 + k^2)$
2. What is the size of a linear system matrix constructed using a difference scheme with n partitions in the x variable and m partitions in the y variable? Grid points located on the bounds of the admissible region are not taken into account.
    a. $(n-1)\times (m+1)$
    b. + $(n-1)\times (m-1)$
    c. $n \times m$
3. What linear system matrix results from the difference scheme?
    a. Symmetric positive definite
    b. + Symmetric negative definite
    c. Non-symmetric positive definite
4. What is the order of value computations in the points of the grid v_{ij} if the SOR method is used for the considered problem? The partition number is n for x and m for y.
    a. outer loop for i from 1 through n and inner loop for j from 1 through m
    b. outer loop for i from 1 through $n-1$ and inner loop for j from 1 through $m-1$.

c. + outer loop for j from 1 through $m-1$ and inner loop for i from 1 through $n-1$.

5. What is the order of parallel value computations in the points of the grid v_{ij} if the SOR method is used for the considered problem?
    a. Wavefront starting from the lower left angle of the system matrix
    b. + Wavefront starting from the upper left angle of the system matrix
    c. Wavefront starting from the upper right angle of the system matrix
6. What auxiliary problem must be solved when Fourier transform is used to solve Poission's equation?
    a. + Finding eigenvalues and eigenfunctions
    b. Finding eigenvalues and eigenvectors
    c. Finding eigenvalues
7. What grid function requires finding Fourier coefficients to solve Poisson's equation?
    a. $f_{ij}(j)$
    b. $v_{ij}(j)$
    c. + $v_{ij}(j)$ and $f_{ij}(j)$
8. What grid function requires solving a linear system for the considered problem?
    a. $f_{ij}(j)$
    b. + $v_{ij}(j)$
    c. $v_{ij}(j)$ *and* $f_{ij}(j)$
9. What computational complexity does the Fourier transform-based Poisson's equation solution algorithm have?
    a. $O(n^2)$
    b. $O(n \log_2 n)$
    c. + $O(n^2 \log_2 n)$
10. What stages of solving Poission's equation using Fourier transform allow for parallelization?
    a. + each step: Fourier coefficient computation, tridiagonal system solution, solution restoration
    b. Fourier coefficient computation and solution restoration
    c. tridiagonal system solution