TMLAI TRANSACTIONS ON MACHINE LEARNING AND ARTIFICIAL INTELLIGENCE
SOCIETY FOR SCIENCE AND EDUCATION UNITED KINGDOM
VOLUME 2, ISSUE 6
ISSN: 2054 - 7390

An Analytical Approach of Computational Complexity for the Method of Multifluid Modeling

[1]A.K. Borah and [2]P.K. Singh
[1]Department of Mathematics, R G Baruah College, Gauhati University, Guwahati-India,
[2]Department of Mathematics, Allahabad University, Allahabad UP, India,
[1]borah.arup@yahoo.com, [2]pramod_ksingh @rediffmail.com

ABSTRACT

In this paper we deal the building blocks of the computer simulation of the multiphase flows. Whole simulation procedure can be viewed as two super procedures, the implementation of VOF method and the solution of Navier Stoke's Equation, Moreover, a sequential code for a Navier Stoke's solver has been studied.

Keywords: Bi-Conjugate Gradient Stabilized (Bi-CGSTAB), SIMPLE algorithm, Krylov Subspace, ILUT function, Preconditioner, multifluid flows

1 Introduction

The solution of multifluid model have been developed in this studies. Implementation of both steps of VOF (PLIC) method have been derived. The SIMPLE algorithm is used for solving the momentum equation generates large sparse linear systems of equations. These methods are solved by iterative method such as K-S methods. On the other hand convergence rate of this method can be accelerated by preconditioning techniques. Development and improvement of numerical schemes have encouraged researchers to investigate almost every branch of fluid dynamics and its application to real life problem.

Multiphase flows occur in mant industrial and natural phenomena, petroleum refining [1], biological flows [2] and interaction of air with the sea surface [3]. The simulation of multiphase fluid flows in one of the most challenging problem in CFD as it involves in modeling of sharp interfaces separating multiple fluids. The numerical simulation—into fluid flow modeling and multiphase modeling in Fig. (1)

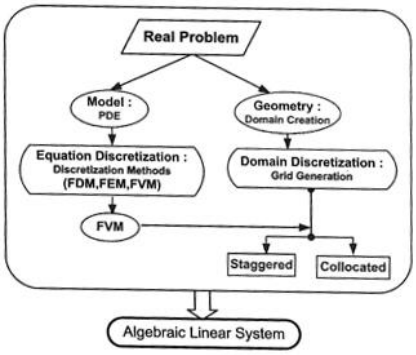

Figure 1: PDE to Linear System

DOI: 10.14738/tmlai.26.560
Publication Date: 21th December, 2014
URL: http://dx.doi.org/10.14738/tmlai.26.560

The fluid flow properties (velocity, pressure, etc.,) can be represented by partial differential equations such as N-S equation. Furthermore, numerical solutions of these equations constitute fluid flow modeling. Multiphase modeling involves defining the interface between various phases and then calculating the flux in all directions by using the solution from fluid flow model.

2 Multiphase modeling

Maintaining sharp interface during fluid transportation is a difficult task in the modeling of interfacial flows [4]. The interface between two phases can be modeled by scalar transport equation [5]. The modelling involves the construction and movement of the interface. An effective approach for interface modeling is interface capturing, we study a VOF method based on interface capturing approach has been applied. This method has two steps reconstruction and advection of the interface between two fluids [6-7]. Moreover, the interface is established by calculating the volume fractions of each fluid. The transportation algorithm is employed for the movement of the interface. The main challenging of the modeling of the interfacial fluid flow is the implementation of a method which can efficiently move the sharp interface without stretching and wrinkling [6] [8]. In the latter step the interface is approximated by a straight line (a plan 3 dimension) [9]. On the other hand, the area of the geometrical shape represented by triangles or rectangle below the lines is calculated to evaluate the volume of the fluid based on the position of the interface. The interface is approximated in the subsequent time steps by using the volume of fluid of the previous time step and this is where time difficulty maintaining the sharpness of the 8interface in the interface approximation arises [10].

3. *Interface Reconstruction* In the volume of fluid (PLIC) method, the interface between two fluids in a grid cell is approximated by a line segment which intersects the cell's faces. But the line segment divides the cell into two parts- each of them containing one of the two fluids as shown in polygon ABFCGD in Fig.(2), the notations are as flows,

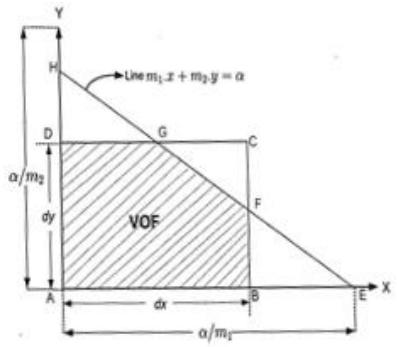

Figure 2: Cell ABCD is cut by the straight line EH and contains fluid 2 in region ABFGD

Rectangle ABCD, represents a grid cell, dx the length of the cell in X direction, dy the length of the cell in Y direction, line EH approximation of the interface, polygon ABFGD: volume of one fluid in the cell and α/m_1, projection of the segment EH on the X axis.

The general equation of straight line L1 with normal m̃ may be represented as;

$$m1x+m2y = 0 \quad (1)$$

Where m1 and m2 are the components of the normal vector in the x and y directions respectively and α being the constant which is related to the distance of the line from the origin. The coordinate at the

points at which the line intersects the axes X and Y are respectively (α/m_1, 0) (0, α/m_2) to the points E and H in Fig. (2).

However, in the simulation the values of volume fractions are provided initially for all the cells. But at the next time step the fluid mixture moves and the interface changes its position and hence new values of the volume fractions have to be calculated. In order to evaluate the volume fraction of one fluid in a cell, one has to calculate the area below the line L1, polygon ABFGD in Fig. (2).

2.1 The Estimation of the Normal Vector

In the first part of reconstruction, a normal vector is estimated by a nine-point finite difference formula [6]

$$m = \nabla C \tag{2}$$

Eq. (2) represents the gradient of the colour function C in the direction of coordinates axes. The discrete approximation to Eq. (2) is given by

$$\nabla C = \begin{pmatrix} \nabla^x & C \\ \nabla^y & C \end{pmatrix} \equiv \begin{pmatrix} m_1 \\ m_2 \end{pmatrix} \tag{3}$$

Where ∇^x and ∇^y denotes the gradient in the x and y direction respectively. For approximating the values of these gradient terms, we choose eight nearest neighbors in 2 dimension all the neighbors sharing the vertex, [11]. For a uniform mesh, in a grid cell at location (i, j), the gradient terms of Eq.(3) in the coordinate form can be expressed [6]

$$(m_1)_{i,j} = \frac{1}{\partial x}\left(C_{i+1,j+1} + 2\,C_{i+1,j} + C_{i+1,j-1} - C_{i-1,j+1} - 2C_{i-1,j} - C_{i-1,j-1}\right) \tag{4}$$

$$(m_2)_{i,j} = \frac{1}{\partial y}\left(C_{i+1,j+1} + 2\,C_{i,j+1} + C_{i-1,j+1} - C_{i+1,j-1} - 2C_{i,j-1} - C_{i-1,j-1}\right) \tag{5}$$

Eq.(3)-(4) represents the normal vector estimation formula for x, y directions [7], this scheme produces a good estimation of the normal vector. An investigation of accuracy test of different normal estimations schemes have been demonstrated [11] that the linear fit (using the nine-point) stencil produce the same order error as other methods such as quadratic fit which require more computations.

2.2 The calculation of VOF from the normal vector and the line constant

$$\text{ABFGD} = \underbrace{\Delta AEH}_{\Delta big} - \underbrace{\Delta BEH}_{\Delta_1} - \underbrace{\Delta BEH}_{\Delta_2} \tag{6}$$

$$\text{Area} = \underbrace{\frac{\alpha^2}{2m_1 m_2}}_{\Delta big}\left\{1 - H(\alpha - m_1\,dx)\left(\frac{\alpha - m_{1dx}}{\alpha}\right)^2 - \underbrace{H(\alpha - m_2\,dy)\left(\frac{\alpha - m_{2\,d}y}{\alpha}\right)^2}_{\Delta_2}\right\} \tag{7}$$

where H(x) be the Heaviside step function

$$H(x) = \{0 \quad x < 0;\ 1 \quad x > 0 \tag{8}$$

Eq.(6)-(7) calculates the area below the line in Fig.(2) , the procedure for calculating area below the line in two cases for the x direction using Eq.(7) the case of a line with positive slope intersecting the grid cell at the axes as shown in Fig.(3) represents the three triangles

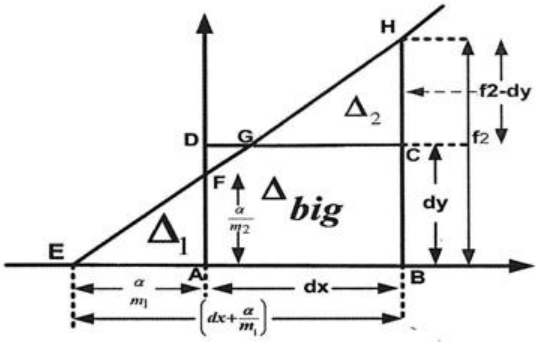

Figure 3: Line of Positive slop intersects the cell faces

Δ_{big}– triangle BEH, Δ_1 – triangle AEF, Δ_2– triangle CGH

Now the area of the polygon ⌂ is required to calculate the actual volume of the fluid contained in the cell.

$$\Delta_{big} = \left|0.5 \times \left(dx + \frac{\alpha}{m_1}\right) f_2\right|$$

$$\Delta_1 = \left|0.5 \times \frac{\alpha}{m_1} \frac{\alpha}{m_2}\right|$$

$$\Delta_2 = \Delta_{big} \frac{(CH)^2}{(BH)^2}$$

$$\Delta_2 = \Delta_{big} \frac{\{(f_2 - dy)^2\}}{f_2^2} = \Delta_{big} \left(1 - \frac{dy}{f_2}\right)^2$$

$$\left(\frac{\alpha}{m_1} < 0\right) \Rightarrow h1 = 1 \; and \; (f_2 - dy) > 0 \Rightarrow h2 = 1$$

$$\text{Area} = \Delta_{big} \{1 - h1.\Delta_1 - h2.\ \Delta_2\} \quad (9)$$

In Eq.(9) h1 and h2 represents the Heaviside step-function H(α —m1dx),

H (α —m2dx) respectively. A clearer picture of this case is depicted in Fig.(4).

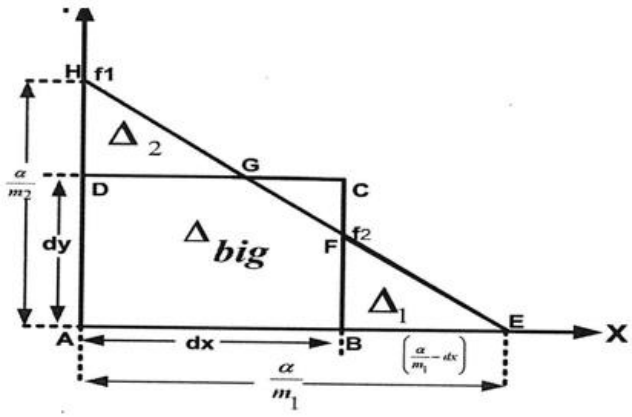

Figure 4: Line of negative slop intersects the cell faces

$$\Delta_{big} = 0.5 \times \frac{\alpha \times \alpha}{m_1 m_2}$$

$$\Delta_1 = \Delta_{big} \left(\frac{BE}{AE}\right)^2 = \Delta_{big} \times \left(\frac{\frac{\alpha}{m_1} - dx}{\frac{\alpha_1}{m_1}}\right)^2$$

$$\Delta_2 = \Delta_{big} \left(\frac{DH}{AH}\right)^2 = \Delta_{big} \times \left(\frac{\frac{\alpha}{m_2} - dy}{\frac{\alpha_2}{m_2}}\right)^2$$

$$\Delta_2 = \Delta_{big} \times \left(\frac{\alpha - m_2 dy}{\alpha}\right)^2$$

$$(\alpha - m_1 dx) > 0 \Rightarrow h1 = 1 \; and \; (\alpha - m_2 dy) > 0 \Rightarrow h2 = 1$$

$$\text{Area} = \Delta_{big} \{1 - h1.\ \Delta_1 - h2.\ \Delta_2\}$$

From the Eqs. (9)-(10) it has been demonstrated that when the line intersects the cell such that it generates two small triangles. But other cases may arise when there is only one triangle h1= 0 or h2= 0 or there is no triangle present at all as such h1=h2=0.

2.3 Performance of Preconditioners

The preconditioners accelerates the convergence rate of the Krylov-Subspace method. During simulation the matrix is being generated at each time step. Thus the matrix entries changes at each time step which changes the matrix properties. In this study, the Bi-CGSTAB method has been employed because it has been found in literature to provide small convergence for non-symmetric matrices. Now the effects of preconditioners applied to the Bi-CGSTAB method implemented on different problem are demonstrated.

To simulate multifluid flow, the VOF method has been implemented. This method treats the mixture of two fluids as one fluid which is determined by the interface. During the advection the mesh is kept fixed and the interface is reconstructed from the values of the colour function and its gradient in a grid cell. These values contribute to the calculation of the coefficients matrix entries.

Since the fluid moves at each time step, it has been observed that the magnitude of the matrix entries change because of the changes in the interface position. Due to this change the condition number- the ratio of the highest to the lowest eigenvalues of the matrix may vary.

A matrix with high condition number makes Krylov-Subspace solvers converge slowly and so, in order to increase the convergence rate preconditioning techniques are applied [12-13]. Incomplete to L-U Symmetric Successive Over Relaxation (SSOR), Diagonal Scaling (DS) are the are the most appropriate preconditioners for the solvers.

3 Computational Complexity

In ILU factorization, the original matrix A is decomposed into two matrices L and U. Its Algorithm in dense format is shown in 4.1. In this Algorithm, there three nested for loops required (Steps -1, 2 and 9), these nested loops

```
1       for ( i = 0 to n)
2       for (j = i+1 to n) do
3       Uj,i = A j,i  ;
```

```
4       if (Uj,i = = 0.0) then
5       Lj,i  = 0.0 ;
6       else
7       L_(j,i=) U_(j,i)/U_(i,i)    ;
8       end if
9       for (k = i+1 to n) do
10      if (Uj, k ≠ 0.0) then
11      Uj, k  − = L j, i X Ui, k
12      end if
13      end for
14      end for
15      L i,, i = 1.0 ;
16      end for
```

Algorithm 4.1 Dense ILUT algorithm [14] generates the data depending of the matrix elements of L and U. This depending implies that for calculating the elements of the (i+1)th , (i+2)th rows, the elements from the ith or previous row are required . Further, this data dependency is a hindrance to parallelizing the Algorithm [15]. In the parallel version the matrix is divided into different parts which are available on different processors of the parallel computer [16]. Therefore, the elements of previous rows may not available on the same processor and those elements have to be brought from other processors.

In Algorithm 4.1 there are three for loops, its computational complexity can be calculated by observing the number of counts in each loop. The outer loop runs from 0 to (n—1) and other loop run from (i+1) to (n—1). The total number of counts can be expressed

$$\sum_{i=0}^{n}(n-i)^2$$

Which has complexity has order $\mathcal{O}\ (n^3 - n^2)$ but in algorithm ILUT algorithm in diagonal format, there is one for loop so its complexity is given as$\mathcal{O}(\acute{n})$. Moreover, the matrix vector product in Algorithm requires only one for loop. Therefore, its computational complexity can also be given as n.

The computational complexity of the matrix-vector products in other sparse formats [17-18] are 5n.

4 Discussion and Conclusion

The main conclusions can be summarized from the above studies:

(i) the diagonal format occupies less memory storing penta diagonal matrices.

(ii) solving the linear system consumes most of the computational time of the simulation.

(iii) a short description of the iterative methods has been investigated.

(iv) the Bi-CGSTAB method requires four inner products and two matrix vectors products.

(v) these products are developed a diagonal format reducing the computational complexity of the solver.

(vi) the need for preconditioners has been highlighted.

(vii) the main computational steps of Krylov-Subspace methods have been demonstrated.

(viii) the parallel Bi-CGSTAB method has been integrated into Navier Stokes solver.

Furthermore, the all the parallel computational tools (C++ code with MPI) for simulating the multiphase flow phenomena have been investigated. In the future work, we will further improve the algorithm.

5 Acknowledgment

The author would like to acknowledge the financial support from the University Grants Commission, Bahadurshah Zafar Marg, New Delhi, India under F. No 40-236/2011 (SR).

REFERENCES

[1] Mayer, A and Lenhard, R., Special Issue on Multiphase Flows and Chemical Transport, Advances in Water Resource, 1998. 21: p.75-76.

[2] Christopher, E., Fundamentals of Multiphase Flows Cambridge University, Press U.K. 2005

[3] Melvile, W., The Role of Surface Wave Breaking in Air-Sea Interaction, Annual Review of Fluid Mechanics, 1996. 28: p.279-321.

[4] Rider, W and Kothe, D., Reconstructing Volume Tracking, Journal of Computational Physics, 1998.141: p.112-152.

[5] Greaves, D., A Quadrative Adaptive method for Simulating Fluid flow with moving Interfaces, Journal of Computational Physics, 2004. 194: p. 35-56.

[6] Rudman, M., Volume Tracking Methods for Interfacial Flow Calculations, International Journal of Numerical methods in Fluids, 1997. 24: p. 671-691.

[7] Denis, G ., Ali, N., Scardovelli, R., and Zaleski, S. Volume-of-fluid interface tracking with Smoothed Surface Stress Methods for Three Dimensional Flows, Journal of Computational Physics, 1999. 152: p. 423-456.

[8] Alibadi, S and Shujaee, K., Free surface flow simulations using parallel finite element method, SIMULATION, 2001. 76(5): p. 257-262.

[9] Scardovelli, R and Zaleski, S., Analytical relations connecting Linear Interfaces and Volume Fractions in Rectangular Grids, Journal of Computational Physics, 2000. 164(6): p. 228-237.

[10] Ruben, S and Zaleski, S., Interface Reconstruction with Least-Square Fit and Split Eulerian Lagragian Advection, International Journal for Numerical Methods in Fluids, 2003. 41: 251-274.

[11] Scardovelli, R and Zaleski, S., Interface Reconstruction with Least Square fit and split Eulerian – Lagragian advection. International Journal of Numerical Methods in Fluids, 2003. 41: p. 251-274.

[12] Saad, Y., Preconditioning Techniques for Nonsymmetric Indefinite Linear systems. Technical Report, Centre for Supercomputing Research and Development, 1992. University of Illinois at Urbana Champaign.

[13] Sun, J, Cao, J and Yang., Paralled Preconconditioners for large scale partial differential equations systems, Journal of Computational and Applied Mathematics, C. 2009. 226: p. 125-135.

[14] Saad, Y., Iterative methods for Sparse Linear System. PWS Publishing Company, 1996. International Thompson Publishing Inc. Boston.

[15] Basermann, A., Parallel Block ILUT/ILDT Preconditioning for Sparse Eigen Problems and Sparse Linear system. Numerical Linear Algebra with Applications, 2000. 7: p. 635-648.

[16] Li, K, First and Scalable Parallel Matrix Computations on Distributed memory Systems. 2005. In 19th IEEE International Parallel and Distributed Processing Symposium.

[17] Shahnaz, R., Usman, A., and Chugati, I Implementation and Evaluation of parallel sparse Matrix-Vector products on Distributed Memory Parallel Computers, Barcelona, Spain. In IEEE International Conference on Cluster Computing (CLUSTER), 2006. Barcelona, Spain, pages 1-6.

[18] Straubhaar, J. Parallel Preconditioners for the conjugate Gradient Algorithm using Gram Scimidt and least square methods, Parallel Computing, 2008. 34(10): p. 551-569.