A FAST AND ACCURATE CFD SOLVER FOR INDOOR AIRFLOWS

Mohammad Mortezazadeh Dorostkar[1], Liangzhu (Leon) Wang[2]

[1,2]Deptartment of Building, Civil & Environmental Engineering

Concordia University, Montreal, Canada

ABSTRACT

Modeling of indoor airflows using computational fluid dynamics (CFD) techniques can be difficult and time consuming, especially for complicated problems or for big buildings. A fast and accurate CFD solver can significantly shorten the time of the analysis with acceptable accuracy for engineering analysis. The aim of this paper is developing a fast and accurate CFD solver to simulate indoor airflow problems at real time. To achieve this goal, we use a combination of Eulerian and Lagrangian algorithms, namely semi-Lagrangian method, equipped with one well-known mathematical method, multigrid solver, to increase the convergence rate. In the present algorithm, convection term is solved by semi-Lagrangian algorithm and diffusion term and Poisson equation are solved by an implicit and V-cycle multigrid solver. The results show that we can use large time steps without any instability problem and consequently achieve better convergence in comparison with standard semi-Lagrangian algorithms and conventional Eulerian methods. Additionally, we discussed the stiffness of solving the Poisson equation that was not found in the previous works of semi-Lagrangian method.

INTRODUCTION

Modeling of indoor airflows using computational fluid dynamics (CFD) techniques can be difficult and time consuming, especially for complicated problems or for big buildings. A fast and accurate CFD solver can significantly shorten the time of the analysis with acceptable accuracy for engineering analysis, in particular, for building industry. For example, building engineers can benefit from this solver in the initial design stage to design a good and optimized air conditioning and ventilation system without lengthy and detailed engineering analysis. Additionally, in the later operating stage of a building, the fast solver can play a crucial role in building operation and controls, for example, when an emergency, e.g. fire, happens in a building, the solver may provide real-time control strategies to help rescuers control the fire or manage occupants' evacuation in the building. Another example of the emergency situations is the spreading of polluted air in a building. With a real-time and accurate solver, we will be able to understand how the pollutant spreads in the building at real time from the source, and possibly predict future states of the spreading so we can react before any damages on occupants' health may occur.

Conventional CFD solvers, such as those based on Eulerian and Lagrangian methods, often perform unsatisfactorily for big and complicated indoor airflow problems. In the Eulerian algorithms, we consider our fluid as a continuum fluid. Then we define a fixed computational domain, Eulerian mesh, within our real domain. Finally, we consider all fluid flow variables, including pressure, velocity, density, etc., as fields, a function of space and time, within the fixed computational domain, and develop and solve the conservation equations on a control volume basis. A fast CFD solver often replies on explicit algorithms, where there is one important constraint, CFL (Courant-Friedrichs-Lewy) condition, causing the time step to be small enough to ensure numerical stability but potentially slow down the simulation significantly. In the Lagrangian algorithms, we will consider our fluid in the flow as a large number of individual particles. Then we follow each particle's path, characteristic curves, and by this way, we will solve conservation of equations. Although the CFL condition may not be relevant in this algorithm, the computational domain, the Lagrangian mesh, is not fixed so new computational domain needs to be generated at each time step, which results in a more complicated and even slower solution. More details about these two approaches are presented through the literatures (Shirolkar (1996), Loth (2000), Lakehal (2002)).

During past decades, many researchers tried to develop some other solvers to simulate indoor environment such as zonal and multizone models (Eriksson et al., 2002, Wang et al., 2008). Many simplifications are considered to achieve simpler and faster solvers than Eulerian and Lagrangian methods. But their major deficiency is the simulation accuracy is case dependent and cannot be guaranteed for all cases.

There are increasing amount of efforts finding a fast and accurate approach known as semiLagrangian approach, which is a combination of both Eulerian and Lagrangian methods. In fact, this method considers the fluid as a large number of discrete particles, then follows their path and finally transfers them to the Eulerian mesh realized by an interpolation scheme. Semi-Lagrangian method is fundamentally explicit without the CFL constraint so it is unconditionally stable, allowing large time steps. Additionally it does not need to regenerate computational domain at each time step as the Lagrangian methods do.

Among all these efforts, this paper reports a fast CFD solver for solving indoor airflow problems at real time based on the semiLagrangian method (Courant et al., 1952, Staniforth et al., 1991, Zuo et al., 2009, Jin et al., 2015). Previous researchers (Zuo et al., 2009, Jin et al., 2013, Jin et al., 2015) developed a Fast Fluid Dynamics (FFD) method based on the semi-Lagrangian method to simulate indoor airflow problems. For example, Jin et al. (2013) used this algorithm to simulate natural ventilation around and inside a single room, buoyancy driven inside a single room, and also airflow inside a complex room. In the previous works most effort was spent to apply semi-Lagrangian method and improve its accuracy. Based on our knowledge, most of the computing time of a FFD simulation is consumed when solving the diffusion terms in the momentum equations and Poisson equation. The accuracy problem associated with mass imbalance of the FFD method can also be caused by a poor linear solver. Thus in this paper, we try to improve the performance of the momentum and Poisson equation solvers by using a wellknown numerical method, multigrid solver, to increase the convergence rate. This paper reports the first effort of developing a fast and accurate CFD solver: the combination of semi-Lagrangian and multigrid methods with a new in-house 3D CFD code using C++programming language.

Here we solve convection and diffusion terms in the Navier-Stokes (N-S) equations separately. The convection term is solved by first-order semi-Lagrangian algorithm and the diffusion term is solved by an implicit multigrid (MG) solver (Wesseling, 1995, Briggs et al., 2000). Here we use V-cycle with three different levels of computational girds in the multigrid.

In the current project, our main concern was the speed of our solver and we tried to speed up the convergence rate. In the present study, we observed that the huge amount of our running time in each time step is consumed by solving the Poisson equation. This problem was not found and studied with details in the previous studies (Staniforth et al., 1991, Zuo et al., 2009, Jin et al., 2013, Jin et al., 2015).

METHODOLOGY

In this section, we explain the methodology for solving the N-S incompressible flow. Equations (1) and (2) shows the continuity and the momentum equations.

$$\frac{\partial U_i}{\partial x_i} = 0 \tag{1}$$

$$\frac{\partial U_i}{\partial t} = -U_j\frac{\partial U_i}{\partial x_j} + \frac{\mu}{\rho}\frac{\partial^2 U_i}{\partial x_j^2} - \frac{1}{\rho}\frac{\partial P}{\partial x_i} + f_i \tag{2}$$

where U_i, x_i, f_i are velocity, space, and source term vectors in i direction. t, x, ρ, μ are time, space, density, and dynamic viscosity of the fluid. The first term on the right hand side of Eq. (2) is the advection (or convection) term. The second term is the diffusion term followed by the pressure gradient term.

To solve Eqs. (1) and (2), we used the projection method based on the decomposition theorem, namely the Helmholtz-Hodge Decomposition. Chorin used this method to solve incompressible N-S equations (Chorin 1967). In his algorithm, we have three step: first, we calculate the intermediate velocity field by solving Eq. (3).

$$\frac{\partial U_i^*}{\partial t} = -U_j\frac{\partial U_i}{\partial x_j} + \frac{\mu}{\rho}\frac{\partial^2 U_i}{\partial x_j^2} + f_i \tag{3}$$

Then we apply the divergence free condition for an incompressible flow field, the continuity equation, to Eq. (3) and calculate a new pressure field:

$$\frac{\partial^2 P^{n+1}}{\partial x_j^2} = \frac{\rho}{\Delta t}\frac{\partial U_i^*}{\partial x_i}\ (Projection) \tag{4}$$

Eq. (4) is a Poisson equation and this step is called the projection step. The next final step is the correction step, where we calculate a new velocity field by using the new pressure field as shown in Eq. (5).

$$\frac{U_i^{n+1} - U_i^*}{\Delta t} = -\frac{1}{\rho}\frac{\partial P^{n+1}}{\partial x_i} \ (Correction) \tag{5}$$

In the current work, we use a first-order time splitting or fractional step method to solve Eq. (3) by the following three-step procedure.

$$\frac{\partial U_i}{\partial t} = \frac{\mu}{\rho}\frac{\partial^2 U_i}{\partial x_j^2} \ (Diffusion\ and\ source\ term) \tag{6}$$

$$\frac{\partial U_i}{\partial t} = -U_j\frac{\partial U_i}{\partial x_j} \ (Advection) \tag{7}$$

$$\frac{\partial U_i}{\partial t} = -\frac{1}{\rho}\frac{\partial P}{\partial x_i} \tag{8}$$

The procedure of solving incompressible N-S equation in the present work is thus:

Diffusion and source term $\Rightarrow$ Advection $\Rightarrow$ Projection $\Rightarrow$ Correction

The first step of solving the diffusion and source terms is approximately similar to the projection term, for which we use geometric multigrid method. For the advection step, we use a first-order backward semi-Lagrangian algorithm. In the following, we will explain the details of the multigrid solver and semiLagrangian used in the present work.

<u>MULTIGRID</u>

Multigrid method is a very powerful and useful method to iteratively solve differential equations. This method was developed to accelerate the convergence rate of conventional iterative methods, such as Jacobi and Gauss-Seidel methods. The procedure of this method is solving the equation on the coarse grids to damp the low frequency errors and increase the convergence rate.

In this paper we use V-cycle geometric multigrid with three levels: fine grid, average, and coarse girds as shown in Figure 1.

Multigrid method has three main steps (Briggs et al., 2000):

1. Smoothing: solving the main equation on the fine grid by using a few iterations of a conventional iterative method, such as Jacobi and Gauss-Seidel methods.

2. Restriction: solving the error equation on the coarse grid by transferring the data from fine to coarse grids.

3. Interpolation or prolongation: transferring the calculated errors to fine girds and modifying the data calculated from smoothing step.

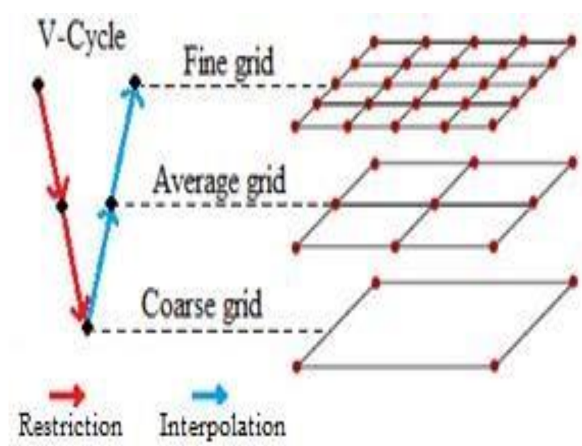

Figure 1. Interpolation and restriction in V-Cycle multigrid solver

For Eqs. (4) and (6), We can cast them in a general matrix form after linearization:

$$AU_i = b_i \tag{9}$$

where A is the coefficients' matrix. Eq. (9) is solved by two iterations of the Gauss-Seidel method on the fine grid to find V_i, the approximate of U_i. Then we calculate the residual R_i:

$$R_i = b_i - AV_i \tag{10}$$

We then write the error equation $e_i = U_i - V_i$:

$$Ae_i = R_i \tag{11}$$

The next step is to transfer the solution of the error equation from the fine grid to the average grid, so-called the restriction step, by ten Gauss-Seidel iterations on the fine grid. For transferring, we used a first-order linear interpolation scheme in all three mesh directions. Then the whole procedure, recalculating residual, reconstructing error equation, and retransferring the error equation from average grid to coarse grid, will be repeated on the coarse grid. Then we go to the final step, the interpolation step when we transfer the results of the error equation from the coarse grid to the average grid by the linear interpolation. Here we solve the error equation by two Gauss-Seidel iterations for smoothing followed by the linear interpolation to transfer the results of the error equation from the average grid to the fine grid. Now the new velocity values can be found from the correction:

$$U_i = e_i + V_i. \tag{12}$$

If the convergence criterion is satisfied, we go to the next time step. Otherwise the whole procedure is repeated.

SEMI-LAGRANGIAN

As mentioned before here we used the firstorder semi-Lagrangian algorithm to solve the advection term, Eq. (7), rewritten in a Lagrangian form:

$$\frac{DU_i}{Dt} = \frac{\partial U_i}{\partial t} + U_j \frac{\partial U_i}{\partial x_j} = 0 \tag{13}$$

Now we solve the Lagrangian form or material derivative with first-order accuracy:

$$\frac{DU_i}{Dt} = \frac{U^{n+1}_{i(x)} - U^n_{i(x - U^n_i \Delta t)}}{\Delta t} = 0 \tag{14}$$

Based on the Lagrangian derivative, if we consider the fluid inside the flow consists of a lot of small particles, we can calculate the particles' velocity by tracing back the particles' path.

$$U^{n+1}_{i(x)} = U^n_{i(x - U^n_i \Delta t)} = U^n_{i(x^{old})} \tag{15}$$

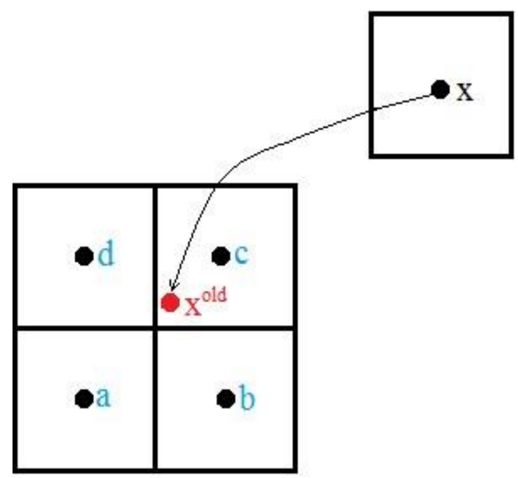

Figure 2. Sketch of computing new velocity field by semiLagrangian method.

Figure 2 shows that the calculation of the velocity at x^{old} needs an interpolation scheme. Here we use a linear interpolation scheme and calculate the velocity at x^{old} by using the values of the neighbor cells, a, b, c, and d in a 2-D setup as an example.

ERROR DEFFINITION

In the present work, two error definitions are used. Solving the diffusion and projection equations by multigrid solver, the error is the residual calculated by Eq. (10). Here the error for solving the diffusion terms is calculated by the velocity terms on the fine grid and for the projection equation is calculated by the pressure variable on the fine grid.

In this solver we also defined another error to show when the results achieve steady state conditions. Here the error is defined by the maximum difference between the value of the velocity field at the current time step and the previous time step. The convergence criterion for both errors is 1×10^{-7}.

SIMULATION

In this section first we will solve the wellknown 3D lid-driven cavity flow problem and compare our results with conventional finite volume methods. The accuracy and time step used in semi-Lagrangian and conventional finite volume method will be discussed including the discussion of the convergence rate of the MG solver added to the semiLagrangian solver. The computational domain of the cavity flow is a cube ($1 \times 1 \times 1$ m^3) with 64,000 structured nodes ($40 \times 40 \times 40$ for the three directions). The lid velocity at the top is set to 1 m/s. All other surfaces are fixed walls. Then, we will solve heat conduction problem with the same computational domain to investigate the efficiency of our multigrid solver.

RESULTS

Figure 3 shows the velocity vectors on the middle-section of the cube, x-y surface in the Cartesian coordinate. In the previous study (Jin et al. 2013), the researchers mentioned that the semi-Lagrangian methods has some problems to capture recirculation flows. From visual inspection, the current solver is shown to be able to capture the recirculation flow field well.

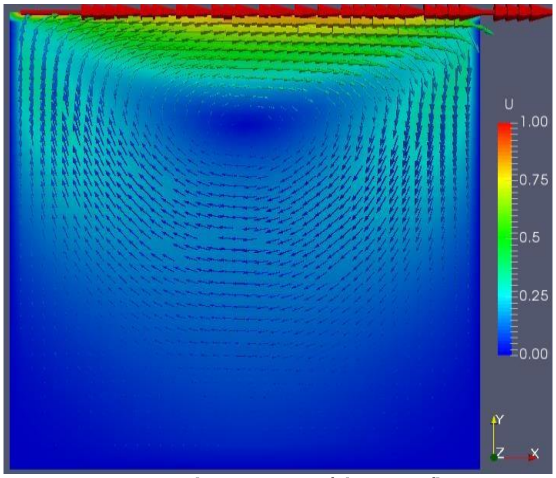

Figure 3. Velocity vectors of the cavity flow

In Figure 4, we compare our results with the conventional finite volume method (FVM) for different time steps, $dt = 0.001, 0.01, 0.1$ s. Note that with conventional FVM method, we cannot use $dt > 0.001$ s because of the stability problem as related to the CFL condition. But thanks to the semi-Lagrangian of the new solver, we can use larger time steps.

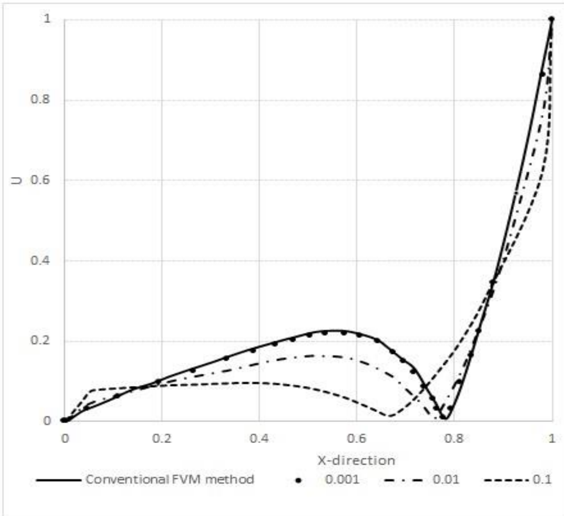

Figure 4. Comparison of velocity magnitude on the center line of the computational domain in the cavity flow.

Figure 4 also shows that a large time step decreases the simulation accuracy when using the finite volume method (FVM) method as a baseline for comparison. This problem has been reported by the previous studies. The accuracy can thus be improved by using small time steps or higher order methods, e.g. second-order semi-Lagrangian methods. We can also improve the accuracy by using high order interpolation schemes.

The advantage of using multigrid solver can be shown by comparing to simple solvers such as Gauss-Seidel. For the same convergence criterion of 1×10^{-7}, Table 1 shows that the MG solver is about 5 times faster than the G-S solver. To show the advantage of the MG solver, we modify the cavity case by only solving the diffusion term, the Laplace equation so the problem becomes a purely heat conduction problem.

$$\frac{\partial^2 T}{\partial x^2}+\frac{\partial^2 T}{\partial y^2}+\frac{\partial^2 T}{\partial z^2}=0.0 \tag{16}$$

Figure 5 shows the results of the heat conduction problem with the following boundary conditions, (temperatures on the different walls are defined as non-dimensional values):

$$T_{top\ face}=1.0, T_{others}=0.0$$

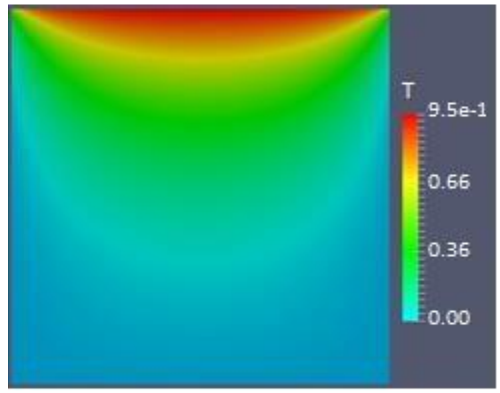

Figure 5. 3 dimensional heat conduction problem

Table 1. Comparison of execution time for simple GS solver and multigrid solver.

	Multigrid (V-cycle)	Simple (Guess-Seidel)
Cavity flow	26.09 [s]	129 [s]
Heat Conduction	1.68 [s]	11.98 [s]

For the heat conduction problem, the MG solver is about 7 times faster than the GS method.

The previous studies did not investigate and report the significance of each term in the conservation equations in terms of computing time and accuracy. In the following, we try to investigate which part of our equations is more time consuming. Figure 6 shows the convergence rate of solving the diffusion terms of the momentum equations in three different directions. For the criterion of 1×10^{7}, the convergence is satisfied after ten iterations in all directions. In comparison, Figure 7 shows that the number of iterations for the Poisson is about 350 for convergence. It means that most of the computing time, due to the numerical operations, is caused by solving the Poisson equation not the diffusion equation. Here the error is the residual calculated by Eq. (10). Therefore, the computational speed can be further improved by other advanced numerical solvers for the Poisson equation, such as adding successive over relaxation methods (Saad 2003).

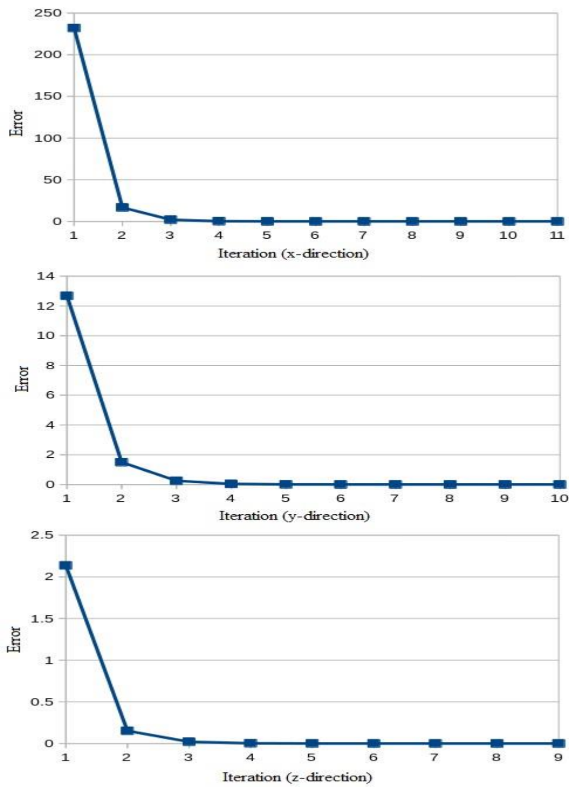

Figure 6. Error vs number of iterations when solving the diffusion term in x, y and z directions.

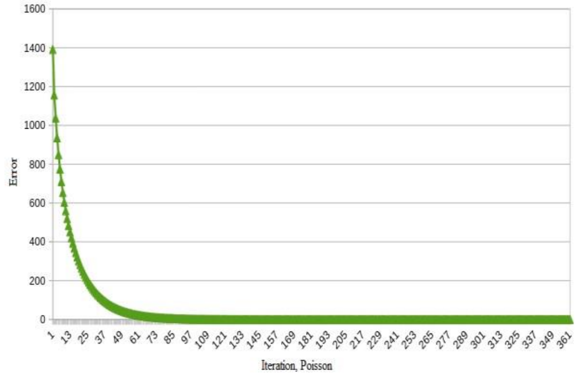

Figure 7. Error vs number of iterations of solving the Poisson equation.

CONCLUSION

In the applications of building engineering, we often encounter big computing domains during a CFD analysis, such as a building with many rooms and many stories. A conventional CFD solver is often not impractical to use in this case, because it is too computationally demanding. Meanwhile other methods such as zonal and multizone models are not accurate enough to capture all characteristic of the flows. Semi-Lagrangian method is a fast algorithm that can improve the speed of CFD simulations. Additionally multigrid method is one of the well-known fast numerical solvers for iteratively solving differential equations. In the current work we combined both of these methods and increased the convergence rate of the standard semi-Lagrangian method. The results show convergence rate is improved.

It is also found that using large time steps can affect the simulation accuracy negatively. For better accuracy, we have two options: small time step and the higher order algorithms. The first choice seems not preferred because small time steps slow down the solver so higher order algorithms should be the solution for the future work. From this study, we also found that the Poisson equation is the major road block for increasing the computational speed, which can be achieved by adding advanced solutions, such as successive over relaxations, to the multigrid solver. In the future, for the applications of using this solver for indoor airflow problems, it is also necessary to add turbulence models and probably using advanced hardware, such as graphical processing unit (GPU), to further accelerate the CFD solver.

REFERENCES

Briggs W. L. and McCormick S. F. (2000), *A multigrid tutorial*. Siam.

Chorin A. J. (1967), 'The numerical solution of the Navier-Stokes equations for an incompressible fluid', *Bulletin of the American Mathematical Society*, 73(6) 928-931.

Courant R., Issacson E., and Rees M. (1952), 'On the solution of nonlinear hyperbolic differential equations by finite differences', *Communications on Pure and Applied Mathematics*, 5(3) 243–255.

Eriksson J. and Wahlstrom A. (2002), 'Use of multizone air exchange simulation to evaluate a hybrid ventilation system', *Transactions-American Society of Heating Refrigerating and Airconditioning Engineers*, 108 811–817.

Jin M., Zuo W., and Chen Q. (2013), 'Simulating Natural Ventilation in and Around Buildings by Fast Fluid Dynamics', *Numerical Heat Transfer, Part A: Applications*, 64(4) 273-289.

Jin M. and Chen Q. (2015), 'Improvement of fast fluid dynamics with a conservative semi-Lagrangian scheme', *International Journal of Numerical Methods for Heat & Fluid Flow*, 25(1) 2-18.

Lakehal D. (2002), 'On the modelling of multiphase turbulent flows for environmental and hydrodynamic applications', *International Journal of Multiphase Flow*, 28 823–863.

Loth E. (2000), 'Numerical approaches for motion of dispersed particles, droplets, and bubbles', *Progress in Energy and Combustion Science*, 26 161–223.

Saad Y. (2003), *Iterative methods for sparse linear systems*, Siam, 2003.

Shirolkar J.S., Coimbra C.F.M. and McQuay M.Q. (1996), 'Fundamental aspects of modeling turbulent particle dispersion in dilute flows', *Progress in Energy and Combustion Science*, 22 363–399.

Staniforth A. and Côté J. (1991), 'Semi-Lagrangian integration schemes for atmospheric models-a review', Monthly Weather Review, 119(9) 22062223.

Wang L. and Chen Q. (2008), 'Evaluation of some assumptions used in multizone airflow network models', *Building and Environment*, 43(10) 1671–1677.

Wesseling P. (1995), *Introduction To Multigrid Methods*, (No. ICASE-95-11). Institute for Computer Applications in Science and Engineering HAMPTON VA.

Zuo W. and Chen Q. (2009), 'Real_time or faster _than _real _time simulation of airflow in buildings', *Indoor Air*, 19(1) 33-44.