


國立交通大學

應用數學系

碩士論文

內嵌界面問題之矩陣分解法



Matrix Factorization Method
for the Immersed Boundary problem

研究生：謝先皓

指導教授：賴明治 教授

中華民國九十七年六月

內嵌界面問題之矩陣分解法

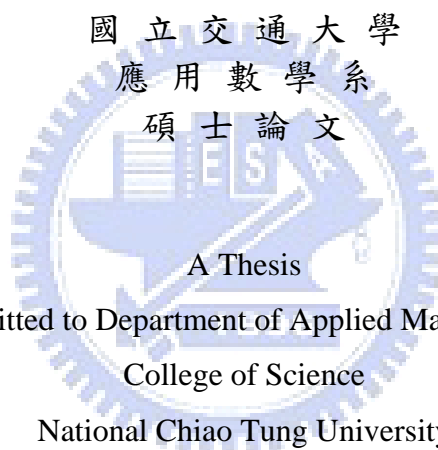
Matrix Factorization Method
for the Immersed Boundary Problem

研究生：謝先皓

Student : Hsen-Hao Hsieh

指導教授：賴明治

Advisor : Ming-Chih Lai



Submitted to Department of Applied Mathematics
College of Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Applied Mathematics

June 2008

Hsinchu, Taiwan, Republic of China

中華民國九十七年六月

內嵌介面問題之矩陣分解法

學生：謝先皓

指導教授：賴明治 教授

國立交通大學應用數學系（研究所）碩士班



摘 要

嵌入邊界法(immersed boundary method)是一種模擬不可壓縮流體的數學模型，他的特色在於解決有無質量的嵌入邊界的情況。而解決嵌入邊界法的問題，矩陣分解法(matrix factorization method)是一種利用類似分部步驟法(fractional step method)的方法，將嵌入邊界法分解成三個步驟，在前兩個步驟我們會面對解一個對稱正定的線性系統，在此我們可以利用共軛梯度法(conjugate gradient method)來解決這個問題。在這份論文之中，我們利用矩陣分解法來模擬流場通過各種嵌入之物體，包括流過靜止與可動的圓柱、兩個靜止的圓柱、以及機翼形狀的物體。

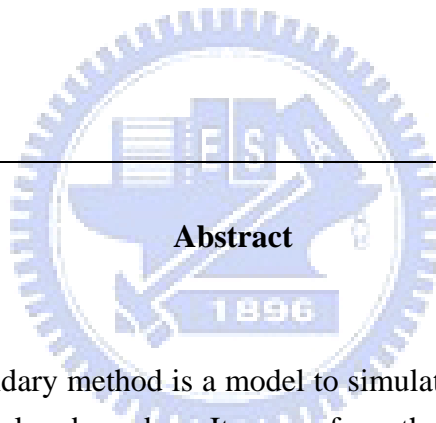
Matrix Factorization Method for the Immersed Boundary Problem

Student : Hsen-Hao Hsieh

Advisor : Ming-Chih Lai

Department of Applied Mathematics

National Chiao Tung University



Abstract

The immersed boundary method is a model to simulate a viscous incompressible fluid with immersed massless boundary. It comes from the Navier-Stokes equation of viscous incompressible fluid with the interaction term between the immersed boundary and the fluid. The matrix factorization method is a formulation of immersed boundary method, and the idea is the fractional step method for Navier-Stokes equation. The immersed boundary problem could be factorized to three steps, and the conjugate gradient method can be applied to solve the first and second step. In this paper, we use the matrix factorization method simulate the flow past stationary or movable immersed object, including the flow past a stationary and a moving cylinder, the flow past two stationary cylinders, and the flow past a winglike object.

誌 謝

本篇論文的完成，首先要感謝我的指導老師—賴明治教授。老師帶領我從大四下學期至今兩年半的時光，領導我走入了數值計算與數學建模這塊領域。每當我有不解與疑惑的時候，老師總是有耐心的指導我如何去面對問題，挑戰問題，並且給我正確的學習態度。我印象最深刻的，是當我遇見瓶頸停滯不前的時候，老師給我的一段話：「做學問就像打網球一樣，一個人開始學網球，絕不是一個人苦練一年才開始出去參加比賽，一定是邊練邊出去比賽。」使我了解到做學問，不是從學中做，而是從做中學，讓我對於研究的本質有更深一層的體會。而老師不僅是學業的傳授者，更是心靈成長的導師。老師給我們的一句銘言：「做人比做學問更重要。」讓我在鑽研知識的同時，更進一步的思考我們能夠為社會以及人群做些什麼。學生在此謹致最誠摯的謝意。在研究的過程中，也感謝曾孝捷學長與陳冠羽學長，給予我的指導與協助，讓我無論在接觸數值分析這塊領域的時候，或是學習新的程式語言，都給我相當大的幫助。也感謝曾昱豪學長，將自己做內嵌邊界法的經驗傳授給我，在此獻上最真誠的謝意。

在論文口試期間，承蒙王偉仲教授、洪子倫教授與吳金典助理教授費心審閱並提供許多寶貴意見，使本論文得以更加完備，學生永銘在心。

此外，我也要感謝曾鈺傑同學、郭又維同學、陳建明同學、蔡修齊同學以及胡偉帆同學。在這兩年的研究生涯裡，一直陪伴我走過了最艱難的日子，在我面對生命中挫折與打擊的時候，始終在我身旁鼓勵著我，在此謝謝他們。

最後，我要感謝我的母親與外婆，是他們悉心的照顧與栽培，才会有今日的我。願與他們，以及所有在我周圍關心我的人，一同分享此篇論文完成之喜悅與榮耀。

目 錄

中文提要	i
英文提要	ii
誌謝	iii
目錄	iv
一、	Introduction	1
二、	Fractional Step Method	2
2.1	Navier-Stokes equation	2
2.2	Staggered grid discretization	2
2.3	Fractional step method	6
2.4	Boundary conditions	10
三、	Matrix Factorization Method	17
3.1	Mathematical formulation	17
3.2	Discretization	18
3.3	Matrix factorization method	21
四、	Numerical Results	23
4.1	Error estimate of fractional step method	23
4.2	The flow past a cylinder	24
4.3	The flow past two cylinders	27
4.4	The flow past a wing	29
4.5	The flow past an oscillating cylinder	31
五、	Conclusion	33
Reference	34

1 Introduction

As we all know, the immersed boundary method has played an important role in the fluid–solid interaction problems. Peskin [7] first introduced the method to simulate the blood flow with an elastic membrane which can be regarded as the immersed boundary in the fluid. He discretizes the flow field with Eulerian grid, and discretizes the immersed boundary with Lagrangian points. The immersed boundary exerts a force on the fluid, and moves with a velocity, so the immersed object can move or be deformed.

The origin fractional step method is introduced by Chorin [1] to solve the incompressible Navier-Stokes equation. Perot [3] regards the fractional step method as a matrix factorization method, and the idea comes from the LU decomposition. The fractional step method can be written in three steps, which the first and second step can be solved by the conjugate gradient method, and the third step is a projection. Taira and Colonius [9] extend the fractional step method from Navier-Stokes equation to immersed boundary problem by observing the symmetric relationship between the discrete interpolation and regularization operators, and also factorize the scheme to three steps. The conjugate gradient method can be applied to the first and second step.

In Section 2, we introduce how to discretize the incompressible Navier-Stokes equations with the staggered marker-and-cell mesh, and review the fractional step method. The detailed matrix forms of each operator are introduced in Section 2.4 with the Dirichlet and the Neumann boundary conditions. In Section 3, we review the immersed boundary method first, and then consider the discretization of the discrete interpolation and regularization operators. The matrix factorization method is introduced in Section 3.3.

The numerical result is shown in Section 4. We can see the second-order temporal accuracy from the error estimate, and the simulation of the flow past a stationary or moving immersed object, including the flow past a stationary and an oscillating cylinder, the flow past two stationary cylinders, and the flow past a winglike object.

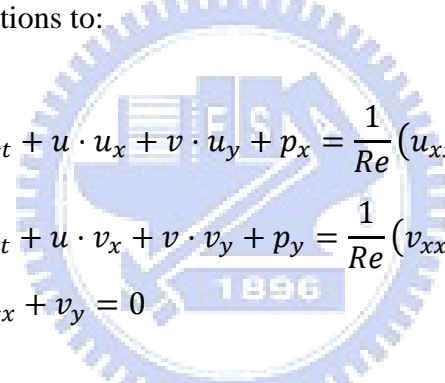
2 Fractional Step Method

2.1 Navier-Stokes equation

Consider the incompressible Navier-Stokes equations consisting of the momentum and continuity equations as:

$$\begin{aligned}\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} &= -\nabla p + \frac{1}{Re} \Delta \mathbf{u} \\ \nabla \cdot \mathbf{u} &= 0\end{aligned}$$

where $\mathbf{u}(t, \mathbf{x})$, $p(t, \mathbf{x})$, Re are the velocity vector, pressure, and Reynolds number, respectively. Usually, the initial condition $\mathbf{u}|_{t=0}$ and the boundary condition $\mathbf{u}|_{\partial\Omega}$ are known. If we consider that $\mathbf{u}(\mathbf{x}, t)$ is a two dimensional vector, that is, $\mathbf{u}(\mathbf{x}, t) = (u(\mathbf{x}, t), v(\mathbf{x}, t))$, then we can rewrite the incompressible Navier-Stokes equations to:


$$\begin{aligned}u_t + u \cdot u_x + v \cdot u_y + p_x &= \frac{1}{Re} (u_{xx} + u_{yy}) \\ v_t + u \cdot v_x + v \cdot v_y + p_y &= \frac{1}{Re} (v_{xx} + v_{yy}) \\ u_x + v_y &= 0\end{aligned}$$

with the initial conditions $u(x, y, 0)$, $v(x, y, 0)$, and the boundary conditions $u(x, y, t)|_{\partial\Omega}$, $v(x, y, t)|_{\partial\Omega}$.

2.2 Staggered grid discretization

To discretize the equations, refer to [1, 2, 3, 4], we can use the staggered marker-and-cell mesh that introduced by Harlow and Welsh [5] with implicit Crank-Nicolson for the viscous term and explicit second-order Adams-Bashforth for the convective term:

$$\begin{aligned}\frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t} + \frac{3}{2} \hat{N}(\mathbf{u}^n) - \frac{1}{2} \hat{N}(\mathbf{u}^{n-1}) &= -\hat{G}(\phi) + \frac{\hat{L}(\mathbf{u}^{n+1}) + \hat{L}(\mathbf{u}^n)}{2Re} \\ \hat{D}(\mathbf{u}^{n+1}) &= 0\end{aligned}$$

where $\mathbf{u}^{n+1} = (u^{n+1}, v^{n+1})^T$ and ϕ are the discrete velocity and pressure. \hat{N} , \hat{G} , \hat{L} and \hat{D} are the discrete convection, gradient, laplacian and divergence operator, respectively. Before introducing how to discretize these terms, we have to know how staggered marker-and-cell mesh can be applied here.

Suppose the two-dimensional domain of this equation is a rectangle. Let the domain $\Omega = [0, X] \times [0, Y]$, then divide $[0, X]$ and $[0, Y]$ into l parts and m parts, respectively. See figure 1, there are $l \times m$ cells. To apply the staggered grid discretization to Navier-Stokes equations, we define the discrete velocity of horizontal direction u at the center of each cell's left and right edges, and the discrete velocity of vertical direction v at the center of each cell's upper and lower edges. The discrete pressure ϕ is defined at the center of each cell.

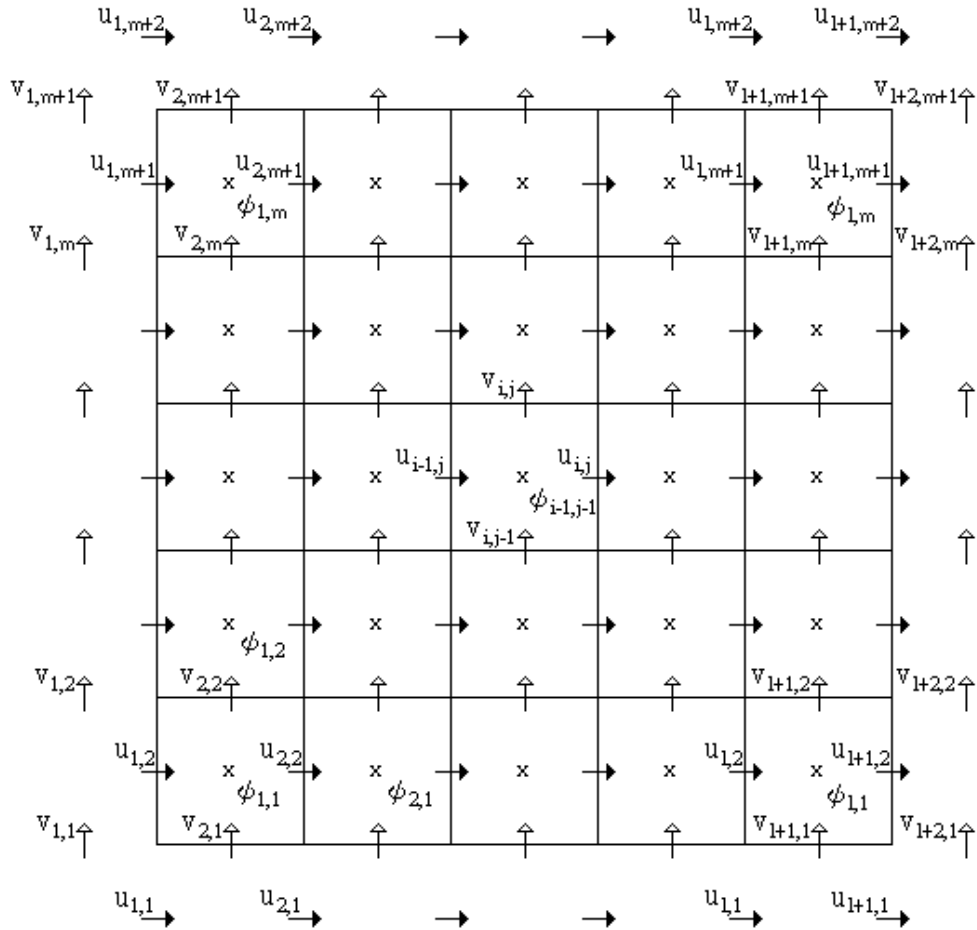


Fig. 1. The staggered marker-and-cell mesh

To make the scheme simple, suppose $\Delta x_i = \Delta y_j = h$ for all $1 \leq i \leq l$ and $1 \leq j \leq m$, then the discrete horizontal velocity $u_{i,j}^n$ can be regarded as the approximation of $u((i-1)h, (j-1.5)h, n\Delta t)$. Similarly, the discrete vertical velocity $v_{i,j}^n$ approximates to $v((i-1.5)h, (j-1)h, n\Delta t)$ and the discrete pressure $\phi_{i,j}^{n+0.5}$ approximates to $p((i-0.5)h, (j-0.5)h, (n+0.5)\Delta t)$.

Use staggered marker-and-cell mesh to discretize the convection, gradient, Laplacian and divergence terms. We can easily obtain the staggered grid discretization of the Navier-Stokes equations. Consider the Laplacian term, using the Taylor expansion, we have

$$u_{i-1,j} \approx u_{i,j} - (u_x)_{i,j} \Delta x_{i-1} + \frac{1}{2!} (u_{xx})_{i,j} \Delta x_{i-1}^2$$

$$u_{i+1,j} \approx u_{i,j} + (u_x)_{i,j} \Delta x_i + \frac{1}{2!} (u_{xx})_{i,j} \Delta x_i^2$$

Add these two formulas, then $(u_{xx})_{i,j}$ is:

$$(u_{xx})_{i,j} = \frac{2u_{i-1,j}}{\Delta x_{i-1}(\Delta x_{i-1} + \Delta x_i)} - \frac{2u_{i,j}}{\Delta x_{i-1}\Delta x_i} + \frac{2u_{i+1,j}}{\Delta x_i(\Delta x_{i-1} + \Delta x_i)}$$

Similarly, let $\Delta y_{j-0.5} = (\Delta y_{j-1} + \Delta y_j)/2$, then

$$(u_{yy})_{i,j} = \frac{2u_{i,j-1}}{\Delta y_{j-1.5}(\Delta y_{j-1.5} + \Delta y_{j-0.5})} - \frac{2u_{i,j}}{\Delta y_{j-1.5}\Delta y_{j-0.5}} + \frac{2u_{i,j+1}}{\Delta y_{j-0.5}(\Delta y_{j-1.5} + \Delta y_{j-0.5})}$$

The discrete Laplacian form can be written as:

$$\Delta u_{i,j} = (u_{xx})_{i,j} + (u_{yy})_{i,j}$$

Since we suppose that $\Delta x_i = \Delta y_j = h$, we obtain:

$$\Delta u_{i,j} = (u_{i,j-1} + u_{i-1,j} - 4u_{i,j} + u_{i+1,j} + u_{i,j+1})/h^2$$

Similarly,

$$\Delta v_{i,j} \approx (v_{i,j-1} + v_{i-1,j} - 4v_{i,j} + v_{i+1,j} + v_{i,j+1})/h^2$$

Although the discrete pressure ϕ is defined at the center of each cell, $\hat{G}(\phi)$ is defined at the same place that the discrete velocity u and v are defined. We use the same index i, j to represent the discrete $(p_x)_{i,j}$ and $(p_y)_{i,j}$. Let $\Delta x_{i-0.5} = (\Delta x_{i-1} + \Delta x_i)/2$. The form can be written as:

$$(p_x)_{i,j} \approx (\phi_{i,j-1} - \phi_{i-1,j-1})/\Delta x_{i-0.5}$$

$$(p_y)_{i,j} \approx (\phi_{i-1,j} - \phi_{i-1,j-1})/\Delta y_{j-0.5}$$

Consider the convection term. The discretization of the convection term is:

$$(u \cdot u_x + v \cdot u_y)_{i,j} \approx u_{i,j} \left(\frac{u_{i+1,j} - u_{i-1,j}}{\Delta x_{i-1} + \Delta x_i} \right) + v_{i+\frac{1}{2},j-\frac{1}{2}} \left(\frac{u_{i,j+1} - u_{i,j-1}}{\Delta y_{j-1.5} + \Delta y_{j-0.5}} \right)$$

where $v_{i+\frac{1}{2},j-\frac{1}{2}}$ is the mean of the values of neighbor v . That is,

$$v_{i+\frac{1}{2},j-\frac{1}{2}} = \frac{v_{i,j-1} + v_{i+1,j-1} + v_{i,j} + v_{i+1,j}}{4}$$

Similarly,

$$(u \cdot v_x + v \cdot v_y)_{i,j} \approx u_{i-\frac{1}{2},j+\frac{1}{2}} \left(\frac{v_{i+1,j} - v_{i-1,j}}{\Delta x_{i-1.5} + \Delta x_{i-0.5}} \right) + v_{i,j} \left(\frac{v_{i,j+1} - v_{i,j-1}}{\Delta y_{j-1} + \Delta y_j} \right)$$

where

$$u_{i-\frac{1}{2},j+\frac{1}{2}} = \frac{u_{i-1,j} + u_{i,j} + u_{i-1,j+1} + u_{i,j+1}}{4}$$

If $\Delta x_i = \Delta y_j = h$, then $\Delta x_{i-0.5} = \Delta y_{j-0.5} = h$

Consider the discrete divergence term. We can use divergence theorem here. Integrate the continuity equation on each cell.

$$0 = \int_{\text{cell}} \nabla \cdot \mathbf{u} dV = \sum_{\text{cell faces}} \int_{\text{face}_i} \mathbf{u} \cdot \mathbf{n} dS$$

where \mathbf{n} is the outward normal vector of cell face. Suppose the domain is two-dimensional, then dV can be rehash to $dx dy$, and dS can be rehash to dx or dy . We obtain the discrete continuity equation:

$$-u_{i,j+1}\Delta y_j + u_{i+1,j+1}\Delta y_j - v_{i+1,j}\Delta x_i + v_{i+1,j+1}\Delta x_i = 0$$

for all $1 \leq i \leq l$ and $1 \leq j \leq m$. Since we suppose that $\Delta x_i = \Delta y_j = h$, it can be written as:

$$(-u_{i-1,j} + u_{i,j} - v_{i,j-1} + v_{i,j})h = 0$$

2.3 Fractional step method

After discretize the incompressible Navier-Stokes equations, we have to solve the velocity \mathbf{u}^{n+1} and the pressure ϕ . Put the unknown \mathbf{u}^{n+1} and ϕ on the left side of the equal sign, and the other terms on the right side. The discrete formulations of incompressible Navier-Stokes equations are:

$$\begin{aligned} \left(\frac{1}{\Delta t} \mathbf{I} - \frac{1}{2Re} \hat{\mathbf{L}}\right) \mathbf{u}^{n+1} + \hat{\mathbf{G}}\phi &= \left(\frac{1}{\Delta t} \mathbf{I} + \frac{1}{2Re} \hat{\mathbf{L}}\right) \mathbf{u}^n - \frac{3}{2} \hat{\mathbf{N}}(\mathbf{u}^n) + \frac{1}{2} \hat{\mathbf{N}}(\mathbf{u}^{n-1}) + \widehat{bc}_1 \\ \hat{\mathbf{D}}\mathbf{u}^{n+1} &= 0 + bc_2 \end{aligned}$$

where \widehat{bc}_1 and bc_2 are the boundary conditions. The boundary conditions appear in equations because we regard the discrete operators as matrix form,

\mathbf{u}^{n+1} and ϕ both are vectors. Let $\hat{\mathbf{A}} = \frac{1}{\Delta t} \mathbf{I} - \frac{1}{2Re} \hat{\mathbf{L}}$, and \hat{r} be the right side of

the momentum equation. That is, $\hat{r} = \left(\frac{1}{\Delta t} \mathbf{I} + \frac{1}{2Re} \hat{\mathbf{L}}\right) \mathbf{u}^n - \frac{3}{2} \hat{\mathbf{N}}(\mathbf{u}^n) + \frac{1}{2} \hat{\mathbf{N}}(\mathbf{u}^{n-1})$.

The equations can be written as:

$$\begin{pmatrix} \widehat{\mathbf{A}} & \widehat{\mathbf{G}} \\ \widehat{\mathbf{D}} & 0 \end{pmatrix} \cdot \begin{pmatrix} \mathbf{u}^{n+1} \\ \phi \end{pmatrix} = \begin{pmatrix} \widehat{r} \\ 0 \end{pmatrix} + \begin{pmatrix} \widehat{bc}_1 \\ bc_2 \end{pmatrix}$$

The details of the matrix form of the discrete operators could be found in [3]. These matrices will be introduced detailed with the boundary conditions in Section 2.4, but brief here. First, we define $\mathbf{u}^{n+1} = (u^{n+1}, v^{n+1})^T$ and ϕ :

$$u^{n+1} = (u_{2,2}, \dots, u_{l,2}, u_{2,3}, \dots, u_{l,3}, \dots, u_{2,m+1}, \dots, u_{l,m+1})^T$$

$$v^{n+1} = (v_{2,2}, \dots, v_{l+1,2}, v_{2,3}, \dots, v_{l+1,3}, \dots, v_{2,m}, \dots, v_{l+1,m})^T$$

$$\phi = (\phi_{1,1}, \dots, \phi_{l,1}, \phi_{1,2}, \dots, \phi_{l,2}, \dots, \phi_{1,m}, \dots, \phi_{l,m})^T$$

Moreover, we define two matrices \mathbf{R} and $\widehat{\mathbf{M}}$:

$$\mathbf{R} = \begin{pmatrix} \Delta y_1 \mathbf{I}_{l-1} & & & 0 \\ & \ddots & & \\ & & \Delta y_m \mathbf{I}_{l-1} & \\ 0 & & & \ddots \\ & & & & \mathbf{R}\mathbf{x}_{m-1} \end{pmatrix}, \mathbf{R}\mathbf{x}_j = \begin{pmatrix} \Delta x_1 & & 0 \\ & \ddots & \\ 0 & & \Delta x_l \end{pmatrix}$$

$$\widehat{\mathbf{M}} = \begin{pmatrix} \mathbf{M}\mathbf{x}_1 & & & 0 \\ & \ddots & & \\ & & \mathbf{M}\mathbf{x}_m & \\ 0 & & & \Delta y_{1.5} \mathbf{I}_l \\ & & & & \ddots \\ & & & & & \Delta y_{m-0.5} \mathbf{I}_l \end{pmatrix}, \mathbf{M}\mathbf{x}_i = \begin{pmatrix} \Delta x_{1.5} & & 0 \\ & \ddots & \\ 0 & & \Delta x_{l-0.5} \end{pmatrix}$$

If $\Delta x_i = \Delta y_j = h$, then $\mathbf{R} = \widehat{\mathbf{M}} = h\mathbf{I}_{(l-1)m+l(m-1)}$, where \mathbf{I} is the identity matrix. We take the discrete momentum equation product $\widehat{\mathbf{M}}$, then there is a new matrix form:

$$\begin{cases} \widehat{\mathbf{M}}\widehat{\mathbf{A}}\mathbf{u}^{n+1} + \widehat{\mathbf{M}}\widehat{\mathbf{G}}\phi = \widehat{\mathbf{M}}\widehat{r} + \widehat{\mathbf{M}}\widehat{bc}_1 \\ \widehat{\mathbf{D}}\mathbf{u}^{n+1} = bc_2 \end{cases}$$

Define $q^{n+1} = \mathbf{R}\mathbf{u}^{n+1}$, then

$$\begin{cases} \widehat{\mathbf{M}}\widehat{\mathbf{A}}\mathbf{R}^{-1}q^{n+1} + \widehat{\mathbf{M}}\mathbf{G}\phi = \widehat{\mathbf{M}}\hat{r} + \widehat{\mathbf{M}}\widehat{bc}_1 \\ \widehat{\mathbf{D}}\mathbf{R}^{-1}q^{n+1} = bc_2 \end{cases}$$

To make the matrix form simpler, let $\mathbf{A} = \widehat{\mathbf{M}}\widehat{\mathbf{A}}\mathbf{R}^{-1}$, $\mathbf{G} = \widehat{\mathbf{M}}\mathbf{G}$, $r = \widehat{\mathbf{M}}\hat{r}$, $bc_1 = \widehat{\mathbf{M}}\widehat{bc}_1$, and $\mathbf{D} = \widehat{\mathbf{D}}\mathbf{R}^{-1}$, the matrix form can be written as:

$$\begin{pmatrix} \mathbf{A} & \mathbf{G} \\ \mathbf{D} & \mathbf{0} \end{pmatrix} \cdot \begin{pmatrix} q^{n+1} \\ \phi \end{pmatrix} = \begin{pmatrix} r \\ 0 \end{pmatrix} + \begin{pmatrix} bc_1 \\ bc_2 \end{pmatrix}$$

There is an interesting fact that $\mathbf{D} = -\mathbf{G}^T$, that is the reason why we rehash the origin matrix form to the new matrix form. The origin fractional step method is introduced by Chorin in [1], Now, we use the fractional step method which is introduced by Perot in [3]. The idea of the fractional step method comes from the LU decomposition:

$$\begin{pmatrix} \mathbf{A} & \mathbf{G} \\ \mathbf{D} & \mathbf{0} \end{pmatrix} \cdot \begin{pmatrix} q^{n+1} \\ \phi \end{pmatrix} = \begin{pmatrix} \mathbf{A} & \mathbf{0} \\ -\mathbf{G}^T & \mathbf{G}^T\mathbf{A}^{-1}\mathbf{G} \end{pmatrix} \begin{pmatrix} \mathbf{I} & \mathbf{A}^{-1}\mathbf{G} \\ \mathbf{0} & \mathbf{I} \end{pmatrix} \cdot \begin{pmatrix} q^{n+1} \\ \phi \end{pmatrix} = \begin{pmatrix} r \\ 0 \end{pmatrix} + \begin{pmatrix} bc_1 \\ bc_2 \end{pmatrix}$$

To handle the inverse matrix of \mathbf{A} , there is a approximation introduce by Témam [16], let $\mathbf{M} = \widehat{\mathbf{M}}\mathbf{R}^{-1}$, $\mathbf{L} = (\widehat{\mathbf{M}}\widehat{\mathbf{L}}\mathbf{R}^{-1})/Re$, the matrix \mathbf{A} can be written as :

$$\mathbf{A} = \widehat{\mathbf{M}}\widehat{\mathbf{A}}\mathbf{R}^{-1} = \widehat{\mathbf{M}} \left(\frac{1}{\Delta t} \mathbf{I} - \frac{1}{2Re} \widehat{\mathbf{L}} \right) \mathbf{R}^{-1} = \frac{1}{\Delta t} \mathbf{M} - \frac{1}{2} \mathbf{L} = \frac{1}{\Delta t} \mathbf{M} \left(\mathbf{I} - \frac{\Delta t}{2} \mathbf{M}^{-1} \mathbf{L} \right)$$

The inverse matrix of \mathbf{A} can be approximated by :

$$\left(\mathbf{I} - \frac{\Delta t}{2} \mathbf{M}^{-1} \mathbf{L} \right)^{-1} \approx \mathbf{I} + \frac{\Delta t}{2} \mathbf{M}^{-1} \mathbf{L} + \frac{\Delta t^2}{2^2} (\mathbf{M}^{-1} \mathbf{L})^2 + \dots + \frac{\Delta t^N}{2^N} (\mathbf{M}^{-1} \mathbf{L})^N$$

$$\mathbf{A}^{-1} \approx \sum_{k=1}^N \frac{\Delta t^k}{2^{k-1}} (\mathbf{M}^{-1} \mathbf{L})^{k-1} \mathbf{M}^{-1} + \frac{\Delta t^{N+1}}{2^N} (\mathbf{M}^{-1} \mathbf{L})^N \mathbf{M}^{-1}$$

$\mathbf{B}^N = \sum_{k=1}^N \frac{\Delta t^k}{2^{k-1}} (\mathbf{M}^{-1} \mathbf{L})^{k-1} \mathbf{M}^{-1}$ is the Nth order Talyor expansion of \mathbf{A}^{-1} ,

let $\mathbf{A}^{-1} = \mathbf{B}^N + \frac{\Delta t^{N+1}}{2^N} (\mathbf{M}^{-1} \mathbf{L})^N \mathbf{M}^{-1}$, then

$$\begin{pmatrix} \mathbf{A} & \mathbf{0} \\ -\mathbf{G}^T & \mathbf{G}^T \mathbf{B}^N \mathbf{G} \end{pmatrix} \begin{pmatrix} \mathbf{I} & \mathbf{B}^N \mathbf{G} \\ \mathbf{0} & \mathbf{I} \end{pmatrix} \cdot \begin{pmatrix} q^{n+1} \\ \phi \end{pmatrix} = \begin{pmatrix} r \\ 0 \end{pmatrix} + \begin{pmatrix} bc_1 \\ bc_2 \end{pmatrix} + \begin{pmatrix} -\Delta t^N (\mathbf{L} \mathbf{M}^{-1})^N \mathbf{G} \phi / 2^N \\ 0 \end{pmatrix}$$

The last term is the truncation error from \mathbf{B}^N . If Δt is small and N is large enough, the last term can be ignored. Here we want to solve q^{n+1} and ϕ , so we define $q^* = q^{n+1} + \mathbf{B}^N \mathbf{G} \phi$, then

$$\begin{pmatrix} \mathbf{I} & \mathbf{B}^N \mathbf{G} \\ \mathbf{0} & \mathbf{I} \end{pmatrix} \cdot \begin{pmatrix} q^{n+1} \\ \phi \end{pmatrix} = \begin{pmatrix} q^{n+1} + \mathbf{B}^N \mathbf{G} \phi \\ \phi \end{pmatrix} = \begin{pmatrix} q^* \\ \phi \end{pmatrix}$$

$$\begin{pmatrix} \mathbf{A} & \mathbf{0} \\ -\mathbf{G}^T & \mathbf{G}^T \mathbf{B}^N \mathbf{G} \end{pmatrix} \begin{pmatrix} q^* \\ \phi \end{pmatrix} = \begin{pmatrix} \mathbf{A} q^* \\ -\mathbf{G}^T q^* + \mathbf{G}^T \mathbf{B}^N \mathbf{G} \phi \end{pmatrix} = \begin{pmatrix} r \\ 0 \end{pmatrix} + \begin{pmatrix} bc_1 \\ bc_2 \end{pmatrix}$$

The fractional step method can be written in three steps:

$$\begin{aligned} \mathbf{A} q^* &= r + bc_1 && \text{(Solve } q^*) \\ \mathbf{G}^T \mathbf{B}^N \mathbf{G} \phi &= \mathbf{G}^T q^* + bc_2 && \text{(Solve } \phi) \\ q^{n+1} &= q^* - \mathbf{B}^N \mathbf{G} \phi && \text{(Get } q^{n+1}) \end{aligned}$$

The matrix $\mathbf{A} = \widehat{\mathbf{M}} \left(\frac{1}{\Delta t} \mathbf{I} - \frac{1}{2Re} \widehat{\mathbf{L}} \right) \mathbf{R}^{-1}$ is symmetric positive definite because the discrete laplacian operator $\widehat{\mathbf{L}}$ is symmetric and negative definite. If Δt is small and N is large enough, \mathbf{B}^N is symmetric and positive definite, too. It is easy to check that $\mathbf{G}^T \mathbf{B}^N \mathbf{G}$ is symmetric and positive definite.

$$\begin{aligned} (\mathbf{G}^T \mathbf{B}^N \mathbf{G})^T &= \mathbf{G}^T (\mathbf{B}^N)^T \mathbf{G} = \mathbf{G}^T \mathbf{B}^N \mathbf{G} \\ \mathbf{x}^T \mathbf{G}^T \mathbf{B}^N \mathbf{G} \mathbf{x} &= (\mathbf{G} \mathbf{x})^T \mathbf{B}^N (\mathbf{G} \mathbf{x}) > 0 \end{aligned}$$

Since \mathbf{A} and $\mathbf{G}^T \mathbf{B}^N \mathbf{G}$ are symmetric and positive definite, conjugate gradient method can be applied to solve q^* and ϕ . The fractional step method has second-order temporal error form implicit Crank-Nicolson and explicit second-order Adams-Bashforth, and Nth order error from \mathbf{B}^N , which is the approximation of \mathbf{A}^{-1} . The numerical results of second-order accurate approximation of velocity could be found in chapter 4. The discrete pressure ϕ is first-order approximation of $p^{n+1/2}$ [3]. The second-order accurate approximation of pressure could be found in [6].

2.4 Boundary conditions

First, we discuss the case of Dirichlet boundary conditions where the domain $\Omega = [0, X] \times [0, Y]$. That is, the boundary conditions $u(0, y, t)$, $u(X, y, t)$, $u(x, 0, t)$, $u(x, Y, t)$, $v(0, y, t)$, $v(X, y, t)$, $v(x, 0, t)$ and $v(x, Y, t)$ are known.

Suppose $\Delta x_i = \Delta y_j = h$, we know the values of the boundary velocity $u_{1,j}^{n+1}$, $u_{l+1,j}^{n+1}$, $v_{i,1}^{n+1}$ and $v_{i,m+1}^{n+1}$, they are

$$\begin{aligned} u_{1,j}^{n+1} &= u(0, (j - 1.5)h, (n + 1)\Delta t) \\ u_{l+1,j}^{n+1} &= u(X, (j - 1.5)h, (n + 1)\Delta t) \\ v_{i,1}^{n+1} &= v((i - 1.5)h, 0, (n + 1)\Delta t) \\ v_{i,m+1}^{n+1} &= v((i - 1.5)h, Y, (n + 1)\Delta t) \end{aligned}$$

for $2 \leq i \leq l + 1$ and $2 \leq j \leq m + 1$.

But the other discrete boundary conditions $u_{i,1}^{n+1}$, $u_{i,m+2}^{n+1}$, $v_{1,j}^{n+1}$ and $v_{l+2,j}^{n+1}$ are not defined at the boundary. We approximate these boundary conditions as follows:

$$\begin{aligned} u_{i,1}^{n+1} &\approx 2u((i - 1)h, 0, (n + 1)\Delta t) - u_{i,2}^n \\ u_{i,m+2}^{n+1} &\approx 2u((i - 1)h, Y, (n + 1)\Delta t) - u_{i,m+1}^n \\ v_{1,j}^{n+1} &\approx 2v(0, (j - 1)h, (n + 1)\Delta t) - v_{2,j}^n \\ v_{l+2,j}^{n+1} &\approx 2v(X, (j - 1)h, (n + 1)\Delta t) - v_{l+1,j}^n \end{aligned}$$

for $1 \leq i \leq l + 1$ and $1 \leq j \leq m + 1$.

After we know the discrete boundary conditions, the discrete operators can be written as matrices. Consider the laplacian term $\hat{\mathbf{L}}$, it contains the laplacian operators to the horizontal velocity and vertical velocity, called \mathbf{L}_1 and \mathbf{L}_2 .

$$\hat{\mathbf{L}} = \begin{pmatrix} \mathbf{L}_1 & 0 \\ 0 & \mathbf{L}_2 \end{pmatrix}$$

The discrete $\Delta u_{i,j}$ and $\Delta v_{i,j}$ have been introduced already. Since we suppose $\Delta x_i = \Delta y_j = h$, \mathbf{L}_1 and \mathbf{L}_2 can be written as :

$$\mathbf{L}_1 = \begin{pmatrix} \mathbf{L}_{11} & \mathbf{I}_{l-1} & & 0 \\ \mathbf{I}_{l-1} & \mathbf{L}_{12} & \ddots & \\ 0 & \ddots & \ddots & \mathbf{I}_{l-1} \\ & & \mathbf{I}_{l-1} & \mathbf{L}_{1m} \end{pmatrix}, \mathbf{L}_{1j} = \begin{pmatrix} -4 & 1 & & 0 \\ 1 & -4 & \ddots & \\ 0 & & \ddots & 1 \\ & & 1 & -4 \end{pmatrix}_{(l-1) \times (l-1)}$$

$$\mathbf{L}_2 = \begin{pmatrix} \mathbf{L}_{21} & \mathbf{I}_l & & 0 \\ \mathbf{I}_l & \mathbf{L}_{22} & \ddots & \\ 0 & \ddots & \ddots & \mathbf{I}_l \\ & & \mathbf{I}_l & \mathbf{L}_{2m-1} \end{pmatrix}, \mathbf{L}_{2j} = \begin{pmatrix} -4 & 1 & & 0 \\ 1 & -4 & \ddots & \\ 0 & & \ddots & 1 \\ & & 1 & -4 \end{pmatrix}_{l \times l}$$

Obviously, $\hat{\mathbf{L}}$ is symmetric and negative definite.

The boundary condition term \widehat{bc}_1 comes from $\hat{\mathbf{L}}$. \widehat{bc}_1 can be written as $\widehat{bc}_1 = (bcu^{n+1} + bcu^n, bcv^{n+1} + bcv^n)^T / 2Reh^2$, where bcu^n and bcv^n are

$$bcu^n = (u_{1,2}^n, 0, \dots, 0, u_{l+1,2}^n, u_{1,3}^n, 0, \dots, 0, u_{l+1,3}^n, \dots, u_{1,m+1}^n, 0, \dots, 0, u_{l+1,m+1}^n)^T$$

$$+ (u_{2,1}^n, u_{3,1}^n, \dots, u_{l+1,1}^n, 0, \dots, 0, u_{2,m+2}^n, u_{3,m+2}^n, \dots, u_{l+1,m+2}^n)^T$$

$$bcv^n = (v_{1,2}^n, 0, \dots, 0, v_{l+2,2}^n, v_{1,3}^n, 0, \dots, 0, v_{l+2,3}^n, \dots, v_{1,m}^n, 0, \dots, 0, v_{l+2,m}^n)^T$$

$$+ (v_{2,1}^n, u_{3,1}^n, \dots, v_{l+1,1}^n, 0, \dots, 0, v_{2,m+1}^n, u_{3,m+1}^n, \dots, v_{l+1,m+1}^n)^T$$

To introduce $\widehat{\mathbf{G}}$ and $\widehat{\mathbf{D}}$, refer to [19], let's see a simple example. Consider the two cells case. There are only four horizontal velocity terms and three vertical velocity terms, called $u_1, u_2, u_3, v_1, v_2, v_3, v_4$. See fig. 2.

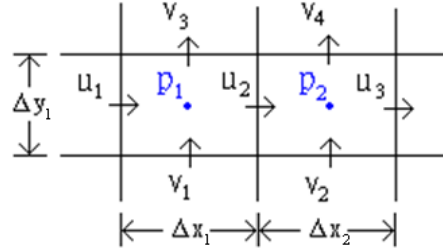


Fig. 2. The two cells case

Consider the discrete divergence term, we know that

$$-u_1 \Delta y_1 + u_2 \Delta y_1 - v_1 \Delta x_1 + v_3 \Delta x_1 = 0$$

$$-u_2 \Delta y_1 + u_3 \Delta y_1 - v_2 \Delta x_2 + v_4 \Delta x_2 = 0$$

If we rewrite it to the matrix form, then

$$\begin{pmatrix} -1 & 1 & 0 & -1 & 0 & 1 & 0 \\ 0 & -1 & 1 & 0 & -1 & 0 & 1 \end{pmatrix} \begin{pmatrix} \Delta y_1 & & & & & & \\ & \Delta y_1 & & & & & \\ & & \Delta y_1 & & & & \\ & & & \Delta x_1 & & & \\ & & & & \Delta x_2 & & \\ & 0 & & & & \Delta x_1 & \\ & & & & & & \Delta x_2 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{pmatrix} = 0$$

Notice that the center matrix is \mathbf{R} . Let the left matrix be \mathbf{D} , so we get the simple factorization $\widehat{\mathbf{D}} = \mathbf{DR}$.

Similarly, we also can get the matrix form of $\widehat{\mathbf{G}}$ and factorize $\widehat{\mathbf{G}}$. In the two cells case, $\widehat{\mathbf{G}}\phi$ can be written as:

$$\widehat{\mathbf{G}}\phi = \begin{pmatrix} \Delta x_{0.5} & & & & & & \\ & \Delta x_{1.5} & & & & & \\ & & \Delta x_{2.5} & & & & \\ & & & \Delta y_{0.5} & & & \\ & & & & \Delta y_{0.5} & & \\ & 0 & & & & \Delta y_{1.5} & \\ & & & & & & \Delta y_{1.5} \end{pmatrix}^{-1} \begin{pmatrix} 1 & 0 \\ -1 & 1 \\ 0 & -1 \\ 1 & 0 \\ 0 & 1 \\ -1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} p_1 \\ p_2 \end{pmatrix}$$

Let the left matrix be $\widehat{\mathbf{M}}^{-1}$ and the center matrix be \mathbf{G} . We can see that $\widehat{\mathbf{G}} = \widehat{\mathbf{M}}^{-1}\mathbf{G}$ and $\mathbf{D} = -\mathbf{G}^T$.

In general, the matrix form of \mathbf{G} is

$$\mathbf{G} = \begin{pmatrix} \mathbf{G}_1 \\ \mathbf{G}_2 \end{pmatrix}$$

$$\mathbf{G}_1 = \begin{pmatrix} \mathbf{G}_{11} & & 0 \\ & \ddots & \\ 0 & & \mathbf{G}_{1m} \end{pmatrix}, \mathbf{G}_{1j} = \begin{pmatrix} -1 & 1 & & 0 \\ & \ddots & \ddots & \\ 0 & & -1 & 1 \end{pmatrix}_{(l-1) \times l}$$

$$\mathbf{G}_2 = \begin{pmatrix} -\mathbf{G}_{21} & \mathbf{G}_{21} & & 0 \\ & \ddots & \ddots & \\ 0 & & -\mathbf{G}_{2m-1} & \mathbf{G}_{2m-1} \end{pmatrix}, \mathbf{G}_{2j} = \mathbf{I}_l$$

The relation that $\mathbf{D} = -\mathbf{G}^T$ still exists in general situation.

The boundary condition term bc_2 comes from $\widehat{\mathbf{D}}$. bc_2 can be written as:

$$bc_2 = (u_{1,2}^n, 0, \dots, 0, -u_{l+1,2}^n, u_{1,3}^n, 0, \dots, 0, -u_{l+1,3}^n, \dots \dots, u_{1,m+1}^n, 0, \dots, 0, -u_{l+1,m+1}^n)^T + (v_{2,1}^n, u_{3,1}^n, \dots, u_{l+1,1}^n, 0, \dots \dots, 0, v_{2,m+1}^n, v_{3,m+1}^n, \dots, v_{l+1,m+1}^n)^T$$

If there are not Dirichlet boundary conditions at all. That is, some boundary conditions maybe are Neumann boundary conditions. Let the boundary $\partial\Omega = \partial\Omega_1 \cup \partial\Omega_2 = \partial\Omega_3 \cup \partial\Omega_4$, and the boundary conditions $u(x, y, t)|_{\partial\Omega_1}$, $v(x, y, t)|_{\partial\Omega_3}$, $\partial_{\mathbf{n}}u(x, y, t)|_{\partial\Omega_2}$, $\partial_{\mathbf{n}}v(x, y, t)|_{\partial\Omega_4}$ are known, where \mathbf{n} is the outer normal vector of the boundary.

For example, we suppose that $\partial\Omega_1 = \{(x, y) \in \Omega | x = 0\}$, $\partial\Omega_2 = \partial\Omega \setminus \partial\Omega_1$, $\partial\Omega_3 = \{(x, y) \in \Omega | x = X\}$, and $\partial\Omega_4 = \partial\Omega \setminus \partial\Omega_3$. In figure 3, it shows the relationship between the boundary condition type and the place. The boundary conditions that we know are $u(0, y, t)$, $u_x(X, y, t)$, $u_y(x, 0, t)$, $u_y(x, Y, t)$, $v(0, y, t)$, $v_x(X, y, t)$, $v(x, 0, t)$ and $v(x, Y, t)$.

But the thing we really care about is the discrete boundary condition. We have known how to handle the discrete boundary condition when the boundary condition is Dirichlet. Now we discuss the discrete boundary condition when the boundary condition is Neumann.

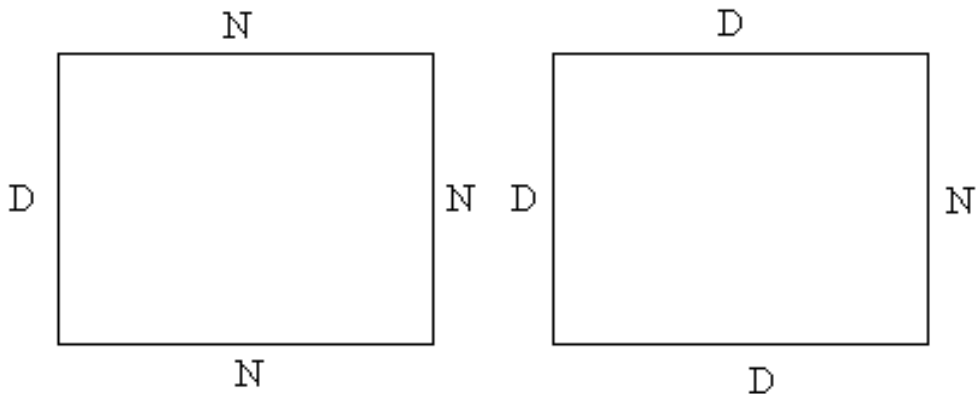


Fig. 3. The left figure represents the boundary condition type of u . The right figure represents the boundary condition type of v .

We can use the central difference to approximate $u_y(x, 0, t)$.

$$u_y((i-1)h, 0, (n+1)\Delta t) \approx \frac{u_{i,2}^{n+1} - u_{i,1}^{n+1}}{\Delta y_{0.5}}$$

The discrete boundary condition $u_{i,1}^{n+1}$ can be approximated by

$$u_{i,1}^{n+1} = u_{i,2}^{n+1} - \Delta y_{0.5} \cdot u_y((i-1)h, 0, (n+1)\Delta t)$$

But we do not know $u_{i,2}^{n+1}$, so this part can be replaced by $u_{i,2}^n$, or use the velocity at the time step n and $n-1$ to approximate $u_{i,2}^{n+1}$. That is,

$$u_{i,2}^{n+1} \approx 2u_{i,2}^n - u_{i,2}^{n-1}$$

This approximation also can be applied when the boundary condition is Dirichlet. So the discrete boundary condition $u_{i,1}^{n+1}$ can be approximated by

$$u_{i,1}^{n+1} \approx (2u_{i,2}^n - u_{i,2}^{n-1}) - \Delta y_{0.5} \cdot u_y((i-1)h, 0, (n+1)\Delta t)$$

Similarly, the other discrete boundary conditions can be written as

$$\begin{aligned} u_{i,m+2}^{n+1} &\approx (2u_{i,m+1}^n - u_{i,m+1}^{n-1}) + \Delta y_{m+0.5} \cdot u_y((i-1)h, Y, (n+1)\Delta t) \\ v_{l+2,j}^{n+1} &\approx (2v_{l+1,j}^n - v_{l+1,j}^{n-1}) + \Delta x_{l+0.5} \cdot v_x(X, (j-1)h, (n+1)\Delta t) \\ u_{l+1,j}^{n+1} &\approx (2u_{l,j}^n - u_{l,j}^{n-1}) + \Delta x_l \cdot u_x(X, (j-1.5)h, (n+1)\Delta t) \end{aligned}$$

Notice that we are not using the central difference to approximate $u_{l+1,j}^{n+1}$. Because these boundary conditions are approximated by \mathbf{u}^n and \mathbf{u}^{n-1} . After solving \mathbf{u}^{n+1} , we need to update the discrete boundary conditions. At the next time step, they will be used to compute $N(\mathbf{u})$.

Although we use the Neumann boundary conditions to approximate the boundary velocity, we would rather use the Neumann boundary conditions than use the approximated boundary velocity.

Consider the Laplacian operator $\hat{\mathbf{L}}$. For example, we would rather use $u_y(0.5h, 0, (n+1)\Delta t)$ than $(u_{2,2}^{n+1} - u_{2,1}^{n+1})/\Delta y_1$ when we compute the approximation of $\Delta u_{2,2}$. Suppose $\Delta x_i = \Delta y_j = h$, the discretization of $\Delta u_{2,2}$ is

$$\begin{aligned}\Delta u_{2,2} &= \frac{u_{3,2} - 2u_{2,2} + u_{1,2}}{h^2} + \frac{\frac{u_{2,3} - u_{2,2}}{h} - u_y(0.5h, 0, (n+1)\Delta t)}{h} \\ &= \frac{u_{1,2} - 3u_{2,2} + u_{3,2} + u_{2,3}}{h^2} - \frac{u_y(0.5h, 0, (n+1)\Delta t)}{h}\end{aligned}$$

Follow this idea, the matrix $\hat{\mathbf{L}}$ and the boundary condition \widehat{bc}_1 also have some change. In this case, the matrix $\hat{\mathbf{L}}$ changes to

$$\hat{\mathbf{L}} = \begin{pmatrix} \mathbf{L}_1 & 0 \\ 0 & \mathbf{L}_2 \end{pmatrix}$$

$$\mathbf{L}_1 = \begin{pmatrix} \mathbf{L}_{11} & \mathbf{I}_{l-1} & & 0 \\ \mathbf{I}_{l-1} & \mathbf{L}_{12} & \ddots & \\ & \ddots & \ddots & \mathbf{I}_{l-1} \\ 0 & & \mathbf{I}_{l-1} & \mathbf{L}_{1m} \end{pmatrix}, \mathbf{L}_2 = \begin{pmatrix} \mathbf{L}_{21} & \mathbf{I}_l & & 0 \\ \mathbf{I}_l & \mathbf{L}_{22} & \ddots & \\ & \ddots & \ddots & \mathbf{I}_l \\ 0 & & \mathbf{I}_l & \mathbf{L}_{2m-1} \end{pmatrix}$$

where \mathbf{L}_{1j} and \mathbf{L}_{2j} are

$$\mathbf{L}_{1j} = \begin{pmatrix} -3 & 1 & & 0 \\ 1 & -3 & \ddots & \\ & \ddots & \ddots & 1 \\ & & 1 & -3 & 1 \\ 0 & & & 1 & -2 \end{pmatrix}_{(l-1) \times (l-1)} \quad \text{if } j = 1 \text{ or } m$$

$$\mathbf{L}_{1j} = \begin{pmatrix} -3 & 1 & & 0 \\ 1 & -4 & \ddots & \\ & \ddots & \ddots & 1 \\ & & 1 & -4 & 1 \\ 0 & & & 1 & -3 \end{pmatrix}_{(l-1) \times (l-1)} \quad \text{for } 2 \leq j \leq m-1$$

$$\mathbf{L}_{2j} = \begin{pmatrix} -4 & 1 & & 0 \\ 1 & -4 & \ddots & \\ & \ddots & \ddots & 1 \\ & & 1 & -4 & 1 \\ 0 & & & 1 & -3 \end{pmatrix}_{l \times l} \quad \text{for } 1 \leq j \leq m-1$$

The corresponding boundary condition \widehat{bc}_1 also has some change. First, we denote some values

$$\begin{aligned}ud_i^{n+1} &= u_y((i-1)h, 0, (n+1)\Delta t) \\ uu_i^{n+1} &= u_y((i-1)h, Y, (n+1)\Delta t) \\ ur_j^{n+1} &= u_x(X, (j-1.5)h, (n+1)\Delta t) \\ vr_j^{n+1} &= v_x(X, (j-1)h, (n+1)\Delta t)\end{aligned}$$

The corresponding boundary condition \widehat{bc}_1 can be written as:

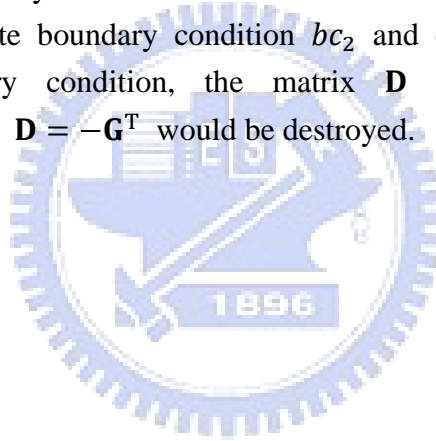
$$\widehat{bc}_1 = (bcu^{n+1} + bcu^n, bcv^{n+1} + bcv^n)^T / 2Reh^2$$

where

$$\begin{aligned} &bcu^n \\ &= (u_{1,2}^n, 0, \dots, 0, ur_2^n h, u_{1,3}^n, 0, \dots, 0, ur_3^n h, \dots \dots, u_{1,m+1}^n, 0, \dots, 0, ur_{l+1}^n h)^T \\ &+ h(-ud_2^n, -ud_3^n, \dots, -ud_l^n, 0, \dots \dots, 0, uu_2^n, uu_3^n, \dots, uu_l^n)^T \end{aligned}$$

$$\begin{aligned} &bcv^n \\ &= (v_{1,2}^n, 0, \dots, 0, vr_2^n, v_{1,3}^n, 0, \dots, 0, vr_3^n, \dots \dots, v_{1,m}^n, 0, \dots, 0, vr_m^n)^T \\ &+ (v_{2,1}^n, u_{3,1}^n, \dots, v_{l+1,1}^n, 0, \dots \dots, 0, v_{2,m+1}^n, u_{3,m+1}^n, \dots, v_{l+1,m+1}^n)^T \end{aligned}$$

Neumann boundary condition also can be used to compute $\widehat{\mathbf{D}}\mathbf{u}^{n+1}$, but we still use the discrete boundary condition bc_2 and original $\widehat{\mathbf{D}}$. If we use the Neumann boundary condition, the matrix \mathbf{D} would be changed, the symmetrization that $\mathbf{D} = -\mathbf{G}^T$ would be destroyed.



3 Matrix Factorization Method

3.1 Mathematical formulation

The immersed boundary method was first introduced by Peskin in [7, 8]. Consider the two-dimensional incompressible Navier-Stokes equation with an immersed massless boundary. Let the two-dimensional domain $\Omega = [0, X] \times [0, Y]$. Let the immersed boundary $\partial B = \Gamma$, which is a closed curve, as shown in Figure 4. The immersed boundary Γ is tracked by the parametric function $\mathbf{X}(s)$, $0 \leq s \leq L_\Gamma$, where L_Γ is the length of the immersed boundary Γ . The mathematical formulation is

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = -\nabla p + \frac{1}{Re} \Delta \mathbf{u} + \int_0^{L_\Gamma} \mathbf{f}(\mathbf{X}(s), t) \delta(\mathbf{x} - \mathbf{X}(s)) ds$$

$$\nabla \cdot \mathbf{u} = 0$$

$$\mathbf{w}(\mathbf{X}(s), t) = \int_\Omega \mathbf{u}(\mathbf{x}, t) \delta(\mathbf{x} - \mathbf{X}(s)) d\mathbf{x}$$

where $\mathbf{x} = (x, y)$, $\mathbf{f}(\mathbf{X}(s), t)$ is the Lagrangian boundary force defined on the immersed boundary, and $\mathbf{w}(\mathbf{X}(s), t)$ is the velocity at the immersed boundary Γ . δ is the two-dimensional Dirac delta function.

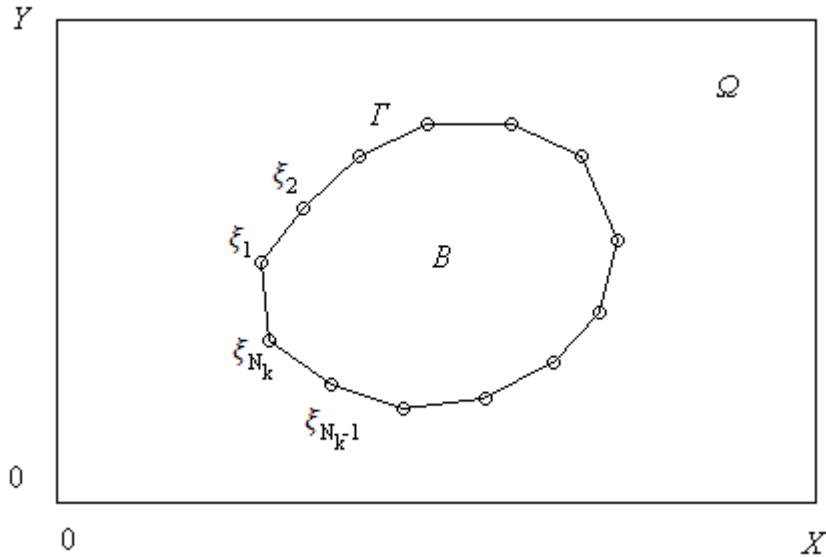


Fig. 4. The domain Ω , immersed object B , immersed boundary Γ , and the Lagrangian points ξ_k

3.2 Discretization

Similar to the discretization of the incompressible Navier-Stokes equations, we can discretize the mathematical formulation. Refer to [9], the discretization can be written as:

$$\begin{cases} \frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t} + \frac{3}{2}\hat{N}(\mathbf{u}^n) - \frac{1}{2}\hat{N}(\mathbf{u}^{n-1}) = -\hat{G}\phi + \frac{\hat{L}\mathbf{u}^{n+1} + \hat{L}\mathbf{u}^n}{2Re} + \hat{H}f + \widehat{\mathbf{bc}}_1 \\ \hat{D}\mathbf{u}^{n+1} = 0 + \mathbf{bc}_2 \\ \hat{E}\mathbf{u}^{n+1} = w^{n+1} \end{cases}$$

where \hat{H} and \hat{E} are the discrete regularization operator and the discrete interpolation operator, respectively.

The discrete boundary force f and the discrete boundary velocity w^{n+1} are defined on the Lagrangian markers ξ_k . We can regard ξ_k as $\mathbf{X}(s_k)$, where s_k is a monotone sequence between 0 and L_Γ , $1 \leq k \leq N_k$, as shown in figure 4. Let $\xi_k = (X_k, Y_k)$, we define the distance between each Lagrangian markers Δl_k as follows:

$$\begin{aligned} \Delta l_k &= |\xi_{k+1} - \xi_k| = \sqrt{(X_{k+1} - X_k)^2 + (Y_{k+1} - Y_k)^2} \quad \text{for } 1 \leq k \leq N_k - 1 \\ \Delta l_{N_k} &= |\xi_{N_k} - \xi_1| = \sqrt{(X_{N_k} - X_1)^2 + (Y_{N_k} - Y_1)^2} \end{aligned}$$

If the arc length of each part is known, we would rather define the arc length Δl_k then use distance. Then, we use Δl_k define Δs_k as follows:

$$\begin{aligned} \Delta s_1 &= (\Delta l_{N_k} + \Delta l_1)/2 \\ \Delta s_k &= (\Delta l_k + \Delta l_{k-1})/2 \quad \text{for } 2 \leq k \leq N_k \end{aligned}$$

Before we introduce \hat{H} and \hat{E} , the discrete delta function should be introduced first. Here, we use the discrete delta function which was introduced by Roma et al [10].

$$\delta(x, y) \approx d(x) \cdot d(y)$$

where the function $d(r)$ is

3.3 Matrix factorization method

Put the unknown terms \mathbf{u}^{n+1} , ϕ , and f on the left side of the equal sign, and the other terms on the right side. The discretization can be written as:

$$\begin{aligned} \left(\frac{\mathbf{I}}{\Delta t} - \frac{\hat{\mathbf{L}}}{2Re}\right)\mathbf{u}^{n+1} + \hat{\mathbf{G}}\phi - \hat{\mathbf{H}}f &= \left(\frac{\mathbf{I}}{\Delta t} + \frac{\hat{\mathbf{L}}}{2Re}\right)\mathbf{u}^n - \frac{3}{2}\hat{N}(\mathbf{u}^n) + \frac{1}{2}\hat{N}(\mathbf{u}^{n-1}) + \hat{bc}_1 \\ \hat{\mathbf{D}}\mathbf{u}^{n+1} &= 0 + bc_2 \\ \hat{\mathbf{E}}\mathbf{u}^{n+1} &= w^{n+1} \end{aligned}$$

Let $\hat{\mathbf{A}} = \frac{\mathbf{I}}{\Delta t} - \frac{\hat{\mathbf{L}}}{2Re}$ and $\hat{r} = \left(\frac{\mathbf{I}}{\Delta t} + \frac{\hat{\mathbf{L}}}{2Re}\right)\mathbf{u}^n - \frac{3}{2}\hat{N}(\mathbf{u}^n) + \frac{1}{2}\hat{N}(\mathbf{u}^{n-1})$, then

$$\begin{aligned} \hat{\mathbf{A}}\mathbf{u}^{n+1} + \hat{\mathbf{G}}\phi - \hat{\mathbf{H}}f &= \hat{r} + \hat{bc}_1 \\ \hat{\mathbf{D}}\mathbf{u}^{n+1} &= 0 + bc_2 \\ \hat{\mathbf{E}}\mathbf{u}^{n+1} &= w^{n+1} \end{aligned}$$

The matrix factorization method is a projection approach of the immersed boundary method introduced by Taira and Colonius [9]. Similar to the fractional step method, we have known the facts that $\hat{\mathbf{D}} = \mathbf{D}\mathbf{R}$ and $\hat{\mathbf{G}} = \hat{\mathbf{M}}^{-1}\mathbf{G}$. The discretization can be rewritten to

$$\begin{aligned} \hat{\mathbf{M}}\hat{\mathbf{A}}\mathbf{R}^{-1}(\mathbf{R}\mathbf{u}^{n+1}) + \hat{\mathbf{M}}\hat{\mathbf{G}}\phi - \hat{\mathbf{M}}\hat{\mathbf{H}}f &= \hat{\mathbf{M}}\hat{r} + \hat{\mathbf{M}}\hat{bc}_1 \\ -\mathbf{D}(\mathbf{R}\mathbf{u}^{n+1}) &= 0 - bc_2 \\ \hat{\mathbf{E}}\mathbf{R}^{-1}(\mathbf{R}\mathbf{u}^{n+1}) &= w^{n+1} \end{aligned}$$

Let $q^{n+1} = \mathbf{R}\mathbf{u}^{n+1}$, $\mathbf{A} = \hat{\mathbf{M}}\hat{\mathbf{A}}\mathbf{R}^{-1}$, $\mathbf{H} = \hat{\mathbf{M}}\hat{\mathbf{H}}$, $r = \hat{\mathbf{M}}\hat{r}$, $bc_1 = \hat{\mathbf{M}}\hat{bc}_1$ and $\mathbf{E} = \hat{\mathbf{E}}\mathbf{R}^{-1}$. It can be represented by the matrix form as follow:

$$\begin{pmatrix} \mathbf{A} & \mathbf{G} & -\mathbf{H} \\ -\mathbf{D} & \mathbf{0} & \mathbf{0} \\ \mathbf{E} & \mathbf{0} & \mathbf{0} \end{pmatrix} \begin{pmatrix} q^{n+1} \\ \phi \\ f \end{pmatrix} = \begin{pmatrix} r \\ 0 \\ w^{n+1} \end{pmatrix} + \begin{pmatrix} bc_1 \\ -bc_2 \\ 0 \end{pmatrix}$$

We know that $-\mathbf{D} = \mathbf{G}^T$, but \mathbf{E} and \mathbf{H} do not have the symmetric relation directly. So a transformed forcing function \hat{f} was introduced by Taira and Colonius [9]. It satisfies $\mathbf{H}f = -\mathbf{E}^T\hat{f}$. That is, $\hat{f} = -(\mathbf{E}\mathbf{E}^T)^{-1}\mathbf{E}\mathbf{H}f$. The matrix form becomes:

$$\begin{pmatrix} \mathbf{A} & \mathbf{G} & \mathbf{E}^T \\ \mathbf{G}^T & \mathbf{0} & \mathbf{0} \\ \mathbf{E} & \mathbf{0} & \mathbf{0} \end{pmatrix} \begin{pmatrix} q^{n+1} \\ \phi \\ \hat{f} \end{pmatrix} = \begin{pmatrix} r \\ 0 \\ w^{n+1} \end{pmatrix} + \begin{pmatrix} bc_1 \\ -bc_2 \\ 0 \end{pmatrix}$$

Let $\mathbf{Q} = (\mathbf{G} \ \mathbf{E}^T)$, $\lambda = (\phi \ \hat{f})^T$, $r_1 = r + bc_1$, and $r_2 = (-bc_2 \ w^{n+1})^T$:

$$\begin{pmatrix} \mathbf{A} & \mathbf{Q} \\ \mathbf{Q}^T & \mathbf{0} \end{pmatrix} \begin{pmatrix} q^{n+1} \\ \lambda \end{pmatrix} = \begin{pmatrix} r_1 \\ r_2 \end{pmatrix}$$

The fractional step method could be applied here.

$$\begin{pmatrix} \mathbf{A} & \mathbf{Q} \\ \mathbf{Q}^T & \mathbf{0} \end{pmatrix} \begin{pmatrix} q^{n+1} \\ \lambda \end{pmatrix} = \begin{pmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{Q}^T & -\mathbf{Q}^T \mathbf{A}^{-1} \mathbf{Q} \end{pmatrix} \begin{pmatrix} \mathbf{I} & \mathbf{A}^{-1} \mathbf{Q} \\ \mathbf{0} & \mathbf{I} \end{pmatrix} \begin{pmatrix} q^{n+1} \\ \lambda \end{pmatrix} = \begin{pmatrix} r_1 \\ r_2 \end{pmatrix}$$

Here we use \mathbf{B}^N to approximate \mathbf{A}^{-1} , there will exist an Nth order error term.

$$\begin{pmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{Q}^T & -\mathbf{Q}^T \mathbf{B}^N \mathbf{Q} \end{pmatrix} \begin{pmatrix} \mathbf{I} & \mathbf{B}^N \mathbf{Q} \\ \mathbf{0} & \mathbf{I} \end{pmatrix} \begin{pmatrix} q^{n+1} \\ \lambda \end{pmatrix} = \begin{pmatrix} r_1 \\ r_2 \end{pmatrix} + \begin{pmatrix} -\Delta t^N (\mathbf{L} \mathbf{M}^{-1})^N \mathbf{Q} \lambda / 2^N \\ 0 \end{pmatrix}$$

Let $q^* = q^{n+1} + \mathbf{B}^N \mathbf{Q} \lambda$, the matrix factorization method can be written in three steps:

$$\begin{aligned} \mathbf{A} q^* &= r_1 && \text{(Solve } q^*) \\ \mathbf{Q}^T \mathbf{B}^N \mathbf{Q} \lambda &= \mathbf{Q}^T q^* - r_2 && \text{(Solve } \lambda) \\ q^{n+1} &= q^* - \mathbf{B}^N \mathbf{Q} \lambda && \text{(Get } q^{n+1}) \end{aligned}$$

Since \mathbf{A} and $\mathbf{Q}^T \mathbf{B}^N \mathbf{Q}$ are symmetric and positive definite, conjugate gradient method can be applied to solve q^* and λ . Consider a special case that $\Delta x_i = \Delta y_j = h$, then $\mathbf{R} = \hat{\mathbf{M}} = h \mathbf{I}_{(l-1)m+l(m-1)}$. $\mathbf{H}f = -\mathbf{E}^T \hat{f}$ can be regard as:

$$\begin{aligned} \hat{\mathbf{M}} \hat{\mathbf{H}} f &= \hat{\mathbf{M}} \mathbf{F} \mathbf{S} f = h \mathbf{F} (\mathbf{S} f) \\ &= -\mathbf{R}^{-1} \hat{\mathbf{E}}^T \hat{f} = -h^2 \mathbf{R}^{-1} \mathbf{F} \hat{f} = h \mathbf{F} (-\hat{f}) \end{aligned}$$

So in this case, $\hat{f} = -\mathbf{S} f$.

4 Numerical Results

4.1 Error estimate of fractional step method

Consider the incompressible Navier-Stokes equations consisting of the momentum and continuity equations. There exists an exact solution that satisfies the Navier-Stokes equations. They are

$$\begin{aligned}u(x, y, t) &= -\cos(\pi x)\sin(\pi y)\exp(-2\pi^2 t/Re) \\v(x, y, t) &= \sin(\pi x)\cos(\pi y)\exp(-2\pi^2 t/Re) \\p(x, y, t) &= -(\cos(2\pi x) + \sin(2\pi y))\exp(-4\pi^2 t/Re)/4\end{aligned}$$

Use this data, we can know the initial and boundary conditions, and compare the numerical solution with the exact solution. We set the computational domain $\Omega = [0,1] \times [0,1]$, the computational time $T = 1$, and the Reynolds number $Re = 40$.

To check the order of accuracy of the fractional step method, let $\Delta t = 1/400, 1/800, 1/1600, 1/3200$, and $1/6400$, respectively. $N = 3$. Suppose that $\Delta x_i = \Delta y_j = h$, and let $h = 40\Delta t$. We define the maximum error by subtracting the numerical solution from the exact solution and take infinite norms. The order is known by taking \log_2 on the ratio of the errors. As shown in table. 1, the fractional step method has second-order accuracy in time.

Table 1

The maximum errors from different Δt

Δt	h	maximum error	order
1/400	1/10	3.2781e-004	2.47
1/800	1/20	5.9040e-005	1.97
1/1600	1/40	1.5094e-005	2.01
1/3200	1/80	3.7449e-006	1.96
1/6400	1/160	9.6555e-007	—

4.2 The flow past a cylinder

Consider a situation that the flow past a cylinder, where the diameter of the cylinder is D . The matrix factorization method can be applied here to simulate this situation.

In this test, the computational domain is set by $\Omega = [0,16] \times [0,8]$. The Reynolds number $Re = 100$ and 200 , respectively. We choose $\Delta t = 1/640$ and $\Delta x = \Delta y = 1/16$. The initial condition is given by $u|_{t=0} = 0$ and $v|_{t=0} = 0$. The boundary condition is shown in Fig.6. To construct the cylinder, we put the center of the cylinder at the position $(4,4)$, and let the diameter $D = 1$. Choose the number of the markers $N_k = 64$, then $\Delta s = \pi/64$. Notice that $\Delta s \approx 0.785h$.

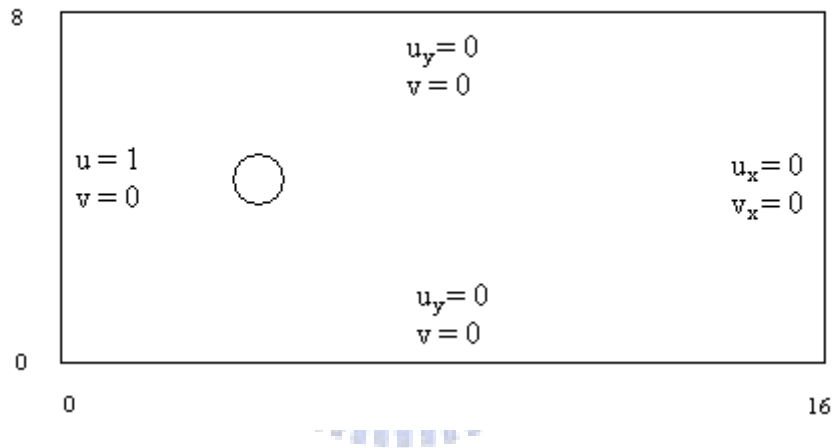


Fig. 6. The boundary condition and the computational domain

The simulations which the flow past a cylinder at Reynolds number 100 and 200 show the periodic vortex shedding. In figure 7, we can see the periodic vortex shedding in the vorticity contours. The vorticity of the flow is defined as $v_x - u_y$.

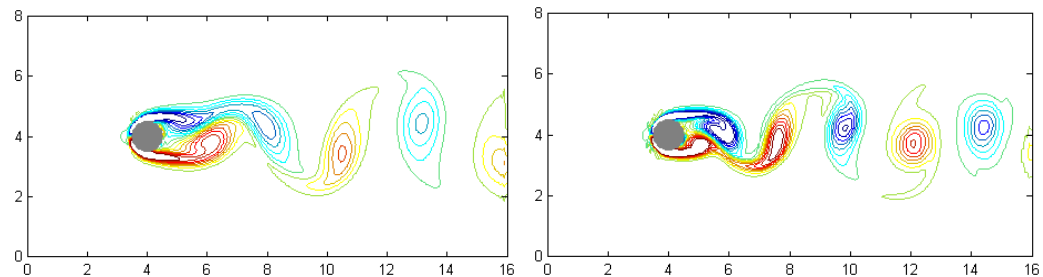


Fig. 7. The vorticity contours at Reynolds number 100(left) and 200(right) at the time $T = 60$

There are three quantities, the drag coefficient C_D , lift coefficient C_L , and the Strouhal number St . We can use the three quantities to compare this simulation with others. The three quantities are defined as:

$$C_D = \frac{F_D}{U_\infty^2 D/2}$$

$$C_L = \frac{F_L}{U_\infty^2 D/2}$$

$$St = \frac{f_q D}{U_\infty}$$

where F_D and F_L are the drag force and lift force, $U_\infty = 1$ in this simulation, f_q is the frequency of the vortex shedding. F_D and F_L can be obtain in [11], which are

$$(F_D, F_L) = - \int_{\Omega} \int_0^{L_F} \mathbf{f}(\mathbf{X}(s), t) \delta(\mathbf{x} - \mathbf{X}(s)) ds dx$$

F_D and F_L are approximated by

$$F_D \approx \sum_i (F_1 \hat{S} \cdot fx)_i h^2$$

$$F_L \approx \sum_i (F_2 \hat{S} \cdot fy)_i h^2$$

In figure 8 and 9, we can observe the periodic vortex shedding. In table 2 and 3, we compare the three quantities with the previous numerical results which refer to [9, 11, 12, 13, 14, 15] at Reynolds number $Re = 100$ and 200.

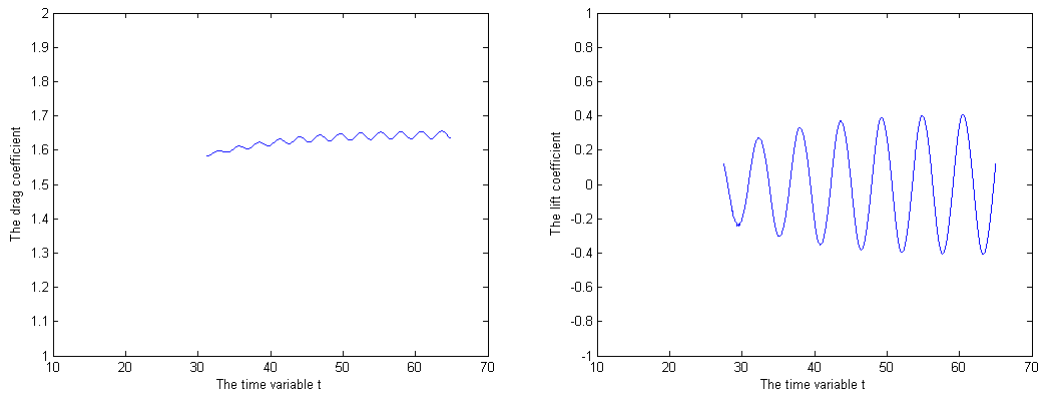


Fig. 8. The time evolution of drag(left) and lift(right) coefficients at Reynolds number $Re = 100$

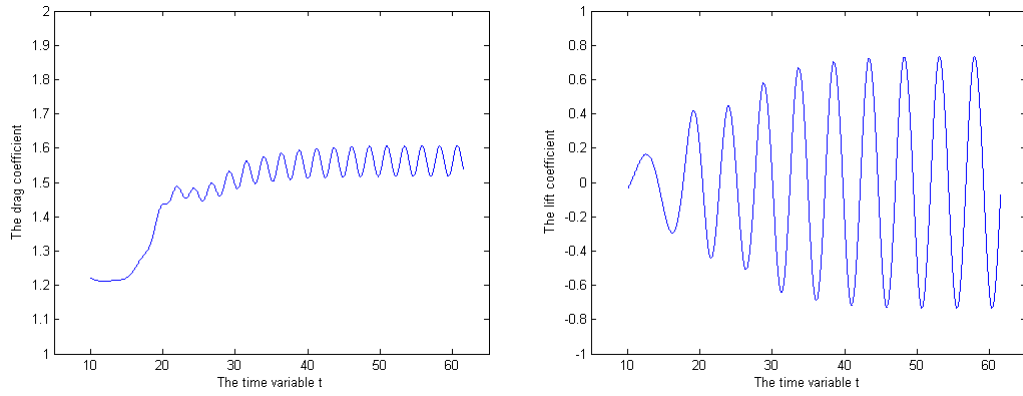


Fig. 9. The time evolution of drag(left) and lift(right) coefficients at Reynolds number $Re = 200$

Table 2

The comparisons of lift and drag coefficients and Strouhal number of $Re = 100$

	Present	Lai and Peskin[11]	Kim et al.[12]	Silva et al.[13]
C_L	0.40	0.33	0.32	-
C_D	1.64	1.44	1.33	1.39
St	0.177	0.165	-	0.16

Table 3

The comparisons of lift and drag coefficients and Strouhal number of $Re = 200$

	Present	Taira and Colonius [9]	Linnick and Fasel[14]	Liu et al[15]
C_L	0.72	0.69	0.69	0.69
C_D	1.56	1.36	1.34	1.31
St	0.206	0.197	0.197	0.192

4.3 The flow past two cylinders

In this test, we simulate the flow past two cylinders at the Reynolds number $Re = 200$. Let the diameter D of the two cylinders are the same, and set to be $D = 1$. Similar to the previous case, we set the computation domain $\Omega = [0,16] \times [0,8]$, $\Delta t = 1/640$ and $\Delta x = \Delta y = 1/16$. The initial and boundary conditions are as before, too. To construct the cylinder, we put the centers of this two cylinders at the position $(4,2.5)$ and $(4,5.5)$. We also choose $\Delta s = \pi/64$, so the number of the markers $N_k = 128$. The time evolution of drag and lift coefficients and the vorticity contours are shown in figure 10 and 11. We can observe that the drag coefficients of the two cylinders are similar, and the lift coefficients of the two cylinders are symmetric.

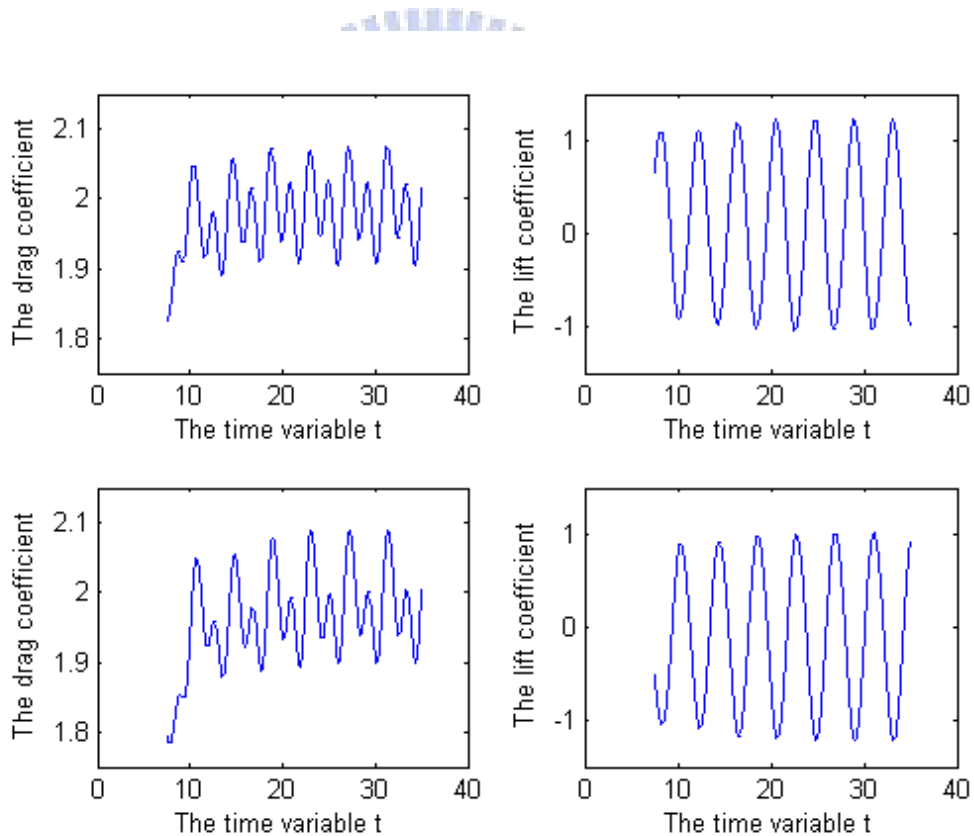


Fig. 10. The time evolution of drag(left) and lift(right) coefficients of the upper(up) and lower (down) cylinders.

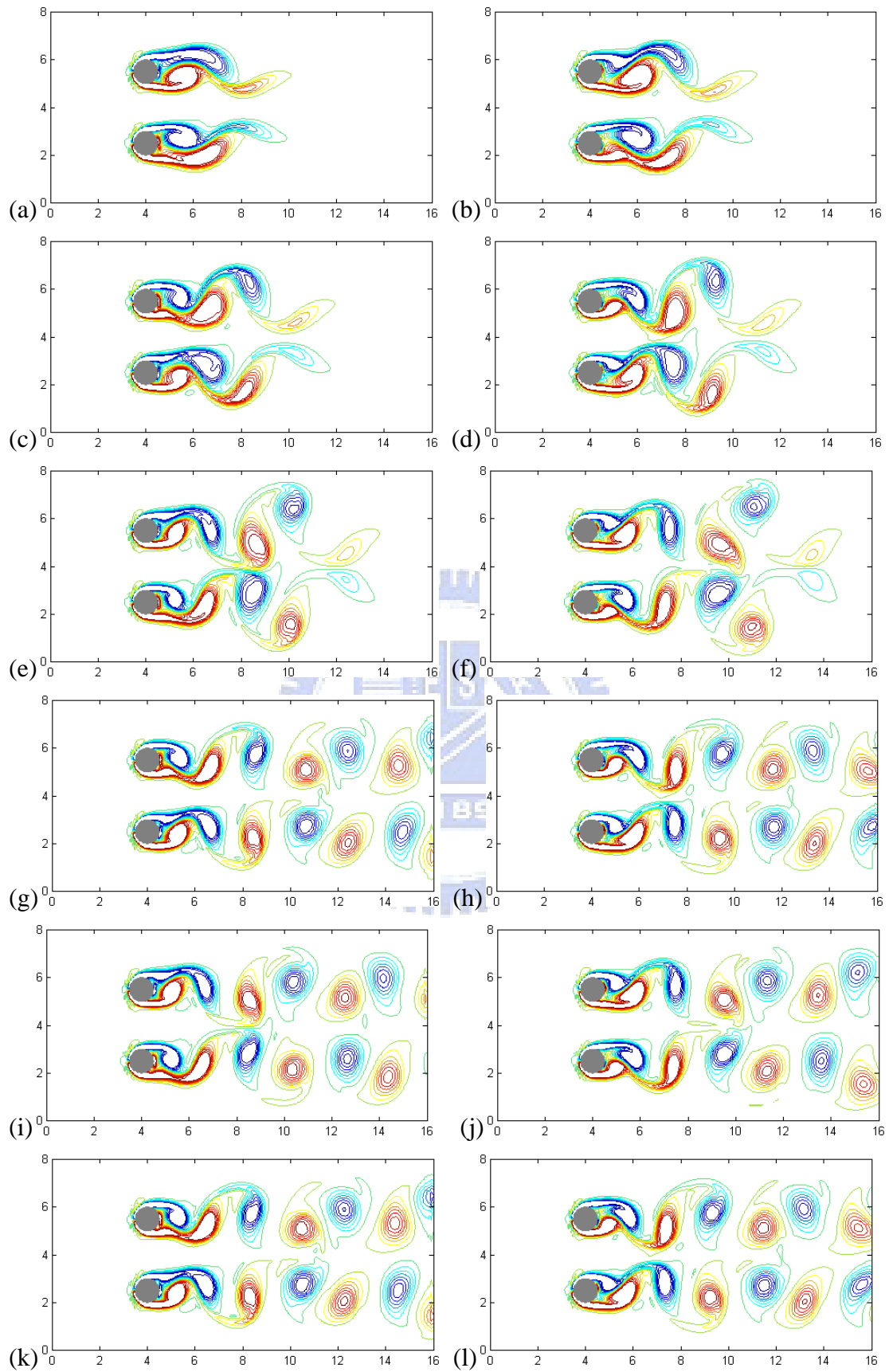


Fig. 11. The vorticity contours when the flow past two cylinders : (a)t=7, (b)t=8, (c)t=9, (d)t=10, (e)t=11, (f)t=12, (g)t=30, (h)t=31, (i)t=32, (j)t=33, (k)t=34, (l)t=35.

4.4 The flow past a wing

Here, we simulate the flow past through a wing of the airplane at the Reynolds number $Re = 200$. We use the same computational domain, mesh, initial condition and boundary conditions as before, but the immersed object. See figure 12, the shape of the immersed object is a airfoil, which is a thin winglike structure. We rotate the wing by an angle, which is -30° here, as shown in figure 13. The periodic vortex shedding also can be observed behind the wing in figure 14.

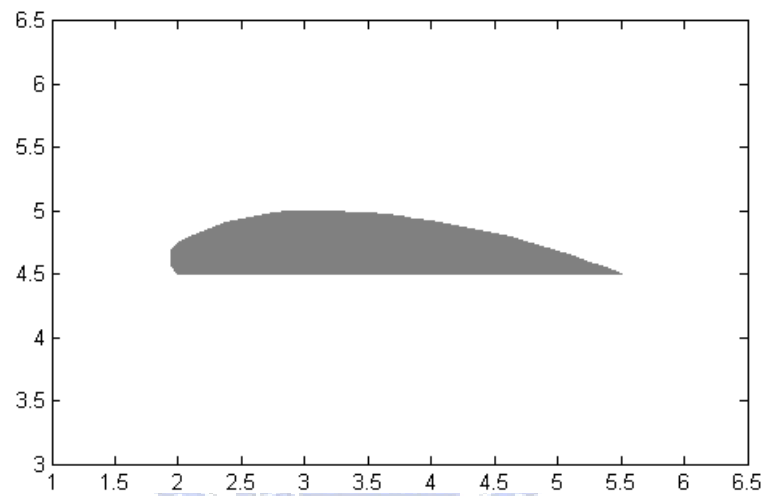


Fig. 12. The shape of the immersed object

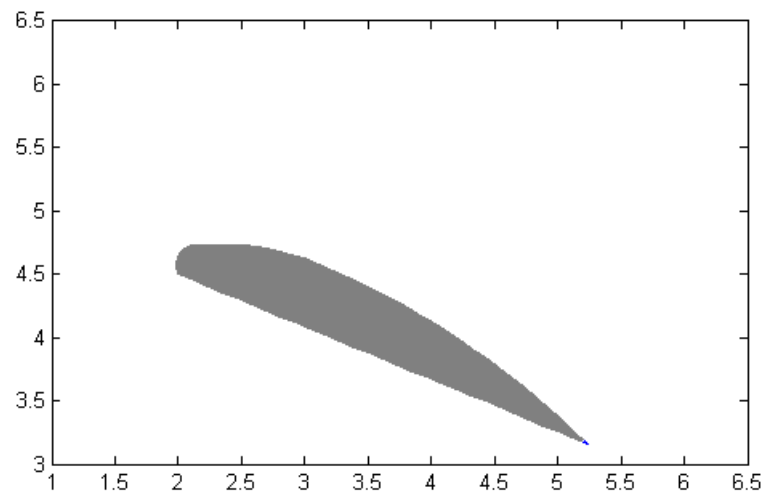


Fig. 13. The placement of the wing

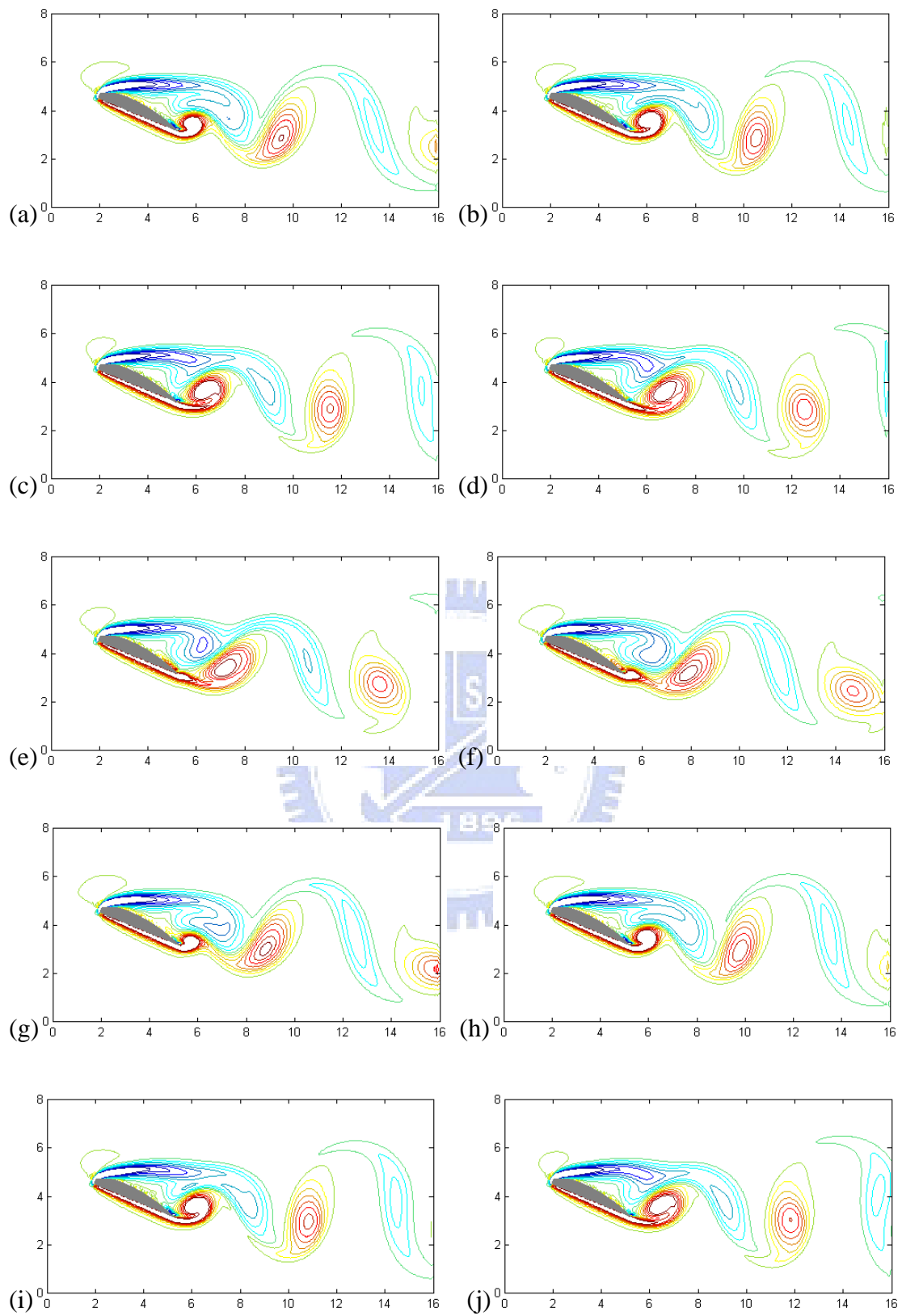


Fig. 14. The vorticity contours when the flow past a wing : (a)t=51, (b)t=52, (c)t=53, (d)t=54, (e)t=55, (f)t=56, (g)t=57, (h)t=58, (i)t=59, (j)t=60.

4.5 The flow past an oscillating cylinder

Consider the same simulation as the flow past a cylinder at Reynolds number $Re = 100$, but the cylinder is moving here. We impose the velocity $\mathbf{w}(\mathbf{X}(s), t)$ at the boundary Γ as $\mathbf{w}(\mathbf{X}(s), t) = (0, 0.14 \cos(2\pi f_c t))$, where f_c is the frequency that the cylinder oscillates. That is, the cylinder oscillates vertical to the stream. Here we choose $f_c = 2f_q$. The time evolution of drag coefficient and the vorticity contours are shown in figure 15 and 16. Compare with figure 8, we can see that the frequency of the vortex shedding is influenced by the oscillation of the cylinder after the cylinder moves. In table 4, we compare the drag and lift coefficients with the previous numerical results which refer to [17, 18].

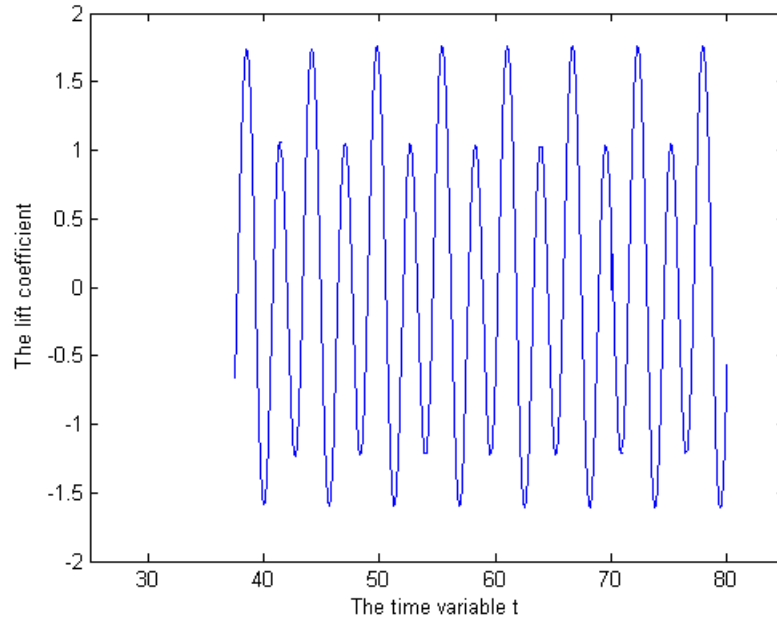


Fig. 15. The time evolution of the lift(right) coefficients

Table 4

The comparisons of the lift and drag coefficients

	Present	Su et al.[17]	Hurlbut et al.[18]
C_L	1.75	0.97	0.95
C_D	1.84	1.70	1.68

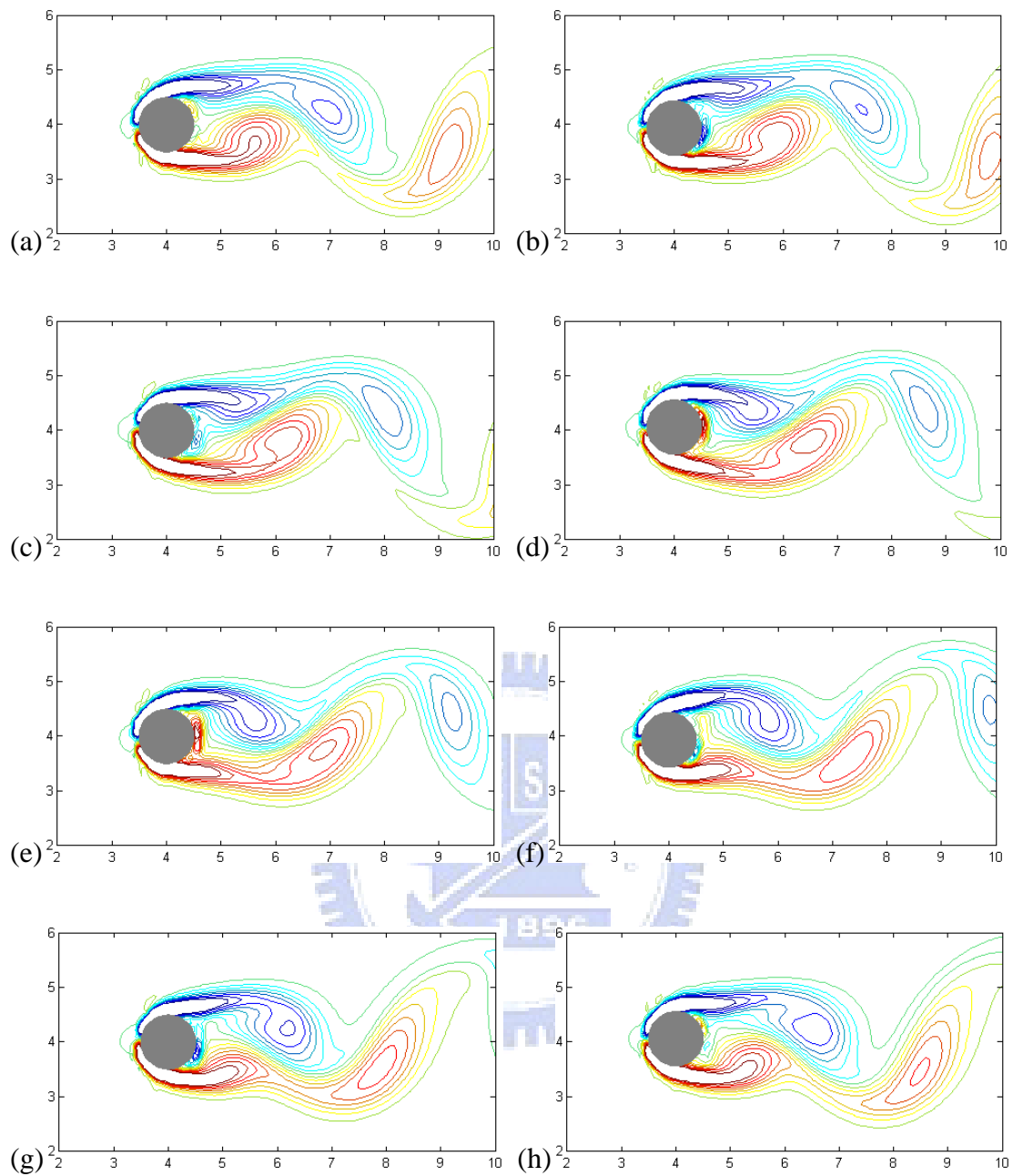


Fig. 16. The vorticity contours when the flow past a oscillating cylinder : (a) $t=4.21875$, (b) $t=4.921875$, (c) $t=5.625$, (d) $t=6.328125$, (e) $t=7.03125$, (f) $t=7.734325$, (g) $t=8.4375$, (h) $t=9.140625$.

5 Conclusion

In this thesis, we use the matrix factorization method introduced by Taira and Colonius [9] to simulate the flow past an immersed object. In the numerical result, we see that this method can handle the immersed object with complex shape, and which the immersed object is moving, even the two or more objects. It is useful in the engineering applications. In Section 4.4, we simulate the situation when the flow past a wing. The flow produces vortex shedding behind the wing.

Here we suppose that the mesh widths of each cell are the same, that is, we let $\Delta x_i = \Delta y_j = h$ for all i, j . In fact, the matrix factorization method can use the different grid size of each cell. If the higher accuracy is desired, the grid size near the immersed object has to be small. We can adapt the code to handle the different mesh widths, to improve the accuracy.



References

- [1] A.J. Chorin, "Numerical solution of the Navier–Stokes equations." *Math. Comput.* Vol. 22, pp. 745–762, 1968.
- [2] J. Kim, P. Moin, "Application of a fractional-step method to incompressible Navier–Stokes equations." *J. Comput. Phys.* Vol. 59, pp. 308–323, 1985.
- [3] J.B. Perot, "An analysis of the fractional step method." *J. Comput. Phys.* 108 51–58, 1993.
- [4] R. Codina, "Pressure stability in fractional step finite element methods for incompressible flows." *J. Comput. Phys.* Vol. 170, pp. 112–140, 2001.
- [5] F.H. Harlow, J.E. Welsh, "Numerical calculation of time-dependent viscous incompressible flow of fluid with a free surface." *Phys. Fluids.* Vol. 8, pp. 2181–2189, 1965.
- [6] D.L. Brown, R. Cortez, M.L. Minion, "Accurate projection methods for the incompressible Navier–Stokes equations." *J. Comput. Phys.* Vol. 168, pp. 464–499, 2001.
- [7] C.S. Peskin, "Flow patterns around heart valves: a numerical method." *J. Comput. Phys.* Vol. 10, pp. 252–271, 1972.
- [8] C.S. Peskin, "The immersed boundary method". *Acta Numer.* Vol. 11, pp. 479–517, 2002.
- [9] K. Taira, T. Colonius, "The immersed boundary method: A projection approach." *J. Comput. Phys.* Vol. 225, pp. 2118–2137, 2007.
- [10] A.M. Roma, C.S. Peskin, M.J. Berger, "An adaptive version of the immersed boundary method." *J. Comput. Phys.* Vol. 153, pp. 509–534, 1999.
- [11] M. Lai, C.S. Peskin, "An immersed boundary method with formal second-order accuracy and reduced numerical viscosity." *J. Comput. Phys.* Vol. 160, pp. 705–719, 2000.
- [12] J. Kim, D. Kim, H. Choi, "An immersed-boundary finite-volume method for simulations of flow in complex geometries." *J. Comput. Phys.* Vol. 171, pp. 132–150, 2001.
- [13] A. L. F. Lima E Silva, A. Silveira-Neto, J. J. R. Damasceno, "Numerical simulation of two-dimensional flows over a circular cylinder using the immersed boundary method." *J. Comput. Phys.* Vol. 189, pp. 351–370, 2003.
- [14] M.N. Linnick, H.F. Fasel, "A high-order immersed interface method for simulating unsteady incompressible flows on irregular domains." *J. Comput. Phys.* Vol. 204, pp. 157–192, 2005.
- [15] C. Liu, X. Zheng, C.H. Sung, "Preconditioned multigrid methods for unsteady incompressible flows." *J. Comput. Phys.* Vol. 139, pp. 35–57, 1998.
- [16] R. Témam, "Sur l'approximation de la solution des équations de Navier–Stokes par la méthode des pas fractionnaires." *Arch. Rat. Mech. Anal.* Vol. 32 (2), pp. 135–153, 1969.
- [17] S. Su, M. Lai, C. Lin, "An immersed boundary technique or simulating complex flows with rigid boundary." *Comput. Fluids.* Vol. 36, pp. 313–324, 2007.
- [18] S.E. Hurlbut, M.L. Spaulding, F.M. White, "Numerical Solution for Laminar Two Dimensional Flow About a Cylinder Oscillating in a Uniform Stream", *Trans ASME, J. Fluids Eng.* Vol. 104, pp. 214–221, 1982.
- [19] W. Chang, F. Giraldo, B. Perot, "Analysis of an exact fractional step method." *J. Comput. Phys.* Vol. 180, pp. 183–199, 2002.