Mahidol University Center for Scientific Computing

2017 Midyear Report

C. Boonyasiriwat, C. Boonthanawat, T. Chantanasaro,
H. Intarak, A. Nakpathom, P. Pukhamwong,
P. Situngnoen, P. Somboon, and W. Thongyoy

September 20, 2017

Contents

MCSC MIDYEAR REPORT 2017

This technical report summarizes the midyear 2017 research accomplishments of the researchers of the Mahidol University Center for Scientific Computing (MCSC). Our research includes numerical simulations of fluid flows, seismic, electromagnetic, and tsunami wave propagation, and dynamical systems.

Currently, MCSC members are composed of one lecturer, one administrative staff, one software developer, five Ph.D. students, six M.Sc. students, and one B.Sc. student.

Chaiwoot Boonyasiriwat and colleagues
Mahidol University Center for Scientific Computing
Department of Physics, Faculty of Science, Mahidol University
272 Rama VI Road, Ratchathewi, Bangkok 10400, THAILAND
Phone: 66-8-0938-4132
Fax: 66-2-354-7159
Email: chaiwoot@gmail.com
Website: http://mcsc.sc.mahidol.ac.th

Unsplit-field PML formulation for shallow-water equations and finite volume method

Chaiwoot Boonyasiriwat

ABSTRACT

This report presents the 6-month research progress on the project titled "Development of parallel programs for tsunami simulation" funded by IPST under the grant number 001/2558. The progress is composed of two parts. In the first part, an unsplit-field PML formulation for the shallow-water equations was derived to modeling tsunami wave propagation in unbounded domains. In the second part, the theory of finite volume method which is a method of choice for solving nonlinear hyperbolic systems are presented with a numerical experiment on the 1D linear advection equation.

INTRODUCTION

Perfectly matched layer (PML) is the most widely used method for modeling wave propagation in unbounded domains due to its effectiveness and efficiency compared to other methods. The original version of PML developed by Bérenger (1994) now known as a split-field PML formulation for the Maxwell's equations was shown to be unstable by Abarbanel and Gottlieb (1997). The split-field PML method was later applied to linearized Euler equations by Hu (1996). Similarly to the case of Maxwell's equations, the split-field PML formulation for linearized Euler equations was also shown to be unstable by Hesthaven (1998). Stable PML formulations were later presented in the so-called unsplit-field form for Maxwell's equations (Abarbanel and Gottlieb, 1998) and linearized Euler equations (Hesthaven, 1998; Abarbanel et al., 1999; Hu, 2001). The split-field PML of Bérenger (1994) was later applied to linearized shallow-water equations by Navon et al. (2004) and, as expected, the resulting PML equations are unstable and a filter was needed to obtain a stable result. Abarbanel et al. (2003) applied the approach of Abarbanel et al. (1999) and Hu (2001) and obtained two unsplit-field PML formulations stable for the linearized shallow-water equations but unstable for the nonlinear case. Recently, Barucq et al. (2010) applied the approach of Nataf (2006) to derive a stable PML formulation for linearized shallow-water equations. Stable PML formulation for nonlinear shallow-water equations is still an open problem.

In this report, I present unsplit-field PML formulations in primitive variables for 1D and 2D shallow-water equations. The derivation of the PML equations for the 1D case is first presented. Then the same procedure was applied to obtain the 2D formulation which are given without a derivation. To compare the effectiveness and stability of the proposed PML formulation, I need to learn the finite volume method which is currently one of the most widely used methods for solving nonlinear systems of conservation laws including the shallow-water equations. The finite volume method can accurately recover discontinuous solutions or shock waves and, therefore, it is suitable for tsunami simulation which include shock-wave solutions near the coastline. The second part of this report presents the result of my learning of the finite volume method which is given in the form of concise theory and preliminary numerical experiment.

PML FORMULATIONS FOR SHALLOW-WATER EQUATIONS

1D Shallow-Water Equations

In this section, we present the derivation of an unsplit-field PML formulation for the 1D shallow-water equations given as

$$\begin{aligned} h_{,t} + uh_{,x} + hu_{,x} &= 0, \\ u_{,t} + uu_{,x} + gh_{,x} &= 0, \end{aligned} \tag{1}$$

where $a_{,b} \equiv \partial a/\partial b$, h is wave height, u is depth-averaged particle velocity in x-direction, g is gravitational acceleration.

Applying temporal Fourier transform to equations 1 yields

$$\begin{aligned} -i\omega\tilde{h} + \tilde{u} * \tilde{h}_{,x} + \tilde{h} * \tilde{u}_{,x} &= 0, \\ -i\omega\tilde{u} + \tilde{u} * \tilde{u}_{,x} + g\tilde{h}_{,x} &= 0, \end{aligned} \tag{2}$$

where $*$ denotes temporal convolution and $\tilde{\square}$ is the temporal Fourier transform of $\square$. Using the coordinate stretching approach of Chew and Weedon (1994), the spatial derivatives in equation 2 is replaced by

$$\frac{\partial}{\partial x} \rightarrow \left(\frac{1}{1 + \frac{i\sigma(x)}{\omega}} \right) \frac{\partial}{\partial x} = \alpha(x) \frac{\partial}{\partial x} \tag{3}$$

where $\sigma(x) > 0$ in the PML region and $\sigma(x) = 0$ in the physical region, and we then obtain the PML equations in the frequency domain, namely,

$$\begin{aligned} -i\omega\tilde{h} + \tilde{u} * (\alpha\tilde{h}_{,x}) + \tilde{h} * (\alpha\tilde{u}_{,x}) &= 0, \\ -i\omega\tilde{u} + \tilde{u} * (\alpha\tilde{u}_{,x}) + g(\alpha\tilde{h}_{,x}) &= 0, \end{aligned} \tag{4}$$

Using the two auxiliary variables $\tilde{H}_x$ and $\tilde{U}_x$ defined as

$$\begin{aligned} -i\omega\tilde{H}_x &= \alpha\tilde{h}_{,x}, \\ -i\omega\tilde{U}_x &= \alpha\tilde{u}_{,x}, \end{aligned} \tag{5}$$

equation 4 becomes

$$\begin{aligned} -i\omega\tilde{h} + \tilde{u} * (-i\omega\tilde{H_x}) + \tilde{h} * (-i\omega\tilde{U_x}) = 0, \\ -i\omega\tilde{u} + \tilde{u} * (-i\omega\tilde{U_x}) + g(-i\omega\tilde{H_x}) = 0, \end{aligned} \tag{6}$$

Applying inverse Fourier transform to equations 5 and 6 yields

$$\begin{aligned} h_{,t} + uH_{x,t} + hU_{x,t} = 0 \\ u_{,t} + uU_{x,t} + gH_{x,t} = 0 \end{aligned} \tag{7}$$

Equation 5 can be rewritten as

$$\begin{aligned} (-i\omega + \sigma)\,\tilde{H_x} = \tilde{h}_{,x}, \\ (-i\omega + \sigma)\,\tilde{U_x} = \tilde{u}_{,x}, \end{aligned} \tag{8}$$

Applying inverse Fourier transform to equations 8 yields

$$\begin{aligned} H_{x,t} + \sigma H_x = h_{,x} \\ U_{x,t} + \sigma U_x = u_{,x} \end{aligned} \tag{9}$$

Using equation 9, equation 7 can be rewritten as

$$\begin{aligned} h_{,t} + u(h_{,x} - \sigma H_x) + h(u_{,x} - \sigma U_x) = 0 \\ u_{,t} + u(u_{,x} - \sigma U_x) + g(h_{,x} - \sigma H_x) = 0 \end{aligned} \tag{10}$$

Equations 9 and 10 constitute the unsplit-field PML formulation in primitive variables for the 1D shallow-water equation 1.

2D Shallow-Water Equations

In this section, we apply the same procedure given in the previous section to the 2D shallow-water equations given as

$$\begin{aligned} h_{,t} + uh_{,x} + vh_{,y} + h(u_{,x} + v_{,y}) = 0, \\ u_{,t} + uu_{,x} + vu_{,y} + gh_{,x} - fv = 0, \\ v_{,t} + uv_{,x} + vv_{,y} + gh_{,y} + fu = 0, \end{aligned} \tag{11}$$

where u and v are the particle velocity in x- and y-directions, respectively, and f is the Coriolis factor, and obtain the unsplit-field PML equations in primitive variables as follows.

$$\begin{aligned} h_{,t} + u(h_{,x} - \sigma_x H_x) + v(h_{,y} - \sigma_y H_y) \\ +h(u_{,x} - \sigma_x u_x + v_{,y} - \sigma_y v_y) = 0, \\ u_{,t} + u(u_{,x} - \sigma_x U_x) + v(u_{,y} - \sigma_y U_y) \\ +g(h_{,x} - \sigma_x H_x) - fv = 0, \\ v_{,t} + u(v_{,x} - \sigma_x V_x) + v(v_{,y} - \sigma_y V_y) \\ +g(h_{,y} - \sigma_y H_y) + fu = 0, \\ H_{x,t} + \sigma_x H_x = h_{,x} \\ H_{y,t} + \sigma_y H_y = h_{,y} \\ U_{x,t} + \sigma_x U_x = u_{,x} \\ U_{y,t} + \sigma_y U_y = u_{,y} \\ V_{x,t} + \sigma_x V_x = v_{,x} \\ V_{y,t} + \sigma_y V_y = v_{,y} \end{aligned} \tag{12}$$

where $H_x, H_y, U_x, U_y, V_x, V_y$ are auxiliary variables, and σ_x and σ_y are the absorption coefficient in x- and y-directions, respectively.

FINITE VOLUME METHOD

In this section, I present the theory of finite volume method which is suitable for numerically solving nonlinear systems of conservation laws. So I begin with the introduction to conservation laws. Then I explain the finite volume method.

Scalar Conservation Laws

A scalar conservation law in one space dimension can be expressed in the integral form as

$$\frac{d}{dt}\int_{x_1}^{x_2} q(x,t)dx = f(q,x_1,t) - f(q,x_2,t), \tag{13}$$

where $q(x,t)$ is the conserved quantity and $f(q,x,t)$ is a flux function. In most cases, we will be dealing with autonomous flux $f(q)$ which does not depend explicitly on x and t. Equation 13 then becomes

$$\frac{d}{dt}\int_{x_1}^{x_2} q(x,t)dx = f(q(x_1,t)) - f(q(x_2,t)). \tag{14}$$

It can be shown that equation 14 can be rewritten as (LeVeque, 2002)

$$\int_{x_1}^{x_2}\left[\frac{\partial q(x,t)}{\partial t} + \frac{\partial f(q(x,t))}{\partial x}\right]dx = 0. \tag{15}$$

Equation 15 implies that the integrand must vanish and leads to the differential form of the conservation law, namely

$$\frac{\partial q(x,t)}{\partial t} + \frac{\partial f(q(x,t))}{\partial x} = 0, \tag{16}$$

which is called the conservative form. Equation 16 can also be written in the quasilinear form as

$$\frac{\partial q(x,t)}{\partial t} + \frac{df}{dq}\frac{\partial q(x,t)}{\partial x} = 0. \tag{17}$$

Method of Characteristics

A characteristic curve along which q is constant can be determined by setting the total derivative of $q(x,t)$ to zero, i.e., The total derivative of $q(x,t)$ is

$$\frac{dq}{dt} = \frac{\partial q}{\partial t} + \frac{dx}{dt}\frac{\partial q}{\partial x} = 0. \tag{18}$$

Comparing equations 17 and 18, we obtain the relation

$$\frac{dx}{dt} = \frac{df}{dq} \tag{19}$$

whose solution is the characteristic curve

$$x = x_0 + \frac{df}{dq}t, \tag{20}$$

where x_0 is a constant of integration.

Linear Advection Equation

Consider the Cauchy problem for the 1D linear advection equation given by

$$\begin{aligned} \frac{\partial q}{\partial t} + u\frac{\partial q}{\partial x} &= 0, \\ q(x,0) &= \tilde{q}(x), \qquad x \in \Re \end{aligned} \tag{21}$$

where the fluid velocity u is constant and $\tilde{q}$ is some function. In this case, the flux function is $f(q) = uq$ and the characteristic curve satisfies

$$\frac{dx}{dt} = \frac{df}{dq} = u, \tag{22}$$

so the curve itself is given by

$$x(t) = x_0 + ut. \tag{23}$$

Since q is constant along the characteristics, we can express the solution to the Cauchy problem 21 as

$$q(x,t) = \tilde{q}(x - ut). \tag{24}$$

Inviscid Burgers Equation

The inviscid Burgers equation given by

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}\left(\frac{1}{2}u^2\right) = \frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = 0 \tag{25}$$

models the conservation of momentum ρu of an inviscid and incompressible fluid with no pressure gradient. Since the mass density ρ is constant, it was removed from the momentum equation resulting in the conservation of the fluid velocity u.

Consider again the Cauchy problem with initial data given by

$$u(x,0) = \tilde{u}(x), \qquad x \in \Re \tag{26}$$

In this case, the characteristics also satisfy

$$\frac{dx}{dt} = u. \tag{27}$$

So a solution to the Cauchy problem 26 for the inviscid Burgers equation 25 is

$$u(x,t) = \tilde{u}(x - ut). \tag{28}$$

Note that this is an implicit equation for the solution since the solution itself appears as the argument of the initial condition on the right-hand side. The exact solution of the inviscid Burgers equation can be obtained using the Cole-Hopf transformation (LeVeque, 2002).

Although the characteristic equation of the Burgers equation has the same form as that of the linear advection equation, all the characteristic curves of the linear advection equation have the same slope and will never collide while the characteristic curves of the Burgers equation can have different slopes depending on the initial function $\tilde{u}$. When the characteristic curves collide, a shock wave is formed even when the initial function is smooth.

Advective Acoustics Equations

When the perturbation is small, the Euler equations can be linearized into the advective acoustics equations

$$q_{,t} + Aq_{,x} = 0, \tag{29}$$

where

$$q(x,t) = \begin{bmatrix} p(x,t) \\ u(x,t) \end{bmatrix} \tag{30}$$

is the perturbation from the fixed state

$$q_0 = \begin{bmatrix} p_0 \\ u_0 \end{bmatrix} \tag{31}$$

and

$$A = \begin{bmatrix} u_0 & K_0 \\ 1/\rho_0 & u_0 \end{bmatrix}. \tag{32}$$

Here K is the bulk modulus, ρ is mass density, p is pressure, and u_0 is fluid particle velocity.

The matrix A has the eigenvalues

$$\begin{aligned} \lambda_1 &= u_0 - c_0, \\ \lambda_2 &= u_0 + c_0, \end{aligned} \tag{33}$$

and the corresponding normalized eigenvectors

$$\begin{aligned} r_1 &= \begin{bmatrix} -Z_0/\sqrt{1+Z_0^2} \\ 1/\sqrt{1+Z_0^2} \end{bmatrix} \\ r_2 &= \begin{bmatrix} Z_0/\sqrt{1+Z_0^2} \\ 1/\sqrt{1+Z_0^2} \end{bmatrix} \end{aligned} \tag{34}$$

where the sound speed $c_0 = \sqrt{K_0/\rho_0}$ and the impedance $Z_0 = \rho_0 c_0$.

Shallow Water Equations

The one-dimensional shallow water equations can be written in the conservation form as

$$q_{,t} + f(q)_{,x} = 0, \tag{35}$$

where the conserved quantity q is

$$q(x,t) = \begin{bmatrix} h \\ hu \end{bmatrix}, \tag{36}$$

and the flux function f is

$$f(x,t) = \begin{bmatrix} uh \\ hu^2 + \frac{1}{2}gh^2 \end{bmatrix}. \tag{37}$$

Here h is the wave height or fluid depth, u is the particle velocity, and g is the gravitational acceleration.

To use the finite-volume method, we need to turn the shallow-water equations into the quasilinear form, given by

$$q_{,t} + f'(q)q_{,x} = 0, \tag{38}$$

where the Jacobian matrix $f'(q)$ is

$$f'(q) = \begin{bmatrix} 0 & 1 \\ -u^2 + gh & 2u \end{bmatrix}. \tag{39}$$

The eigenvalues of $f'(q)$ are

$$\begin{aligned} \lambda^1 &= u - \sqrt{gh}, \\ \lambda^2 &= u + \sqrt{gh}, \end{aligned} \tag{40}$$

and the corresponding eigenvectors are

$$\begin{aligned} r^1 &= \begin{bmatrix} 1 \\ u - \sqrt{gh} \end{bmatrix}, \\ r^2 &= \begin{bmatrix} 1 \\ u + \sqrt{gh} \end{bmatrix}. \end{aligned} \tag{41}$$

Conservative Numerical Schemes

In finite volume method, the spatial domain is discretized into a finite number of cells or finite volumes. In 1D, the ith cell is denoted by

$$C_i = (x_{i-1/2}, x_{i+1/2}). \tag{42}$$

The integral form of the conservation law 14 for the ith cell is given by

$$\frac{d}{dt}\int_{C_i} q(x,t)dx = f(q(x_{i-1/2},t)) - f(q(x_{i+1/2},t)). \tag{43}$$

Integrating equation 43 in time from t_n to t_{n+1} and rearranging the resulting equation, we obtain

$$Q_i^{n+1} = Q_i^n - \frac{\Delta t}{\Delta x}(F_{i+1/2}^n - F_{i-1/2}^n), \tag{44}$$

where the cell average Q_i^n and the numerical flux $F_{i-1/2}^n$ are defined as

$$\begin{aligned} Q_i^n &\approx \frac{1}{\Delta x}\int_{C_i} q(x,t_n)dx, \\ F_{i-1/2}^n &\approx \frac{1}{\Delta t}\int_{t_n}^{t_{n+1}} f(q(x_{i-1/2},t))dt. \end{aligned} \tag{45}$$

Here, $\Delta x = x_{i+1/2} - x_{i-1/2}$ and $\Delta t = t_{n+1} - t_n$. Any numerical scheme that can be written in the form 44 are said to be a conservative scheme (LeVeque, 2002).

Lax-Friedrichs Scheme

The Lax-Friedrichs (LxF) scheme for a scalar conservation law is given by

$$Q_i^{n+1} = \frac{1}{2}(Q_{i-1}^n + Q_{i+1}^n) - \frac{\Delta t}{2\Delta x}\left[f(Q_{i+1}^n) - f(Q_{i-1}^n)\right], \tag{46}$$

When written in the form of conservative scheme 44, the numerical flux of the LxF scheme is

$$F_{i-1/2}^n = \frac{1}{2}\left[f(Q_{i-1}^n) + f(Q_i^n)\right] - \frac{\Delta x}{2\Delta t}(Q_i^n - Q_{i-1}^n), \tag{47}$$

The LxF scheme is first-order accurate in both time and space, and can be applied to both linear and nonlinear scalar hyperbolic problems.

First-Order Upwind Scheme

An upwind scheme uses the flow direction to determine how to approximate the spatial derivative terms. Let's use the linear advection equation 21 as an example. When $u > 0$, the conserved quantity q is advected from left to right so the spatial derivative $q_{,x}$ is approximated as

$$q_{,x} \approx \frac{q(x_i,t) - q(x_{i-1},t)}{\Delta x}. \tag{48}$$

Using this approximation, the linear advection equation becomes the semi-discrete scheme

$$\frac{dq_i(t)}{dt} + u\frac{q_i(t) - q_{i-1}(t)}{\Delta x} = 0, \tag{49}$$

where $q_i(t) = q(x_i,t)$. If the forward Euler method is used, we obtain the fully discrete scheme

$$q_i^{n+1} = q_i^n - \frac{u\Delta t}{\Delta x}(q_i^n - q_{i-1}^n). \tag{50}$$

Note that this is an upwind finite-difference scheme which becomes an upwind finite-volume scheme if q_i^n is thought of as the cell average Q_i^n.

Riemann Problem

The Riemann problem is composed of a hyperbolic PDE and an initial condition in the form of a piecewise-constant function with a discontinuity,

$$q_0(x) = \begin{cases} q_l, & x < 0, \\ q_r, & x > 0. \end{cases} \tag{51}$$

For the linear advection equation 21, the solution to the Riemann problem consists of a discontinuity $q_r - q_l$ propagating at speed u along the characteristic (LeVeque, 2002). In the nonlinear case, the discontinuity or shock wave must propagate at speed s satisfying the Rankine-Hugonoit condition, given by

$$s = \frac{f(q_r) - f(q_l)}{q_r - q_l}. \tag{52}$$

The numerical flux $F_{i-1/2}^n$ at the cell interface located at $x = x_{i-1/2}$ can be obtained by solving the local Riemann problem with $q_l = Q_{i-1}^n$ and $q_r = Q_i^n$ and $x_{i-1/2}$ as the origin of the local coordinate system. The solution to the local Riemann problem at $x_{i-1/2}$ from time t_n to t_{n+1}, denoted as $Q_{i-1/2}^{\downarrow}$, will be a constant for any value of jump discontinuity $q_r - q_l$.

Godunov Scheme

The Godunov scheme can be written in the flux-differencing form given in equation 44 with the numerical flux

$$F_{i-1/2}^n = f(Q_{i-1/2}^{\downarrow}) \tag{53}$$

For the linear advection equation, the numerical flux can also be written in the form

$$F_{i-1/2}^n = u^+ Q_{i-1}^n + u^- Q_i^n, \tag{54}$$

where u^+ and u^- are defined as

$$\begin{aligned} u^+ &\equiv max(u,0) \\ u^- &\equiv min(u,0) \end{aligned} \tag{55}$$

The upwind scheme can be rewritten in the wave-propagating form as

$$Q_i^{n+1} = Q_i^n - \frac{\Delta t}{\Delta x}(u^+ W_{i-1/2} + u^- W_{i+1/2}), \tag{56}$$

where the wave $W_{i-1/2}$ is defined as

$$W_{i-1/2} \equiv \Delta Q_{i-1/2}^n = Q_i^n - Q_{i-1}^n. \tag{57}$$

The first-order upwind scheme is equivalent to the Godunov's method in this linear case and is first-order accurate in both time and space.

Lax-Wendroff Scheme

The Lax-Wendroff method is a second-order accurate, central scheme that uses the PDE to be solved to replace time derivatives in the Taylor's series expansion. To clarify this statement, I will use the linear advection equation $q_{,t} + uq_{,x} = 0$ as an example.

The Taylor's expansion of $q(x, t_{n+1})$ is given by

$$q(x,t_{n+1}) = q(x,t_n) + q_{,t}(x,t_n)\Delta t + q_{,tt}(x,t_n)\frac{(\Delta t)^2}{2!} + \cdots \tag{58}$$

Using the advection equation, the first time derivative in equation 58 is replaced by $q_{,t} = -uq_{,x}$, and differentiating this with respect to time gives

$$q_{,tt} = -uq_{,xt} = -u(q_{,t})_{,x} = u^2 q_{,xx} \tag{59}$$

Equation 58 then becomes

$$\begin{aligned} q(x,t_{n+1}) = \; & q(x,t_n) - uq_{,x}(x,t_n)\Delta t + u^2 q_{,xx}(x,t_n)\frac{(\Delta t)^2}{2!} \\ & + O((\Delta t)^3) \end{aligned} \tag{60}$$

Replacing the spatial derivatives in equation 60 by central finite difference approximations yields the Lax-Wendroff scheme

$$\begin{aligned} Q_i^{n+1} = \; & Q_i^n - \frac{1}{2}C(Q_{i+1}^n - Q_{i-1}^n) \\ & + \frac{1}{2}C^2(Q_{i+1}^n - 2Q_i^n + Q_{i-1}^n) \end{aligned} \tag{61}$$

where $C = u\Delta t/\Delta x$ is the Courant number.

The corresponding flux function for the Lax-Wendroff method is given by

$$F_{i-1/2}^n = \frac{1}{2}u(Q_{i-1}^n + Q_i^n) - \frac{1}{2}\frac{\Delta t}{\Delta x}u^2(Q_i^n - Q_{i-1}^n). \tag{62}$$

Beam-Warming Scheme

While the Lax-Wendroff scheme is centered and second-order accurate, the Beam-Warming scheme is also second-order accurate but one-sided, given by

$$\begin{aligned} Q_i^{n+1} = \; & Q_i^n - \frac{1}{2}C(3Q_i^n - 4Q_{i-1}^n + Q_{i-2}^n) \\ & + \frac{1}{2}C^2(Q_i^n - 2Q_{i-1}^n + Q_{i-2}^n) \end{aligned} \tag{63}$$

The corresponding flux function for the Beam-Warming method is given by

$$F_{i-1/2}^n = uQ_{i-1}^n + \frac{u}{2}(1-C)(Q_{i-1}^n - Q_{i-2}^n). \tag{64}$$

Fromm Scheme

The Fromm scheme is second-order accurate and non-symmetric, given by

$$\begin{aligned} Q_i^{n+1} = \; & Q_i^n - \frac{1}{4}C(Q_{i+1}^n + 3Q_i^n - 5Q_{i-1}^n + Q_{i-2}^n) \\ & + \frac{1}{4}C^2(Q_{i+1}^n - Q_i^n - Q_{i-1}^n + Q_{i-2}^n) \end{aligned} \tag{65}$$

Piecewise-Linear-Reconstruction Schemes

The Godunov's method uses the cell average Q_i^{n+1} to reconstruct the solution function $q(x, t_{n+1})$ as a piecewise constant function. This results in a first-order accurate method. To improve the accuracy, a piecewise linear reconstruction can be used instead by introducing a slope σ in each cell. The resulting method is second-order accurate. For the linear advection equation, the piecewise-linear-reconstruction (PLR) scheme can be written as

$$\begin{aligned} Q_i^{n+1} = \; & Q_i^n - C(Q_i^n - Q_{i-1}^n) \\ & - \frac{1}{2}C(\Delta x - u\Delta t)(\sigma_i^n - \sigma_{i-1}^n), \end{aligned} \tag{66}$$

where σ_i^n is the slope in the ith cell.

The Lax-Wendroff, Beam-Warming, and Fromm schemes can be obtained using the slopes defined below.

$$\begin{aligned} \text{Lax-Wendroff}: \;\; \sigma_i^n &= \frac{Q_{i+1}^n - Q_i^n}{\Delta x} \\ \text{Beam-Warming}: \;\; \sigma_i^n &= \frac{Q_i^n - Q_{i-1}^n}{\Delta x} \\ \text{Fromm}: \;\; \sigma_i^n &= \frac{Q_{i+1}^n - Q_{i-1}^n}{2\Delta x} \end{aligned} \tag{67}$$

Total Variation

These second-order schemes are more accurate than the Lax-Friedichs and Godunov methods but also introduce spurious oscillations where the solution is discontinuous. A mathematical way to measure the oscillation in a function is to use the total variation defined for grid function Q and differentiable function q as

$$TV(Q) \equiv \sum_{i=-\infty}^{\infty} |Q_i - Q_{i-1}| \tag{68}$$

$$TV(q) \equiv \int_{-\infty}^{\infty} |q'(x)| dx \tag{69}$$

TVD Schemes

A numerical scheme is called total variation diminishing (TVD) if it satisfies the inequality (LeVeque, 2002)

$$TV(Q^{n+1}) \le TV(Q^n) \tag{70}$$

From this definition, all the previously mentioned second-order schemes are not TVD. The next section introduces second-order TVD schemes.

Slope-Limiter Schemes

Monotonic Upstream-centered Scheme for Conservation Laws (MUSCL) is a class of numerical methods originally developed by van Leer (1979). The MUSCL scheme is the second-order extension of the first-order Godunov's method by applying a slope-limiter method to the piecewise linear reconstruction of the conserved variable q so that the monotonicity of the solution is preserved. This results in a second-order TVD scheme.

Here three slope-limiter methods presented in LeVeque (2002) are given as follows.

minmod limiter:

$$\sigma_i^n = \text{minmod}\left(\frac{Q_i^n - Q_{i-1}^n}{\Delta x}, \frac{Q_i^n - Q_{i-1}^n}{\Delta x}\right), \tag{71}$$

where the minmod function is defined as

$$\text{minmod}(a,b) = \begin{cases} a \text{ if } |a| < |b| \text{ and } ab > 0, \\ b \text{ if } |b| < |a| \text{ and } ab > 0, \\ 0 \text{ if } ab \le 0 \end{cases} \tag{72}$$

superbee limiter:

$$\sigma_i^n = \text{maxmod}(\sigma_i^{(1)}, \sigma_i^{(2)}), \tag{73}$$

where the maxmod function is defined as

$$\text{maxmod}(a,b) = \begin{cases} b \text{ if } |a| < |b| \text{ and } ab > 0, \\ a \text{ if } |b| < |a| \text{ and } ab > 0, \\ 0 \text{ if } ab \le 0 \end{cases} \tag{74}$$

and

$$\begin{aligned} \sigma_i^{(1)} &= \text{minmod}\left[\left(\frac{Q_{i+1}^n - Q_i^n}{\Delta x}\right), 2\left(\frac{Q_i^n - Q_{i-1}^n}{\Delta x}\right)\right], \\ \sigma_i^{(2)} &= \text{minmod}\left[2\left(\frac{Q_{i+1}^n - Q_i^n}{\Delta x}\right), \left(\frac{Q_i^n - Q_{i-1}^n}{\Delta x}\right)\right]. \end{aligned} \tag{75}$$

Monotonized central-difference (MC) limiter:

$$\sigma_i^n = \text{minmod}\left[\left(\frac{Q_{i+1}^n - Q_{i-1}^n}{2\Delta x}\right), 2\left(\frac{Q_i^n - Q_{i-1}^n}{\Delta x}\right), 2\left(\frac{Q_{i+1}^n - Q_i^n}{\Delta x}\right)\right] \tag{76}$$

Flux-Limiter Schemes

The piecewise-linear-reconstruction scheme can be rewritten into the flux-differencing form (equation 44) for the linear advection equation 21 with flux

$$F_{i-1/2}^n = u^- Q_i^n + u^+ Q_{i-1}^n + \frac{1}{2}|u|(1 - |C|)\delta_{i-1/2}^n, \tag{77}$$

where

$$\delta_{i-1/2}^n = \phi(\theta_{i-1/2}^n)\Delta Q_{i-1/2}^n. \tag{78}$$

Here $\theta_{i-1/2}^n$ is defined as

$$\theta_{i-1/2}^n \equiv \frac{\Delta Q_{I-1/2}^n}{\Delta Q_{i-1/2}^n} \tag{79}$$

where the index I is

$$I = \begin{cases} i-1 & \text{if } u > 0, \\ i+1 & \text{if } u < 0. \end{cases} \tag{80}$$

All the methods mentioned previously can be implemented as a flux-limiter method with the flux function $\phi(\theta)$ set as follows.

$$\begin{aligned} \text{Godunov}: &\ \phi(\theta) = 0, \\ \text{Lax-Wendroff}: &\ \phi(\theta) = 1, \\ \text{Beam-Warming}: &\ \phi(\theta) = \theta, \\ \text{Fromm}: &\ \phi(\theta) = \frac{1}{2}(1+\theta), \\ \text{minmod}: &\ \phi(\theta) = \text{minmod}(1,\theta), \\ \text{superbee}: &\ \phi(\theta) = \max(0, \min(1, 2\theta), \min(2, \theta)), \\ \text{MC}: &\ \phi(\theta) = \max(0, \min((1+\theta)/2, 2, 2\theta)). \end{aligned} \tag{81}$$

TVD Schemes for Nonlinear Problems

In nonlinear problems, a discontinuity in the initial data could lead to either a shock or rarefaction wave as shown in Figure 1. In most cases, the solution to the Riemann problem at $x_{i-1/2}$ denoted by $Q_{i-1/2}^{\downarrow}$ (the value of solution along the dotted line in Figure 1) will be either Q_{i-1} or Q_i. Only in the case of a centered rarefaction wave, $Q_{i-1/2}^{\downarrow}$ must be determined by the value q_s satisfying the conditions $Q_{i-1} < q_s < Q_i$ and $f'(q_s) = 0$; this is called the stagnation point or sonic point (LeVeque, 2002). In summary, $Q_{i-1/2}^{\downarrow}$ is given by

$$Q_{i-1/2}^{\downarrow} = \begin{cases} Q_{i-1} & \text{if } Q_{i-1} = Q_i \text{ or } s = 0, \\ Q_{i-1} & \text{if } Q_{i-1} > q_s \text{ and } s > 0, \\ Q_i & \text{if } Q_i < q_s \text{ and } s < 0, \\ q_s & \text{if } Q_{i-1} < q_s < Q_i, \end{cases} \tag{82}$$

where $s = (f(Q_i) - f(Q_{i-1}))/(Q_i - Q_{i-1})$ is the shock speed satisfying the Rankine-Hugonoit jump condition. Then, we can use the flux-differencing scheme in equation 44 with $F_{i-1/2}^n = f(Q_{i-1/2}^{\downarrow})$.

In addition to the flux-differencing form, the Godunov scheme can also be written in the fluctuation form given by

$$Q_i^{n+1} = Q_i^n - \frac{\Delta t}{\Delta x}(A^+ \Delta Q_{i-1/2}^n + A^- \Delta Q_{i+1/2}^n), \tag{83}$$

where the fluctuations $A^{\pm}\Delta Q_{i-1/2}^n$ are defined by

$$\begin{aligned} A^+ \Delta Q_{i-1/2}^n &\equiv f(Q_i) - f(Q_{i-1/2}^{\downarrow}), \\ A^- \Delta Q_{i-1/2}^n &\equiv f(Q_{i-1/2}^{\downarrow}) - f(Q_{i-1}^n. \end{aligned} \tag{84}$$

The Godunov scheme in the fluctuation form can be further extended to a high-resolution method by adding a correction term as (LeVeque, 2002)

$$Q_i^{n+1} = Q_i^n - \frac{\Delta t}{\Delta x}(A^+\Delta Q_{i-1/2}^n + A^-\Delta Q_{i+1/2}^n) - \frac{\Delta t}{\Delta x}(\tilde{F}_{i+1/2}^n - \tilde{F}_{i-1/2}^n), \tag{85}$$

where the corrected flux $\tilde{F}_{i-1/2}^n$ is defined by

$$\tilde{F}_{i-1/2}^n = \frac{1}{2}|s_{i-1/2}|\left(1 - \frac{\Delta t}{\Delta x}|s_{i-1/2}|\right)\delta_{i-1/2}^n, \tag{86}$$

and the speed $s_{i-1/2}$ is defined by

$$s_{i-1/2} = \begin{cases} (f(Q_i) - f(Q_{i-1}))/(Q_i - Q_{i-1}) & \text{if } Q_{i-1} \neq Q_i, \\ f'(Q_i) & \text{if } Q_{i-1} = Q_i. \end{cases} \tag{87}$$

Godunov Method for Linear Systems

Given a linear system

$$q_{,t} + Aq_{,x} = 0, \tag{88}$$

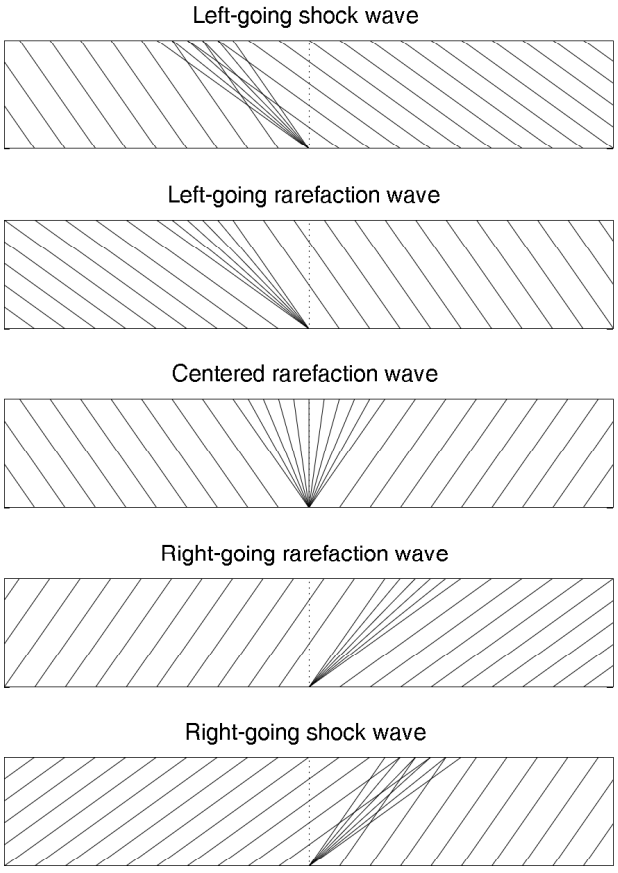


Figure 1: Patterns of characteristics corresponding to 5 initial piecewise-constant data with a jump discontinuity for the Burgers equation.

where q is an m-vector of conserved quantities and A is a constant coefficient matrix. If the system is hyperbolic, all the eigenvalues of A must be real and the system can be diagonalized by the similarity transformation

$$R^{-1}AR = \Lambda, \tag{89}$$

where R is the matrix whose columns are the right eigenvectors r_p, $p = 1, 2, ..., m$. If all the eigenvectors r_p are unit vector, R will be a diagonal matrix, i.e., $RR^T = R^TR = I$, where I is the identity matrix. By introducing the characteristic variable $w = R^{-1}q$, we obtain the diagonalized system

$$w_{,t} + \Lambda w_{,x} = 0, \tag{90}$$

which corresponds to m decoupled linear advection equations.

The Godunov method can be implemented for the linear system using the fluctuation form (equation 83) with A^+ and A^- defined by

$$A^+ = R\Lambda^+R^{-1}, \quad A^- = R\Lambda^-R^{-1}, \tag{91}$$

where

$$\Lambda^+ = \text{diagonal}(\lambda_p^+), \quad \Lambda^- = \text{diagonal}(\lambda_p^-). \tag{92}$$

Here λ_p^+ and λ_p^- are defined as

$$\lambda_p^+ \equiv \max(\lambda_p, 0), \quad \lambda_p^- \equiv \min(\lambda_p, 0). \tag{93}$$

High-Resolution Methods for Linear Systems

The correct flux $\tilde{F}_{i-1/2}^n$ for linear systems is

$$\tilde{F}_{i-1/2}^n = \frac{1}{2}|A|\left(1 - \frac{\Delta t}{\Delta x}|A|\right)\sum_{p=1}^m \tilde{\alpha}_{i-1/2}^p r^p, \tag{94}$$

where $|A| = A_+ - A^-$ and r^p is the p-th right eigenvector of A. Here $\tilde{\alpha}_{i-1/2}^p$ is the limited version of $\alpha_{i-1/2}^p = R^{-1}\Delta Q_{i-1/2}^n$, given by

$$\tilde{\alpha}_{i-1/2}^p = \alpha_{i-1/2}^p \phi(\theta_{i-1/2}^p), \tag{95}$$

where ϕ is one of the limiter functions given previously,

$$\theta_{i-1/2}^p = \frac{\alpha_{I-1/2}^p}{\alpha_{i-1/2}^p} \tag{96}$$

and the index I is

$$I = \begin{cases} i-1 & \text{if } \lambda^p > 0, \\ i+1 & \text{if } \lambda^p < 0. \end{cases} \tag{97}$$

Since $|A|r^p = |\lambda^p|r^p$, the corrected flux $\tilde{F}_{i-1/2}^p$ can also be written as

$$\tilde{F}_{i-1/2}^n = \frac{1}{2}\sum_{p=1}^m |\lambda^p|\left(1 - \frac{\Delta t}{\Delta x}|\lambda^p|\right)\tilde{\alpha}_{i-1/2}^p r^p. \tag{98}$$

Numerical Results

Linear Advection Equation

In this section, I apply the numerical schemes presented earlier to numerically solve the linear advection equation. The numerical calculations were performed in the domain $[0, 1]$ with the grid spacing $\Delta x = 0.01$ m, the Courant number $C = u\Delta t/\Delta x = 0.8$, the flow velocity $u = 1$ m/s, the time limit $T = 5$ s, and the initial condition

$$q(x,0) = \begin{cases} 1, & 0.6 \le x \le 0.8, \\ e^{-200(x-0.3)^2}, & \text{otherwise.} \end{cases} \tag{99}$$

The numerical results at the time limit shown in Figure 2 show the strong numerical dissipation property of the Lax-Friedichs and Godunov methods which are first-order accurate. There are small-amplitude oscillations in the Lax-Friedichs solution. Since the Godunov's method is TVD, its solution has no spurious oscillation. Among the second-order non-TVD methods, the Fromm method provides the most accurate result with small oscillation compared to that of the Lax-Wendroff and Beam-Warming methods. Among the second-order TVD methods, the MUSCL scheme with superbee limiter provides the most accurate result in both the smooth and discontinous regions.

Burgers Equation

In this section I applied the numerical schemes to solve the Burgers equation in various cases.

Rarefaction wave only: In the first case, the calculations were performed in the domain $[0, 1]$ with the grid spacing $\Delta x = 0.04$ m, the time sampling interval $\Delta t = 0.032$ s, the time limit $T = 0.3$ s, and the initial condition is

$$u(x,0) = \begin{cases} -1 & \text{if } x < 0.5, \\ 1 & \text{if } x > 0.5. \end{cases} \tag{100}$$

This initial condition will lead to a centered rarefaction wave propagating in both directions. The numerical results shown in Figure 3 show that, among the first-order methods, the Godunov method provides a more accurate result than the Lax-Friedichs method. For an explanation on the staircase pattern of the Lax-Friedichs method, see section 12.5 of LeVeque (2002). For the second-order methods, all methods provide comparably accurate results.

Shock wave only: In the second case, the calculations were performed in the domain $[0, 1]$ with the grid spacing $\Delta x = 0.01$ m, the time sampling interval $\Delta t = 0.008$ s, the time limit $T = 0.3$ s, and the initial condition is

$$u(x,0) = \begin{cases} 1 & \text{if } x < 0.5, \\ 0.1 & \text{if } x > 0.5. \end{cases} \tag{101}$$

This initial condition will lead to a shock wave propagating to the right with speed $s = 1.05$. The numerical results shown in Figure 4 show that all TVD methods and the Lax-Wendroff method can accurately capture the shock wave. However, the Lax-Wendroff method also gives rise to an upshoot at the discontinuity. In the case of the

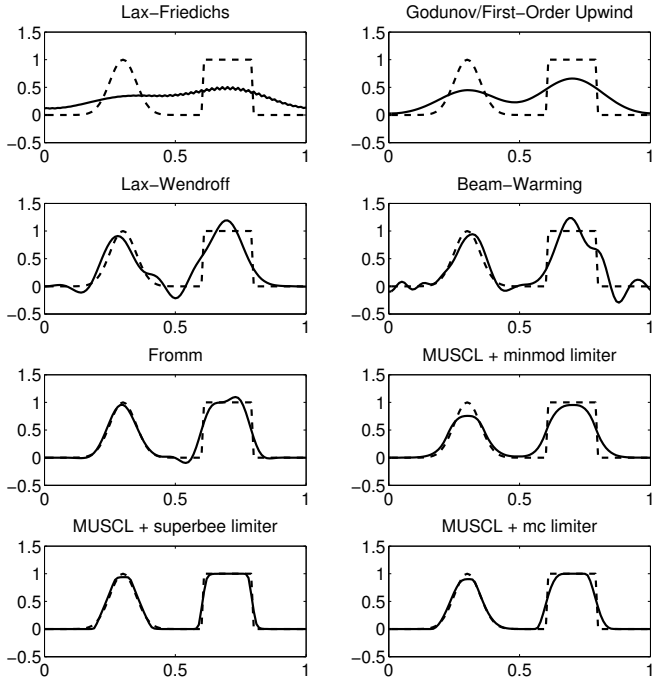


Figure 2: Numerical results to the linear advection equation using various methods. The dashed and solid lines represent the exact and numerical solutions, respectively.

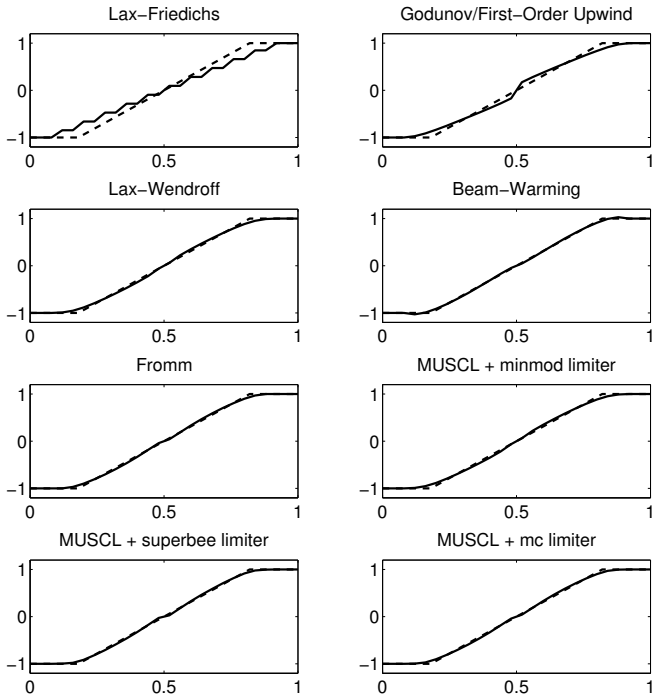


Figure 3: Numerical results to the Burgers equation in the case of a rarefaction wave only. The dashed and solid lines represent the exact and numerical solutions, respectively.

Beam-Warming and Fromm methods, there is an instability initiating from the discontinuity and propagating away in both directions so the result is missing in this region.

Shock and rarefaction waves: In the third case, the calculations were performed in the domain $[0, 1]$ with the grid spacing $\Delta x = 0.01$ m, the time sampling interval $\Delta t = 0.008$ s, the time limit $T = 0.5$ s, and the initial condition is

$$u(x, 0) = e^{-200(x-0.3)^2}. \tag{102}$$

According to the numerical results shown in Figure 5, only the Lax-Wendroff and Fromm methods provide solution with spurious oscillations while the Beam-Warming scheme provide a sharp result at the shock wave front even though it is not TVD. The Beam-Warming result is comparable to that of the MUSCL schemes. The Godunov method provides a slightly smooth result compared to that of the MUSCL schemes. The Lax-Friedichs method provides the smoothest result.

Acoustics Equations

In this section I applied the numerical schemes to solve the acoustics equations. The calculations were performed in the domain $[0, 1]$ with the grid spacing $\Delta x = 0.002$ m, the bulk modulus $K_0 = 0.1$, the density $\rho_0 = 1$, the fluid velocity $u_0 = 0$, the time sampling interval $\Delta t = 0.0032$ s, the time limit $T = 1$ s, and the initial conditions are

$$p(x, 0) = \begin{cases} 1 & \text{if } 0.5 < x < 0.6, \\ e^{-400(x-0.4)^2} & \text{otherwise,} \end{cases} \tag{103}$$
$$u(x, 0) = 0.$$

The numerical results shown in Figure 6 only include the results of the Godunov scheme, Lax-Wendroff scheme, and MUSCL schemes with midmod and superbee limiters. The Lax-Friedichs scheme gave quite an inaccurate result and so its result is not included. The Beam-Warming and Fromm schemes are unstable in this case. According to the results shown in Figure 6, the Godunov scheme provides a slightly smooth result compared to the other second-order methods.

1D Shallow-Water Equations

In this section I applied the numerical schemes to solve the 1D shallow-water equations. The calculations were performed in the domain $[-5, 5]$ with the grid spacing $\Delta x = 0.002$ m, the time sampling interval $\Delta t = 0.5\Delta x / \max(v(x, 0))$ where the gravitational wave speed $v(x, 0) = \sqrt{gh(x, 0)}$, the gravitational acceleration $g = 1$ m/s^2, the time limit $T = 3$ s, and the initial conditions for the wave height h and fluid particle velocity u are

$$\begin{aligned} h(x, 0) &= 1 + 0.4e^{-5x^2}, \\ u(x, 0) &= 0. \end{aligned} \tag{104}$$

In this case, the initial wave height contains a large-amplitude Gaussian hump which then later becomes shock waves propagating in both directions as shown in Figure 7. I only

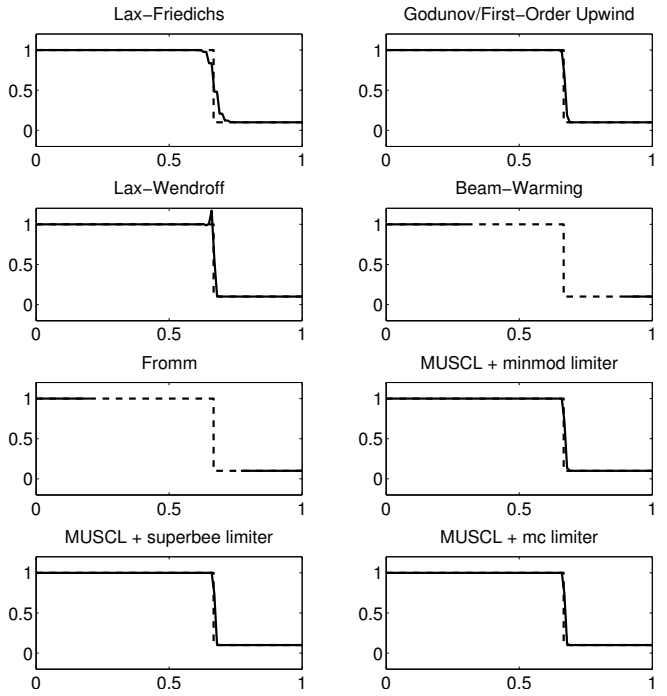


Figure 4: Numerical results to the Burgers equation in the case of a shock wave only. The dashed and solid lines represent the exact and numerical solutions, respectively.

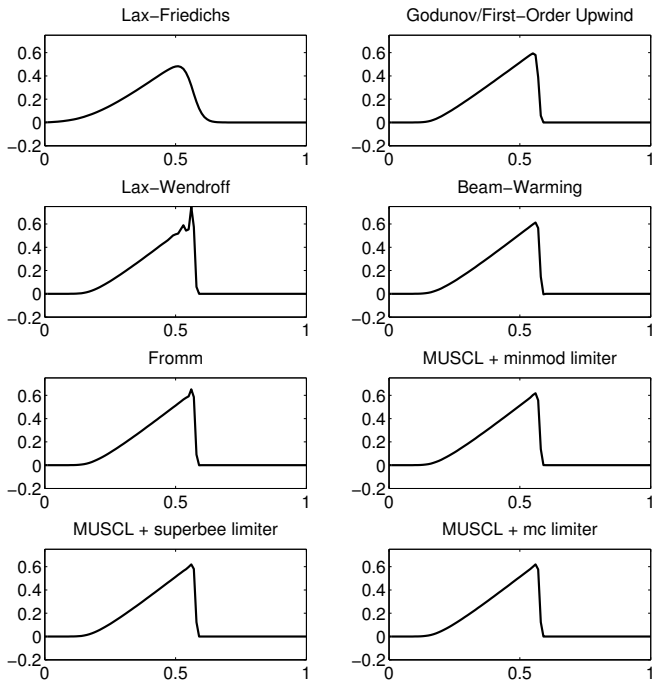


Figure 5: Numerical results to the Burgers equation in the case of shock and rarefaction waves.

applied some methods to this problem including the Godunov scheme, the Lax-Wendroff scheme, and the MUSCL schemes with minmod, superbee, and MC limiters. The Lax-Wendroff scheme provides a result with spurious oscillations at the shock wave fronts. All MUSCL schemes provide comparable results so only the results from minmod and superbee limiters are shown. The Godunov method provides a slightly smooth result at the shock wave fronts compared to the MUSCL schemes.

SUMMARY

In this report, I present a new PML formulation for the shallow-water equations and the theory of finite volume method. Due to the lack of a stable PML formulation for the nonlinear shallow-water equations, the proposed PML formulation could, hopefully, lead to a new research finding and, consequently, a publication. Using the finite volume method, I will numerically solve the proposed PML formulation in the next six-month period to assess its effectiveness and stability. In the section of finite volume method, I also present the numerical results obtained from solving various problems using the finite-volume methods presented in this report. The results show that the MUSCL schemes are very effective for solving problems containing discontinuitites in the form of shock and/or rarefaction waves.

REFERENCES

Abarbanel, S. and D. Gottleib, 1997, A mathematical analysis of the pml method: Journal of Computational Physics, **134**, 357–363.

———, 1998, On the construction and analysis of absorbing layers in cem: Applied Numerical Mathematics, **27**, 331–340.

Abarbanel, S., D. Gottleib, and J. S. Hesthaven, 1999, Well-posed perfectly matched layers for advective acoustics: Journal of Computational Physics, **154**, 266–283.

Abarbanel, S., D. Stanescu, and M. Y. Hussaini, 2003, Unsplit variables perfectly matched layers for the shallow water equations with coriolis forces: Computational Geosciences, **7**, 275–294.

Barucq, H., J. Diaz, and M. Tlemcani, 2010, New absorbing layers conditions for short water waves: Journal of Computational Physics, **229**, 58–72.

Bérenger, J. P., 1994, A perfectly matched layer for the absorption of electromagnetic waves: Journal of Computational Physics, **114**, 185–200.

Chew, W. C. and W. H. Weedon, 1994, A 3d perfectly matched medium from modified maxwell's equations with stretched coordinates: Microwave and Optical Technology Letters, **7**, 599–604.

Hesthaven, J. S., 1998, On the analysis and construction of perfectly matched layers for the linearized euler equations: Journal of Computational Physics, **142**, 129–147.

Hu, F. Q., 1996, On absorbing boundary conditions for linearized euler equations by a perfectly matched layer: Journal of Computational Physics, **129**, 201–219.

———, 2001, A stable, perfectly matched layer for linearized euler equations in unsplit physical variables: Journal of Computational Physics, **173**, 455–480.

LeVeque, R. J., 2002, Finite volume methods for hyper-

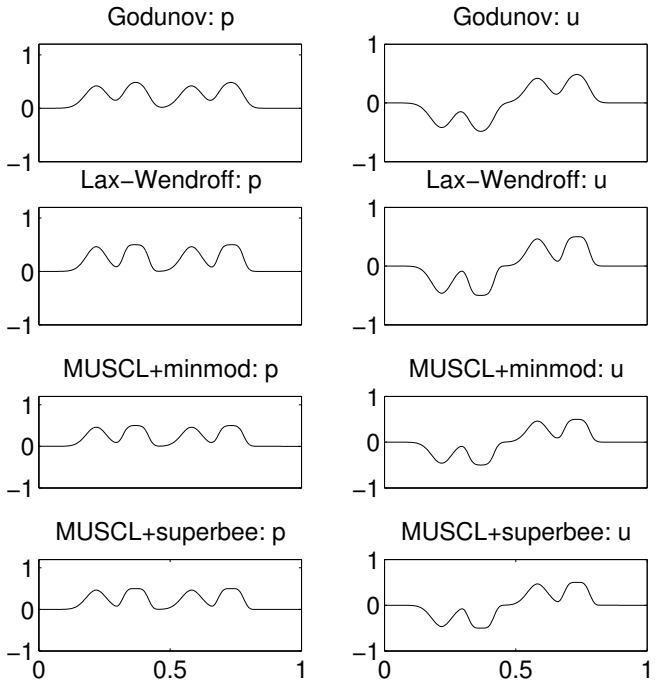


Figure 6: Numerical results to the acoustics equations.

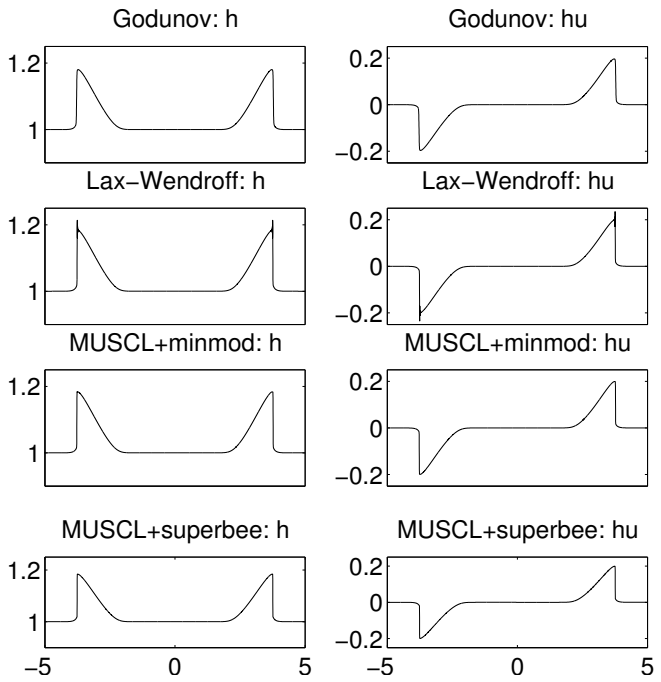


Figure 7: Numerical results to the 1D shallow-water equations.

bolic problems: Cambridge University Press.

Nataf, F., 2006, A new approach to perfectly matched layers for the linearized euler system: Journal of Computational Physics, **214**, 757–772.

Navon, I. M., B. Neta, and M. Y. Hussaini, 2004, A perfectly matched layer approach to the linearized shallow water equations models: Monthly Weather Review, **132**, 1369–1378.

van Leer, B., 1979, Towards the ultimate conservative difference scheme v. a second order sequel to godunov's method: Journal of Computational Physics, **32**, 101–136.

Comparative study of conventional full waveform inversion and envelope Inversion in the time domain

Apirujee Nakpathom and Chaiwoot Boonyasiriwat

ABSTRACT

Full waveform inversion is a method to reconstruct high resolution of Earth's structure from the observed data. Due to mostly relying on local optimization, this method does not preform well when the initial model is far from the true model. One way to overcome this challenge is to perform FWI at a very low frequency component in order to get the smooth version of the true model. However, sometimes, low frequency components are not available. Therefore, we have to use another alternative method. In this study, we will recover the smooth model by using envelope inversion method.

INTRODUCTION

In seismic exploration, seismic waves are generated and travel through the earth's subsurface. Where there are contrasts in acoustic impedance, the waves reflect and are recorded by receivers on the surface.

Full waveform inversion (FWI) is a method to estimate the subsurface properties by minimizing the difference between the observed and predicted data. This technique can reconstruct high-resolution models of physical properties of the subsurface.

However, in the case where the initial model is far from the actual model, FWI based on local optimization is often trapped in a local minimum. This problem can be solved in frequency-domain FWI by starting the inversion at a very low-frequency component of data which will give us a large-scale structure of the model. However, there is a case where low-frequency components are not available. Then, we have to find alternative methods to recover the large-scale of the model.

Wu et al. (2014) proposed an envelope inversion. This method is done by minimizing the envelope of the data instead of data itself. They showed that ultralow-frequency can be retrieved which leads to the smooth part of the model.

In this study, we will use the method of Wu et al. (2014) and compare the method against the conventional FWI. We will also study the inequality constraint updating and the gradient weighted by pseudo-Hessian.

THEORY

Conventional FWI in the time domain

In full waveform inversion, subsurface's properties, e.g. p-wave velocity, are reconstructed by minimizing the difference between the observed and calculated data. The calculated data are obtained using forward modeling.

Objective function

Least-square misfit function in the time domain is

$$f(\mathbf{m}) = \frac{1}{2}\sum_{\mathbf{s},\mathbf{r}}\int_0^T \left(p(t) - d(t)\right)^2 \mathrm{d}t \tag{1}$$

where d and p represent the observed and predicted data traces respectively. $\mathbf{m}$ is the model parameter which, in the acoustic case, is a p-wave velocity model.

Gradient expression

The gradient of $f(\mathbf{m})$ with respect to $\mathbf{m}$ can be written as

$$\frac{\partial f}{\partial \mathbf{m}} = \sum_{\mathbf{s},\mathbf{r}}\int_0^T \left(p(t) - d(t)\right)\frac{\partial p(t)}{\partial \mathbf{m}}\mathrm{d}t = \mathbf{J}^T\eta \tag{2}$$

- $\mathbf{J} = \dfrac{\partial p(t)}{\partial \mathbf{m}}$ is Jacobian or the Frechet derivative
- $\eta = p(t) - d(t)$ is the data residual

Equation 2 is simply computed by zero-lag crosscorrelation between forward-propagated source wavefield and backward propagated residual wavefield.

Envelope inversion

Envelope of a signal

Given a real signal $f(t)$

- $f_h(t) = \mathcal{H}\{f(t)\}$ is the Hilbert transform of $f(t)$ which can be obtained from

$$f_h(t) = \mathcal{H}\{f(t)\} = -\frac{1}{\pi}P\int_{-\infty}^{\infty}\frac{f(\tau)}{t-\tau}\mathrm{d}\tau \tag{3}$$

where P is a Cauchy principle value

- $\tilde{f}(t)$ is an analytic signal (a signal without amplitude at negative frequency), where

$$\tilde{f}(t) = f(t) + if_h(t)$$

- $e_f(t)$ is the envelope of $f(t)$ and can be obtained by

$$e_f(t) = \sqrt{f^2(t) + f_h^2(t)}$$

Objective function

The misfit function of envelope inversion:

$$f(\mathbf{m}) = \frac{1}{2}\sum_{\mathbf{s},\mathbf{r}}\int_0^T \left[e_p^k(t) - e_d^k(t)\right]^2 \mathrm{d}t = \frac{1}{2}\sum_{\mathbf{s},\mathbf{r}}\int_0^T E(t)^2 \mathrm{d}t$$

$$E(t) = \left[p^2(t) + p_h^2(t)\right]^k - \left[d^2(t) + d_h^2(t)\right] k$$

where

- k is the power of the envelope. According to Luo and Wu (2015), $k = 2$ can recover the long-wavelength part of the model better.

- $E(t)$ is called the instant envelope data residual

Gradient expression

The gradient can be obtained in the similar way to the equation 2, except that η now becomes

$$\eta = E(t)p(t)e_p^{k-2} - \mathcal{H}\left\{E(t)p_h(t)e_p^{k-2}\right\} \tag{4}$$

Inequality constraint updating

At each iteration of the inversion, we can calculate the gradient, ∇f^k, which gives a suitable updating direction to the model that makes the residual decreases. The estimated model at each iteration can be updated directly by finding a suitable step length, λ,

$$\mathbf{m}^{k+1} = \mathbf{m}^k + \delta\mathbf{m}^k \qquad \text{where : } \delta\mathbf{m}^k = -\lambda^k \nabla f^k \tag{5}$$

This updated model can be bounded by setting the element that is out of the bound as a value at the bound, as

$$m_i^{k+1} = \begin{cases} m_{\max}, & \text{if } m_i^{k+1} > m_{\max} \\ m_{\min}, & \text{if } m_i^{k+1} < m_{\min} \\ m_i^{k+1}, & \text{otherwise} \end{cases} \tag{6}$$

where i is the number of the elements in the model.

Another approach to update the model is to use Inequality constraint (Kim et al., 1999). First, we assume that our model is bounded by a and b (or $m_{\min}$ and $m_{\max}$),

$$a < m_i < b, \qquad i = 1, 2, \ldots, N \tag{7}$$

The new parameter x_i can be defined to include constraints as

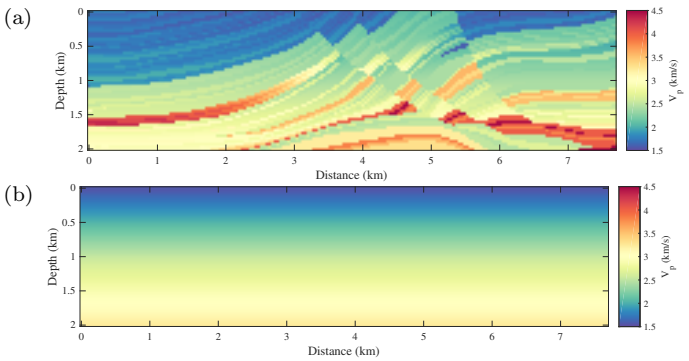


Figure 1: (a) True Marmousi model and (b) an initial model

$$x_i = \ln\left(\frac{m_i - a}{b - m_i}\right) \tag{8}$$

Then the perturbation of x is given by

$$\delta x_i^k = \frac{b-a}{\left(m_i^k - a\right)\left(b - m_i^k\right)}\delta m_i^k \tag{9}$$

The updated $\mathbf{x}$ is obtained by

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \delta\mathbf{x}^k \tag{10}$$

Hence, the updated model (m_i^{k+1}) can be given in term of δx_i as

$$m_i^{k+1} = \frac{a\left(b - m_i^k\right) + b\left(m_i^k - a\right)e^{\delta x_i^k}}{\left(b - m_i^k\right) + \left(m_i^k - a\right)e^{\delta x_i^k}} \tag{11}$$

SYNTHETIC DATA EXAMPLE

Experimental settings

The time-domain forward modeling is isotropic acoustic with constant density. The synthetic observed data is generated by a finite difference algorithm with a regular grid discretization as in the inversion process.

Figure 1 shows the true and initial models. The model size is 7680 × 2000 m with a grid spacing of 40 m. There are 91 sources and 187 receivers located at 40 m depth. The Ricker source is used in both observed and predicted data with central frequency at 10 Hz. The time-domain inversion was run for 100 iterations.

Numerical results

Figure 2 shows the results from 2 updating schemes. Figure 2a was updated using equation 6. Figure 2b was updated using equation 11. It can be clearly seen that the result from the inequality constraint is significantly better than the other scheme.

Figure 3 compares the inversion results from the gradient weighted without (3a) and with pseudo-Hessian (3b).

It shows that, by weighting the gradient with pseudo-Hessian, we can recover deeper part of the model.

Figure 4 shows the inversion results from different data sets. Both results used the same setting expect that Figure 4b is obtained from the data set generated by a staggered grid finite difference. This experiment shows that time-domain FWI is not robust when it comes to differences in the numerical method used to generate the synthetic data. In the previous experience, results from frequency-domain FWI are less suffered in this case.

Figure 5 compares the inversion results at the 5^{th} iteration of (5a) conventional FWI and EI with (5b) $k = 1$, (5c) $k = 2$, and (5d) $k = 3$. It can be clearly seen that a higher p gives a smoother result. However, these are not the results that we expected or as are showed in the paper (Wu et al., 2014). Their result with $k = 2$ shows much smoother results of the large-scale structure.

Figure 6 shows the final results at 100^{th} iteration of conventional FWI with the initial models from Figure 5.

SUMMARY

By using the inequality constraint updating, we can recover significantly better results. The pseudo-Hessian helps with recovering the deeper part of the structure. Our current FWI code is not robust when it comes to inversion the data generated with difference numerical schemes.

The envelope inversion that we implemented by following Wu et al. (2014) does not give the results that we expected. The alternative way is needed to find the way to recover large-scale structure.

REFERENCES

Kim, H. J., Y. Song, and K. H. Lee, 1999, Inequality constraint in least-squares inversion of geophysical data: Earth, Planets and Space, **51**, 255–259.

Luo, J., and R.-S. Wu, 2015, Seismic envelope inversion: reduction of local minima and noise resistance: Geophysical Prospecting, **63**, 597–614.

Wu, R.-S., J. Luo, and B. Wu, 2014, Seismic envelope inversion and modulation signal model: GEOPHYSICS, **79**, WA13–WA24.

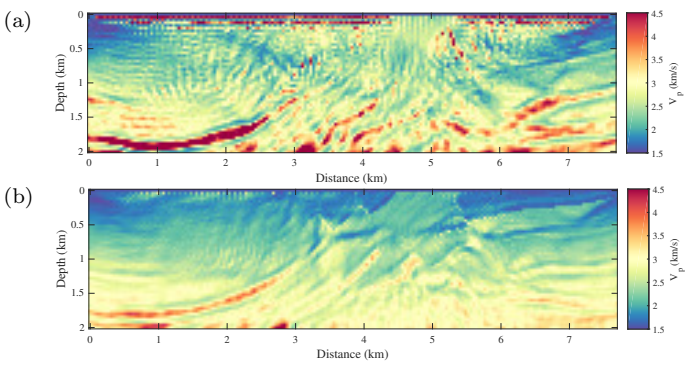


Figure 2: Inversion results obtained by (a) the cut-off updating and (b) the inequality constraint method

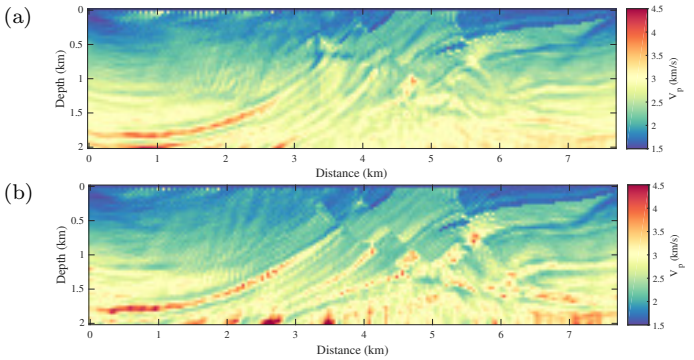


Figure 3: Inversion results from FWI (a) without and (b) with pseudo-Hessian weighting

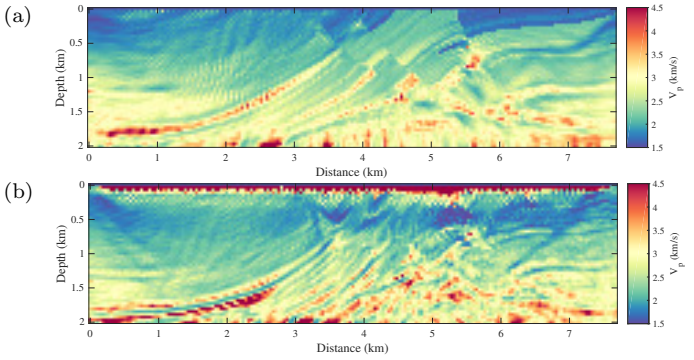


Figure 4: Inversion results from (a) a regular grid data set and (b) a staggered grid data set

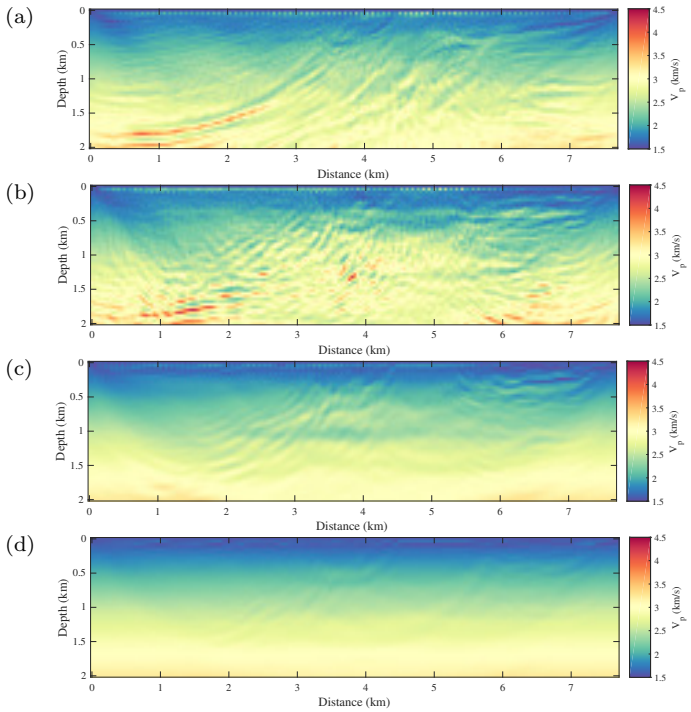


Figure 5: Inversion results at 5^{th} iteration of (a) conventional FWI and EI with (b) $k = 1$, (c) $k = 2$, and (d) $k = 3$

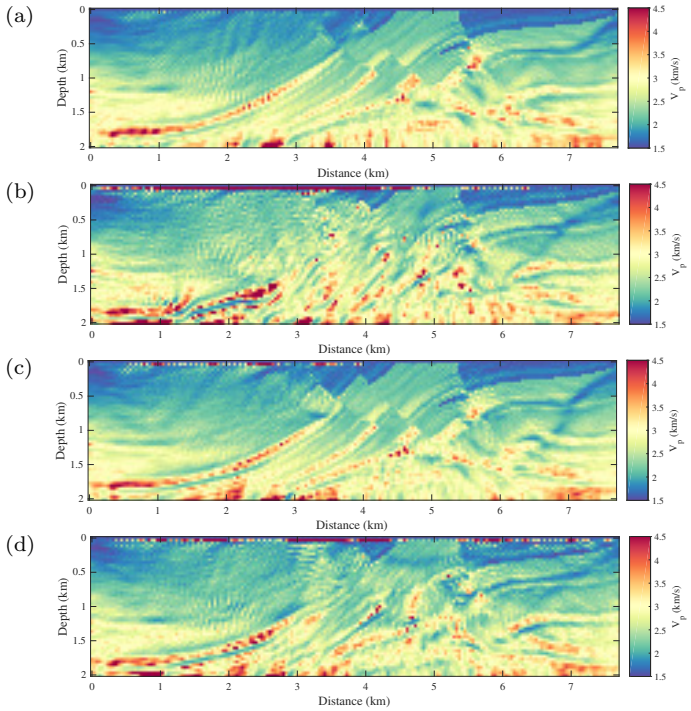


Figure 6: Inversion results at 100^{th} iteration from FWI with the initial models in figure 4

A web-based tool for rectifying the misconception on using Bernoulli's principle to explain airfoil lift

Harirak Intarak and Chaiwoot Boonyasiriwat

ABSTRACT

In this work, it aims to rectify the misconception of the usage of Bernoulli's principle. The common mistake is when we explain about the airfoil lift, we usually describe the airfoil lift by using Bernoulli's principle and we add some concepts that it is not true in physics. This misconception maybe found in many old textbooks (but for the latest they don't) or student's understanding.Therefore this work will rectify the students misconception by using the web-based tool using the HTML and Javascript to build learning system and we hope that this system will help the student understand more and correctly about Bernoulli's principle.

INTRODUCTION

Usually when we talk about why the airfoil produces lift (Center, 2017), we ask for the Bernoulli's principle. The explanation is about when air particle flow it will split at the leading edge and must be travelled to trailing edge at the same time and also the upper part of the airfoil has more distance compare to the lower part, so the air particle in the upper part must have the velocity greater than lower one. Hence from the Bernoulli's principle which state that when the velocity is larger it imply the pressure must be lower, we can conclude that when there is pressure difference it cause force occurred which in this case point upward and we called "Lift force", see Figure 1.

So far this explanation is wrong. If we know about the streamline, we should know that this is wrong since it is NOT the same streamline we cannot use that Bernoulli's principle to conclude that. Moreover if we consider at this explanation we can ask why the air particle split and must arrive at the same time.

Consider another explanation of using Bernoulli's principle (Babinsky, 2003). The Bernoulli's principle is often demonstrate by blowing a piece of paper held between both hand, see Figure 2.

The result when we blow that paper is that paper will rise up. And explain by referring to Bernoulli's principle which is when we blow upper surface the average velocity of the upper surface will greater than lower surface and again from Bernoulli's principle the upper surface will have lower pressure. From this pressure difference it cause the force that will rise that paper up. But if we try hold the paper vertically, and we blow it at one side, we will not see paper that be pushed. So this is wrong explanation of using Bernoulli's principle again.

So analogous explanation, we cannot explain the airfoil lift in that way. Also if we explain in that way, there are two things that is not true. The first one is the distance argument, as we can see from the airfoil there is difference in distance between upper and lower. But that is not necessary condition to generate lift force. Imagine sailboat, it also generates side force to sail whether this distance between both side is the same. And also there is some airfoil that is symmetrical but it still lift. The second thing is the equal time argument, as we said earlier that why air particle should split and arrive at the same time. So there is the experiment that verify this statement is true or not by recording smoke wind tunnel experiment of airfoil (Bastianello, 2013). See Figure 3.

From Figure 3, we see that at the end of airfoil there is no meeting at all and also the upper part of airfoil air particle travel faster.

If we still want to explain the airfoil lift by Bernoulli's

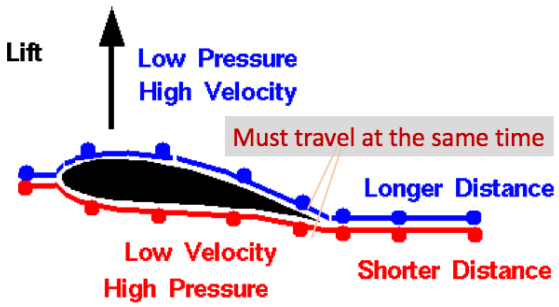


Figure 1: Explanation of airfoil lift

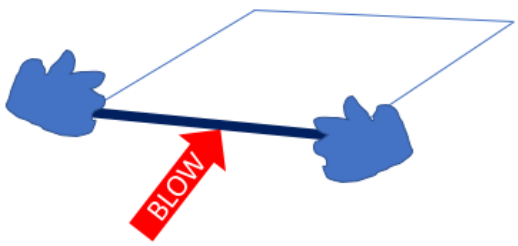


Figure 2: Demonstation of Bernoulli's principle

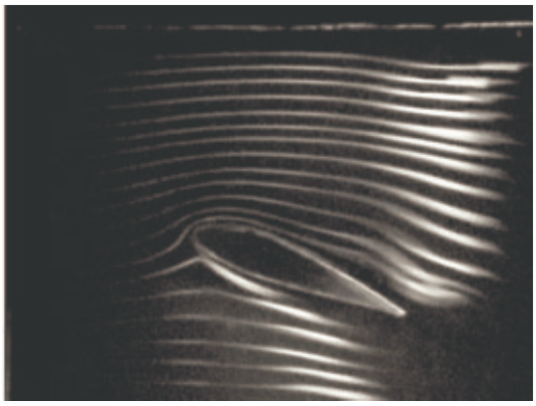


Figure 3: Wind tunnel flow along airfoil

principle, there is other approach which is the Bernoulli's principle across the streamline. The derivation isn't difficult but still require some mathematical to obtain the equation. This is not for high school students.

The easiest way to describe lift force is the Newton's Laws. We can say that when the air flow passes the airfoil and if airfoil is inclined so it turning the flow direction. Since there is change in direction will cause force according to the Newton's Second Law. And from Newton's third law, when the airfoil force the air moving downward, the air must exert force moving upward to the airfoil. The resulting force is the lift force.

METHODS

The method to make the tool for rectifying the misconception of Bernoulli's principle is base on the potential theory (E.L.Houghton, 2013). Consider the momentum equation in 2D which we assume the Reynold number is very high, so the viscosity is small we obtain the Euler's equations

$$\rho\left(\frac{\partial u}{\partial t}+u\frac{\partial u}{\partial x}+v\frac{\partial u}{\partial y}\right)=\rho g_x-\frac{\partial p}{\partial x}$$
$$\rho\left(\frac{\partial v}{\partial t}+u\frac{\partial v}{\partial x}+v\frac{\partial v}{\partial y}\right)=\rho g_y-\frac{\partial p}{\partial y}$$

And since the boundary layer is so small, we can assume that the transverse velocity is neglected, which means there is no vorticity and we can think of the steady flow. And also neglected the gravity, we obtained

$$\rho\left(u\frac{\partial u}{\partial x}+v\frac{\partial u}{\partial y}\right)=-\frac{\partial p}{\partial x}$$
$$\rho\left(u\frac{\partial v}{\partial x}+v\frac{\partial v}{\partial y}\right)=-\frac{\partial p}{\partial y}$$

Consider in x-direction, and rearrange equation

$$\frac{\partial}{\partial x}\frac{u^2}{2}+v\frac{\partial u}{\partial y}+\frac{1}{\rho}\frac{\partial p}{\partial x}+\frac{\partial}{\partial x}\frac{v^2}{2}-\frac{\partial}{\partial x}\frac{v^2}{2}=0$$
$$\frac{\partial}{\partial x}\left[\frac{u^2}{2}+\frac{v^2}{2}+\frac{p}{\rho}\right]=v\left[\frac{\partial v}{\partial x}-\frac{\partial u}{\partial y}\right]$$

And we say that it is no vorticity, therefore the last term is zero. Then we obtain the Bernoulli's equation that can use across streamline.

$$\frac{\partial}{\partial x}\left(\frac{u^2}{2}+\frac{v^2}{2}+\frac{p}{\rho}\right)=0$$
$$\frac{u^2}{2}+\frac{v^2}{2}+\frac{p}{\rho}=\text{const} \quad (1)$$

And from this we can assume that at the upstream it has velocity equal U_0 and pressure equal P_0. From equation 1, we get that

$$\frac{u^2}{2}+\frac{v^2}{2}+\frac{p}{\rho}=\frac{U^2}{2}+\frac{P_0}{2}$$

And we can define the dimensionless pressure coefficient to use to find the pressure at each side of airfoil later.

$$C_p=\frac{p-p_0}{\frac{1}{2}\rho U^2}=1-\frac{(u^2+v^2)}{U^2}$$

For the potential theory, since we assume that the flow is irrotational flow we can define velocity potential (ϕ)

$$\nabla\times u=0$$
$$\nabla\times(\nabla\phi)=0$$
$$u=\frac{\partial\phi}{\partial x},\quad v=\frac{\partial\phi}{\partial y}$$

And from the continuity equation, we obtain the Laplace equation as equation 2 and 3.

$$\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}=0$$
$$\frac{\partial^2\phi}{\partial x^2}+\frac{\partial^2\phi}{\partial y^2}=0 \quad (2)$$

Also the same for stream function ($u=\frac{\partial\psi}{\partial y}$, $v=-\frac{\partial\psi}{\partial x}$)

$$\frac{\partial^2\psi}{\partial x^2}+\frac{\partial^2\psi}{\partial y^2}=0 \quad (3)$$

From this we can find the analytic solution which the solution can be obtained by superposition of the particular solution.

Since we will consider the flow around cylinder and later we will transform the flow around cylinder to the flow around the airfoil. Therefore, the flow around the cylinder consists of the uniform flow solution plus the doublet solution.

$$\phi=Ur\cos\theta+\frac{K\cos\theta}{r}$$
$$\psi=ur\sin\theta+\frac{K\sin\theta}{r}$$

But in this case there is no lift force, so we should add the

vortex at the origin to give the lift.

$$\phi = Ur\cos\theta + \frac{K\cos\theta}{r} + \frac{\Gamma\theta}{2\pi}$$
$$\psi = Ur\sin\theta + \frac{K\sin\theta}{r} - \frac{\Gamma}{4\pi}\ln r$$

From now we will find each velocity component (let define $R = \sqrt{K/U}$)

$$u_r = \frac{\partial\phi}{\partial r} = \left[1 - \frac{R^2}{r^2}\right] U\cos\theta$$
$$u_\theta = \frac{\partial\phi}{r\partial\theta} = -\left(1 + \frac{R^2}{r^2}\right) U\sin\theta + \frac{\Gamma}{2\pi r}$$

And we have the stagnation point around the cylinder. Let say at $r = R$ it give $u_r = 0$, from this we can find the pressure by

$$p - p_0 = \frac{1}{2}\rho\left[U^2 - \left(-2U\sin\theta + \frac{\Gamma}{2\pi R}\right)^2\right]$$

Then we integrate around the cylinder to obtain the lift force

$$L = \int \vec{p}\cdot d\vec{s} = -\int_0^{2\pi} p\sin\theta R d\theta = -\rho U\Gamma$$

As you can see from this if there is no induced vorticity ($\Gamma = 0$) there is no lift.

And also from $u_\theta = 0$

$$0 = -\left(1 + \frac{R^2}{r^2}\right) U\sin\theta + \frac{\Gamma}{2\pi r}$$
$$\sin\theta = \frac{\Gamma}{4\pi RU}$$

We will define this angle as $\theta \equiv \alpha$, which is the angle that deviate from normal line, see Figure 4. Later we will transform it to airfoil, it will be called as "attack angle". So we can find lift by

$$\Gamma = 4\pi RU\sin\alpha$$
$$L = -\rho U\Gamma = -4\pi RU^2\sin\alpha$$

And we define the dimensionless lift coefficient

$$C_L = \frac{L}{\frac{1}{2}\rho U^2 R} = 8\pi\sin\alpha$$

Now after we obtain all information about the flow around cylinder, we can transform it to be flow around airfoil by using the Jowkouski's transformation. The Jowkouski's transformation is the transformation in complex plane by this formula

$$W = Z + \frac{1}{Z} \tag{4}$$

Where W is transformed complex plane, and Z is complex plane which we want to transform. Before that we must change a little bit of some variable which we defined above. From the transformation the length of the airfoil will be four times of the radius of cylinder ($C = 4R$). So the lift coefficient must be edited.

$$C_L = \frac{L}{\frac{1}{2}\rho U^2 C} = 2\pi\sin\alpha$$

RESULTS

This is the results after using Jowkouski's transformation. There are many ways to apply the Jowkouski's transformation. The first one is using

$$W = Z + \frac{1}{Z}$$

The result is in Figure 5. This transformation cylinder to ellipse.

For airfoil-like there must be some edit to the formula

$$W = Z + (-0.15) + \frac{1}{Z + (-0.15)}$$

This will add the real component of the transformation and give this result in the Figure 6.

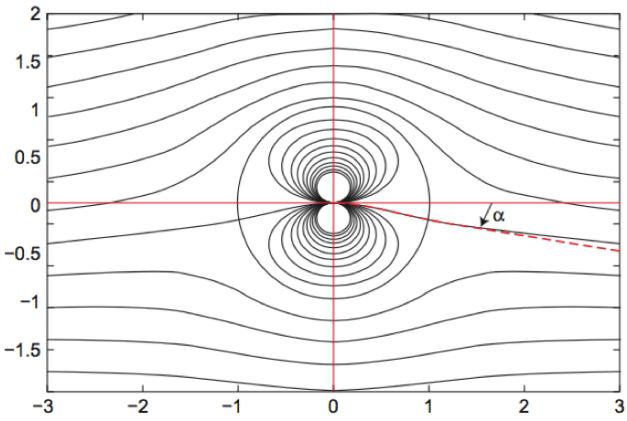


Figure 4: The angle which deviate from normal line

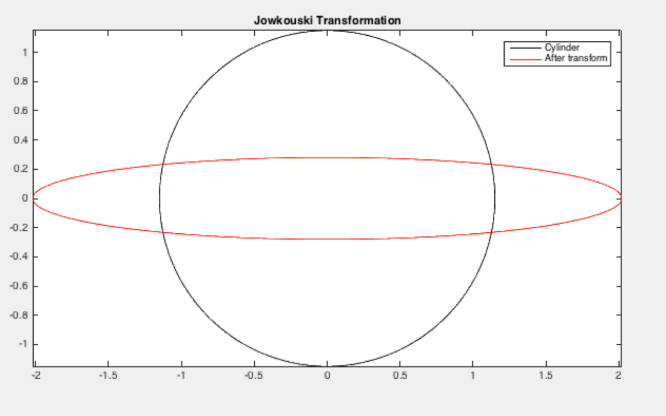


Figure 5: Transformation from cylinder to ellipse

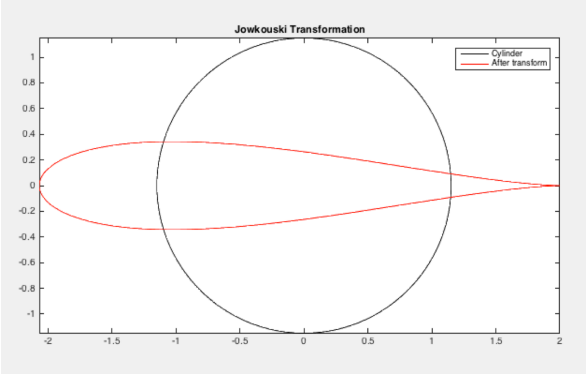


Figure 6: Transformation from cylinder to airfoil-like

For more realistic we should add some imaginary term like this

$$W = Z + (-0.15 + 0.15i) + \frac{1}{Z + (-0.15 + 0.15i)}$$

The result is in Figure 7. Note that this number can be vary.

After we can transform the shape to airfoil, we can transform the flow. So here are the Figure of MATLAB program that plot contour of the flow. The flow around cylinder is in Figure 8. The flow around cylinder with vortex added is in Figure 9. And in Figure 11 is the flow around the airfoil which come from transformation from Figure 10 by Jowkouski's transformation.

DISCUSSION

And after we can do everything about this transformation. Now we can move forward to port this MATLAB code to be what we aim for that is web-based tool. Now for the first time I intend to use WebGL to make the graphic look nicer. The problem is that how can we plot the contour. So there is the method called "Marching Method" which in 2D it called "Marching Square" (Wikipedia, 2017). It state that we can plot the isosurface which we define, for example $f(x, y) = 3$. Consider in square grid. See Figure 12. Since there are 4 points, there are 2^4 possible ways as see in the Figure. Hence from that we can plot the contour of any interested function.

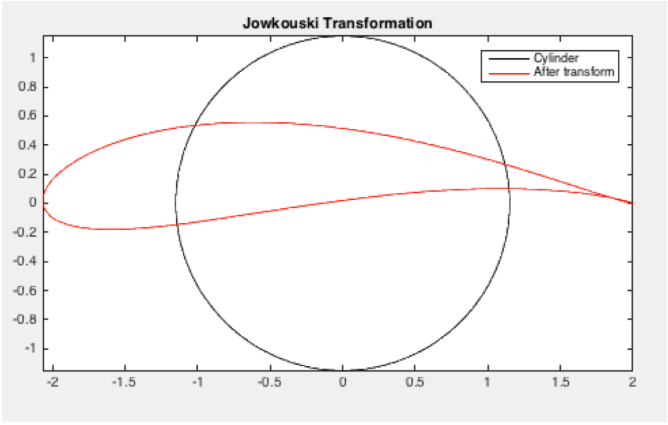


Figure 7: Transformation from cylinder to realistic airfoil-like

As you can see from these Figure 13-18 are the result after using WebGL to plot contour.

But there is a problem that the program cannot be run in some computer. So I have changed from WebGL to 2D canvas. And I found that there is problem that I cannot overcome which is it run slowly when I try to make smooth contour curve, see Figure(19). To be compromise the speed of calculation and smoothness of curve, I look for the way that we can make curve more smoothness without calculation so I ended up with give Marching Square linear interpolation but it still does not work.

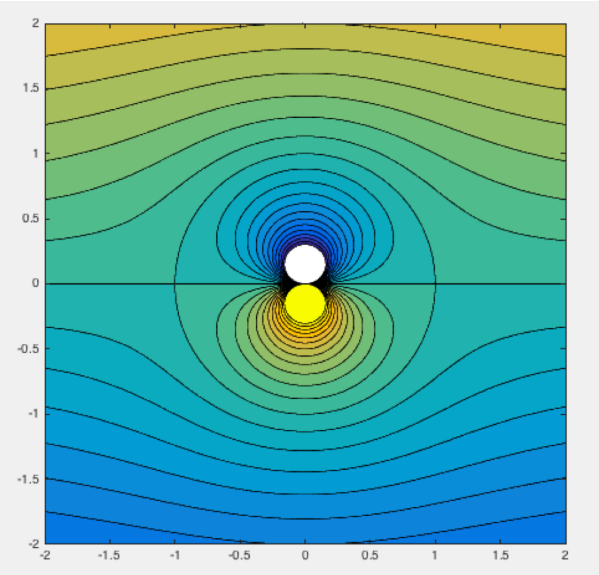


Figure 8: Flow around cylinder

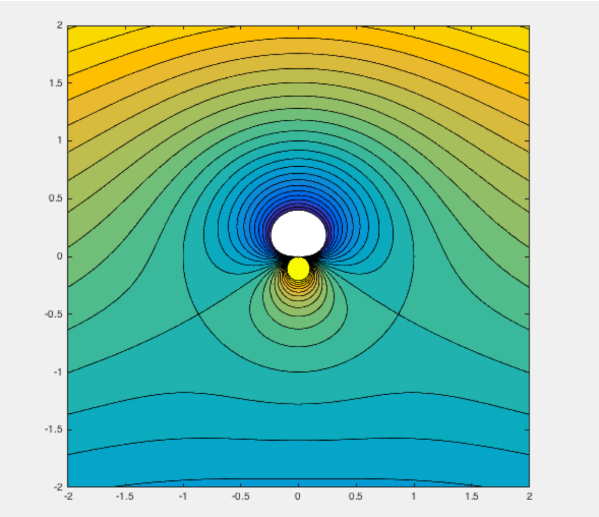


Figure 9: Flow around cylinder with vortex

CONCLUSION

So far I have created the webpage that contain the flow around the airfoil and can be vary each of parameter. This flow pattern come from plotting the contour line of the complex function define by potential theory. Then transform it to be the flow around the airfoil. But it still have some problem, such as slowness of the program when using marching square, and the lack some introduction to be what we called web-based learning tool. So I will work out for those later.

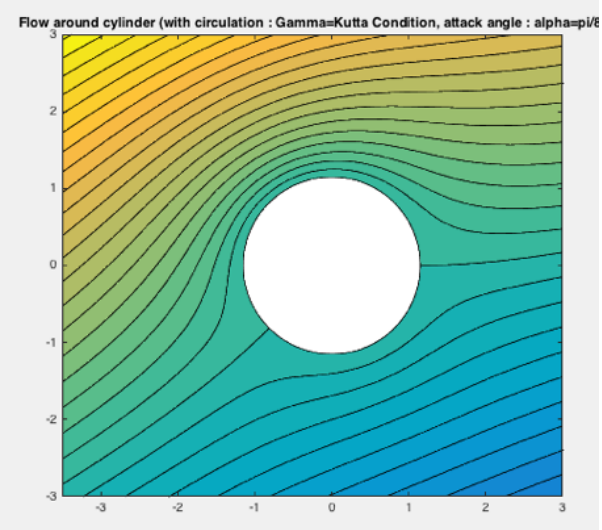


Figure 10: Flow around cylinder

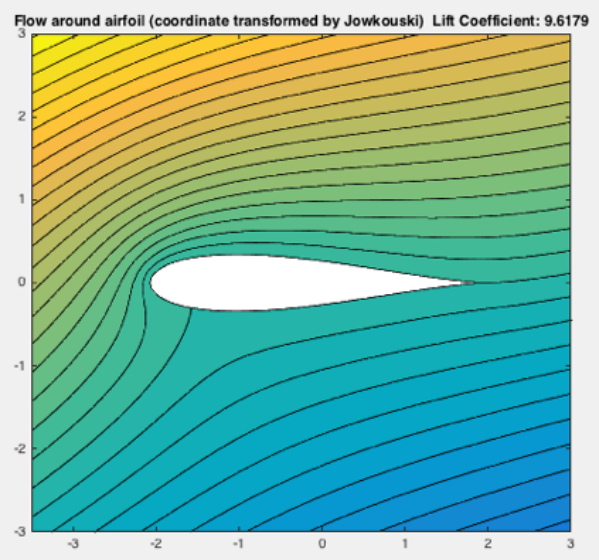


Figure 11: Flow around airfoil

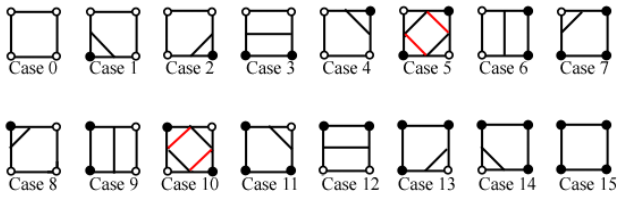


Figure 12: Marching Square Method

Figure 13: Flow of flat plate with no attack angle

Figure 14: Flow of flat plate with attack angle

Figure 15: Flow of ellipse airfoil with no attack angle

Figure 16: Flow of ellipse airfoil with attack angle

ACKNOWLEDGMENTS

I want to thanks MCSC group to give me advice for this work.

REFERENCES

Babinsky, H., 2003, How do wings work?: **38**, 497–503.

Bastianello, F., 2013, Lift generation, some misconceptions and truths about lift: Young Scientists Journal, **13**.

Center, G. R., 2017, Incorrect theory: `https://www.grc.nasa.gov/www/k-12/airplane/wrong1.html`. (Accessed : 2017-08-24).

E.L.Houghton, 2013, Aerodynamics for engineering students: Elsevier Publishing, 149–207.

Wikipedia, 2017, Marching square: `https://en.wikipedia.org/wiki/Marching_squares`. (Accessed : 2017-08-24).

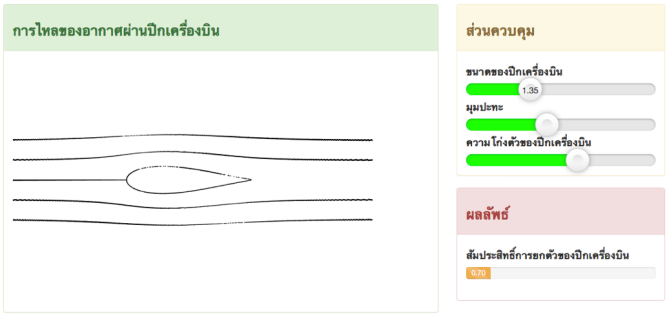


Figure 17: Flow of symmetrical airfoil with no attack angle

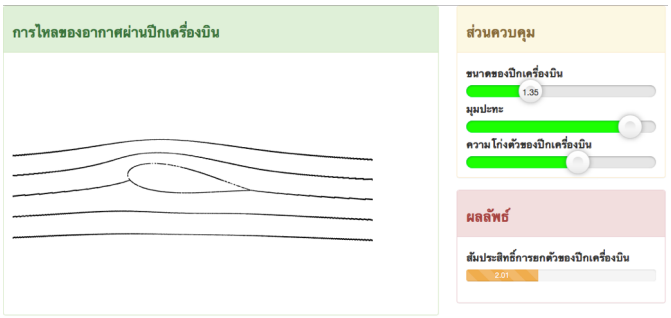


Figure 18: Flow of symmetrical airfoil with attack angle

Figure 19: Smoothness of contour curve when the program run quickly

An optimized finite-difference scheme based on least-squares optimization

Panithan Somboon, Wisart Thongyoy and Chaiwoot Boonyasiriwat

ABSTRACT

The conventional finite-difference (FD) method provides a numerical dispersion error which disperses the shape of wavelet and cause some errors in numerical solution. To improve the finite difference scheme, we optimize the coefficients of the finite-difference operator using least-square optimization and a given error limitation. We selected the error limitation to be 0.0001 which gives a comparable result between theoretical analyzes and numerical experiments. The optimized FD operator increases the wavenumber coverage and the accuracy compared to the convention FD methods. The optimized FD methods have the same order of accuracy as higher-order unoptimized FD methods this means that the computational cost can be reduced by the optimized FD.

INTRODUCTION

The finite-difference (FD) schemes are widely used in seismic wave modeling. The conventional FD operator usually creates numerical dispersion which cause some errors in the numerical solution especially in the presence of high-frequency components or when a coarse grid is used. In the large-scale models, it is impractical to use a fine grid due to the large computational cost. The numerical dispersion even causes a serious problem so we need to exclude high-frequency components to be able to reduce the dispersion but the risk of doing so is to achieve a lower resolutions.

In this report, we tried to study the FD methods to reduce the numerical dispersion following Thongyoy (2015). First, we construct an objective functions by transforming a finite difference approximation into the wavenumber domain. Then, we set the error limitation to be 0.0001 which is proposed by Zhang (2013). Finally, we apply the least-squares optimization to search for the best satisfied FD coefficients corresponding to the objective function.

SPATIAL DERIVATIVE IN THE WAVENUMBER DOMAIN

The conventional FD operator for the m-order spatial differentiation of function $f(x)$ is

$$\frac{d^m f(x)}{dx^m} \approx \frac{1}{\Delta x^m} \sum_{n=-N/2}^{N/2} b_n f_n \tag{1}$$

where N is the order of approximation, Δx is the spatial discretization among x , $f_n = f(x + n\Delta x)$, and b_n are constant FD coefficients which we want to optimize. We transform the spatial derivative into the wavenumber domain as follows

$$\frac{d^m f(x)}{dx^m} \leftrightarrow (ik_x)^m F(k_x) \tag{2}$$

where k_x is wavenumber. Next, equation (2) can be written in the wavenumber form for $m = 1$ and $m = 2$, respectively, as follows (see Appendix A)

$$ik_x \approx \frac{i}{\Delta x} \sum_{n=-N/2}^{N/2} b_n \sin(k_x n\Delta x), \quad \text{for m} = 1 \tag{3}$$

$$-k_x^2 \approx \frac{1}{\Delta x^2} \sum_{n=-N/2}^{N/2} b_n \cos(k_x n\Delta x), \quad \text{for m} = 2 \tag{4}$$

OBJECTIVE FUNCTIONS

We need to optimize coefficients b_n in an interested range of wavenumbers. We directly optimize the coefficients by examining the 2-norm of an error between the optimized FD operator in the wavenumber domain and the analytical wavenumber, and we apply the least-squares algorithm to optimize the following objective functions:

$$E_{m=1} = \sum_{i=0}^{c} \frac{1}{2} \left(k_{x,i}\Delta x - \sum_{n=-N/2}^{N/2} b_n \sin(k_{x,i} n\Delta x) \right)^2 \tag{5}$$

$$E_{m=2} = \sum_{i=0}^{c} \frac{1}{2} \left(-k_{x,i}^2\Delta x^2 - \sum_{n=-N/2}^{N/2} b_n \cos(k_{x,i} n\Delta x) \right)^2 \tag{6}$$

where $k_{x,c}$ is the maximum value of interested wavenumber. The coefficients will be minimized in range of $k_x(k_{x,0} \rightarrow k_{x,c})$, for any signal, the maximum wavenumber that can be detected is the Nyquist wavenumber which is equal to $k_{x,max} = \pi/\Delta x$, and the minimum wavenumber is $k_{x,min} = 0$, thus $k_x \in [0, \pi/\Delta x]$. We defined a variable $k = k_x\Delta x$, so $k \in [0, \pi]$ then the objective functions can be written as

$$E_{m=1} = \sum_{i=0}^{c} \frac{1}{2} \left(k_i \Delta x - \sum_{n=-N/2}^{N/2} b_n \sin(k_i n) \right)^2 \quad (7)$$

$$E_{m=2} = \sum_{i=0}^{c} \frac{1}{2} \left(-k_i^2 \Delta x^2 - \sum_{n=-N/2}^{N/2} b_n \cos(k_i n) \right)^2 \quad (8)$$

Minimizing the objective functions with respect to the coefficients b_n leads to a linear system which can be exactly solved. Two criteria are based on the conventional FD are used to constrain b_n: (1) the coefficients should be real number $b_n \in R$, and the operator should be symmetric for even-order derivative, i.e. $b_n = b_{-n}$ and be anti-symmetric for odd-order derivative, i.e. $b_n = -b_{-n}$; (2) for even-order derivative $\sum_{n=-N/2}^{N/2} b_n = 0$ so we can calculate $b_0 = -2\sum_{n=1}^{N/2} b_n$, and for odd-order derivative $b_0 = 0$.

LEAST-SQUARE OPTIMIZATION

To minimize the FD coefficients, we differentiate the objective functions respected to k_i and set the equal zero, here shows the result of differentiation.

$$\frac{\partial E}{\partial k_j} = 0 = \sum_{i}^{c} (k_i - 2\sum_{n=1}^{N/2} b_n \sin(nk_i)), \qquad \text{for m} = 1 \quad (9)$$

$$\frac{\partial E}{\partial k_j} = 0 = \sum_{i}^{c} (-k_i^2 + 2\sum_{n=1}^{N/2} b_n (1 - \cos(nk_i)), \quad \text{for m} = 2 \quad (10)$$

We can calculate the optimized FD coefficients by forming the matrix equations of equation 9 and 10. To set the stopping criteria, we define the error limitation or the error threshold T as follows

$$T = \sum_{i}^{c} (k_i - \sum_{n=N/2}^{N/2} b_n \sin(nk_i)), \qquad \text{for m} = 1 \quad (11)$$

$$T = \sum_{i}^{c} (-k_i^2 - \sum_{n=N/2}^{N/2} b_n \cos(nk_i)), \qquad \text{for m} = 2 \quad (12)$$

in this report, we set the error threshold as 0.0001, proposed by Zhang (2013), which give a comparable result between theoretical analyzes and numerical experiments.

RESULTS AND DISCUSSION

The first derivative optimized FD coefficients are shown in the table 1 from the 4th-12th order of approximation, we can see that the higher the order of approximation increases the higher the wavenumber coverage is. This means when the order of approximation increases, we should get less numerical dispersions in the numerical result.

From Figure 1, we can see that the optimized FD operator has higher wavenumber coverage when compare to the conventional FD operator in the same order of approximation. The smaller order of optimized FD operator even gives almost the same wavenumber coverage to the higher order of conventional FD operator for example, the 4th-order optimized FD operator has almost the same wavenumber coverage to the 6th-order conventional FD operator. This means the optimized FD operator can reduce the computational cost. The errors of optimized FD operator shown the behavior of oscillation around the zero error. The maximum peak of the oscillation is the error threshold that was set in the beginning of the process as 0.0001. The optimized FD coefficients are valuable in the range of $k = 0$ to $k =$ wavenumber coverage out of this range the error will increase dramatically.

Looking into the Table 2 which shows the optimized FD coefficients for second-derivative, we see the same trend to the first-derivative.The difference is that the second-derivative has higher wavenumber coverage than the first-derivative in the same order of approximation.

We can calculate the minimum number of grid points per wavelength G which need to be considered in the numerical modeling to reduce the numerical dispersions and to sustain a stability of modeling by:

$$G = \frac{2 \times 100}{\% \text{ of wavenumber coverage}} \quad (13)$$

The equation implies that the higher wavenumber coverage will gives the smaller number of grid points per wavelength that need to sustain the modeling and the less number of grid points per wavelength gives the bigger Δx which will reduce the computational cost.

CONCLUSION

We tried to study the optimized finite difference which we believe will help to reduce the numerical dispersions. Applying the least-squares optimization to the objective functions and solving the system of equations. Finally, we have the sets of optimized FD coefficients shown in the Tables 1 and 2. These coefficients were used to calculate the errors of FD operators by equation 11 and 12. The plots between the error and the wavenumber coverage shown in the Figure 1 and 2 implied that the optimized finite difference methods give better result than the conventional FD methods, the smaller order of optimized FD operator almost gives the same wavenumber coverage to the higher order of conventional FD operator.

Finally, The results shown that the numerical dispersions and computational cost can be reduced by using the optimized finite difference method.

APPENDIX A

From the Fourier transform

Table 1: Finite-difference coefficients for $1st$ derivative

Variable	4th-Order	6th-Order	8th-Order	10th-Orde r	12th-Order
k_c	17.82	30.87	41.47	49.73	56.16
$k_{coverage}$	16.97	30.02	40.70	49.05	55.55
b_1	0.6782	0.7770	0.8393	0.8803	0.9083
b_2	-0.0893	-0.1730	-0.2427	-0.2971	-0.3383
b_3		0.0231	0.0593	0.0986	0.1350
b_4			-0.0080	-0.0248	-0.0467
b_5				0.00347	0.0121
b_6					-0.0018

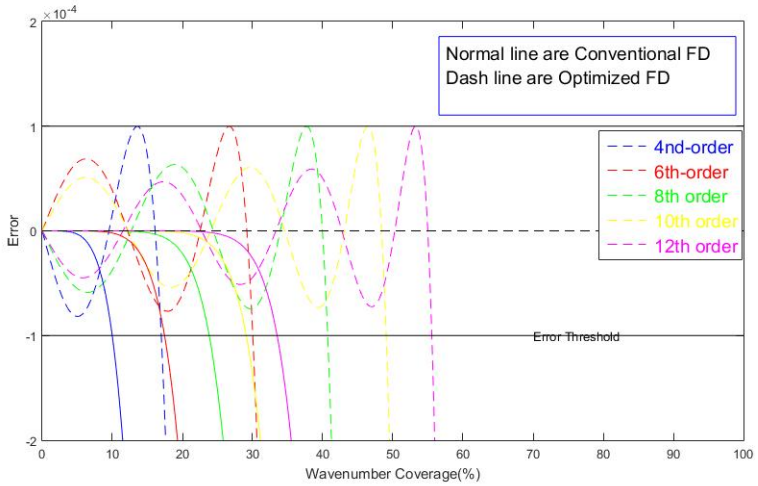


Figure 1: the $1st$ derivative plots show the relation between error and wavenumber coverage

Table 2: Finite-difference coefficients for $2nd$ derivative

Variable	4th-Order	6th-Order	8th-Order	10th-Orde r	12th-Order
k_c	25.42	39.20	49.91	58.00	64.16
$k_{coverage}$	24.50	28.33	49.13	57.31	63.54
b_0	-2.5539	-2.8179	-2.9692	-3.0619	-3.1214
b_1	1.3698	1.5737	1.7009	1.7830	1.8376
b_2	-0.0929	-0.1821	-0.2560	-0.3126	-0.3544
b_3		0.0173	0.0447	0.0739	0.1001
b_4			-0.0049	-0.0154	-0.0283
b_5				0.0019	0.0066
b_6					-0.0009

$$f(x) = \frac{1}{2\pi}\int_{-\infty}^{\infty} F(k_x)e^{ik_x x}dk_x \qquad \text{(A-1)}$$

$$F(k_x) = \int_{-\infty}^{\infty} f(x)e^{-ik_x x}dx \qquad \text{(A-2)}$$

Differentiating A-2 with respect to x

$$\frac{df(x)}{dx} = \frac{1}{2\pi}\int_{-\infty}^{\infty} ik_x F(k_x)e^{ik_x x}dk_x \qquad \text{(A-3)}$$

$$\frac{d^2 f(x)}{dx^2} = -\frac{1}{2\pi}\int_{-\infty}^{\infty} k_x^2 F(k_x)e^{ik_x x}dk_x \qquad \text{(A-4)}$$

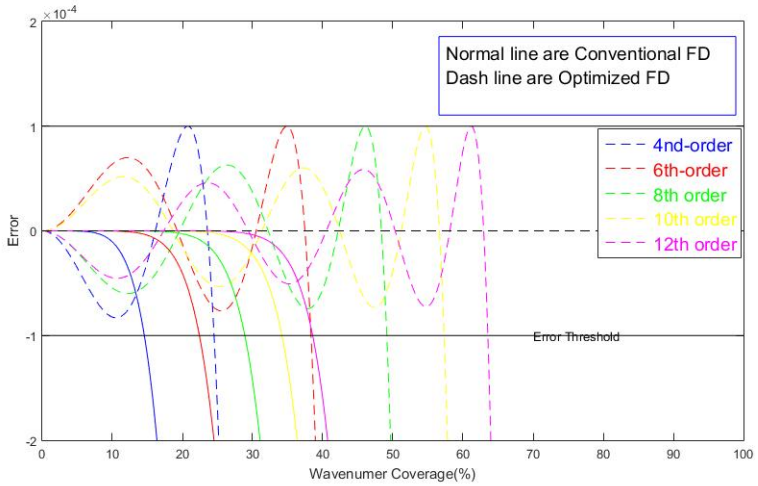


Figure 2: the $2nd$ derivative plots show the relation between error and wavenumber coverage

Fourier transform of $\partial f(x)/\partial x = \frac{1}{\Delta x}\sum_{n=-N/2}^{N/2} b_n f_n$, substituting equation A-2 into $\partial f(x)/\partial x$, we get

$$\frac{1}{2\pi}\int_{-\infty}^{\infty} ik_x F(k_x)e^{ik_x x}dk_x = \tag{A-5}$$

$$\frac{1}{2\pi\Delta x}\left[\sum_{n=-N/2}^{N/2} b_n \int_{-\infty}^{\infty} F(k_x)e^{ik_x(x+n\Delta x)}dk_x\right] \tag{A-6}$$

Since $b_n = -b_{-n}$ for odd derivative ,then

$$0 = \frac{1}{2\pi\Delta x}\left[b_0\int_{-\infty}^{\infty} F(k_x)e^{ik_x x}dk_x \right. \tag{A-7}$$

$$\left. +\sum_{n=1}^{N/2} b_n \int_{-\infty}^{\infty} F(k_x)(e^{ik_x(x+n\Delta x)} - e^{ik_x(x-n\Delta x)})dk_x\right]$$

$$= \frac{1}{2\pi\Delta x}\left[b_0\int_{-\infty}^{\infty} F(k_x)e^{ik_x x}dk_x \right.$$

$$\left. +\sum_{n=1}^{N/2} b_n \int_{-\infty}^{\infty} F(k_x)e^{ik_x x}(2i\sin(k_x n\Delta x))\right]$$

$$= \int_{-\infty}^{\infty} F(k_x)e^{ik_x x}\left(ik_x - \frac{1}{\Delta x}\left[b_0 + 2i\sum_{n=1}^{N/2} b_n \sin(k_x n\Delta x)\right]\right)dk_x$$

We known that $b_0 = 0$ for odd derivative

$$ik_x = \frac{i}{\Delta x}\sum_{n=-N/2}^{N/2} b_n \sin(k_x n\Delta x) \tag{A-8}$$

Fourier transform of $\partial^2 f(x)/\partial x^2 = \frac{1}{\Delta x^2}\sum_{n=-N/2}^{N/2} b_n f_n$, substituting equation A-2 into $\partial^2 f(x)/\partial x^2$, we get

$$-\frac{1}{2\pi}\int_{-\infty}^{\infty} k_x^2 F(k_x)e^{ik_x x}dk_x$$

$$= \frac{1}{2\pi\Delta x^2}\left[\sum_{n=-N/2}^{N/2} b_n \int_{-\infty}^{\infty} F(k_x)e^{ik_x(x+n\Delta x)}dk_x\right] \tag{A-9}$$

Since that $b_b = -b_n$ for even derivative, thus

$$0 = \frac{1}{2\pi\Delta x^2}\left[b_0\int_{-\infty}^{\infty} F(k_x)e^{ik_x x}dk_x \right. \tag{A-10}$$

$$\left. +\sum_{n=1}^{N/2} b_n \int_{-\infty}^{\infty} F(k_x)(e^{ik_x(x+n\Delta x)} + e^{ik_x(x-n\Delta x)})dk_x\right]$$

$$= \frac{1}{2\pi\Delta x^2}\left[b_0\int_{-\infty}^{\infty} F(k_x)e^{ik_x x}dk_x \right.$$

$$\left. +\sum_{n=1}^{N/2} b_n \int_{-\infty}^{\infty} F(k_x)e^{ik_x x}(2\cos(k_x n\Delta x))dk_x\right]$$

$$= \int_{-\infty}^{\infty} F(k_x)e^{ik_x x}\left(k_x^2 - \frac{1}{\Delta x^2}\left[b_0 + 2\sum_{n=1}^{N/2} b_n \cos(k_x n\Delta x)\right]\right)dk_x$$

We known that $b_0 = 0$ for odd derivative

$$-k_x^2 = \frac{1}{\Delta x^2}\sum_{n=-N/2}^{N/2} b_n \cos(k_x n\Delta x) \tag{A-11}$$

REFERENCES

Thongyoy, W., 2015, Least-squares finite difference operator: Undergrade Senior Project.

Zhang, J. H., 2013, Optimized finite-difference operator for broadband seismic wave modeling: GEOPHYSICS, **78**.

MPI implementation of the parallel fast marching method

Wisart Thongyoy and Chaiwoot Boonyasiriwat

ABSTRACT

In this work, we implement the parallel fast marching method in order to speed up the calculation time of finding travel time. We also optimize the implementation of the fast marching method by allowing repeated element in the binary heap. Using element method, the calculation time is up to 100 times faster. The parallelization result shown the success of the method, but the speed up factor was dropped when using many processing cores.

INTRODUCTION

The parallel method in this work adopted from the method of Yang and Stern (2017) which uses a domain decomposition method. Yang and Stern (2017) suggested that instead of updating the minimum time point in the domain, the process can be parallelized by decomposing the domain and update minimum time point in each subdomain. Since each subdomain run separately, it can affect the continuity of the wavefront between neighboring subdomains. To alleviate this, instead of continuously propagate wavefront in each subdomain separately, the subdomains can only be updated to a certain point of time and must wait until all of the subdomains reached the same point, after that, a new limiting time point is set and the process is repeated. This will be clarified and the pseudocode will be given in the next section.

THEORY AND METHODS

Sequential fast marching method

The fast marching method was developed to solve the Eikonal equation:

$$|\nabla\psi(\mathbf{x})|F(\mathbf{x}) = 1,\ \mathbf{x}\in\Omega, \tag{1}$$

where $\psi(\mathbf{x})$ is travel time, Ω is the domain, and $F(\mathbf{x})$ is a positive speed function. In geophysics, the equation is generally used to calculate travel time from source to any point in the domain.

Eq. 1 can be solved numerically by discretizing the domain. After that, use the Godunov-type finite difference approximation, suppose that our domain is in the 3-dimensional space, the result is

$$[\max(D^{-x}_{i,j,k}\psi, -D^{+x}_{i,j,k}, 0)^2 + \max(D^{-y}_{i,j,k}\psi, -D^{+y}_{i,j,k}, 0)^2 + \max(D^{-z}_{i,j,k}\psi, -D^{+z}_{i,j,k}, 0)^2]^{1/2} = \frac{1}{F_{i,j,k}}, \tag{2}$$

where $D^{-x}_{i,j,k}\psi$ and $D^{+x}_{i,j,k}\psi$ are backward and forward finite difference approximations of the spatial derivative $\frac{\partial\psi}{\partial x}$, respectively. For the first-order scheme, which is used in this work, the approximations are

$$\begin{aligned} D^{-x}_{i,j,k}\psi &= \frac{\psi_{i,j,k} - \psi_{i-1,j,k}}{\Delta x} \\ D^{+x}_{i,j,k}\psi &= \frac{\psi_{i+1,j,k} - \psi_{i,j,k}}{\Delta x} \end{aligned} \tag{3}$$

Note that, this rule can also be applied similarly along the y and z directions.

Algorithm 1 is the serial procedure of using the Godunov approximation in the fast marching method. In fast marching method, each point in the domain has status represented by tag(G). There are three types of tag: 1. Upwind node, whose travel time at that point was calculated and will not be changed, 2. Band node, whose travel time was calculated but can still be changed, and 3. Downwind node, whose travel time is not calculated, normally the traveltime of points with unknown travel time are set to be infinity, practically a very large number. Fast marching method mimics the behavior of wave front expanding, which means that the travel time will be calculated first at the source point which, certainly, has the least traveltime. Then, its neighbors is updated and

Algorithm 1 Sequential fast marching method

```
ψ ← +∞
G ← Downwind
ψ(x_s) ← ψ_0,  x_s are source positions
G(x_s) ← Upwind
G(N(x_s)) ← Band, N(x_s) are neighbors of x_s
ψ(N(x_s)) ← solve Godunov
while Band is not empty
    x_m ← x : gives min(ψ(x)), x ∈ Band
    G(x_m) ← Upwind
    ψ(x_m) ← solve Godunov
    for all G(N(x_m)) is not Upwind
        G(N(x_m)) ← Band
        ψ(N(x_m)) ← solve Godunov
    end for
end while
```

represent as band node. The process is repeatd by selecting the least traveltime node to be updated until reached the domain limit.

The method starts with assigning infinity to every point and tagging them to be Downwind. Next, the travel time at source point(s) is set and tag to be Upwind, note that the travel time of the source point can be set to be any value, but, in geophysics, since it mimics the traveling of wave, the travel time here is set to be zero. Then, the neighbors of the source point(s) are tag to Band and the travel times are assigned by solving equation 2. After that, the point with the least traveltime in the band is picked, tag to band, and updated travel time. Finally, the neighbors of the minimum travel time point is updated and tag to be Band if its tag is not Upwind. The process is repeated until there is no point with Band tag.

Speed up the fast marching method

Finding the minimum time in the band can be very slow, so the binary heap algorithm is generally used to speed up the process. The heap stores values of both $\mathbf{x}$ and $\psi(\mathbf{x})$. The algorithm of adding a new point to the heap is shown in **Algorithm 2**.

Algorithm 2 Adding point to heap

```
1: x_a ← point to be added
2: if G(x_a) is Downwind
3:     add x_a and ψ(x_a) to heap
4: else if G(x_a) is Downwind
5:     Find old x_a and ψ(x_a), from heap
6:     replace old value with x_a and ψ(x_a)
7: end if
```

When we add a new point to the heap, it is possible that the point is already in the heap (the point is in the heap and then updated to be a new value). If this happens, we have to replace the old value in the heap to make sure that the point will not be updated incorrectly, twice. The deleting from heap process is used to make sure that there are no repeated elements in the heap. However, we found that the finding old point in the heap to delete (5th line in **Algorithm 2**) is very expensive: it can consume about 80-90% of the total calculation time. To overcome this, we allow the heap to have repeated elements and add more conditions. There are two possible values in the heap: $\mathbf{x}$ and $\psi(\mathbf{x})$, since when the element is repeated the only quantity that is repeated is only $\mathbf{x}$, for example, we can have $[(\mathbf{x}, \psi_{old}(\mathbf{x})), (\mathbf{x}, \psi_{new}(\mathbf{x}))]$ in the heap. Now, if we have to find and delete the minimum value in the heap, we have to check whether or not the travel time in the heap and in the domain are equal or not, if not we skip the value. This is done based on the fact that the newest travel time added to the heap must be equal to the present travel time in the domain.

Parallelization by domain decomposition

To ensure the continuity of the wavefront, Yang and Stern (2017) used overlapping domain decomposition as shown in Figure 1.

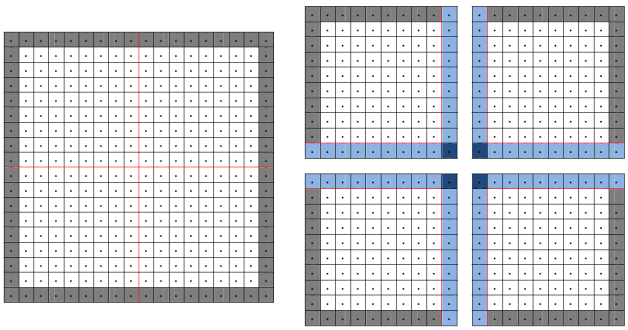


Figure 1: Domain decomposition in parallel computation (Yang and Stern, 2017)

Algorithm 3 Parallel fast marching by domain decomposition

```
1: while all points are not Upwind
2:   mintime ← minimum time in heap
3:   global mintime ← All reduce(mintime)
4:   time limit ← global mintime + stride
5:   while mintime < timelimit AND band size != 0
6:     mintime ← minimum time in heap
7:     Genenral fast marching method
8:   end while
9: the domain synchronization
10: end while
```

Algorithm 3 shows the pseudo code of parallel fast marching method, the stride value is the time step that the wave front will be advanced. Yang and Stern (2017), suggest to use stride equal to $2\Delta x$, In this work, we also used this value.

The domain synchronization is done as overlapping domain decomposition. The overlapping part shown in Figure 2 will set the new value after synchronization as minimum time. For example, if there are points $\mathbf{x}$ in 4 different sub domain with travel time t_1, t_2, t_3, and t_4, after synchronization all of the time value at point $\mathbf{x}$ are set to be $\min(t_1, t_2, t_3, t_4)$. To speed up the synchronization process, the process is performed selectively, that is not all of the points in the overlapping region are sent to neighboring processes but only points that are updated or the traveltimes are changed.

Testing model and machine

To test the performance of the code, we set up testing models. The testing models are cubic with a single point source at the center of the domain. The size of the domain will be varied. In this work, we use variable nh to indicate

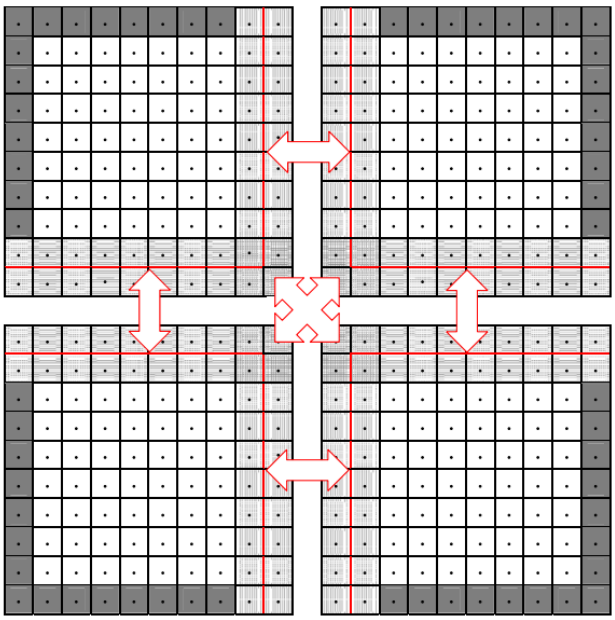


Figure 2: Data exchange between neighboring processes (Yang and Stern, 2017)

the size of the domain, where nh is number of points in each direction or nh^3 is amount of total grid points in the domain. The testing machine is a shared-memory system server with 24 Intel Xeon 2.5GHz cpus.

RESULTS

Before and after allowing repeated element

Figure 3 shows the time of before and after allowing repeated element. It is clear that the calculation decreases dramatically. This is because our added condition has much less calculation time than searching element in the heap. Note that Figure 3 is using old version of our program where synchronization is done by exchanging every points in the overlapping region between neighboring nodes, and the measurement of calculation time is not so appropriate. However, both before and after results are dealing exactly in the same way except with and without repeated heap value, then the result is fair to be used to determine the success of the method.

Speed up factor and calculation time

Figure 4 shows the calculation time of our final result. The calculation times are decreasing with increasing number of computational cores. However, the calculation time is increasing when amount of core is greater than 12.

Figure 5 shows the speed up of our code. The speed up is defined as

$$\text{speed up} = \frac{T_N^p}{T_1^p} \tag{4}$$

where T is calculation time, p indicate that the code is

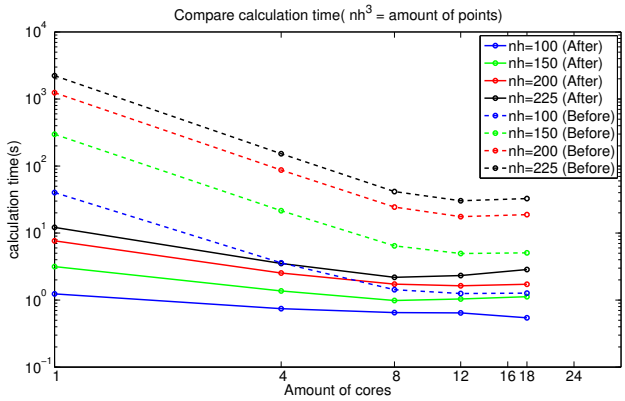


Figure 3: The comparison between before and after allowing heap to have repeated element. nh^3 is amount of points(old version of the code).

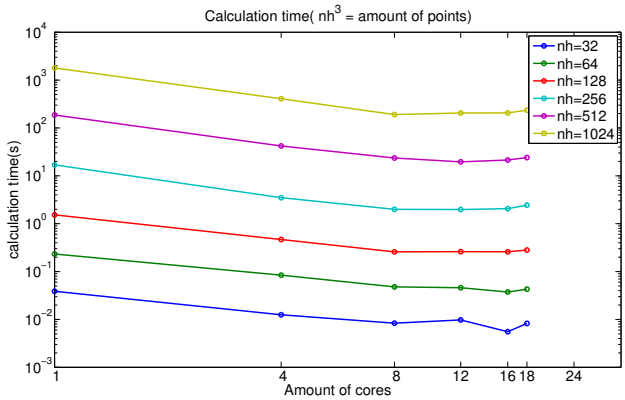


Figure 4: Calculation time of the final code(with selective synchronization and proper time measurement).

parallel code, and the subscript denote amount of operating cores.

From the result, we can see that the speed up is better when there is high amount of data, and the speed up dropping with increasing amout of cores.

DISCUSSION

Calculation time

Figure 6 shows the result of our code and from Yang and Stern (2017) is compared. The resolution of the figure is low but it is needed to make figures can be compared easily. Note also that the figure from Yang and Stern (2017) contain many value of stride, but in our work we consider only the cas of stride(δs) = $2.0\Delta h$. The comparison shows that our code performs better that the original work in the showed region. However, our computational resources are limited and can not test our code to be able to compare with Yang and Stern (2017) thoroughly.

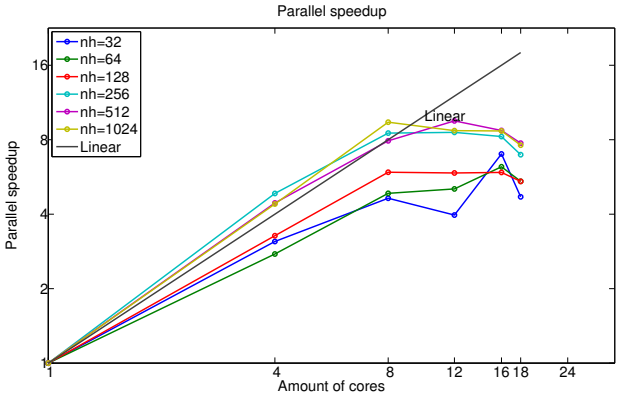


Figure 5: Speed up result.

Parallel speed up

Figure 6 shows the result of parallel speedup from Yang and Stern (2017). The graph is plot separately for each value of nh. Note again that, we are considering the black line where $\delta s = 2.0\Delta h$. Comparing with Figure 5 , we can see that our result is out-perform since the parallel speed up of our result are drop faster than Yang and Stern (2017). Moreover, our result start to perform worse when amount of cores reached 16, but for Yang and Stern (2017) this happen at far more numbeer of cores.

CONCLUSION

In this work, we try to implement the parallel fastmarching method from Yang and Stern (2017). The suggested method is to decompose a domain in to several subdomain and updated each domain semi-separately. The update have to be stop frequently and implement synchronization before update further. The results of our program is better when comparing computational time but worse for parallel speed up. However, sine we have limited computational resources, we cannot compare thoroughly with the original work. We also need to solve our scaling problem.

REFERENCES

Yang, J. and F. Stern, 2017, A highly scalable massively parallel fast marching method for the eikonal equation: Computational Physics, **332**, 333–362.

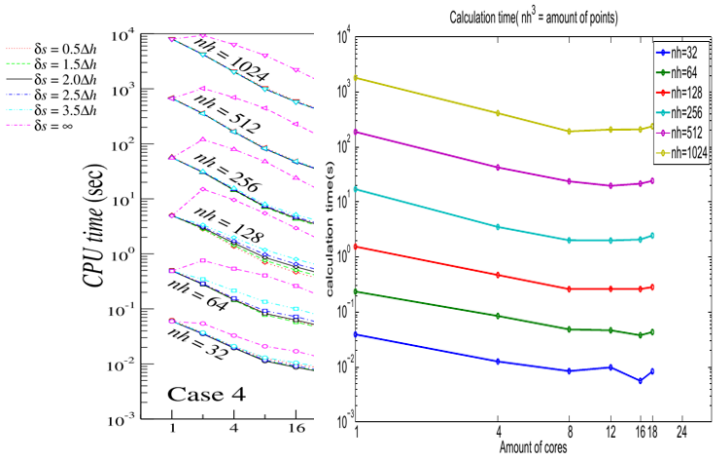


Figure 6: Comparing of calculation time between our result(right) and from Yang and Stern (2017)(left).

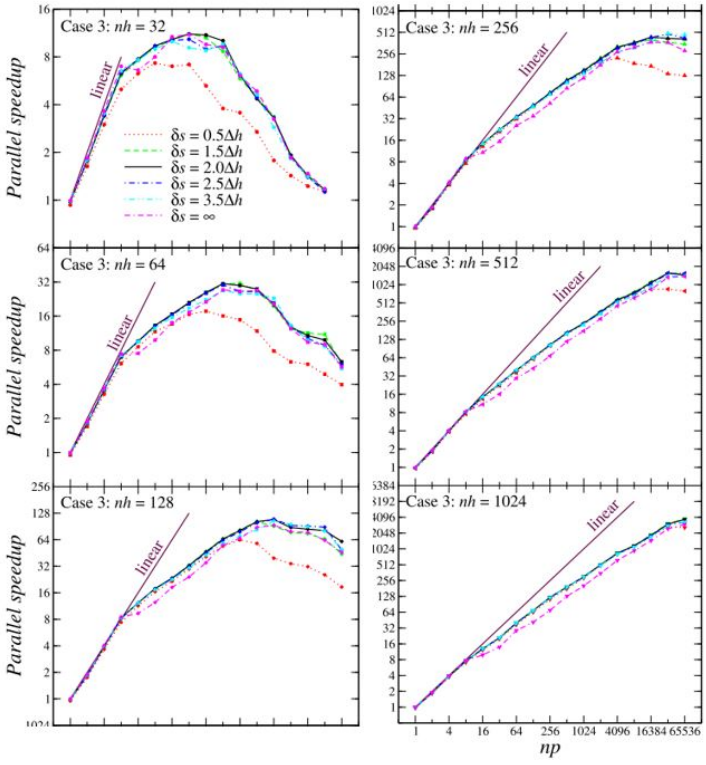


Figure 7: Parallel speed up result from Yang and Stern (2017)(left).

Introduction to computer vision: Models, Learning, and Inference

Chaloemrat Boonthanawat and Chaiwoot Boonyasiriwat

ABSTRACT

The problem of computer vision can be approached mathematically in term of probabilistic model. The probabilistic context is necessary for the real world image data which are always full with noise and uncertainty. There are three main components of the solution. The first one is the probabilistic model that related the image data to the world state. The model can be classified into two types, discriminative and generative model. The second component is the learning algorithm. The learning algorithm is used to find the parameters in the model using many paired training examples. The third component is the inference algorithm. The inference algorithm is used to infer the world state from the new image data.

INTRODUCTION

The goal of computer vision is to build the machine that can extract useful information from images. The kind of task we are interested in is object classification, image recognition, motion analysis, image restoration, and etc. To put it simply, we seek to automate tasks that human visual system can do.

Mathematically, the problem can be stated as "given the image data $\boldsymbol{x}$, the program should be able to infer the world state $\boldsymbol{w}$". The world state $\boldsymbol{w}$ may be continuous (the 3D pose of a body model) or discrete (the presence or absence of a particular object). When the world state is continuous, we call regression model and when it is discrete, we call classification model. The image data $\boldsymbol{x}$ can be discrete (RGB integer value from 0 to 255) or continuous (pixel intensity by real value).

The model of interest is based on probabilistic approach. It is useful because usually the relation between image data and world state is many to one. Moreover,the image data is often full with noise from environment and we need probabilistic model to capture this information.

There are three components for the solution of the problem (Prince, 2012) :

- Model: relates the visual data $\boldsymbol{x}$ and the world state $\boldsymbol{w}$ specify by model parameters $\boldsymbol{\theta} = \{\boldsymbol{\theta_i}, \boldsymbol{\theta_j}, \ldots\}$.
- Learning algorithm: allows us to fit the parameters $\boldsymbol{\theta}$ using paired training examples $\mathcal{D} = \{\boldsymbol{x_i}, \boldsymbol{w_i}\}_{i=1}^{I}$.
- Inference algorithm: takes a new observation $\boldsymbol{x^*}$ and uses the model with a trained parameters $\hat{\boldsymbol{\theta}}$ to return the posterior probability $\Pr(\boldsymbol{w}|\boldsymbol{x^*}, \hat{\boldsymbol{\theta}})$ over the world state $\boldsymbol{w}$.

MODELS

Probability distribution and conjugacy

The choices of probability distribution depend on the task of interest. Generally, it is depending on the domain of image data $\boldsymbol{x}$ or the domain of world state $\boldsymbol{w}$ (continuous or discrete and the range of possible value) (Table 1.)

Each probability distribution will have parameters ($\boldsymbol{\theta} = \{\boldsymbol{\theta_1}, \boldsymbol{\theta_2}, \ldots\}$) corresponding to the model itself. These parameters will be specified by leaning algorithm from paired training examples. In addition, we can also have the probability distribution of parameters from the main model meaning the main model doesn't necessary have a unique parameters. It can have a wide range of possible parameters value. Its parameters will be called hyperparameters.

The choices of probability distribution are chosen such that each is a conjugacy to each other (Table 2.)

The property of conjugacy is that when two distributions multiply, it will give the same distribution with some constant. This property is extremely useful in both learning and inference algorithm by arranging the expression in a closed form.

Table 1: Domain of probability distribution

Distribution	Domain
Bernoulli	$y \in \{0, 1\}$
categorical	$y \in \{1, 2, \ldots, K\}$
univariate normal	$y \in \mathbb{R}$
multivariate normal	$\boldsymbol{y} \in \mathbb{R}^k$

Table 2: Example of possible distribution and its conjugacy

Distribution of model	Distribution of model parameters
Bernoulli	beta
categorical	Dirichlet
univariate normal	normal inverse gamma
multivariate normal	normal inverse Wishart

Hidden variable and more complex probability distribution

Even though, the probability distribution showed in Table 1 is useful for many tasks. It is still limited in real world application because real world image data is complex. To model this type of complex data, we need an additional probability distribution.

The idea of hidden variables is useful for creating more complex distribution from an simpler one. The probability of the model with hidden variables can be written as,

$$\Pr(y) = \int \Pr(y,h)dh \tag{1}$$

$$= \int \Pr(y|h)P(h)dh \tag{2}$$

for continuous case or,

$$\Pr(y) = \sum \Pr(y|h) \tag{3}$$

$$= \sum \Pr(y|h)P(h) \tag{4}$$

for discrete case.

The idea is that we choose probability distribution of $\Pr(y|h)$ and $\Pr(h)$ such that when we compute $\Pr(y)$, it will give the new probability distribution (Table 3.)

Table 3: Complex probability distribution

Distribution	$\Pr(y\|h)$	$\Pr(h)$
Mixture of Gaussian	multivariate normal	categorical
t-distribution	multivariate normal	gamma
Subspace model	multivariate normal	normal

With the knowledge of probability distribution, every model that related the image data and world state fall into either one of this two categories.

Discriminative model

In this model, we will choose the probability distribution on the world state based on the image data.

$$\Pr(\boldsymbol{w}|\boldsymbol{x},\boldsymbol{\theta_m}) \tag{5}$$

The advantage of this approach is that the inference algorithm is extremely easy by just using the new image data x^* in the discriminative model.

Generative model

In this model, we will choose the probability distribution on the image data based on the world state. We have to model both the probability distribution between image data and world state and also the probability distribution of the world state alone.

$$\Pr(\boldsymbol{x}|\boldsymbol{w},\boldsymbol{\theta_m}) \tag{6}$$

$$\Pr(\boldsymbol{w}|\boldsymbol{\theta_w}) \tag{7}$$

The advantage of this approach is that we can incorporate the knowledge about how world state generate the image data into the model. But the disadvantage is that inference algorithm is more complicated for computing.

LEARNING

There are three main ways we can used to fit the parameters $\boldsymbol{\theta}$ by using I paired training examples $\mathcal{D} = \{\boldsymbol{x_i}, \boldsymbol{w_i}\}_{i=1}^{I}$. The assumption we used is that each pair of data and world state is independent from each other.

Maximum likelihood

The maximum likelihood (ML) method finds the set of parameters $\hat{\theta}$ in which paired training examples $\mathcal{D} = \{\boldsymbol{x_i}, \boldsymbol{w_i}\}_{i=1}^{I}$ are most likely. For a discriminative model, we can write the equation as,

$$\hat{\boldsymbol{\theta}}_{\boldsymbol{m}} = \underset{\boldsymbol{\theta_m}}{\operatorname{argmax}} \left[\Pr(\boldsymbol{w_1},\ldots,\boldsymbol{w_I}|\boldsymbol{x_1},\ldots,\boldsymbol{x_I},\boldsymbol{\theta_m})\right] \tag{8}$$

$$= \underset{\boldsymbol{\theta_m}}{\operatorname{argmax}} \left[\prod_{i=1}^{I} \Pr(\boldsymbol{w_i}|\boldsymbol{x_i},\boldsymbol{\theta_m})\right] \tag{9}$$

For a generative model, we can write the equation as,

$$\hat{\boldsymbol{\theta}}_{\boldsymbol{m}} = \underset{\boldsymbol{\theta_m}}{\operatorname{argmax}} \left[\Pr(\boldsymbol{x_1},\ldots,\boldsymbol{x_I}|\boldsymbol{w_1},\ldots,\boldsymbol{w_I},\boldsymbol{\theta_m})\right] \tag{10}$$

$$= \underset{\boldsymbol{\theta_m}}{\operatorname{argmax}} \left[\prod_{i=1}^{I} \Pr(\boldsymbol{x_i}|\boldsymbol{w_i},\boldsymbol{\theta_m})\right] \tag{11}$$

and for parameters of the world state probability distribution $\Pr(\boldsymbol{w}|\boldsymbol{\theta_w})$, we use the training world state $\{\boldsymbol{w_i}\}_{i=1}^{I}$

$$\hat{\boldsymbol{\theta}}_{\boldsymbol{w}} = \underset{\boldsymbol{\theta_w}}{\operatorname{argmax}} \left[\Pr(\boldsymbol{w_1},\ldots,\boldsymbol{w_I}|\boldsymbol{\theta_w})\right] \tag{12}$$

$$= \underset{\boldsymbol{\theta_w}}{\operatorname{argmax}} \left[\prod_{i=1}^{I} \Pr(\boldsymbol{w_i}|\boldsymbol{\theta_w})\right] \tag{13}$$

Maximum a posteriori

In maximum a posteriori (MAP), we introduce prior information about the parameters $\boldsymbol{\theta}$. These may come from previous experience or knowledge. For a discriminative

model, we can write the equation as,

$$\hat{\boldsymbol{\theta}}_{\boldsymbol{m}} = \underset{\boldsymbol{\theta_m}}{\operatorname{argmax}} \left[\Pr(\boldsymbol{\theta_m}|\{\boldsymbol{x_i}, \boldsymbol{w_i}\}_{i=1}^{I})\right] \tag{14}$$

$$= \underset{\boldsymbol{\theta_m}}{\operatorname{argmax}} \left[\Pr(\{\boldsymbol{x_i}, \boldsymbol{w_i}\}_{i=1}^{I}|\boldsymbol{\theta_m}) \Pr(\boldsymbol{\theta_m})\right] \tag{15}$$

$$= \underset{\boldsymbol{\theta_m}}{\operatorname{argmax}} \left[\prod_{i=1}^{I} \Pr(\boldsymbol{w_i}, \boldsymbol{x_i}|\boldsymbol{\theta_m}) \Pr(\boldsymbol{\theta_m})\right] \tag{16}$$

$$= \underset{\boldsymbol{\theta_m}}{\operatorname{argmax}} \left[\prod_{i=1}^{I} \Pr(\boldsymbol{w_i}|\boldsymbol{x_i}, \boldsymbol{\theta_m}) \Pr(\boldsymbol{x_i}|\boldsymbol{\theta_m}) \Pr(\boldsymbol{\theta_m})\right] \tag{17}$$

$$= \underset{\boldsymbol{\theta_m}}{\operatorname{argmax}} \left[\prod_{i=1}^{I} \Pr(\boldsymbol{w_i}|\boldsymbol{x_i}, \boldsymbol{\theta_m}) \Pr(\boldsymbol{\theta_m})\right] \tag{18}$$

For a generative model, we can write the equation as,

$$\hat{\theta}_m = \underset{\boldsymbol{\theta_m}}{\operatorname{argmax}} \left[\Pr(\boldsymbol{\theta_m}|\{\boldsymbol{x_i}, \boldsymbol{w_i}\}_{i=1}^{I})\right] \tag{19}$$

$$= \underset{\boldsymbol{\theta_m}}{\operatorname{argmax}} \left[\prod_{i=1}^{I} \Pr(\boldsymbol{w_i}, \boldsymbol{x_i}|\boldsymbol{\theta_m}) \Pr(\boldsymbol{\theta_m})\right] \tag{20}$$

$$= \underset{\boldsymbol{\theta_m}}{\operatorname{argmax}} \left[\prod_{i=1}^{I} \Pr(\boldsymbol{x_i}|\boldsymbol{w_i}, \boldsymbol{\theta_m}) \Pr(\boldsymbol{w_i}|\boldsymbol{\theta_m}) \Pr(\boldsymbol{\theta_m})\right] \tag{21}$$

$$= \underset{\boldsymbol{\theta_m}}{\operatorname{argmax}} \left[\prod_{i=1}^{I} \Pr(\boldsymbol{x_i}|\boldsymbol{w_i}, \boldsymbol{\theta_m}) \Pr(\boldsymbol{\theta_m})\right] \tag{22}$$

We can see that maximum likelihood is a special case of maximum a posteriori when we don't have any information about parameters in advance.

In order to simplify the learning algorithm, we can use the logarithm of the expression because it is monotonic function. The maximum parameters of transformed function will still remain the same, but it will give an easier expression to dealt with.

Expectation maximization algorithm

The EM algorithm is used to find the parameters $\hat{\boldsymbol{\theta}}_{\boldsymbol{m}}$ with the equation of the form (maximum likelihood),

$$\hat{\boldsymbol{\theta}}_{\boldsymbol{m}} = \underset{\boldsymbol{\theta_m}}{\operatorname{argmax}} \left\{\sum_{i=1}^{I} \log \left[\sum \Pr(\boldsymbol{x_i}, \boldsymbol{w_i}|\boldsymbol{h_i}, \boldsymbol{\theta_m}) \Pr(\boldsymbol{h_i})\right]\right\} \tag{23}$$

or in this form (maximum a posteriori)

$$= \underset{\boldsymbol{\theta_m}}{\operatorname{argmax}} \left\{\sum_{i=1}^{I} \log \left[\sum \Pr(\boldsymbol{x_i}, \boldsymbol{w_i}|\boldsymbol{h_i}, \boldsymbol{\theta_m}) \Pr(\boldsymbol{h_i})\right] \Pr(\boldsymbol{\theta_m})\right\} \tag{24}$$

The same is true for the case when hidden variables are continuous. There are two main steps in the algorithm which will be iterated until the maximum parameters are found. 1.) the E-step

$$q_i(\boldsymbol{h_j}) = \frac{\Pr(\boldsymbol{x_i}, \boldsymbol{w_i}|\boldsymbol{h_j}, \boldsymbol{\theta}_{\boldsymbol{m}}^{[t]}) \Pr(\boldsymbol{h_j}|\boldsymbol{\theta}_{\boldsymbol{m}}^{[t]})}{\Pr(\boldsymbol{x_i}, \boldsymbol{w_i})} \tag{25}$$

2.) the M-step

$$\boldsymbol{\theta}_{\boldsymbol{m}}^{[t+1]} = \underset{\boldsymbol{\theta_m}}{\operatorname{argmax}} \left\{\sum_{i=1}^{I} \sum q_i(\boldsymbol{h_j}) \log \left[\Pr(\boldsymbol{x_i}, \boldsymbol{w_i}, \boldsymbol{h_i}|\boldsymbol{\theta_m})\right]\right\} \tag{26}$$

The Bayesian approach

The Bayesian approach is similar to the maximum a posteriori. Instead of finding a unique parameters $\hat{\boldsymbol{\theta}}$, we find the probability of all parameters. Each parameters will contribute to the inference based on its probability, the expression is

$$\Pr(\boldsymbol{\theta}|\mathcal{D}) = \Pr(\boldsymbol{\theta}|\{\boldsymbol{x_i}, \boldsymbol{w_i}\}_{i=1}^{I}) \tag{27}$$

$$= \frac{\Pr(\{\boldsymbol{x_i}, \boldsymbol{w_i}\}_{i=1}^{I}|\boldsymbol{\theta}) \Pr(\boldsymbol{\theta})}{\Pr(\{\boldsymbol{x_i}, \boldsymbol{w_i}\}_{i=1}^{I})} \tag{28}$$

$$= \frac{\prod_{i=1}^{I} \Pr(\boldsymbol{w_i}, \boldsymbol{x_i}|\boldsymbol{\theta}) \Pr(\boldsymbol{\theta})}{\Pr(\{\boldsymbol{x_i}, \boldsymbol{w_i}\}_{i=1}^{I})} \tag{29}$$

INFERENCE

In the maximum likelihood (ML) and maximum a posteriori method (MAP), we would get a unique parameters $\hat{\boldsymbol{\theta}}$. These parameters is used to find the probability of each possible world state $\boldsymbol{w}$ given the new image data $\boldsymbol{x}^*$.

Inference algorithm for discriminative model (ML, MAP method)

For discriminative model, we have already directly constructed an expression for the posterior distribution, and we simply evaluate it with the new data.

$$\Pr(\boldsymbol{w}|\boldsymbol{x}^*, \hat{\boldsymbol{\theta}}_{\boldsymbol{m}}) \tag{30}$$

Inference algorithm for generative model (ML, MAP method)

For generative model, we use Bayes' rule to calculate the posterior distribution $\Pr(\boldsymbol{w}|\boldsymbol{x}^*, \hat{\boldsymbol{\theta}})$. For continuous case, we got

$$\Pr(\boldsymbol{w}|\boldsymbol{x}^*, \hat{\boldsymbol{\theta}}) = \frac{\Pr(\boldsymbol{x}^*|\boldsymbol{w}, \hat{\boldsymbol{\theta}}_{\boldsymbol{m}}) \Pr(\boldsymbol{w}|\hat{\boldsymbol{\theta}}_{\boldsymbol{w}})}{\int \Pr(\boldsymbol{x}^*|\boldsymbol{w}, \hat{\boldsymbol{\theta}}_{\boldsymbol{m}}) \Pr(\boldsymbol{w}|\hat{\boldsymbol{\theta}}_{\boldsymbol{w}}) d\boldsymbol{w}} \tag{31}$$

For discrete case, we got

$$\Pr(\boldsymbol{w}|\boldsymbol{x}^*, \hat{\boldsymbol{\theta}}) = \frac{\Pr(\boldsymbol{x}^*|\boldsymbol{w}, \hat{\boldsymbol{\theta}}_{\boldsymbol{m}}) \Pr(\boldsymbol{w}|\hat{\boldsymbol{\theta}}_{\boldsymbol{w}})}{\sum_{\boldsymbol{w}} \Pr(\boldsymbol{x}^*|\boldsymbol{w}, \hat{\boldsymbol{\theta}}_{\boldsymbol{m}}) \Pr(\boldsymbol{w}|\hat{\boldsymbol{\theta}}_{\boldsymbol{w}})} \tag{32}$$

Inference algorithm for Bayesian approach

Evaluating the predictive distribution is more difficult for the Bayesian case since we have not estimated a unique parameter for a model. But we instead have a probability distribution over all possible models (all possible parameters). To calculate the probability of world state given the new data $\boldsymbol{x^*}$, we need to weight the probability of all possible models. For a discriminative model, we can write the equation as (Bishop, 2006),

$$\begin{aligned}\Pr(\boldsymbol{w^*}|\boldsymbol{x^*}) &= \int \Pr(\boldsymbol{\theta_m}|\mathcal{D}) \Pr(\boldsymbol{w^*}|\boldsymbol{x^*}, \boldsymbol{\theta_m}) d\boldsymbol{\theta_m} \\ &= \int \left[\frac{\prod_{i=1}^{I} \Pr(\boldsymbol{w_i}, \boldsymbol{x_i}|\boldsymbol{\theta_m}) \Pr(\boldsymbol{\theta_m})}{\Pr(\{\boldsymbol{x_i}, \boldsymbol{w_i}\}_{i=1}^{I})} \right] \Pr(\boldsymbol{w^*}|\boldsymbol{x^*}, \boldsymbol{\theta_m}) d\boldsymbol{\theta_m} \\ &= \int \left[\frac{\prod_{i=1}^{I} \Pr(\boldsymbol{w_i}|\boldsymbol{x_i}, \boldsymbol{\theta_m}) \Pr(\boldsymbol{\theta_m})}{\prod_{i=1}^{I} \Pr(\boldsymbol{w_i}|\boldsymbol{x_i})} \right] \Pr(\boldsymbol{w^*}|\boldsymbol{x^*}, \boldsymbol{\theta_m}) d\boldsymbol{\theta_m} \end{aligned} \tag{33}$$

For a generative model, we can write the equation as,

$$\Pr(\boldsymbol{w^*}|\boldsymbol{x^*}) = \int \Pr(\boldsymbol{\theta}|\mathcal{D}) \Pr(\boldsymbol{w^*}|\boldsymbol{x^*}, \boldsymbol{\theta}) d\boldsymbol{\theta} \tag{34}$$

$$= \int \Pr(\boldsymbol{\theta_m}, \boldsymbol{\theta_w}|\{\boldsymbol{x_i}, \boldsymbol{w_i}\}_{i=1}^{I}) \left[\frac{\Pr(\boldsymbol{x^*}|\boldsymbol{w}, \boldsymbol{\theta_m}) \Pr(\boldsymbol{w}|\boldsymbol{\theta_w})}{\int \Pr(\boldsymbol{x^*}|\boldsymbol{w}, \boldsymbol{\theta_m}) \Pr(\boldsymbol{w}|\boldsymbol{\theta_w}) d\boldsymbol{w}} \right] d\boldsymbol{\theta_m} d\boldsymbol{\theta_w} \tag{35}$$

$$= \int \Pr(\boldsymbol{\theta_m}|\{\boldsymbol{x_i}, \boldsymbol{w_i}\}_{i=1}^{I}) \Pr(\boldsymbol{\theta_w}|\{\boldsymbol{x_i}, \boldsymbol{w_i}\}_{i=1}^{I}) \left[\frac{\Pr(\boldsymbol{x^*}|\boldsymbol{w}, \boldsymbol{\theta_m}) \Pr(\boldsymbol{w}|\boldsymbol{\theta_w})}{\int \Pr(\boldsymbol{x^*}|\boldsymbol{w}, \boldsymbol{\theta_m}) \Pr(\boldsymbol{w}|\boldsymbol{\theta_w}) d\boldsymbol{w}} \right] d\boldsymbol{\theta_m} d\boldsymbol{\theta_w} \tag{36}$$

$$= \int \left[\frac{\prod_{i=1}^{I} \Pr(\boldsymbol{x_i}|\boldsymbol{w_i}, \boldsymbol{\theta_m}) \Pr(\boldsymbol{\theta_m})}{\prod_{i=1}^{I} \Pr(\boldsymbol{x_i}|\boldsymbol{w_i})} \right] \left[\prod_{i=1}^{I} \Pr(\boldsymbol{w}|\boldsymbol{\theta_w}) \Pr(\boldsymbol{\theta_w}) \right] \left[\frac{\Pr(\boldsymbol{x^*}|\boldsymbol{w}, \boldsymbol{\theta_m}) \Pr(\boldsymbol{w}|\boldsymbol{\theta_w})}{\int \Pr(\boldsymbol{x^*}|\boldsymbol{w}, \boldsymbol{\theta_m}) \Pr(\boldsymbol{w}|\boldsymbol{\theta_w}) d\boldsymbol{w}} \right] d\boldsymbol{\theta_m} d\boldsymbol{\theta_w} \tag{37}$$

In practice, if we choose an appropriate probability distribution (conjugacy), we do not necessary have to calculate the integral directly.

CONCLUSION

Even though the three components are the foundation of the solution, there are still many issues about performance, robustness and the amount of training data required for learning algorithm. More technique and method is needed to ensure that the program work nicely in real life. Nevertheless, these knowledge provided a foundation that can be applied to any task at hand.

ACKNOWLEDGMENTS

We would like to thank members of the Mahidol University Center for Scientic Computing (MCSC) group for comments and suggestions.

REFERENCES

Bishop, C. M., 2006, Pattern recognition and machine learning: Springer.

Prince, S. J. D., 2012, Computer vision: Models, learning, and inference: Cambridge University Press, New York.

Nonstandard finite difference for solving wave equation

Panuwat Pukhamwong and Chaiwoot Boonyasiriwat

ABSTRACT

Nonstandard finite difference (NSFD) are numerical methods that can used to solve differential equations. These methods were implemented in order to achieve higher accuracy compared to conventional finite difference. Nonstandard finite difference scheme is specific to a problem and has no trivial instruction for general problems. Normally it was devised by using information of known-solution of the PDE. When applies to the wave equation, it can provide better results for specific frequency of the wave. In 2D wave equation, the results are better for only plane wave form.

INTRODUCTION

Differential equations can be solved numerically by turning differential equation to difference equation which is discrete equation. A simple method is conventional finite difference. Solutions from this method are approximation. Simplest way to get higher accuracy of solution is decreasing step size. But decreasing step size lead to a higher computational cost.

Another way to achieve higher accuracy without decreasing step size is to modify the usual finite difference using known-solution form. This method can provide better approximate solution or may be exact solution for that solution form. These method, modified form usual finite difference, were called "Nonstandard Finite Difference (NSFD)". Because this method use information from known-solution which is specific for a problem, the NSFD scheme is also specific to that problem.

The application of nonstandard finite difference on wave equation have been done by James B. Cole (Mickens, 2000).

This report contains only theoretical part without numerical result.

MATHEMATICAL MODEL

One Dimensional wave equation

$$\left(\partial_{tt} - v(x)^2 \frac{\Delta t^2}{h^2}\partial_{xx}\right)\psi(x,t) = 0 \tag{1}$$

One solution to 1D wave equation 1 is

$$\psi_0(x,t) = e^{i(kx-\omega t)} \tag{2}$$

Standard finite difference or usual finite difference defined by

$$d_x^2 f(x) = \frac{\tilde{d}_x^2 f(x)}{h^2} \tag{3}$$

where h is step size and

$$\tilde{d}_x^2 f(x) = f(x+h) + f(x-h) - 2f(x) \tag{4}$$

which is central finite difference second order.

Now we want to modify this scheme to get a better result for 1D wave equation that its solution is eq. 2. Substitute solution 2 into usual finite difference scheme 3, we get

$$\left(\tilde{d}_t^2 - v(x)^2 \frac{\Delta t^2}{h^2}\tilde{d}_x^2\right)\psi_0(x,t) = \epsilon \tag{5}$$

where ϵ is an error from approximation.

To make an error to be zero $\epsilon = 0$, we use the constrain equation

$$\epsilon = 2\psi_0(x,t)\left((\cos(\omega\Delta t) - 1) - v^2(x)\frac{\Delta t^2}{h^2}(\cos(kh) - 1)\right) \tag{6}$$

$$v^2(x)\frac{\Delta t^2}{h^2} = \frac{cos(\omega\Delta t) - 1}{\cos(kh) - 1} \tag{7}$$

Using this constrain (eq. 7) to the scheme 3. We get

$$\left(\tilde{d}_t^2 - \frac{cos(\omega\Delta t) - 1}{\cos(kh) - 1}\tilde{d}_x^2\right)\psi(x,t) = 0 \tag{8}$$

This is nonstandard finite difference scheme on 1D wave equation which provide exact solution $\psi_0 = e^{i(kx-\omega t)}$.

With dispersive relation $v = \omega/k$, equation 8 can be rearranged to

$$\left(\frac{\tilde{d}_t^2}{4cos^2(\omega\Delta t/2)/\omega^2} - v^2(x)\frac{\tilde{d}_x^2}{4\sin^2(k(x)h/2)/k^2}\right)\psi(x,t) = 0 \tag{9}$$

As we can see, it just turn h from usual finite difference (eq. 3) to function of h, k, ω

Two Dimensional wave equation

$$\left(\partial_{tt} - \vec{v}^2 \frac{\Delta t^2}{h^2}\nabla^2\right)\psi(x,t) = 0 \tag{10}$$

Turning equation 10 to a finite difference equation

$$(\tilde{D}_t^2 - \vec{v}^2 \frac{\Delta t^2}{h^2}\tilde{D}_1^2)\psi(x,t) = 0 \tag{11}$$

where D_1 is defined as

$$\tilde{D}_1^2 = \tilde{d}_x^2 + \tilde{d}_y^2 \tag{12}$$

One known solution is plane wave $\psi_0(x,t) = \Sigma_{k=|k|} e^{i(\vec{k}\cdot\vec{x}-\omega t)}$ Put this solution into equation 11, we get an error equation similar to 1D case.

$$\epsilon = 2e^{i\vec{k}\cdot\vec{x}}(\Sigma(\sin^2(k_i h/2))/h^2 - k^2) \tag{13}$$

But this error depends on the direction of wave vector. We can make the error zero for only one direction. The wave can move in any direction, so this approach does not work.

Another way is to modify the difference operator. Let us use more grid points on space by define

$$\begin{aligned} 2\tilde{D}_2^2 f(x,y) =& f(x+h,y+h) + f(x-h,y+h) \\ &+ f(x+h,y-h) + f(x-h,y-h) - 4f(x,y) \end{aligned}$$

and using

$$\tilde{D}_0^2 = \gamma\tilde{D}_1^2 + (1-\gamma)\tilde{D}_2^2 \tag{14}$$

Assuming that function of h in finite difference depends on only the magnitude of k like in 1D wave scheme.

$$\frac{\tilde{D}_0^2}{4\sin^2(kh/2)} e^{i\vec{k}\cdot\vec{x}} = e^{i\vec{k}\cdot\vec{x}} \tag{15}$$

γ can define as a function of k, h and an angle of wave vector. But after analysis, it seems that the dependence of wave direction on γ is weak. That means it can approximate function $\gamma(k,h)$. Now our scheme dose not depend on the direction anymore.

Two 2D scheme is similar to 1D except that the difference operator was changed to D_0

DISCUSSION

For this mathematical model, solution of 1D NSFD wave equation is exact if the source have only one frequency and source should be a sinusoidal. In 2D wave equation, NSFD give a good approximation for a plane wave at specific frequency.

Future work is to do this in simulation and try to use difference scheme and difference known-solution.

REFERENCES

Mickens, R. E., 2000, Application of nonstandard finite difference scheme: World Scientific, 109–154.

A study on rupture process and multiple fault segments in 2004 Indian Ocean Tsunami

Pawin Situngnoen and Chaiwoot Boonyasiriwat

ABSTRACT

The occurrence of the 2004 Indian Ocean Tsunami had two phenomena, the rupture process and the multiple fault segments of tsunami source characteristics. There are several tsunami source characteristics used in tsunami simulation, but this work focuses on 5 and 6 fault segments of tsunami source. To study the effect of among of fault segments and determine the best tsunami source characteristics at the Bay of Bengal, the 2D linearized shallow-water equations are solved by using the explicit finite difference method. The tsunami gauge data at some stations are used to compare the accuracy of tsunami simulated results. From my results, we find that the tsunami profile of 6 fault segments mostly fits the tsunami gauge data in both cases of dynamic rupture and static rupture. However, the satellite altimeter data must be used to compare the accuracy of tsunami simulated results, too. We have some issue about the reading process on our satellite altimeter data. This problem is going solved and the satellite altimeter data is going studied in future work.

REVIEW OF RUPTURE PROCESS

The rupture process of the 2004 Indian Ocean Tsunami Event was studied (Bilham, 2005; Ammon and et al., 2005; Lay and et al., 2005). After we review their works, we find that the important parameters of the rupture process are a total duration time, a rise time, and a rupture velocity when an underwater earthquake uplifts.

In addition, we know that the rupture zone of this tsunami covers northward from the epicenter (3.316^{o}N and 95.854^{o}E) along about 1,300 km (Ammon and et al., 2005). The rupture zone can be separated into a region of main tsunami excitation and region of weak tsunami excitation. However, Lay and et al. (2005) subdivided the rupture zone into three segments; Sumatra, Nicobar and Andaman Segments that can be presented in Figure 1.

From the study of Lay and et al. (2005), the region of main tsunami excitation covers the Sumatra and Nicobar Segments along 745 km, while the region of weak tsunami excitation covers only the Andaman Segment. In addition, each fault segments had their seismic parameters that are presented in Figure 2.

Moreover, the seafloor uplifted by two phenomena. First one is called "rapid slip" by using the total duration time about 10 minutes with the rupture velocity 2.5 km/sec (Lay and et al., 2005; Bilham, 2005). The another one is called "slow slip" by using the total duration time about 1 hour with the rupture velocity 0.75 km/sec (Lay and et al., 2005).

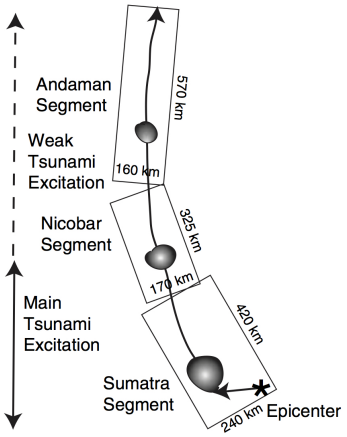


Figure 1: Three segments of rupture zone; Sumatra, Nicobar and Andaman Segments (Lay and et al., 2005).

Segment	Length (km)	Width (km)	Rake (deg)	Dip (deg)	Slip (m)	Rise time (min)
Sumatra	420	240	110	14	5-10	1 (rapid slip)
Nicobar	325	170	120	15	5	4-6 (rapid slip)
Andaman	570	160	150	18	<2	58 (slow slip)

Figure 2: Some of the seismic parameters and the rise time of underwater earthquake's uplift for three fault segments.

In all of the previous studies, they concluded that the rapid slip occurred in the region of main tsunami excitation when underwater earthquake uplifts. In addition, the slow slip occurred in the region of weak tsunami excitation.

In addition, the rupture process of this event is related to multiple fault segments of an underwater earthquake that is reviewed in the next section.

REVIEW OF MULTIPLE FAULT SEGMENTS

The 2004 Indian Ocean Tsunami was generated not only by the rupture process when the underwater earthquake uplifts, but also by multiple fault segments. The multiple

fault segments are used as multiple tsunami sources that are called tsunami source characteristics.

The tsunami source characteristics necessarily require a set of seismic parameters that are consists of a center of fault (x_0, y_0), focal depth (d), length of fault (L), width of fault (W), slip displacement (Δ), strike angle (ϕ), dip angle (δ), and rake angle (λ). Figure 3 presents the various tsunami source characteristics for simulation (Koh and et al., 2009; Guo and et al., 2015; Kowalik and et al., 2005; Son and et al., 2011; Arcas and Titov, 2006; Grilli and et al., 2007; Rhie and et al., 2007; Lay and et al., 2005).

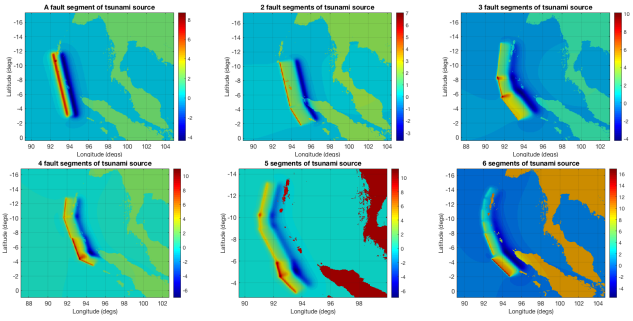


Figure 3: 6 types of tsunami source characteristics used in the simulation of the 2004 Indian Ocean Tsunami.

Grilli and et al. (2007) and Ioualalen and et al. (2007) studied the constraints of tsunami source and effect of dispersion by using the 5 fault segments of tsunami source characteristics.

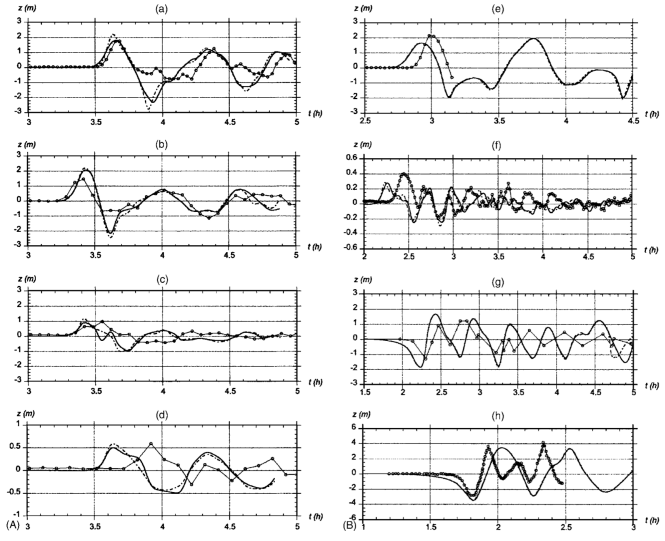


Figure 4: Tsunami profile and simulated result at some selected gauge stations (a) Hannimaadhoo; (b) Male; (c) Gan; (d) Diego Garcia; (e) Columbo; (f) Cocos Island; (g) Taphao-Noi; and (h) Mercator yacht (Grilli and et al., 2007).

They simulated the tsunami propagation using FUNWAVE and GEOWAVE programs. These programs solve the Boussinesq and shallow-water equations by a higher-order finite volume method. They predicted the tsunami profile at some gauge stations and satellite JASON-1's track. Figure 4 and 5 present their simulated results by comparing with the tsunami gauge data and altimeter data, respectively.

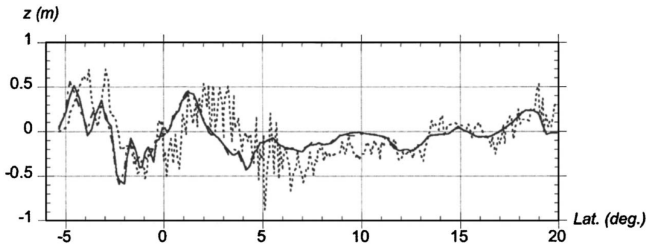


Figure 5: The numerical result and the satellite's altimeter data during record the sea surface elevation at the Indian Ocean in this event (Grilli and et al., 2007).

The 6 fault segments of tsunami source characteristic for the 2004 Indian Ocean Tsunami is used in the study of Lay and et al. (2005) and Rhie and et al. (2007). However, this work focuses only on the work of Rhie and et al. (2007).

The tsunami source characteristic of 6 fault segments in Rhie and et al. (2007) is used to study the best seismic model in tsunami simulation by Poisson and et al. (2011). In the work of Poisson and et al. (2011), they simulated the tsunami propagation with the altimeter data that is presented in Figure 6.

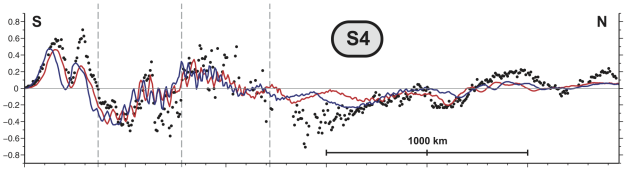


Figure 6: The numerical result and the satellite's altimeter data (Poisson and et al., 2011).

All of the previous studies motivate this work to study about the 5 and 6 fault segments of tsunami source characteristics in tsunami simulation. Moreover, our simulated results are compared with tidal gauge data and satellite altimeter data.

EXPERIMENT OF MULTIPLE FAULTS

The Governing Equations

The 5 and 6 fault segments of tsunami source are used in this work. The effect of fault number is tested by using the 2D linearized shallow-water equations with the PML boundary condition as

$$\frac{\partial \eta}{\partial t} = -(\sigma_x+\sigma_y)\eta - \sigma_x\sigma_y\psi - H\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}\right), \quad (1)$$

$$\frac{\partial u}{\partial t} = -\sigma_x u - \bar{U}\frac{\partial u}{\partial x} - \bar{V}\frac{\partial u}{\partial y} - g\frac{\partial \eta}{\partial x}, \quad (2)$$

$$\frac{\partial v}{\partial t} = -\sigma_x v - \bar{U}\frac{\partial v}{\partial x} - \bar{V}\frac{\partial v}{\partial y} - g\frac{\partial \eta}{\partial y}, \quad (3)$$

$$\frac{\partial \psi}{\partial t} = \eta, \quad (4)$$

where η is a sea-surface elevation, (u, v) are horizontal velocities, H is a total water column by $H = \eta + d$ (d is unperturbed water depth), $(\bar{U}, \bar{V})$ are the averaged velocities (that equal to 1 in this work), (σ_x, σ_y) are the absorption coefficients, and ψ is an auxiliary field.

For solving the governing equations, the staggered grid of finite difference and the first-order upwind are used for solving of the conservation of momentum and mass equations, respectively.

The numerical index form of the governing equations can be written as

$$\begin{aligned}
\psi_{i,j}^{n+1} &= \psi_{i,j}^{n} + (\Delta t)\eta_{i,j}^{n}, \\
u_{i,j}^{n+1} &= \left[1-(\Delta t)(\sigma_x)_{i,j}\right]u_{i,j}^{n} - g\frac{\Delta t}{\Delta x}\left(\eta_{i+1,j}^{n}-\eta_{i,j}^{n}\right) \\
&- \frac{1}{2}\frac{\Delta t}{\Delta x}\bar{U}\left(u_{i+1,j}^{n}-u_{i-1,j}^{n}\right) - \frac{1}{2}\frac{\Delta t}{\Delta y}\bar{V}\left(u_{i,j+1}^{n}-u_{i,j-1}^{n}\right), \\
v_{i,j}^{n+1} &= \left[1-(\Delta t)(\sigma_y)_{i,j}\right]v_{i,j}^{n} - g\frac{\Delta t}{\Delta y}\left(\eta_{i,j+1}^{n}-\eta_{i,j}^{n}\right) \\
&- \frac{1}{2}\frac{\Delta t}{\Delta x}\bar{U}\left(v_{i+1,j}^{n}-v_{i-1,j}^{n}\right) - \frac{1}{2}\frac{\Delta t}{\Delta y}\bar{V}\left(v_{i,j+1}^{n}-v_{i,j-1}^{n}\right), \\
\eta_{i,j}^{n+1} &= \left[1-(\Delta t)(\sigma_x+\sigma_y)_{i,j}\right]\eta_{i,j}^{n} - \Delta t(\sigma_x\sigma_y)_{i,j}\psi_{i,j}^{n} \\
&- \frac{\Delta t}{\Delta x}\left(u_e^{+}H_{i,j}^{n} + u_e^{-}H_{i+1,j}^{n} - u_w^{+}H_{i-1,j}^{n} - u_w^{-}H_{i,j}^{n}\right) \\
&- \frac{\Delta t}{\Delta y}\left(v_n^{+}H_{i,j}^{n} + v_n^{-}H_{i,j+1}^{n} - v_s^{+}H_{i,j-1}^{n} - v_s^{-}H_{i,j}^{n}\right),
\end{aligned}$$

where

$$\begin{aligned}
u_e^{+} &= 0.5\left(u_{i,j}^{n}+|u_{i,j}^{n}|\right), & u_e^{-} &= 0.5\left(u_{i,j}^{n}-|u_{i,j}^{n}|\right), \\
u_w^{+} &= 0.5\left(u_{i-1,j}^{n}+|u_{i-1,j}^{n}|\right), & u_w^{-} &= 0.5\left(u_{i-1,j}^{n}+|u_{i-1,j}^{n}|\right), \\
v_n^{+} &= 0.5\left(v_{i,j}^{n}+|v_{i,j}^{n}|\right), & v_n^{-} &= 0.5\left(v_{i,j}^{n}-|v_{i,j}^{n}|\right), \\
v_s^{+} &= 0.5\left(v_{i,j-1}^{n}+|v_{i,j-1}^{n}|\right), & v_s^{-} &= 0.5\left(v_{i,j-1}^{n}+|v_{i,j-1}^{n}|\right).
\end{aligned}$$

Physical Domain

The physical domain in this work is the Bay of Bengal region that is located at latitude 15^oS to 25^oN and longitude 70^oE to 105^oE (Figure 7). We use the topography and bathymetry data from 2-minute guided global relief data (ETOPO2) [http:www.ngdc.noaa.gov/mgg/fliers/01mgg04.html] that contain the $1,050 \times 1,200$ grid points with $3,712$ meters of a grid spacing.

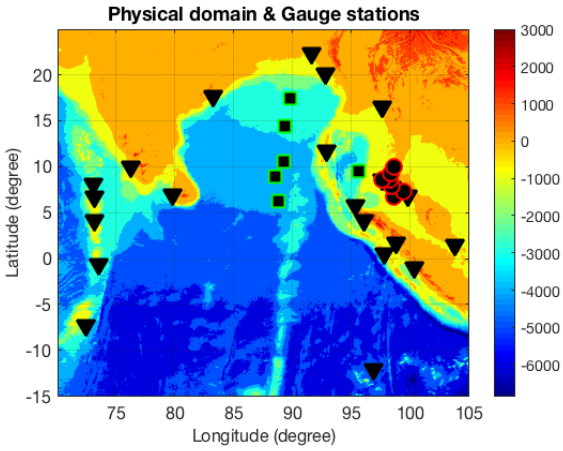


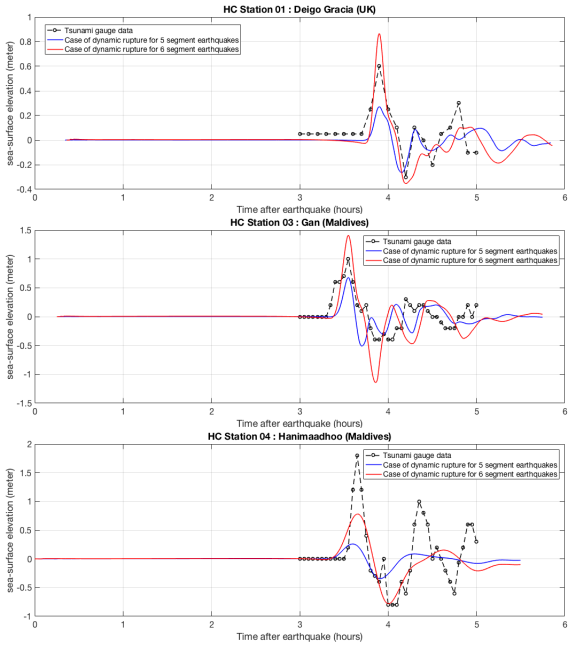
Figure 7: The Bay of Bengal region and our gauge stations.

NUMERICAL RESULTS

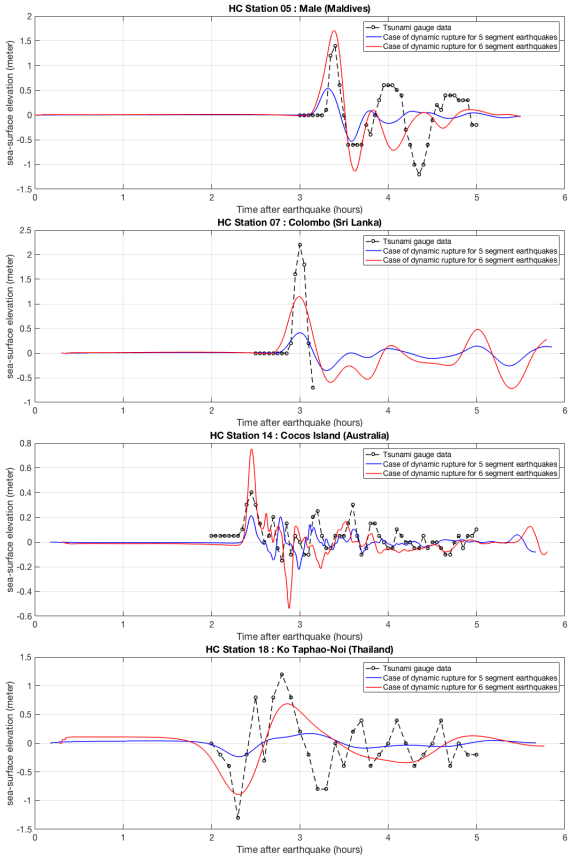
After we simulate the propagation of 2004 Indian Ocean Tsunami, we record tsunami profile at our gauge stations. However, we select some of the gauge stations to compare the simulated results with tsunami gauge data. Now we have two cases of simulated results that are the case of dynamic rupture (underwater earthquake uplifts at a different time) and the case of static rupture.

Case of Dynamic Rupture

Tsunami profile at some selected gauge stations are presented as

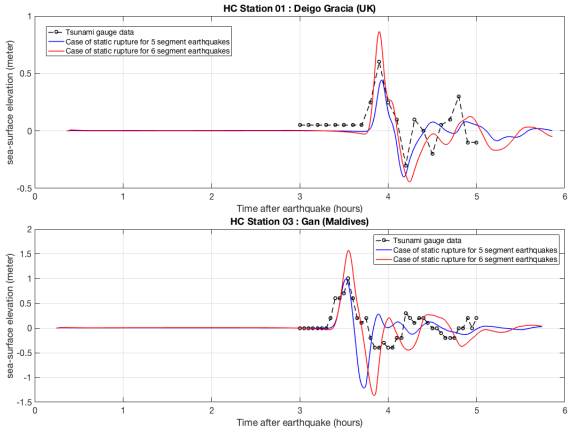


From these result, we can notice that tsunami profile of 6 fault segments can fit tsunami gauge data more than 5 fault segments.



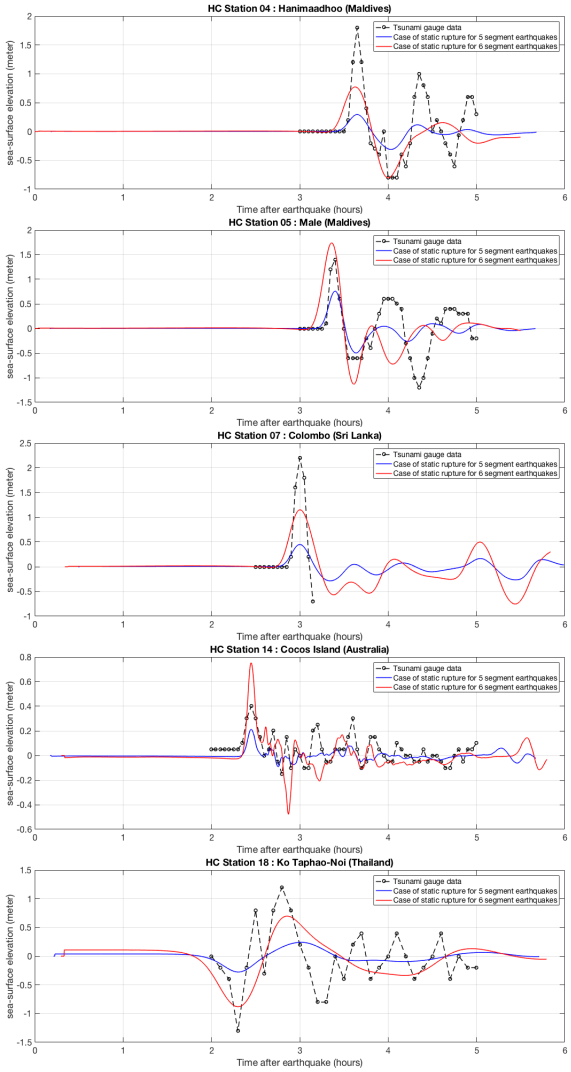
Case of Static Rupture

Tsunami profile at some selected gauge stations are presented as



From these result, we can notice that tsunami profile of 6 fault segments can fit tsunami gauge data more than 5 fault segments like in case of dynamic rupture. Thus, we can suppose that amount of fault segments can affect the accuracy of tsunami predictions.

However, we have another type of tsunami data that is the satellite altimeter data. Therefore, the detail about the preparation of satellite data is present in the next section.



SATELLITE ALTIMETER DATA

In the 2004 Indian Ocean Tsunami, the satellite altimeter (JASON-1) can first observe tsunami wave in the Indian Ocean by detecting sea surface height. The sea surface height data are published as the following link https://opendap.jpl.nasa.gov/.

After we finish the process of file reading, the satellite altimeter data is presented in Figure 8. In addition, Figure 9 and 10 are presented the satellite altimeter data of the previous works.

Notice that our satellite altimeter data is incorrect. This means that we have some problem with a technique for the process of file reading. Thus, this part of our work is continuously done.

SUMMARY

For the 2004 Indian Ocean Tsunami, the rupture process of multiple fault segments is studied by several re-

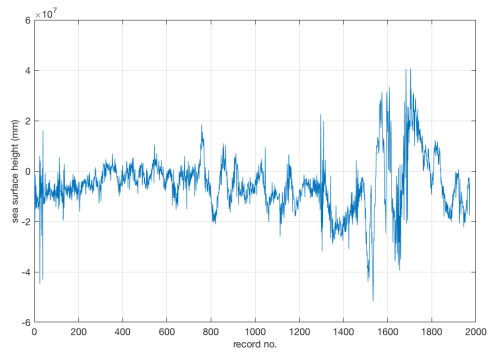


Figure 8: The satellite altimeter data in this work.

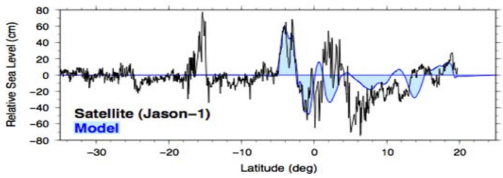


Figure 9: The satellite altimeter data of Arcas and Titov (2006).

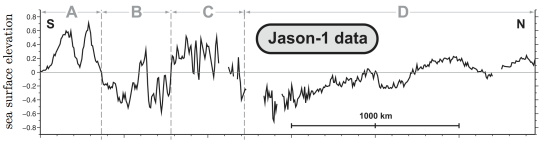


Figure 10: The satellite altimeter data of Poisson and et al. (2011).

searchers. They concluded that this tsunami has two rupture process; rapid slip and slow slip.

Furthermore, the several tsunami researchers proposed the various tsunami source characteristics. Thus, this work would like to study two type of tsunami source characteristics that are 5 and 6 fault segments of tsunami source.

The 2D linearized shallow-water equations with PML are solved by using The staggered-grid and first-order upwind schemes of explicit finite difference method. The physical domain in this work is the Bay of Bengal region that is located at latitude 15^oS to 25^oN and longitude 70^oE to 105^oE.

After we simulate the propagation of 2004 Indian Ocean Tsunami, we record tsunami profile at some selected gauge stations with gauge data. In addition, we have two cases of simulated results that are the case of dynamic rupture and the case of static rupture. We found that tsunami profile of 6 fault segments can fit tsunami gauge data more than 5 fault segments in two cases. Thus, we can suppose that amount of fault segments can affect the accuracy of tsunami predictions.

Furthermore, the satellite altimeter data is used in this work. However, we have some issue about file reading that results in the incorrect satellite altimeter data. Thus, this problem is continuously solved in future work.

REFERENCES

Ammon and et al., 2005, Rupture process of the 2004 sumatra-andaman earthquake: Science, **308**, 1133–1139.

Arcas and Titov, 2006, Sumatra tsunami: lessons from modeling: Surveys Geophysics, **27**, 679–705.

Bilham, 2005, A flying start, then a slow slip: Science, **308**, 1126–1127.

Grilli and et al., 2007, Source constraints and model simulation of the december 26, 2004, indian ocean tsunami: Journal of Waterway, Port, Coastal, and Ocean Engineering, **133**, 414–428.

Guo and et al., 2015, Time?space decoupled explicit method for fast numerical simulation of tsunami propagation: Pure Applied Geophysics, **172**, 569–587.

Ioualalen and et al., 2007, Modelling of the 26 december 2004 indian ocean tsunami: Case study of impact in thailand: Journal of Geophysics Research, **112**.

Koh and et al., 2009, Simulation of andaman 2004 tsunami for assessing impact on malaysia: Journal of Asian Earth Sciences, **36**, 74–83.

Kowalik and et al., 2005, Numerical modeling of the global tsunami: Indonesian tsunami of 26 december 2004: Science of Tsunami Hazards, **23**, 40–56.

Lay and et al., 2005, The great sumatra-andaman earthquake of 26 december 2004: Science, **308**, 1127–1133.

Poisson and et al., 2011, Is there a best source model of the sumatra 2004 earthquake for simulating the consecutive tsunami?: Geophysical Journal International, **185**, 1365–1378.

Rhie and et al., 2007, Slip of the 2004 sumatra?andaman earthquake from joint inversion of long- period global seismic waveforms and gps static offsets: Bulletin of the Seismological Society of America, **97**.

Son and et al., 2011, Nested and multi-physics modeling of tsunami evolution from generation to inundation: Ocean Modelling, **38**, 96–113.

OpenFOAM tutorials

Tanakorn Chantanasaro and Chaiwoot Boonyasiriwat

ABSTRACT

OpenFOAM is an open-source software for multi-physics simulation which mostly used to simulate computational fluid dynamics (CFD) problems. What I did here is summarizing some provided tutorials both official and non-official.

INTRODUCTION

The acronym OpenFOAM stands for Open Source Field Operation and Manipulation. It is a free-to-use open source numerical software with extensive CFD and multi-physics capabilities. Because the software itself is under active development and not that hard to use, it is popular among CFD researchers.The program is written by C++ language. Its numerical discretization based on finite-volume method so it can be used to capture the shock phenomena. Moreover, its syntax for PDEs closely resembles the equations being solved so the user can modify available solvers for their own uses easily.

METHODS

There are 3 main steps of doing the simulations in OpenFOAM. The first step is preprocessing which is about setting up the domain of the problem and meshing the domain. The user can use commands `blockMesh` or `snappyHexMesh` to create and mesh the geometry. The user can also use other programs to create geometry and mesh such as, Ansys, CFX and import into OpenFOAM.

The second step is choosing solver. There is a number of solvers provided in OpenFOAM. The examples are incompressible or compressible solver, multiphase flow, combustion solver, and electromagnetics solver.

The last step is postprocessing. The user can look at the result data and visualize the data by ParaView which is given when install OpenFOAM. The user can also export the result to other program such as, Fluent.

Tutorial 1 : Lid-driven cavity flow

The simulation of fluid in 2D rectangular box. The walls are fixed constantly except the top wall which moves to the right with velocity 1 m/s. The geometry is created and meshed using `blockMesh`. The solver is `icoFoam` which is the transient solver for incompressible, laminar flow of Newtonian fluid. Figure 1 and 2 show the the velocity at time $t = 0.5$ s, and the pressure at time $t = 0.5$ s respectively.

Tutorial 2 : 2D Dam break

The simulation of dam breaking in 2D. The geometry is created and meshed using `blockMesh`. The solver is `interFoam` which is the solver for 2 incompressible, isothermal immiscible fluid. Figure 3 show the simulation at time $t = 0$, 0.25, 0.5, 0.75, and 1 s respectively.

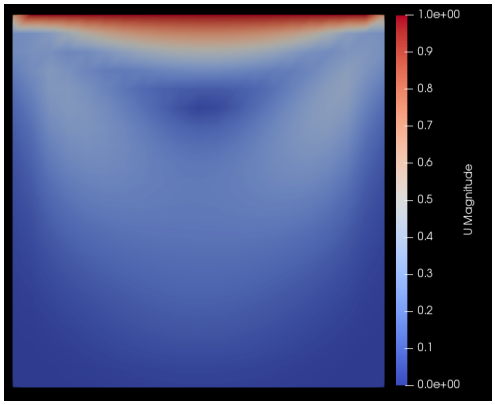


Figure 1: The velocity of the fluid at time $t = 0.5$ s

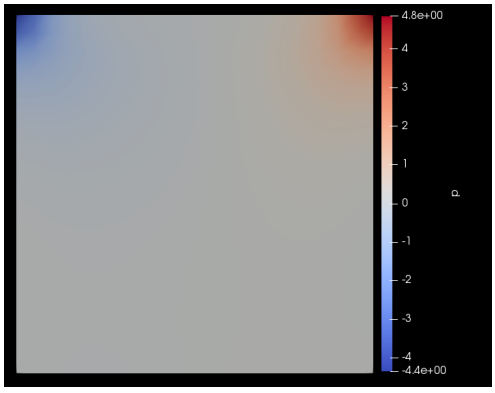


Figure 2: The pressure of the fluid at time t 0.5 s

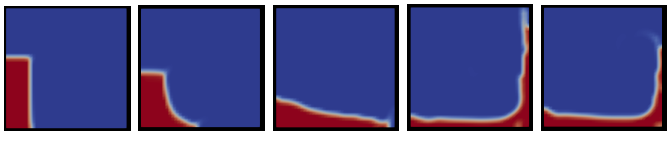


Figure 3: The simulation of 2D dam breaking at time $t = 0$, 0.25, 0.5, 0.75, and 1 s

Tutorial 3 : 3D Dam break

The simulation of dam breaking in 3D. Preprocessing step and solver are the same as in 2D case. Figure 4 show the simulation at time $t = 0$, 0.5, and 1 s respectively.

Tutorial 4 : Shock in Tube

The 2D simulation of gas in a tube. Initially, the left haft and the right haft gas has different pressure and temperature. The geometry is created and meshed using `blockMesh`. The solver is `sonicFoam` which is the transient solver for trans-sonic/supersonic, turbulence flow of a compressible gas. Figure 5 shows the initial pressure of the case. Figure 6, 7, and 8 show the pressure, velocity, and temperature of the gas at time $t = 0.007$ s respectively.

Tutorial 5 : Flow pass cylinder in 2D

The simulation of flow pass cylinder. The Reynold number is 195. The geometry is created and meshed using `blockMesh`. The solver is `icoFoam`. Figure 9 shows the magnitude of velocity field at time $t = 0.78$ s.

Tutorial 6 : Cut pipe

The 3D simulation of flow inside a cut pipe. The flow is from the right wall of the pipe. The geometry is created using `FreeCAD`. Then import and mesh in OpenFOAM using `snappyHexMesh`. The solver is `icoFoam`. Figure 10 shows the geometry and meshing of the case. Figure 11 shows the magnitude of velocity field on a middle cut plane of the pipe at time $t = 50$ s.

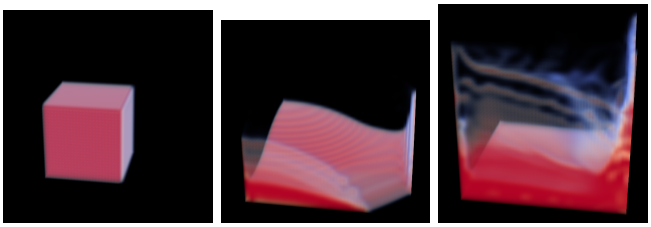


Figure 4: The simulation of 3D dam breaking at time $t =$ 0, 0.5, and 1 s

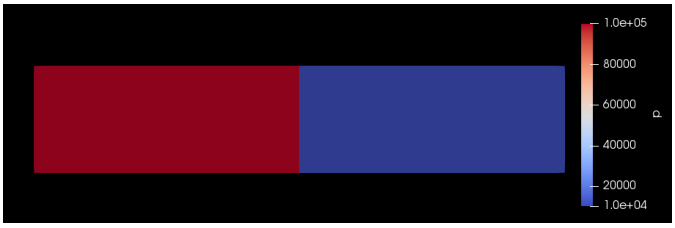


Figure 5: The initial condition of tutorial 4

CONCLUSION

Overall, OpenFOAM can be well understood and easy to understand by following the tutorials. In the part of creating and meshing. For simple object like in tutorial 1-5, `blockMesh` is a great tool for creating and meshing the geometry. But for complicate object like in tutorial 6 or others such as, airfoil, motorbike, building. The user should have other software for creating complicate object like `FreeCAD`, `AutoCAD`, `Blender` then import and mesh inOpenFOAM using `snappyHexMesh`. Or the user can finish both creating and meshing in other software such

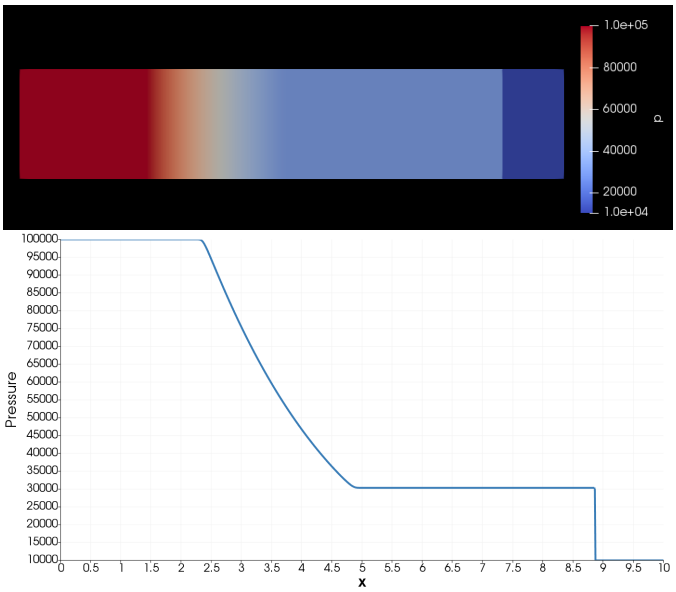


Figure 6: The pressure of gas in tube at time $t = 0.007$ s and its plot along the horizontal axis

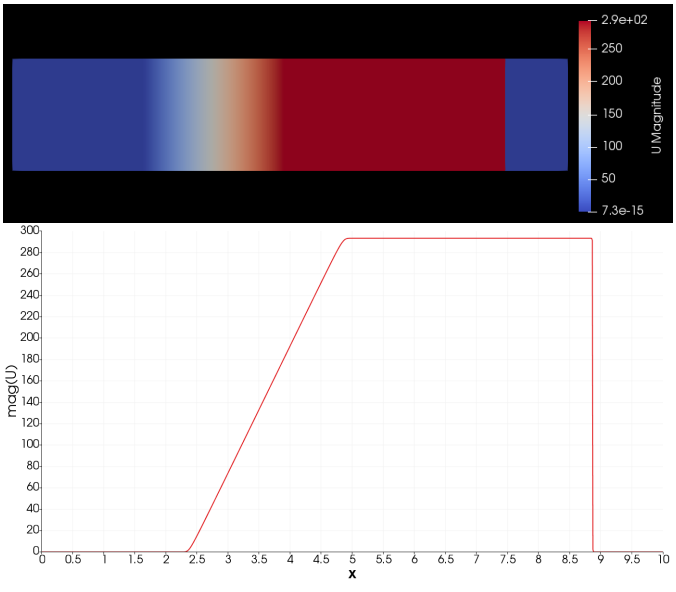


Figure 7: The velocity of gas in tube at time $t = 0.007$ s and its plot along the horizontal axis

as Ansys, STAR-CD then import into OpenFOAM.

In the solver step. Since there are many solver, the user can choose the solver which fit their own problem the most or modify to fit their equations easily.

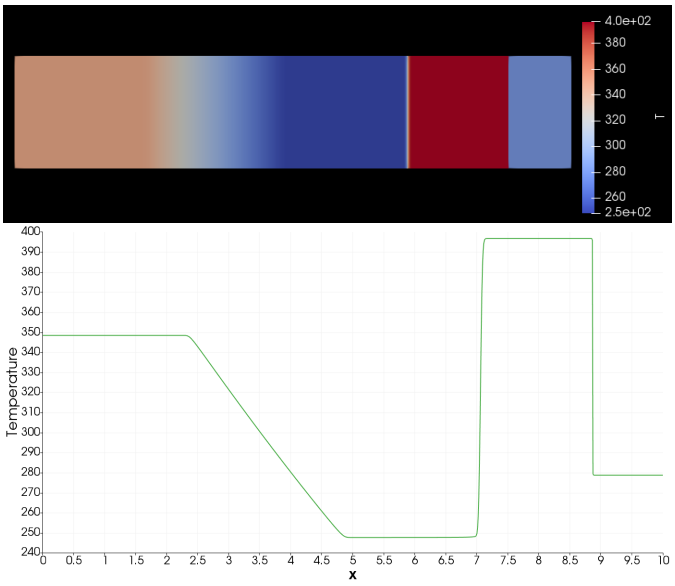


Figure 8: The temperature of gas in tube at time $t = 0.007$ s and its plot along the horizontal axis

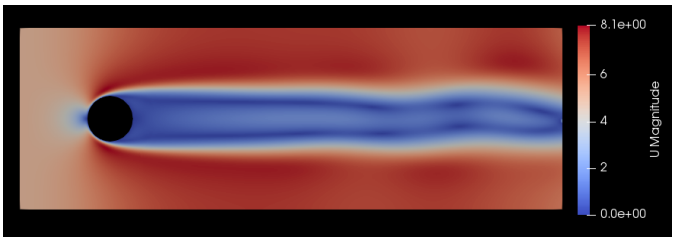


Figure 9: The magnitude of velocity field of flow pass cylinder at Re = 195 at time $t = 0.78$ s

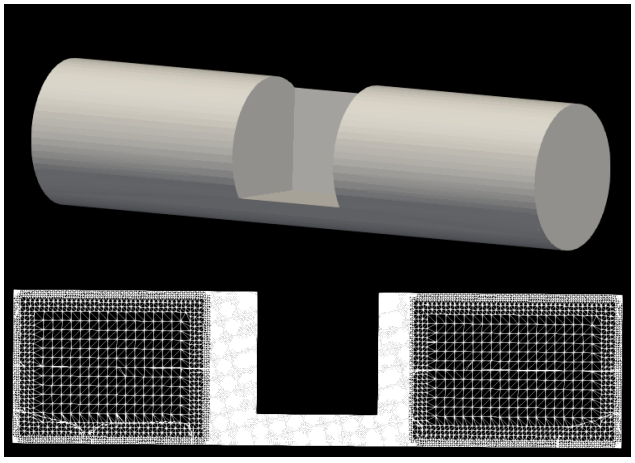


Figure 10: (Top) The geometry of tutorial 6 (Bottom) Meshing inside the pipe

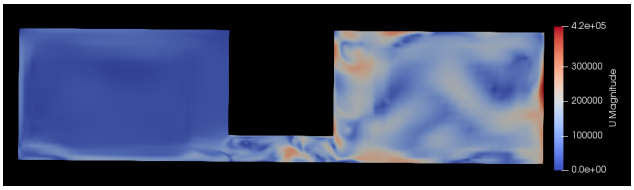


Figure 11: (Top) The magnitude of velocity field on a middle plane of the cut pipe at time $t = 50$ s

ACKNOWLEDGMENTS

I would like to thank Dr.Chaiwoot Boonyasiriwat for his useful devices and kindness. I also thanks the website in the reference list for providing and giving useful and clear tutorials so I can easily follow.

REFERENCES

https://cfd.direct/openfoam/user-guide/
http://the-foam-house5.webnode.es/
https://youtu.be/lrGO9QSBq1g