Available online at www.sciencedirect.com

ScienceDirect

Publishing Services by Elsevier

International Journal of Naval Architecture and Ocean Engineering 9 (2017) 390–403

http://www.journals.elsevier.com/international-journal-of-naval-architecture-and-ocean-engineering/

Comparison of different iterative schemes for ISPH based on Rankine source solution

Xing Zheng [a,*], Qing-wei Ma [a,b], Wen-yang Duan [a]

[a] College of Shipbuilding Engineering, Harbin Engineering University, Harbin 150001, China

[b] Schools of Engineering and Mathematical Science, City University, London EC1V 0HB, UK

Received 3 June 2016; revised 17 October 2016; accepted 23 October 2016

Available online 28 December 2016

Abstract

Smoothed Particle Hydrodynamics (SPH) method has a good adaptability for the simulation of free surface flow problems. There are two forms of SPH. One is weak compressible SPH and the other one is incompressible SPH (ISPH). Compared with the former one, ISPH method performs better in many cases. ISPH based on Rankine source solution can perform better than traditional ISPH, as it can use larger stepping length by avoiding the second order derivative in pressure Poisson equation. However, ISPH_R method needs to solve the sparse linear matrix for pressure Poisson equation, which is one of the most expensive parts during one time stepping calculation. Iterative methods are normally used for solving Poisson equation with large particle numbers. However, there are many iterative methods available and the question for using which one is still open. In this paper, three iterative methods, CGS, Bi-CGstab and GMRES are compared, which are suitable and typical for large unsymmetrical sparse matrix solutions. According to the numerical tests on different cases, still water test, dam breaking, violent tank sloshing, solitary wave slamming, the GMRES method is more efficient than CGS and Bi-CGstab for ISPH method.

Keywords: SPH; ISPH; ISPH_R; Iterative scheme; Dam breaking; Violent tank sloshing; Solitary wave slamming

1. Introduction

With the development of numerical methods, meshless particle methods get the robust advantage for breaking waves and their interaction with marine structures in naval architecture and ocean engineering. There are many different meshfree methods, such as Smoothed Particle Hydrodynamics (SPH) method (Monaghan, 1994), Moving Particle Semi-implicit (MPS) method (Koshizuka and Oka, 1996; Zhang et al., 2006; Khayyer and Gotoh, 2011), Meshless Local Petrov-Gelerkin (MLPG) method (Ma and Zhou, 2009) and so on. SPH is arguably one of most often-used meshfree methods and has been widely applied in marine and ocean engineering (Oger et al., 2007; Xu et al., 2009; Lind et al., 2012; Rafiee et al., 2012; Colagrossi and Landrini, 2003; Liu and Liu, 2006; Schwaiger, 2008; Ferrand et al., 2013; Zheng et al, 2014). There are two SPH schemes. One is weakly compressible SPH and the other is incompressible SPH (ISPH). The latter one is based on the time project method and need to solve the Poisson equation, which also meets the problems of large sparse matrix solution. There are many applications of ISPH method for water wave simulations (Rafiee et al., 2012; Lind et al., 2012; Xu et al., 2009; Shao and Lo Edmond, 2003; Shao et al., 2006; Shao, 2009) as it performs betters in many cases.

The principle of ISPH is to solve the partial differential equation for the pressure through the projection method. The project method was firstly implemented to the SPH method by Cummins and Rudman (1999). Many researchers have also improved and modified the projection method to make it more accurate and efficient. Compared to WCSPH, ISPH is a typically implicit by dealing with the pressure and velocity as

* Corresponding author.

E-mail address: zhengxing@hrbeu.edu.cn (X. Zheng).

Peer review under responsibility of Society of Naval Architects of Korea.

http://dx.doi.org/10.1016/j.ijnaoe.2016.10.007

primitive variables. WCSPH can be easy to program (Shadloo et al., 2012) and it is more widely used at present. However, some researcher (Hu and Adams, 2007; Xu et al., 2009; Zheng et al., 2014) suggested that ISPH was more accurate especially in the pressure representation. The reason is that when handling fluid flow with larger Reynolds number (typically >100), the standard WCSPH method has be found to suffer from large density variations. Hu and Adams (2007), Ellero et al. (2007) and Zheng et al. (2014) pointed out that WCSPH was computationally less efficient than ISPH in the case of fluids with different numerical cases.

With the improvement of ISPH method, some key numerical technologies are applied. Xu et al. (2009) and Lind et al. (2012) introduced a fick shift method to avoid the particle pattern distribution. Bonet and Lok (1999), Khayyer et al. (2008) proposed a corrected kernel formulation of the pressure gradient calculation, which can improve the accuracy of first derivative computing. In order to improve the pressure distribution, Zhang et al. (2006) introduced a combined source term for Poisson equation, further more Khayyer and Gotoh (2011) introduced an error-compensating terms for source term to improve the accuracy. According to the low accuracy of Laplace operator, Schwaiger (2008) and Khayyer and Gotoh (2011) give different forms for second order particle approximation, which are helpful methods to remedy the low accuracy for second order derivative. In order to avoid the second order calculation, Ma and Zhou (2009) and Zheng et al. (2014) introduces the Rankine source solution to decrease the second order of the derivatives in pressure Poisson equation. The transformed Poisson equation does not include any derivative of the functions to be solved. Using the new formulation, one just needs to approximate the functions themselves during discretization, instead of approximating their second order derivatives as in the other incompressible SPH, which is abbreviated as ISPH_R in this paper.

Ultimately, all incompressible SPH methods need to solve sparse linear system in pressure Poisson equation. Solving large sparse matrix systems is of great significance, which can meet great challenge even in mesh base method. In many practical applications, the coefficient matrix might be ill-conditioned and challenging for iterative methods. Since one of the main bottlenecks in the process of solving such linear systems is always high computational cost. In addition, the solution of linear system requires more simulation time when numerical models are large and highly heterogeneous. The coefficient matrices of large-scale sparse linear systems are nonsingular and they have two distinctive characteristics. The first one is that the size of linear system is very large. Many of them have millions of rows. The second one is the matrix from different discretized form are sparse, and whose patterns are determined by discretized form and boundary handling conditions. Our goal is to investigate a suitable iterative solver, which may contain some fast iterative solvers as a preconditioner.

The ISPH_R also meets the problem of large sparse linear matrix solution, which is the most expensive part for numerical calculation. The sparse matrix structure is more complex, as the neighbor particles are not fixed and can be changed as time stepping. As there is no paper focused on the comparison of different iteration solutions especially for ISPH method, this paper gives a pioneer work for iterative solvers for these particle methods. Furthermore, with the effects of solid boundary condition, the pressure Poisson equation generated by ISPH is an unsymmetrical linear matrix. It is very suitable to apply the iteration method to solve these sparse linear matrixes. One option of sparse linear matrix solvers is stationary iterative method, such as Jacobi method, Gauss-Seidel method and the Successive over-relaxation (SOR) method. While these methods are simple to derive and implement, convergence is only guaranteed for a limited class of matrices. Krylov subspace methods are a strand of most commonly used iterative method. These techniques of Krylov subspace methods are based on projection processes, which can be divided to two groups. One is based on the Lanczos biorthogonalization, like CGS, BiCG, BiCGstab. Other one is based on the Arnoldi orthogonalization, like Gram-Schmidt (GS), Modified Gram-Schmidt (MGS), Modified Gram-Schmidt with reorthogonalization (MGSR), Householder (HO) and Generalized Minimum Residual Method (GMRES) (Saad, 2003).

These techniques require the computation of some parameters depending on the spectrum of the matrix. As the Incomplete Cholesky decomposition Conjugate Gradient (ICCG) method was first introduced for Poisson equation iteration calculation by Koshizuka et al. (1999), it is suitable for symmetrical sparse matrix solution. But the sparse matrixs of ISPH used in this paper are nonsymmetical sparse matrix, so this method is not included. Shao and Lo Edmond (2003) introduced a preconditioned conjucate gradient (PCG) to solve the pressure Poisson equation efficiently. Lee et al. (2008) introduced a BI-CGSTAB method to solve the linear matrix and without preconditioner. Xu et al. (2009) solved the linear matrix by using a BI-CGSTAB with a Jacobi preconditioner. Scale Conjucate Gradient (SCG) method is applied for pressure calculation (Hori et al., 2011). Liu et al. (2013) employed a parallel direct sparse solver call PARDISO (in Intel Math Kernel Library) to solve the pressure Poisson equation. In order to shown the properties of typical iteration methods, CGS (Sonneveld, 1989), Bi-CGstab (Van der Vorst, 1992) and GMRES (Saad and Schultz, 1986) are chosen, which are the most popular methods for large sparse matrix solution. There are many different types of CGS, Bi-CGStab and GMRES (Sonneveld and Van Gijzen, 2009; Sleijpen and Fokkema, 1993; Saad, 2003; Mittal and Alaurdi, 2003; Vogel, 2007; Fujino, 2002; Spyropoulos et al., 2004), which are in different ways to making more efficient use of a related information. It is better to do further investigation of different typical CGS, Bi-CGstab or GMRES, but its variable improvement methods will not be shown at present. Although these iterative methods are not new for solving Poisson equation, the comparison and their convergent features for ISPH_R method have not be discussed so far in literature. The results of this paper will also help us improving the efficiency of computation and forthcoming parallel computation.

This paper is organized as follows. In Section 2, it introduces the governing equations and mathematical

formulations of the ISPH_R method. In Section 3, the discretization of the pressure Poisson equation is introduced, which includes free surface particle identification and solid boundary condition. In Section 4, the basic steps of CGS, Bi-CGstab and GMRES are discussed. In Section 5, comparison of different iteration methods and analysis are given by still water simulation, which includes the comparison of iteration accuracy, CPU time, preconditioner and tolerance effects. The paper then presents the numerical tests and discussions for several cases, which includes dam breaking, violent tank sloshing and solitary wave slamming in Section 6.

2. ISPH methodology

The formulation of the SPH is generally based on the Lagrangian form of continuity equation and the Navier–-Stokes equation for compressible flow, which may be written as

$$\frac{D\rho}{Dt} + \rho\nabla\cdot\mathbf{u} = 0 \tag{1}$$

$$\frac{D\mathbf{u}}{Dt} = -\frac{1}{\rho}\nabla p + \mathbf{g} + \nu\nabla^2\mathbf{u} \tag{2}$$

where ρ is the fluid density, $\mathbf{u}$ is the fluid velocity, t is the time, p is the fluid pressure, $\mathbf{g}$ is the gravitational acceleration, and ν is the kinematic viscosity. In the incompressible SPH method, the fluid density is considered as a constant, and as a result, the continuity equation can be written as

$$D\rho/Dt = 0 \quad \nabla\cdot\mathbf{u} = 0 \tag{3}$$

The computation in the ISPH method is composed of two basic steps. The first step is a prediction, in which the velocity field is computed without imposing incompressibility. The second step is a correction in which incompressibility is enforced, leading to the Poisson equation for solving pressure. More details can be found in Shao et al. (2006). Summary will be given below.

(a) Prediction step

Assuming that velocities and positions of particles at time t have been found, their velocities and positions at $t + \Delta t$ are first predicted by considering gravitational term and viscous term in Eq. (2) using the following equations

$$\mathbf{u}_* = \mathbf{u}_t + \Delta\mathbf{u}_* \tag{4}$$

$$\Delta\mathbf{u}_* = \left(\mathbf{g} + \nu\nabla^2\mathbf{u}\right)\Delta t \tag{5}$$

$$\mathbf{r}_* = \mathbf{r}_t + \mathbf{u}_*\Delta t \tag{6}$$

where $\mathbf{u}_t$ and $\mathbf{r}_t$ are the velocities and positions at time t, respectively, Δt is the time step, $\mathbf{r}_*$ and $\Delta\mathbf{u}_*$ are the predicted intermediate position and velocity of particles at the new time step.

(b) Correction step

The velocity changed during the correction step is estimated by

$$\mathbf{u}_{**} = -\frac{\Delta t}{\rho}\nabla p_{t+\Delta t} \tag{7}$$

where $p_{t+\Delta t}$ is the pressure at $t + \Delta t$. The velocities and positions of particles at $t + \Delta t$ are then given by

$$\mathbf{u}_{t+\Delta t} = \mathbf{u}_* + \mathbf{u}_{**} \tag{8}$$

$$\mathbf{r}_{t+\Delta t} = \mathbf{r}_t + \frac{\mathbf{u}_t + \mathbf{u}_{t+\Delta t}}{2}\Delta t \tag{9}$$

Combining Eqs. (8) with (3), one obtains the following equation for pressure

$$\nabla^2 p_{t+\Delta t} = \frac{\rho\nabla\cdot\mathbf{u}_*}{\Delta t} \tag{10}$$

Similarly, Shao and Lo Edmond (2003) proposed a projection-based incompressible method to impose density invariance Eq. (10), which leading to the equation below

$$\nabla\cdot\left(\frac{1}{\rho^*}\nabla p_{t+\Delta t}\right) = \frac{\rho - \rho^*}{\rho\Delta t^2} \tag{11}$$

where ρ^* is the density at the intermediate time step and can be estimated by $\rho^* = \sum_{j=1,}^{N} m_j W_{ij}$. For the incompressible fluids, the intermediate density is not much different from the specified fluid density. As indicated by Hu and Adams (2007), Eqs. (10) and (11) are equivalent and both valid for incompressible fluids theoretically. They suggested solving the two incompressibility equations simultaneously. The solution of the density invariant equation (Eq. (11)) was used to adjust the positions of particles while the solution of the velocity-divergence-free equation (Eq. (10)) was used to adjust their velocity. In contrast, Zhang et al. (2006) used the mixed one given below

$$\nabla^2 p_{t+\Delta t} = \gamma\frac{\rho - \rho^*}{\Delta t^2} + (1-\gamma)\frac{\rho\nabla\cdot\mathbf{u}_*}{\Delta t} \tag{12}$$

which was also used by Ma and Zhou (2009) for the MLPG_R method, where γ is the artificial value and in the range of 0–1. According to numerical tests presented in Ma and Zhou (2009) and also suggested by Zhang et al. (2006), the results for violent water waves obtained by using Eq. (12) seems to be better if γ is specified a proper small value than those for $\gamma = 0$ (velocity-divergence-free equation). $\gamma = 0.01$ is used for all numerical tests in this paper.

3. Poisson equation discretization and boundary conditions

The main difference between the ISPH method and the ISPH_R method lies in the approach to discretization of the pressure Poisson equation defined in Eq. (12). In other ISPH method, the Laplace operator in Eq. (12) is directly

approximated like in finite different methods. There are different order schemes available as reviewed and discussed by Zheng et al. (2014). No matter which scheme is used, there is always a difficulty with accurately modelling the functions to be solved, in particular when neighbour particles are distributed in a disorderly manner. Distribution of particles always becomes disorderly when modelling violent waves even they are regularly distributed at the start of simulation. Therefore, it is obviously advantageous to eliminate use of direct numerical approximation to second derivatives when solving the pressure Poisson equation in the ISPH formulation. Ma and Zhou (2009) have presented a new method. The main idea of the new approach comes from another meshless method called as the Meshless Local Petrov-Galerkin Method based on Rankine Source Solution (MLPG_R), that is reformulating Eq. (12) into another form which does not include any derivative of pressure and velocity. For this purpose, Eq. (12) is integrated over a small sub-domain Ω_i(to be distinctive, notation of particles for the ISPH_R method is denoted by capital i or j) surrounding a particle after multiplication by the Rankine source solution φ, and then it reads

$$\int_{\Omega_i} \varphi \nabla^2 p_{t+\Delta t} d\Omega_i = \int_{\Omega_i} \left[\gamma \frac{\rho - \rho^*}{\Delta t^2} + (1-\gamma) \frac{\rho}{\Delta t} \nabla \cdot \mathbf{u}_* \right] \varphi d\Omega_i \tag{13}$$

where φ can be chosen as

$$\varphi = \frac{1}{2\pi} \ln(r/R_i) \text{ for 2D cases} \tag{14}$$

that satisfies $\nabla^2 \varphi = 0$, in Ω_i except for the center and $\varphi = 0$, on $\partial\Omega_i$, which is the boundary of Ω_i and R_i is its radius. The radius is usually smaller than the distance between two particles. After some mathematical manipulations, Eq. (13) becomes the following form

$$\int_{\partial\Omega_i} \mathbf{n} \cdot (p_{t+\Delta t} \nabla \varphi) dS - (p_{t+\Delta t})_i = \gamma \frac{\rho_i - \rho_i^*}{\Delta t^2} \frac{R_i^2}{4} + (1-\gamma) \int_{\Omega_i} \frac{\rho}{\Delta t} \mathbf{u}_* \cdot \nabla \varphi d\Omega \tag{15}$$

which will be applied to each of inner particles. More details of mathematical manipulations can be found in Ma and Zhou (2009). It has been noted that the increment of the density $\rho - \rho^*$ assumed to a constant within the sub-domain and so equal to its value at Particle i when Eq. (13) is derived. This may not cause unacceptable error. Not only because the density should not change much due to the change in the intermediate position of the particle as pointed above, but also because the small error caused due to the assumption is further reduced by multiplying the coefficient γ that is normally chosen in a range of 0–0.3, which is taken as 0.1 in this paper. The term may be evaluated in the same way as that for the second term but such a way will not improve the accuracy significantly due to the reasons discussed here.

For the ISPH_R method, with the approximation to pressure, $p(\mathbf{r}_i) \approx \sum_{j=1}^{N} \Phi_j(\mathbf{r}_j) p_j$, Eq. (15) becomes

$$\mathbf{A} \cdot \mathbf{P} = \mathbf{B} \tag{16}$$

The entries of $\mathbf{A}$ and $\mathbf{B}$ are given, respectively, by

$$A_{ij} = \begin{cases} \int_{\partial\Omega_I} \Phi_j(\mathbf{r}_j) \mathbf{n} \cdot \nabla \varphi ds - \Phi_i(\mathbf{r}_i) & \text{for inner nodes} \\ \Psi_{ij} & \text{for solid boundary nodes} \end{cases} \tag{17}$$

$$B_i = \begin{cases} \alpha \frac{\rho_i - \rho_i^*}{\Delta t^2} \frac{R_i^2}{4} + (1-\alpha) \int_{\Omega_i} \frac{\rho}{\Delta t} \overrightarrow{u}^* \cdot \nabla \varphi d\Omega & \text{for inner nodes} \\ \frac{\rho}{\Delta t} \overrightarrow{\mathbf{n}} \cdot \left(\overrightarrow{\mathbf{u}}_* - \overrightarrow{\mathbf{U}}^{n+1} \right) & \text{for solid boundary nodes} \end{cases} \tag{18}$$

where Ψ_{ij} is given below. When forming the above equations, the pressure at the free surface particles has been imposed to be zero, which is shown as

$$p = 0 \tag{19}$$

according to Eq. (17). In Eq. (18), one needs to evaluate the integrals at each particle over its sub-domain. This potentially takes significant computational time but the semi-analytical technique suggested by Ma and Zhou (2009) helps reducing the costs considerably and it is adopted in this paper.

On solid boundaries, the following conditions should be satisfied

$$\mathbf{u} \cdot \mathbf{n} = \mathbf{U} \cdot \mathbf{n} \tag{20}$$

and

$$\mathbf{n} \cdot \nabla p = \rho \left(\mathbf{n} \cdot \mathbf{g} - \mathbf{n} \cdot \dot{\mathbf{U}} + \nu \mathbf{n} \cdot \nabla^2 \mathbf{u} \right) \tag{21}$$

where $\mathbf{n}$is the unit normal vector of the solid boundaries, $\mathbf{g}$ is the vector of gravitational acceleration, $\mathbf{U}$ and $\dot{\mathbf{U}}$ are the velocity and acceleration of the solid boundaries, respectively.

It is obvious that one must compute the term $\nabla^2 \mathbf{u}$ when applying this condition in Eq. (21), which needs to estimate the second order derivative at the rigid boundary. To avoid the computation of the second order derivative in the equation, Ma and Zhou (2009) combined Eqs. (5) with (21) and gave an alternative as follows:

$$\mathbf{n} \cdot \nabla p = \frac{\rho}{\Delta t} \mathbf{n} \cdot \left(\mathbf{u}^* - \mathbf{U} \right) \tag{22}$$

This one is used in this paper.

The condition on the free surface is very simple, which is stated that the pressure of water on its free surface is equal to the atmospheric pressure, which can be taken as zero as shown

in Eq. (19). In the traditional SPH method, this condition is automatically satisfied as long as the density on the free surface is estimated correctly. However, in the incompressible SPH method, this condition has to be imposed when solving the boundary value problem defined above. In order to impose this condition, one needs to know which particles are on the free surface. This is not a problem for non-broken water waves, where the water particles on the free surface at start always remain on the free surface and does not need to be identified during simulation. However, for breaking or violent water waves, the particles on the free surface at start can become inner particles and inner particles can become the free surface particles during a simulation. Therefore, the free surface particles have to be identified at every time step after wave breaking occurs. In this paper free surface particles are identified by density and three auxiliary functions, as tested by Zheng et al. (2014). This technique can give significant improvement on identifying the particles on the free surface. It is noted nevertheless that a few particles near the free surface may still be identified as free surface particles but such incorrect identification may not lead to significant error on pressure. That is because the pressures of these particles are very close to the pressure on the free surface. The following section will focus on the discussion what methods would be better to solve Eq. (17).

After get the discretized form of Eq. (16), the next work is focused on how to solve it efficiently, which is also the most key problem for solving sparse linear matrix. Although this problem appeared in many meshed based problems and had done many researches on improving its computation costs and speed, but it is still in open discussion. It is more difficult in particle-based method, as the neighbour particles are not fixed and can be changed with time stepping, elements in each row may reach 20–30 in 2D cases and 40–60 in 3D cases, which are more complex than normal mesh-based method. The target of this paper will give some numerical tests of different typical iteration methods and some useful advice for utility of ISPH_R method, which can also be applied to other particle methods.

4. Different iterative schemes

The particle discretization for Poisson pressure equation leads to a large, sparse and unsymmetrical system of linear equations. Iterative schemes are usually employed for solving such a system. There are many iterative methods available but the question is open about which one is the better for solving the linear system associated with ISPH_R method. The main aim of this paper is to compare three schemes. Biconjugate Gradient Square (CGS) method (Sonneveld, 1989) is the first coming Krylov subspace method. Biconjugate Gradient Stabilized (Bi-CGstab) method (Van der Vorst, 1992) is the most important iterative method for Krylov subspace methods based Lanczos biorthogonalization. Generalized Minimal Residual (GMRES) method (Saad and Schultz, 1986) is other typical Krylov subspace methods based on the Arnoldi orthogonalization. Their calculation efficiency and the convergent rate

Table 1
CGS.

Step 0 Construct a preconditioner $\mathbf{K}$ for a linear equations $\mathbf{Ax} = \mathbf{b}$
Step 1 Solve $\mathbf{Kx}^{(0)} = \mathbf{b}$ for $\mathbf{x}^{(0)}$
Step 2 Compute $\mathbf{r}^{(0)} = b - \mathbf{Ax}^{(0)}$, where $\mathbf{r}$ is the residual vector
Step 3 Set $\mathbf{P}^{(0)} = \mathbf{u}^{(0)} = K^T\mathbf{r}^{(0)}$
Step 4 For $n = 0, 1, 2,\ldots$ carry out the following computations
4.1 Compute $\alpha^{(n)} = (r^{(n)}, r^{(0)})/(Ap^{(n)}, r^{(0)})$, $q^{(n)} = u^{(n)} - \alpha^{(n)}Ap^{(n)}$
4.2 Compute$d^{(n)} = u^{(n)} + q^{(n)}$, $\mathbf{x}^{(n+1)} = \mathbf{x}^{(n)} + \alpha^{(n)}d^{(n)}$
4.3 Compute $r^{(n+1)} = r^{(n)} - \alpha^{(n)}Ap^{(n)}$
4.4 Check convergence $\left\|\mathbf{r}^{(n)}\right\|_2 \le RTOL\left\|\mathbf{r}^{(0)}\right\|_2 + ATOL$, if not proceed
4.5 Compute $\beta^{(n)} = (r^{(n+1)}, r^{(0)})/(r^{(n)}, r^{(0)})$,
4.6 Compute $u^{(n+1)} = r^{(n+1)} + \alpha^{(n)}Ad^{(n)}$,
4.7 Compute $p^{(n+1)} = u^{(n+1)} + \beta^{(n)}(q^{(n)} + \beta^{(n)}p^{(n)})$

Return to step 4.

Table 2
Bi-CGstab.

Step 0 Construct a preconditioner $\mathbf{K}$ for a linear equations $\mathbf{Ax} = \mathbf{b}$
Step 1 Solve $\mathbf{Kx}^{(0)} = \mathbf{b}$ for $\mathbf{x}^{(0)}$
Step 2 Compute $\mathbf{r}^{(0)} = b - \mathbf{Ax}^{(0)}$, where $\mathbf{r}$ is the residual vector
Step 3 Set $\mathbf{P}^{(0)} = \mathbf{r}^{(0)}$, and $\overline{\mathbf{r}}^{(0)}$(for example, $\overline{\mathbf{r}}^{(0)} = \mathbf{r}^{(0)}$)
Step 4 Define $\rho^{(0)}$ as the inner product of $\overline{\mathbf{r}}^{(0)}$ and $\mathbf{r}^{(0)}$, or $\rho^{(0)} = (\overline{\mathbf{r}}^{(0)}, \mathbf{r}^{(0)})$
Step 5 For $n = 0, 1, 2, \cdots$ carry out the following computations
5.1 Solve $\mathbf{K}\overline{\mathbf{p}}^{(0)} = \mathbf{p}^{(n)}$ for $\overline{\mathbf{p}}$
5.2 Compute $\mathbf{V}^{(n)} = \mathbf{A}\overline{\mathbf{p}}$
5.3 Compute $\alpha^{(n)} = \rho^{(n)}/(\overline{\mathbf{r}}^{(0)}, \mathbf{V}^{(n)})$
5.4 Compute $\mathbf{s}^{(n)} = \mathbf{r}^{(n)} - \alpha^{(n)}\mathbf{V}^{(n)}$
5.5 Solve $\mathbf{K}\overline{\mathbf{s}}^{(n)} = \mathbf{s}^{(n)}$ for $\overline{\mathbf{s}}^{(n)}$
5.6 Compute $\mathbf{t} = \mathbf{A}\overline{\mathbf{s}}$
5.7 Compute $\omega^{(n)} = (\mathbf{t}, \mathbf{s})/(\mathbf{t}, \mathbf{t})$
5.8 Compute $\mathbf{r}^{(n+1)} = \mathbf{s} - \omega^{(n)}\mathbf{t}$
5.9 Check convergence $\left\|\mathbf{r}^{(n)}\right\|_2 \le RTOL\left\|\mathbf{r}^{(0)}\right\|_2 + ATOL$, if not proceed
5.10 Compute $\mathbf{x}^{(n+1)} = \mathbf{x}^{(n)} + \alpha^{(n)}\overline{\mathbf{p}} + \omega^{(n)}\overline{\mathbf{s}}$
5.11 Compute $\rho^{(n+1)} = (\overline{\mathbf{r}}^{(0)}, \mathbf{r}^{(n+1)})$
5.12 Compute $\beta^{(n)} = (\rho^{(n+1)}/\rho^{(n)})(\alpha^{(n)}/\omega^{(n)})$
5.13 Set $\mathbf{P}^{(n+1)} = \mathbf{r}^{(n)} + \beta^{(n)}(\mathbf{P}^{(n)} - \omega^{(n)}\mathbf{V}^{(n)})$

Return to step 5.

Table 3
GMRES.

Step 0 Construct a preconditioner $\mathbf{K}$ for a linear equations $\mathbf{Ax} = \mathbf{b}$
Step 1 Solve $\mathbf{Kx}^{(0)} = \mathbf{b}$ for $\mathbf{x}^{(0)}$
Step 2 compute $\mathbf{r}^{(0)} = \mathbf{b} - \mathbf{Ax}^{(0)}$, $\beta = \left\|\mathbf{r}^{(0)}\right\|_2$ and $\boldsymbol{\upsilon}^{(1)} = \mathbf{r}^{(0)}/\beta$
Step 3 For $n = 0, 1, 2, \cdots$ carry out the following computations
3.1 $h_{m,n} = (\mathbf{K}^{-1}\mathbf{A}\boldsymbol{\upsilon}^{(n)}, \boldsymbol{\upsilon}^{(m)})$, $m = 1, 2, \cdots, n$
3.2 $\overline{\boldsymbol{\upsilon}}^{(n+1)} = \mathbf{K}^{-1}\mathbf{A}\boldsymbol{\upsilon}^{(n)} - \sum_{m=1}^{n}(h_{m,n} - \boldsymbol{\upsilon}^{(m)})$
3.3 $h_{n+1,n} = \left\|\overline{\boldsymbol{\upsilon}}_{n+1}\right\|_2$
3.4 $\boldsymbol{\upsilon}^{(j+1)} = \overline{\boldsymbol{\upsilon}}^{(n+1)}/h_{n+1,n}$
Define $\mathbf{H}_i$ as the $(i+1) \times i$ upper Hessenberg matrix whose nonzero entries are coefficients $h_{m,n}$
Step 4. form an approximate solution $\mathbf{x}^{(i)} = \mathbf{x}^{(0)} + \mathbf{V}^{(i)}\mathbf{y}^{(i)}$

where $\mathbf{V}^{(i)} \equiv [\boldsymbol{\upsilon}_1\ \boldsymbol{\upsilon}_2 \cdots \boldsymbol{\upsilon}_i]^T$; $\mathbf{y}^{(i)} = \min_n \left\|\beta\mathbf{e}_1 - \mathbf{H}_i\mathbf{y}^{(n)}\right\|_2$ and $\mathbf{e}_1 \equiv (1\ 0\ \cdots 0)^T$
Step 5 Compute $\mathbf{r}^{(n)} = \mathbf{b} - \mathbf{Ax}^{(n)}$
Step 6 Check convergence $\left\|\mathbf{r}^{(n)}\right\|_2 \le RTOL\left\|\mathbf{r}^{(0)}\right\|_2 + ATOL$

If not, set $\mathbf{x}^{(0)} = \mathbf{x}^{(n)}$ compute $\mathbf{r}^{(0)} = \mathbf{b} - \mathbf{Ax}^{(0)}\beta = \left\|\mathbf{r}^{(0)}\right\|_2$ and $\mathbf{v}^{(1)} = \mathbf{r}^{(0)}/\beta$, return the step 3.

will be examined. For the completeness, the main steps of three iteration methods are shown briefly in Tables 1–3.

The **K** of three iterative schemes can be set as Jacobi preconditioner and ILU(0) of the same form. The convergence of the linear solver is achieved when the iteration number reaches the maximum iteration number, or

$$\left\|\mathbf{r}^{(n)}\right\|_2 \le RTOL\left\|\mathbf{r}^{(0)}\right\|_2 + ATOL \tag{23}$$

where $\|\cdot\|_2$ is the l_2-norm, n and 0 are for i th iteration and the initial value respectively, the linear solver tolerance $RTOL = 1.0\times10^{-6}$ and $ATOL = 1.0\times10^{-15}$.

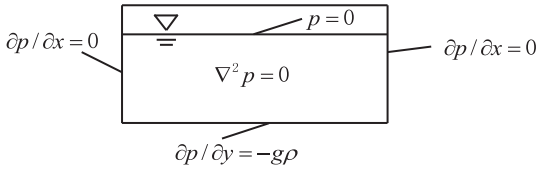


Fig. 1. Calculation domain and its boundary conditions.

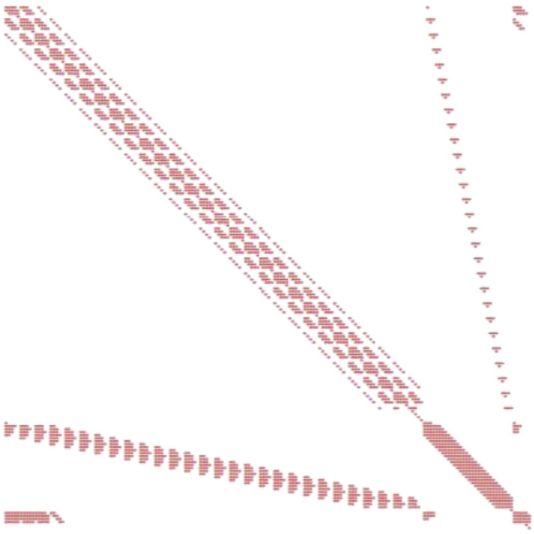


Fig. 2. Matrix elements distribution of ISPH_R (* = non-zero element, blank space = zero element, total particle number = 10*20).

5. Comparison of different iteration methods and numerical analysis

In order to give the comparison of different iteration methods in details, this section gives the simple case for p calculation. Fig. 1 gives the sketch of calculation domain and it boundary conditions. The initial $p = 0$, the length of calculation domain is $l = 1.0$ m, the height $h = 0.5$ m, $\nabla^2 p = 0$ in inner domain. According to the boundary condition, $p = -g\rho y$ is obtained by analytical solution. Fig. 2 gives the numerical matrix elements distribution of ISPH_R.

Fig. 3 (a) gives the pressure distribution of whole calculation domain by GMRES method when tolerance error $RTOL = 10e{-}6$. Fig. 3 (b) gives the comparison of different iteration methods for pressure distribution when $x = 0.5$ m. In order to show the accuracy of different iteration methods, Table 4 gives the comparison of different particle numbers in vertical direction Ny and accuracy comparisons of different iteration methods. According to the comparison of Table 4, at the initial stage iteration there are some differences of the accuracy for different iteration methods. According to the comparisons of Table 4, GMRES method can get the highest accuracy among these three iteration methods.

In order to show the iteration steps of different iteration methods, Table 5 gives the comparison of different iteration

Table 4
Comparison of different iteration methods by different particle numbers.

Particle number Ny	Analytical results	CGS	BiCGStab	GMRES
30	0.48333	0.48355	0.48359	0.48345
40	0.48750	0.48855	0.48854	0.48777
50	0.49000	0.49026	0.49023	0.49012

Table 5
Comparison of iteration steps of different iteration methods and different preconditioner types Ny = 50.

Preconditioner type	CGS (CPU time)	BiCGStab (CPU time)	GMRES (CPU time)
No precondtioner	163 (0.0223 s)	151 (0.0211 s)	107 (0.0194 s)
Jacobi precondtioner	111 (0.0204s)	101 (0.018 s)	87 (0.0172 s)
ILU(0) preconditioner	56 (0.0189 s)	45 (0.0171 s)	32 (0.0162 s)

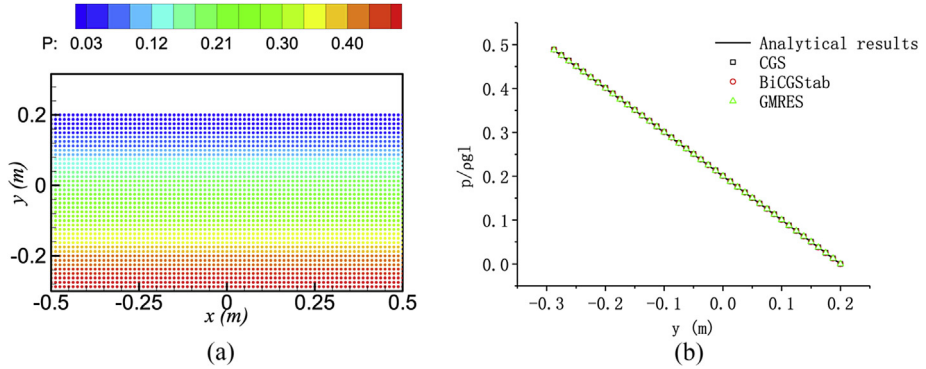


Fig. 3. Pressure distribution by different iteration methods when tolerances error $RTOL = 10e{-}6$: a) Total pressure distribution Ny = 40; b) Comparison of different iteration methods when $x = 0.5$ m.

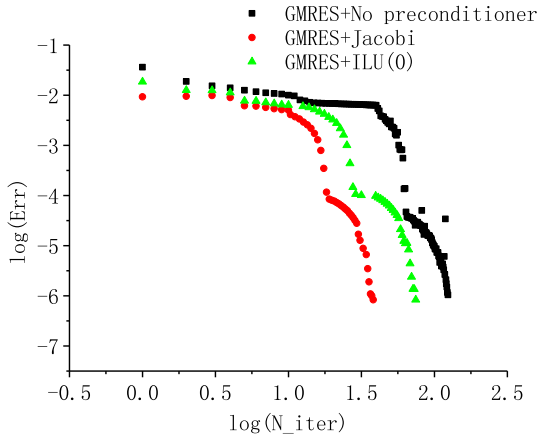


Fig. 4. Pressure distribution by different iteration methods when tolerances error *RTOL* = 10e−6.

Table 6
Comparison of calculation accuracy of different iteration methods with different tolerances when Ny = 50.

Tolerence (*RTOL*)	Analytical results	1.0e−5	1.0e−6	1.0e−10	1.0e−15
CGS	0.49000	0.49031	0.49027	0.49026	0.49026
BiCGStab	0.49000	0.49028	0.49026	0.49025	0.49025
GMRES	0.49000	0.49013	0.49012	0.49012	0.49012

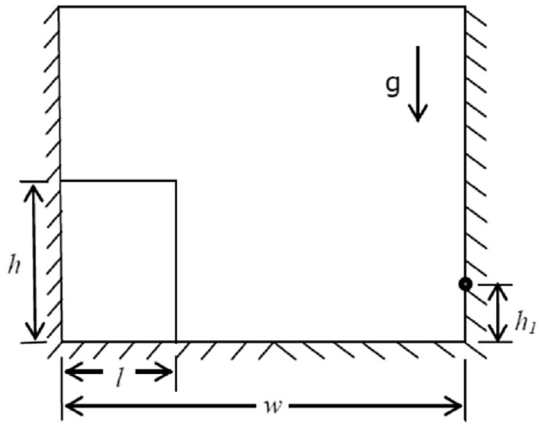


Fig. 5. The sketch of dam breaking model.

steps of different iteration methods and different preconditioner when Ny = 50. In order to show the convergence curve of iterative tolerance, Fig. 4 gives the comparison of convergence tests of different preconditioner for the GMRES as an example, and *RTOL* = 10e−6. In Fig. 4 N_iter is the iteration step number and Err is the value of $\mathbf{r}^{(n)}$ in Table 3. According the results of Fig. 4, with the help of suitable preconditioner, GMRES can get fast convergence speed and less iteration steps. According to the results of Table 5, GMRES method can get the least iteration steps. Furthermore, preconditioner is helpful for decreasing the iteration steps. According to the comparison of no preconditioner, Jacobi preconditioner and ILU(0) preconditioner, ILU(0) can get the least iteration steps. Although many different iteration methods can be set as the preconditioner, Jacobi and ILU(0) are the most popular and typical precondtioners. The comparisons of more complex preconditioner are not included at present, which can be done in further investigation.

According to the comparisons of CPU time in Table 5, ILU(0) can get the fastest convergence speed. As the calculation process of ILU(0) is more difficult than Jacobi preconditioner, so the CPU time of ILU(0) preconditioner is a litter bit more than the ones of Jacobi compared with the a large decreasing of iteration steps. In order to show the effects of *RTOL*, Table 6 shows the accuracy of different *RTOL* by different iteration methods, and in this case the preconditioner is set as ILU(0). According to the comparison of different *RTOL*, when *RTOL* < 1.0e−6, *RTOL* does not affect the accuracy of last results obviously. The rules are almost the same for these three iteration methods. So based on the series comparison of different iteration methods and different preconditioners, the preconditioner is set as ILU(0) and the *RTOL* = 1.0e−6 during the part of numerical tests.

6. Numerical tests and analysis

6.1. Dam breaking

Dam breaking is often used as a benchmark for violent free surface flow. In this numerical test, a rectangular water column is confined by bottom, top wall and two vertical walls, as illustrated in Fig. 5. The width of the water column is *l* and its

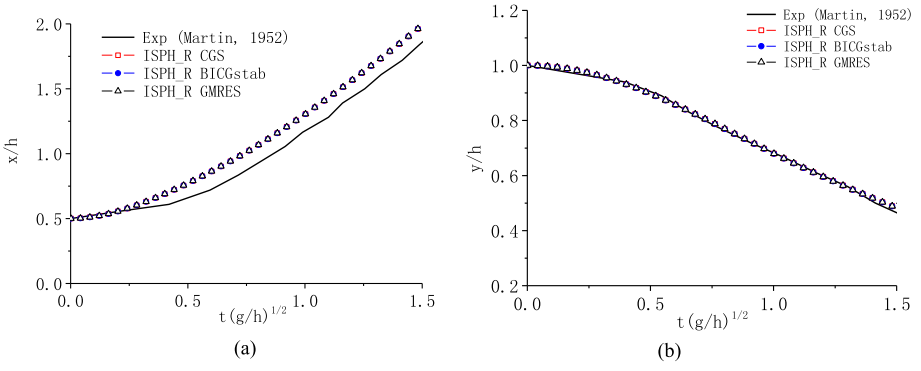


Fig. 6. Comparison of different iteration methods with experimental results: (a) wave front on the bottom wall (b) wave height on vertical wall.

height is h. At the beginning of the computation, the dam is instantaneously removed and the water collapses and flows out along a dry horizontal bed. w is the distance between two vertical walls. There is one pressure sensors p_1 on the right vertical wall, the height of p_1 from the bottom is h_1. In this section, all variables and parameters are non-dimensionalised using h and g, such as $t = \tilde{t}\sqrt{g/h}$ unless mentioned otherwise.

Although the case of non-breaking waves is not necessarily dealt with by the ISPH, it is used here for preliminary validation of different iteration methods. In the first case, $h = 1.0$ m, $l/h = 0.5$, $w/h = 4.5$, the water collapses, just flows out along a dry horizontal bed, and stops before the wave front hits the right vertical wall. There are some experimental data for wave front and wave height on left vertical wall (Martin and Moyce, 1952). Fig. 6 gives the comparison of wave fronts and wave heights obtained by three iteration methods using 100*200 particles. There is very little difference among three methods. The numerical results of the ISPH_R can get a good agreement with experimental ones by using the three iteration schemes. Although it can be found some difference on this case of wave front, it is as justified by other mesh based methods, which is shown in Zheng et al. (2014).

The pressure distribution and time histories are then examined. For the case with$h = 1.0$ m, $l/h = 2.0$, $w/h = 5.366$, there is a pressure sensor installed on the right vertical wall and with the height of $h_1/h = 0.1833$. Fig. 7 gives the results of dam breaking profiles and pressure distribution at different times. In this case the particle number is 20,000, time stepping

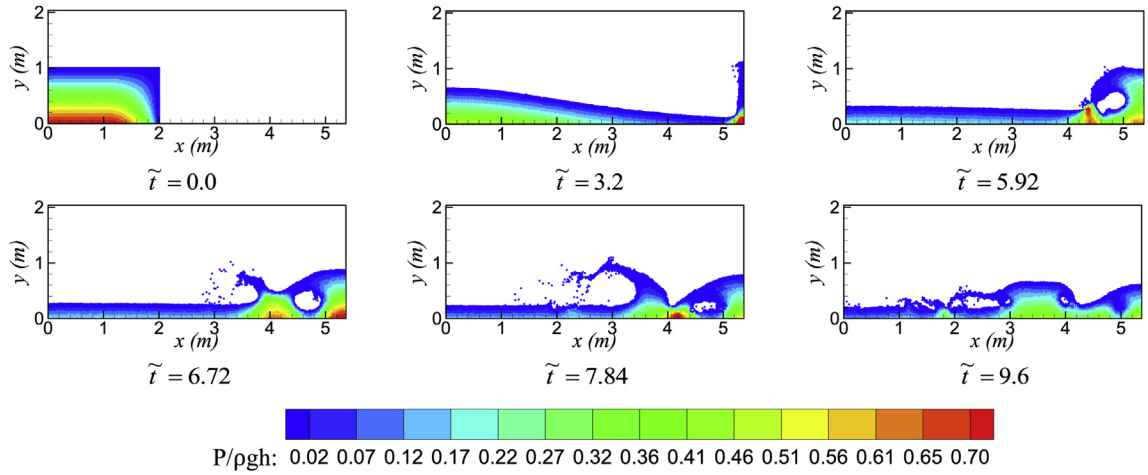


Fig. 7. Snapshot of wave profile and pressure distribution of dam breaking at different time.

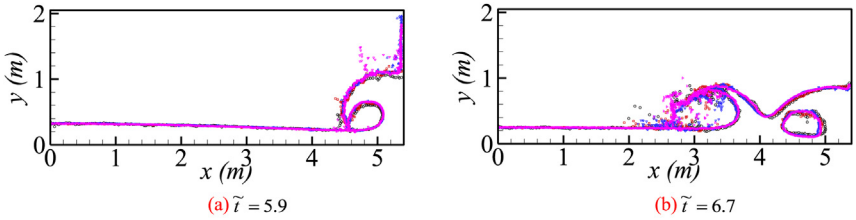


Fig. 8. Convergence test of wave profiles according to different particle numbers by GMRES: ○ N = 40*80, □ N = 60*120; △ N = 80*160; ▽ N = 100*200.

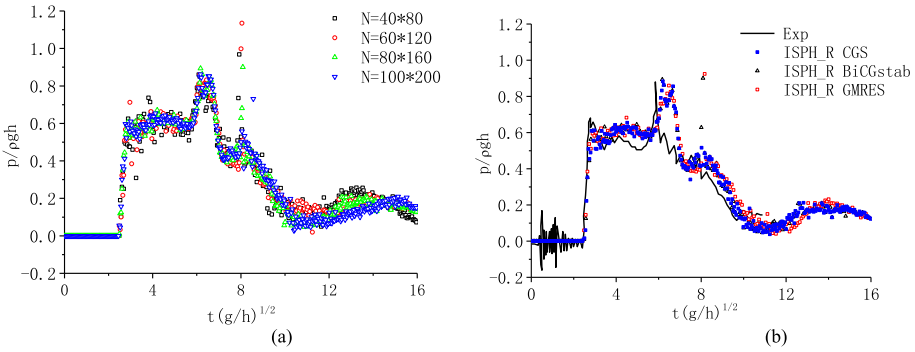


Fig. 9. Comparison of pressure time history of ISPH_R by different iteration methods: (a) Convergence test of different particle numbers by GMRES; (b) Comparison of CGS, Bi-CGstab, GMRES and experimental data, N = 100*200.

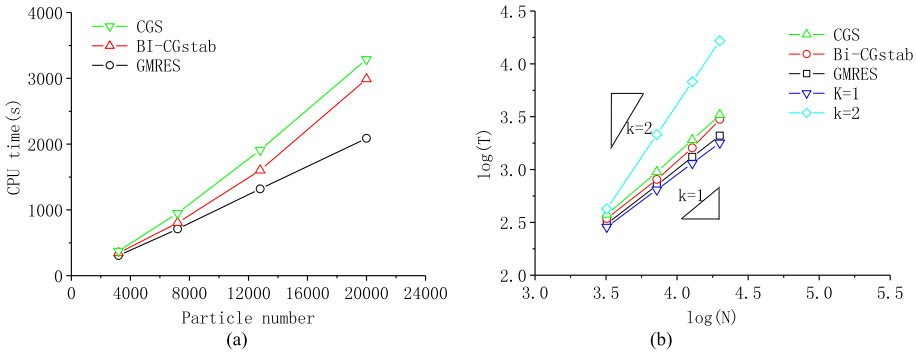


Fig. 10. Comparison of CPU time for dambreaking simulation by using different iteration methods: (a) CPU-time comparison, (b) Convergence-rate comparison.

Table 7
CPU time comparison for dam breaking simulation.

Particle number		40*80	60*120	80*160	100*200
CPU time(s)	GMRES	305.8	709.4	1328.3	2082.2
	Bi-CGstab	343.6	804.1	1602.2	2987.7
	CGS	375.2	950.2	1910.2	3289.2

length $d\tilde{t} = 0.008$, the iteration method is GMRES as example. Fig. 8 gives the results of convergence tests on wave profiles according to different particle numbers. Fig. 9 gives the comparison results of impact pressure at point p_1 by different iteration methods. Fig. 9 (a) is the convergence tests of pressure time histories by different particle numbers by GMRES method. Fig. 9 (b) is the comparison of numerical results with experiment data (Colagrossi and Landrini, 2003) when particle number N = 20,000. There is very little difference between pressure time histories obtained by these three iteration methods. The results of pressure time histories have a good agreement with experimental data.

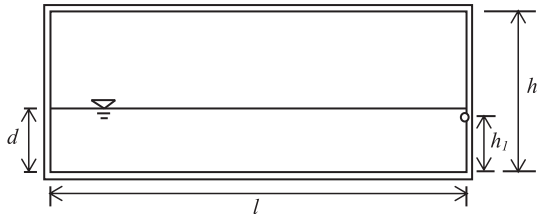


Fig. 11. Sketch of sloshing tank.

All codes run on the same computer, the CPU is Xeon E5-2665, 2.4 GHz, and the RAM is 16.0 GB. Fig. 10 gives the comparison of CPU time corresponding to the particle numbers. In this figure, T is the CPU time and N is the particle number. When the particle number is small, e.g., 3200, the difference in the results from the three iteration methods is not very large. With the particle number increasing, such as 20,000, the CPU time of GMRES is 0.697 of that for the Bi-CGstab and 0.633 of that of the CGS. According to the results of Fig. 6 (b), in the case of the dam breaking simulation, the convergent rate of three iteration methods is between the first order and second order. Table 7 gives the details of CPU time of three iteration methods. It can be seen that the GMRES can save more CPU time for the cases with more particles.

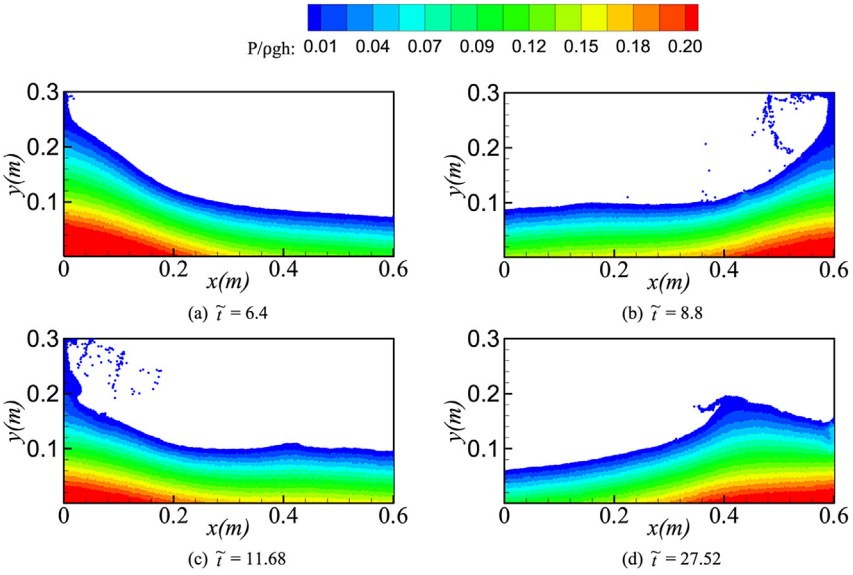


Fig. 12. Wave profiles and compress distribution of different time by GMRES method when N = 50*250.

6.2. Violent water sloshing

In this section, violent sloshing flow will be considered. The geometry of this tank is rectangle with $l = 0.6\ m$, $h = 0.5l$, $d = 0.4h$. The tank motion in sway displacement is given by $X_s = a_0(1 - \cos \Omega t)$, where a_0 and Ω are the amplitude and frequency of excited motion respectively. The parameters for this case are taken as $a_0 = 0.05$ m and $T_0 = 2\pi/\Omega = 1.5$ s, which are the same as those in Kishev et al. (2006) and is shown in Fig. 11. The behaviors of the ISPH_R method are further examined by modeling this case. For this purpose, different numbers of particles are employed with $N = 2000$, 4500, 8000 and 12,500 respectively. Time step length is $d\tilde{t} = 0.008$. Fig. 12 gives the free surface profiles and pressure distribution at different time instants by GMRES when $N = 8000$. Fig. 13 gives the convergence test of free surface profile at different time by different particle numbers. According to the comparison, the free surfaces for different number of particles are the almost same when breaking does not occur. However when breaking happens, there exists some differences between the profiles for different particle numbers. Fig. 14 (a) gives the convergence test of pressure time histories for different particle numbers with the GMRES method at Point $h_1/l = 0.1667$ on the left wall. The pressure time histories obtained by different particle numbers do not show significant differences, though there is some little difference near wave impact peaks. Fig. 14 (b) shows the comparison of pressure time histories of experimental data (Kishev et al., 2006) and numerical results obtained by CGS, BI-CGstab and GMRES respectively. It is noted that the experimental

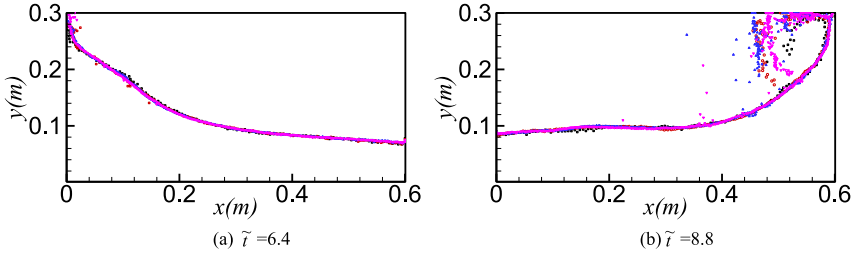


Fig. 13. Convergence test of free surface profile by GMRES method for violent water sloshing: ○ N = 20*100, □ N = 30*150; △ N = 40*200; ▽ N = 50*250.

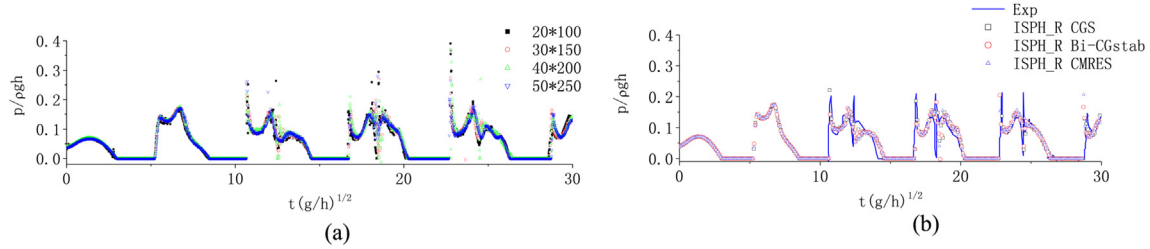


Fig. 14. Comparison of pressure time history of ISPH_R by different iteration methods for violent sloshing simulation: (a) Convergence test of pressure time history by different particle numbers with GMRES method; (b) Comparison of pressure time history of CGS, Bi-CGstab, GMRES and experimental data, N = 50*250.

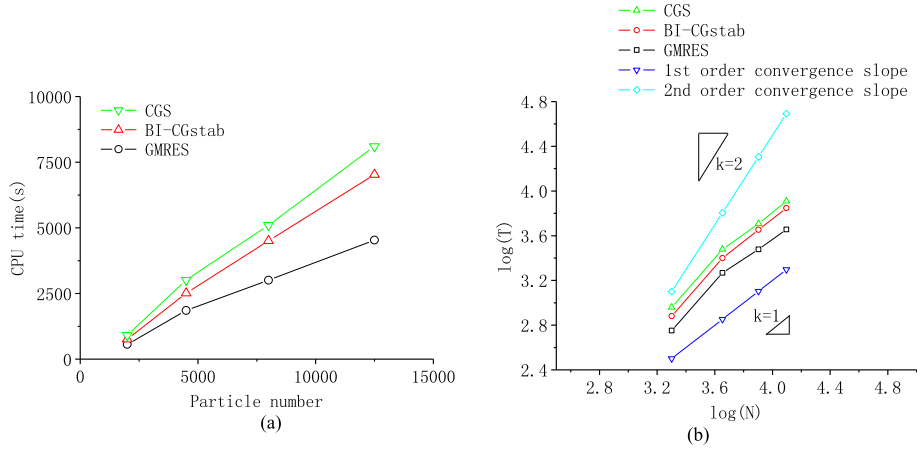


Fig. 15. Comparison of CPU time for violent water sloshing simulation by different iteration methods: (a) CPU time comparison, (b) Convergence slope comparison.

pressure time history given in Kishev et al. (2006) did not show the transient period that must exist and they did not mention whether it was recorded on the left or on right walls. To compare our results with their experimental data, the time for the experimental results in the figure has been adjusted so that the 3rd pressure peak of the numerical results corresponds to the first peak in their paper. These figures show that the results from all the methods have similar patterns to the experimental one and that the time intervals between two consecutive impacts are almost the same.

The computer used for running these cases is the same as in the previous section. Fig. 15 gives the comparison of CPU time corresponding to the different particle numbers. In the numerical tests, the CPU time of GMRES is still less than that of CGS and Bi-CGStab, and the GMRES saves more time for the cases with large particle numbers again. The convergence rates of three methods are almost the same, but the values of the GMRE are always below the ones of CGS and Bi-CGstab. Table 8 gives the details of CPU time of three iteration methods for this case.

6.3. Solitary wave slamming

This section considers the test of solitary wave slamming. The interaction between solitary wave and structures are the topic related to many coastal and ocean engineering. The solitary wave is often used to represent certain characteristics of the incident wave. In this section, the ISPH_R method is applied to the simulation of the solitary wave impact on walls with a fixed angle. The numerical results will be compared with the experimental data. Furthermore, the results of different iterative schemes are also given for CPU efficiency and precision behavior comparison.

Fig. 16 shows the geometry and setup of simulation on the cases in the section. On the left end it has a piston wave maker and on the other end it has a slope beach. In order to make a solitary wave by a piston wave maker, the analytical solution for the wave profile can be derived from the Boussinesq equation, the displacement of piston of different time is shown as

$$x_p(\tau) = \sqrt{\frac{4a}{3d}}d\left[\tanh\left\{\sqrt{3a/4d^3}\,[c\tau - x(\tau) - \lambda]\right\} + \tanh\left(\lambda\sqrt{3a/4d^3}\right)\right] \quad (24)$$

where x_p is piston displacement, a is wave amplitude, d is water depth and $c = \sqrt{g(d+a)}$ is the solitary wave celerity, τ is the non-dimensional time. It should adopt the iterative method to get the value of x_p. In this case, h is the height of whole slope and $h = 1.0$ m, wave amplitude $a/h = 0.15$, water depth $d/h = 0.25$, the total tank length is $l/h = 10.0$, slope angle $\alpha = 150°$. In the area of slope beach, a local coordination transformation is used

Table 8
CPU time comparison for violent water sloshing calculation.

Particle number		20*100	30*150	40*200	50*250
CPU time(s)	GMRES	564.3	1853.7	3006.2	4534.5
	Bi-CGstab	760.2	2511.3	4504.9	7030.6
	CGS	910.6	3010.3	5101.2	8103.3

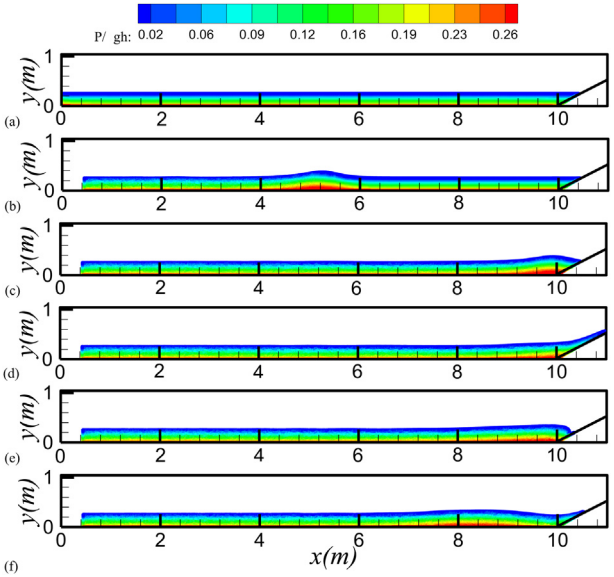


Fig. 17. Snapshots of profile and pressure distribution for solitary wave slamming of different time: (a) $\tilde{t}$=0.0; (b) $\tilde{t}$=10.5; (c) $\tilde{t}$=18; (d) $\tilde{t}$=20.4; (e) $\tilde{t}$=22.5; (f) $\tilde{t}$=24.0

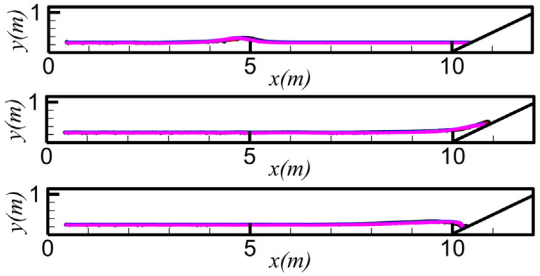


Fig. 18. Convergence test of free surface profile by GMRES method for solitary wave slamming: ○ N = 9185, □ N = 16333; △ N = 25525; ▽ N = 36760.

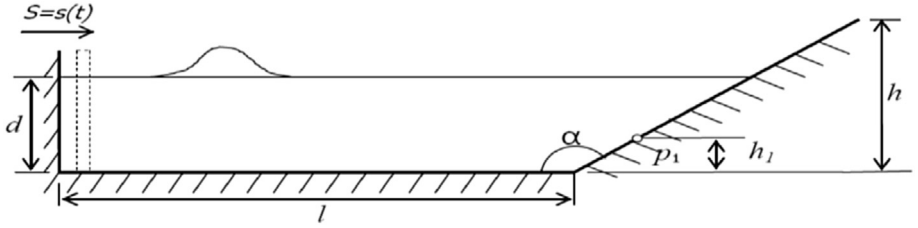
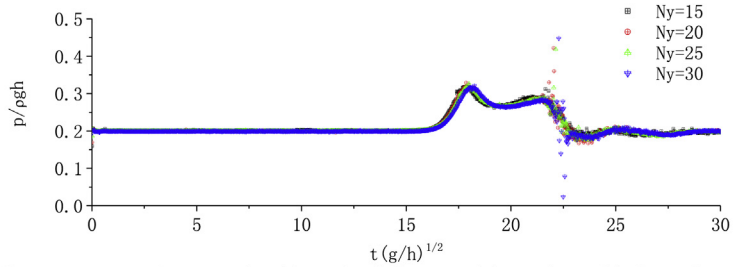
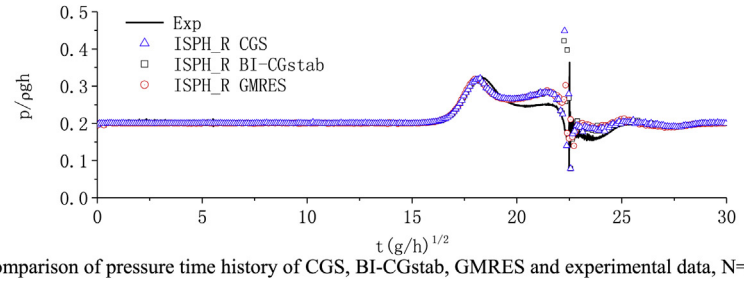


Fig. 16. The model of wave tank for solitary wave slamming.



(a) Convergence test of pressure time history by different particle numbers with GMRES method



(b) Comparison of pressure time history of CGS, BI-CGstab, GMRES and experimental data, N=36760

Fig. 19. Comparison of pressure time history of ISPH_R by different iteration methods for solitary wave slamming.

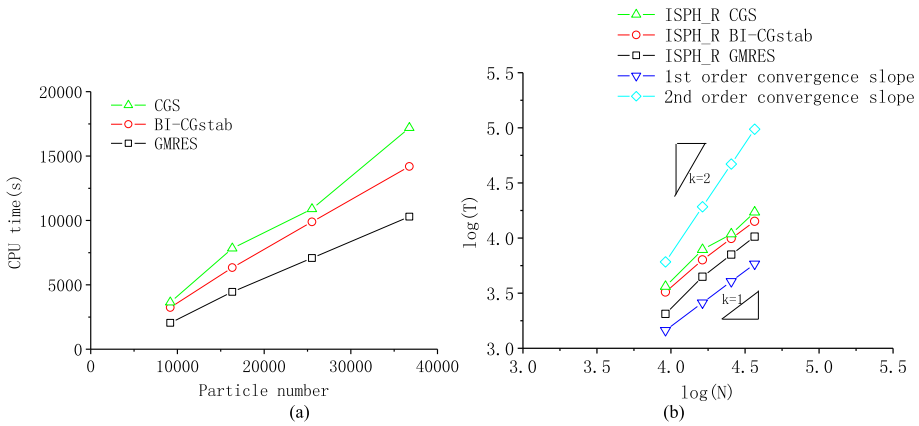


Fig. 20. Comparison of CPU time for solitary wave slamming by different iteration methods: (a) CPU-time comparison; (b) Convergence-rate comparison.

for the solid wall handling of a certain angle. Fig. 17 depicts the results of wave profile and pressure distribution at different time. The iteration method uses GMRES method as example and particle number for this case is 36,760, and the time step is $d\tilde{t} = 0.008$. Fig. 18 gives the convergence test of wave elevation by different particle numbers and the iteration method is GMRES. There is a pressure sensor fixed on the slope, the height h_1 is the vertical distance from bottom and $h_1/h = 0.05$. The same scale experiment are carried on HEU (Zheng et al., 2015). Fig. 19 gives the comparison of pressure time history at p_1 between numerical and experimental results by three iteration methods. The results of the ISPH_R have a quite good agreement with experimental ones. There still exist some oscillations in time history for $\tilde{t} = 22.5$ in experimental data, similar behaviors are observed in the results from three iteration methods. Fig. 20 depicts the comparison of CPU time for solitary wave slamming simulation by different iteration methods. The conclusion is very similar to that in previous section, i.e., the GMRES saves more time for the cases with large particle numbers and the convergence rates of both method are almost the same. More details of CPU time for different iterative methods is shown in Table 9.

Table 9
CPU time comparison for solitary wave slamming calculation.

Particle number		9185	16333	25525	36760
CPU time(s)	GMRES	2048.2	4449.0	7075.2	10289.9
	Bi-CGstab	3232.3	6335.4	9895.7	14189.1
	CGS	3638.9	7835.4	10897.9	17188.6

7. Conclusions

This paper presents a comparative study of different iteration methods for solving pressure Poisson equation for incompressible SPH based on Rankine source solution. Three iteration methods includes CGS, BI-CGstab and GMRES. The different CPU time and convergence rates are examined. The numerical test cases include still water test, dam breaking,

violent water sloshing and solitary wave slamming, which are the typical applications for ocean engineering. The numerical results are compared with experimental data. From the numerical test results and comparisons, we come to the following conclusions:

(a) With the help of one of these three linear equation solvers, incompressible SPH based on Rankine source solution can give smoothed and reliable pressure distribution.
(b) All iteration methods, CGS, BI-CGstab and GMRES have the convergence rate of between first order and second order. The preconditioner is import for different iteration methods, suitable preconditioner can decrease the iteration steps obviously. Furthermore, the tolerance of different iteration schemes can affect the accuracy of last results, but when *RTOL* is small enough (like $RTOL < 1.0e{-}6$), the iteration results may not get very improvement significantly. From the comparison of different iteration schemes, the CPU time spent by GMRES is about 0.7–0.6 time of that by Bi-CGstab and CGS.

The above conclusions are based on the series code but the study on the iteration methods in parallel calculation for ISPH_R method is still ongoing.

Acknowledgement

This work is sponsored by The National Natural Science Funds of China (51009034, 51279041, 51379051, 51579054, 51639004), Foundational Research Funds for the central Universities (HEUCDZ1202, HEUCF120113) and Defense Pre Research Funds program (9140A14020712CB01158), to which the authors are most grateful. The second author also wishes to thank the Chang Jiang Visiting Chair professorship of Chinese Ministry of Education, hosted by the HEU.

References

Bonet, J., Lok, T.S., 1999. Variational and momentum preservation aspects of smooth particle hydrodynamics formulation. Comput. Methods Appl. Mech. Eng. 180, 97–115.
Colagrossi, A., Landrini, M., 2003. Numerical simulation of interfacial flows by smoothed particle hydrodynamics. J. Comput. Phys. 191, 448–475.
Cummins, S.J., Rudman, M., 1999. An SPH projection method. J. Comput. Phys. 152, 584–607.
Ellero, M., Serrano, M., Espanol, P., 2007. Incompressible smoothed particle hydrodynamics. J. Comput. Phys. 226, 1731–1752.
Ferrand, M., Laurence, D.R., Rogers, B.D., Violeau, D., Kassiotis, C., 2013. Unified semi-analytical wall boundary conditions for inviscid, laminar or turbulent flows in the meshless SPH method. Int. J. Numer. Meth. Fluids 71, 446–472.
Fujino, S., 2002. GPBICG (m, l): a hybrid of BiCGSTAB and GPBiCG methods with efficiency and robustness. Appl. Numer. Math. 41, 107–117.
Hori, C., Gotoh, H., Ikari, H., Khayyer, A., 2011. GPU-acceleration for moving particle semi-implicit method. Comput. Fluids 51, 174–183.
Hu, X.Y., Adams, N.A., 2007. An incompressible multi-phase SPH method. J. Comput. Phys. 227 (2), 264–278.
Khayyer, A., Gotoh, H., 2011. Enhancement of stability and accuracy of the moving particle semi-implicit method. J. Comput. Phys. 230, 3093–3118.
Khayyer, A., Gotoh, H., Shao, S.D., 2008. Corrected incompressible SPH method for accurate water-surface tracking in breaking waves. Coast. Eng. 55 (3), 26–50.
Kishev, Z.R., Hu, C.H., Kashiwagi, M., 2006. Numerical simulation of violent sloshing by a CIP-based method. J. Mar. Sci. Technol. 11, 111–122.
Koshizuka, S., Oka, Y., 1996. Moving particle semi-implicit method for fragmentation of incompressible fluid. Nucl. Sci. Eng. 123 (3), 421–434.
Koshizuka, S., Ikeda, H., Oka, Y., 1999. Numerical analysis of fragmentation mechanisms in vapour explosions. Nucl. Eng. Des. 189 (1–3), 423–433.
Lee, E.S., Moulinec, C., Xu, R., Violeau, D., Laurence, D., Stansby, P., 2008. Comparisons of weakly compressible and truly incompressible algorithms for the SPH mesh free particle method. J. Comput. Phys. 227, 8417–8436.
Liu, X., Xu, H.H., Shao, S.D., Lin, P.Z., 2013. An improved incompressible SPH model for simulation of wave - structure interaction. Comput. Fluids 71, 113–123.
Lind, S.J., Xu, R., Stansby, P.K., Rogers, B.D., 2012. Incompressible smoothed particle hydrodynamics for free-surface flows: a generalised diffusion-based algorithm for stability and validations for impulsive flows and propagating waves. J. Comput. Phys. 231, 1499–1523.
Liu, M.B., Liu, G.R., 2006. Restoring particle consistency in smoothed particle hydrodynamics. Appl. Numer. Math. 56, 19–36.
Ma, Q.W., Zhou, J., 2009. MLPG_R method for numerical simulation of 2D breaking waves. Comput. Model. Eng. Sci. 43 (3), 277–304.
Martin, J.C., Moyce, W.J., 1952. Part IV: an experimental study of the collapse of liquid columns on a rigid horizontal plane. Phil. Trans. R. Soc. Lond. A 244 (882), 312–324.
Mittal, R.C., Alaurdi, A.H., 2003. An efficient method for constructing an ILU preconditioner for solving large sparse nonsymmetric linear systems by the GMRES method. Comput. Math. Appl. 45, 1757–1772.
Monaghan, J.J., 1994. Simulation free surface flows with SPH. J. Compu. Phys. 110 (4), 399–406.
Oger, G., Doring, M., Alessandrini, B., Ferrant, P., 2007. An improved SPH method: towards higher order convergence. J. Comput. Phys. 225, 1472–1492.
Rafiee, A.R., Cummins, S., Rudman, M., Thiagarajan, K., 2012. Comparative study on the accuracy and stability of SPH schemes in simulating energetic free-surface flows. Eur. J. Mech. B/Fluids 36, 1–16.
Saad, Y., 2003. Iterative Methods for Sparse Linear Systems (Second Version). Society for Industrial and Applied Mathematic.
Saad, Y., Schultz, M.H., 1986. GMRES: a generalized minimum residual algorithm for solving nonsymmetric linear systems. SIAM J. Sci. Stat. Comput. 7, 856–869.
Schwaiger, H.F., 2008. An implicit corrected SPH formulation for thermal diffusion with linear free surface boundary conditions. Int. J. Numer. Meth. Engng. 75, 647–671.
Shadloo, M.S., Zainali, A., Yildiz, M., Suleman, A., 2012. A robust weakly compressible SPH method and its comparison with an incompressible SPH. Int. J. Numer. Methods Eng. 89 (8), 939–956.
Shao, S.D., 2009. Incompressible SPH simulation of water entry of a free-falling object. Int. J. Numer. Meth. Fluids 59 (1), 91–115.
Shao, S.D., Lo Edmond, Y.M., 2003. Incompressible SPH method for simulating Newtonian and non-Newtonian flows with a free surface. Adv. Water Resour. 26 (7), 787–800.
Shao, S.D., Ji, C.M., Graham, D.I., Reeve, D.E., James, P.W., Chadwick, A.J., 2006. Simulation of wave overtopping by an incompressible SPH model. Coast. Eng. 53 (9), 723–735.
Sleijpen, G.L., Fokkema, D.R., 1993. BiCGstab(L) for linear equations involving unsymmetric matrices with complex spectrum. Electron. Trans. Numer. Analysis 1, 11–32.
Sonneveld, P., 1989. CGS, A fast Lanczos-type solver for nonsymetric linear systems. SIAM J. Sci. Stat. Comput. 10, 36–52.
Sonneveld, P., Van Gijzen, M.B., 2009. IDR(s): a family of simple and fast algorithms for solving large nonsymmetric systems of linear equations. SIAM J. Sci. Comput. 31 (2), 1035–1062.

Spyropoulos, A.N., Palyvos, J.A., Boudouvis, A.G., 2004. Bifurcation detection with the (un) preconditioned GMRES(m). Comput. Methods Appl. Mech. Eng. 193, 4707−4716.

Vogel, J.A., 2007. Flexible BiCG and flexible Bi-CGSTAB for nonsymmetric linear systems. Appl. Math. Comput. 188, 226−233.

Van der Vorst, H.A., 1992. Bi-CGSTAB: a fast and smoothly converging variant of Bi-CG for solution of non-symmetric linear system. SIAM J. Sci. Stat. Comput. 13, 631−644.

Xu, R., Stansby, P., Laurence, D., 2009. Accuracy and stability in incompressible SPH (ISPH) based on the projection method and a new approach. J. Comput. Phys. 228 (18), 6703−6725.

Zhang, S., Morita, K., Kenji, F., Shirakawa, N., 2006. An improved mps method for numerical simulations of convective heat transfer problems. Int. J. Numer. Methods Fluids 51, 31−47.

Zheng, X., Ma, Q.W., Duan, W.Y., 2014. Incompressible SPH method based on Rankine source solution for violent water wave simulation. J. Comput. Phys. 276, 291−314.

Zheng, X., Hu, Z.H., Ma, Q.W., Duan, W.Y., 2015. Incompressible SPH based on Rankine source solution for water wave impact simulation. Procedia Eng. 126, 650−654.