Simulation of Non-Linear Water Waves Using Meshless Local Petrov-Galerkin (MLPG) Method

Q.W. Ma
q.ma@city.ac.uk
School of Engineering, City University, Northampton Square, London, EC1V 0HB

Introduction

Simulation of non-linear water waves has received a numerous studies. Among them, finite element, finite volume, finite difference and boundary element methods are widely used. These methods have provided many useful and satisfactory results. However, their successes rely on good meshes. Those meshes may require a large amount of computational costs and may become over-distorted during simulation. Although the over-distortion problem may not arise if using fixed meshes, numerical diffusion due to advection terms may become severe and the motion of the floating body is not easy to cope with when such meshes are adopted.

Recently, numerical methods which do not need any grid or mesh have been developed and extended to simulate flows with a free surface (see, e.g., Dalrymple & Kino, 2001 and Lo & Shao, 2002). Among them, the smoothed particle hydrodynamics (SPH) and moving particle semi-implicit (MPS) methods have demonstrated the promising features of meshless methods to model the problems of this kind.

In addition to SPH and MPS, another meshless method, called meshless local petrove-galerkin (MLPG) method, has been proposed by Atluri and Zhu (2000) and Atluri and Shen (2002). The success of the MLPG method has been reported in solving convection-diffusion problems, fracture mechanics problems, plate bending problems and so on. In this paper, brief discussions will be given about our attempt to apply the MLPG method to simulating the nonlinear water waves.

Numerical formulations

The flow of impressible and non-viscous fluids is considered, which is governed by the following equations and conditions:

$$\nabla \cdot \vec{u} = 0 \tag{1}$$

$$\frac{D\vec{u}}{D\bar{t}} = -\frac{1}{\rho}\nabla p + \vec{g} \tag{2}$$

$p = p_{atm}$ on the free surface and

$\vec{u}.\vec{n} = \vec{U}.\vec{n}$ and $\frac{\partial p}{\partial n} = \rho\left(\vec{n}\cdot\vec{g} - \vec{n}\cdot\dot{\vec{U}}\right)$ on solid boundaries

where, ρ is the density of fluids, p the pressure, $\vec{u}$ the velocity of fluids, $\vec{U}$ the velocity of solid boundaries and $\vec{g}$ the gravitational acceleration.

It is well known that the non-viscous flow can be formulated by velocity potential. However, because our research efforts will not be restricted to non-viscous flow in future, the potential formulation is not adopted here.

To solve the above equations in Lagrangian form, a similar methodology for SPH method, as described by Lo and Shao (2002), will be used. In this method the velocities and the positions of fluids are updated at each time step by the following procedure:

1) Computing the intermediate velocities and positions:

$$\vec{u}^{(*)} = \vec{u}^{(n)} + \vec{g}\Delta t \tag{3}$$

$$\vec{r}^{(*)} = \vec{r}^{(n)} + \vec{u}^{(*)}\Delta t \tag{4}$$

2) Estimating the pressure

In order to establish the equation for the pressure, the velocity at the end of each time step is split into

$$\vec{u}^{(n+1)} = \vec{u}^{(*)} + \vec{u}^{(**)}. \tag{5}$$

It must satisfy the momentum equation (2), which, after approximating the time derivative by a finite difference, results in

$$\vec{u}^{(**)} = -\frac{\Delta t}{\rho}\nabla p^{(n+1)}. \tag{6}$$

The velocity in Equation (5) must also satisfy the continuity equation (1), leading to

$$\nabla^2 p^{(n+1)} = \frac{\rho}{\Delta t}\nabla \cdot \vec{u}^{(*)}. \tag{7}$$

This equation is used to find the pressure $p^{(n+1)}$.

2) Computing the velocity and the position at $t = t_{n+1}$

After the solution for $p^{(n+1)}$ is found, the velocities and so the positions of fluids at the end of current time step may be estimated using Equation (5) and $\vec{r}^{(n+1)} = \vec{r}^{(n)} + \vec{u}^{(n+1)}\Delta t$.

It should be noted that the above formulation is for the non-viscous flow but it is not difficulty to extend it to viscous flow by adding viscous stress terms to Equation (3).

In order to solve Equation (7), the MLPG method is used in this work. The details of MLPG method can be found in Atluri and Zhu (2000). The main idea of the method is that: using a set of nodes to represent the fluid particles in the flow domain; defining a circle sub-domain Ω_I (called integration sub-domain) for each node; deriving a weak form of differential equations over each of all the integration sub-domains; approximating unknown variables in term of a weight function which is then inserted into the weak form to form a set of algebraic equations.

Over each integration sub-domain, the weak form of the Equation (7) may be written as:

$$\int_{\Omega_I}\left[\nabla^2 p - \frac{\rho}{\Delta t}\nabla \cdot \vec{u}^{(*)}\right]\varphi d\Omega = 0 \tag{8}$$

where φ is the test function that can be arbitrary. Atluri and Shen (2002) have suggested six options for the test function. The Heaviside step function, which equals to one in the sub-domain and zero elsewhere, will be used in our work because the formulation based on this test function may be more efficient than others as suggested by them. Using the Heaviside step function as the test function, Equation (8) can be rewritten as:

$$\int_{\partial\Omega_I}\vec{n}\cdot\nabla p dS = \frac{\rho}{\Delta t}\int_{\partial\Omega_I}\vec{n}\cdot\vec{u}^{*}dS \tag{9}$$

where $\partial\Omega_I$ is the boundary of Ω_I and $\vec{n}$ is the normal vector of $\partial\Omega_I$ pointing out of the sub-domain.

The unknown function p needs to be approximated by a set of discretised variables. Generally, the approximation may be written as

$$p(\vec{x}) \approx \sum_{j=1}^{N}\Phi_j(\vec{x})\hat{p}_j \tag{10}$$

where $\hat{p}_j$ are nodal variables and $\Phi_j(\vec{x})$ interpolation functions which are called shape function as they play a similar role to a shape function in a finite element method. Atluri and Shen (2002) have discussed various available options for the shape function. One of them is based on a moving least-square (MLS) method, which is adopted in our work. Using this method, the shape function may be given by

$$\Phi_J(\vec{x}) = \sum_{m=1}^{M} \psi_m(\vec{x}) \left[\mathbf{A}^{-1}(\vec{x}) \mathbf{B}(\vec{x}) \right]_{mJ} \tag{11}$$

with $\boldsymbol{\psi}^T(\vec{x}) = [1, x, y]$ (M=3) for 2D cases and $\boldsymbol{\psi}^T(\vec{x}) = [1, x, y, z]$ (M=4) for 3D cases; and with the matrixes $\mathbf{A}(\vec{x})$ and $B(\vec{x})$ being defined by

$$\mathbf{A}(\vec{x}) = \boldsymbol{\Psi}^T(\vec{x}_I) \mathbf{W}(\vec{x} - \vec{x}_I) \boldsymbol{\Psi}(\vec{x}_I)) \tag{12}$$

$$\mathbf{B}(\vec{x}) = \boldsymbol{\Psi}^T(\vec{x}_I) \mathbf{W}(\vec{x} - \vec{x}_I) \tag{13}$$

where $\mathbf{W}(\vec{x} - \vec{x}_I)$ and $\boldsymbol{\Psi}(\vec{x}_I)$ are, respectively, given by

$$\mathbf{W}(\vec{x} - \vec{x}_I) = \begin{bmatrix} w_1(\vec{x} - \vec{x}_I) & 0 & \cdots & 0 \\ 0 & & & \\ \cdots & & & \\ 0 & & & w_N(\vec{x} - \vec{x}_I) \end{bmatrix} \text{ and} \tag{14a}$$

$$\boldsymbol{\Psi}^T(\vec{x}_I) = \left[\boldsymbol{\psi}(\vec{x}_1), \boldsymbol{\psi}(\vec{x}_1), \cdots, \boldsymbol{\psi}(\vec{x}_N) \right]. \tag{14b}$$

The weight function, w_I, may be chosen as a spline function defined by

$$w_I = \begin{cases} 1 - 6\left(\frac{d_I}{r_I}\right)^2 + 8\left(\frac{d_I}{r_I}\right)^3 - 3\left(\frac{d_I}{r_I}\right)^4 & 0 \le \frac{d_I}{r_I} \le 1 \\ 0 & \frac{d_I}{r_I} > 1 \end{cases} \tag{15}$$

where r_I is the size of support domain of the weight function and $d_I = |\vec{x} - \vec{x}_I|$ the distance between the node I and the point $\vec{x}$.

Inserting Equation (10) into Equation (9) and applying it to all inner nodes yields

$$\mathbf{K} \cdot \hat{\mathbf{P}} = \mathbf{F} \tag{16}$$

where $K_{IJ} = \int_{\partial\Omega_I} \vec{n} \cdot \nabla \Phi_J(\vec{x}) ds$ and $F_I = \frac{\rho}{\Delta t} \int_{\partial\Omega_I} \vec{n} \cdot \vec{u}^* dS$.

Equation (16) does not include the nodes on the boundaries where boundary conditions should be satisfied. The condition on the solid boundary may be imposed on by applying Equation (8) to non-circular sub-domains of boundary nodes. The condition on the free surface may be imposed on by adding a penalty term in Equation (8), as suggested in Atluri and Shen (2002). However it was found in this work that imposing the boundary conditions in the following way is better:

$$\sum_{j=1}^{N} \Phi_j(\vec{x}) \hat{p}_j = p_{atm} \quad \text{for the nodes on the free surface} \tag{17a}$$

$$\sum_{j=1}^{N} \vec{n} \cdot \nabla \Phi_j(\vec{x}) \hat{p}_j = \vec{n} \cdot \left(\vec{g} - \dot{\vec{U}} \right) \text{ for nodes on the solid boundary} \tag{17b}$$

Equation (16) and (17) can be solved to give $\hat{p}_I$ at all nodes and so the pressure can be estimated from (10), which is then used to update the velocity and the position of each node.

Numerical Validation and Examples

Preliminary numerical validation has been made by applying the method to a piston wavemaker problem in a 2D tank with L/d=8, where L is length of the tank and d is the depth of water. The motion of the wavemaker is specified by $S = a \cdot (1 - \cos \omega t)$ with a/d=0.0041 and ω=1.45 $\sqrt{g/d}$. At the end of the tank, an artificial damping zone with length l/d=3 is added to eliminate the

reflection. The comparison with the analytical solution (Eatock Taylor, Wang & Wu, 1994) is presented in Figure 1 and shows a good agreement.

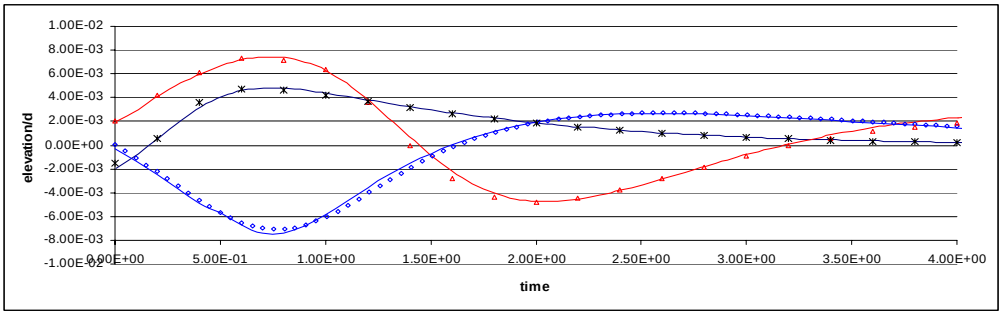

Figure 1 Comparison of wave elevation with analytical solutions for time at 2.5, 4.5 and 6.5 (Solid line: numerical)

The wave generated by the wavemaker with a larger amplitude a/d=0.1 is also simulated. The configuration of nodes in the vicinity of the wavemaker is shown in Figure 2.

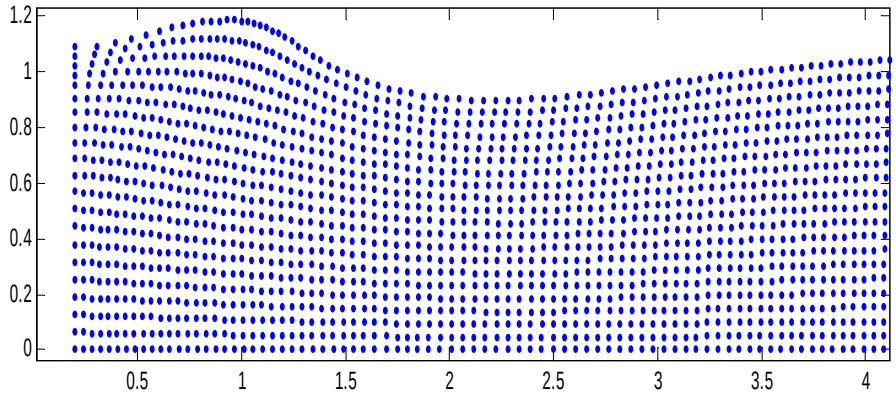

Figure 2 The node configuration for wave at t=6.5 generated by a piston wavemaker

The code for the method is being improved. Further results and discussions will be presented in the workshop.

Acknowledgement

This work is sponsored by EPSRC, UK (GR/R78701), to which the author are most grateful.

References

Atluri, S.N. and Zhu, T., "New Concepts in Meshless Methods", *International J. Numerical Methods In Engineering 47 (1-3): 537-556, JAN 10 2000*.

Atluri, S.N. and Shen, S., "The Meshless Local Petrov-Galerkin (MLPG) Method: A Simple & Less-costly Alternative to the Finite Element and Boundary Element Methods", *Computer Modelling in Engineering & Sceinces, Vol. 3 No.1: 11-52, 2002*.

Dalrymplem, R.A. and Knio, O., "SPH Modelling of Water Waves", *Proc. Coastal Dynamics, Lund, 2001*.

Eatock Taylor, R, Wang, B.T. & Wu, G.X., "On the transient analysis of the wavemaker", *9th International Workshop on Water Waves and Floating Bodies,* Kuju, Oita, Japan, 1994.

Lo, E.Y.M. and Shao, S., "Simulation of near-shore solitary wave mechanics by an incompressible SPH method", *Applied Ocean Research, Volume 24, Issue 5, Pages 275-286, October 2002.*

Discusser: P. Ferrant

One of the difference between your method and SPH is that you explicitly impose a pressure condition on the free surface. This seems to be problematic in case of complex free surface configuration. Is your method capable of simulating such events as overturning waves?

Author's reply:

The MLPG method described in this paper can theoretically deal with any kind of boundary value problems, independent of the way imposing the boundary conditions on the free surface. There are several approaches to impose the dynamics boundary conditions on the surface. The approach employed in this work is to directly force the water pressure equal to the atmospheric pressure. As long as one can identify which particles lie on the free surface when overturning waves occur, this approach would also be applicable to the cases.