New cut cell method for finite differences approximation of the Poisson Equations applied to a semi-averaged model for water waves propagation

Giuseppina Colicchio[1,2], **Matteo Antuono**[1]
[1] CNR-INM, Institute of Marine Engineering, Rome, Italy.
[2] AMOS, Marine Technology Department, NTNU, Trondheim, Norway.
giuseppina.colicchio@cnr.it

1 INTRODUCTION

In incompressible fluids approximation, both finite-differences and finite-volumes solutions of the Poisson equation rely on a gridding of the fluid domain. The most straightforward way to grid the domain is to use a mesh fitted to the fluid boundaries. However, the generation of body fitted grids is time-consuming and relies on the experience of the people for a good quality representation of the fluid flow; in fact, poor meshes can cause non-physical results and undermine the stability of the solution. Moreover, for moving boundaries (e.g. the free surfaces), the boundary deformation can change the topology of the domain and a new gridding is necessary. To avoid all these problems, immersed boundary methods have been devised to describes solid boundaries on Cartesian grids. A full analysis of all the immersed boundaries methods can be found in [1]. There, they are catalogued as a) indirect Boundary Conditions (BC) with a forcing extending into the fluid region and b) direct BC either in the form of Ghost Cell methods in Finite-Differences approaches and Cut-Cell methods in Finite-Volumes approaches. In this work a Cut-Cell approach is devised for finite-differences modelling, taking into account both still or moving boundaries with either Neumann or Dirchlet conditions.

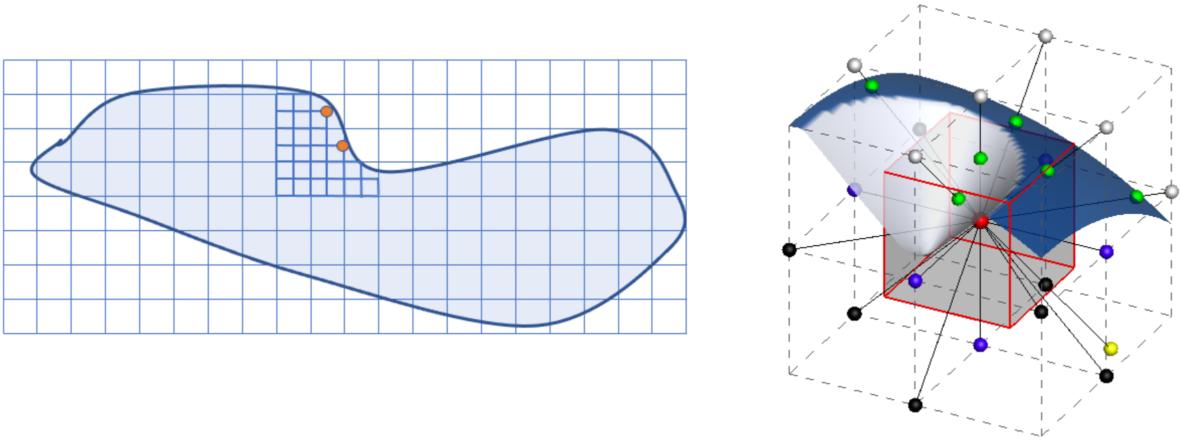

Figure 1: Left: example of local mesh refinement; in orange the cell centers of the coarse grid where care must be taken for the interpolation from the finer mesh. Right: Definition of the 20-points stencil for a cell cut by the domain boundaries. The actually used points in equation (5) are plotted in red, green, blue, black and yellow.

Left part of figure 1 shows how Adaptive Mesh Refinement (AMR) is also introduced into the modelling without significant changes in the way the boundary conditions are treated but for some points. Differently from what is done in [2], where an energy bound is set up on the boundary, here, the geometry of boundary is reconstructed on the Cartesian grid cells to describe even the finest detail of the surface geometry.

2 MATHEMATICAL FORMULATION

The aim of this work is to solve the classical Poisson equation $\nabla^2\phi = RHS$ on an irregular domain represented on a Cartesian grid with both Neumann and Dirchlet BC. The classical second order accurate 7-points stencil is used to discretize equation in points well inside the fluid domain. In the cell centre P_0, close to the boundaries, the function ϕ_0, is written as combination of ϕ_i in the surrounding points P_i at a distance $(\Delta x_i, \Delta y_i, \Delta z_i)$ from P_0. The forth order accurate interpolation of the function ϕ around the point P_i (either in the fluid domain or on the Dirchlet boundary) can be written using a Taylor expansion as:

$$\begin{aligned}
\phi_i - \phi_0 \;=\;& \frac{\partial\phi_0}{\partial x}\Delta x_i + \frac{\partial\phi_0}{\partial y}\Delta y_i + \frac{\partial\phi_0}{\partial z}\Delta z_i + \frac{\partial^2\phi_0}{\partial x^2}\frac{\Delta x_i^2}{2} + \frac{\partial^2\phi_0}{\partial y^2}\frac{\Delta y_i^2}{2} + \frac{\partial^2\phi_0}{\partial z^2}\frac{\Delta z_i^2}{2} + \\
+\;& \frac{\partial^2\phi_0}{\partial x\partial y}\Delta x_i\Delta y_i + \frac{\partial^2\phi_0}{\partial x\partial z}\Delta x_i\Delta z_i + \frac{\partial^2\phi_0}{\partial y\partial z}\Delta y_i\Delta z_i + \frac{\partial^3\phi_0}{\partial x^3}\frac{\Delta x_i^3}{6} + \frac{\partial^3\phi_0}{\partial y^3}\frac{\Delta y_i^3}{6} + \\
+\;& \tfrac{\partial^3\phi_0}{\partial z^3}\tfrac{\Delta z_i^3}{6} + \frac{\partial^3\phi_0}{\partial x^2\partial y}\frac{\Delta x_i^2}{2}\Delta y_i + \frac{\partial^3\phi_0}{\partial x^2\partial z}\frac{\Delta x_i^2}{2}\Delta z_i + \frac{\partial^3\phi_0}{\partial y^2\partial x}\frac{\Delta y_i^2}{2}\Delta x_i + \\
+\;& \frac{\partial^3\phi_0}{\partial y^2\partial z}\frac{\Delta y_i^2}{2}\Delta z_i + \frac{\partial^3\phi_0}{\partial z^2\partial x}\frac{\Delta z_i^2}{2}\Delta x_i + \frac{\partial^3\phi_0}{\partial z^2\partial y}\frac{\Delta z_i^2}{2}\Delta y_i + \frac{\partial^3\phi_0}{\partial x\partial y\partial z}\Delta x_i\Delta y_i\Delta z_i
\end{aligned} \tag{1}$$

Similarly, in the point P_j (on a Neumann boundary), the derivative of function ϕ normal to the Neumann boundary can be written as

$$\begin{aligned}
\frac{\partial\phi_j}{\partial n} \;=\;& \frac{\partial\phi_0}{\partial x}n_{xj} + \frac{\partial\phi_0}{\partial y}n_{yj} + \frac{\partial\phi_0}{\partial z}n_{zj} + \frac{\partial^2\phi_0}{\partial x^2}\Delta x_j n_{xj} + \frac{\partial^2\phi_0}{\partial y^2}\Delta y_j n_{yj} + \frac{\partial^2\phi_0}{\partial z^2}\Delta z_j n_{zj} + \\
+\;& \frac{\partial^2\phi_0}{\partial x\partial y}(\Delta y_j n_{xj} + \Delta x_j n_{yj}) + \frac{\partial^2\phi_0}{\partial x\partial z}(\Delta x_j n_{zj} + \Delta z_j n_{xj}) + \frac{\partial^2\phi_0}{\partial y\partial z}(\Delta y_j n_{zj} + \Delta z_i n_{yj}) + \\
+\;& \frac{\partial^3\phi_0}{\partial x^3}\frac{\Delta x_j^2}{2}n_{xj} + \frac{\partial^3\phi_0}{\partial y^3}\frac{\Delta y_j^2}{2}n_{yj} + \frac{\partial^3\phi_0}{\partial z^3}\frac{\Delta z_i^2}{2}n_{zj} + \frac{\partial^3\phi_0}{\partial x^2\partial y}(n_{xj}\Delta x_j\Delta y_j + n_{yj}\frac{\Delta x_j^2}{2}) + \\
+\;& \frac{\partial^3\phi_0}{\partial x^2\partial z}(n_{xj}\Delta x_j\Delta z_j + n_{zj}\frac{\Delta x_j^2}{2}) + \frac{\partial^3\phi_0}{\partial y^2\partial x}(n_{yj}\Delta x_j\Delta y_j + n_{xj}\frac{\Delta y_j^2}{2}) + \\
+\;& \frac{\partial^3\phi_0}{\partial y^2\partial z}(n_{yj}\Delta y_j\Delta z_j + n_{zj}\frac{\Delta y_j^2}{2}) + \frac{\partial^3\phi_0}{\partial z^2\partial x}(n_{zj}\Delta x_j\Delta z_j + n_{xj}\frac{\Delta z_j^2}{2}) + \\
+\;& \frac{\partial^3\phi_0}{\partial z^2\partial y}(n_{zj}\Delta y_j\Delta z_j + n_{yj}\frac{\Delta z_j^2}{2}) + \frac{\partial^3\phi_0}{\partial x\partial y\partial z}(n_{xj}\Delta y_j\Delta z_j + n_{yj}\Delta x_j\Delta z_j + n_{zj}\Delta x_i\Delta y_j)
\end{aligned} \tag{2}$$

with (n_{xj}, n_{yj}, n_{zj}) the normal-unitary vector to the the boundary through point P_j.

Both the Taylor expansions are written with 19 terms. Supposing that the normal derivatives are known in N_n points and the values of the function ϕ is known in N_m points (with $N_n + N_m = 19$), it is possible to multiply each of the equations (1) by a coefficient a_i and each of the equations (2) by a coefficient b_j so that summing them we have:

$$\begin{aligned}
& \textstyle\sum_i^{N_m} a_i(\phi_i - \phi_0) + \sum_j^{N_n} b_j\frac{\partial\phi_j}{\partial n} = \\
=\;& \tfrac{\partial\phi_0}{\partial x}(\textstyle\sum_i^{N_m} a_iF_1 + \sum_j^{N_n} G_1) + \tfrac{\partial\phi_0}{\partial y}(\sum_i^{N_m} a_iF_2 + \sum_j^{N_n} b_jG_2) + \tfrac{\partial\phi_0}{\partial z}(\sum_i^{N_m} a_iF_3 + \sum_j^{N_n} b_jG_3) + \\
+\;& \tfrac{\partial^2\phi_0}{\partial x^2}(\textstyle\sum_i^{N_m} a_iF_4 + \sum_j^{N_n} b_jG_4) + \tfrac{\partial^2\phi_0}{\partial y^2}(\sum_i^{N_m} a_iF_5 + \sum_j^{N_n} b_jG_5) + \tfrac{\partial^2\phi_0}{\partial z^2}(\sum_i^{N_m} a_iF_6 + \sum_j^{N_n} b_jG_6) + \\
+\;& \tfrac{\partial^2\phi_0}{\partial x\partial y}(\textstyle\sum_i^{N_m} a_iF_7 + \sum_j^{N_n} b_jG_7) + \tfrac{\partial^2\phi_0}{\partial x\partial z}(\sum_i^{N_m} a_iF_8 + \sum_j^{N_n} b_jG_8) + \tfrac{\partial^2\phi_0}{\partial y\partial z}(\sum_i^{N_m} a_iF_9 + \sum_j^{N_n} b_jG_9) + \\
+\;& \tfrac{\partial^3\phi_0}{\partial x^3}(\textstyle\sum_i^{N_m} a_iF_{10} + \sum_j^{N_n} b_jG_{10}) + \tfrac{\partial^3\phi_0}{\partial y^3}(\sum_i^{N_m} a_iF_{11} + \sum_j^{N_n} b_jG_{11}) + \\
+\;& \tfrac{\partial^3\phi_0}{\partial z^3}(\textstyle\sum_i^{N_m} a_iF_{12} + \sum_j^{N_n} b_jG_{12}) + \tfrac{\partial^3\phi_0}{\partial x^2\partial y}(\sum_i^{N_m} a_{i13} + \sum_j^{N_n} b_jG_{13}) + \\
+\;& \tfrac{\partial^3\phi_0}{\partial x^2\partial z}(\textstyle\sum_i^{N_m} a_iF_{14} + \sum_j^{N_n} b_jG_{14}) + \tfrac{\partial^3\phi_0}{\partial y^2\partial x}(\sum_i^{N_m} a_iF_{15} + \sum_j^{N_n} b_jG_{15}) + \\
+\;& \tfrac{\partial^3\phi_0}{\partial y^2\partial z}(\textstyle\sum_i^{N_m} a_iF_{16} + \sum_j^{N_n} b_jG_{16}) + \tfrac{\partial^3\phi_0}{\partial z^2\partial x}(\sum_i^{N_m} a_iF_{17} + \sum_j^{N_n} b_jG_{17}) + \\
+\;& \tfrac{\partial^3\phi_0}{\partial z^2\partial y}(\textstyle\sum_i^{N_m} a_iF_{18} + \sum_j^{N_n} b_jG_{18}) + \tfrac{\partial^3\phi_0}{\partial x\partial y\partial z}(\sum_i^{N_m} a_iF_{19} + \sum_j^{N_n} b_jG_{19})
\end{aligned} \tag{3}$$

where the coefficients F_α depende on $\Delta x_i, \Delta y_i, \Delta z_i$ (see equation (1)) and the G_β depende on $\Delta x_j, \Delta y_j, \Delta z_j, n_{xj}, n_{yj}, n_{zj}$ from equation (2).

In order to have that the left hand side of equation (3) represents a second order accurate approximation of the Laplacian operator in point P_0, we have to impose that the coefficient of each of the derivative of ϕ_0 is null but for those of $\frac{\partial^2\phi_0}{\partial x^2}$, $\frac{\partial^2\phi_0}{\partial y^2}$ and $\frac{\partial^2\phi_0}{\partial z^2}$. Each of these last coefficients has to be equal to 1. So we identify a system of 19 equations with 19 unknowns (the N_m a_i coefficients and the N_n b_j coefficients). Practically, we define a linear system

$$\mathbf{M}\left\{\begin{array}{c} \mathrm{a}_i \\ \mathrm{b}_j \end{array}\right\} = \{I_{19}\} \tag{4}$$

where $\{I_{19}\}$ is a vector with all its elements equal to zero but for the 4th, 5th and 6th ones and where the solution vector $\{a_i, b_j\}^T$ is later used to approximate the Laplacian operator from the left hand side of equation (3), namely:

$$\nabla^2\phi_o \simeq \sum_i^{N_m}(a_i\phi_i) - \left(\sum_i^{N_m} a_i\right)\phi_0 + \sum_j^{N_n}\left(b_j\frac{\partial\phi_j}{\partial n}\right) + O(\Delta P^2) \tag{5}$$

Practically we have enlarged the stencil to 20 points close to the domain boundaries. In same cases, system (4) becomes overdetermined. To overcome this problem, a Singlular Value Decomposition SVD is used.

2.1 Definition of the points on the stencil

To identify the 20-points of the stencil close to the boundary, we refer to the right side of figure 1. There, the cell of interest is plotted with red edges and it is cut by the boundary surface represented with a dark and light blue surface. The two colors indicate two different boundary conditions that can be given on that surface. In the same picture, the blue spheres represents the classical 7-points stencil. Other 12 points are used here and shown in black. Actually, some of those points, as the top ones (shaded in gray), are outside of the fluid domain. For this reason, they are substituted with the green points, that are the intersection between the boundary surface and the line connecting the cell-center with those outside points. In this particular case, two of those points are characterized by a kind of boundary condition while four are characterized by the other BC.

The last point for the solution of system (4) should be possibly inside the domain. To satisfy this requirement, it is obtained using $\mathbf{n}$, the unitary normal vector to the boundary surface in the closest point to the cell centre, pointing inside the domain. This new point is represented in yellow in figure 1 and it is at distance $(2N(n_x)\Delta x, 2N(n_y)\Delta y, 2N(n_z)\Delta z)$ from the cell centre, where $N(f)$ stands for nearest whole number to f.

2.2 Local Mesh Refinment

To handle different mesh sizes, the solution of the Poisson equations is based on an iterative process similarly to [3]. *S0: At the first iteration level*, the solution is calculated on the coarsest subsiding mesh as described in paragraph 2.1; *Sl: then the solution on any further level of refinement* is calculated only on that level with Dirchlet BC on the border with a coarser mesh. *CH: At the end of this up-level process*, the solutions form the local finest level is interpolated on the coarsest mesh. If the difference between the interpolated value and the initial coarsest solution is lower than a threshold value, the solution can be accepted as it is. Otherwise, on the coarsest level, these interpolated values are used to recalculate the forcing term of the Poisson equation in the refined region, taking into account the details that the coarse mesh is unable to see and the whole process from *S0* to *CH* is repeated. Particular care has to be taken for the cell centers highlighted in orange in the left side of figure 1. There, the set of points used for the interpolation from the finer mesh should be deprived of those points outside the fluid domain.

3 APPLICATION TO THE SEMI-AVARAGED MODEL

In [4], the Depth Semi Averaged Model (DSAM) was applied to the study of the interaction of a solitary wave with a submerged obstacle. In that case, an indirect immersed boundary condition has been applied to the solution of the Poisson equation $\nabla^2\Gamma = \nabla \cdot \left(\frac{\mathbf{M}}{d}\right)$ for Γ the semi-averaged vertical velocity component written as $\Gamma = \int_z^\eta w dz$, where w is the vertical component of the velocity, h the local bathymetry, d is the local water depth and $\mathbf{M}$ is the generalized mass flux.

In [4], the results from DSAM were compared with those obtained with OpenFOAM characterized by a mesh three times finer. The results were in reasonable good agreement but for the local interaction with the upper edges of the submerged obstacle. Figure 2 shows the comparison of water heigh obtained with the Navier-Stokes solver (bottom part of the panels) with the indirect immersed boundary on the left and with the Cut-Cell method on the right for DSAM. When interacting with the backward facing step, the new Cut-Cell method allows to better capture the details of the interaction with the edge. After that time step, the high steepness of the free surface causes a local mesh refinement that allows to reach the breaking of the wave front. However, the details of the breaking cannot be modelled in DSAM and they are not fully captured by OpenFOAM without any further refinement.

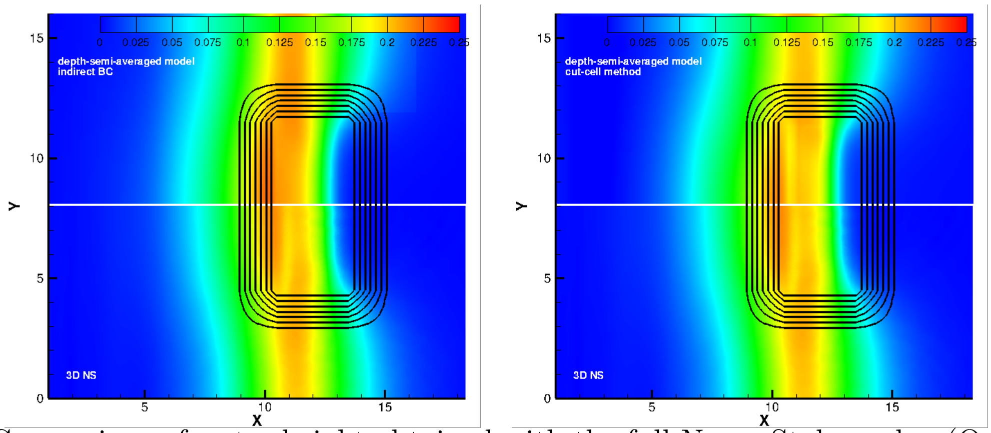

Figure 2: Comparison of water height obtained with the full Naver-Stokes solver(OpenFOAM) versus the solution of DSAM with indirect immersed BC (left side) and with Cut-Cell method (right).

REFERENCES

[1] Mittal, R., and Iaccarino, G. 2005. *Immersed Boundary Methods. Annu. Rev. Fluid Mech. 37*, 239–261.

[2] Sharan, N., Brady, P., and Livescu, C. 2021. *Finite-difference Cartesian cut-cell method for hyperbolic systems. AIAA 2021-0746*.

[3] Martin, D., and Cartwright, K. Oct 1996. Solving poisson's equation using adaptive mesh refinement. Tech. Rep. UCB/ERL M96/66, EECS Department, University of California, Berkeley.

[4] Colicchio, G., and Antuono, M. 2020. *Study of the interaction of a solitary wave with a submerged obstacle through a depth semi-averaged model. 35th IWWWFB*.