Manual for *CRestRelax.py*

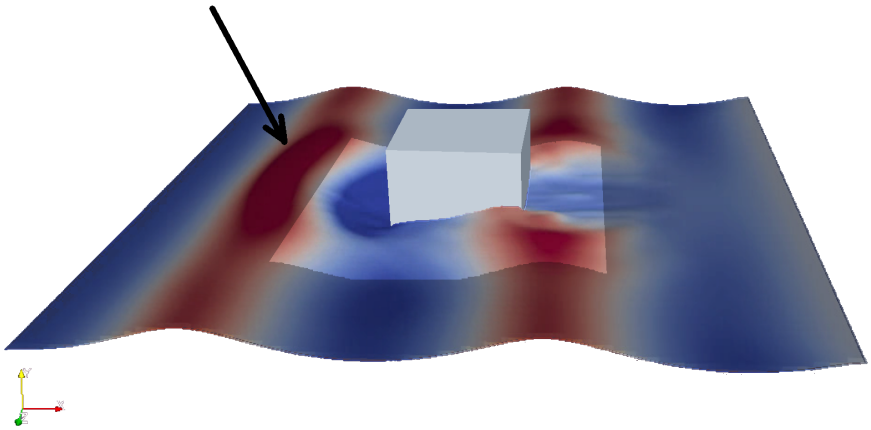
a computer program for estimating reflection coefficient C_R for relaxation zones in flow simulations of free-surface wave propagation

written by Robinson Perić

at the Institute for Fluid Dynamics and Ship Theory,
Hamburg University of Technology (TUHH), Germany
and at the Fakultet Strojarstva i Brodogradnje,
Sveučilište u Zagrebu, Croatia

version August 12, 2019

relaxation zones (shaded gray) gradually blend-out the governing equations and blend-in the far-field wave solution



Contents

1 One-minute-explanation of the code

When to use this code?

- You perform flow simulations of free-surface wave propagation
- You want to minimize undesired wave reflections at the domain boundaries
- To do so, you want to use *relaxation zones*, which blend the flow solution towards the far-field wave in a zone usually attached to the vertical domain boundaries

What does the code do?

- The user can enter wave period, water depth, wavelength, zone thickness, blending function $b(\mathbf{x})$, and blending exponent n
- The code calculates reflection coefficient C_R for various values of relaxation zone parameter τ according to [1] and writes these results to the file 'C_R.csv' in the same folder
- If matplotlib is installed: A window will open with an interactive plot of C_R over τ
- Thus the code can be used to quickly fine-tune the case-dependent relaxation zone parameters

Requirements:

- Check that your relaxation zone can be formulated in terms of Eqs. (2) and (3)
- Install *python* programming language
- For full functionality, install *matplotlib*
- Operating systems: Linux, macOS, Windows

Tuning relaxation zones:

- Increasing the zone thickness x_d tends to lower the reflection coefficient and to widen the range of wavelengths for which reflections will be satisfactorily minimized
- Typically, confidence in minimizing reflections is more important than the last few percent efficiency $\Rightarrow$ use slightly thicker zones than necessary
- Common values for zone thickness are $1\lambda \leq x_\mathrm{d} \leq 2\lambda$
- For irregular waves, tune the zone to the peak period or the longest period (quick approach [3]) or calculate the reflection coefficient for each wave component's period and then tune accordingly (more accurate approach)

Benefits and Limitations:

- The code satisfactorily predicted reflection coefficients and optimum relaxation zone parameters in 2D- and 3D-flow simulations with regular, irregular, linear, and highly nonlinear waves
- For complex flows (e.g. steep waves, breaking waves or 3D oblique wave incidence) reflection coefficients were in some cases slightly larger than predicted, so that thicker zones are recommended to ensure satisfactory wave absorption
- Undesired reflections can also be due to other mechanisms, e.g. the use of inappropriate grids
- Therefore, tuning the relaxation zone parameters according to this code does NOT guarantee

that the actual reflection coefficient in the simulation will equal the prediction

2 Requirements

The programming language python version 2.7 or 3.0 or higher (`https://www.python.org/downloads/`) must be installed.

It is recommended to also have matplotlib (`https://matplotlib.org/users/installing.html`) installed[1]. Then the code will open a window with an interactive plot of the results.

3 Motivation and theory behind the code

Relaxation zones can reduce undesired wave reflections at boundaries of the computational domain. However, their source terms contain user-defined parameters; these parameters are case-dependent and must be tuned for every simulation. Otherwise strong reflection may occur.

The present code can guide the tuning of these case-dependent parameters. The code is an implementation of the theory from [1]. Please see [1] to find out if and how the theory applies to the relaxation zone implementation that you use.

In [1], a theory was presented which predicts the reflection coefficient for a given relaxation zone setup. The reflection coefficient C_R is the ratio of the reflected wave amplitude to the incidence wave amplitude. Thus the theory can be used to tune the case-dependent parameters *before running the simulation.*

For practical discretizations ($\gtrsim$ 30 cells per wavelength), relaxation zones were found to behave independent of the discretization and the used flow solver, so the present code can be expected to apply to most implementations of relaxation zones.

The code's predictions for reflection coefficient and optimum relaxation zone parameters were demonstrated to be of satisfactory accuracy in 2D- and 3D-flow simulations with regular, irregular, linear, and highly nonlinear waves in both shallow, intermediate and deep water depths (cf. [1]–[4]). Thus the present code can be recommended for practical 3D-flow simulations with relaxation zones featuring nonlinear wave phenomena.

Future research will provide further details to more accurately assess the accuracy of the theory's predictions. In the meantime, it is recommended to select the relaxation zone thickness slightly larger than theoretically necessary, to ensure satisfactory wave absorption.

The theory in [1] was derived so that it works for any continuous or discontinuous blending function. A few common blending functions are already implemented. Custom blending functions can be entered at the location indicated in the source code.

[1] For debain-based linux systems, install e.g. via: `sudo apt-get install python3-matplotlib`.

4 Introduction to relaxation zones

Relaxation zones blend, say a general transport equation $\mathcal{T}$ for transport quantity ϕ, over to a reference solution via

$$(1 - b(\mathbf{x}))\,\mathcal{T} + \frac{b(\mathbf{x})}{\tau}\mathcal{R} = 0 \quad , \tag{1}$$

where $b(\mathbf{x})$ is a blending function such as e.g. Eqs. (4) to (11), $\mathcal{T}$ corresponds e.g. to the conservation equations for mass or momentum (without relaxation source terms), and $\mathcal{R}$ corresponds to $\int_V (\phi - \phi_{\mathrm{ref}})\ \mathrm{d}V$ with reference solution ϕ_{ref} for transport quantity ϕ. The relaxation parameter τ regulates the magnitude of the source term in such a way that a large value of τ implicates a small source term and vice versa.

Note that τ has sometimes been interpreted as a numerical stability parameter, and thus has occasionally been omitted[2] from Eq. (1), so in literature Eq. (1) is sometimes written in the following form: $(1 - b(\mathbf{x}))\,\mathcal{T} + b(\mathbf{x})\mathcal{R} = 0$. However, dimensional analysis shows that τ has the unit $[s]$, so it is recommended to include τ in the equation as in Eq. (1).

Hence the conservation equations for momentum and volume fraction take the form

$$\begin{aligned}(1 - b(\mathbf{x}))\Bigg[&\frac{\mathrm{d}}{\mathrm{d}t}\int_V \rho u_i\ \mathrm{d}V + \int_S \rho u_i(\mathbf{v} - \mathbf{v}_g)\cdot\mathbf{n}\ \mathrm{d}S \\ &- \int_S (\tau_{ij}\mathbf{i}_j - p\mathbf{i}_i)\cdot\mathbf{n}\ \mathrm{d}S - \int_V \rho\mathbf{g}\cdot\mathbf{i}_i\ \mathrm{d}V\Bigg] \\ &+ \frac{b(\mathbf{x})}{\tau}\Bigg[\int_V \rho\,(u_i - u_{i,\mathrm{ref}})\ \mathrm{d}V\Bigg] = 0 \quad ,\end{aligned} \tag{2}$$

$$\begin{aligned}(1 - b(\mathbf{x}))\Bigg[&\frac{\mathrm{d}}{\mathrm{d}t}\int_V \alpha\ \mathrm{d}V + \int_S \alpha(\mathbf{v} - \mathbf{v}_{\mathrm{g}})\cdot\mathbf{n}\ \mathrm{d}S\Bigg] \\ &+ \frac{b(\mathbf{x})}{\tau}\Bigg[\int_V (\alpha - \alpha_{\mathrm{ref}})\ \mathrm{d}V\Bigg] = 0 \quad ,\end{aligned} \tag{3}$$

with volume V of control volume (CV) bounded by the closed surface S, fluid velocity $\mathbf{v} = (u_1, u_2, u_3)^{\mathrm{T}} = (u, v, w)^{\mathrm{T}}$, grid velocity $\mathbf{v}_g$, unit vector $\mathbf{n}$ normal to S and pointing outwards, time t, pressure p, fluid density ρ, components τ_{ij} of the viscous stress tensor, unit vector $\mathbf{i}_j$ in direction x_j, and volume fraction α of the liquid phase, reference velocities $u_{i,\mathrm{ref}}$ and reference volume fraction α_{ref}.

Blending function $b(\mathbf{x})$ regulates how the magnitude of the source term varies within the relaxation zone. Outside the relaxation zone holds $b(\mathbf{x}) = 0$, and inside the relaxation zone holds $0 \leq b(\mathbf{x}) \leq 1$. Many different types of blending functions can be applied. The following common choices are implemented in the code:

Constant blending

$$b(\mathbf{x}) = 0.5 \quad , \tag{4}$$

[2] If the description of your code does not include τ, then it is probably set internally to $\tau = 1\,\mathrm{s}$ or to some other default value. With a look at Eqs. (1–3), the reader should have all tools available to judge whether this is so and, if necessary, to modify the source code accordingly so that the default value of τ can be adjusted by the user, which is necessary to obtain optimum tuning of the relaxation zone's parameters.

linear blending

$$b(\mathbf{x}) = \left(\frac{x_\mathrm{d} - \tilde{x}}{x_\mathrm{d}}\right) \quad , \tag{5}$$

quadratic blending

$$b(\mathbf{x}) = \left(\frac{x_\mathrm{d} - \tilde{x}}{x_\mathrm{d}}\right)^2 \quad , \tag{6}$$

$\cos^2$-blending

$$b(\mathbf{x}) = \cos^2\left(\frac{\pi}{2} + \frac{\pi}{2}\left(\frac{x_\mathrm{d} - \tilde{x}}{x_\mathrm{d}}\right)\right) \quad , \tag{7}$$

exponential blending

$$b(\mathbf{x}) = \left(\frac{e^{((x_\mathrm{d}-\tilde{x})/x_\mathrm{d})^2} - 1}{e^1 - 1}\right) \quad , \tag{8}$$

power blending

$$b(\mathbf{x}) = \left(\frac{x_\mathrm{d} - \tilde{x}}{x_\mathrm{d}}\right)^n \quad , \tag{9}$$

exponential blending with power n

$$b(\mathbf{x}) = \left(\frac{e^{((x_\mathrm{d}-\tilde{x})/x_\mathrm{d})^n} - 1}{e^1 - 1}\right) \quad , \tag{10}$$

and $\cos^{2n}$-blending

$$b(\mathbf{x}) = \left[\cos^2\left(\frac{\pi}{2} + \frac{\pi}{2}\left(\frac{x_\mathrm{d} - \tilde{x}}{x_\mathrm{d}}\right)\right)\right]^n \quad , \tag{11}$$

where $\tilde{x}$ is the shortest distance of location $\mathbf{x}$ to the closest domain boundary to which a relaxation zone of thickness x_d is attached, and n regulates the shape of the blending function. Some of these blending functions are illustrated in Fig. 1.

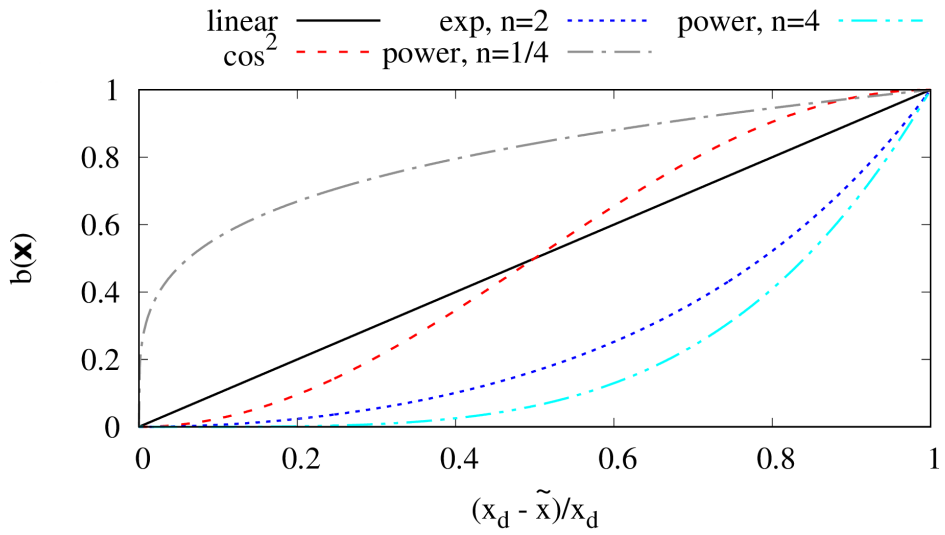


Figure 1: Different blending functions $b(\mathbf{x})$ over location in relaxation zone, with entrance to the relaxation zones at $\frac{x_\mathrm{d}-\tilde{x}}{x_\mathrm{d}} = 0$ and boundary to which the relaxation zone is attached at $\frac{x_\mathrm{d}-\tilde{x}}{x_\mathrm{d}} = 1$; for blending via Eq. (9) with $n = 1$ ('linear'), $n = 1/4$ ('power, n=1/4') and $n = 4$ ('power, n=4'), blending via Eq. (7) with $n = 1$ ('$\cos^2$, n=1'), and blending via Eq. (8) with $n = 2$ ('exp, n=2')

Thus relaxation zones have three user-defined parameters, which are case-dependent and have to be adjusted for each simulation: the relaxation parameter τ, the blending function $b(\mathbf{x})$ and the

zone thickness x_d.

The difference between forcing zones and relaxation zones is that relaxation zones 'blend out' the governing equations via the factor $(1 - b(\mathbf{x}))$; forcing zones do not have this factor, thus with forcing zones the whole governing equations remain active in the whole domain, whereas within relaxation zones as in Eqs. (2) and (3), the terms from the governing equations that are active in the solution domain of interest are faded out and the reference solution is faded in.

Since the theory for predicting reflection coefficients for forcing zones [2] could be extended to relaxation zones in [1], one can conclude that forcing zones and relaxation zones are closely related. The results suggest that both approaches perform equally well, so the choice for using relaxation or forcing zones can be made based on which one is already implemented or which one is easier to implement in one's flow solver.

In contrast to forcing zones, where constant or linear blending functions were found to typically perform worse than the other blending functions, for relaxation zones this is different: For example, constant blending with $b(\mathbf{x}) = 1$ should be avoided, since then the entrance to the relaxation zone would behave like a velocity boundary condition, i.e. may be fully reflective.

Consider that the theory for predicting relaxation zone behavior was derived by reformulating Eqs. (2) and (3) as a forcing zone, which was done be multiplying both equations with $(1 - b(\mathbf{x}))^{-1}$. Therefore in relaxation zones, the simultaneous blending-out of one solution (the parts in Eqs. (2) and (3) which are multiplied by $(1-b(\mathbf{x}))$) while blending-in another solution (the parts in Eqs. (2) and (3) which are multiplied by $\frac{b(\mathbf{x})}{\tau}$) can be interpreted as a forcing 'with a different blending', so the "effective" increase in source term magnitude is not directly proportional to $b(\mathbf{x})$, but rather to $b(\mathbf{x})/\left(1 - b(\mathbf{x})\right)$. The optimum choice of blending $b(\mathbf{x})$ was found to depend on the zone thickness x_d. Since the dependence of reflection coefficient C_R on the case-dependent parameters is quite complex, the theory was found to be a useful tool for tuning the parameters. See [1] for a detailed discussion.

IMPORTANT
If your governing equations do not contain the factor $(1 - b(\mathbf{x}))$, i.e. if they do not look like Eq. (1), then your implementation may be a forcing zone, not a relaxation zone. **If so, do not use the present code!** Instead, please use the following code:
`https://github.com/wave-absorbing-layers/relaxation-zones-for-free-surface-waves`
Under the above link, you also find a more detailed description of forcing zones.

5 Benefits and limitations

The code's predictions for reflection coefficient and optimum relaxation zone parameters were demonstrated to be of satisfactory accuracy in 2D- and 3D-flow simulations with regular, irregular, linear, and highly nonlinear waves (cf. [1]–[4]).

Reflection coefficients in flow simulations were typically lower or similar to the code's predictions, but never more than a few percent larger when optimally tuned using the code.

Although the code's predictions are usually quite accurate [1], please keep in mind that every theory has its limitations! For highly non-linear waves, such as breaking waves, or for complex 3D-flows

with oblique wave incidence, reflection may be larger than predicted.

Further, undesired reflections can be due other mechanisms as well, e.g. the use of inappropriate grids or certain relaxation zone arrangements[3]. Therefore, tuning the relaxation zone's parameters according to the present theory does NOT guarantee that the actual reflection coefficient in the simulation will equal the prediction. See [1] and [4] for a detailed discussion.

6 Recommendations for tuning relaxation zones

At the wave-maker, it is usually known which waves are generated. For a given wave period T, the wavelength λ depends on the water depth and on the wave steepness, thus this has to be entered in the code. In practice, a relaxation zone thickness of $1\lambda \leq x_\mathrm{d} \leq 2\lambda$ is usually satisfactory. With correct tuning thinner zones may be used; in practice however, usually confidence in the minimization of undesired wave reflections is more important than the last few percent efficiency. Therefore it is recommended to use zone thicknesses x_d slightly larger than theoretically necessary.

Increasing the zone thickness x_d tends to widen the range of wavelengths which will be damped satisfactorily and to lower the reflection coefficient at the optimum parameter setting; thus if the wave absorption is not satisfactory, then zone thickness x_d should be increased.

For irregular waves, a quick approach is to tune the zone to the peak period or the longest period[3]. A more accurate approach is to calculate the reflection coefficient for each wave component's period and in this manner tune the zone parameters accordingly.

7 How to run the code

In windows, double click on the executable file *CRestRelax.py*.

In Linux and macOS, open a terminal and type:
`python CRestRelax.py`

Example output of the code is shown in Fig. 2.

Alternatively, experienced users may pass the parameters as arguments[4].

[3] For example, relaxation of fluid momentum and volume fraction towards the far-field wave solution can create flow disturbances in relaxation zones which lie tangential to the wave propagation direction, if there is a mismatch between computed flow solution (including discretization and iteration errors) and the reference solution (without discretization and iteration errors). Such flow disturbances can be radiated as undesired waves into the domain. If the reference solution is highly accurate, these disturbances vanish on fine grids; however, finer grids than commonly used may be required. Luckily, there are several forcing zone arrangements that do not have these problems, see [4]. Constructing according relaxation zone arrangements is straight-forward.

[4] To use the code in batch-mode for blending via Eqs. (4-8), type :
`python CRestRelax.py T h L xd b`
and replace `T` with the wave period in seconds, `h` with the water depth in meters, `L` with the wavelength in meters, `xd` by the zone thickness in meters, and `b` by the number representing the desired blending function (cf. Fig. 2).
To use the code in batch-mode for blending via Eqs. (9-11), type :
`python CRestRelax.py T h L xd b n`
and replace n by the blending function exponent

```
python CRestRelax.py


Please enter wave period (s):
1.6

Please enter water depth (m):
4

Please enter wavelength (m):
4

Please enter forcing zone thickness (m):
8

Please select a blending function b(x)


--- Commonly used b(x) ---
Enter 1 for CONSTANT blending, i.e. b(x) = 1
Enter 2 for LINEAR blending, i.e. b(x) = x
Enter 3 for QUADRATIC blending, i.e. b(x) = x^2
Enter 4 for COSINE-SQUARED blending, i.e. b(x) = ( cos(x) )^2
Enter 5 for EXPONENTIAL blending, i.e. b(x)= ( exp(x^2) - 1 ) / ( exp(1) - 1 )
--- Expert b(x) ---
Enter 6 for POWER blending with exponent n, i.e. b(x) = x^n
Enter 7 for EXPONENTIAL blending with exponent n, i.e. b(x)= ( exp(x^n) - 1 ) / ( exp(1) - 1 )
Enter 8 for COSINE-SQUARED blending with exponent n, i.e. ( ( cos(x) )^2 )^n
Enter 9 for CUSTOM blending

Please enter number of blending function:
7

Please enter blending exponent n:3.5
```

Figure 1

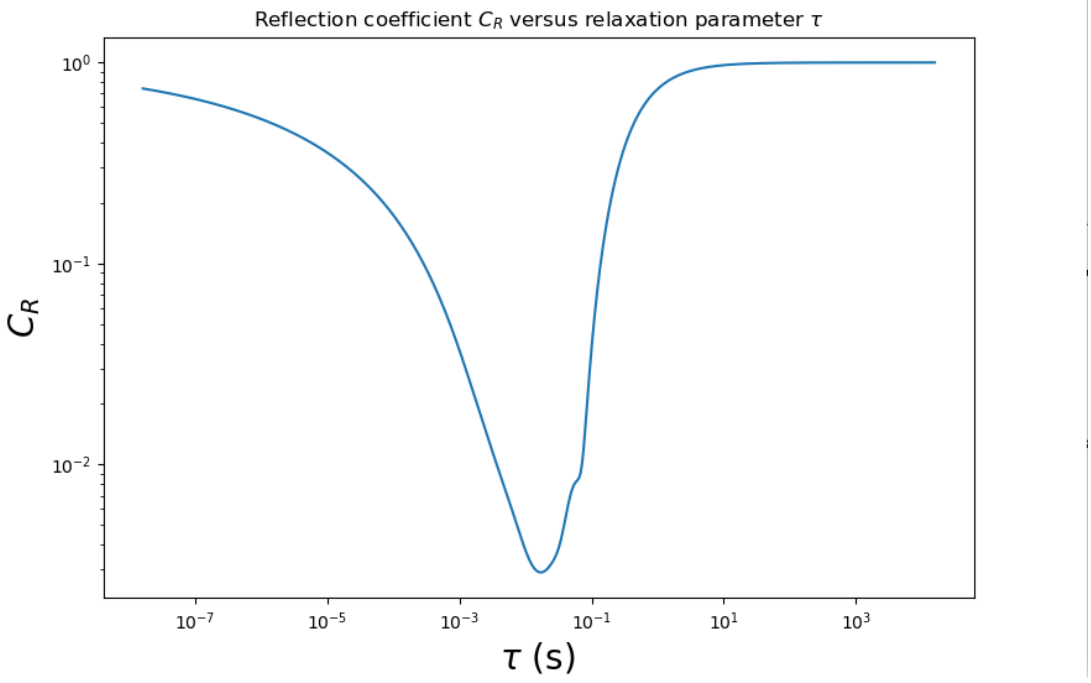


Figure 2: Example output of the code to compute the reflection coefficient C_R depending on the value of relaxation parameter τ; for a wave with period $T = 1.6\,\mathrm{s}$, water depth $h = 4\,\mathrm{m}$, wavelength $\lambda = 4\,\mathrm{m}$; with a relaxation zone of thickness $x_\mathrm{d} = 8\,\mathrm{m}$ and exponential blending via Eq. (10) with exponent $n = 3.5$

8 Reporting bugs

So far, no bugs are known.

If you find bugs, or if you have questions or suggestions, please contact the author:

M.Sc. Robinson Perić
Hamburg University of Technology (TUHH)
Institute for Fluid Dynamics and Ship Theory (M8)
Am Schwarzenberg-Campus 4
D-21073 Hamburg, Germany
Room C5.004
Phone: +49 40 42878 6031
Fax: +49 40 42878 6055
E-mail: robinson.peric@tuhh.de
URL: http://www.tuhh.de/fds/staff/

The code is available at
`https://github.com/wave-absorbing-layers/relaxation-zones-for-free-surface-waves`
Updates and further useful information will be posted there as well.

9 Bug fixes & previous versions of the code

The 2017 version of this code produced slightly inaccurate predictions for shallow water conditions. This has been amended, and now the code should work for all water depths and all relaxation zone settings. Furthermore, the interactive-mode was adjusted to be more user-friendly, and a batch mode was added, so parameters can also be passed as arguments.

10 Copyright

11 References

[1] Perić, R., Vukčević, V., Abdel-Maksoud, M., & Jasak, H. (2018). Tuning the Case-Dependent Parameters of Relaxation Zones for Flow Simulations With Strongly Reflecting Bodies in Free-Surface Waves. arXiv preprint arXiv:1806.10995.

[2] Perić, R., & Abdel-Maksoud, M. (2018). Analytical prediction of reflection coefficients for wave absorbing layers in flow simulations of regular free-surface waves. Ocean Engineering, 147, 132-147.

[3] Perić, R., & Abdel-Maksoud, M. (2016). Reliable damping of free-surface waves in numerical simulations. Ship Technology Research, 63, 1, 1–13.

[4] Perić, R., & Abdel-Maksoud, M. (2019). Reducing Undesired Wave Reflection at Domain Boundaries in 3D Finite Volume–Based Flow Simulations via Forcing Zones. Journal of Ship Research.