Time-dependent wavemaker

1. Summary

1) The problem was caused by the wave-coherence which occurs as phases for specified wave components are the same. The finding provides an extreme example with 100% wave-coherence. .

2) I used a simple MATLAB program to identify the problem. See the matlab scripts in the package.

3) In the modified funwave code I added a random function to generate random phases if a user does not specify phases in input files. I also implemented an option which a user can make an output of the linear theoretical solution to compare the numerical solution.

4) The time-dependent spectral wavemaker is documented in details. Hope it can help with further documentation/reports.

5)Please use the matlab programs provided in the package to check the results. Feel free to ask me if you have any questions.

2. The problem

Figure 1 shows the spectral distribution from your file: spectral_data_1.txt. I assume that the data is wave amplitude for each df_ddir cell and has a unit of meter. The significant wave height is 0.3704. The matlab program for this plot can be found /Linear_Program/mk_spec_brocc.m.

The problem is that, when you directly use the spectral files, the wave surface calculated from the model shows an inhomogeneous distribution of wave height as shown in Figure 2. In fact, the inhomogeneous distribution is not from any bugs in the code. It is from a strong wave-coherence, which occurs as the wave phases are the same at the wavemaker. Figure 3 shows the theoretical solution of wave surface using the spectral file. It has the same pattern as the numerical solution. The theoretical solution will be described later in this doc.

3. Time-dependent spectral wavemaker

The wavemaker was developed based on Larsen and Dancy (1983), Chen et al. (1999), and Zhang et al. (2014) (in /references/ of the package). Waves are generated in a sponge regime, with a damping function of C_s. For each wave component, the dependent variables (η, u, v) at the grid cell (i, j) are attenuated as

$$\eta_{i,j} = \eta_s + (\eta_{i,j} - \eta_s)/C_s \tag{1}$$

$$u_{i,j} = u_s + (u_{i,j} - u_s)/C_s \tag{2}$$

$$v_{i,j} = v_s + (v_{i,j} - v_s)/C_s \tag{3}$$

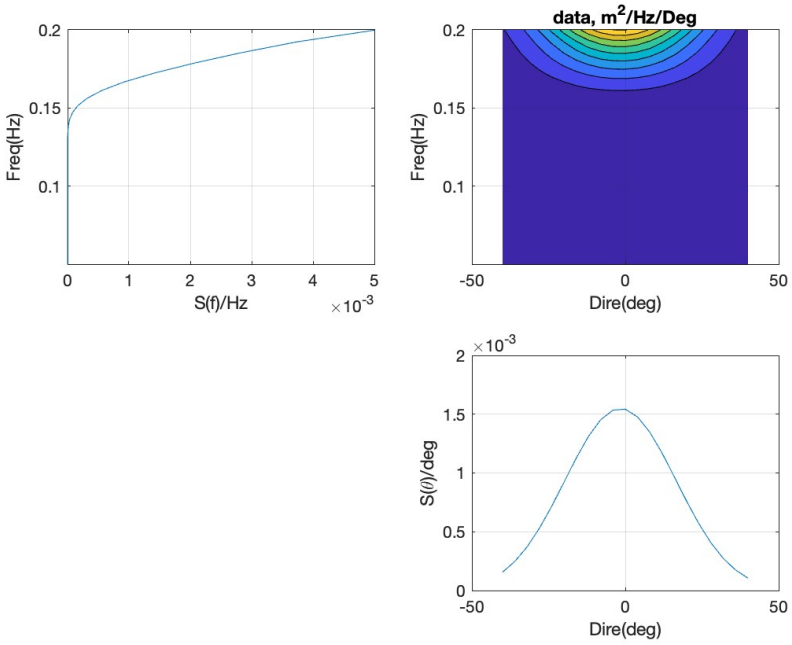

Figure 1: Spectral distribution from spectra_data_1.txt. Use mk_spec_brocc.m.

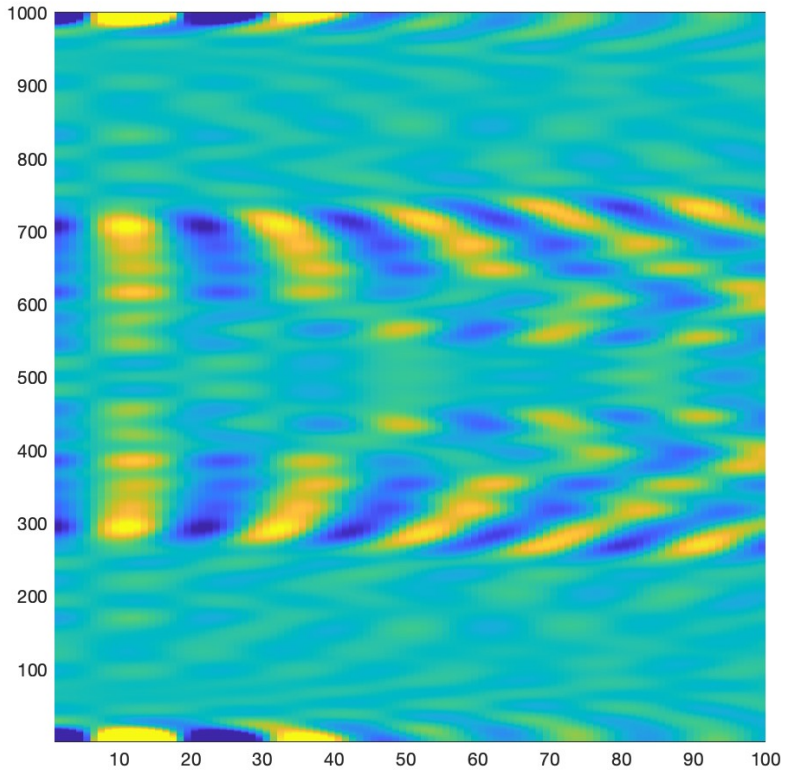

Figure 2: Wave surface from the direct use of the spectra files.

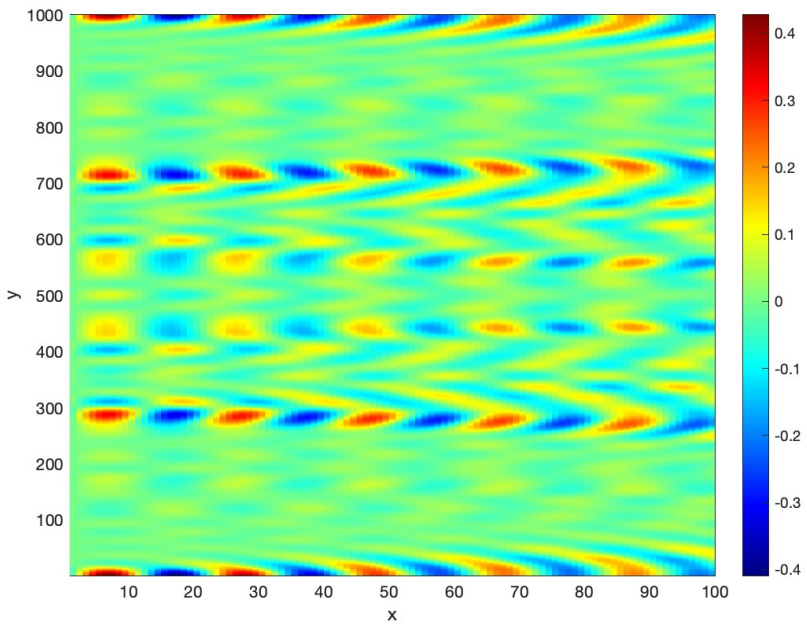

Figure 3: Theoretical solution of wave surface based on the given spectra.

where $(_s)$ is the sources which can be any spatial distribution of hydrodynamic conditions, such as tidal, surge conditions, and wave conditions. Here, we focus on wave conditions, which are the linear solution on a flat bottom. The damping function is the same as that used in a sponge layer (see sponge layer section for detail), i.e., at the west boundary,

$$C_s(i,j) = \alpha_s^{\gamma_s^{i-1}} \tag{4}$$

in which α_s and γ_s are parameters.

The source functions are based on the linear wave solutions at the flat bottom, for component l

$$\eta^l = a^l \cos(k_x^l x + k_y^l y - \sigma^l t + \phi^l) \tag{5}$$

$$u^l = a^l \sigma^l \frac{\cosh k^l(h+z_\alpha)}{\sinh k^l h} \cos(k_x^l x + k_y^l y - \sigma^l t + \phi^l)) \cos(\theta^l) \tag{6}$$

$$v^l = a^l \sigma^l \frac{\cosh k^l(h+z_\alpha)}{\sinh k^l h} \cos(k_x^l x + k_y^l y - \sigma^l t + \phi^l)) \sin(\theta^l) \tag{7}$$

where $(a, k, \sigma, \theta)^l$ are wave amplitude, wavenumber, angular frequency, and wave angle, respectively, for component l. (k_x^l, k_y^l) are the wavenumber components in (x, y): $k_x^l = k^l \cos\theta^l, k_y^l = k^l \sin\theta^l$. z_α is the z at the reference level defined in the Boussinesq theory. Note that ϕ^l) is the phase for component l. You will see later that is super-important for us to solve the issue.

The solution for $l = [1, 2, ...L]$ can be obtained by summating all wave components, for example, for wave surface,

$$\eta = \sum_{l=1}^{L} a^l \cos(k_x^l x + k_y^l y - \sigma^l t + \phi^l) \tag{8}$$

To increase the computational efficiency, we separate the time-independent variables from the formula (Shi et al., 2003). For wave component l, we use two-index numbers, $l(l_f, l_d)$, representing indexes for frequency and wave direction, respectively. $l_f = 1 \sim N_f$, $l_d = 1 \sim N_d$, and $L = N_f \times N_d$.

$$\eta = \sum_{l_f}^{N_f} C^{l_f} \cos\sigma^{l_f} t + \sum_{l_f}^{N_f} S^{l_f} \sin\sigma^{l_f} t \tag{9}$$

where

$$C^{l_f} = \sum_{l_d=1}^{N_d} D^{(l_f,l_d)} \cos\left(k_x^{(l_f,l_d)} x + k_y^{(l_f,l_d)} y + \phi^{(l_f,l_d)}\right) \tag{10}$$

$$S^{l_f} = \sum_{l_d=1}^{N_d} D^{(l_f,l_d)} \sin\left(k_x^{(l_f,l_d)} x + k_y^{(l_f,l_d)} y + \phi^{(l_f,l_d)}\right) \tag{11}$$

Again, the phase information, $\phi^{(l_f,l_d)}$, is important for solving the issue. (9) provides the linear solution of the wave surface.

4. Verification

The problem can be verified using a simple matlab program (in package /Linear_Program/), mk_spec_brocc.m. It reads the file, spectra_data_1.txt (now brocch_data.txt), and draws the wave surface based on (9).

When set up case1='zero_phase', $\phi^{(l_f,l_d)} = 0.0$, the wave surface is shown in Figure 4. You can make an animation by setting TIME=[0:100].

When set up case1='rand_phase', the program uses the random function *rand* to generate $\phi^{(l_f,l_d)}$ with random numbers. The solution for the surface is shown in Figure 5.

5. Improvement of the code for avoiding the problem

The matlab program proved that the issue was caused by the coherence between wave components. Here, the code was modified by adding the random function in the time-dependent spectra module. When a user doesn't provide phase information (a 2D array), the program uses random phases to construct the boundary condition. A check-up output is provided by setting TMP = T in input.txt. The output files are a series of surface elevation based on the linear solution (flat bottom), tmp_xxxxx. A user can compare eta_xxxxx and tmp_xxxxx to see the consistency.

As an example, Figure 6 shows a comparison between the funwave result (eta_00150) and the linear solution (tmp_00150).

In conclusion, my feeling is that wave-coherence can be an important problem if the phases are not prescribed in the model. Salatin et al., (2021) recently addressed the issue in general applications and implemented a new wavemaker in FUNWAVE, which is able to model $0 \sim 100\%$ wave coherence. Please refer to the sample cases in the FUNWAVE package /simple_cases/wave_coherence/.

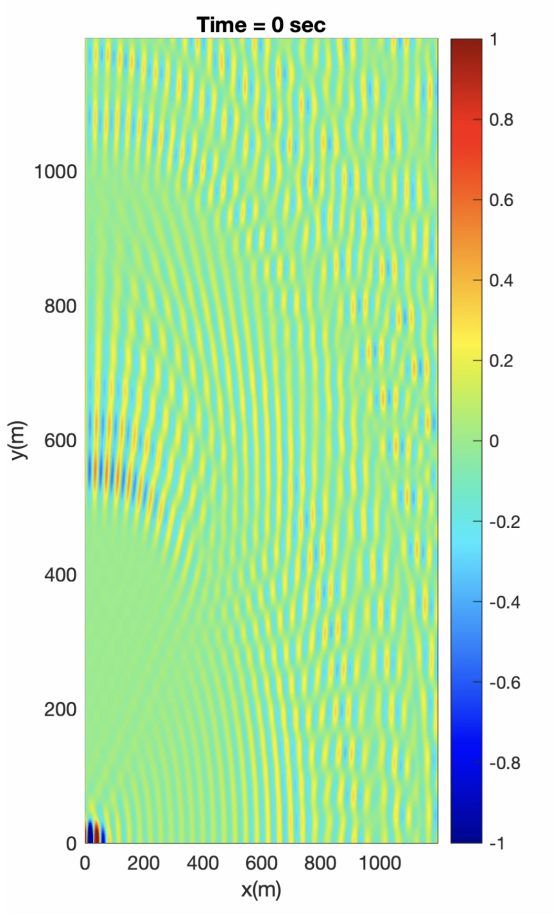

Figure 4: Theoretical solution by setting $\phi^{(l_f, l_d)} = 0.0$.

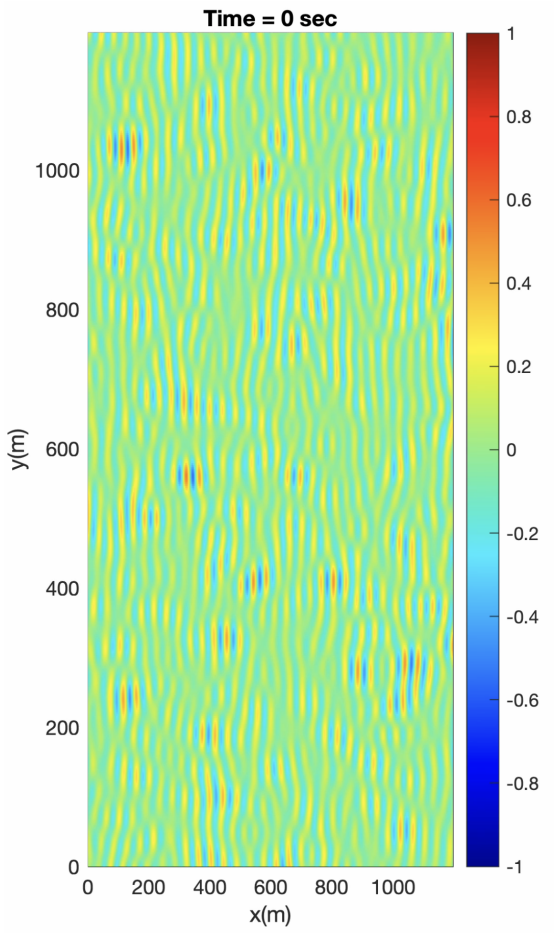

Figure 5: Theoretical solution from random phases.

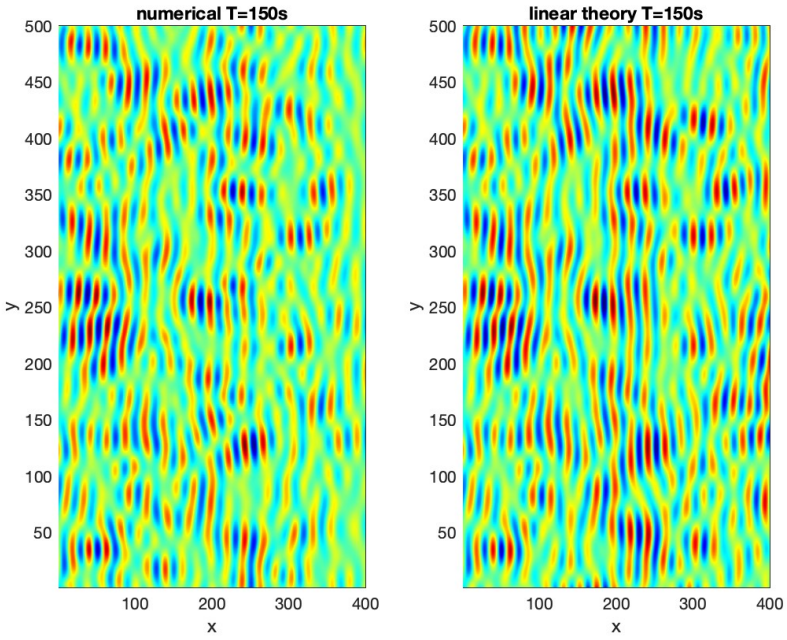

Figure 6: Funwave result (left) versus the linear solution (right).