

WEC-Sim

USER MANUAL

Version 1.0

June 30, 2014

License

Copyright 2014 the National Renewable Energy Laboratory and Sandia Corporation

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Acknowledgments

WEC-Sim is a multilaboratory project sponsored by the U.S. Department of Energy's Wind and Water Power Technologies Office. WEC-Sim code development is a collaboration between the National Renewable Energy Laboratory (NREL)¹ and Sandia National Laboratories (SNL)².

Principal Developers

Kelley Ruehl (SNL)
Carlos Michelen (SNL)

Michael Lawson (NREL)
Yi-Hsiang Yu (NREL)

Contributors

Nathan Tom (NREL)
Sam Kanner (University of
California at Berkeley)

Adam Nelessen (Georgia Tech)
Chris McComb (Carnegie
Mellon University)



¹The National Renewable Energy Laboratory is a national laboratory of the U.S. Department of Energy, Office of Energy Efficiency and Renewable Energy, operated by the Alliance for Sustainable Energy, LLC, under contract No. DE-AC36-08GO28308.

²Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Contents

1	Introduction	5
2	Theory	6
2.1	Introduction	6
2.2	Boundary Element Method	6
2.3	Coordinate System in WEC-Sim	7
2.4	Time-Domain Numerical Method	7
2.4.1	Sinusoidal Steady-State Response Scenario	8
2.4.2	Convolution Integral Formulation	8
2.4.3	Wave Spectrum	9
2.4.4	Ramp Function	11
2.5	Power Take-off Forces	12
2.6	Mooring Forces	13
2.7	Viscous Drag	13
3	Getting Started	14
3.1	Downloading and Installing WEC-Sim	14
3.2	Running WEC-Sim	15
3.2.1	File Structure Overview	15
3.2.2	Steps To Run WEC-Sim	16
3.2.3	Simulation Outputs	18
3.2.4	Postprocessing using User Defined Functions (UDFs)	19
4	Library Structure	20
4.1	Library Structure Overview	20
4.2	Body Elements Sublibrary	20
4.2.1	Rigid Body Block	21
4.3	Frames Sublibrary	21
4.3.1	Global Reference Frame Block	21
4.4	Constraints Sublibrary	22
4.4.1	Floating Block	23
4.4.2	Heave Block	23
4.4.3	Surge Block	24
4.4.4	Pitch Block	24
4.4.5	Fixed Block	24
4.5	PTOs Sublibrary	24
4.5.1	Translation PTO (Local Z) Block	24
4.5.2	Translation PTO (Local X) Block	25
4.5.3	Rotational PTO (Local RY) Block	25

4.6	Other SimMechanics Blocks	26
5	Code Structure and Input	
	Parameters	27
5.1	Units	27
5.2	WEC-Sim Input File	27
5.2.1	Specification of Simulation Parameters	27
5.2.2	Specification of Body Parameters	27
5.2.3	Specification of Wave Parameters	29
5.2.4	Specification of Power Take-Off and Constraint Parameters	29
5.3	WEC-Sim Code	29
5.3.1	<code>simulationClass</code>	29
5.3.2	<code>bodyClass</code>	30
5.3.3	<code>waveClass</code>	32
5.3.4	<code>constraintClass</code>	33
5.3.5	<code>ptoClass</code>	34
6	Applications	35
6.1	Reference Model 3 Two-Body Point Absorber	35
6.1.1	Geometry Definition	35
6.1.2	Modeling RM3 in WEC-Sim	36
6.2	Oscillating Surge-Pitch Device	40
6.2.1	Geometry Definition	40
6.2.2	Modeling OSWEC in WEC-Sim	40
6.3	WEC-Sim Test Cases	45
6.3.1	RM3 Test Case	45
6.3.2	OSWEC Test Case	47

Chapter 1: Introduction

WEC-Sim (Wave Energy Converter SIMulator) is an open source wave energy converter (WEC) simulation tool. The code is developed in MATLAB and Simulink using the multi-body dynamics solver SimMechanics. WEC-Sim has the ability to model devices that are comprised of rigid bodies, power-take-off (PTO) systems, and mooring systems. Hydrodynamic forces are modeled using a radiation and diffraction method, and the system dynamics is performed in the time domain by solving the governing WEC equations of motion in 6 degrees of freedom (DOF).

In this user manual, the theory of WEC-Sim is described in Chapter 2. The installation of MATLAB and WEC-Sim and how to run WEC-Sim is described in Chapter 3. The pre-developed WEC component library structure is presented in Chapter 4. The code WEC-Sim structure and the input parameters for running WEC-Sim are described in Chapter 5. Finally, the application of WEC-Sim to model a point absorber and a oscillating surge device is presented in Chapter 6.

Chapter 2: Theory

2.1 Introduction

Modeling a WEC involves the interaction between the incident waves, device motion, PTO mechanism, and mooring (Figure 2.1). WEC-Sim uses a radiation and diffraction method [1, 2] to predict power performance and design optimization. The radiation and diffraction method generally obtains the hydrodynamic forces from a frequency-domain boundary element method (BEM) solver using linear coefficients to solve the system dynamics in the time domain.

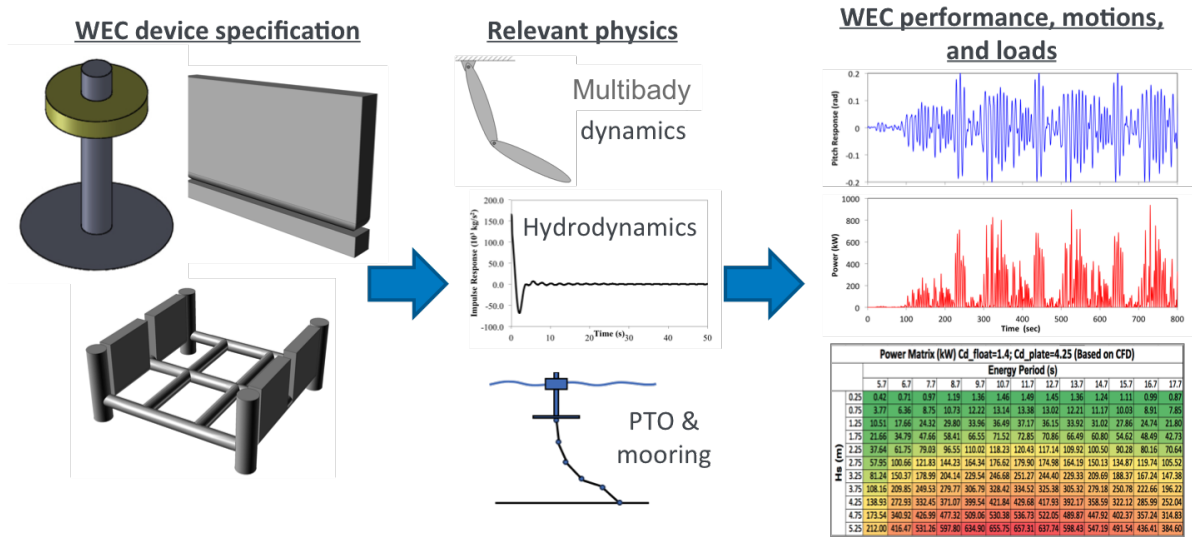


Figure 2.1: Methodology for WEC-Sim

2.2 Boundary Element Method

A common approach to determining the hydrodynamic forces is to presume that they are the sum of incident, radiated, and diffracted wave components. These forcing components are modeled using linear coefficients ideally obtained from a frequency-domain potential flow BEM solver (e.g., WAMIT [3], AWQA-FER [4], and Nemoh [5]). The BEM solutions are obtained by solving the Laplace equation for the velocity potential, which assumes the flow is inviscid, incompressible, and irrotational. More details on the theory for the frequency-domain BEM can be found in [3].

2.3 Coordinate System in WEC-Sim

Figure 2.2 illustrates a 3-D floating point absorber subject to incoming waves in water. The figure also defines the coordinates and the 6 DOF in WEC-Sim. The WEC-Sim coordinate system assumes that the X axis is in the direction of wave propagation if the wave heading angle is equal to zero. The Z axis is in the vertical upwards direction, and the Y axis direction is defined by the right-hand rule.

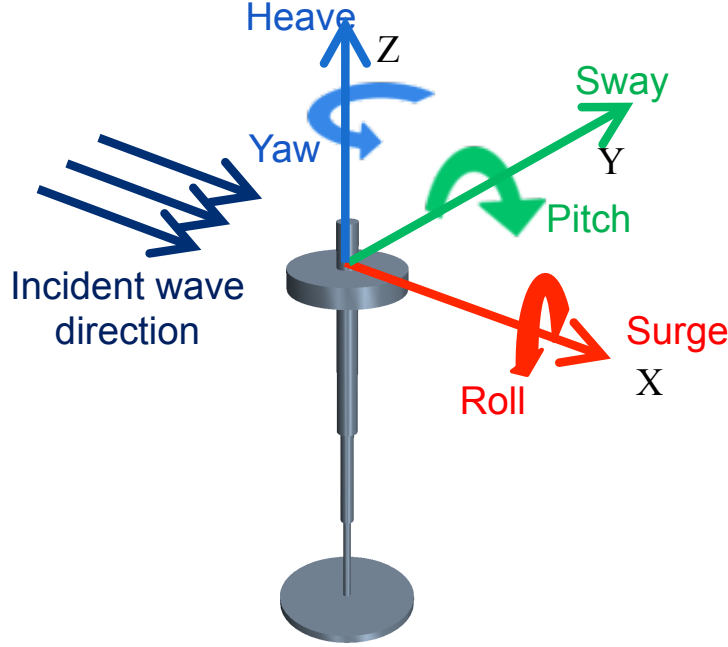


Figure 2.2: Sketch defining the coordinate system

2.4 Time-Domain Numerical Method

The dynamic response of the system was calculated by solving the equation of motion for WEC systems [2, 6]. The equation of motion for a floating body, about its center of gravity, can be given as:

$$m\ddot{X} = F_{ext} + F_{rad} + F_{PTO} + F_v + F_B + F_m, \quad (2.1)$$

where \ddot{X} is the (translational and rotational) acceleration vector of the device, m is the mass matrix, F_{ext} is the wave excitation force vector, F_{rad} is the force vector resulting from wave radiation, F_{PTO} is the PTO force vector, F_v is the viscous damping force vector, F_B is the net buoyancy restoring force vector, and F_m is the force vector resulting from the mooring connection.

Both F_{ext} and F_{rad} are calculated from values provided by the frequency-domain BEM solver. The radiation term includes an added-mass and wave damping term associated with the acceleration and velocity of the floating body, respectively. The wave excitation term includes a Froude–Krylov force component generated by the undisturbed incident waves and a diffraction component that results from the presence of the floating body. WEC-Sim can be used for regular and irregular wave simulations, but note that F_{ext} and F_{rad} are calculated differently for sinusoidal steady-state response scenarios and random sea simulations. The sinusoidal steady-state response is often used for simple WEC designs with regular incoming waves. However, for random sea simulations or any simulations where fluid memory effects of the system are essential, the convolution integral method is recommended to represent the fluid memory retardation force on the floating body.

2.4.1 Sinusoidal Steady-State Response Scenario

This approach assumes that the system response is in sinusoidal steady-state form, and is only valid for regular wave simulations. The radiation term can be calculated using the added mass and the wave radiation damping term for a given wave frequency, which is obtained from

$$F_{rad} = -A(\omega_r)\ddot{X} - B(\omega_r)\dot{X}, \quad (2.2)$$

where $A(\omega_r)$ and $B(\omega_r)$ are the added-mass and wave radiation damping matrices, respectively. ω_r is the wave frequency (in rad/sec), and \dot{X} is the velocity vector of the floating body.

The free surface profile is based on linear wave theory for a given wave height, wave frequency, and water depth. The regular wave excitation force is obtained from

$$F_{ext} = \Re \left[R_f \frac{H}{2} F_X(\omega_r) e^{i(\omega_r t)} \right], \quad (2.3)$$

where \Re denotes the real part of the formula, R_f is the ramp function, H is the wave height, and F_X is the excitation vector, including the magnitude and phase of the force.

2.4.2 Convolution Integral Formulation

To include the fluid memory effect on the system, the convolution integral calculation, which is based upon the Cummins equation [7], is used. The radiation term can be calculated by

$$F_{rad} = -A_\infty \ddot{X} - \int_0^t K(t - \tau) \dot{X}(\tau) d\tau, \quad (2.4)$$

where A_∞ is the added mass matrix at infinite frequency and K is the impulse response function.

For regular waves, Eq. 2.3 is used to calculate the wave excitation force. For irregular waves, the free surface elevation is constructed from a linear superposition of a number of regular wave components. It is often characterized using a wave spectrum (Figure 2.3) that describes the wave energy distribution over a range of wave frequencies, characterized by a significant wave height and peak wave period. The irregular excitation force can be calculated as the real part of an integral term across all wave frequencies as follows

$$F_{ext} = \Re \left[R_f \int_0^{\infty} \sqrt{2S(\omega_r)} F_X(\omega_r) e^{i(\omega_r t + \phi)} d\omega_r \right], \quad (2.5)$$

where S is the wave spectrum and ϕ is a random phase angle.

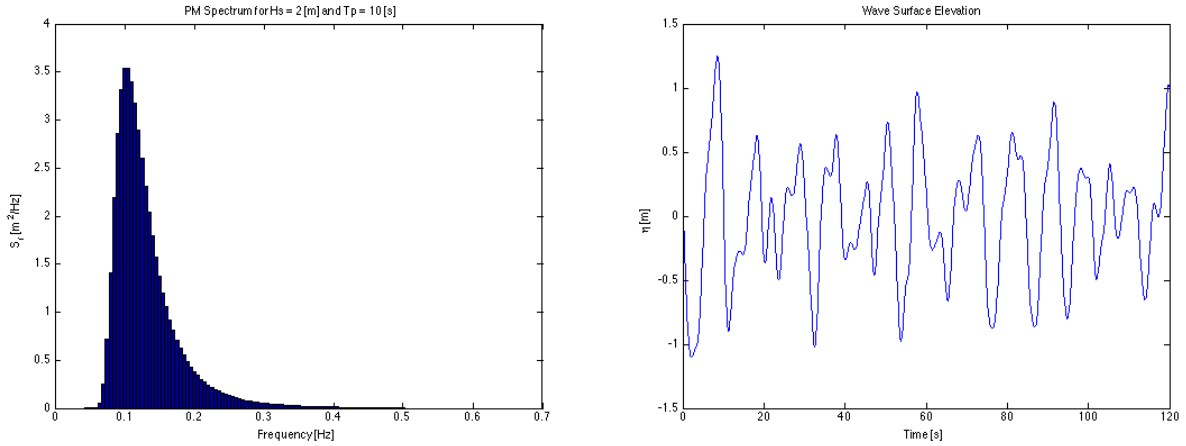


Figure 2.3: An example of wave spectrum and irregular wave elevation generated by WEC-Sim (Pierson–Moskowitz spectrum)

2.4.3 Wave Spectrum

The ability to generate regular waves provides an opportunity to observe the response of a model under specific conditions. Sea states with constant wave heights and periods are rarely found outside wave tank test. Normal sea conditions are more accurately represented by random-wave time series that model the superposition of various wave forms with different amplitudes and periods. This superposition of waves is characterized by a sea spectrum. Through statistical analysis, spectra are characterized by specific parameters such as significant wave height, peak period, wind speed, fetch length, and others. The common types of spectra that are used by the offshore industry are discussed in the following sections. The general form of the sea spectrums available in WEC-Sim is given by:

$$S(f) = Af^{-5} \exp[-Bf^{-4}] \quad , \quad (2.6)$$

where f is the wave frequency (in Hertz) and \exp stands for the exponential function.

Pierson–Moskowitz

One of the simplest spectra was proposed by [8]. It assumed that after the wind blew steadily for a long time over a large area, the waves would come into equilibrium with the wind. This is the concept of a fully developed sea, where a "long time" is roughly 10,000 wave periods, and a "large area" is roughly 5,000 wave-lengths on a side. The spectrum is calculated from

$$S(f) = \frac{\alpha_{PM} g^2}{(2\pi)^4} f^{-5} \exp \left[-\frac{5}{4} \left(\frac{f_p}{f} \right)^4 \right] , \quad (2.7)$$

$$A = \frac{\alpha_{PM} g^2}{(2\pi)^4} , \quad B = \frac{5}{4} f_p^4 , \quad (2.8)$$

where $\alpha_{PM} = 0.0081$, g is gravitational acceleration, and f_p is the peak frequency of the spectrum. However, this spectrum representation does not allow the user to define the significant wave height. To facilitate the creation of a power matrix, in WEC-Sim the α_{PM} coefficient was calculated such that the desired significant wave height of the sea state was met. The α_{PM} fit was calculated as follows:

$$\alpha_{PM} = \frac{H_{m0}^2}{16 \int_0^\infty S^*(f) df} , \quad (2.9)$$

$$S^*(f) = \frac{g^2}{(2\pi)^4} f^{-5} \exp \left[-\frac{5}{4} \left(\frac{f_p}{f} \right)^4 \right] . \quad (2.10)$$

Note that related to the spectrum is a series of characteristic numbers called the spectral moments. These numbers, denoted m_k , $k = 0, 1, 2, \dots$ are defined as

$$m_k = \int_0^\infty f^k S(f) df . \quad (2.11)$$

The spectral moment, m_0 is the variance of the free surface, which allows one to define

$$H_{m0} = 4\sqrt{m_0} . \quad (2.12)$$

Bretschneider Spectrum

This two-parameter spectrum is based on significant wave height and peak wave frequency. For a given significant wave height, the peak frequency can be varied to cover a range of conditions including developing and decaying seas. In general, the parameters depend on wind speed (most important), wind direction, as well as fetch and locations of storm fronts. The spectrum is given as

$$S(f) = \frac{H_{m0}^2}{4} (1.057 f_p)^4 f^{-5} \exp \left[-\frac{5}{4} \left(\frac{f_p}{f} \right)^4 \right] , \quad (2.13)$$

$$A = \frac{H_{m0}^2}{4} (1.057 f_p)^4 \approx \frac{5}{16} H_{m0}^2 f_p^4 , \quad (2.14)$$

$$B = (1.057 f_p)^4 \approx \frac{5}{4} f_p^4 , \quad (2.15)$$

where H_{m0} is the significant wave height which is generally defined as the mean wave height of the one third highest waves.

JONSWAP (Joint North Sea Wave Project) Spectrum

Spectrum used in WEC-Sim

The spectrum was purposed by Hasselmann et al. [9], and the original formulation was given as

$$S(f) = \frac{\alpha_j g^2}{(2\pi)^4} f^{-5} \exp \left[-\frac{5}{4} \left(\frac{f_p}{f} \right)^4 \right] \gamma^\Gamma$$

$$\Gamma = \exp \left[-\left(\frac{\frac{f}{f_p} - 1}{\sqrt{2}\sigma} \right)^2 \right], \sigma = \begin{cases} 0.07 & f \leq f_p \\ 0.09 & f > f_p \end{cases}, \quad (2.16)$$

$$A = \frac{\alpha_j g^2}{(2\pi)^4}, \quad B = \frac{5}{4} f_p^4, \quad (2.17)$$

where α_j is a nondimensional variable that is a function of the wind speed and fetch length.

Empirical fits were applied in an attempt to find a mean value that would capture the spectral shape of most measured sea states. To fit α_j to match the desired significant wave height the following calculation must be performed

$$\alpha_j = \frac{H_{m0}^2}{16 \int_0^\infty S^*(f) df}, \quad (2.18)$$

$$S^*(f) = \frac{g^2}{(2\pi)^4} f^{-5} \exp \left[-\frac{5}{4} \left(\frac{f_p}{f} \right)^4 \right] \gamma^\Gamma. \quad (2.19)$$

Spectrum purposed at ITTC

Another form of JONSWAP spectrum was purposed at the 17th International Towing Tank Conference (ITTC). It was defined as

$$S(f) = \frac{155}{(2\pi)^4} \frac{H_{m0}^2}{(0.834T_p)^4} f^{-5} \exp \left[-\frac{5}{4} \left(\frac{f_p}{f} \right)^4 \right] \gamma^\Gamma$$

$$\approx \frac{310}{(2\pi)^4} H_{m0}^2 f_p^4 f^{-5} \exp \left[-\frac{5}{4} \left(\frac{f_p}{f} \right)^4 \right] \gamma^\Gamma, \quad (2.20)$$

$$\Gamma = \exp \left[-\left(\frac{\frac{f}{f_p} - 1}{\sqrt{2}\sigma} \right)^2 \right], \quad \sigma = \begin{cases} 0.07 & f \leq f_p \\ 0.09 & f > f_p \end{cases}, \quad (2.21)$$

$$A = \frac{310}{(2\pi)^4} H_{m0}^2 f_p^4, \quad B = \frac{5}{4} f_p^4. \quad (2.22)$$

Figure 2.4 shows the comparison of the JONSWAP spectrum obtained from the α_j fit and the ITTC description. It is clear that the two methods have very good agreement.

2.4.4 Ramp Function

A ramp function (R_f), necessary to avoid strong transient flows at the earlier time steps of the simulation, is used to calculate the wave excitation force. The ramp function is given by

$$R_f = \begin{cases} \frac{1}{2}(1 + \cos(\pi + \frac{\pi t}{t_r})), & \frac{t}{t_r} < 1 \\ 1, & \frac{t}{t_r} \geq 1 \end{cases}, \quad (2.23)$$

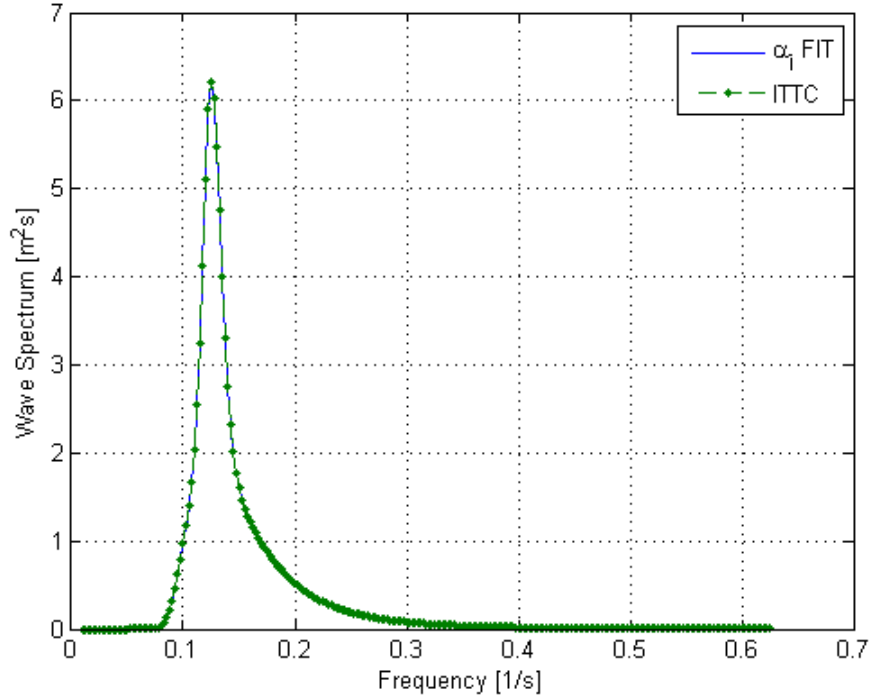


Figure 2.4: Comparison of α_j fit to the ITTC description of the JONSWAP spectrum with $H_{m0} = 2$ m and peak period (T_p) of 8 sec.

where t is the simulation time and t_r is the ramp time.

2.5 Power Take-off Forces

The PTO mechanism is represented as a linear spring-damper system, where the reaction force is given by:

$$F_{PTO} = -K_{PTO}X_{rel} - C_{PTO}\dot{X}_{rel}, \quad (2.24)$$

where K_{PTO} is the stiffness of the PTO, C_{PTO} is the damping of the PTO, and X_{rel} and \dot{X}_{rel} are the relative motion and velocity between two bodies. The power consumed by the PTO is given by:

$$P_{PTO} = -F_{PTO}\dot{X}_{rel} = \left(K_{PTO}X_{rel}\dot{X}_{rel} + C_{PTO}\dot{X}_{rel}^2 \right). \quad (2.25)$$

However, the relative motion and velocity between two bodies is out of phase by $\pi/2$, resulting in a time-averaged product of 0. This allows the absorbed power to be written as

$$P_{PTO} = C_{PTO}\dot{X}_{rel}^2. \quad (2.26)$$

2.6 Mooring Forces

The mooring load is represented using a linear quasi-static mooring stiffness, which can be calculated by

$$F_m = -K_m X, \quad (2.27)$$

where K_m is the stiffness matrix for the mooring system, and X is the response of the body.

2.7 Viscous Drag

Generally, the effect of viscosity on the WEC dynamics needs to be considered as neglecting this effect may lead to an overestimation of the power generation of the system, particularly when a linear model is applied. A common way of modeling the viscous damping is to add a (Morison-equation-type) quadratic damping term to the equation of motion;

$$F_V = \frac{1}{2} C_d \rho A_D \dot{X} |\dot{X}|, \quad (2.28)$$

where C_d is the viscous drag coefficient, ρ is the fluid density, and A_D is the characteristic area. The viscous drag coefficient for the device must be carefully selected [1, 2]; however, it is dependent on device geometry, scale, and relative velocity between the body and the flow around it. The drag coefficient becomes much larger when the Reynolds and the Keulegan–Carpenter number are smaller. Note that empirical data on the drag coefficient can be found in various literature and standards. The available data may, however, be limited to existing simple geometries. For practical point absorber geometry, the hydrodynamic forces may have to be evaluated by conducting wave tank tests or prescribed motion computational fluid dynamic simulations.

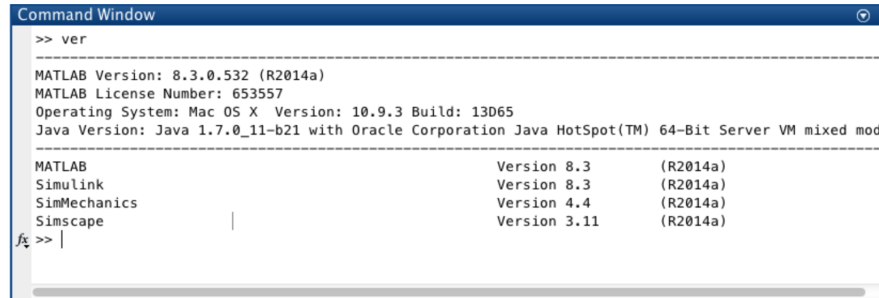
Chapter 3: Getting Started

WEC-Sim is implemented in MATLAB [10] and running the code requires that you install MATLAB, the MATLAB toolboxes presented in Table 3.1, and the WEC-Sim source code. WEC-Sim was developed in MATLAB R2014a and we recommend using this MATLAB version. In this chapter we describe how to download and install WEC-Sim (Section 3.1) and how to run a WEC-Sim simulation (Section 3.2).

3.1 Downloading and Installing WEC-Sim

Step 1: Install MATLAB

Download and install MATLAB and the MATLAB toolboxes presented in Table 3.1. Ensure you have the required toolboxes installed by running the `ver` command from the MATLAB Command Window (Figure 3.1). Consult the MathWorks website <http://www.mathworks.com> for more information on performing this step.



```
Command Window
>> ver

-----
MATLAB Version: 8.3.0.532 (R2014a)
MATLAB License Number: 653557
Operating System: Mac OS X Version: 10.9.3 Build: 13D65
Java Version: Java 1.7.0_11-b21 with Oracle Corporation Java HotSpot(TM) 64-Bit Server VM mixed mod
-----

MATLAB                               Version 8.3      (R2014a)
Simulink                             Version 8.3      (R2014a)
SimMechanics                         Version 4.4      (R2014a)
Simscape                             Version 3.11     (R2014a)
fz >> |
```

Figure 3.1: Running the `ver` command from the MATLAB Command Window.

Step 2: Download WEC-Sim

Download WEC-Sim from the OpenEI website <http://en.openei.org/wiki/WEC-Sim/>.

Table 3.1: Required MATLAB toolboxes

Matlab package	Required release
MATLAB Base	R2014a Version 8.3
Simulink	R2014a Version 8.3
SimMechanics	R2014a Version 4.4
Simscape	R2014a Version 3.11

Step 3: Add the WEC-Sim Source Code to the MATLAB Search Path

Open the `wecSimStartup.m` file that is located in the `functions` folder within the WEC-Sim source code directory (referred to as `wecSimSource` in this document). Copy the code in the `wecSimStartup.m` file and paste it into the `startup.m` file within the MATLAB Startup Folder. Set the `wecSimPath` variable to the location of the `wecSimSource` folder on your computer. Close MATLAB, restart the code, and then run the `path` command from the MATLAB Command Window to verify that the directories listed in `wecsim-path-setup.m` are in the MATLAB Search Path.

Step 4: Add the WEC-Sim Blocks to the Simulink Library Browser

Open the Simulink Library Browser by typing `simulink` from the MATLAB Command Window. Once the Simulink Library Browser opens, select **View**▷**Refresh Tree View**. At this point, you should be able to expand the WEC-Sim menu in the **Libraries** pane to view the WEC-Sim Body, Constraints, PTOs, and Frame blocks. The function of these blocks will be explained in Chapter 4. From now on, every time you open Simulink in the WEC-Sim Library Browser, the WEC-Sim Body Elements, Constraints, PTOs, and Frames blocks will be available. For more help using and modifying library blocks, please refer to the Simulink Documentation <http://www.mathworks.com/help/simulink/>.

3.2 Running WEC-Sim

This section gives an overview of the WEC-Sim work flow and how to run WEC-Sim. Here we describe the file structure of the WEC-Sim runs and the steps for setting up and running a WEC-Sim simulation. Detailed descriptions and options for input parameters for the input file are described in Chapter 5. Specific examples of using WEC-Sim to simulate WEC devices are presented in Chapter 6.

3.2.1 File Structure Overview

Table 3.2 shows the default file structure for WEC-Sim. All the necessary files for running WEC-Sim are located within a user defined folder, referred to herein as the WEC-Sim case folder or case directory.

Table 3.2: Default files names and their locations

	File name	Location
Input file	<code>wecSimInputFile.m</code>	Case directory
WEC Model	<code>WEC Model Name.slx</code>	Case directory
WAMIT	<code>WAMIT File Name.out</code>	wamit
Geometry	<code>STL File Name.stl</code>	geometry

3.2.2 Steps To Run WEC-Sim

The overview of the WEC-Sim work flow is shown in Figure 3.2. We describe the steps in setting up and running a WEC-Sim simulations in the following:

Step 1: Pre-Processing

Run WAMIT to generate the hydrodynamic coefficients for each body of the WEC device. WEC-Sim will read the WAMIT-generated hydrodynamic coefficients from the WAMIT output file (<wamit file name>.out). To ensure that WEC-Sim uses the correct hydrodynamic coefficients to model the WEC system, the center of gravity for each body MUST be at the origin of the body coordinate system (XBODY) in WAMIT. Note that the current version of WEC-Sim does not account for the multidirectional wave heading and WEC-Sim will use whatever wave heading was modeled in WAMIT. More details on WAMIT setup are given in the WAMIT User Manual [3].

Next the user must create representations of the WEC bodies in the STL file format. The STL files are used to visualize the WEC bodies in the WEC-Sim/MATLAB graphical user interface.

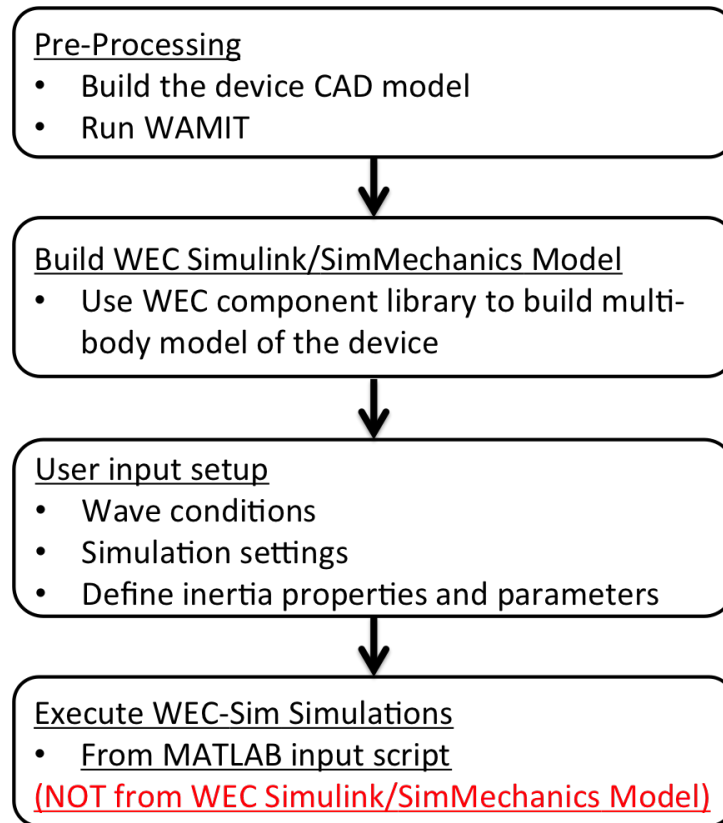


Figure 3.2: Work flow diagram for running WEC-Sim simulations

Step 2: Build Device Simulink/SimMechanics Model

Next, the user must build the device model using the Simulink/SimMechanics toolboxes and the WEC-Sim Library (see Chapter 4). Figure 3.3 shows an example of a two-body point absorber modeled in Simulink/SimMechanics.

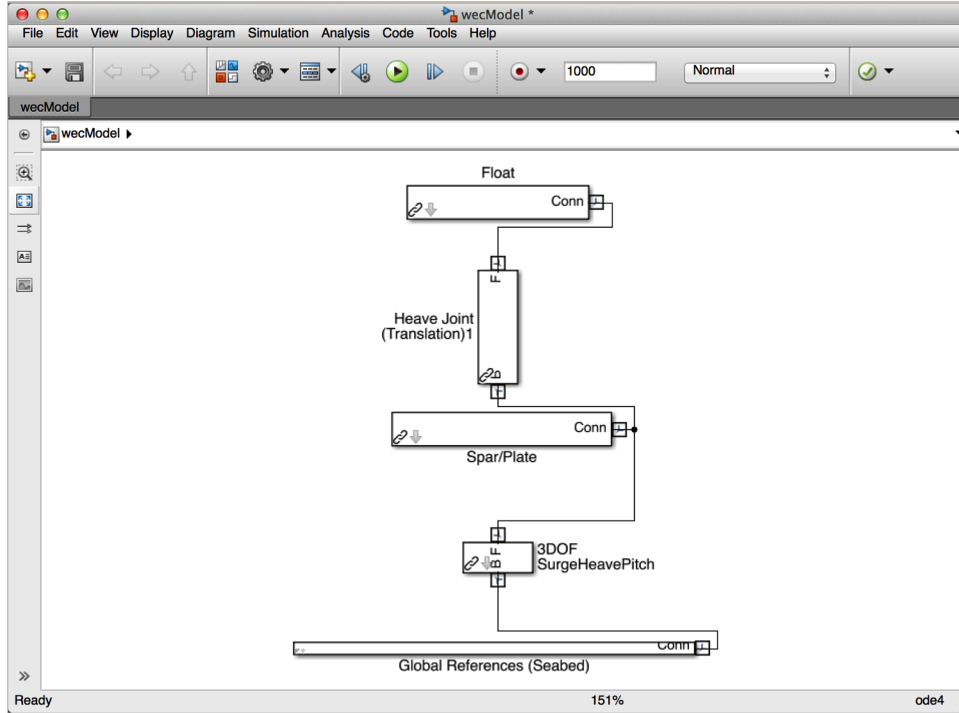


Figure 3.3: An example of device Simulink/SimMechanics model for a two-body point absorber

Step 3: Create WEC-Sim Input File

A WEC-Sim input file needs to be created in the case directory, and it MUST be named `wecSimInputFile.m`. An example of the input file for a two-body point absorber is shown in Figure 3.4. In the input file, the simulation settings, sea state, body mass properties, PTO, and constraints are specified. In addition, users MUST specify the Simulink/SimMechanics model file name in the `wecSimInputFile.m`, which is

```
simu.simMechanicsFile='<WEC Model Name>.slx'.
```

Step 4: Execute WEC-Sim

Finally, execute the simulation by running the `wecSim` command from the MATLAB Command Window (Figure 3.4). The `wecSim` command must be executed in the WEC-Sim case directory where the `wecSimInputFile.m` is located.

WEC-Sim simulations should always be executed from the MATLAB Command Window and not from the Simulink/SimMechanics model. This ensures that the correct variables are in the MATLAB workspace during simulation.

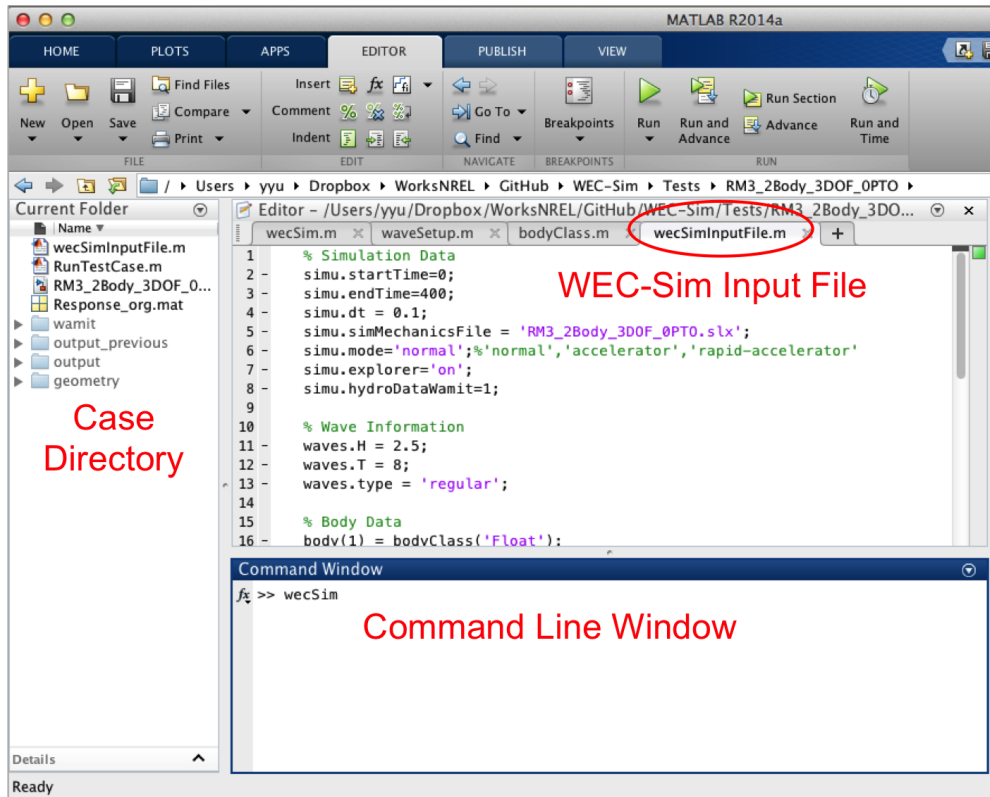


Figure 3.4: An example of running WEC-Sim

3.2.3 Simulation Outputs

All simulation outputs are saved in the `output` variable within the MATLAB workspace. Specifically, the output variable contains forces and motions of the WEC bodies, PTSs, and constraints. The `output` data file also contains time step information from the simulation.

At the completion of a simulation, WEC-Sim also saves all simulation data in the `output` directory within the WEC-Sim case file in three data files:

`<case name>_simulationLog.txt` : This ascii text file contains all information displayed in the MATLAB command window during a simulation.

`<caseName>_output.mat` : This MATLAB data file contains forces and motions of the WEC bodies, PTSs, and constraints. The data file also contains time step information from the simulation.

`<caseName>_matlabWorkspace` : This MATLAB data file contains all workspace variables from the simulation. Note that this variable also contains the `output.mat` data in a variable named `output`.

3.2.4 Postprocessing using User Defined Functions (UDFs)

Just before completion of a simulation, WEC-Sim looks for a MATLAB m-file named `userDefinedFunctions.m` within the WEC-Sim case directory. If the file exist, and WEC-Sim executes any commands within this m-file. The user can add code to this m-file to perform common post-processing tasks before the simulation exits. One example would be to plot the device motions and forces acting on the bodies and joints of a WEC device. The RM3 test case provides an example of how to plot force and response using user defined functions. Note that the user can perform any calculation desired within the `userDefinedFunctions.m`, and the user has access to all data within the MATLAB workspace.

Chapter 4: Library Structure

4.1 Library Structure Overview

The WEC-Sim library is divided into 4 sublibraries. The user should be able to model their WEC device using the available WEC-Sim blocks, and possibly some SimMechanics blocks. Table 4.1 lists the WEC-Sim blocks and their organization into sublibraries.

Table 4.1: WEC-Sim library structure

WEC-Sim Library	
Sublibrary	Blocks
Body Elements	Rigid Body
Frames	Global Reference Frame
Constraints	Heave Pitch Surge Fixed Floating
PTOs	Rotational PTO (Local RY) Translational PTO (Local X) Translational PTO (Local Z)

In the following sections, we will describe the four sublibraries and their general purpose. The **Body Elements** sublibrary contains the **Rigid Body** block used to simulate the different bodies. The **Frames** sublibrary contains the **Global Reference Frame** block necessary for every simulation. The **Constraints** sublibrary contains blocks that are used to constrain the DOF of the bodies, without including any additional forcing or resistance. The **PTOs** sublibrary contains blocks used to both simulate a PTO system and restrict the body motion. Both constraints and PTOs can be used to restrict the relative motion between multibody systems.

4.2 Body Elements Sublibrary

The **Body Elements** sublibrary (Figure 4.1) contains one block, **Rigid Body** block. It is used to represent rigid bodies. At least one instance of this block is required in each model.

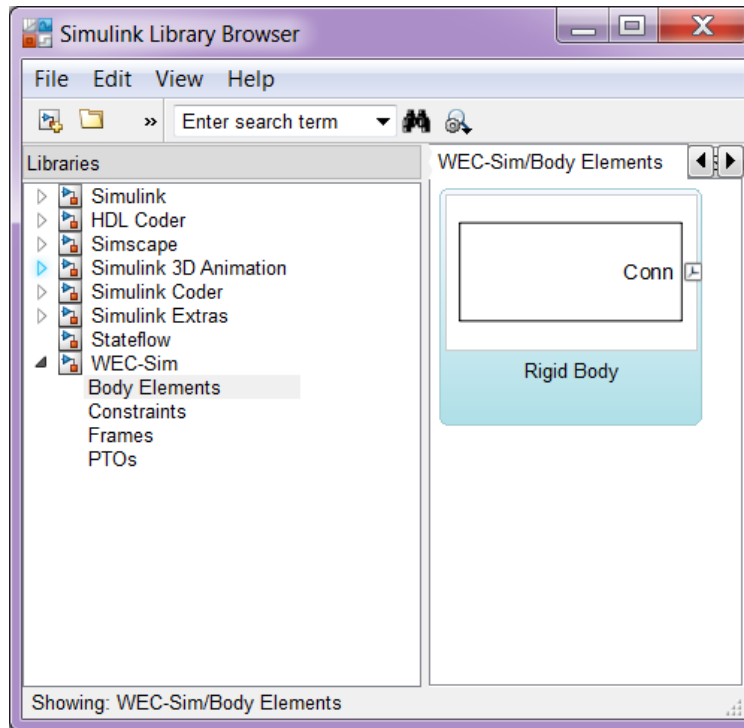


Figure 4.1: Body Elements sublibrary

4.2.1 Rigid Body Block

The **Rigid Body** block is used to represent a rigid body in the simulation. The user has to name the blocks 'body(i)' where $i=1,2,\dots$. The mass properties, hydrodynamic data, geometry file, mooring, and other properties are then specified in the input file. Within the body block the wave radiation, wave excitation, hydrostatic restoring, viscous damping and mooring forces are calculated.

4.3 Frames Sublibrary

The **Frames** sublibrary, shown in Figure 4.2, contains one block that is necessary in every model. The **Global Reference Frame** block defines global references and can be thought of as the seabed.

4.3.1 Global Reference Frame Block

The **Global Reference Frame** block defines the solver configuration, seabed and free surface description, simulation time, and other global settings. It can be useful to think of the **Global Reference Frame** as being the seabed when creating a model. Every model requires one

instance of the **Global Reference Frame** block. The **Global Reference Frame** block uses the simulation class variable `simu` and the wave class variable `waves`, which must be defined in the input file.

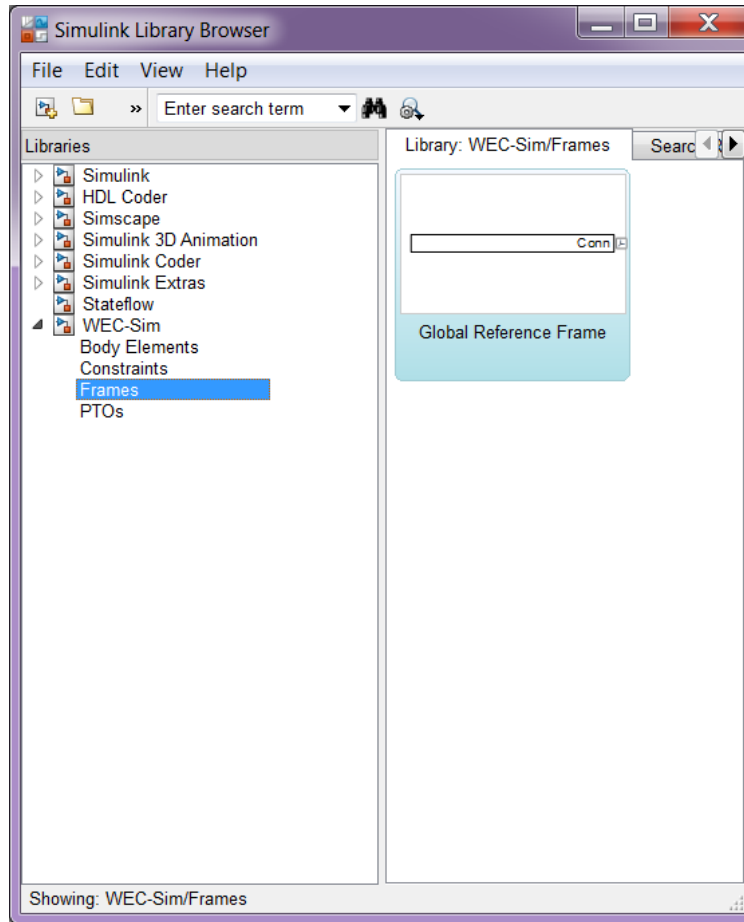


Figure 4.2: **Frames** sublibrary

4.4 Constraints Sublibrary

The blocks within the **Constraints** sublibrary, shown in Figure 4.3, are used to define the DOF of a specific body. **Constraints** blocks define only the DOF, but do not otherwise apply any forcing or resistance to the body motion. Each **Constraints** block has two connections, a base (B) and a follower (F). The **Constraints** block restricts the motion of the block that is connected to the follower relative to the block that is connected to the base. The base of these blocks is typically the **Global Reference Frame** (which can be thought of as the seabed) and the follower is a **Rigid Body**.

There are five **Constraints** blocks, including three that restrict motion to one DOF (**Heave**, **Surge**, **Pitch**), a free-floating (**Floating**) block, and a rigid connection (**Fixed**) block. The rest of this section will describe each **Constraints** block in more detail.

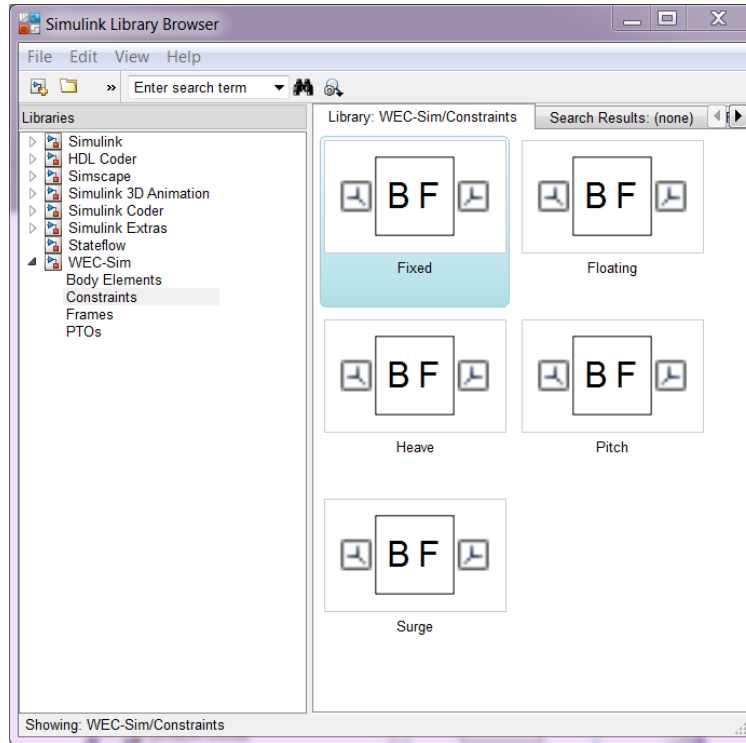


Figure 4.3: Constraints sublibrary

4.4.1 Floating Block

The **Floating** block is used to simulate a free-floating body. It constrains the motion of the follower to be along the XZ plane of the base. That is, it allows translation in the X- and Z-axis, and rotation about the Y-axis. It is usually used with the base connected to the **Global Reference Frame** (seabed), in which case the motion of the follower is along the global XZ plane.

4.4.2 Heave Block

The **Heave** block constrains the motion of the follower relative to the base to be along the Z-axis. In the case of the base connected to the **Global Reference Frame** (seabed), the body is allowed to move only in the vertical (Z) direction. In the case of the **Heave** block connecting two bodies, the relative motion of the two bodies is constrained to be only along their Z-axes. The Z-axis of the follower and base will always be parallel and their perpendicular distance will be constant. The actual direction of movement of the follower depends on the orientation of the base.

4.4.3 Surge Block

The **Surge** block constrains the motion of the follower relative to the base to be along the X-axis. If the base is connected to the **Global Reference Frame** (seabed), the body is allowed to move only in the horizontal (X) direction. If **Surge** block is connects two bodies, the relative motion of the two bodies is constrained to be only along their X-axes. The X-axis of the follower and base will always be parallel and their perpendicular distance will be constant. The actual direction of movement of the follower depends on the orientation of the base.

4.4.4 Pitch Block

The **Pitch** block constrains the relative motion between the follower and the base to be pitch rotation only (about the Y-axis). The distance from both body-fixed coordinate systems to the point of rotation stays constant. The orientation of both body-fixed Y-axes also stays constant. The user **MUST** enter the point about which the rotation occurs as the constraint's location in the input file.

4.4.5 Fixed Block

The **Fixed** block is a rigid connection that constrains all motion between the base and follower. It restricts translation in the X- and Z-axis, and rotation about the Y-axis. Its most common use is for a rigid body fixed to the seabed.

4.5 PTOs Sublibrary

The **PTOs** sublibrary, shown in Figure 4.4, is used to simulate simple PTO systems and to restrict relative motion between multiple bodies or between one body and the seabed. The **PTO** blocks can simulate simple PTO systems by applying a linear stiffness and damping to the connection. Similar to the **Constraints** blocks, the **PTO** blocks have a base (B) and a follower (F). Users **MUST** name each **PTO** block 'pto(i)' where $i=1,2, \dots$, and then define their properties in the input file.

4.5.1 Translation PTO (Local Z) Block

The **Translation PTO (Local Z)** is identical to the **Heave** constraint, but applies a linear stiffness and damping coefficient to the connection. The user has to name the PTOs as described earlier. The user then specifies the stiffness coefficient (in N/m), and damping coefficient (in Ns/m) in the input file.

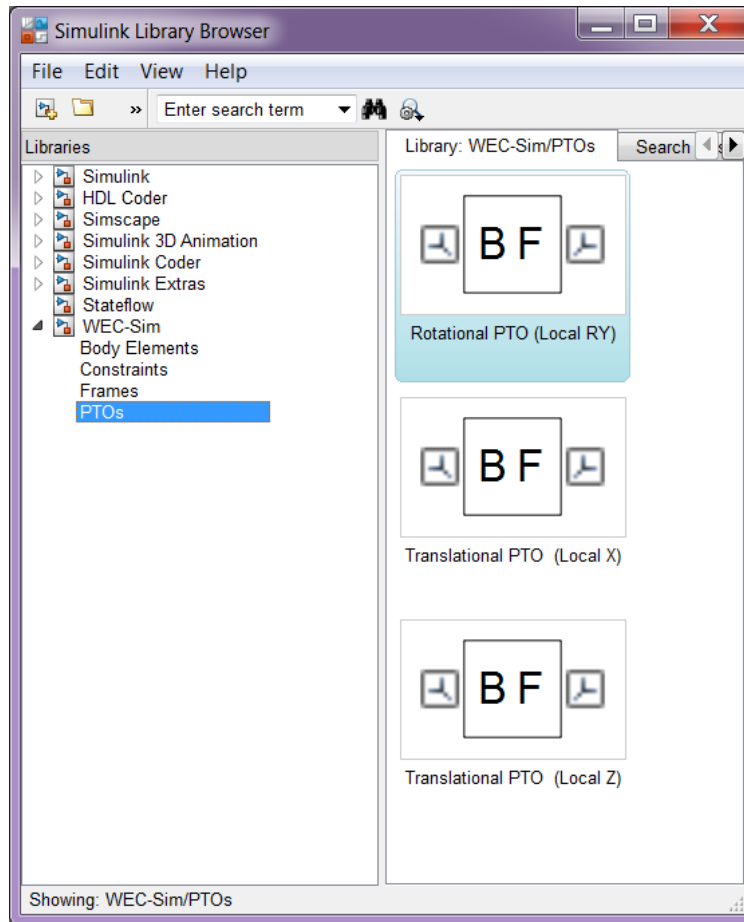


Figure 4.4: PTOs sublibrary

4.5.2 Translation PTO (Local X) Block

The Translation PTO (Local X) is identical to the **Surge** constraint, but additionally applies a linear stiffness and damping coefficient to the connection. The user has to name the PTOs as described earlier. The user then specifies the stiffness coefficient (in N/m), and damping coefficient (in Ns/m) in the input file.

4.5.3 Rotational PTO (Local RY) Block

The Rotational PTO (Local RY) is identical to the **Pitch** constraint, but adds a linear rotational stiffness and damping coefficient to the connection. The user has to name the PTOs as described earlier. The user then specifies the stiffness coefficient (in Nm/rad) and damping coefficient (in Nms/rad) in the input file.

4.6 Other SimMechanics Blocks

In some situations, users may have to use SimMechanics blocks not included in the WEC-Sim Library to build their WEC model. One commonly used block is the **Rigid Transform**, which can be used to rotate the frames on PTOs, cconstraints, and bodies. This is also explained in the SimMechanics User's Guide [11].

Chapter 5: Code Structure and Input Parameters

This section describes the WEC-Sim source code and the code structure. For the purposes of this document, we define the the source code as the MATLAB m-files that read the user input data, perform pre-processing calculations that take place before the Simulink/SimMechanics time-domain simulations are performed.

5.1 Units

All units within WEC-Sim are in the MKS (meters-kilograms-seconds system) and angular measurements are specified in radians unless otherwise specified.

5.2 WEC-Sim Input File

A WEC-Sim input file is required for each run. It MUST be placed inside the case directory for the run and MUST be named `wecSimInputFile.m`. Figure 5.1 shows an example of the input file for a two-body point absorber. The input file contains information needed to run WEC-Sim simulations. Specifically, it serves four primary functions, each of which will be described in the following paragraphs.

5.2.1 Specification of Simulation Parameters

Within the input file, the user specifies simulation parameters, such as simulation duration and time step. As shown in Figure 5.1, simulation parameters are specified within the `simu` variable, which is a member of the `simulationClass`. Users also specify the name of the Simulink/SimMechanics WEC model within the `simu` variable. The simulation parameters that are available for the user to set within the `simulationClass` are described in more detail in Section 5.3.1.

5.2.2 Specification of Body Parameters

WEC-Sim assumes that every WEC devices is composed of rigid bodies that are exposed to wave forces. For each body, users MUST specify body properties within the `body` variable in the input file, including mass, moment of inertia, center of gravity, and the WAMIT files

that describe the hydrodynamic properties (see Figure 5.1). The body parameters that are available for the user to set within the `bodyClass` are described in more detail in Section 5.3.2.

```
% Simulation Data
simu.startTime=0;
simu.endTime=400;
simu.dt = 0.1;
simu.simMechanicsFile = 'RM3.slx';
simu.mode='normal';

simu.explorer='on';

% Wave Information
waves.H = 2.5;
waves.T = 8;
waves.type = 'regular';

% Body Data
body(1) = bodyClass('Float');
body(1).hydroDataType = 'wamit';
body(1).hydroDataLocation = ...
    './wamit/rm3.out';
body(1).mass = 'wamitDisplacement';
body(1).cg = 'wamit';
body(1).momOfInertia = ...
    [20907301 21306090.66 37085481.11];
body(1).geometry = 'geometry/float.stl';

body(2) = bodyClass('Spar_Plate');

body(2).hydroDataType = 'wamit';
body(2).hydroDataLocation = ...
    './wamit/rm3.out';
body(2).mass = 'wamitDisplacement';
body(2).cg = 'wamit';
body(2).momOfInertia = ...
    [94419614.57 94407091.24 28542224.82];
body(2).geometry = 'geometry/plate.stl';

% PTO and Constraint Parameters
constraint(1) = ...
    constraintClass('Constraint1');

pto(1) = ptoClass('PTO1');
pto(1).k=0;
pto(1).c=1200000;
```

```
% Simulation Start Time [s]
% Simulation End Time [s]
% Simulation Delta Time [s]
% Specify Simulink Model File
% Specify Simulation Mode
%   (normal/accelerator
%   /rapid-accelerator)
% Turn SimMechanics Explorer
%   (on/off)

% Wave Height [m]
% Wave Period [s]
% Specify Type of Waves

% Initialize bodyClass for Float
% Specify BEM solver
% Location of WAMIT *.out file

% Mass from WAMIT [kg]
% Cg from WAMIT [m]
% Moment of Inertia [kg-m^2]

% Geometry File

% Initialize bodyClass for
%   Spar/Plate
% Specify BEM solver
% Location of WAMIT *.out file

% Mass from WAMIT [kg]
% Cg from WAMIT [m]

% Moment of Inertia [kg-m^2]
% Geometry File

% Initialize Constraint Class
%   for Constraint1

% Initialize ptoClass for PTO1
% PTO Stiffness Coeff [N/m]
% PTO Damping Coeff [Ns/m]
```

Figure 5.1: Example of a WEC-Sim input file (same to Figure 6.5).

5.2.3 Specification of Wave Parameters

Within the input file, users MUST provide information on the wave condition for the simulation. The user MUST specify the wave condition within the `waves` variable. The options that are available for the user to set within the `waveClass` are discussed in more detail in Section 5.3.3.

5.2.4 Specification of Power Take-Off and Constraint Parameters

PTO and constraint blocks connect WEC bodies to each other (and possibly to the seabed). The properties of the PTO and constraint are defined within `pto` variable and `constraint` variable, respectively. The options that are available for the user to specify are described in more detail in Sections 5.3.4 and 5.3.5 .

5.3 WEC-Sim Code

All data that are needed for a WEC-Sim simulation are contained within `simu`, `body`, `waves`, `pto` and `constraint` variables that are instances of the `simulationClass`, `bodyClass`, `wavesClass`, and `jointClass` objects, respectively. These objects are created in MATLAB classes. The user can interact with these variables within the WEC-Sim input file, `wecSimInputFile.m`, (see Figure 5.1). The remainder of this section describes the data within the WEC-Sim objects and how to interact with the objects to set relevant simulation input parameters. Examples of using WEC-Sim to simulate WEC devices and input files are described in Chapter 6.

5.3.1 `simulationClass`

The `simulationClass` contains the simulation parameters and solver settings needed to execute WEC-Sim. The user can set the relevant simulation properties in the `wecSimInputFile.m`. The user MUST specify the name of the Simulink/SimMechanics WEC model, which can be set by entering the following command in the input file

```
simu.simMechanicsFile='<WEC Model Name>.slx';
```

By doing nothing, users have the option of accepting the default values for all the other simulation parameters. Available simulation properties and the default values for each, shown in Figure 5.2, can be explored further by typing `doc simulationClass` from within the MATLAB Command Window. The users can also specify simulation parameters and solver settings in the input file to overwrite the default values. For example, the end time of a simulation can be set by entering the following command

```
simu.endTime = <user specified end time>
```

Property Summary

CITime	Convolution integral time (default = 60 s)
CIkt	
CTTime	Convolution integral time series (default = dependent)
caseDir	WEC-Sim case directory (default = 'NOT DEFINED')
domainSize	Size of free surface and seabed. This variable is only used for visualization (default = 200 m)
dt	Simulation time step (default = 0.1 s)
endTime	Simulation end time (default = 500 s)
explorer	SimMechanics Explorer 'on' or 'off' (default = 'on')
g	Acceleration due to gravity (default = 9.81 m/s)
hydroDataWamit	Equal to 1 if data from 1 WAMIT file, Equal to 0 if data from more than 1 input file (default = 0)
inputFile	Name of WEC-Sim input file (default = 'wecSimInputFile')
logFile	
maxIt	Total number of simulation time steps (default = dependent) CIkt % Calculate the number of convolution integral timesteps (default = dependent)
mode	'normal','accelerator','rapid-accelerator' (default = 'normal')
numConstraints	Number of constraints in the wec model (default = 'NOT DEFINED')
numFreq	Number of wave frequencies for interpolation (default = 201)
numPtos	Number of power take-off elements in the model (default = 'NOT DEFINED')
numWecBodies	make these dependent variables % Number of hydrodynamic bodies that comprise the WEC device (default = 'NOT DEFINED')
outputDir	Data output directory name
rampT	Ramp time for wave forcing (default = 100 s)
rho	Density of water (default = 1000 kg/m ³)
simMechanicsFile	Simulink/SimMechanics model file (default = 'NOT DEFINED')
solver	PDE solver used by the Simulink/SimMechanics simulation (default = 'ode4')
startTime	Simulation start time (default = 0 s)
time	Simulation time [s] (default = 0 s)
version	WEC-Sim version
zeroVel	Matrix of zeros with a size of 6,obj.CIkt+1 (default = dependent)

Figure 5.2: Data contained within the `simulationClass`.

5.3.2 bodyClass

The `bodyClass` object contains the mass and hydrodynamic properties of each body that comprises the WEC device being simulated. Each body must have a `bodyClass` initiated in the input file. we recommend that these body objects be named `body(<body number>)` as shown in the input file in Figure 5.1. Each body object MUST be initiated by entering the command

```
body(<body number>)=bodyClass('<body name>'),
```

Users can specify the mass and hydrodynamic properties after the body object is initiated for each body. Note that the `hydroDataType`, `hydroDataLocation`, `mass`, `cg`, `momOfInertia` and `geometry` parameters all have to be specified for each body as shown in Figure 5.1.

The users have the option of accepting the default values for the remaining body parameters by doing nothing or specify their own values. The options available within the `bodyClass` are shown in Figure 5.3. For example, the viscous drag can be specified by entering the viscous drag coefficient (nondimensional) and the projected characteristic area (in m^2) in vector format the input file as follows,

```
body(<body number>).cd= [0 0 1.3 0 0 0],
body(<body number>).characteristicArea= [0 0 100 0 0 0],
```

Property Summary

cd	Drag coefficient (format [Cd_x Cd_y Cd_z Cd_rotationX Cd_rotationY Cd_rotationZ], default = [0 0 0 0 0 0])
cg	Center of gravity (format: [x y z])
cgCalcMethod	Method of setting the body cg (options: 'user' or 'wamit', default = 'wamit')
characteristicArea	Characteristic area for viscous drag calculations (format [Area Area Area Area Area Area], default = [0 0 0 0 0 0]).
fixed	Default is 0. If the value is equal to 1, it means the body is fixed to the ground and the mass, MOI and CG will equal to the default value and are meaning less in the calculation.
geom	Structure that defines the geometry for visualization and non-linear buoyancy and excitation force calculations
geometry	Location of the .stl file that defines the geometry of the body (default = 'NOT DEFINED')
hydro	Structure that contains the hydrodynamic data for the body (This structure is currently populated by reading WAMIT data)
hydroDataLocation	Location of the wamit .out file (default = 'NOT DEFINED')
hydroDataType	Code used to generate hydrodynamic coefficients (options: 'wamit', default = 'wamit')
hydroForce	Structure used to calculate hydrodynamic forces acting on the body
initAngularDispAngle	Initial displacement of cog - Angle of rotation - used for decay tests (format: [radians], default = 0)
initAngularDispAxis	Initial displacement of cog - axis of rotation - used for decay tests (format: [x y z], default = [1 0 0])
initLinDisp	Initial displacement of center of gravity - used for decay tests (format: [displacement in m], default = [0 0 0])
mass	Body mass (options: 'wamitDisplacement' or [mass], default = 'wamitDisplacement')
massCalcMethod	Method of calculating the center of gravity (default = dependent)
momOfInertia	Moment of inertia (format: [Ixx Iyy Izz], default = [999 999 999])
mooring	Data structure that contains the mooring stiffness and damping matrices
name	Name of the body used (default = 'NOT DEFINED')
storage	Structure to store simulation data for post processing

Figure 5.3: Data contained within the `bodyClass`.

5.3.3 waveClass

The `waveClass` contains all the information that defines the wave conditions for the time-domain simulation. Specifically, Table 5.1 lists the types of wave environment that is supported by WEC-Sim.

Table 5.1: List of supported wave environments

Option	Additional required inputs	Description
<code>waves.type</code> <code>= 'noWave'</code>	<code>waves.noWaveHydrodynamicCoeffT</code>	Free decay test with constant hydrodynamic coefficients
<code>waves.type</code> <code>= 'noWaveCIC'</code>	None	Free decay test with convolution integral
<code>waves.type</code> <code>= 'regular'</code>	<code>waves.H</code> ; <code>waves.T</code>	Sinusoidal steady-state Reponse Scenario
<code>waves.type</code> <code>= 'regularCIC'</code>	<code>waves.H</code> ; <code>waves.T</code>	Regular waves with convolution integral
<code>waves.type</code> <code>= 'irregular'</code>	<code>waves.H</code> ; <code>waves.T</code> ; <code>waves.spectrumType</code>	Irregular waves with typical wave spectrum
<code>waves.type</code> <code>= 'irregularImport'</code>	<code>waves.spectrumDataFile</code>	Irregular waves with user-defined wave spectrum

- **No waves (`waves.type='noWave'`):** It is the wave type for running simulations without waves and using constant added mass and radiation damping coefficients. Accordingly, the user must still run WAMIT before executing WEC-Sim. In addition, users MUST specify the period from which the hydrodynamic coefficients are selected by setting the `waves.noWaveHydrodynamicCoeffT` variable. This option is typically used for decay tests for comparison with analytical solutions that use given radiation added-mass and damping coefficients.
- **No waves with convolution integral calculation (`waves.type='noWaveCIC'`):** The wave type is the same as `noWave`, except the radiation forces are calculated using the convolution integral and the infinite frequency added mass .
- **Regular waves (`waves.type='regular'`):** It is the wave type for running simulations using regular waves with constant added mass and radiation damping coefficients. Wave period `wave.T` and wave height `wave.H` need to be specified in the input file. Using this option, we assume that the system dynamic response is in sinusoidal steady-state form, where constant added mass and damping coefficients are used and the convolution integral is NOT used to calculate wave radiation forces.
- **Regular waves with convolution integral (`waves.type='regularCIC'`):** The wave type is the same as `waves.type='regular'`, except the radiation forces are calculated using the convolution integral and the infinite frequency added mass.

- **Irregular waves** (`waves.type='irregular'`): It is the wave type for irregular wave simulations using given wave spectrum. Significant wave height `wave.H`, peak period `wave.T` and wave spectrum type (`waves.spectrumType`) need to be specified in the input file. The available wave spectrum options are listed in Table 5.2.

Table 5.2: WEC-Sim wave spectrum options (with `waves.type='irregular'`)

Wave Spectrum Type	Input File Parameter
Pierson–Moskowitz	<code>waves.spectrumType='PM'</code>
Bretschneider	<code>waves.spectrumType='BS'</code>
JONSWAP	<code>waves.spectrumType='JS'</code>

- **Irregular waves with user-defined spectrum** (`waves.type='irregularImport'`): It is the wave type for irregular wave simulations using user-defined wave spectrum. Users need to specify the wave spectrum file name in input file as follows,

`waves.spectrumDataFile='<wave spectrum file>.txt',`

The user-defined wave spectrum must be defined with the wave frequency (Hz) in the first row, and the spectral energy density $\frac{m^2}{Hz}$ in the second row. An example of which is given in the `ndbcBuoyData.txt` file in the applications folder of the WEC-Sim download. This format can be copied directly from NDBC buoy data. For more information on NDBC buoy data measurement descriptions, please refer to NDBC website at <http://www.ndbc.noaa.gov/measdes.shtml>.

Note that, by default, the random wave phase for irregular waves are generated arbitrarily (`waves.randPreDefined=0`). If the user specifies `waves.randPreDefined=1` in the input file, the random phase of the waves will be generated using the `rand` function in MATLAB with a seed value of 1. This gives the user an option to generate the same random wave every time if needed.

Typing `doc waveClass` from the MATLAB Command Window provides more information on the class functionality and the available wave properties are shown and described in Figure 5.4.

5.3.4 constraintClass

The constraint object is used to connect bodies to the Global Reference Frame. The constraint variable should be initiated by entering the following:

`constraint(<constraint number>)=constraintClass('<constraint name>');`

For rotational constraint (i.e., pitch), the user also needs to specify its location of the rotational joint with respect to the global reference frame in the `constraint(<constraint number>).loc` variable.

Property Summary

<u>A</u>	make dependent % [m] Wave amplitude for regular waves or sqrt(wave spectrum vector) for irregular waves
<u>H</u>	[m] Wave height (regular waves) or significant wave height (irregular waves) (default = 1)
<u>I</u>	[s] Wave period (regular waves) or peak period (irregular waves) (default = 8)
<u>bemFreq</u>	make dependent % Number of wave frequencies from WAMIT
<u>noWaveHydrodynamicCoeffT</u>	Period of BEM simulation used to determine hydrodynamic coefficients for simulations with no wave. This option is only used with the 'noWave' wave type.
<u>numFreq</u>	make dependent % Number of interpolated wave frequencies (default = 'NOT DEFINED')
<u>phaseRand</u>	[rad] Random wave phase (only used for irregular waves)
<u>spectraType</u>	Type of wave spectra. Only PM, BS, JS, and Imported spectra are supported.
<u>spectrumDataFile</u>	Data file that contains the spectrum data file
<u>type</u>	Wave type. Options for this varaibale are 'noWave' (no waves), 'regular' (regular waves), 'regularCIC' (regular waves using convolution integral to calculate radiation effects), 'irregular' (irregular waves), 'irregularPRE' (irregular waves with pre defined phase). The default is 'regular'.
<u>typeNum</u>	make dependent % Number to represent different type of waves
<u>w</u>	make dependent % [rad/s] Wave frequency (regular waves) or wave frequency vector (irregular waves)
<u>waterDepth</u>	make dependent % [m] Water depth (from WAMIT)
<u>waveAmpTime</u>	[m] Wave elevation time history

Figure 5.4: Data contained within the waveClass.

5.3.5 ptoClass

The pto object extracts power from body motion with respect to a fixed reference frame or another body. The pto objects can also constrain motion to certain degrees of freedom. The pto variable should be initiated by entering

```
pto(<pto number>)=ptoClass('<pto name>'),
```

For rotational ptos (Local RY), the user also needs to set its location. The users also have the option to specify the damping (`pto(<pto number>).c`) and stiffness (`pto(<pto number>).k`) values to represent the PTO system. Both have a default value of 0, and the users can overwrite the default values in the input file. For example, the users can specify a damping value by entering the following:

```
pto(<pto number>).c=<pto damping value>,
```

Chapter 6: Applications

In this section, we describe how to use WEC-Sim to model two different WECs. The first application models a two-body point absorber WEC and the second application models an OSWEC.

6.1 Reference Model 3 Two-Body Point Absorber

6.1.1 Geometry Definition

As the first application of the WEC-Sim code, we selected the Reference Model 3 (RM3) two-body point absorber design. Although the WEC is free to move in all 6DOF in response to wave motion, power is captured in the relative heave direction. The RM3 device was selected only because the design has already been well characterized both numerically and experimentally as a result of the DOE-funded Reference Model Project, more information on this project available at <http://energy.sandia.gov/rmp>. In addition, the device has relatively simple operating principles and is representative of what WEC industry is currently pursuing. RM3 is a simple two-body point absorber, consisting of a float and a reaction plate. The full-scale dimensions of the RM3 are shown in Figure 6.1, and the mass properties are defined in Table 6.1.

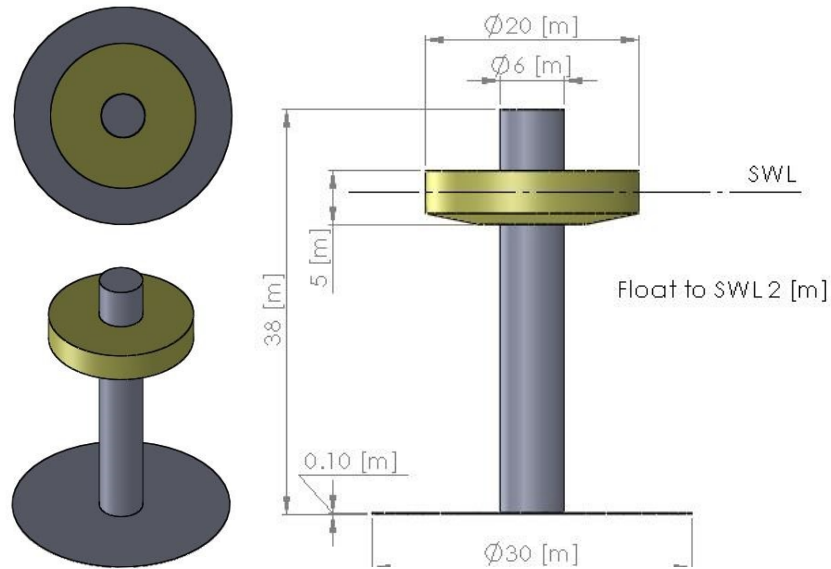


Figure 6.1: RM3 heaving two-body point absorber full-scale dimensions

Table 6.1: RM3 heaving two-body point absorber full-scale mass properties

Float Full Scale Properties				
CG [m]	Mass [tonne]	Moment of Inertia [kg-m ²]		
0	727.01	20907301	0	0
0		0	21306090.7	4304.89323
-0.72105		0	4304.89323	37085481.1
Plate Full Scale Properties				
CG [m]	Mass [tonne]	Moment of Inertia [kg-m ²]		
0	878.30	94419614.6	0	0
0		0	94407091.2	217592.785
-21.285		0	217592.785	28542224.8

6.1.2 Modeling RM3 in WEC-Sim

In this section, we provide a step by step tutorial on how to set up and run the RM3 simulation in WEC-Sim. We have also created a supplemental RM3 WEC-Sim Tutorial Video to demonstrate how to set up and run the RM3 simulation in WEC-Sim. The tutorial video is included in the WEC-Sim code download package under the documentation folder.

As described in Chapter 3, all WEC-Sim models consist of a input file (`wecSimInputFile.m`), and a Simulink model file (`RM3.slx`). To run the WEC-Sim simulation, the user needs to provide results from the WAMIT, frequency-domain BEM solver, to populate the WEC-Sim hydrodynamic coefficients. The WAMIT hydrodynamic results were pregenerated. The WAMIT output file corresponds to the `buoywamit.out` file, contained in the `wamit` subfolder. Note that all the hydrodynamic coefficients MUST be output at the center of gravity. In addition, the user needs to specify the 3-D geometry file in the form of a `<STL file name>.stl` file with the origin of the coordinate system at the center of gravity for the WEC-Sim visualizations. For the RM3 run consisting of a buoy and a spar plate, these files correspond to the `float.stl` and `plate.stl` files, respectively, which are located in the `geometry` subfolder.

RM3 Simulink Model File

The first step to initiate a WEC-Sim simulation is to create the Simulink model file by dragging and dropping blocks from the WEC-Sim library into the `<WEC model name>.slx` file (see Chapter 4 for more details).

Step 1: Place two **Rigid Body** blocks from the WEC-Sim library in the Simulink model file, one for each RM3 rigid body, as shown in Figure 6.2.

Step 2: Double click on the **Rigid Body** block, and rename the instances of the body. The first body should be titled `body(1)`, and the second body should be titled `body(2)`.

Additional properties of these body blocks are defined in the following RM3 MATLAB Input File section (Section 6.1.2).

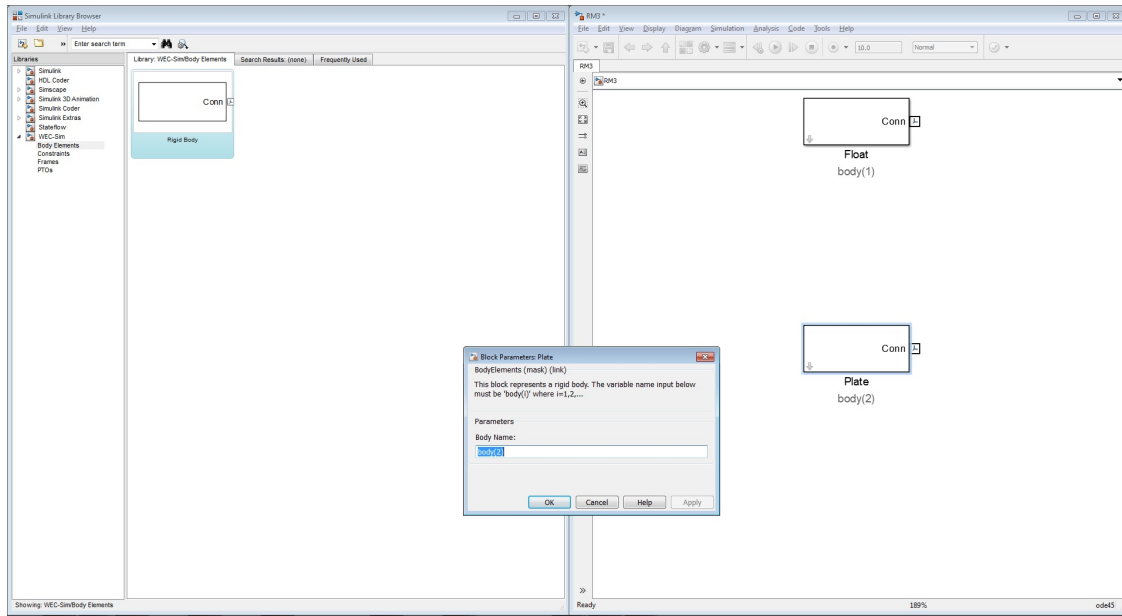


Figure 6.2: Adding two Rigid Body blocks to the RM3 WEC-Sim model

Step 3: Place the **Global Reference Frame** from the WEC-Sim library in the Simulink model file, as shown in Figure 6.3. The global reference frame acts as the seabed to which all other bodies are linked through joints or constraints.

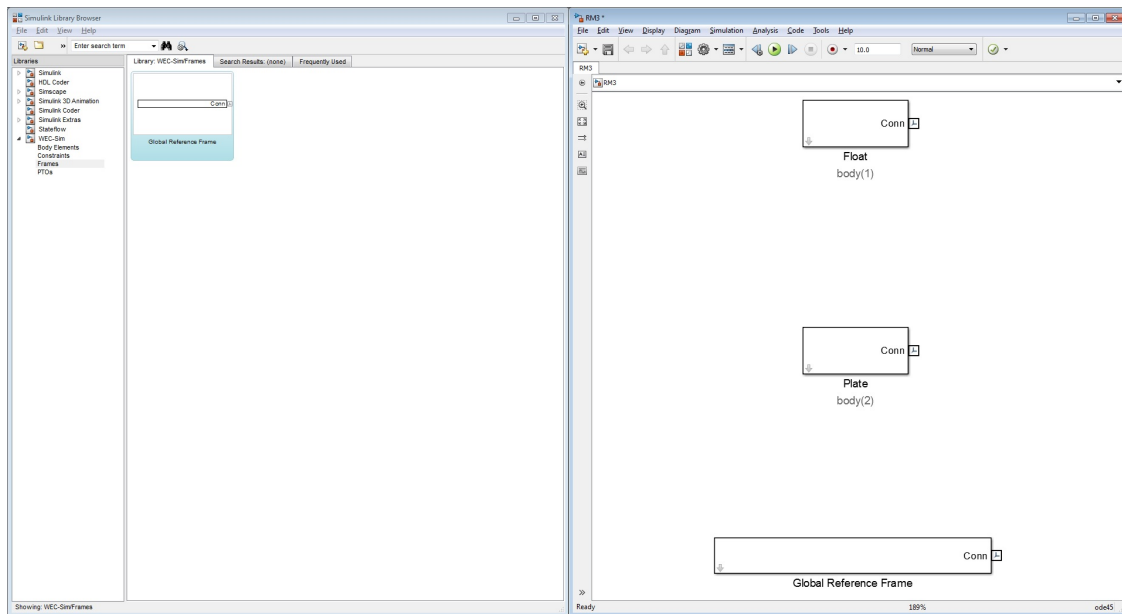


Figure 6.3: Adding the Global Reference Frame block to the RM3 WEC-Sim model

Step 4: Use the **Floating** constraint block to connect the plate to the seabed. This is done because the RM3 is free to move in all 6 DOF relative to the global reference frame. Step 4 and 5 connections are shown in Figure 6.4.

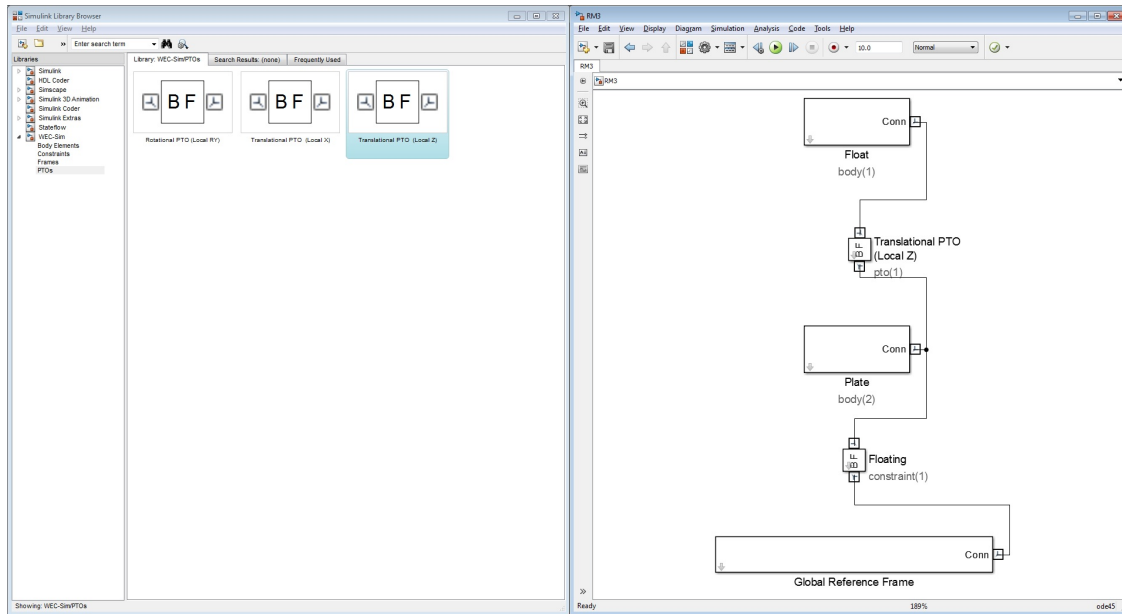


Figure 6.4: Adding the pto and constraint blocks to the RM3 WEC-Sim Simulink model

Step 5: Place a **Translational PTO (Local Z)** PTO block to connect the float to the spar. This is necessary because the float is restricted to heave motion relative to the plate. For the RM3 simulation, the translational PTO block is used to model the WEC's PTO as a linear damper. The parameters are defined in the RM3 MATLAB Input File section (Section 6.1.2).

When setting up a WEC-Sim model, it is important to note the base and follower frames. For example, for the constraint between the plate and the seabed, the seabed should be defined as the base because it is the Global Reference Frame. Similarly, for the PTO between the float and the plate, the plate should be defined as the base.

RM3 MATLAB Input File

In this section, we define the WEC-Sim MATLAB input file for the RM3 model. Each of the lines are commented to explain the purpose of the defined parameters. For the RM3 model, the user must define the simulation parameters, body properties, PTO, and constraint definitions. The specified input parameters for RM3 are shown in Figure 6.5.

```

% Simulation Data
simu.startTime=0;
simu.endTime=400;
simu.dt = 0.1;
simu.simMechanicsFile = 'RM3.slx';
simu.mode='normal';

simu.explorer='on';

% Wave Information
waves.H = 2.5;
waves.T = 8;
waves.type = 'regular';

% Body Data
body(1) = bodyClass('Float');
body(1).hydroDataType = 'wamit';
body(1).hydroDataLocation = ...
    './wamit/rm3.out';
body(1).mass = 'wamitDisplacement';
body(1).cg = 'wamit';
body(1).momOfInertia = ...
    [20907301 21306090.66 37085481.11];
body(1).geometry = 'geometry/float.stl';

body(2) = bodyClass('Spar.Plate');

body(2).hydroDataType = 'wamit';
body(2).hydroDataLocation = ...
    './wamit/rm3.out';
body(2).mass = 'wamitDisplacement';
body(2).cg = 'wamit';
body(2).momOfInertia = ...
    [94419614.57 94407091.24 28542224.82];
body(2).geometry = 'geometry/plate.stl';

% PTO and Constraint Parameters
constraint(1) = ...
    constraintClass('Constraint1');

pto(1) = ptoClass('PTO1');
pto(1).k=0;
pto(1).c=1200000;

% Simulation Start Time [s]
% Simulation End Time [s]
% Simulation Delta-Time [s]
% Specify Simulink Model File
% Specify Simulation Mode
% (normal/accelerator
% /rapid-accelerator)
% Turn SimMechanics Explorer
% (on/off)

% Wave Height [m]
% Wave Period [s]
% Specify Type of Waves

% Initialize bodyClass for Float
% Specify BEM solver
% Location of WAMIT *.out file

% Mass from WAMIT [kg]
% Cg from WAMIT [m]
% Moment of Inertia [kg-m^2]

% Geometry File

% Initialize bodyClass for
% Spar/Plate
% Specify BEM solver
% Location of WAMIT *.out file

% Mass from WAMIT [kg]
% Cg from WAMIT [m]

% Moment of Inertia [kg-m^2]
% Geometry File

% Initialize Constraint Class
% for Constraint1

% Initialize ptoClass for PTO1
% PTO Stiffness Coeff [N/m]
% PTO Damping Coeff [Ns/m]

```

Figure 6.5: WEC-Sim input file for the RM3 point absorber).

RM3 WEC-Sim Model

Once the WEC-Sim Simulink model is set up and the RM3 properties are defined in the MATLAB input file, the user can then run the RM3 model in WEC-Sim. Figure 6.6 shows

the final RM3 Simulink model and the WEC-Sim GUI during the simulation. For more information on using WEC-Sim to model the RM3 device, refer to [12] and [13].

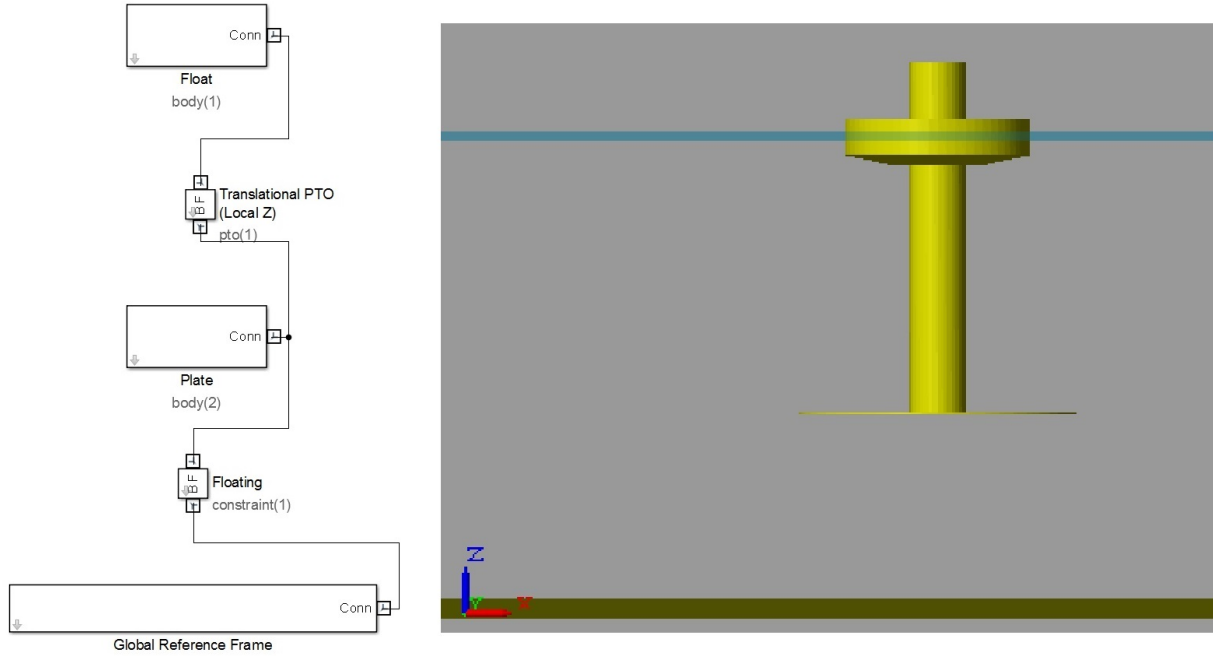


Figure 6.6: RM3 modeled in WEC-Sim (left-hand side) and with the GUI (right hand side)

6.2 Oscillating Surge-Pitch Device

6.2.1 Geometry Definition

As the second application of the WEC-Sim code, the oscillating surge WEC (OSWEC) device. We selected the OSWEC because its design is fundamentally different from the RM3. This is critical because WECs span an extensive design space, and it is important to model devices in WEC-Sim that operate under different principles. The OSWEC is fixed to the ground and has a flap that is connected through a hinge to the base that restricts the flap to pitch about the hinge. The full-scale dimensions of the OSWEC are shown in Figure 6.7, and the mass properties are defined in Table 6.2.

6.2.2 Modeling OSWEC in WEC-Sim

In this section, we provide a step by step tutorial on how to set up and run the OSWEC simulation in WEC-Sim.

As described in Chapter 3, all WEC-Sim models consist of a input file (`wecSimInputFile.m`), and a Simulink model file (`OSWEC.slx`). The WAMIT hydrodynamic results were also pre-

generated. The WAMIT output file corresponds to the `oswec.out` file, contained in the `wamit` subfolder. In addition, the user needs to specify the 3-D geometry file in the form of a `<WEC model name>.stl` file about the center of gravity for the WEC-Sim visualizations. For the OSWEC run consisting of a flap and a base, these files correspond to the `flap.stl` and `base.stl` files, respectively, which are located in the `geometry` subfolder.

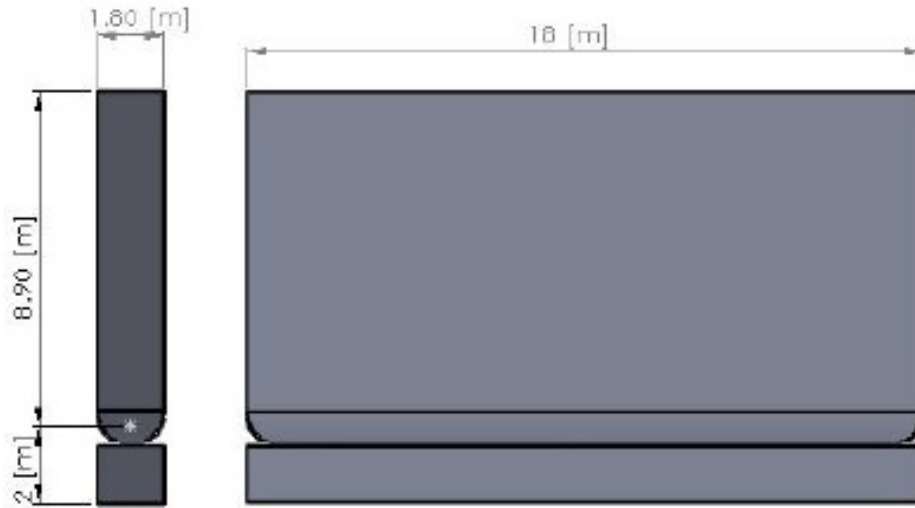


Figure 6.7: OSWEC pitching device full-scale dimensions

Table 6.2: OSWEC pitching device full-scale mass properties

Flap Full Scale Properties		
CG [m]	Mass [kg]	Pitch Moment of Inertia [kg-m ²]
0	127,000	1,850,000
0		
-3.9		

OSWEC Simulink Model File

The first step to set up a WEC-Sim simulation is to populate the Simulink model file by dragging and dropping blocks from the WEC-Sim library into the `<WEC model name>.slx` file, see Chapter 4.

Step 1: Place two **Rigid Body** blocks from the WEC-Sim library in the Simulink model file, one for each OSWEC rigid body, as shown in Figure 6.8.

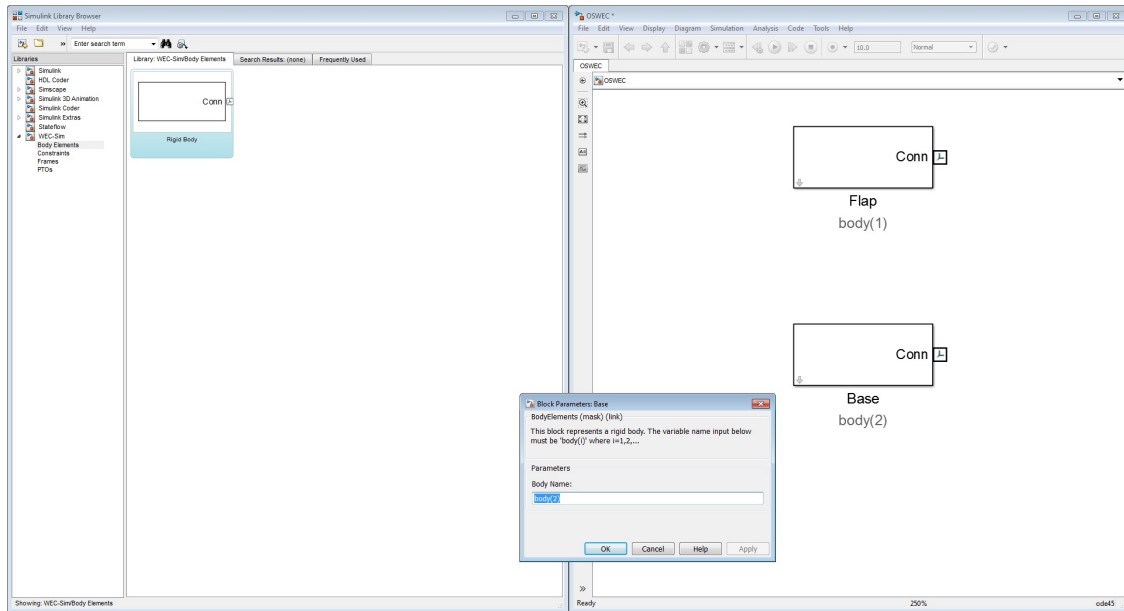


Figure 6.8: Adding two Rigid body blocks to the OSWEC WEC-Sim model

Step 2: Double click on the body block, and rename the instances of the body. The first body should be titled body(1), and the second body should be titled body(2). Additional properties of these body blocks are defined in the OSWEC MATLAB Input File section (Section 6.2.2).

Step 3: Place the **Global Reference** block from the WEC-Sim library in the Simulink model file, as shown in Figure 6.9. The global reference frame acts as the base to which all other bodies are linked through joints or constraints.

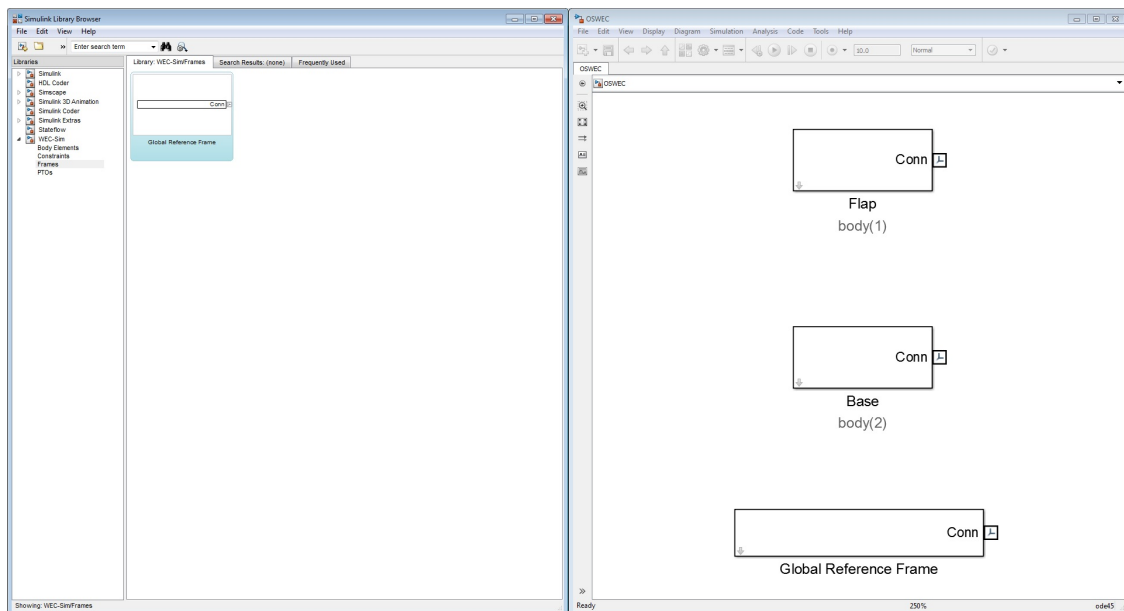


Figure 6.9: Adding a Global Reference Frame block to the OSWEC WEC-Sim model

Step 4: Place a **fixed** constraint block to connect the base to the seafloor. This is done because the OSWEC base is fixed relative to the global reference frame. Step 4 and 5 connections are shown in Figure 6.10.

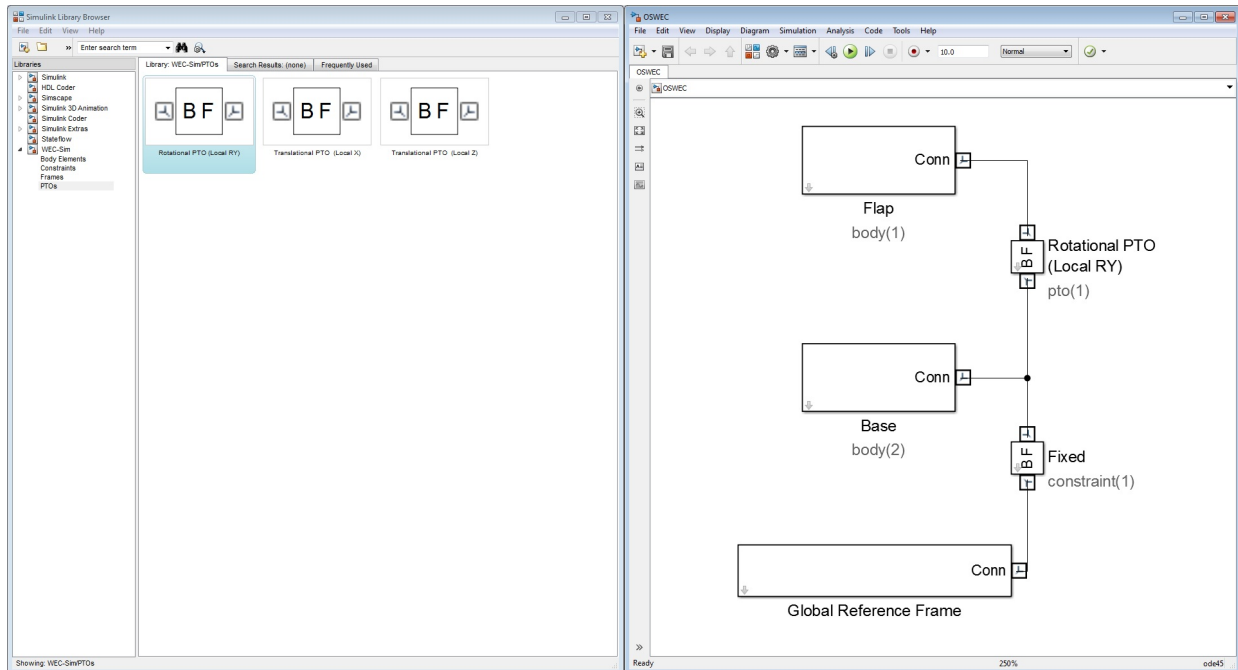


Figure 6.10: Adding pto and constraint blocks to the OSWEC WEC-Sim Simulink model

Step 5: Place a **rotational** PTO block to connect the base to the flap. This is done because the flap is restricted to pitch motion relative to the base. For the OSWEC simulation, the rotational PTO is used to model the WEC's PTO as a linear rotary damper. The input parameters are defined in the OSWEC MATLAB Input File section (Section 6.2.2).

When setting up a WEC-Sim model, it is important to note the base and follower frames. For example, for the constraint between the base and the seabed, the seabed should be defined as the base because it is the Global Reference Frame.

OSWEC MATLAB Input File

In this section, the WEC-Sim MATLAB input file, `wecSimInputFile.m`, for the OSWEC model is defined. Each of the lines are commented to explain the purpose of the defined parameters. For the OSWEC model, the user must define the simulation parameters, body properties, PTO, and constraint definitions. Each of the specified parameters for OSWEC are defined below. The specified input parameters for RM3 are shown in Figure 6.11.

```

% Simulation Data
simu.startTime=0;
simu.endTime=400;
simu.dt = 0.1;
simu.simMechanicsFile = 'OSWEC.slx';
simu.mode='normal';

simu.explorer='on';

% Wave Information
waves.H = 2.5;
waves.T = 8;
waves.type = 'regular';

% Body Data
body(1) = bodyClass('Flap');
body(1).hydroDataType = 'wamit';
body(1).hydroDataLocation = ...
    'wamit/oswec.out';
body(1).mass = 127000;
body(1).cg = 'wamit';
body(1).momOfInertia = ...
    [1.85e6 1.85e6 1.85e6];
body(1).geometry = 'geometry/flap.stl';

body(2) = bodyClass('Base');
body(2).hydroDataType = 'wamit';
body(2).hydroDataLocation = ...
    'wamit/oswec.out';
body(2).geometry = 'geometry/base.stl';
body(2).fixed = 1;

% PTO and Constraint Parameters
constraint(1)=...
    constraintClass('Constraint1');

pto(1)=ptoClass('PTO1');
pto(1).k=0;
pto(1).c=0;
pto(1).loc=[0 0 -8.9];

% Simulation Start Time [s]
% Simulation End Time [s]
% Simulation Delta-Time [s]
% Specify Simulink Model File
% Specify Simulation Mode
% (normal/accelerator
% /rapid-accelerator)
% Turn SimMechanics Explorer
% (on/off)

% Wave Height [m]
% Wave Period [s]
% Specify Type of Waves

% Initialize bodyClass for Flap
% Specify BEM solver
% Location of WAMIT *.out file

% User-Defined mass [kg]
% Cg from WAMIT [m]

% Moment of Inertia [kg-m^2]
% Geometry File

% Initialize bodyClass for Base
% Specify BEM solver
% Location of WAMIT *.out file

% Geometry File
% Creates Fixed Body

% Initialize ConstraintClass
% for Constraint1

% Initialize ptoClass for PTO1
% PTO Stiffness Coeff [Nm/rad]
% PTO Damping Coeff [Nsm/rad]
% PTO Global Location [m]

```

Figure 6.11: WEC-Sim input file for the OSWEC).

OSWEC WEC-Sim Model

Once the WEC-Sim Simulink model is set up and the OSWEC properties are defined in the MATLAB input file, the user can then run the OSWEC model in WEC-Sim. Figure 6.12 shows the final OSWEC Simulink model and the WEC-Sim GUI showing the OSWEC during the simulation. For more information on using WEC-Sim to model the OSWEC

device, refer to [13] and [14].

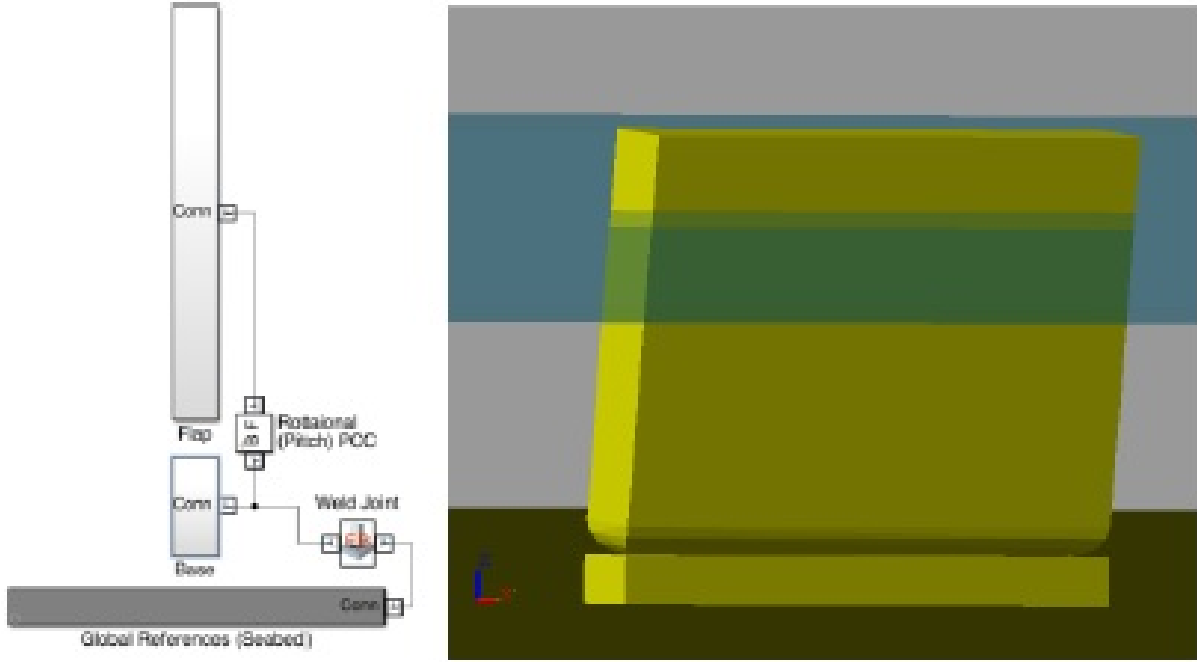


Figure 6.12: OSWEC modeled in WEC-Sim (left-hand side) and with the GUI (right-hand side)

6.3 WEC-Sim Test Cases

Provided within the applications folder in the WEC-Sim directory are the two applications of the WEC-Sim code. The first application uses the WEC-Sim code to model the the RM3 WEC, and the second application uses the code to model the OSWEC device. Should the WEC-Sim user make changes to the WEC-Sim code, scripts are provided that compare the previous version of WEC-Sim to the latest run, for debugging purposes. These test cases can be used by running the `RunTestCase.m` script. This script imports the `*.mat` file that contains results from the original WEC-Sim run, and then plots a comparison of the original run to the latest run of WEC-Sim.

6.3.1 RM3 Test Case

For the RM3, the test case is set up for direct comparison using the following model parameters:

Wave Environment

Wave Type = Regular Waves (Sinusoidal Steady-State)

Wave Height H (m) = 2.5

Wave Period T (sec) = 8

WEC-Sim Simulation Settings

Time Marching Solver = Fourth-Order Runge-Kutta Formula

Start Time (sec) = 0

End Time (sec) = 400

Time Step Size (sec) = 0.1

Ramp Function Time (sec) = 100

List of PTO(s)

Number of PTOs = 1

PTO Stiffness (N/m) = 0

PTO Damping (Ns/m) = 1.2E+06

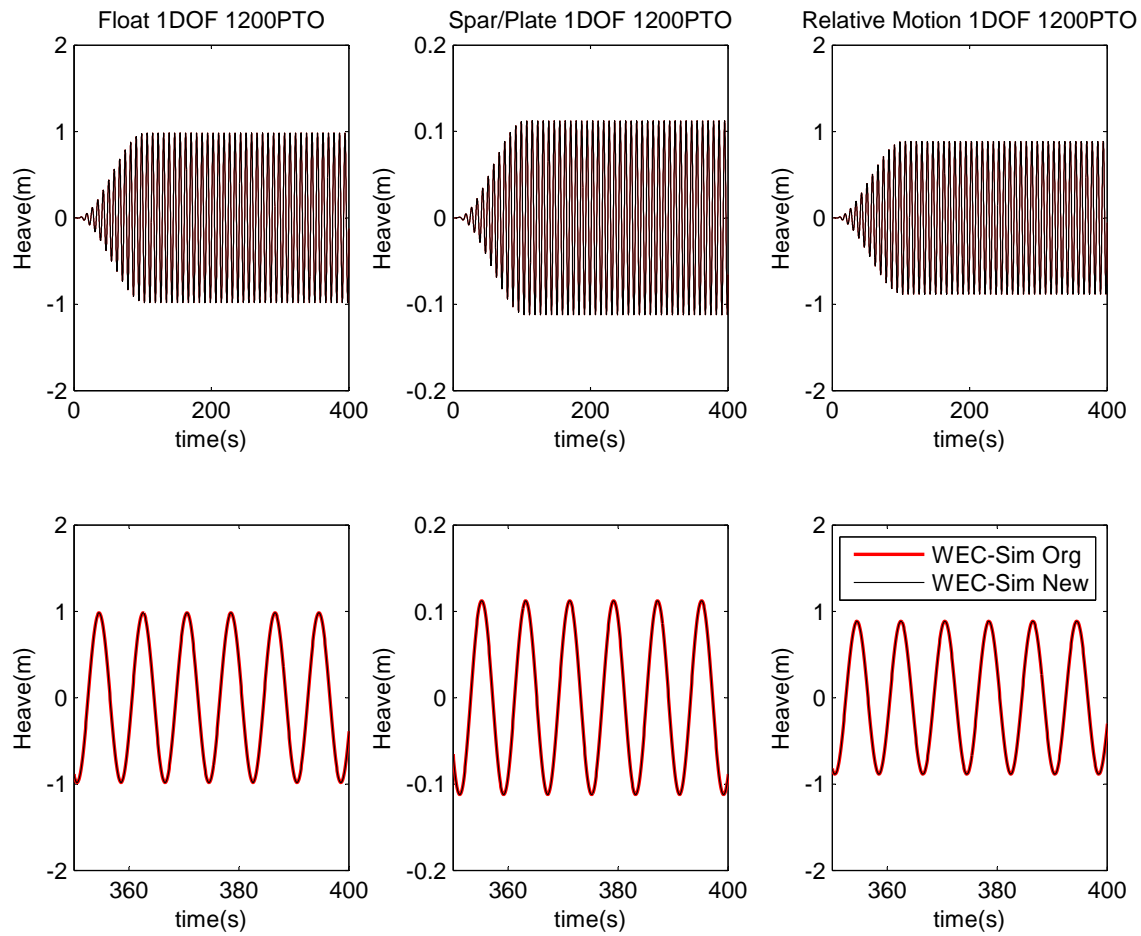


Figure 6.13: WEC-Sim test plot for RM3: Full simulation (top); last six periods (bottom)

6.3.2 OSWEC Test Case

For the OSWEC, the test case is set up for direct comparison using the following model parameters:

Wave Environment

Wave Type = Regular Waves (Sinusoidal Steady-State)

Wave Height H (m) = 2.5

Wave Period T (sec) = 8

WEC-Sim Simulation Settings

Time Marching Solver = Fourth-Order Runge-Kutta Formula

Start Time (sec) = 0

End Time (sec) = 400

Time Step Size (sec) = 0.1

Ramp Function Time (sec) = 100

List of PTO(s)

Number of PTOs = 1

PTO Stiffness (Nm/rad) = 0

PTO Damping (Nsm/rad) = 0

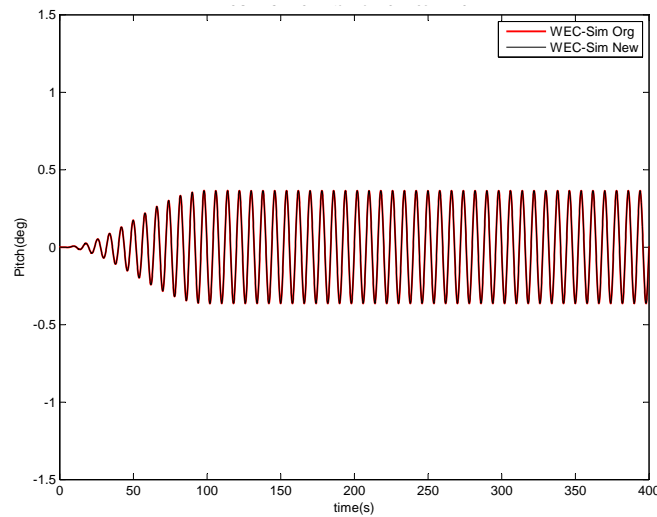


Figure 6.14: WEC-Sim test plot for OSWEC full simulation

Bibliography

- [1] Y. Li and Y.-H. Yu, “A Synthesis of Numerical Methods for Modeling Wave Energy Converter-Point Absorbers,” *Renewable and Sustainable Energy Reviews*, vol. 16, no. 6, pp. 4352–4364, 2012.
- [2] A. Babarit, J. Hals, M. Muliawan, A. Kurniawan, T. Moan, and J. Krokstad, “Numerical Benchmarking Study of a Selection of Wave Energy Converters,” *Renewable Energy*, vol. 41, pp. 44–63, 2012.
- [3] C. Lee and J. Newman, “WAMIT User Manual,” Chestnut Hill, MA, USA, 2006.
- [4] “ANSYS Aqwa.” [Online]. Available: <http://www.ansys.com/Products/Other+Products/ANSYS+AQWA>
- [5] “Nemoh a Open source BEM.” [Online]. Available: <http://openore.org/tag/nemoh/>
- [6] J. D. Nolte and R. C. Ertekin, “Wave power calculations for a wave energy conversion device connected to a drogue,” *Journal of Renewable and Sustainable Energy*, vol. 6, no. 1, 2014.
- [7] W. Cummins, “The Impulse Response Function and Ship Motions,” David Taylor Model Basin-DTNSRDC, Tech. Rep., 1962.
- [8] W. J. Pierson and L. A. Moskowitz, “Proposed Spectral Form for Fully Developed Wind Seas Based on the Similarity Theory of S. A. Kitaigorodskii,” *Geophysical Research*, vol. 69, pp. 5181–5190, 1964.
- [9] K. Hasselman, T.P. Barnett, E. Bouws, H. Carlson, D.E. Cartwright, K. Enke, J.A. Ewing, H. Gienapp, D.E. Hasselmann, P. Kruseman, A. Meerburg, P. Mller, D.J. Olbers, K. Richter, W. Sell, and H. Walden, “Measurements of wind-wave growth and swell decay during the Joint North Sea Wave Project (JONSWAP),” German Hydrographic Institute, Tech. Rep. 12, 1973.
- [10] “MATLAB.” [Online]. Available: <http://http://www.mathworks.com/>
- [11] MATLAB, “SimMechanics Link User s Guide R2014a,” 2014.
- [12] Kelley Ruehl, Carlos Michelen, Samuel Kanner, Michael Lawson, and Y. Yu, “Preliminary verification and validation of WEC-Sim, and open-source wave energy converter design tool,” in *Proceedings of OMAE 2014*, San Francisco, CA, 2014.
- [13] Y. Yu, Michael Lawson, Kelley Ruehl, and Carlos Michelen, “Development and demonstration of the WEC-Sim wave energy converter simulation tool,” in *Proceedings of the 2nd Marine Energy Technology Symposium*, Seattle, WA, USA, 2014.

- [14] Y. Yu, Ye Li, Kathleen Hallett, and Chad Hotinsky, “Design and analysis for a floating oscillating surge wave energy converter,” in *Proceedings of OMAE 2014*, San Francisco, CA, 2014.