WEC-Sim Code Structure 訳

佐賀大学
今井康貴(imaiy@cc.saga-u.ac.jp)

Code Structure
コード構造
This section provides a description of the WEC-Sim source code and its structure.
この節では WEC-Sim のソースコードとその構造を説明する
For more information about the WEC-Sim code structure, refer to the webinar on WEC-Sim Code Structure.
WEC-Sim のコード構造の詳細は WEC-Sim Code Structure ウェビナーを参照

WEC-Sim Source Code
WEC-Sim ソースコード
The WEC-Sim source code consists of a series of MATLAB *.m objects (defined in WEC-Sim as classes) and Simulink *.slx library blocks which are executed by the wecSim.m script.
WEC-Sim ソースコードは、(クラス WEC-SIM で定義された) MATLAB *.m オブジェクトと Simulink *.slx ライブラリブロックから構成される．これは wecSim.m スクリプトで実行される
Executing wecSim.m parses the user input data, performs preprocessing calculations in each of the classes, selects and initializes variant subsystems in the Simulink model, and runs the time-domain simulations in WEC-Sim.
wecSim.m はユーザ入力データを解析し，各クラスで前処理演算を行い，Simulink モデルの有効なサブシステムを選択・初期化し，WEC-SIM の時間領域シミュレーションを実行する
The WEC-Sim source code files are contained within a source directory referred to as $SOURCE.
WEC-Sim のソースコードファイルは、ソースディレクトリ$SOURCE にある

File Type	File name	Directory
WEC-Sim Executable Script	wecSim.m	$SOURCE
WEC-Sim MATLAB Objects	<object>Class.m	$SOURCE/objects
WEC-Sim Simulink Library	<block>_Lib.slx	$SOURCE/lib

WEC-Sim Objects
WEC-Sim オブジェクト
All information required to run WEC-Sim simulations is contained within the simu, waves, body(i), pto(i), constraint(i), and mooring(i) objects
(instances of the simulationClass, waveClass, bodyClass, constraintClass, ptoClass, and mooringClass).
WEC-Sim シミュレーションの実行に必要なすべての情報が simu, wave, body(i), pto(i), constraint(i), mooting(i) オブジェクトに収容される
(simulationClass, waveClass, bodyClass, constraintClass, ptoClass, mooringClass のインスタンス)
The user can interact with these classes within the WEC-Sim input file (wecSimInputFile.m).
ユーザーは WEC-Sim 入力ファイル(wecSimInputFile.m)でこれらのクラスを設定できる
The remainder of this section describes the role of the WEC-Sim objects, and how to interact with the WEC-Sim objects to define input properties.
以降は、WEC-Sim オブジェクトの役割を説明し、入力プロパティを定義するために WEC-Sim オブジェクトと対話する方法を説明する
There are two ways to look at the available properties and methods within a class.
クラス内で利用可能なプロパティとメソッドを見るには二つの方法がある
The first is to type doc <className> in Matlab Command Window,
and the second is to open the class definitions located in the //WEC-Sim/source/objects directory by typing open <className> in MATLAB Command Window.
第一は MATLAB コマンドウィンドウで doc <クラス名> を入力する方法、
第二は //WEC-Sim/source/objects ディレクトリにあるクラス定義を見る方法である．

これは MATLAB コマンドウィンドウで open <クラス名> を入力する.
The latter provides more information since it also defines the different fields in a structure.
後者は構造内の異なるフィールドの定義を提供する

Simulation Class
シミュレーションクラス
The simulation class file, simulationClass.m, is located in the //WEC-Sim/source/objects directory.
シミュレーションのクラスファイル simulationClass.m は //WEC-Sim/source/objects にある
The simulation class contains the simulation parameters and solver settings necessary to execute the WEC-Sim code.
シミュレーションクラスは WEC-Sim コードの実行に必要なシミュレーションパラメタとソルバ設定を含む
Within the wecSimInputFile.m, users must initialize the simulation class (simulationClass) and specify the name of the WEC-Sim (*.slx) model file by including the following lines:
ユーザーは wecSimInputFile.m 内でシミュレーションクラス(simulationClass)を初期化し, WEC-Sim モデルファイル(*.slx)名を指定する必要がある. 以下の行を含める
simu=simulationClass();
simu.simMechanicsFile='<WEC Model Name>.slx'

Users may specify other simulation class properties using the simu object in the wecSimInputFile.m, such as: simulation start time (simu.startTime), end time (simu.endTime), ramp time (simu.rampTime) and time step (simu.dt).
ユーザーは, wecSimInputFile.m 中で simu オブジェクトを用いることで, シミュレーションクラス特性を指定する. 例えばシミュレーション開始時間(simu.startTime), 終了時間(simu.endTime), ランプ時間(simu.rampTime), 時間ステップ(simu.dt).
All simulation class properties are specified as variables within the simu object as members of the simulationClass.
すべてのシミュレーションクラスのプロパティは、simu オブジェクト内の変数として、simulationClass のメンバーとして指定される
The WEC-Sim code has default values defined for the simulation class properties.
WEC-Sim のコードは、シミュレーションクラスのプロパティに定義されたデフォルト値を持つ
These default values can be overwritten by the user, for example, the end time of a simulation can be set by entering the following command: simu.endTime = <user specified end time>.
これらのデフォルト値はユーザが上書できる。例えば、シミュレーションの終了時刻は次のコマンドで上書きできる
simu.endTime = <ユーザ指定終了時間>
Available simulation properties, default values, and functions can be found by typing doc simulationClass in the MATLAB command window, or by opening the simulationClass.m file in //WEC-Sim/source/objects directory by typing open simulationClass in MATLAB Command Window.
利用可能なシミュレーションプロパティ, デフォルト値, 機能は, MATLAB コマンドウィンドウで doc simulationClass を入力することで、または MATLAB コマンドウィンドウで open simulationClass を入力し //WEC-Sim/source/objects/simulationClass.m にある
Refer to the following section for more information about WEC-Sim Simulation Features.
WEC-Sim シミュレーション機能の詳細は次節を参照

Wave Class
Wave クラス
The wave class file, waveClass.m, is located in the //WEC-Sim/source/objects directory.
wave クラスファイル waveClass.m は //WEC-Sim/source/objects ディレクトリにある
The wave class contains all wave information necessary to define the incident wave condition for the WEC-Sim time-domain simulation.

wave クラスは、WEC-Sim 時間領域シミュレーションの入射波の条件定義に必要な波情報を含む
Within the wecSimInputFile.m, users must initialize the wave class (waveClass) and specify the wave type by including the following lines:
wecSimInputFile.m 内で、ユーザは波クラス(waveClass)を初期化する必要があり、次のように波タイプを指定する
waves = waveClass('type');
Users must specify additional wave class properties using the waves object depending on which wave type is selected, as shown in the table below.
選択した波タイプに応じて，ユーザは追加の波クラスのプロパティを次表のように指定すること
A more detailed description of the available wave types is given in the following sections.
利用可能な波型のより詳細な説明は以下の通り

Wave Type	Required Properties
noWave	waves.T
noWaveCIC	N/A なし
regular	waves.H, waves.T
regularCIC	waves.H, waves.T
irregular	waves.H, waves.T, waves.spectrumType
spectrumImport	waves.spectrumDataFile
etaImport	waves.etaDataFile

Available wave class properties, default values, and functions can be found by typing doc waveClass in the MATLAB command window, or by opening the waveClass.m file in //WEC-Sim/source/objects directory by typing open wavenClass in Matlab Command Window.
利用可能な波クラスのプロパティ、デフォルト値、関数は，MATLAB コマンドウィンドウで doc waveClass を入力すると表示される．または MATLAB コマンドウィンドウで open wavenClass と入力し //WEC-Sim/source/objects/waveClass.m を見る．

noWave
The noWave case is for running WEC-Sim simulations with no waves and constant added mass and radiation damping coefficients.
noWave ケースは入射波無し，付加質量および放射線減衰係数が定数の WEC-Sim シミュレーション用である．
The noWave case is typically used to run decay tests.
noWave ケースは、一般的に自由動揺(decay)テストに使用される
Users must still provide hydro coefficients from a BEM solve before executing WEC-Sim and specify the period (wave.T) from which the hydrodynamic coefficients are selected.
ユーザーは WEC-Sim を実行する前に BEM ソルバによる流体力係数を用意する必要がある．
また，選択した流体力係数の波周期(wave.T)を指定する必要がある
The noWave case is defined by including the following in the input file:
noWave ケースは入力ファイルに次行を定義する
waves = waveClass('noWave');
waves.T = <user specified wave period>;

noWaveCIC
The noWaveCIC case is the same as the noWave case described above, but with the addition of the convolution integral calculation.
noWaveCIC ケースは、上記 noWave と同じであるが、畳み込み積分演算を追加した
The only differences is that the radiation forces are calculated using the convolution integral and the infinite frequency added mass.
唯一の違いは、ラディエーション力が畳み込み積分と無限周波数付加質量で計算される
The noWaveCIC case is defined by including the following in the input file:
noWaveCIC は入力ファイルに次行を定義する

waves = waveClass('noWaveCIC');

regular
The regular wave case is for running simulations with regular waves and constant added mass and radiation damping coefficients.
regular wave ケースは、規則入射波と一定の付加質量および放射線減衰係数を用いるシミュレーション用である
Using this option, WEC-Sim assumes that the system dynamic response is in sinusoidal steady-state form, where constant added mass and damping coefficients are used (instead of the convolution integral) to calculate wave radiation forces.
このオプションで、WEC-Sim はシステムの動的応答が正弦定常状態であることを前提とする．ラディエーション力の計算のため，(畳み込み積分の代わりに)一定の付加質量と減衰係数を使用する
Wave period (wave.T) and wave height (wave.H) must be specified in the input file.
波周期(wave.T)と波高(wave.H)を入力ファイルで指定すること
The regular case is defined by including the following in the input file:
regular case は、入力ファイルに次行を定義する
waves = waveClass('regular');
waves.T = <user specified wave period>;

regularCIC
The regularCIC is the same as regular wave case described above, but with the addition of the convolution integral calculation.
regularCIC は上記 wave ケースと同様だが，畳み込み積分演算を追加した
The only difference is that the radiation forces are calculated using the convolution integral and the infinite frequency added mass.
唯一の違いは、ラディエーション力が畳み込み積分と無限周波数付加質量を用いて計算されることである
Wave period (wave.T) and wave height (wave.H) must be specified in the input file.
入力ファイルに波周期(wave.T)と波高(wave.H)を指定すること
The regularCIC case is defined by including the following in the input file:
regularCIC case は、入力ファイルに次行を定義する
waves = waveClass('regularCIC');
waves.T = <user specified wave period>;
waves.H = <user specified wave height>;

irregular
The irregular wave case is the wave type for irregular wave simulations using a Pierson Moskowitz (PM), Bretschneider (BS), or JONSWAP (JS) wave spectrum.
irregular wave case は、不規則波シミュレーションの wave type である．ピアソン-モスコウィッツ(PM)、ブレトシュナイダー(BS)、JONSWAP(JS)波スペクトルを使用できる
Significant wave height (wave.H), peak period (wave.T), and wave spectrum type (waves.spectrumtype) must be specified in the input file.
入力ファイルで有義波高(wave.H)、ピーク周期(wave.T)、波スペクトルタイプ(waves.spectrumtype)を指定する
The available wave spectra and their corresponding waves.spectrumType are listed below:
利用可能な波スペクトルとそれに対応する waves.spectrumType は以下の通り
Wave Spectrum , spectrumType
Pierson Moskowitz , PM
Bretschneider , BS
JONSWAP , JS
The irregular case is defined by including the following in the input file:

irregular case は、入力ファイル内に次行を定義する
waves = waveClass('irregular');
waves.T = <user specified wave period>;
waves.H = <user specified wave height>;
waves.spectrumType = '<user specified spectrum>';
Users have the option of defining gamma for the JONSWAP spectrum by specifying waves.gamma = <user specified gamma>;.
オプション gamma が JONSWAP スペクトルを規定する
waves.gamma = <user specified gamma> で指定する
If gamma is not defined, the default value of gamma equal to 3 is used.
gamma が定義されていない場合、デフォルト値 3 が使用される
Refer to the following section for more information about WEC-Sim's irregular Wave Features.
WEC-Sim の irregular Wave の詳細は次節を参照すること

spectrumImport
The spectrumImport case is the wave type for irregular wave simulations using an imported wave spectrum (ex: from buoy data).
spectrumImport case は不規則波のシミュレーションの wave case である．ブイデータ等の波スペクトルを読み込む
The user-defined wave spectrum must be defined with the wave frequency (Hz) in the first row and the spectral energy density (m^2/Hz) in the second row.
ユーザ定義の波スペクトルは、1 行に wave frequency (Hz)，2 行目に spectral energy density (m^2/Hz)を記述する．
An example of this is given in the ndbcBuoyData.txt file in the tutorials directory folder of the WEC-Sim source code.
WEC-Sim のソースコードのチュートリアルディレクトリの ndbcBuoyData.txt ファイルに例が記載される
This format can be copied directly from NDBC buoy data.
この形式は NDBC ブイデータから直接コピーできる
For more information on NDBC buoy data measurement descriptions, refer to the NDBC website.
NDBC ブイの詳細は NDBC の Web サイトを参照
The spectrumImport case is defined by including the following in the input file:
spectrumImport case は入力ファイルに次行を定義する
waves = waveClass('spectrumImport');
waves.spectrumDataFile='<wave spectrum file>.txt';

etaImport
The etaImport case is the wave type for wave simulations using user-defined time-series (ex: from experiments).
etaImport case は，実験などユーザ定義時系列を用いた波シミュレーションである
The etaImport case is defined by including the following in the input file:
etaImport ケースは、入力ファイルに次行を定義する
waves = waveClass('etaImport');
waves.etaDataFile ='<eta file>.txt';
Refer to the following section for more information about WEC-Sim Wave Features.
WEC-Sim Wave 関数の詳細は次節を参照すること

Body Class
Body クラス
The body class file, bodyClass.m, is located in the //WEC-Sim/source/objects directory.
Body クラスファイル bodyClass.m は //WEC-Sim/source/objects にある
The body class contains the mass and hydrodynamic properties of each body that comprises the

WEC being simulated.
Body クラスはシミュレートされる WEC 各物体の質量と流体力特性を含む
Within the wecSimInputFile.m, users must initialize each iteration of the body class (bodyClass), and specify the location of the hydrodynamic data file (*.h5) and geometry file (*.stl) for each body.
wecSimInputFile.m 内で、各反復における Body クラス(bodyClass)を初期化し、各物体の流体力データファイル(*.h5)と形状ファイル(*.STL)の場所を指定する
The body class is defined by including the following lines in the WEC-Sim input file, where # is the body number '<bem_data>.h5' is the name of the h5 file containing the BEM results:
body クラスは WEC-Sim 入力ファイルに次行を記述して定義される．# は物体番号，<bem_data>.h5 は BEM 結果の H5 ファイルである
body(<#>)=bodyClass('<bem_data>.h5')
body(<#>).geometryFile = '<geom>.stl';
Users may specify other body class properties using the body object for each body in the wecSimInputFile.m.
ユーザは wecSimInputFile.m で各物体用の Body オブジェクトを用い，他の Body クラスのプロパティを指定する
WEC-Sim assumes that every WEC is composed of rigid bodies exposed to wave forcing.
WEC-Sim は，WEC は波力を受ける剛体で構成されることを前提とする
Body class properties include mass (body(#).mass) and moment of inertia (body(#).momOfInertia).
Body クラスのプロパティは質量(body(#).mass) と慣性モーメント(body(#).momOfInertia)を含む
For example, viscous drag can be specified by entering the viscous drag coefficient and the characteristic area in vector format the WEC-Sim input file as follows:
例えば，粘性抗力は，粘性抵抗係数、特徴領域をベクトル形式で WEC-Sim 入力ファイルに記述する
body(<#>).viscDrag.cd= [0 0 1.3 0 0 0]
body(<#>).viscDrag.characteristicArea= [0 0 100 0 0 0]
Available body properties, default values, and functions can be found by typing doc bodyClass in the MATLAB command window, or opening the bodyClass.m file in //WEC-Sim/source/objects directory by typing open bodyClass in Matlab Command Window.
利用可能な物体プロパティ、デフォルト値、機能は、MATLAB コマンドウィンドウで doc bodyClass を入力、あるいは MATLAB コマンドウィンドウで open bodyClass を入力し //WEC-Sim/source/objects/bodyClass.m にある．
Refer to the following section for more information about WEC-Sim Body Features.
WEC-Sim Body 機能の詳細は、次節を参照すること

Constraint Class
Constraint クラス
The constraint class file, constraintClass.m, is located in the //WEC-Sim/source/objects directory.
Constraint クラスファイル constraintClass.m は //WEC-Sim/source/objects にある
WEC-Sim constraint blocks connect WEC bodies to on one another (and possibly to the seabed) by constraining DOFs.
WEC-Sim 制約ブロックは、DOF(運動自由度)を制約することによって、WEC 本体同士を相互に（場合によっては海底まで）接続する
The properties of the constraint class (constraintClass) are defined in the constraint object.
拘束クラス(constraintClass)の設定値は constraint オブジェクトに定義される
Within the wecSimInputFile.m, users must initialize each iteration the constraint class (constraintClass) and specify the constraint name, by including the following lines:
wecSimInputFile.m 内で、ユーザは拘束クラス(constraintClass)を各反復で初期化する必要があり、次に示す行で拘束名を指定する
constraint(<#>)=constraintClass('<constraint name>');
For rotational constraint (ex: pitch), the user also needs to specify the location of the rotational joint

with respect to the global reference frame in the constraint(<#>).loc variable.
回転制約（例：ピッチ）の場合、ユーザーはグローバル参照フレームに対して回転ジョイントの位置を constraint(<#>).loc 変数で指定する必要がある.
Available constraint properties, default values, and functions can be found by typing doc constraintClass in the MATLAB command window, or opening the constraintClass.m file in //WEC-Sim/source/objects directory by typing open constraintClass in MATLAB Command Window.
使用可能な拘束プロパティ、デフォルト値、および機能は、MATLAB コマンドウィンドウでドキュメントの constraintClass を入力、または//WEC-Sim/source/objects/constraintClass.m にある.
Refer to the following section for more information about WEC-Sim Constraint Features.
WEC-Sim の拘束の機能の詳細は次節を参照すること

PTO Class
PTO クラス
The pto class file, ptoClass.m, is located in the //WEC-Sim/source/objects directory.
PTO クラスファイル ptoClass.m は //WEC-Sim/source/objects にある.
WEC-Sim Power Take-Off (PTO) blocks connect WEC bodies to one other (and possibly to the seabed) by constraining DOFs and applying linear damping and stiffness.
WEC-Sim パワーテイクオフ（PTO）ブロックは DOF を制限し、線形ダンピングと剛性を適用し WEC ボディを他のボディ（場合によっては海底）に接続する.
The pto class (ptoClass) extracts power from relative body motion with respect to a fixed reference frame or another body.
PTO クラス(ptoClass)は，固定された基準フレームまたは他の物体に対する物体相対運動から動力を取り出す.
The properties of the PTO class (ptoClass) are defined in the pto object.
PTO クラス(ptoClass)の特性は PTO オブジェクトに定義される
Within the wecSimInputFile.m, users must initialize each iteration the pto class (ptoClass) and specify the pto name, by including the following lines:
wecSimInputFile.m 内で、ユーザは PTO クラス(ptoClass)各反復を初期化する必要があり、次行のように PTO 名を指定する
pto(<#>) = ptoClass('<pto name>');
For rotational ptos, the user also needs to specify the location of the rotational joint with respect to the global reference frame in the constraint(<#>).loc variable.
回転 pto の場合、ユーザは constraint(<#>).loc 変数でグローバル基準フレームに対する回転ジョイントの位置を指定する必要がある.
In the PTO class, users can also specify linear damping (pto(<#>).c) and stiffness (pto(<#>).k) values to represent the PTO system (both have a default value of 0).
PTO クラスでは、PTO システムを表す線形減衰(pto(<#>).c)と剛性(pto(<#>).k)を指定できる（両方ともデフォルト値はゼロ)。
Users can overwrite the default values in the input file.
ユーザーは入力ファイル内のデフォルト値を上書きできる
For example, users can specify a damping value by entering the following in the WEC-Sim input file:
例えば WEC-Sim の入力ファイルに以下入力して減衰値を指定する
pto(<#>).c = <pto damping value>;
pto(<#>).k = <pto stiffness value>;
Available pto properties, default values, and functions can be found by typing doc ptoClass in the MATLAB command window, or opening the ptoClass.m file in //WEC-Sim/source/objects directory by typing open ptoClass in MATLAB Command Window.
使用可能な pto プロパティ、デフォルト値、および関数は、MATLAB コマンドウィンドウで doc ptoClass と入力するか、MATLAB コマンドウィンドウで open ptoClass と入力して //WEC-Sim/source/objects/ptoClass.m にある

Refer to the following section for more information about WEC-Sim PTO Features.
WEC-Sim の PTO 機能の詳細は次節を参照すること

Mooring Class
係留クラス
The mooring class file, mooringClass.m`, is located in the //WEC-Sim/source/objects directory.
係留クラス mooringClass.m は //WEC-Sim/source/objects にある。
The properties of the mooring class (mooringClass) are defined in the mooring object.
係留クラス（mooringClass）のプロパティは係留オブジェクトで定義される.
Within the wecSimInputFile.m, users must initialize the mooring class and specify the mooring name, by including the following lines:
mooring(#)= mooringClass('name');
wecSimInputFile.m 内で、次行のように係留を初期化して係留名を指定する必要がある.
The mooring class (mooringClass) allows for different fidelity simulation of mooring systems.
係留クラス（mooringClass）は、係留システムの異なる忠実度シミュレーションを可能にする。
Available mooring properties, default values, and functions can be found by typing doc mooringClass in the MATLAB command window, or opening the mooringClass.m file in //WEC-Sim/source/objects directory by typing open mooringClass in MATLAB Command Window.
使用可能な係留プロパティ、デフォルト値、および関数は、MATLAB コマンドウィンドウで doc mooringClass と入力する. あるいは //WEC-Sim/source/objects/mooringClass.m を見る.
Refer to the following section for more information about WEC-Sim Mooring Features.
WEC-Sim の係留機能の詳細は次節を参照すること

Response Class
応答クラス
The response class is not initialized by the user.
応答クラスはユーザーが初期化できない
Instead, it is created at the end of a WEC-Sim simulation.
その代わり WEC-Sim シミュレーション終了時に作成される
It contains all the output time-series and methods to plot and interact with the results.
これは、すべての出力の時系列およびメソッドを含む
The available parameters are explained in the Output Structure section.
使用可能なパラメタは出力構造(Output Structure)の節で説明される

WEC-Sim Library
WEC-Sim ライブラリ
In addition to the wecSimInputFile.m, a WEC-Sim simulation requires a simulink model (*.slx) that represents the WEC system components and connectivities.
wecSimInputFile.m に加えて、WEC-Sim シミュレーションには、WEC システムのコンポーネントと接続性を表す Simulink モデル（* .slx）が必要である
Similar to how the input file uses the WEC-Sim classes, the Simulink model uses WEC-Sim library blocks.
入力ファイルが WEC-Sim クラスを使用するのと同様に、Simulink モデルは WEC-Sim ライブラリブロックを使用する
There should be a one-to-one between the objects defined in the input file and the blocks used in the Simulink model.
入力ファイルで定義されたオブジェクトと Simulink モデルで使用されるブロックとは 1 対 1 対応の必要がある
The WEC-Sim library is divided into 5 different types of library blocks.
WEC-Sim のライブラリは 5 つのライブラリブロックに分かれる
The user should be able to model their WEC device using the available WEC-Sim blocks (and possibly other Simulink/Simscape blocks).

使用可能な WEC-Sim ブロック（場合によっては他の Simulink/Simscape ブロック）を使用して WEC デバイスをモデル化できる
The image below shows the WEC-Sim block grouping by type.
下の画像は、タイプ別の WEC-Sim ブロックのグループ化を示す

This section describes the five different library types and their general purpose.
この節は 5 つの異なるライブラリの種類とその一般的な目的を説明する
The Body Elements library contains the Rigid Body block used to simulate the different bodies.
Body Elements ライブラリには、さまざまな物体のシミュレートに使用される剛体ブロックが含まれる
The Frames library contains the Global Reference Frame block necessary for every simulation.
Frames ライブラリは、すべてのシミュレーションに必要なグローバル基準フレームブロックを含む
The Constraints library contains blocks that are used to constrain the DOF of the bodies without including any additional forcing or resistance.
Constraints ライブラリは、追加の強制や抵抗を含まずにボディの DOF を制限するブロックを含む
The PTOs library contains blocks used to both simulate a PTO system and restrict the body motion.
PTO ライブラリは、PTO システムをシミュレートし，かつ物体運動を制限するブロックを含む
Both constraints and PTOs can be used to restrict the relative motion between multi-body systems.
制約と PTO の両方を使用して、複数物体間の相対運動を制限できる
The Mooring library contains blocks used to simulate mooring systems.
Mooring ライブラリは係留システムをシミュレートするブロックを含む

Body Elements
物体要素
The Body Elements library shown below contains one block: the Rigid Body block.
以下に示すボディ Elements ライブラリは一つのブロック(Rigid Body)を含む.
It is used to represent rigid bodies.
これは剛体を表すために使用される
At least one instance of this block is required in each model.
各モデルには、このブロックの少なくとも 1 つのインスタンスが必要である
The Rigid Body block is used to represent a rigid body in the simulation.
剛体ブロックは、シミュレーションで剛体を表すために使用される
The user has to name the blocks body(i) (where i=1,2,...).
ユーザーはブロックに名前を付ける必要がある．body（i）(i= 1,2,...)
The mass properties, hydrodynamic data, geometry file, mooring, and other properties are then specified in the input file.
質量プロパティ、流体力学データ、ジオメトリファイル、係留、その他のプロパティは入力ファイルで指定される
Within the body block, the wave radiation, wave excitation, hydrostatic restoring, viscous damping, and mooring forces are calculated.
Body ブロック内では、波ラディエーション、波強制力、静水圧復原、粘性減衰、係留力が計算される

Frames
フレーム
The Frames library contains one block that is necessary in every model.
フレームライブラリには、すべてのモデルで必要なブロックが 1 つ含まれる
The Global Reference Frame block defines the global coordinates, solver configuration, seabed and free surface description, simulation time, and other global settings.
グローバル基準フレームブロックは、グローバル座標、ソルバ構成、海底および自由表面の記述、

シミュレーション時間、その他のグローバル設定を定義する
It can be useful to think of the Global Reference Frame as being the seabed when creating a model.
モデルを作成するときに、グローバル参照フレームを海底と考えることは有益である
Every model requires one instance of the Global Reference Frame block.
すべてのモデルで、グローバル参照フレームブロックのインスタンスが 1 つ必要である
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.
グローバル基準フレームブロックは、simulation クラス変数 simu と wave クラス変数 wave を使用する。これらは入力ファイルで定義する必要がある

Constraints
拘束
The blocks within the Constraints library are used to define the DOF of a specific body.
Constraints ライブラリのブロックは物体の運動自由度を定義する
Constraints blocks define only the DOF, but do not otherwise apply any forcing or resistance to the body motion.
Constraints ブロックは DOF だけを定義し，物体運動に強制的に力や抵抗を適用しない
Each Constraint block has two connections: a base (B) and a follower (F).
Constraint ブロックは Base(B)と follower(F)の 2 つの接続をもつ
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.
Constraints ブロックは、フォロアに接続されているブロックの動きを、ベースに接続されているブロックに対して相対的に制限する
For a single body system, the base would be the Global Reference Frame and the follower is a Rigid Body.
単一の物体系の場合、ベースはグローバル参照フレームになり、フォロワーはリジッドボディになる
A brief description of each constraint block is given below.
Constraint ブロックの簡単な説明を以下に示す
More information can also be found by double clicking on the library block and viewing the Block Parameters box.
ライブラリブロックをダブルクリックし、[ブロックパラメータ]ボックスを表示すると詳細情報が表示される

Constraint Library
Constraint ライブラリ

Block	DOFs	Description
Fixed	0	Rigid connection. Constrains all motion between the base and follower 強固な接続、ベースとフォロワーの間のすべての動きを拘束
Translational	1	Constrains the motion of the follower relative to the base to be translation along the constraint's Z-axis ベースに対するフォロアの相対運動を Z 軸に沿う運動のみに制限する
Rotational	1	Constrains the motion of the follower relative to the base to be rotation about the constraint's Y-axis ベースに対するフォロワーの動きを Y 軸中心回転のみに制限する
Floating (3DOF)	3	Constrains the motion of the follower relative to the base to planar motion with translation along the constraint's X- and Z- and rotation about the Y- axis ベースに対するフォロワーの運動を X 軸,Z 軸に沿う平行移動と Y 軸周りの回転に制限する
Floating (6DOF)	6	Allows for unconstrained motion of the follower relative to the base ベースに対するフォロワの運動は制限なし

PTOs
PTO
The PTOs library is used to simulate linear PTO systems and to restrict relative motion between multiple bodies or between one body and the seabed.
PTO ライブラリは線形 PTO システムをシミュレートし、複数の物体間または 1 物体と海底の相対運動を制限する
The PTO blocks can simulate simple PTO systems by applying a linear stiffness and damping to the connection.
PTO ブロックは簡単な PTO システムをシミュレートする．接続に線形剛性とダンピングを適用する
Similar to the Constraints blocks, the PTO blocks have a base (B) and a follower (F).
Constraints ブロックと同様に、PTO ブロックはベース(B)とフォロワ(F)をもつ
Users must name each PTO block pto(i) (where i=1,2,...) and then define their properties in the input file.
ユーザは各 PTO ブロックに名前を付け PTO(i)(i=1,2,...)，入力ファイル内でプロパティを定義する．
The Translational PTO and Rotational PTO are identical to the Translational and Rotational constraints, but they allow for the application of linear damping and stiffness forces.
並進 PTO と回転 PTO は，並進拘束および回転拘束と同じだが、線形減衰と剛性力が適用できる
Additionally, there are two other variations of the Translational and Rotational PTOs.
また、並進 PTO と回転 PTO には二つの他のバリエーションが存在する
The Actuation Force/Torque PTOs allow the user to define the PTO force/torque at each time-step and provide the position, velocity and acceleration of the PTO at each time-step.
Actuation Force/Torque PTOs は、各時間ステップにおいて PTO 力/トルクを定義でき、各時間ステップで、PTO の位置、速度及び加速度を出力する．
The user can use the response information to calculate the PTO force/torque.
ユーザは PTO 力/トルクを計算する応答情報を使用する
The Actuation Motion PTOs allow the user to define the motion of the PTO.
Actuation Motion PTOs はユーザーが PTO の運動を指定する
These can be useful to simulate forced-oscillation tests.
これらは強制動揺実験シミュレートに有用である

Note
注意
When using the Actuation Force/Torque PTO or Actuation Motion PTO blocks, the loads and displacements are specified in the local (not global) coordinate system.
Actuation Force/Torque PTO または Actuation Motion PTO ブロックを使用する場合，荷重と変位は局所座標系で指定される
This is true for both the sensed (measured) and actuated (commanded) loads and displacements.
これは、検出された(測定)と作動(指令)負荷と変位の両方で正しい

Mooring
係留
The mooring library is used to simulate mooring systems.
mooring ライブラリは係留システムをシミュレートする
The MooringMatrix block applies linear damping and stiffness based on the motion of the follower relative to the base.
MooringMatrix ブロックは，ベースに対するフォロワの相対運動に基づいて、線形減衰及び剛性を適用する
The MoorDyn block uses the compiled MoorDyn executables and a MoorDyn input file to simulate a realistic mooring system.

MoorDyn ブロックはコンパイル済の MoorDyn 実行ファイルと MoorDyn 入力ファイルを使用して現実的な係留システムをシミュレートする
There can only be one MoorDyn block per Simulink model.
Simulink モデルで 1 つの MoorDyn ブロックのみ使用できる
There are no restrictions on the number of MooringMatrix blocks.
MooringMatrix ブロックは数に制限が無い

Simulink/Simscape Blocks
Simulink/Simscape ブロック
In some situations, users want to use Simulink/Simscape blocks that are not included in the WEC-Sim Library to build their WEC model.
WEC-Sim ライブラリに含まれない Simulink/Simscape ブロックを使用して WEC モデルを構築できる

Output Structure
出力構造
After WEC-Sim is done running, there will be a new variable called output in your Matlab workspace.
WEC-Sim 実行後、MATLAB ワークスペースに新しい変数 output が生成される.
The output variable is an instance of the responseClass class.
output 変数は responseClass クラスのインスタンスである
It contains all the relevant time-series results of the simulation.
シミュレーションのすべての時系列結果を含む
The structure of the output variable is shown in the table below.
出力変数の構造を以下の表に示す.
Time series are given as [(# of time-steps) x 6] arrays, where 6 is the degrees of freedom.
時系列は[(# of time-steps) x 6]で与えられる. 6 は運動の自由度である.
In addition to these time-series, the output for each object contains the object's name or type and the time vector.
これら時系列に加え、各オブジェクトの出力は、オブジェクトの名前や種類、時間ベクトルを含む
In addition to the responseClass output variable, the outputs can be written to ASCII files by using simu.outputtxt = 1; in the input file.
responseClass 出力変数に加え、入力ファイルに simu.outputtxt=1 を記述すると，出力が ASCII ファイルに書き込まれる

output
出力

wave	elevation	array: (# of time-steps)x1

bodies(i)	position	array: (# of time-steps) x 6
	velocity	array: (# of time-steps) x 6
	acceleration	array: (# of time-steps) x 6
	forceTotal	array: (# of time-steps) x 6
	forceExcitation	array: (# of time-steps) x 6
	forceRadiationDamping	array: (# of time-steps) x 6
	forceAddedMass	array: (# of time-steps) x 6
	forceRestoring	array: (# of time-steps) x 6
	forceMorrisonAndViscous	array: (# of time-steps) x 6
	forceLinearDamping	array: (# of time-steps) x 6
	cellPressures_time	array: (# nlHydro time-steps) x (# cells)
	cellPressures_hydrostatic	array: (# nlHydro time-steps) x (# cells)
	cellPressures_waveLinear	array: (# nlHydro time-steps) x (# cells)

	cellPressures_waveNonLinear	array: (# nlHydro time-steps) x (# cells)

ptos(i)	position	array: (# of time-steps) x 6
	velocity	array: (# of time-steps) x 6
	acceleration	array: (# of time-steps) x 6
	forceTotal	array: (# of time-steps) x 6
	forceActuation	array: (# of time-steps) x 6
	forceConstraint	array: (# of time-steps) x 6
	forceInternalMechanics	array: (# of time-steps) x 6
	powerInternalMechanics	array: (# of time-steps) x 6

constraints(i)	position	array: (# of time-steps) x 6
	velocity	array: (# of time-steps) x 6
	acceleration	array: (# of time-steps) x 6
	forceConstraint	array: (# of time-steps) x 6

mooring(i)	position	array: (# of time-steps) x 6
	velocity	array: (# of time-steps) x 6
	forceMooring	array: (# of time-steps) x 6

moorDyn	Lines	struct: outputs in the Line#.out file
	Line# (for each line)	struct: outputs in the Line#.out file

ptosim , See PTO-Sim section for details ,

Functions & External Codes
関数と外部コード
While the bulk of the WEC-Sim code consists of the WEC-Sim classes and the WEC-Sim library, the source code also includes supporting functions and external codes.
WEC-Sim の大部分は WEC-Sim のクラスと WEC-Sim のライブラリで構成されているが、ソースコードはサポート関数及び外部のコードを含む
These include third party Matlab functions to read *.h5 and *.stl files, WEC-Sim Matlab functions to write *.h5 files and run WEC-Sim in batch mode, MoorDyn compiled executables, python macros for ParaView visualization, and the PTO-Sim class and library.
*.h5 と *.stl を読み込むサードパーティの Matlab 関数、*.h5 ファイルを書き込んで WEC-Sim をバッチモードで実行する WEC-Sim Matlab 関数, MoorDyn コンパイル済み実行ファイル, ParaView 可視化用の Python マクロ, PTO-Sim クラスとライブラリである
Additionally, BEMIO can be used to create the hydrodynamic *.h5 file required by WEC-Sim.
さらに、BEMIO を使用して、WEC-Sim に必要な流体力 *.h5 ファイルを作成する
MoorDyn is an open source code that must be downloaded separately.
MoorDyn はオープンソースコードであり、別途ダウンロードする必要がある
Users may obtain, modify, and recompile the code as well as desired.
ユーザーは、コードを取得し、変更し、再コンパイルできる.

[END OF FILE]