

WEC-Sim Training Course for users and developers

August 17, 2017

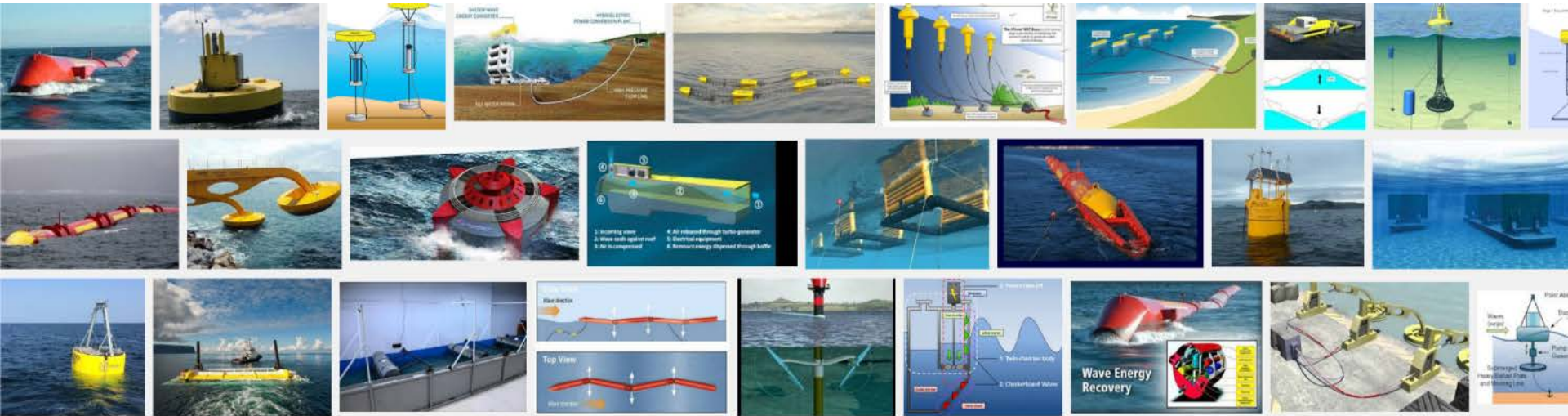
Yi-Hsiang Yu (NREL)

Kelley Ruehl (Sandia)

Course Agenda

Time	Topic	Description
9:00 am	WEC-Sim Overview ~20min	Overview of course topics and WEC-Sim code
9:30 am	Theory & Workflow ~20min	Cummins' equation and WEC-Sim workflow (BEM->BEMIO->WEC-Sim)
10:00 am	Running WEC-Sim ~30min	Description of what happens when you execute WEC-Sim (<i>wecSim.m</i>)
11:00 am	Code Structure Overview ~1hr total	Overview of WEC-Sim's input file (<i>wecSimInputFile.m</i>), classes (<i>*.m</i>) and library blocks (<i>*.s/x</i>)
1:00 pm	Wave Implementation ~30min	Description wave modeling implementation in WEC-Sim, in the classes (<i>*.m</i>) and blocks (<i>*.s/x</i>)
1:30 pm	Body Implementation ~30min	Description body implementation in WEC-Sim, in the classes (<i>*.m</i>) and blocks (<i>*.s/x</i>)
2:00pm	Q&A ~1hr	Open Q&A for attendees to WEC-Sim Lab team

Time (MT)	Topic (~Duration)	Description
1:00 pm	Wave Implementation ~30min	<ul style="list-style-type: none">• Detail of what is done in the wave class and library blocks and how they are linked<ul style="list-style-type: none">• Somewhat line-by-line, give overview of what type of information/calculations are done in the wave class• Ex 1: specify wave type 'regular' 'regularCIC' and 'spectrumImport' to show which method in waveClass are executed and which variant subsystems are turned on



Wave Implementation

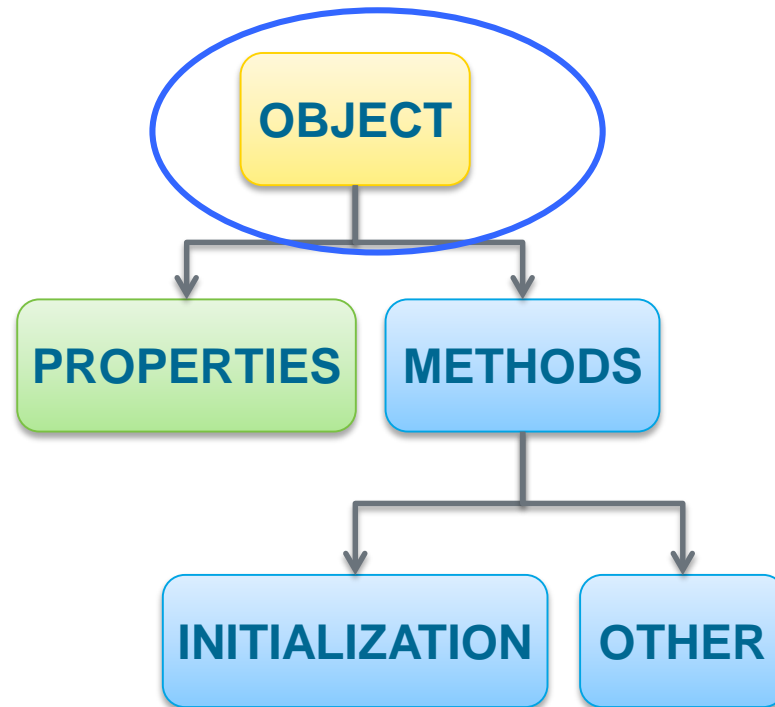
Kelley Ruehl (Sandia)

waveClass.m

The wave class contains all wave information necessary to define the incident wave condition for the WEC-Sim time-domain simulation.

- Wave Class is initialized and defined in the WEC-Sim input file.
- *waveClass.m* → *waves*
- Required Properties:
 - type
 - Each wave 'type' has different required properties

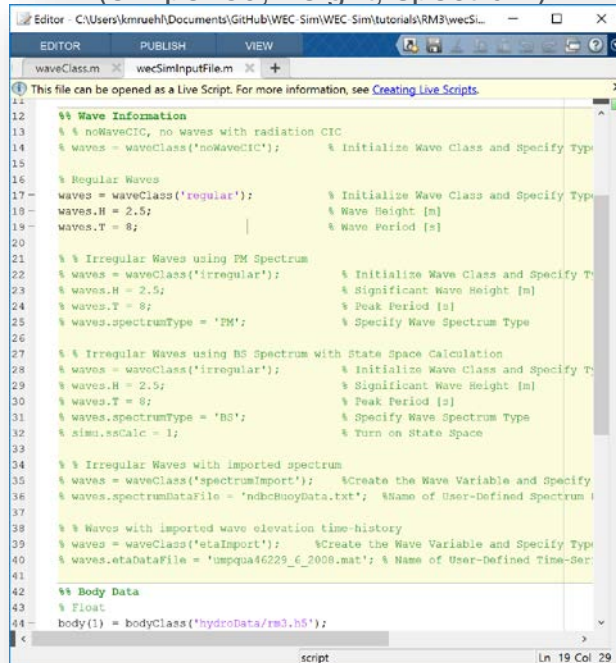
Wave Type	Required Properties
noWave	waves.T
noWaveCIC	N/A
regular	waves.H , waves.T
regularCIC	waves.H , waves.T
irregular	waves.H , waves.T , waves.spectrumType
spectrumImport	waves.spectrumDataFile
etaImport	waves.etaDataFile



Wave Implementation

wecSimInputFile.m

Initializes wave class, user specifies wave type and properties (ex: period, height, spectrum)



```
11 % This file can be opened as a Live Script. For more information, see Creating Live Scripts.
12 % Wave Information
13 % % noWaveCIC, no waves with radiation CIC
14 % waves = waveClass('noWaveCIC'); % Initialize Wave Class and Specify Type
15
16 % Regular Waves
17 waves = waveClass('regular'); % Initialize Wave Class and Specify Type
18 waves.H = 2.5; % Wave Height [m]
19 waves.T = 8; % Wave Period [s]
20
21 % Irregular Waves using PM Spectrum
22 waves = waveClass('irregular'); % Initialize Wave Class and Specify Type
23 waves.H = 2.5; % Significant Wave Height [m]
24 waves.T = 8; % Peak Period [s]
25 waves.spectrumType = 'PM'; % Specify Wave Spectrum Type
26
27 % Irregular Waves using BS Spectrum with State Space Calculation
28 waves = waveClass('irregular'); % Initialize Wave Class and Specify Type
29 waves.H = 2.5; % Significant Wave Height [m]
30 waves.T = 8; % Peak Period [s]
31 waves.spectrumType = 'BS'; % Specify Wave Spectrum Type
32 % simu.ssCalc = 1; % Turn on State Space
33
34 % Irregular Waves with imported spectrum
35 waves = waveClass('spectrumImport'); % Create the Wave Variable and Specify
36 waves.spectrumDataFile = 'ndbcBuoyData.txt'; % Name of User-Defined Spectrum
37
38 % Waves with imported wave elevation time-history
39 waves = waveClass('etaImport'); % Create the Wave Variable and Specify Type
40 waves.etaDataFile = 'umpqua46229_6_2008.mat'; % Name of User-Defined Time-Series
41
42 % Body Data
43 % Float
44 body(1) = bodyClass('hydroData/rm3.h5');
```

waveClass.m

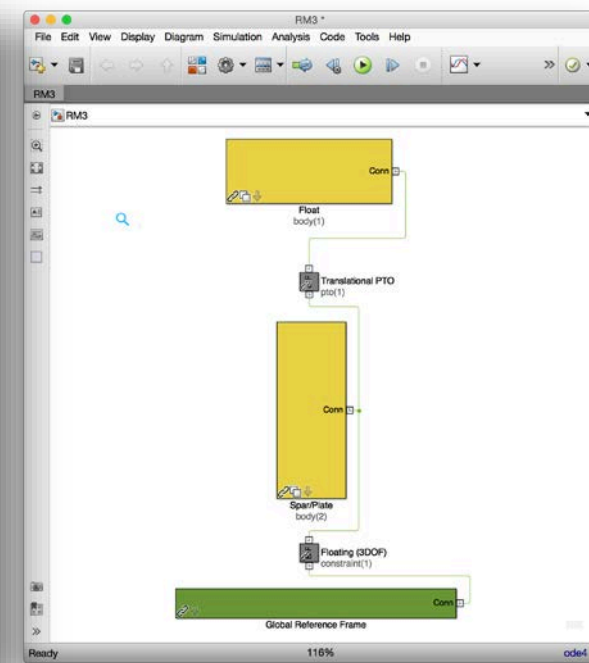
Creates 'waves' object, parses input file, turns on variant subsystems, generates incident wave for WEC-Sim

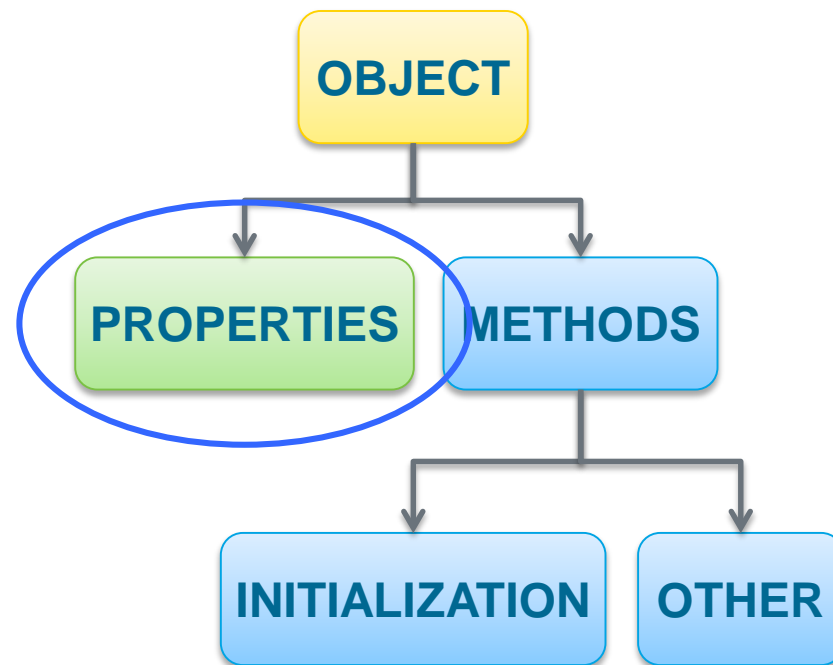


```
16 classdef waveClass < handle
17
18     properties (SetAccess = 'public', GetAccess = 'public') %input file
19         type = 'NOT DEFINED';
20         T = 'NOT DEFINED';
21         H = 'NOT DEFINED';
22         spectrumType = 'NOT DEFINED';
23         randPreDefined = 0;
24         spectrumDataFile = 'NOT DEFINED';
25         etaDataFile = 'NOT DEFINED';
26         numFreq = 1001;
27         freqRange = [];
28         waveBdir = 0;
29         viz = struct(...
30             'numPointsX', 50, ...
31             'numPointsY', 50);
32
33     statisticsDataLoad = [];
34
35 end
36
37 properties (SetAccess = 'private', GetAccess = 'public') %internal
38     typeNum = [];
39     beamFreq = [];
40     waterDepth = [];
41     deepWaterWave = [];
42     waveAmpTime = [];
43     A = [];
44     w = [];
45     phaseRand = 0;
46     dw = 0;
47     k = [];
48
49 end
50
51 methods (Access = 'public')
52     function obj = waveClass(type)
53         % initialization function
54     end
55 end
```

Simulink Model File

Looks the same on the top level, but has different variant subsystems active based on inputs





waveClass Properties

Wave Class Properties

waveClass.m

>>open waveClass

The wave class contains all wave information necessary to define the incident wave condition for the WEC-Sim time-domain simulation.

- Required Properties:
 - type
 - Each wave 'type' has different required properties

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

waveClass Properties

Available at the beginning of waveClass.m file, can see 'public' versus 'private' properties

- Public properties
- Defined in input file

- Private properties
- Calculated by methods in wave class

```
17 classdef waveClassHandle
18     properties (SetAccess = 'public', GetAccess = 'public')%input file
19         type = 'NOT DEFINED'
20         T = 'NOT DEFINED'
21         H = 'NOT DEFINED'
22         spectrumType = 'NOT DEFINED'
23         randPreDefined = 0;
24         spectrumDataFile = 'NOT DEFINED'
25         etaDataFile = 'NOT DEFINED'
26         numFreq = 1001;
27         freqRange = [];
28         waveDir = 0;
29         viz = struct(...
30             'numPointsX', 50, ...
31             'numPointsY', 50)
32         statisticsDataLoad = [];
33     end
34     properties (SetAccess = 'private', GetAccess = 'public')%internal
35         typeNum = []
36         benFreq = []
37         waterDepth = []
38         deepWaterWave = []
39         waveAmplitude = []
40         A = []
41         w = []
42         phaseRand = 0;
43         dw = 0;
44         k = []
45     end
46     methods (Access = 'public')
47         function obj = waveClass(type)
48             % initialization function
49         end
50     end
```


waveClass.m

>> wecSim

>> waves

The wave class contains all wave information necessary to define the incident wave condition for the WEC-Sim time-domain simulation.

- Required Properties:
 - type
 - Each wave ‘type’ has different required properties

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

Wave Object Properties

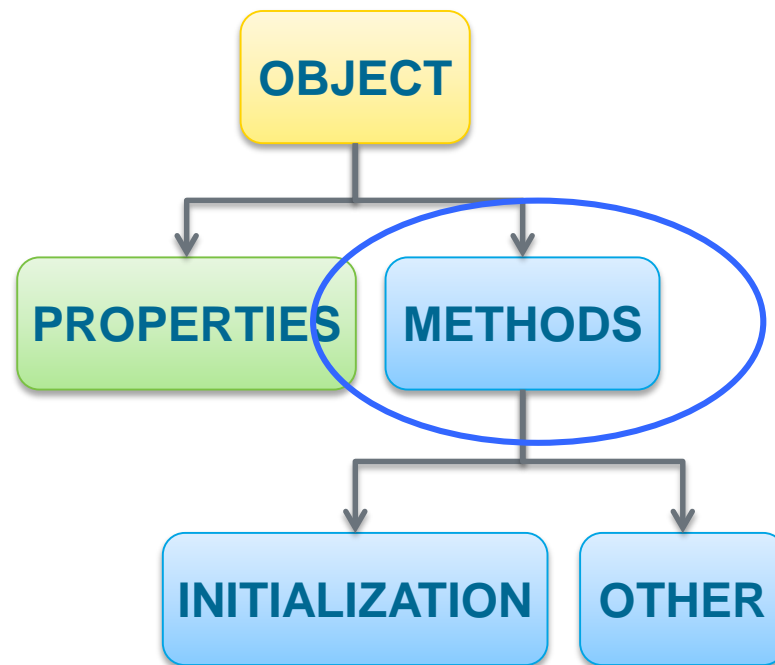
Created by waveClass, available upon completion of WEC-Sim run

```
Command Window
>> waves

waves =

waveClass with properties:

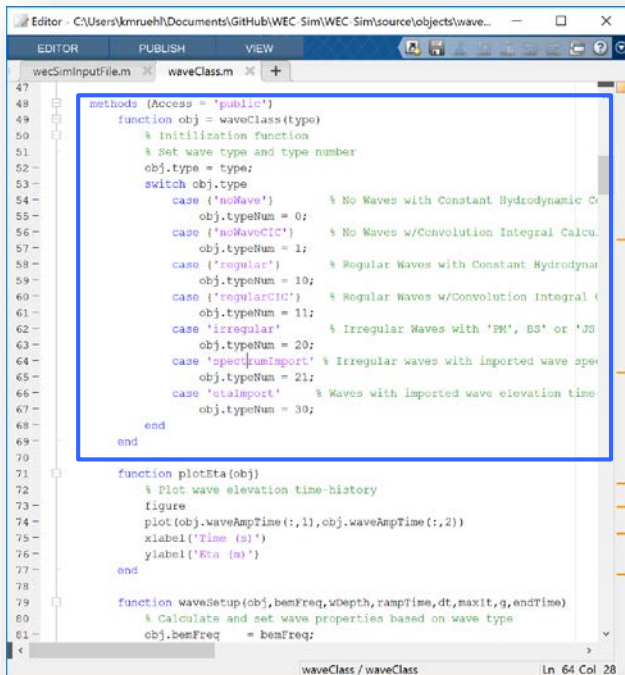
    type: 'regular'
         T: 8
         H: 2.5000
    spectrumType: 'NOT DEFINED'
    randPreDefined: 0
    spectrumDataFile: 'NOT DEFINED'
    etaDataFile: 'NOT DEFINED'
    numFreq: 1001
    freqRange: []
    waveDir: 0
    viz: [1x1 struct]
    statisticsDataLoad: []
    typeNum: 10
    bemFreq: [1x260 double]
    waterDepth: 200
    deepWaterWave: 1
    waveAmpTime: [4001x2 double]
                A: 1.2500
                w: 0.7854
    phaseRand: 0
    dw: 0
    k: 0.0629
```



waveClass Methods

waveClass

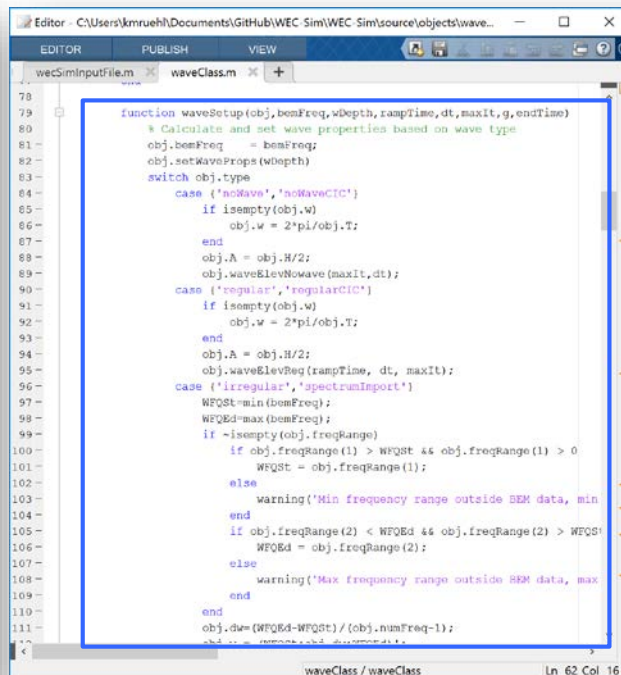
Creates waves object, reads wave 'type' from WEC-Sim input file



```
47
48 methods (Access = 'public')
49 function obj = waveClass(type)
50 % Initialization function
51 % Set wave type and type number
52 obj.type = type;
53 switch obj.type
54 case {'noWave'} % No Waves with Constant Hydrodynamic Co
55 obj.typeNum = 0;
56 case {'noWaveCIC'} % No Waves w/Convolution Integral Calcul
57 obj.typeNum = 1;
58 case {'regular'} % Regular Waves with Constant Hydrodyna
59 obj.typeNum = 10;
60 case {'regularCIC'} % Regular Waves w/Convolution Integral C
61 obj.typeNum = 11;
62 case {'irregular'} % Irregular Waves with 'PM', 'BS' or 'JS
63 obj.typeNum = 20;
64 case {'spectrumImport'} % Irregular waves with imported wave spec
65 obj.typeNum = 21;
66 case {'etaImport'} % Waves with imported wave elevation time
67 obj.typeNum = 30;
68 end
69 end
70
71 function plotEta(obj)
72 % Plot wave elevation time-history
73 figure
74 plot(obj.waveElevTime(:,1),obj.waveElevTime(:,2))
75 xlabel('Time (s)')
76 ylabel('Eta (m)')
77 end
78
79 function waveSetup(obj,bemFreq,wDepth,rampTime,dt,maxIt,g,endTime)
80 % Calculate and set wave properties based on wave type
81 obj.bemFreq = bemFreq;
```

waveSetup

Specifies wave parameters based on wave 'type'



```
78
79 function waveSetup(obj,bemFreq,wDepth,rampTime,dt,maxIt,g,endTime)
80 % Calculate and set wave properties based on wave type
81 obj.bemFreq = bemFreq;
82 obj.setWaveProps(wDepth)
83 switch obj.type
84 case {'noWave','noWaveCIC'}
85 if isempty(obj.w)
86 obj.w = 2*pi/obj.T;
87 end
88 obj.A = obj.R/2;
89 obj.waveElevNoWave(maxIt,dt);
90 case {'regular','regularCIC'}
91 if isempty(obj.w)
92 obj.w = 2*pi/obj.T;
93 end
94 obj.A = obj.R/2;
95 obj.waveElevReg(rampTime, dt, maxIt);
96 case {'irregular','spectrumImport'}
97 WFGST=min(bemFreq);
98 WFGED=max(bemFreq);
99 if ~isempty(obj.freqRange)
100 if obj.freqRange(1) > WFGST && obj.freqRange(1) > 0
101 WFGST = obj.freqRange(1);
102 else
103 warning('Min frequency range outside BEM data, min
104 end
105 if obj.freqRange(2) < WFGED && obj.freqRange(2) > WFGST
106 WFGED = obj.freqRange(2);
107 else
108 warning('Max frequency range outside BEM data, max
109 end
110 end
111 obj.dw=(WFGED-WFGST)/(obj.numFreq-1);
```

Other Methods...

irregWaveSpectrum

- Generates 'PM', 'BS', 'JS' wave spectrum based on user inputs

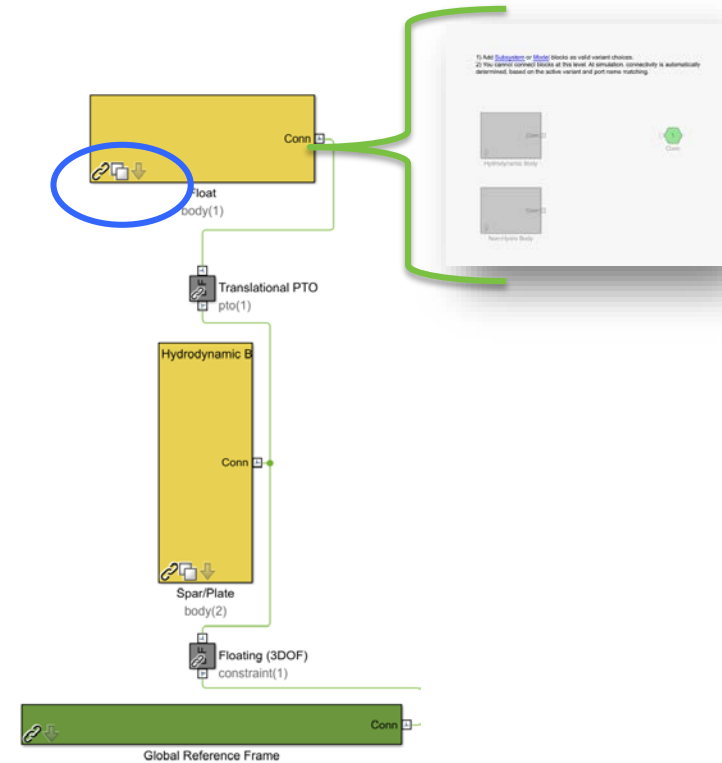
waveElevReg & *waveElevIrreg*

- Generate wave surface elevation time-series with wave ramp

write_paraview_vtp

- Function for generating ParaView Visualization of wave surface elevation

Wave Class Variant Subsystems



Variant Subsystem – Radiation Calc

The collage illustrates the development of a variant subsystem for radiation calculation. It includes several Simulink block diagrams, a MATLAB script for initialization, and a block parameter dialog box.

Simulink Diagrams: The diagrams show the integration of the variant subsystem into a larger model. Key blocks include 'Constant Coefficients', 'SS CI and Constant Damping-Convolution Subsystem', 'Convolution Integral Calculation', and 'State-Space Calculation'. A context menu is shown over one of the diagrams, listing options like 'Explore', 'Open in New Tab', 'Copy', 'Paste', 'Comment Through', 'Comment Out', 'Delete', 'Find Referenced Variables', 'Subsystem & Model Reference', 'Format', 'Rotate & Flip', 'Arrange', 'Variant', 'Mask', 'Library Link', 'Signals & Ports', 'Requirements Traceability', 'Model Advisor', 'Fixed-Point Tool...', 'Block Parameters (Subsystem)', 'Properties...', and 'Help'.

Initialization Script: The script `waveClass.m` initializes the variant subsystem by setting the `radiation_option` based on the `simulink_ssCalc` parameter. It uses `simulink.Variant` objects to set the active choice for the subsystem.

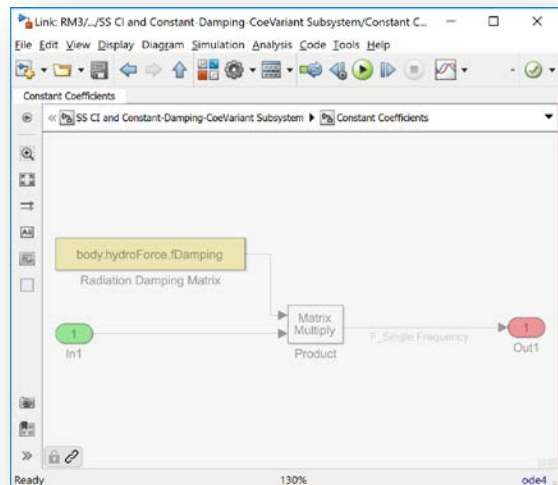
```
164 % Radiation Damping
165 if waves.typeNum==0 || waves.typeNum==10 &'noWave' & 'regular'
166     radiation_option = 1;
167 elseif simulink_ssCalc == 1
168     radiation_option = 3;
169 else
170     radiation_option = 2;
171 end
172
173 sv_constantCoeff=Simulink.Variant('radiation_option=1');
174 sv_convolution=Simulink.Variant('radiation_option=2');
175 sv_stateSpace=Simulink.Variant('radiation_option=3');
176
177 % Wave type
178 typeNum = waves.typeNum;
179 sv_noWave=Simulink.Variant('typeNum=0 & typeNum=10');
180 sv_regularWaves=Simulink.Variant('typeNum=0 & typeNum=20');
181 sv_irregularWaves=Simulink.Variant('typeNum=20 & typeNum=30');
182 sv_udfWaves=Simulink.Variant('typeNum=30');
```

Block Parameters Dialog: The 'Block Parameters: SS CI and Constant Damping Convolution Subsystem' dialog shows the 'Variant Subsystem' tab. It lists the variant choices and their corresponding variant controls and conditions.

Name (read-only)	Variant control	Condition (read-only)
Constant Coefficients	sv_constantCoeff	(N/A)
Convolution Integral Calculation	sv_convolution	(N/A)
State-Space Calculation	sv_stateSpace	(N/A)

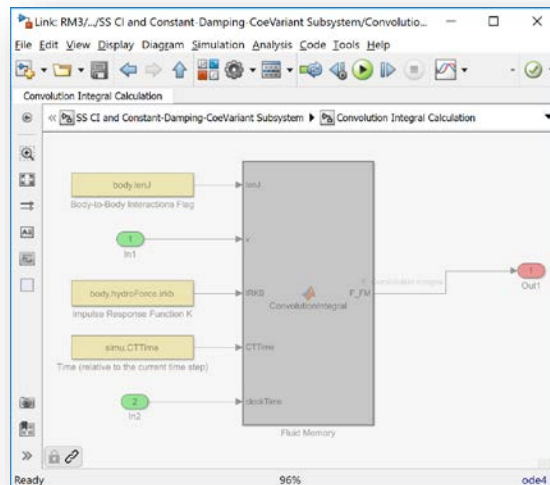
The dialog also includes options to 'Override variant conditions and use the following variant', 'Analyze all choices during update diagram and generate preprocessor conditionals', and 'Propagate conditions outside of variant subsystem'. The 'Variant' field is set to `sv_constantCoeff (Constant Coefficients)`.

Constant Coefficient Block



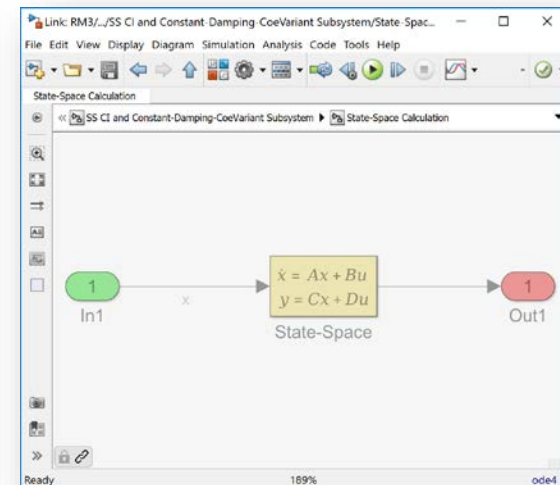
- 'noWave'
- 'regular'

Convolution Integral Block



- 'noWaveCIC'
- 'regularCIC'
- 'irregular'
- 'spectrumImport'
- 'etaImport'

State Space Block



- 'simu.ss'
- (Except for noWave or regular)

Wave Implementation Examples...

Thank you!

All the webinar materials and recordings are available online:

<http://wec-sim.github.io/WEC-Sim/webinars.html>

A screenshot of the WEC-Sim webinars page. The page has a dark sidebar on the left with navigation links: "Getting Started", "Examples", "Theory", "Code Structure", "Advanced Features", "Webinars", "License", "Publications", "Release Notes", and "Contact Us". The main content area is titled "Webinars" and includes a table of upcoming webinars. Below the table, there is a section for "WEC-Sim Webinar #1 - BEMIO & MCR" with a description and a video player showing a boat model.

Date	Topic
April 18, 2017	BEMIO and MCR
May 24, 2017	Nonlinear Hydro, Non-Hydro, and B2B
June 13, 2017	PTO and Control
July 18, 2017	Moorings and Visualization
August 17, 2017	WEC-Sim Training Course

WEC-Sim Webinar #1 - BEMIO & MCR

The presentation and recordings of WEC-Sim Webinar #1 on BEMIO & MCR hosted on April 18, 2017 are available below. Download the presentation by clicking the image below.

WEC-Sim Webinar #1

April 18, 2017
(Copyright © and credits are by EPRI.)