MoorDyn User's Guide 訳

Matthew Hall, Department of Mechanical Engineering, University of Maine
matthew.hall@umit.maine.edu
December 15, 2015

1. Introduction and Acknowledgements
1. 緒言と謝辞
This document is a guide to using MoorDyn, an open-source lumped-mass mooring line model.
この文書はオープンソース集中質量係留ラインモデル MoorDyn の使用ガイドである。
MoorDyn was designed with the mindset of using only the features that are necessary for predicting the dynamics of typical mooring systems and probably isn't suited for modeling cables with appreciable bending and torsional stiffnesses.
MoorDyn は、典型的な係留システムの運動予測に必要な機能のみ使用する考え方で設計された。ケーブルのかなり大きい曲げやねじり剛性のモデリングには適していない。
It can be used as a stand-alone mooring simulator
if fairlead motions are prescribed from a separate data file, or it can be coupled with floating platform models for coupled simulation of a moored floating structure.
フェアリード（導索器）の運動が別のデータファイルで規定されれば、スタンドアロンの係留シミュレータとして使用できる。または、係留浮体構造の連成シミュレーションのため、浮体プラットフォームモデルと結合できるか。
Two versions exist: one generic and one part of FAST v81.
2 つのバージョンが存在する。一般と FAST V81 の一部。
MoorDyn supports arbitrary line interconnections, clump weights and floats, and different line properties.
MoorDyn は、任意のラインの相互接続、クランプの重量や浮体、異なるライン仕様をサポートする。
The model accounts for internal axial stiffness and damping forces, weight and buoyancy forces, hydrodynamic forces from Morison's equation, and vertical spring-damper forces from contact with the seabed.
モデルは、内部軸剛性と減衰力、重力、浮力、モリソン流体力、海底接触による垂直方向ばね減衰力を考慮する。
The formulation supports inclusion of wave kinematics in the hydrodynamic force calculations, but that functionality is currently disabled in the absence of a standardized method for receiving wave kinematics data in coupled simulations.
流体力計算に波運動を含んで定式化できるが、その機能は、無効化している。
現在連成シミュレーションで波運動データを受ける標準方法が存在しないため。
In the FAST v8 version, hydrodynamic loads will eventually be handled externally by coupling with

HydroDyn
FAST V8 では、流体荷重は HydroDyn と連成させ外部処理する
(in the current version, hydrodynamic forces are calculated assuming still water).
（現在、流体力は静水と仮定して計算される)。
The model is still being improved, and I hope other users will contribute to it as they adapt it to their specific needs.
このモデルはまだ改善途上であり、他のユーザー貢献により彼らの特定ニーズに適合させることを願う。
MoorDyn began as a course project in Spring 2014 and emerged as a working mooring model in Fall 2014.
MoorDyn は 2014 年春にコースのプロジェクトとして始まり、2014 年秋に係留モデルとして登場した。
Marco Masciola (ABS) provided advice at many stages of the development process.
マルコ・マシオラ（ABS）は、開発プロセスの多くの段階で助言を提供した。
I then validation MoorDyn against 1:50-scale floating wind turbine test data under the advising of Andrew Goupee (UMaine) [1].
Andrew Goupee（UMaine）の助言の下、1:50 スケール風車実験データで MoorDyn を検証した[1]。
I created a separate FORTRAN implementation of MoorDyn for inclusion in FAST v8, with input from Bonnie Jonkman and Jason Jonkman (NREL).
FAST V8 に含めるため MoorDyn の別 FORTRAN 実装を作成した。Bonnie Jonkman and Jason Jonkman (NREL)のデータを使用した。
I also collaborated with Giacomo Vissio (Politecnico di Torino) to couple MoorDyn to Matlab/Simulink-based wave energy converter models.
MATLAB/Simulink ベース波エネルギー変換器モデルと MoorDyn の結合に Giacomo Vissio (Politecnico di Torino)と協力した。
This was supported by an INORE International Collaboration Incentive Scholarship2.
これは INORE International Collaboration Incentive Scholarship2 によってサポートされた。
Most recently, I helped Senu Sirnivas and Yi-Hsiang Yu in creating a coupling between MoorDyn and WEC-Sim3.
最近 MoorDyn と WEC-SIM の結合を作成する際に Senu Sirnivas and Yi-Hsiang Yu を助けた。
The MoorDyn User's Guide benefited heavily from the reviewing of Jason Jonkman.
MoorDyn ユーザーズガイドは Jason Jonkman のレビューから多くの利益を得た。
My PhD studies are supported by the Natural Sciences and Engineering Research Council of Canada.
私の博士課程の研究は Natural Sciences and Engineering Research Council of Canada によってサポートされた。

2. Model Structure

2.モデル構造

MoorDyn uses a lumped-mass approach to discretize the cable dynamics over the length of the mooring line.

MoorDyn は、係留ライン全長にわたるケーブル運動の離散化に集中質量（lumped-mass）アプローチを使用する。

A cable is broken up into N evenly-sized line segments connecting N+1 node points.

ケーブルは、N 個の均等サイズのライン要素に分割される。N 個の要素は N+1 の節点を連結する。

The indexing starts at the anchor (or lower end), with the anchor node given a value of zero, and the cable segment between nodes 0 and 1 given an index of 1/2.

番号はアンカー（または下端）から始まる。アンカー節点は番号ゼロである。

節点 0 と 1 の間のライン要素は 1/2 とする。

The model uses a right-handed inertial reference frame with the z axis being measured positive up from the water plane, consistent with NREL's FAST simulator.

モデルは右手系慣性座標系を使用する。Z 軸を水面から鉛直上向きにとる。NREL の FAST と同じにした。

Each node's position is defined by a vector r.

各節点の位置はベクトル r で定義される。

Each segment of the cable has identical properties of unstretched length, diameter, density, and Young's modulus.

各ライン要素は未延伸長さ、直径、密度、ヤング率が同一。

Different cables can have different sets of properties,

and cables can be connected together at the ends,

enabling mooring systems with interconnected lines.

異なるケーブルは異なる特性を持てる。ケーブルは端部で互いに接続できる。

相互接続ラインを有する係留システムを可能にする。

The ends of each mooring line are defined by Connection objects, which can be considered a special type of node.

各係留ラインの端部は Connection オブジェクトによって定義される。

これは節点の特殊なタイプと考えられる

Using the same terminology as MAP [2], there are three Connection node types:

MAP [2]と同じ。3 つのタイプの接続節点がある。

Fixed nodes have a certain location and never move. They can be used as anchor points.

Fixed 節点は特定の場所から移動しない。アンカーポイントとして使用する。

Vessel nodes can move under the control of an outside program. They can be used as fairlead connections.

Vessel 節点は外部プログラムの制御下で移動する。フェアリード接続に使用される。

Connect nodes are not fixed in space but rather are moved according to the forces acting on them.

Connent 節点は空間に固定されないが、作用する力に応じて移動する。
They are what can be used to connect two or more mooring lines together.
2 つ以上の係留ライン接続に使用される。
The forces they experience can include the forces from the attached mooring lines
(which Fixed and Vessel node types also experience)
but also constant external forces, buoyancy forces,
inertial and gravitational forces, and hydrodynamic drag and added mass forces.
受ける力は係留ラインからの力だけでなく（Fixed、Vessel 節点タイプも受ける）、外部の一定力、浮力、慣性力および重力、流体抗力、負荷質量力を含む。
Hydrodynamic loads are calculated directly at the node points rather than at the segment centers.
流体荷重は要素中央ではなく節点位置で計算される。
This ensures damping of transverse cable vibrations having a wavelength of twice the cable segment length (which may or may not affect anything).
これは横方向ケーブル振動の減衰を確実にする。ライン要素長さの 2 倍の波長をもつ。
（これは何も影響しない）
To approximate the cable direction at the node points, the cable tangent at each node is assumed to be the average of the tangent directions of the two adjacent cable elements.
節点におけるケーブル方向を近似するため、各節点におけるケーブル接線は、隣接する 2 つのライン要素の接線方向の平均と仮定する。
Aside from this detail, the formulation of the mooring model is fairly standard.
別に詳細から、係留モデルの定式化はかなり標準的である。
Further technical details and some validation results are available in a paper in Ocean Engineering [1].
技術的な詳細と、いくつかの検証結果は、論文に示される[1]。
Some technical details and results related to MoorDyn's capabilities for interconnected lines and mass/buoyancy/drag elements in the mooring system can be found in a recent EWTEC paper [3].
相互接続されたラインと係留システムの質量/浮力/抗力要素の MoorDyn の能力に関連するいくつかの技術的な詳細と結果は、最近の EWTEC 論文に記載される[3]。

3. Model Operation
3.モデル操作
MoorDyn is meant to be used in conjunction with another program that tells it how the fairlead ends of the mooring lines are moving.
MoorDyn は別プログラムと組み合わせて使用するように意図される。別プログラムは係留索端のフェアリードの移動を伝える。
This other program can be as simple as a Matlab script driving MoorDyn with sinusoidal fairlead motions or as complicated as a FAST simulation of a floating platform and wind turbine.
別プログラムは、正弦的なフェアリード運動で MoorDyn を駆動する MATLAB スクリプトのよ

うな単純な場合があれば、浮動プラットフォームと風力タービンの FAST シミュレーションのような複雑な場合もある。
Two versions of MoorDyn exist.
MoorDyn に 2 つのバージョンが存在する。
The 'C' version, written in C++, can be coupled with a variety of codes.
C ++で書かれた「C」バージョンは、様々なコードと連成できる。
The 'F' version, written in FORTRAN, is a module contained in FAST v8.
FORTRAN で記述された「F」は、FAST V8 に含まれるモジュールである。
The underlying model is similar in both cases, just the implementation is different.
基礎となるモデルは同じであり、単に実装が異なる。
One important difference between the two is that MoorDyn C currently couples about the platform reference point;
両者の重要な違いは、現在 MoorDyn C はプラットフォーム基準点で連成することである。
platform motions and mooring reaction forces/moments are communicated with respect to a single point and the platform is assumed rigid.
プラットフォームの運動と係留反力/モーメントは単一点に対して伝達され、プラットフォームは剛体と仮定する。
MoorDyn F, however, couples about the individual fairleads; platform motions and mooring reaction forces are communicated separately for each fairlead, allowing the possibility of flexible/multi-body platforms.
しかし、MoorDyn F は個々のフェアリードで連成する。
各フェアリードにおいてプラットフォームの運動及び係留反力は個別に伝達される。
これは変形/複数プラットフォームを可能にする
Regardless of the version, the basic operation of MoorDyn is the same.
バージョンに関係なく、MoorDyn の基本的な操作は同じである。
During initialization, MoorDyn reads the input file describing the mooring system,
constructs the mooring system data structures, determines the initial fairlead positions based on the initial platform position specified by the calling program,
and then determines the initial equilibrium state of the mooring system.
初期化に、MoorDyn は係留システムを記述した入力ファイルを読み、
係留システムデータ構造を構築し、呼び出しプログラムによって指定された初期プラットフォームの位置に基づいて、初期フェアリード位置を決定する。
次に係留システムの初期平衡状態を判定する。
Determination of the initial state happens in two steps.
初期状態は 2 つのステップで決定される。
In the first step, a quasi-static model is used to determine the locations of the nodes along each mooring line.
第 1 ステップでは、準静的モデルが使用され、各係留ラインに沿った節点位置を決定する。

The line ends are located according to the fairlead, anchor, and connect (if applicable) coordinates provided in the input file.
ラインの端部はフェアリード、アンカーに従って配置され、入力ファイルの座標を接続する（該当する場合）
In the second step, dynamic relaxation is used to allow the mooring system to settle to equilibrium according to the MoorDyn model.
第 2 ステップでは動的緩和が使用される。係留システムは MoorDyn モデルに従って平衡に沈降する
If there are no connect nodes, this will simply fine-tune the results of the quasi-static model to account for the discrete approach of MoorDyn.
connect 節点がない場合、これは調整準静的モデルの結果を調整する
MoorDyn の離散アプローチを考慮する。
If there are connect nodes, this will allow them to settle to their correct positions, rather than the guessed positions provided in the input file.
connect 節点がある場合、入力ファイルで提供される推測位置よりも、
彼らの正しい位置に落ち着く。

During each coupling time step, MoorDyn accepts the latest platform or fairlead position and velocity information provided by the calling program and applies these to the appropriate fairlead nodes in its model.
各連成時間ステップで、MoorDyn は呼び出しプログラムによって提供される最新のプラットフォームまたはフェアリード位置および速度情報を受け取り、そのモデル内の適切なフェアリード節点にこれらを適用する。
It adjusts its internal time step size (dtm) to ensure that the coupling time step size (dtg) is a multiple of dtm.
これは、連成時間ステップサイズ（DTG）が DTM の倍数であることを確実にするため、内部時間ステップサイズ（DTM）を調整する。
It then runs its internal RK2 integrator for Nt time steps, where Nt=dtg/dtm.
その後、内部 RK2 積分を Nt 時間ステップ行う。Nt=dtg/dtm
During each model evaluation from the RK2 integrator, a number of steps take place:
RK2 積分から各モデル評価の間、以下の多数のステップ数が実施される：

The fairlead kinematics at times t and t+dtm/2 are calculated.
時刻 t と t+dtm/2 のフェアリード運動が計算される
The forces on the nodes of every Line are calculated.
ラインの節点の力が計算される。
The accelerations of the internal nodes of every Line are calculated.
ラインの内部節点の加速度が計算される。

The forces on each Connection node are calculated by summing the contributions of any connected lines as well as any external forces.
Connection 節点の力は、接続ラインと外力の加算から算出される。
The accelerations of any connect-type Connection nodes are calculated.
接続タイプの Connection 節点の加速度を計算する。
The calculated accelerations of the internal and connect nodes are integrated twice to find the velocities and positions of the internal and connect nodes at time t+dtm.
内部節点と接続節点の加速度は、時刻 t+dtm の節点の速度、位置を求めるため 2 回積分される。
At the end of the coupling time step, MoorDyn returns the resulting net mooring force (in six directions on the platform) or individual fairlead forces to the calling program.
連成時間ステップの終了時に、MoorDyn は正味の係留力（プラットフォームの 6 方向）もしくは個々のフェアリード力を返す。
One or more output files may be written at this point depending on the MoorDyn version and the settings.
MoorDyn バージョンや設定によっては、この時点で一つまたは複数の出力ファイルが書かれる。
During termination, MoorDyn deallocates variables and closes the output files.
終了時、MoorDyn は、変数割当てを解除して出力ファイルを閉じる。
More details about the function calls available to the calling program to make MoorDyn run are described in Sections 5 and 6.
MoorDyn 実行できる呼び出しプログラムの関数呼び出し詳細が節 5 および 6 に記載される。

In the C++ version, the fairlead position at each model evaluation is calculated
by integrating the most recent velocity supplied by the calling program.
C++バージョンでは、各モデル評価のフェアリード位置は、
呼出しプログラムで提供される最新速度の積分から計算される。
This assumes constant fairlead velocity within each coupling time step.
これは、各連成時間ステップ内で一定フェアリード速度を想定する。
In the FAST v8 version, the modularization framework's ExtrapInterp subroutine is used to provide a more accurate estimate of the fairlead kinematics within each coupling time step.
FAST V8 では、モジュール化フレームワークの ExtrapInterp サブルーチンが使用される。
各連成時間ステップ内でフェアリード運動のより正確な推定値を提供する。

4. Describing the Mooring System
4.係留システムの記述
The entire description of the mooring system as used by MoorDyn is contained in one input file.
MoorDyn で使用される係留システム全体の説明は 1 つの入力ファイルに含まれる。
The structure of this file is based on the MAP input file format by Marco Masciola [2], but without

MAP's 'depth' and 'repeat' functions and with some additions for supporting a dynamic mooring model.
このファイルの構造は、Marco Masciola の MAP 入力ファイル形式に基づく[2]。
しかし MAP には「深さ」と「リピート」機能がないため、動的係留モデルをサポートのためいくつか追加された。
There are a few differences depending on whether the C++ or FAST v8 version of MoorDyn is used.
MoorDyn の C ++または FAST V8 で、いくつか違いがある。
In the C++ version, the input file must be called 'lines.txt' and exist in a subdirectory named 'Mooring'.
C ++では、入力ファイルは「lines.txt」固定であり、Mooring ディレクトリにおく。
In the FAST v8 version that filename can be specified separately.
FAST V8 ではファイル名は別々に指定できる。
Below is an example MoorDyn input file for the OC3-Hywind mooring system.
以下は、OC3-Hywind 係留システムの MoorDyn 入力ファイルである。
Lines in blue are specific to the FAST v8 version of MoorDyn; they should be omitted when using the C++ version.
青線は MoorDyn の FAST V8 固有である。 C ++では省略される。

---------- MoorDyn Input File ----------
MoorDyn input file of the mooring system for OC3-Hywind
*FALSE Echo - echo the input file data (flag)
---------- LINE TYPES ----------
*1 NTypes - number of LineTypes
Name Diam MassDen EA BA/-zeta Can Cat Cdn Cdt
(-) (m) (kg/m) (N) (N-s/-) (-) (-) (-) (-)
main 0.09 77.7066 384.243E6 -0.8 1.0 0.0 1.6 0.1
---------- CONNECTION PROPERTIES ----------
*6 NConnects - number of connections including anchors and fairleads
Node Type X Y Z M V FX FY FZ CdA Ca
(-) (-) (m) (m) (m) (kg) (m^3) (kN) (kN) (kN) (m^2) (-)
1 fixed 853.87 0.0 -320.0 0 0 0 0 0 0 0
2 fixed -426.94 739.47 -320.0 0 0 0 0 0 0 0
3 fixed -426.94 -739.47 -320.0 0 0 0 0 0 0 0
4 vessel 5.2 0.0 -70.0 0 0 0 0 0 0 0
5 vessel -2.6 4.5 -70.0 0 0 0 0 0 0 0
6 vessel -2.6 -4.5 -70.0 0 0 0 0 0 0 0
---------- LINE PROPERTIES ----------
*3 NLines - number of line objects

Line LineType UnstrLen NumSegs NodeAnch NodeFair Flags/Outputs
(-) (-) (m) (-) (-) (-) (-)
1 main 902.2 20 1 4 p
2 main 902.2 20 2 5 -
3 main 902.2 20 3 6 -
---------- SOLVER OPTIONS ----------
0.001 dtM - time step to use in mooring integration (s)
3.0e6 kBot - bottom stiffness (Pa/m)
3.0e5 cBot - bottom damping (Pa-s/m)
320 WtrDpth - water depth (m)
1.0 dtIC - time interval for analyzing convergence during IC gen (s)
60.0 TmaxIC - max time for IC gen (s)
4.0 CdScaleIC - factor by which to scale drag coefficients during dynamic relaxation (-)
0.001 threshIC - threshold for IC convergence (-)
---------- OUTPUTS ----------
*FairTen1
*FairTen2
*FairTen3
*AnchTen3
*L2N4pX
*END
---------- need this line ----------

The Line Types section of the file contains one or more definitions of physical line properties and four hydrodynamic coefficients.
Line Types では、ラインの物理特性の 1 つ以上の定義、4 つの流体力係数が含まれる。
The columns are, in order, as follows:
列の順番は以下のとおりである。
Name:　an identifier word for the line type
Name：ラインの識別子
Diam:　the volume-equivalent diameter of the line - the diameter of a cylinder having the same displacement per unit length (m)
Diam：ラインの体積相当直径。単位長当たり同じ体積を有するシリンダの直径（m）
MassDen:　the mass per unit length of the line (kg/m)
MassDen：単位長さ当たりの質量（kg/m）
EA: the line stiffness, product of elasticity modulus and cross-sectional area (N)
EA：ライン剛性、弾性率と断面積の積（N）
BA/-zeta:　the line internal damping (measured in N-s) or, if a negative value is entered, the

desired damping ratio (in fraction of critical) for the line type
(and MoorDyn will set the BA of each line accordingly - see Section 4.1 for more information)
BA/-zeta：内部減衰（単位 N-s)。負値の場合、ラインタイプの所望の減衰比（臨界分率）
（MoorDyn は各ラインに応じて BA を設定する。詳細は 4.1 節参照）
Can: transverse added mass coefficient (with respect to line displacement)
Can: 横方向の負荷質量係数（ライン変位に対応する）
Cat: tangential added mass coefficient (with respect to line displacement)
Cat: 接線方向の付加質量係数（ライン変位に対応する）
Cdn: transverse drag coefficient (with respect to frontal area, d*l)
Cdn: 横方向抗力係数（前面面積に対して、d*l)
Cdt: tangential drag coefficient (with respect to surface area, π*d*l)
Cdt: 接線方向抗力係数（表面積に対して、π*d*l)

The Connection Properties section defines the connection node points which mooring lines can be connected to.
Connection Properties 節は係留ラインが接続可能な接続節点を定義する。
The columns are as follows:
列は以下のとおりである。
Node: the ID number of the connection (must be sequential starting with 1)
Node: 接続の ID 番号（1 はじまりで連続のこと）
Type: one of 'Fixed', 'Vessel', or 'Connect', as described in Section 2.
Type: 第 2 章説明のように'Fixed', 'Vessel', 'Connect'のどれか
X,Y,Z: Coordinates of the connection (relative to inertial reference frame if 'fixed' or 'connect', relative to platform reference frame if 'vessel').
In the case of 'connect' nodes, it is simply an initial guess for position before MoorDyn calculates the equilibrium initial position. (m)
X,Y,Z: 接続の座標 ('fixed' or 'connect' では慣性基準フレームに関して、'vessel'の場合プラットフォームの基準フレームに関して)。
'connect'節点では、MoorDyn が平衡初期位置を算出する前の、位置の初期推定(m)
M: node mass in the case of clump weights (kg)
M: クランプ重量の節点質量(kg)
V: node displacement in the case of floats (m^3)
V: フロートの節点体積(m^3)
FX,FY,FZ: any steady external forces applied to the node (N)
FX,FY,FZ: 節点に作用する定常外力(N)
CdA: product of drag coefficient and projected area (assumed constant in all directions) to calculate a drag force for the node (m^2)
CdA: 節点の抗力計算用の抗力係数と投影面積の積（全方向一様と仮定）(m^2)

Ca: added mass coefficient used along with V to calculate added mass on node
Ca: 節点の負荷質量計算するために V と一緒に使用される付加質量係数

The Line Properties section defines each uniform-property section of mooring line to be simulated, specifying which physical properties it uses, its length, how many segments it is discretized into, which nodes it is connected to, and any data to be output in a dedicated output file for that line.
Line Properties は、計算する各係留ラインの均一な仕様を定義する。
使用する物理仕様、長さ、離散要素数、接続される節点、出力されるデータを指定する。
This last entry expects a string of one or more characters without spaces, each character activating a given output property.
この最後のエントリは、空白のない 1 文字以上の文字列が必要。
各文字は、指定された出力仕様をアクティブにする。
A placeholder character such as '-' should be used if no outputs are wanted. Eight output properties are currently possible:
出力が必要ない場合は、 ' - 'などのプレースホルダ文字を使用する必要がある。
現在 8 つの出力プロパティが可能：
p: node positions
P：節点位置
v: node velocities
V：節点速度
U: wave velocities at each node
U：各節点の波速
D: hydrodynamic drag force at each node
D：各節点の流体抗力
t: tension force at each segment
T：各要素の張力
c: internal damping force at each segment
C：各要素の内部減衰力
s: strain of each segment
S：各要素のひずみ
d: rate of strain of each segment
D：各要素のひずみ率

For example, outputting node positions and segment tensions could be achieved by writing 'pt' for this last column.
たとえば、節点位置と要素張力を出力するには、最後の列に「pt」を記述する。
These outputs will go to a dedicated output file for each line only.
これらの出力は各行専用の出力ファイルに移動する。

For sending values to the global output file, use the Outputs section instead.
グローバルファイルに値を出力する場合、代わりに Outputs セクションを使用する。
The Solver Options section can contain any number of optional settings for the overall model, including seabed properties, initial condition (IC) generation settings, and the time step size.
ソルバーオプション部は全体モデルのオプションの設定を任意の数含む。
海底特性、初期条件（IC）の生成設定、時間ステップサイズなどが含まれる。
Any of these lines can be omitted, in which case default values will be used.
これらの行は省略でき、その場合デフォルト値が使用される
As such, they are all optional settings, although some of them (such as time step size) often need to be set by the user for proper operation.
このように、すべてはオプション設定である。しかし、いくつかは適切な運用のためユーザが設定する必要がある。(時間ステップサイズなど)
Note that the names for these have been changed in the latest C++ version, v1.0.1C.
これらの名前は、最新の C++版、v1.0.1C で変更されていることに注意。

The list of possible options is:
可能なオプションは以下のとおり。
dtM: desired mooring model time step (s)
dtM：所望の係留モデル時間ステップ（s）
g: gravitational constant (m/s^2)*
g：重力定数(m/s^2)*
rhoW: water density (kg/m^3)*
rhoW：水の密度(kg/m^3)*
WtrDpth: water depth (m)*
WtrDpth：水深(m)*
kBot: bottom stiffness constant (Pa/m)
kBot：海底剛性定数(Pa/m)
cBot: bottom damping constant (Pa-s/m)
cBot：海底減衰定数（Pa-s/m）
dtIC: period for analyzing convergence of dynamic relaxation IC generation (s)
dtIC：初期条件 IC（Initial Condition）生成の動的緩和の収束分析の期間（s）
TmaxIC: maximum simulation time to allow for IC generation without convergence (s)
TmaxIC：収束なしで IC 生成可能な最大のシミュレーション時間（s）
CdScaleIC: factor by which to scale drag coefficients to accelerate convergence of IC generation (-)
CdScaleIC：IC 収束を加速する抗力係数のスケーリング係数（-）
ThreshIC: convergence threshold for IC generation, acceptable relative difference between three successive fairlead tension measurements (-)
ThreshIC：IC を生成する収束しきい値。連続する 3 つのフェアリード張力測定値の許容差（ - ）

*In the FAST v8 version, the default values for g, rhoW, and WtrDpth are the values provided by FAST, so it is recommended to not use custom values for the sake of consistency.
* FAST V8 では、g,rhoW,WtrDpth のデフォルト値は FAST が提供するため、カスタム値を使用しない。
The bottom contact parameters, kBot and cBot, result in a pressure which is then applied to the cross-sectional area (d*l) of each contacting line segment to give a resulting vertical contact force for each segment.
海底接触パラメータ kBot および cBot は、各接触ライン要素の断面積（d*l）に加えられる圧力をもたらし、結果として各要素の垂直接触力を得る。
The Outputs section is used in the FAST v8 version to specify general outputs,
which are written to the main MoorDyn output file and
also sent to the driver program for inclusion in the global output file.
出力セクションは FAST v8 バージョンで一般的な出力を指定するために使用される。
これらの出力はメインの MoorDyn 出力ファイルに書き込まれ、グローバル出力ファイルに含めるためにドライバプログラムにも送られる。
Each output channel name should have its own line.
各出力チャンネル名は、独自のラインを持つこと。
There are intuitive keywords for fairlead and anchor tensions of a given line:
指定されたラインのフェアリードとアンカー張力の直感的なキーワードがある。
fairten# and anchten#, where # is the line number.
fairten # , anchten # ， # は行番号である。
There is also a flexible naming system for outputting other quantities.
他の量を出力するための柔軟なネーミングシステムもある。
There are currently five supported types of output quantities:
現在 5 つの出力がサポートされる
pX, pY , pZ: x/y/z coordinate (m)
PX, PY, pZ：x/y/z（m）
vX, vY, vZ: velocity (m/s)
vX, vY, vZ：速度（m/s）
aX, aY, aZ: acceleration (m/s^2)
aX, aY, aZ：加速度（m/s^2）
T or Ten: tension (N)
T or Ten：張力（N）
fX, fY, fZ: net force in x/y/z direction (N)
fX, fY, fZ：x/y/z 方向の正味の力（N）
These can be produced at a connection object, denoted by the prefix Con#, where # is the connect number.
接続オブジェクトは Con#で表される。#は接続番号である。

Or, they can be produced at a node along a line, denoted by the prefix L#N@, where # is the line number and @ is the number of the node along that line.
あるいは、ラインに沿った節点で生成され、L#N@で表される。#はライン番号、@はラインに沿った節点である。
For example, Con3vY outputs the connection 3 y velocity, L2N4pX outputs the line 2, node 4 x position.
例えば、Con3vY は connection 3 の Y 速度を出力、L2N4pX はライン 2、節点 4 の X 位置を出力する。
These capabilities are not yet included in the C++ version of MoorDyn; instead, this version always creates a lines.out output file containing the tensions of all fairlead connections.
これらの機能は、まだ MoorDyn の C ++に含まれていない。代わりに、常にすべてのフェアリード接続の張力を含む lines.out ファイルを作成する。
For now, any additional quantities can be obtained by using the optional line-specific output files as defined in the Line Properties section.
ここでは、任意の追加量は、Line Properties セクションで定義されたオプションのライン固有の出力ファイルを使用することによって得ることができる。

4.1. Model Stability and Segment Damping
4.1。モデルの安定性と要素減衰
Most of the entries in the input file are pretty straightforward and can be set according to common sense.
入力ファイル内の項目のほとんどは非常に簡単であり、常識的に設定できる。
Two of the trickier input parameters are the internal damping (BA) for each line type, and the mooring simulation time step (dtM).
トリッキーな入力パラメタは、各ラインタイプの内部減衰（BA)、と係留シミュレーション時間ステップ（DTM）である。
Both relate to the discretization of the lines.
両方ともラインの離散化に関係する。
The highest axial vibration mode of the lumped-mass cable representation would be when adjacent nodes oscillate out of phase with each other, as depicted below.
集中質量ケーブル表現における軸方向振動の最大モードは、隣接節点が互いに逆位相振動するときに発生する。
In this mode, the midpoint of each segment would not move.
このモードでは、各要素の中点が移動しない。
The motion of each node can then be characterized by mass-spring-damper values of
その後、各節点の運動は、質量-ばね-ダンパ値で特徴づけられる
m=wL/N, c=4NBA/L, k=4NEA/L
m=wL/N, c=4NBA/L, k=4NEA/L

The natural frequency of this mode is then
このモードの固有振動数は以下である
wn=sqrt(k/m)=2/l*sqrt(EA/w)=2N/L*sqrt(EA/w)
wn=sqrt(k/m)=2/l*sqrt(EA/w)=2N/L*sqrt(EA/w)
and the damping ratio, Xi, is related to the internal damping coefficient, BA, by
減衰比ξは内部減衰係数 BA に関係する
Xi=c/c_crit=B/l*sqrt(A/Ew)=NBA/L*sqrt(1/EAw)
Xi=c/c_crit=B/l*sqrt(A/Ew)=NBA/L*sqrt(1/EAw)
-> BA=XiL/N*sqrt(EAw)
The line dynamics frequencies of interest should be lower than wn in order to be resolved by the model.
ラインの運動周波数は wn より低いこと。モデルによって解くため。
Accordingly, line dynamics at wn, which are likely to be dominated by the artificial resonance created by the discretization, can be damped out without necessarily impacting the line dynamics of interest.
したがって、wn でのラインの運動は、離散化によって作られた擬似共振に支配され、ラインの興味ある運動に影響を与えることなく減衰させることができる。
This is advisable because the resonances at wn can have a large impact on the results.
wm の共振が結果に大きな影響を持つことができるので、これはお勧めである。
To damp out the segment vibrations, a damping ratio approaching the critical value (Xi=1) is recommended.
要素振動を減衰させるため、臨界値（XI=1）に近い減衰比が推奨される。
Care should be taken to ensure that the line dynamics of interest are not affected.
ラインの運動が影響されないよう注意する。
To simplify things, a desired line segment damping ratio can be specified in the input file.
物事を簡単にするため、所望の要素減衰比は、入力ファイルで指定できる。
This is done by entering the negative of the desired damping ratio in the BA/-zeta field of the Line Types section.
これは、Line Types の BA/-zeta に所望の負の減衰比を入力する。
A negative value here signals MoorDyn to interpret it as a desired damping ratio and then calculate the damping coefficient (BA) for each mooring line that will give every line segment that damping ratio
(accounting for possible differences in segment length between lines).
ここで、負の値を設定すると、MoorDyn はこれを希望の減衰比として解釈し、すべての線分に減衰比を与える各係留線の減衰係数(BA)を計算する。(ライン間の要素長の差異を考慮に入れる)。
Note that the damping ratio is with respect to the critical damping of each segment along a mooring line, not with respect to the line as a whole or the floating platform as a whole.
減衰比は、係留ラインに沿った各要素の臨界減衰に関連しており、ライン全体または浮体プラットフォーム全体に対してではない。

It is just a way of letting MoorDyn calculate the damping coefficient automatically from the perspective of damping non-physical segment resonances.
MoorDyn に物理的でない部分の共振を減衰させるという観点から自動的に減衰係数を計算させる方法である。
If the model is set up right, this damping can have a negligible contribution to the overall damping provided by the moorings on the floating platform.
モデルが正しく設定されていれば、この減衰は、浮動プラットフォーム係留によって提供される全体的な減衰にはほとんど貢献しない可能性があります。
However, if the damping contribution of the mooring lines on the floating platform is supposed to be significant, it is best to (1) set the BA value directly to ensure that the expected damping is provided and then (2) adjust the number of segments per line to whatever provides adequate numerical stability.
しかし、フローティングプラットフォーム上の係留線の減衰寄与が重要であると想定される場合、（1）期待される減衰が確実に提供されるように BA 値を直接設定し、次に（2）要素数を調整する適切な数値安定性を提供するものであれば何でもよい。

4.2. Diagnosing Problems
4.2。問題の診断
The factors described in the previous section are the source of most problems encountered by new users.
前節で説明した要因は、新規ユーザーが遭遇するほとんどの問題の源である。
Another source of problems is the initial positions of connection points in more complex mooring systems.
問題の別の源は、より複雑な係留システムにおける接続ポイントの初期位置である。
Most problems reveal themselves during initialization, while MoorDyn does a dynamic relaxation process, running the model with the initial fairlead positions to allow the mooring system to settle to equilibrium.
ほとんどの問題は初期化中に明らかになりますが、MoorDyn は動的緩和プロセスを実行し、最初のフェアリードの位置でモデルを実行し、係留システムを平衡状態に落ち着かせる。
In the case of instability (NaN results) or other suspected problems during this stage, the following is a method to see what is happening.
この段階で不安定（NaN）や他の疑わしい問題が発生した場合、次のことが起こっているかどうかを確認する。
First, ensure that the node position outputs are enabled for each line (set with output flag 'p').
最初に、ノード位置出力が各ラインに対して有効になっていることを確認する（出力フラグ 'p'で設定）。
These allow post-process visualization of the mooring line behavior.
これにより、係留線の挙動をポストプロセスで可視化できる。

Then, set TmaxIC = 0.
次に、TmaxIC = 0 に設定する。
This will bypass the dynamic relaxation stage and start the simulation only from initial conditions calculated by the catenary quasi-static algorithm for each line.
これは、動的緩和ステージをバイパスし、各ラインのカテナリ準静的アルゴリズムによって計算された初期条件からのみシミュレーションを開始する。
Connection points will start at their initial positions as specified in the input file.
接続点は、入力ファイルで指定された初期位置から開始する。
This mimics the dynamic relaxation process in a way in which data can be output, in order to see what is actually happening.
これは、実際に何が起こっているかを見るために、データを出力することができるように動的緩和プロセスを模倣する。
The only difference in the model between this normal operation and dynamic relaxation is that the damping forces cannot be exaggerated.
この通常運動と動的緩和間のモデルにおける唯一の違いは、減衰力を誇張することができないことである。
(This is not normally the issue with dynamic relaxation problems. If it is, it can be solved by just setting CdScaleIC = 1.)
（これは通常、動的緩和問題の問題ではない。もしそうであれば、CdScaleIC＝1 を設定することで解決できる）
Visual analysis of the line motions given by the above process usually indicates what sort of problem the model is encountering.
上記のプロセスによって与えられた線の動きの視覚的分析は、通常、モデルが遭遇している問題の種類を示す。
Once problems are resolved, the dynamic relaxation initialization can be turned back on (TmaxIC > 0) and a damping exaggeration (CdScaleIC > 1) can be applied to provide a faster initialization to static equilibrium.
問題が解消されると、動的緩和初期化をオンに戻すことができ（TmaxIC> 0)、ダンピングオーバーグレーディング(CdScaleIC> 1)を適用して静的均衡化をより早く初期化することができます。

5. MoorDyn for FAST v8
5. MoorDyn for FAST V8
In parallel with the C++ version (see Section 6), MoorDyn has been completely rewritten in FORTRAN for inclusion in FAST v8, with guidance from Marco Masciola, Bonnie Jonkman, and Jason Jonkman.
C++版（6 章を参照）と並行して、Marco Masciola, Bonnie Jonkman, and Jason Jonkman のガイダンスと、FAST V8 に含めるため FORTRAN に書き換えられた。
The original model structure was retained as much as possible.

オリジナルのモデル構造は、可能な限り保持された。
There are no known/intentional differences in the mooring dynamics represented by this version of MoorDyn compared to the original C++ version.
オリジナルのC++版に比べ、MoorDynによって表される係留運動には既知/意図的な違いはない。
There are important differences in the interfacing functions, however, since MoorDyn F follows the FAST Modularization Framework [5], which specifies certain function forms and data structures to achieve a high degree of control over the coupling.
しかし、MoorDyn Fは、連成の高度な制御を実現するための特定の関数形式とデータ構造を指定する FAST Modularization Framework [5]に従っているため、インタフェース関数に重要な違いがある。
The important subroutines for coupling with MoorDyn F are:
MoorDyn Fとの結合の重要なサブルーチンは、次のとおりである。
MD_Init: initializes MoorDyn, including reading the input file, creating the mooring system data structures, and calculating the initial conditions.
MD_Init：MoorDynの初期化。入力ファイルを読み取る係留システムデータ構造を作成し、初期条件を計算する。
MD_UpdateStates: instructs MoorDyn to run its model from time t up to time t+dtg in a loose coupling arrangement. It accepts inputs about the fairlead kinematics and returns the fairlead forces at the end of the time integration.
MD_UpdateStates：MoorDynに時刻 tから t+dtgまでの疎連成計算を指示する。フェアリードの運動を入力として受け入れ、時間積分の最後でフェアリード力を返す。
MD_CalcOutput: calculates all requested output quantities based on the provided states of the mooring system.
MD_CalcOutput：要求された出力量を係留システムの提供状態に基づいて計算する。
Requested general output quantities are written to the MoorDyn output file and also returned to FAST for possible inclusion in the global output file.
要求された一般出力量は、MoorDyn出力ファイルに書き込まれ、グローバル出力ファイルに含めることができるように FASTにも戻される。
Outputs for line-specific files will be written if enabled.
有効にした場合、ライン固有のファイルの出力が書き込まれます。
MD_CalcContStateDeriv: contains the core of the MoorDyn model.
MD_CalcContStateDeriv：MoorDynモデルのコアが含まれる
Based on the current inputs and state variables, it calculates the instantaneous forces on the mooring system nodes.
現在の入力と状態変数に基づいて、係留システムの節点における瞬間の力を計算する。
From these, it calculates the node accelerations, which are the derivatives of the state variables that can be integrated to move the model forward in time.
これらから、節点の加速度が計算される。これは、モデルを時間進展するために積分する状態変

数の導関数である。
This subroutine is called by MD_UpdateStates in a loose coupling arrangement.
このサブルーチンは、疎連成における MD_UpdateStates によって呼び出される。
Alternatively, it can be called by the driver program in a tight coupling arrangement. It is also called by CalcOutput.
あるいは、密連成におけるドライバプログラムによって呼び出すことができる。また、CalcOutput によって呼び出される。
MD_End: terminates the MoorDyn portion of the simulation and cleans up memory.
MD_End：シミュレーションの MoorDyn 部分を終了し、メモリをクリーンアップする。
The arguments and operation of these functions follow the FAST Modularization Framework, which can be referenced for more information.
これらの関数の引数と操作は、FAST モジュール化フレームワークに従います。これについては、詳細を参照してください。
Refer to the source code for details specific to MoorDyn.
MoorDyn への具体的な詳細については、ソースコードを参照してください。
This FORTRAN version of MoorDyn is included as a module in FAST v8, available at nwtc.nrel.gov/FAST8.
MoorDyn FORTRAN 版は nwtc.nrel.gov/FAST8 で利用可能な FAST V8 内のモジュールとして含まれています。
The MoorDyn F web page is nwtc.nrel.gov/MoorDyn.
MoorDyn F Web ページは nwtc.nrel.gov/MoorDyn である。
The FAST modularization framework [5] focuses on a standard centrally-controlled data structure that imposes strict requirements on constituent models.
FAST モジュール化フレームワーク[5] は、集中制御データ構造に焦点を当てており、構成モデルに厳しい要件を課す。
Accordingly, MoorDyn F could be easily coupled with other FORTRAN codes following the same modularization framework.
したがって、MoorDyn F は、同じモジュール化フレームワークに従う他の FORTRAN コードと簡単に結合できる。
More specifically, if a driver or glue code adheres to the framework in how it calls mooring models, MoorDyn F will be able to work with it.
より具体的には、ドライバまたは中間コードがフレームワークに係属して係留モデルと呼ばれる方法を遵守すれば、MoorDyn F はそれに対応できる。
For coupling directly with codes not following the framework or written in other languages, MoorDyn C is recommended.
フレームワークに則らないコードや他の言語で書かれていないコードへの直接結合には、MoorDyn C 版が推奨される。

6. MoorDyn in C++
6. MoorDyn in C++
The C++ version of MoorDyn was written from the beginning with the goal of making the coupling with other models as simple and generic as possible.
MoorDyn C++版は、可能な限りシンプルかつ汎用的に他のモデルと連成することを目標に書かれた。
In contrast to the FORTAN version, MoorDyn C functions are designed for coupling arrangements in which models keep track of their data structures internally and the data passed between models is kept to a minimum.
FORTAN バージョンとは対照的に、MoorDyn C 関数は、モデルが内部的にデータ構造を追跡し、モデル間で渡されるデータを最小限に保つ結合配列のために設計された。
By working toward generic, minimalistic coupling functions, it should be easier to set up couplings between different simulation tools, across different programming languages, and perhaps without requiring source code changes.
一般的に最小限の連成機能に取り組むことで、異なるシミュレーションツール間、異なるプログラミング言語間、およびおそらくソースコードの変更なしにカップリングを設定する方が簡単である。
The source code, Windows binaries, and examples can be obtained from www.matt-hall.ca/moordyn.
ソースコード、Windows バイナリ、および例は www.matt-hall.ca/moordyn から入手できる。
Coupling MoorDyn C with other programs relies on a few simple core function calls, as shown below with their arguments.
他のプログラムとの MoorDyn C 連成は、いくつかの簡単なコア関数の呼び出しに依存する。その引数を以下に示す。
LinesInit(double X[], double XD[]): initializes MoorDyn, loading the MoorDyn input file and calculating initial conditions based on platform position specified by array X (size 6). It will write the t=0 output line to any output files.
LinesInit(double X[], double XD[])：MoorDyn を初期化する。MoorDyn 入力ファイルをロードし、配列 X（サイズ 6）で指定されたプラットフォームの位置に基づいて初期条件を計算する。t=0 の出力行を出力ファイルに書き込みます。
LinesCalc(double X[], double XD[], double Flines[], double* t, double* dt): makes MoorDyn simulate the mooring system starting at time t and ending at time t+dt.
LinesCalc(double X[], double XD[], double Flines[], double* t, double* dt)：MoorDyn が時間 t から t+dt までの係留システムをシミュレートする。
The fairlead kinematics are driven by the platform position and velocity vectors (X and Xd) which correspond to time t.
フェアリードの運動は、時間 t のプラットフォームの位置 X と速度ベクトル Xd によって駆動される。
For each internal MoorDyn time step, the platform velocity is assumed constant at Xd and the

position is adjusted accordingly at each step from the initial value X.
各内部 MoorDyn 時間ステップについて、プラットフォーム速度は Xd で一定と仮定され、位置は初期値 X から各ステップでそれに応じて調整される。
The resulting net mooring force about the platform in six directions is returned via vector Flines.
その結果、プラットホームの 6 方向の正味係留力はベクトル Flines を介して返される。
LinesClose(void): This function deallocates the variables used by MoorDyn. It should be called last before unloading the MoorDyn DLL.
LinesClose(void)：この関数は MoorDyn で使用される変数の割り当てを解除する。MoorDyn DLL をアンロードする前に、最後に呼び出される。
In addition to the core functions, additional functions exist for specific applications.
コア機能に加えて、特定の用途のために追加の機能が存在する。
A number of these functions exist, and the idea is that additional ones can be created as needed, without altering the fundamental structure of the model.
これらの関数がいくつか存在し、モデルの基本構造を変更することなく、必要に応じて追加の関数を作成できるという考えがある。
As new functions are created, I hope they will be shared for the convenience of all users.
新しい機能はすべてのユーザーの便宜のために共有されることを願う。
Several of these additional functions are currently implemented in the released version:
これらの追加機能のいくつかは、現在、リリースされたバージョンで実装されています：
double GetFairTen(int i): This is an optional function to return the tension at the fairlead of a given line (line number i), which is ideally called after LinesCalc.
double GetFairTen(int i)：これは任意の関数であり、与えられた行（行番号 i）のフェアリードに張力を戻す。これは理想的には LinesCalc の後に呼び出される。
GetFASTtens(int* numLines, float FairHTen[], float FairVTen[], float AnchHTen[], float AnchVTen[]): This is an optional function that returns the line tension variables expected by FAST v7: horizontal and vertical components of the fairlead and anchor tensions.
GetFASTtens(int* numLines, float FairHTen[], float FairVTen[], float AnchHTen[], float AnchVTen[])：これは、FAST v7 で期待されるラインの張力変数を返すオプションの関数。フェアリードの水平および垂直成分とアンカー張力。
GetStates (incomplete) and SetStates (incomplete): these not yet implemented functions will allow for getting and setting of the full MoorDyn state vector describing the node positions and velocities. This can allow for saving simulation states for later continuation, or running the MoorDyn analysis multiple times for a given coupling time step.
GetStates (incomplete) and SetStates (incomplete)：これら未実装の関数は、節点の位置と速度を記述する完全な MoorDyn 状態ベクトルを取得・設定する。 これにより、後で継続するためにシミュレーション状態を保存したり、所定の結合時間ステップで MoorDyn 解析を複数回実行することができます。
All the above functions are accessible to outside programs so that MoorDyn can be compiled as a

DLL for use with other already-compiled codes.
上記の関数はすべて外部のプログラムからアクセスできるので、MoorDyn は既にコンパイルされた他のコードで使用する DLL としてコンパイルすることができます。
Using these functions, it should be easy to use MoorDyn with other simulation tools.
これらの機能を使用すると、他のシミュレーションツールで MoorDyn を使用するのは簡単です。
The following subsections describe simulation tools that have already been used with MoorDyn C.
以下の節では、すでに MoorDyn C で使用されているシミュレーションツールについて説明します。
6.1. Using MoorDyn with FAST v7
6.1. MoorDyn with FAST V7
MoorDyn C was originally designed for coupling with FAST v7 [4].
MoorDyn C は、もともと FAST V7 [4]との連成のために設計された。
Doing so requires a customized version of FAST containing additional functions for calling an external program for the mooring dynamics.
そうすることで、FAST 修正バージョンが必要である。修正版は係留運動に外部プログラムを呼び出す追加機能をもつ。
Source code for a suitable FAST version is available by request.
適し FAST のソースコードは、リクエストにより利用可能である。
Using this FAST version, MoorDyn can be enabled in place of the normal quasi-static mooring model
by setting LineMod to 5 and deleting any mooring line entries in the FAST platform file.
この FAST 使用で、MoorDyn は通常の準静的係留モデルの代わりに有効にできる。
LineMod を 5 に設定し、FAST プラットフォームのファイル内の任意の係留ラインのエントリを削除する。
The required FAST modifications for coupling with MoorDyn are very similar to those used for coupling with OrcaFlex.
MoorDyn 連成に必要な FAST 修正は、OrcaFlex 連成の修正と非常に類似する。
6.2. Using MoorDyn with Matlab
6.2. Using MoorDyn with Matlab
MoorDyn C can be easily used with Matlab.
MoorDyn C は MATLAB で使用できる。
Aside from the correct setup of the input file, which is common to all MoorDyn use, coupling between MoorDyn and Matlab can be accomplished in about a dozen lines of code.
入力ファイルの正確な設定は別として、MoorDyn と MATLAB の連成は約十行のコードで達成できる。
The following lines show a minimalist example.
最低限のコードを示す。

```
------------------------------------------------------------
%% Setup
X = zeros(6,1); % platform position
XD = zeros(6,1); % platform velocity
N = 10; % number of coupling time steps
dt = 0.5; % coupling time step size (time between MoorDyn calls)
Ts = zeros(N,1); % time step array
FairTens1 = zeros(N+1,1); % array for storing fairlead 1 tension time series
FLines_temp = zeros(1,6); % going to make a pointer so LinesCalc can modify FLines
FLines_p = libpointer('doublePtr',FLines_temp); % access returned value with FLines_p.value
%% Initialization
loadlibrary('Lines','MoorDyn'); % load MoorDyn DLL
calllib('Lines','LinesInit',X,XD) % initialize MoorDyn
%% Simulation
XD(1)=0.1; % give platform 0.1 m/s velocity in surge
for i=1:N
  calllib('Lines','LinesCalc',X,XD,FLines_p,Ts(i),dt); % some MoorDyn time stepping
  FairTens1(i+1)=calllib('Lines','GetFairTen',1); % store fairlead 1 tension
  X = X + XD*dt; % update position
  Ts(i+1) = dt*i; % store time
end
%% Ending
calllib('Lines','LinesClose'); % close MoorDyn
unloadlibrary Lines; % unload library (never forget to do this!)
------------------------------------------------------------
```

Always ensure that LinesClose is called (calllib('Lines','LinesClose')) and the library is unloaded (unloadlibrary Lines) before trying to load it again (particularly if the Matlab script hits an error and doesn't finish) to avoid Matlab closing or crashing.
LinesClose が呼ばれることを確認し(calllib('Lines','LinesClose'))、
再ロードの前に（特に Matlab スクリプトがエラーを出して終了していない場合)、
ライブラリをアンロードすること(unloadlibrary Lines)。
さもなければ Matlab が終了あるいはクラッシュする

6.3. Using MoorDyn with Simulink
6.3. Using MoorDyn with Simulink
Using MoorDyn within Simulink adds more components than coupling with Matlab alone (refer to the Matlab section also).

Simulink 内で MoorDyn を使用すると、Matlab だけで結合するより多数の要素が追加される（Matlab のセクションも参照のこと）。
An example is included with MoorDyn.
例は MoorDyn に含まれる。
This approach was developed in partnership with Giacomo Vissio at Politecnico di Torino.
このアプローチは、トリノ工科 Giacomo Vissio と提携して開発された。
The MoorDyn DLL can be loaded and initialized (with the LinesInit function) by placing the appropriate Matlab code within the InitFcn6 callback function window of Simulink.
MoorDyn DLL は、Simulink の InitFcn6 コールバック関数ウィンドウ内に適切な Matlab コードを配置することにより、ロードして初期化できる（LinesInit 関数を使用）。
Similarly, MoorDyn can be closed (with the LinesClose function) and the DLL unloaded using the CloseFcn callback function window.
同様に、MoorDyn は（LinesClose 関数で）閉じることができ、DLL は CloseFcn コールバック関数ウィンドウを使用してアンロードされる。
During time stepping, we found it best to call MoorDyn's LinesCalc and GetFairTen functions using a separate Matlab function, which can be called in Simulink as a triggered subsystem.
タイムステップ処理中は、MoorDyn の LinesCalc 関数と GetFairTen 関数を、Simulink をトリガされたサブシステムとして呼び出すことのできる別の Matlab 関数を使用して呼び出すことを推奨した。

The callback functions can be accessed by right clicking in the Simulink workspace, selecting Model Properties, then going to the Callbacks tab.
コールバック関数にアクセスするには、Simulink ワークスペースを右クリックし、[モデルプロパティ]を選択して[コールバック]タブに移動する。

The time stepping can be implemented using a triggered subsystem block connected to a pulse generator operating at the desired coupling time (dtg) of the simulation (shown on left).
時間ステップは、シミュレーションの所望の結合時間（dtg）（左側に示されている）で動作するパルス発生器に接続されたトリガーサブシステムブロックを使用して実施できる。
Inside this triggered subsystem, a Matlab function block can handle the communication with MoorDyn (shown on right).
このトリガされたサブシステムの内部では、Matlab のファンクションブロックが MoorDyn との通信を処理できます（右図参照）。
Below is an example of this function.
以下は、この関数の例である。
It passes the platform position and velocity as well as the current time and time step size to LinesCalc.
プラットフォームの位置と速度、現在の時間と時間のステップサイズを LinesCalc に渡す。
It then gets any output quantities of interest, in this case fairlead 3 and 4 tensions, and returns them

along with the net mooring force vector.
次に、関心のある出力量、この場合はフェアリード 3 と 4 の張力を得て、正味の係留力ベクトルと一緒にそれらを返す。

```
function [ FLines,Line1_Tens ] = MoorDyn_caller( X,XD,Time,CoupTime )
FLines_value = zeros(1,6);
FLines = zeros(1,6);
Line1_Tens = 0;
FLines_p = libpointer('doublePtr',FLines_value);
calllib('Lines','LinesCalc',X,XD,FLines_p,Time,CoupTime);
Line1_Tens = calllib('Lines','GetFairTen',1);
FLines = FLines_p.value;
end
```

6.4. Using MoorDyn with WEC-Sim
6.4. Using MoorDyn with WEC-Sim
The coupling between MoorDyn and WEC-Sim is a recent development and examples of how to use it should be sought from the WEC-Sim distribution7 for the time being.
MoorDyn と WEC-Sim の結合は近年の発展であり、使用方法の例は当面は WEC-Sim で配布される。
In general, the information about MoorDyn C contained in this document should apply for use with WEC-Sim.
一般的には、本文書の MoorDyn C 情報が WEC-Sim で使用される。

7. References 参考文献
（略）

訳 ver 1.0 今井康貴 imaiy@cc.saga-u.ac.jp