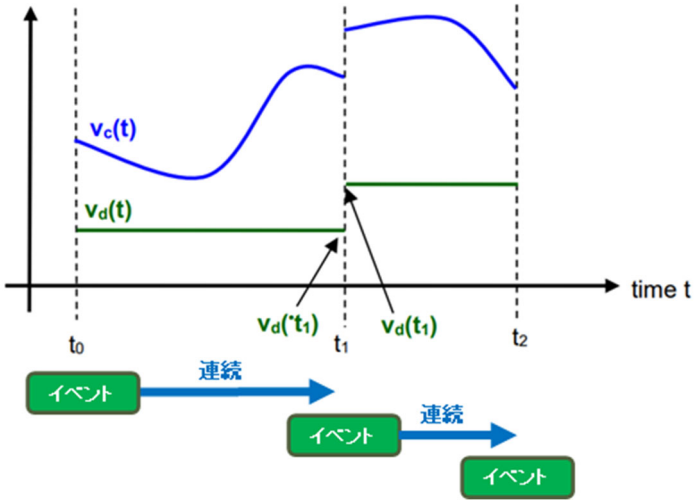
オンラインセミナー
OpenModelica の実践的活用と展望（機械学習とモデル流通）

2021年7月9日(金)開催

プログラム



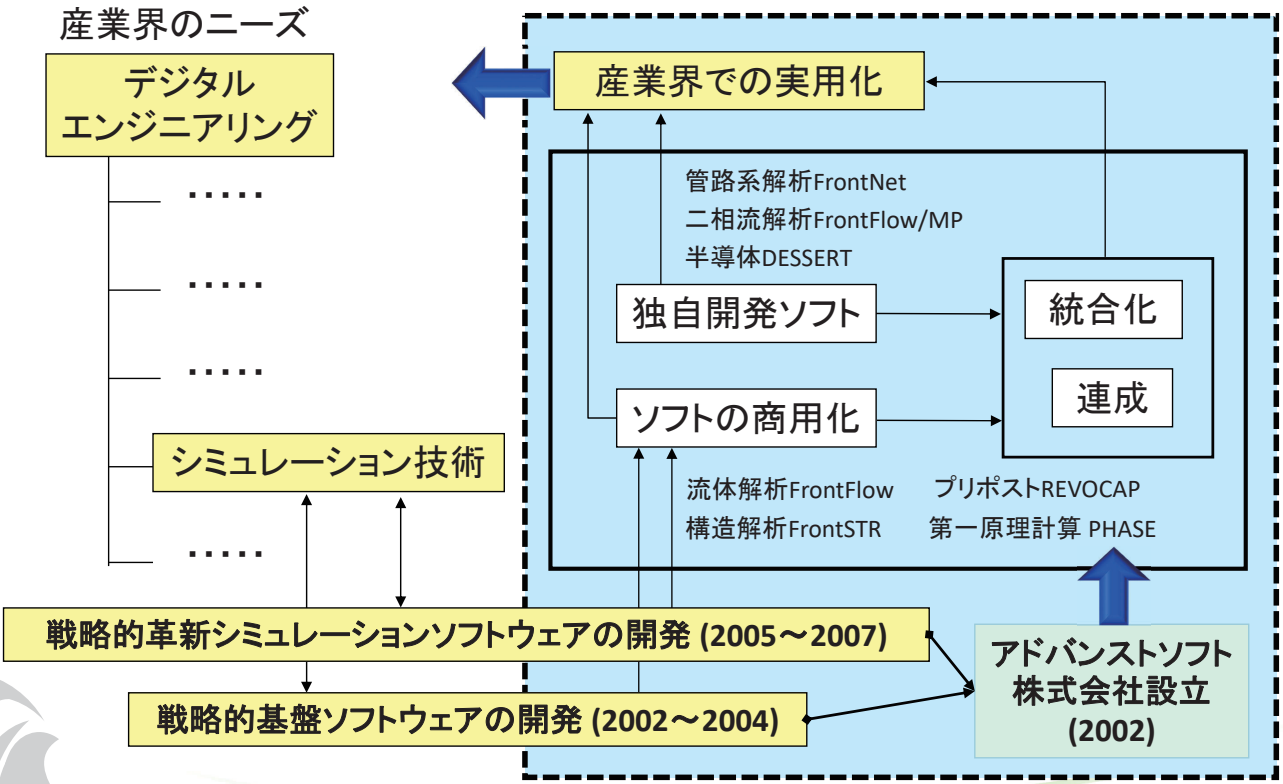
AdvanceSoft

アドバンスソフト株式会社のご紹介

第5事業部　事業部長　松澤 邦裕

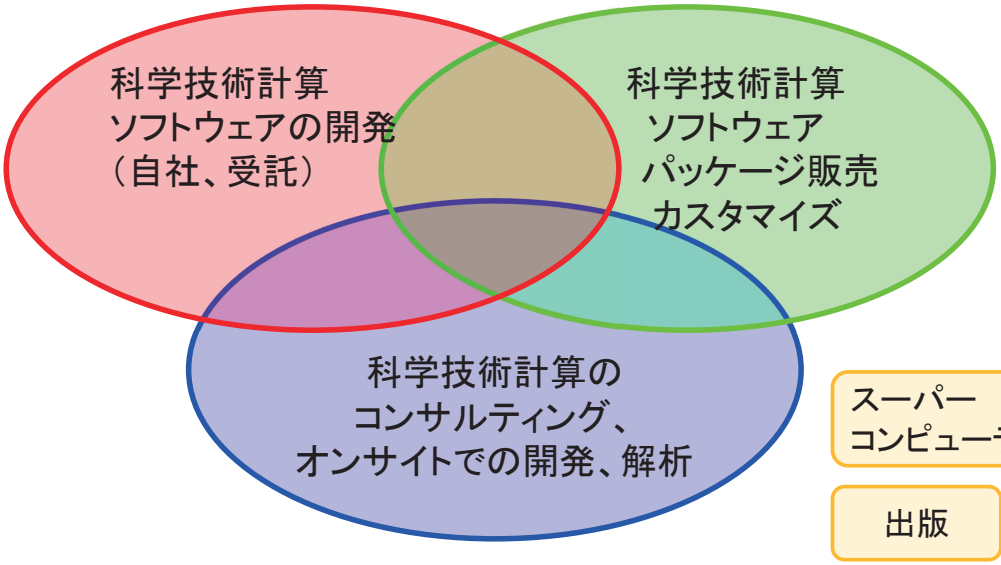
OpenModelica の実践的活用と展望（機械学習とモデル流通）
2021年 7月 9日（金）
アドバンスソフト株式会社

アドバンスソフトとは



事業内容

アドバンスソフトがご提供するサービス

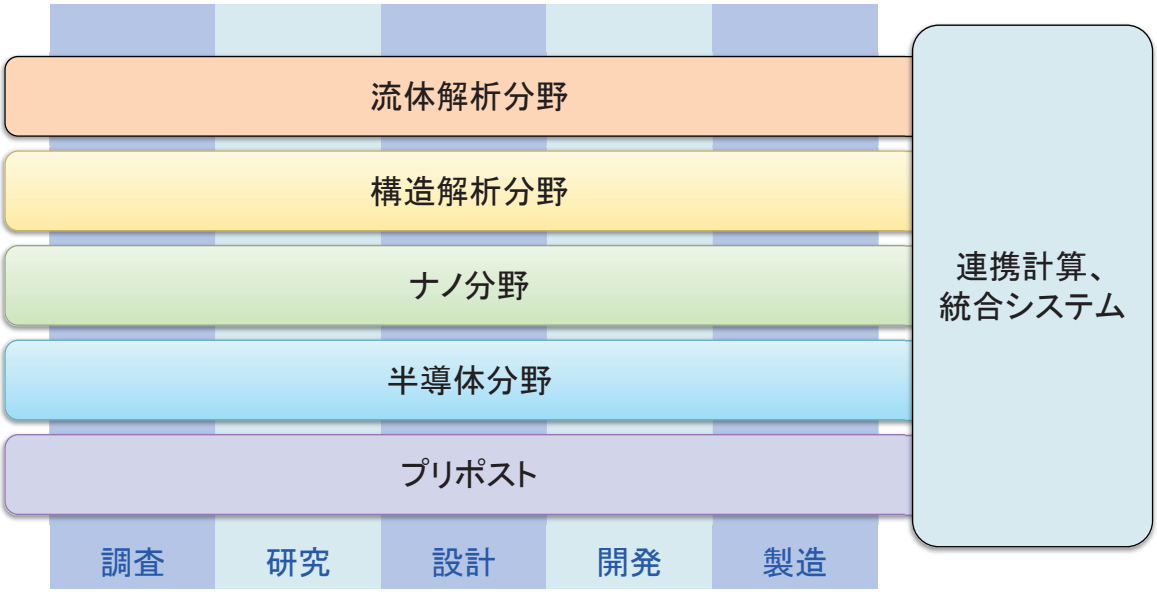


スーパー
コンピューティング

出版

科学技術計算ソフトウェアの開発を基礎とした、
科学技術計算に関する様々なソリューションをご提供します。

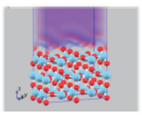
事業分野



産業の主要な分野のあらゆるフェーズで直面する課題に対し、
科学技術計算によるソリューションをご提供します。

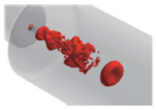
ソフトウェアご紹介

第一原理計算ソフトウェア
Advance/PHASE
ナノ材料 GUI 付属
密度汎関数理論に基づき、物質の性質を原子・分子レベルから解析する第一原理計算ソフトウェアです。

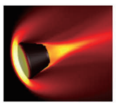


ナノ材料解析統合 GUI
Advance/NanoLabo
ナノ材料 プリポスト
材料解析ソフトウェア QuantumESPRESSO と LAMMPS に対応した総合 GUI です。

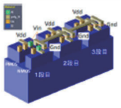
流体解析ソフトウェア
Advance/FrontFlow/red
流体
非圧縮性から圧縮性流れまで、広範囲で複雑な流れに対応した汎用３次元流体解析ソフトウェアです。



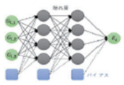
圧縮性流体解析ソルバー
Advance/FOCUS-i
流体
非構造格子に対応した圧縮性流体解析ソルバーです。特に遷音速や超音速の流れに適しており、高い並列化効率で計算出来ます。



大規模３次元 TCAD システム
Advance/TCAD
半導体デバイス GUI 付属
超微細半導体デバイスからパワーデバイスまで、高度な機能と使いやすい GUI を備えた３次元 TCAD システムです。



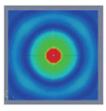
ニューラルネットワーク分子動力学システム
Advance/NeuralMD
ナノ材料 AI・機械学習
Neural Network Potential に基づいた分子動力学のソフトウェアです。第一原理計算の結果を教師データとして分子力場を作成します。



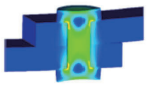
気液二相流解析ソフトウェア
Advance/FrontFlow/MP
流体
沸騰と凝縮を伴う気液二相流の流動特性や伝熱特性を３次元で解析するソフトウェアです。

管路系流体過渡解析ソフトウェア
Advance/FrontNet
流体 GUI 付属
配管や流体機器から成る管路系内流体に対する１次元過渡解析の実用的なソフトウェアです。

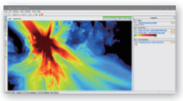
大規模電磁波解析ソフトウェア
Advance/ ParallelWave
光波・電磁波
マクスウェル方程式を FDTD 法で３次元的に解く電磁波解析ソフトウェアです。アンテナの電波解析から光の干渉や回折を考慮した光波解析まで幅広く適用できます。



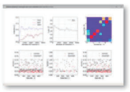
構造解析ソフトウェア
Advance/FrontSTR
構 造
固体の変形や熱伝導を、有限要素法を用いた３次元で解析するソフトウェアです。



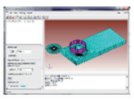
大気拡散影響予測システム
Advance/Emerg
流体 GUI 付属
大気拡散物質の挙動予測と影響評価のためのソフトウェアシステムです。



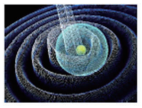
深層学習用ツール
Advance/ iMacle
AI・機械学習
機械学習のうち、ニューラルネットワークによる深層学習に特化、最小限度の機能に絞り込んだ比較的軽いツールです。



汎用プリポストプロセッサ
Advance/ REVOCAP
プリポスト
解析の一連の流れをスムーズに行う事を実現した汎用プリポストプロセッサです。



音響解析ソフトウェア
Advance/FrontNoise
音 響
環境騒音、機器内の共鳴等における音場を有限要素法を用いた３次元で解析するソフトウェアです。



自社による開発（国プロ含む）
開発チームによる質の高いサポートサービス
カスタマイズや機能追加も応相談
並列数無制限（追加料金なし）

ソフトウェアの解析事例

解析事例Webページをご覧ください。

アドバンスソフト　事例集　検索

http://case.advancesoft.jp

- ソフトウェア名からだけでなく、産業分野別、解析分野別の検索が可能となりました。
- 最新の事例を掲載しました。今後も逐次最新事例を紹介します。

産業分野別	解析分野別
自動車・運輸	流体
材料・化学	爆発・燃焼
産業機械	構造
航空宇宙	振動音響
エレクトロニクス	ナノ・バイオ
建設土木	プリポスト
原子力	半導体デバイス
エネルギー	光・電磁波
環境・防災	

facebook、YouTubeでも関連記事を掲載中

http://www.facebook.com/advancesoft.jp

http://www.youtube.com/user/advancesoft

【過去】2020/12/22開催のMBDセミナー

オープンソースソフトウェアを利用したモデル流通

	プログラム
①	**主催者あいさつ　アドバンスソフト株式会社のご紹介** アドバンスソフト株式会社　松澤　邦裕
②	**招待講演「自動車技術会でのFMI活用・展開活動の紹介」** TRA-AD　平野　豊　様
③	**MBDによる自動車設計が本格化するに伴って必要となる企業間のモデル流通の課題** アドバンスソフト株式会社　加藤 廣
④	**MBDプラットフォーム構築のためのOpenModelicaの活用に関する課題と取り組み** アドバンスソフト株式会社 佐藤　甫
⑤	**関連サービスのご紹介** アドバンスソフト株式会社　田口 浩一

※このMBDセミナーの講演資料は、弊社webページに御座います「シミュレーション図書館」で公開されております。ご興味のある方は、ユーザ登録をして頂き、ダウンロードをお願い致します。

【過去】2020/12/22開催のMBDセミナー

オープンソースソフトウェアを利用したモデル流通

- **モデルベース開発（MBD）の先に・・・**
  - ここ10年以内には「モデル流通」の時代
  - 「モデル流通」によるものづくりが世界標準になる
- **モデル流通 とは・・・**
  - 「モデル」を、企業内だけでなく、企業間も含め、産業構造全体で活用し、ものづくりの効率化に取り組むこと
- **モデル流通の課題**
  - 「モデル」は、機密情報やノウハウの塊
  - 異なるツール間・モデル間での連携の難しさ（FMIの活用）
  - 産業構造全体への普及（人材教育）
- **オープンソース・ソフトウェア（OSS）の活用**
  - OpenModelocaによる開発サポートや受託開発、AI技術の活用

次の講演

OpenModelica の実践的活用と展望（機械学習とモデル流通）

講演項目：

① 可搬性の高いモデルを作成するためのコーディング規約

② FMIモデル接続の解説と、トラブル解決へのヒント

③ 機械学習プログラムとの連携によるパラメータ同定の事例

※ 上記講演のあとに10分程度の質疑応答を予定しております。

※ 質問の方法 ※質問の前にご所属とお名前をお願いします。

- 講演中でもQ&A機能でご質問下さい。質疑応答の際に優先的に回答します。
- 口頭で質問がある方は、講演中でも予め挙手機能で手を挙げてください。順番に口頭でご質問をいただきます。

OpenModelica の実践的活用と展望（機械学習とモデル流通）

第5事業部　佐藤 甫

OpenModelica の実践的活用と展望（機械学習とモデル流通）
2021年 7月 9日（金）
アドバンスソフト株式会社

OpenModelica の実践的活用と展望（機械学習とモデル流通）

講演項目：

① **可搬性の高いモデルを作成するためのコーディング規約**

② **FMIモデル接続の解説と、トラブル解決へのヒント**

③ **機械学習プログラムとの連携によるパラメータ同定の事例**

講演者：　第5事業部　研究員　佐藤 甫

概要

- OpenModelicaによるモデル開発課題とは？
- Modelica標準ライブラリの実装を参考に、
  可搬性の高いコーディング規約を採用して
  開発課題を解決する。

Modelica標準ライブラリとは？

OpenModelicaの特長

- オープンソース・ソフトウェア（OSS）
  - 無料である
  - ソースが公開されている
  - 利用コストを開発コストに転移できる
  - 実装や計算方法を、検証可能である

- モデルシミュレーションに特化
  - プログラミング自体ではなく、モデルの記述に集中できる

OpenModelica における開発課題

- ソフトウェア開発の難しさ
  - 開発者の人数？　スキルは？
  - 開発のゴールはどこか？　仕様は明確か？

- Modelica言語の学習コスト
  - 独学では難しい。良いテキストがあるか？

- OpenModelica環境の変化 … バージョン対応
  - 結局、どのバージョンを使えばいいのか？
  - 作成者と利用者の環境をどう揃えるか

ソフトウェア開発の一般的な課題

- 複数人で足並みを揃えて開発する難しさ
  - 作業内容の分担
  - スキルのばらつき
  - 考え方の違い

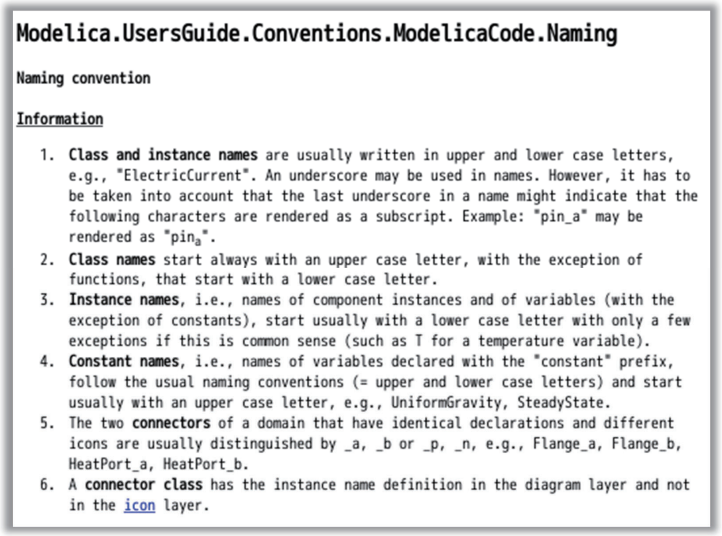
- コーディング規約　足並みを揃えるルール
  - 構成のルール（どこにあるかがすぐわかるように）
  - 名前のルール（自分がわかればいい、ではだめ）
  - 機能の推奨/非推奨

コーディング規約を決める難しさ

- コーディング規約を決められる時点で、実力者
  - 具体的で実効的なルールには、根拠が必要
  - 現場の経験がなければ規約ができない
  - 規約がなければ開発できない

- 標準ライブラリのコーディング規約を叩き台に
  - 標準ライブラリの理解が深まる
  - 標準ライブラリは、高品質な設計の手本になる

Modelica標準ライブラリ命名規則



Modelica.UsersGuide.Conventions.ModelicaCode.Naming

Naming convention

Information

1. **Class and instance names** are usually written in upper and lower case letters, e.g., "ElectricCurrent". An underscore may be used in names. However, it has to be taken into account that the last underscore in a name might indicate that the following characters are rendered as a subscript. Example: "pin_a" may be rendered as "pin_a".
2. **Class names** start always with an upper case letter, with the exception of functions, that start with a lower case letter.
3. **Instance names**, i.e., names of component instances and of variables (with the exception of constants), start usually with a lower case letter with only a few exceptions if this is common sense (such as T for a temperature variable).
4. **Constant names**, i.e., names of variables declared with the "constant" prefix, follow the usual naming conventions (= upper and lower case letters) and start usually with an upper case letter, e.g., UniformGravity, SteadyState.
5. The two **connectors** of a domain that have identical declarations and different icons are usually distinguished by _a, _b or _p, _n, e.g., Flange_a, Flange_b, HeatPort_a, HeatPort_b.
6. A **connector class** has the instance name definition in the diagram layer and not in the icon layer.

1. 単語に関する規則
   (大文字・小文字・区切り文字)
2. モデル名と関数名に関する規則
3. 個々の部品の名前に関する規則
4. 定数の名前に関する規則
5. コネクタ名に関する規則
6. コネクタ名に関する詳細な規則

【参考】https://doc.modelica.org/om/Modelica.UsersGuide.Conventions.ModelicaCode.Naming.html

Modelica標準ライブラリにおける単語のルール

- 単語は、英数字と“_”からなる。

○	×
Dog1	犬1
position_start	position-start

- 最後の“_”以降は、添字として理解できるとよい

○	△	×	備考
position_start	startPosition	start_position	$position_{start}$
mass_1, mass_2, …	mass1, mass2, …		$mass_1, mass_2, \ldots$

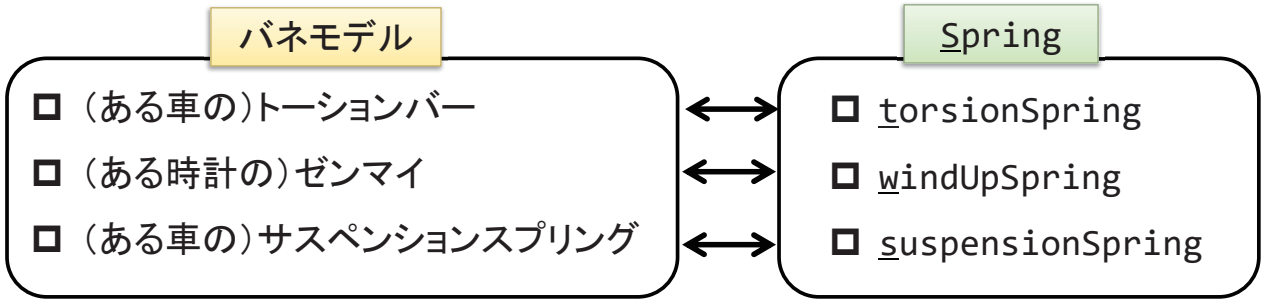
Modelica標準ライブラリにおける単語のルール

- Modelica標準ライブラリは、物理学における量記号で単語を省略する。
- 必要に応じて追加してコーディング規約に追加する。

Variable	Quantity	Variable	Quantity	Variable	Quantity
a	acceleration	HFlow	enthalpyFlow	r	radius
A	area	i	current	R	radius, resistance
C	capacitance	J	inertia	t	time
d	damping, density, diameter	l	length	T	temperature
dp	pressureDrop	L	Inductance	tau	torque
e	specificEntropy	m	mass	U	internalEnergy
E	energy, entropy	M	mutualInductance	v	electricPotential, specificVolume, velocity, voltage
eta	efficiency	mFlow	massFlow	V	volume
f	force, frequency	P	power	w	angularVelocity
G	conductance	p	pressure	X	reactance
H	enthalpy	Q	heat	Z	impedance
h	height, specificEnthalpy	Qflow	heatFlow		

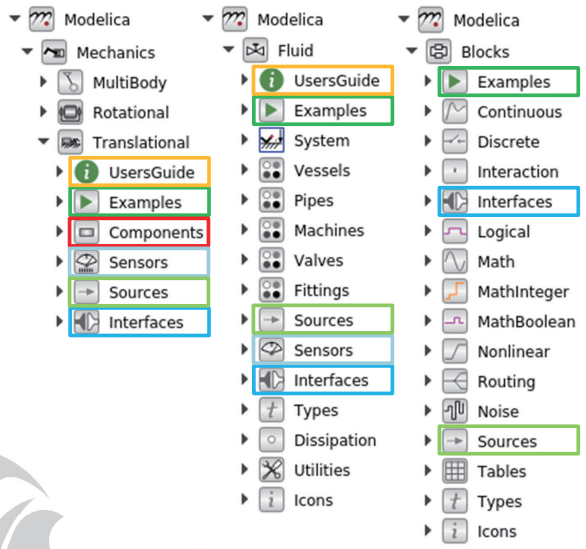
Modelica標準ライブラリにおけるモデル名と部品名のルール

- モデル名と、個々の部品名は、語頭が大文字か小文字で区別



Modelica標準ライブラリ構成の不文律

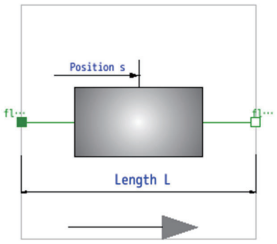
- 標準ライブラリの各物理ドメインで、共通する構成

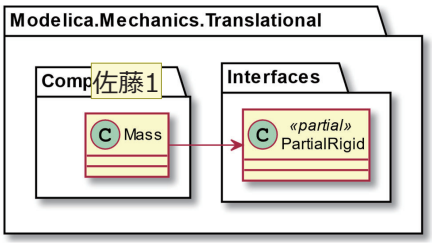


名前	役割
Examples	動作確認用モデル
UserGuide	モデル解説ドキュメント
Interfaces	入出力仕様※
Components	実際に使うモデル
Sources	境界条件
Sensors	物理量を数値で読み取る

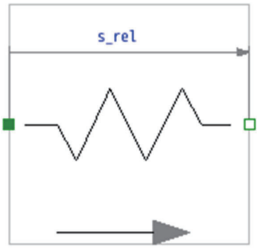
Modelicaにおけるインターフェース①

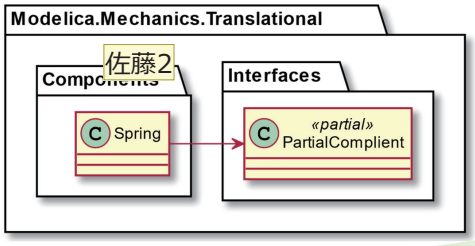
- 「質量要素」の部品モデルは、「PartialRigid」インターフェースを継承





- 「ばね要素」の部品モデルは、「PartialCompliant」インターフェースを継承





Modelicaにおけるインターフェース②

- 「PartialRigid」両端の距離L[m]が一定

  ⇒ 左と右が固定されている状態

- 「PartialComplient」両端の距離s_rel[m]が変化

  ⇒ 左右の間隔が伸び縮みできる。

- 共通の性質をくくりだす

  コネクタ定義を一箇所にまとめ、接続法を統一する。

インターフェースの活用

- 標準ライブラリとの互換性を持つモデルを作成するためにインターフェースを活用する。

- モデルの設計

  1. MSLのインターフェースを把握しておく。
  2. 作りたいモデルのコネクタ構成を決める。
  3. コネクタ構成と対応するインターフェースを探す。
  4. インターフェースの前提条件を確認。
  5. インターフェースを継承する。
  6. 必要な方程式を実装する。

まとめ

① 可搬性の高いモデルを作成するためのコーディング規約

- コーディング規約は、標準ライブラリのコーディング規約を叩き台にして決めるのが良い。
- 標準ライブラリとの互換性を持つモデルを作成するために インターフェースを継承の起点として活用する。

OpenModelica の実践的活用と展望（機械学習とモデル流通）

講演項目：

① 可搬性の高いモデルを作成するためのコーディング規約

② FMIモデル接続の解説と、トラブル解決へのヒント

③ 機械学習プログラムとの連携によるパラメータ同定の事例

講演者： 第5事業部 研究員 佐藤 甫

概要

- 「SURIAWASE2.0」におけるモデル流通では、
  モデルの受け渡しフォーマットにFMI規格を利用している。
- FMI規格によるモデル開発の利便性・簡便性のため
  FMI規格書に準拠した、fmi2++ライブラリを新規開発した。
- 開発過程で得た、FMI規格の重要なポイントを紹介する。

FMI規格とは？　.fmuファイルとは？

- FMIとは規格の名称であり
  様々なモデルプラットフォームでモデルを流通できる。
- 個々のモデルは.fmuファイルとして実装される。
- FMI規格は2つの実行方法を定義している。
  - **M**odel**E**xchange （**ME**）
  - **C**o-**S**imulation (**CS**)
- 本講演のターゲットは、MEの.fmuファイルの作成

.fmuファイル作成の課題

- MBDツールで作った.fmuファイルが動かない？
- FMI規格におけるME実行のプロセスと、
  .fmuファイルの実装について知る必要がある。
- 原因解決とFMI規格の理解のために、FMI規格書を読む。

  ⇒本公演では、その要点をお話します。

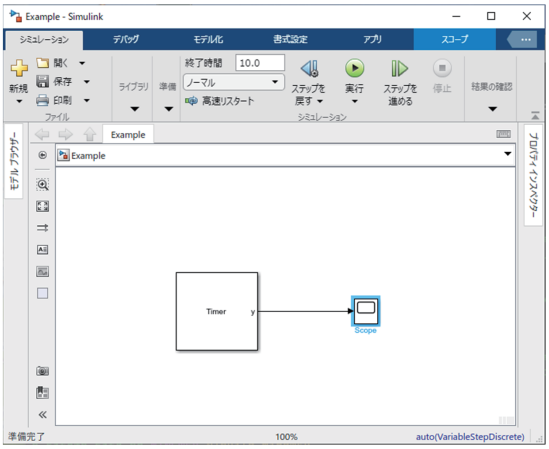
【参考】FMI規格 https://fmi-standard.org/

fmi2++ ライブラリの開発

- FMI規格書から、弊社で新規開発したC++ライブラリ
- FMI規格に準拠した、MEの.fmuファイルを
  C/C++言語で実装することが可能。
- C/C++コードを、無料で.fmuファイルにできる！
- クロスプラットフォームで利用可能。
  - Windows10 | Linux
    - 無償C++コンパイラ(gcc, clang)
    - 有償C++コンパイラ (Visual Studio)

デモンストレーション

- fmi2++ライブラリの利用例
  1. C++コードと.xmlファイルを用意
  2. ビルドし、.fmuファイルの生成
  3. Timer.fmuの動作方法




```
Timer.fmu
├── binaries
│   └── win64
│       └── Timer.dll
└── modelDescription.xml

2 directories, 2 files
```

fmi2++ライブラリで実装する

- fmi2++ライブラリの基本クラスから経過時間を取得。
  出力変数に書き込む。

C++コード

```
double
component::get_y()
{
  return get_time().r();
}
```

xmlモデル記述

```
<ScalarVariable
  name="y" valueReference="0"
  causality="output"
>
  <Real/>
</ScalarVariable>
```


FMI規格の基礎知識：API関数

- MEの.fmuファイルには
  右の表で示したAPIで必須なものをすべて実装
- fmi2++ライブラリがデフォルト実装を提供
  ⇒ 何もしないと、空っぽの.fmuファイルになる
- 関数のリスト
  - CSと共通な必須の関数が21個
  - CSと共通なオプションの関数が4個
  - MEで必須な 10個
  - (CSで必須 な9個)

Function	FMI 2.0 for Model Exchange							
	start, end	Instantiated	Initialization Mode	Event Mode	Continuous-Time Mode	terminated	error	fatal
fmi2GetTypesPlatform	x	x	x	x	x	x	x	
fmi2GetVersion	x	x	x	x	x	x	x	
fmi2SetDebugLogging		x	x	x	x	x	x	
fmi2Instantiate	x							
fmi2FreeInstance		x	x	x	x	x	x	
fmi2SetupExperiment		x						
fmi2EnterInitializationMode		x						
fmi2ExitInitializationMode			x					
fmi2Terminate				x	x			
fmi2Reset		x	x	x	x	x	x	
fmi2GetReal			2	x	x	x	7	
fmi2GetInteger			2	x	x	x	7	
fmi2GetBoolean			2	x	x	x	7	
fmi2GetString			2	x	x	x	7	
fmi2SetReal		1	3	4	5			
fmi2SetInteger		1	3	4				
fmi2SetBoolean		1	3	4				
fmi2SetString		1	3	4				
fmi2GetFMUstate		x	x	x	x	x	7	
fmi2SetFMUstate		x	x	x	x	x	x	
fmi2FreeFMUstate		x	x	x	x	x	x	
fmi2SerializedFMUstateSize		x	x	x	x	x	x	
fmi2SerializeFMUstate		x	x	x	x	x	x	
fmi2DeSerializeFMUstate		x	x	x	x	x	x	
fmi2GetDirectionalDerivative			x	x	x	x	7	
fmi2EnterEventMode				x	x			
fmi2NewDiscreteStates				x				
fmi2EnterContinuousTimeMode				x				
fmi2CompletedIntegratorStep					x			
fmi2SetTime				8	x			
fmi2SetContinuousStates					x			
fmi2GetEventIndicators			x	x	x	x	7	
fmi2GetContinuousStates			x	x	x	x	7	
fmi2GetDerivatives			x	x	x	x	7	
fmi2GetNominalsOfContinuousStates		x		x	x	x	7	

21
4
10

fmi2++ライブラリでAPIを実装

- Timer.fmuは、現在時刻を出力するために `fmi2GetReal` 関数だけを上書きしている。
- 残りのAPIは、デフォルト実装のまま
- 使いたい機能に対応するAPIを、書き足すだけ

FMI規格の基礎知識 状態遷移

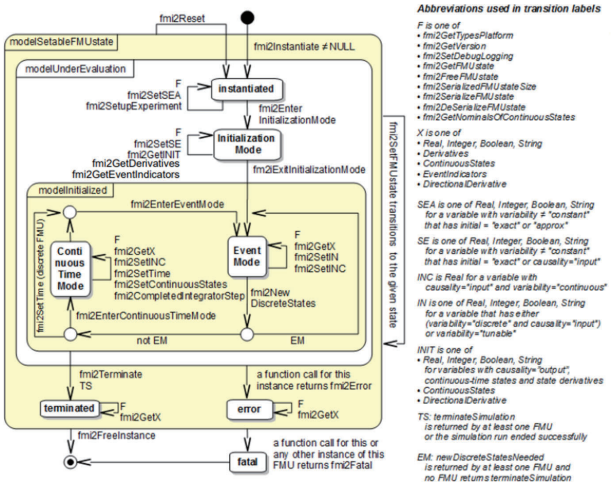
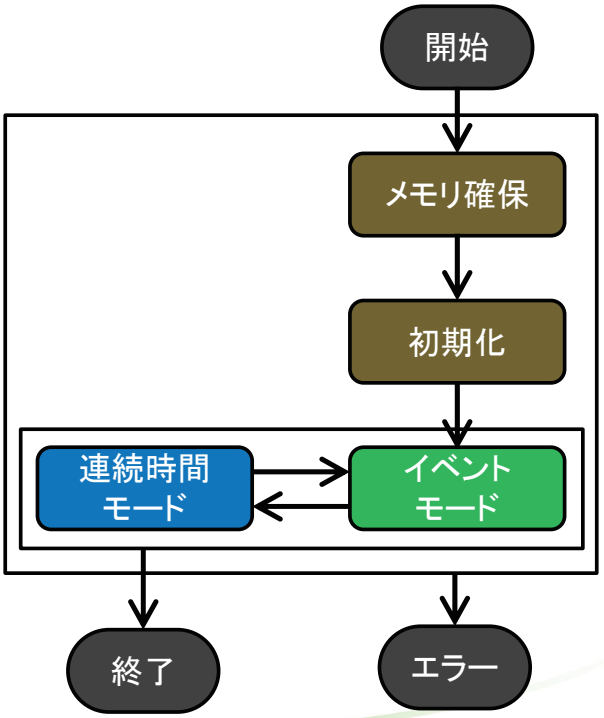
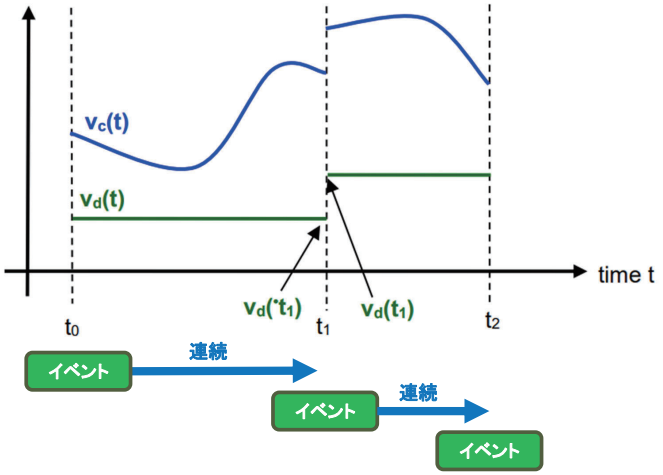


Figure 6: Calling sequence of Model Exchange C functions in form of an UML 2.0 state machine.



FMI規格 MEの計算モード

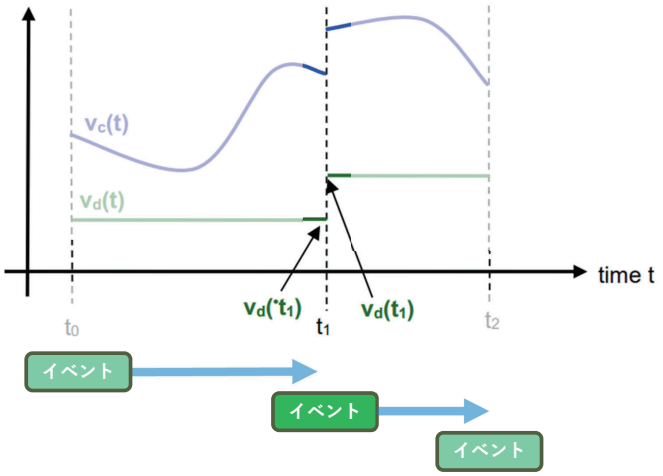
- MEの計算ステップは、2つのモードを往復
  - イベントモード （離散変数を更新できる）
  - 連続時間モード （連続変数の積分）



イベントモードの計算内容

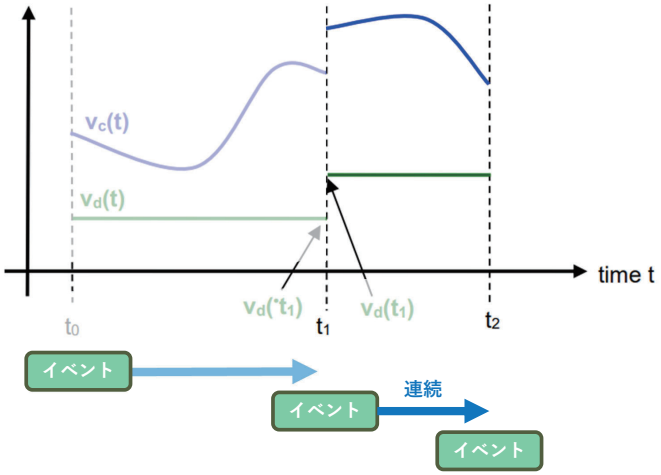
- イベントモードは、ある瞬間での変化

  時刻はt1で変化せず、イベント前後の変化を計算



連続時間モードの計算内容

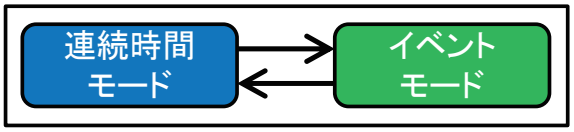
- 連続時間モードは時間発展を計算
  - (t2, t2) 不連続点を含まない区間を扱う。
  - 離散変数は変更されず、定数になる。

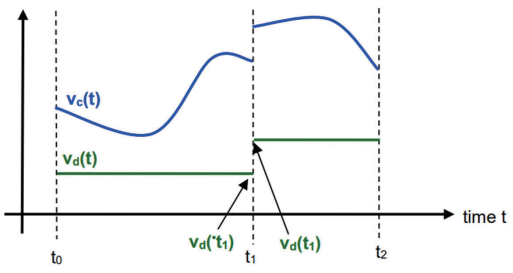


FMI規格　MEの特長

- イベントモードと連続時間モードの往復による
  連続と不連続の取り扱いの明確化。

  ⇒ Piecewise Continuous Time System

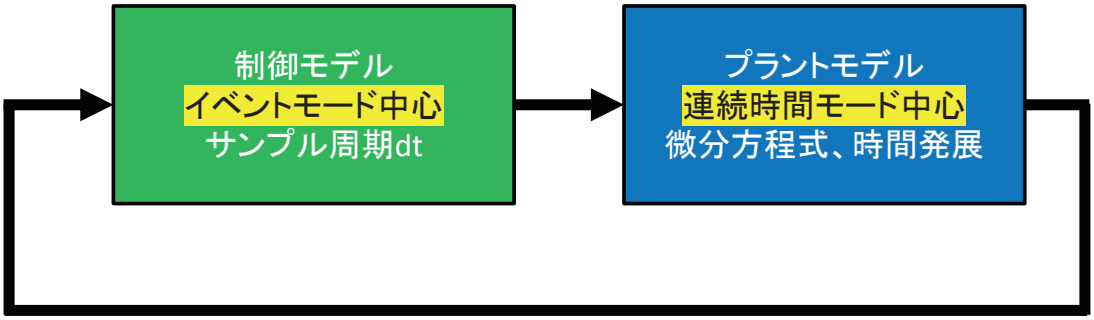




FMI規格を踏まえた設計

- Piecewise Continuous Time System
- 連続と不連続を正確に扱える。
  - しかし、一つのモデルに両方実装するのは難しい。

  ⇒ 役割を分離し片方を実装する設計を意識



まとめ

② FMIモデル接続の解説と、トラブル解決へのヒント

- .fmuファイルの作成のため、FMI規格の実行プロセスと.fmuファイルの実装法について知る必要がある。
- トラブル発生時は、FMI規格における状態遷移やAPI関数などの情報を頼りに検討すると良い。
- 弊社が開発したfmi2++ ライブラリを利用して、MEの.fmuファイルを作成する例を紹介した。

OpenModelica の実践的活用と展望（機械学習とモデル流通）

講演項目：

① 可搬性の高いモデルを作成するためのコーディング規約

② FMIモデル接続の解説と、トラブル解決へのヒント

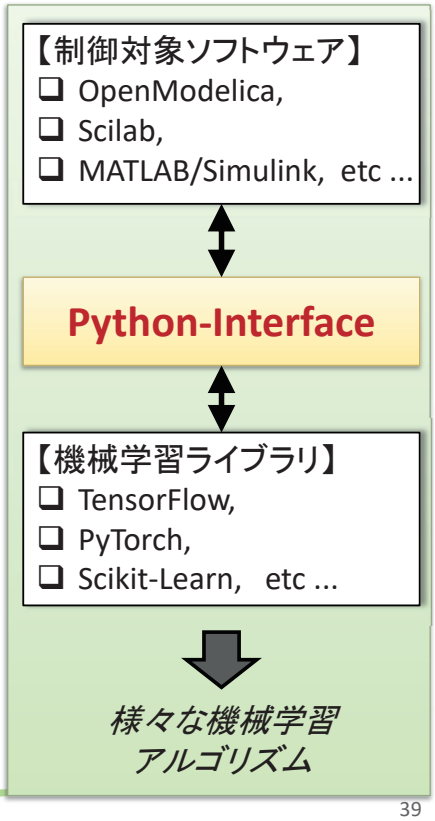
③ 機械学習プログラムとの連携によるパラメータ同定の事例

講演者： 第5事業部 事業部長 松澤 邦裕

機械学習プログラムとの連携によるパラメータ同定の事例

概要

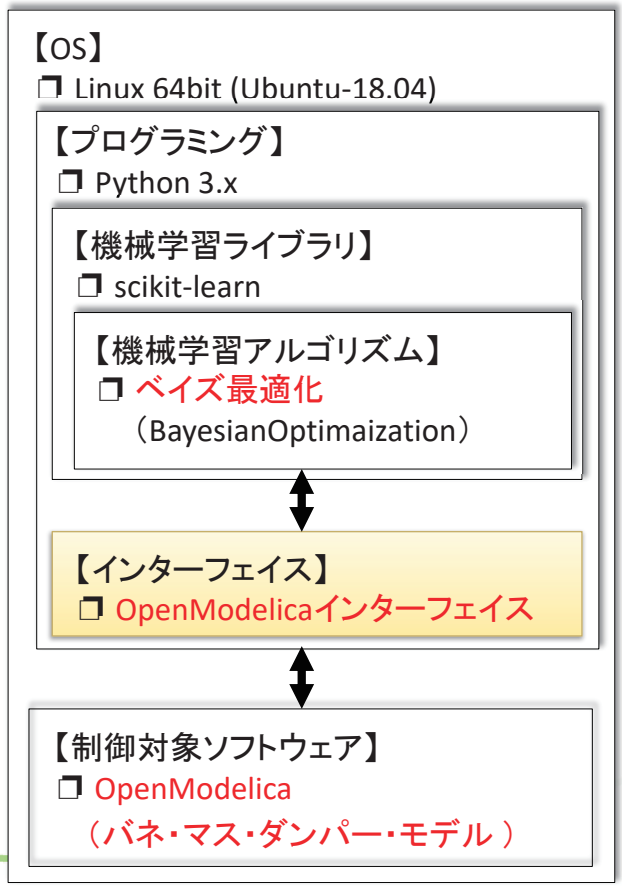
- 「OpenModelica のシミュレーション機能と機械学習アルゴリズムの連携」という新たな枠組みを開発
- OpenModelica のプロセスを機械学習プログラムから呼び出し、シミュレーションの外部制御やシミュレーション結果の外部保存、モデルパラメータへの読み書きなどを可能にする Python インターフェイスを開発（スクリプトで呼び出し可能）
- PyTorch や TensorFlow、scikit-learn などの機械学習ライブラリを組み合わせることで、ベイズ最適化や強化学習を利用した設計・解析の効率化などが期待できる。



機械学習プログラムとの連携によるパラメータ同定の事例

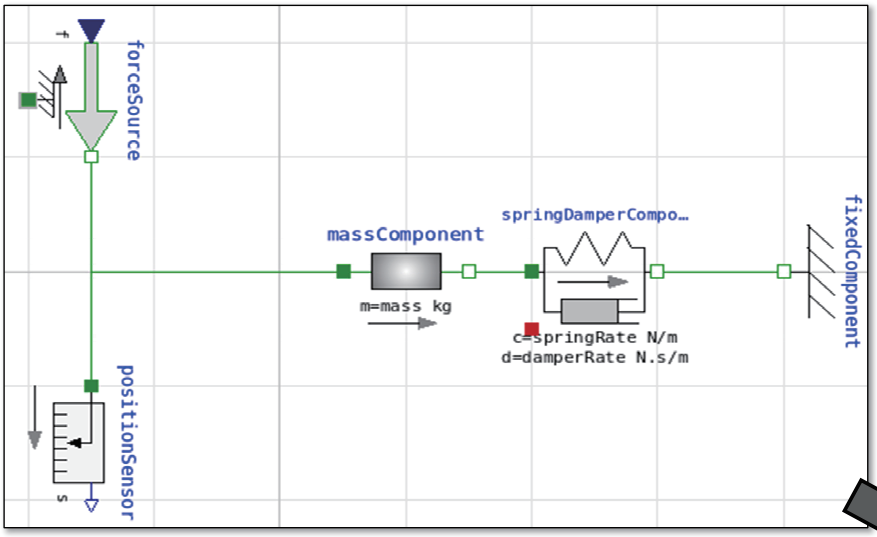
適用テスト

- 機械学習アルゴリズムとしてベイズ最適化を採用する。
- 対象モデルは、OpenModelicaで記述した単純なバネ・マス・ダンパーモデルとする。
- 予め対象モデルでシミュレーションを実施し、ダミー実測データを準備する。
- ベイズ最適化を用いて、ダミー実測データに対し、バネ定数とダンパー定数のパラメータ同定を行う。

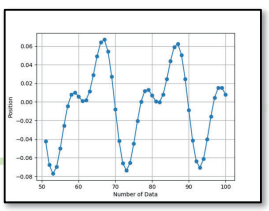


対象モデル

- OpenModelicaでバネ・マス・ダンパーモデルを作成した。
- シミュレーションにより物体（massComponent）の位置の時間変化を得る。



シミュレーション実行

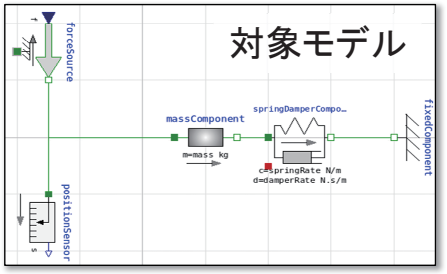


ダミー実測データの作成

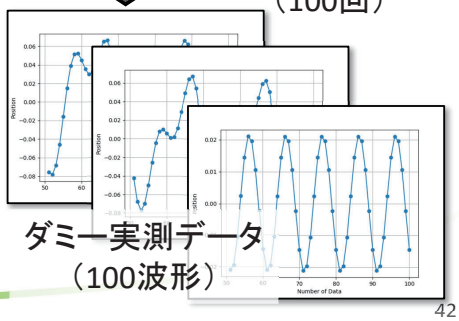
- 実測データの代わりに、OpenModelicaシミュレーションを実行して
  ダミー実測データ を作成する。
- 最適化パラメータは、バネ定数とダンパー定数である。

シミュレーション条件

- ❑ バネ定数　　：2.0 [N/m]　　【固定】
- ❑ ダンパー定数：2.0 [N・s/m]【固定】
- ❑ 物体の質量　：1.0 [kg]　　【固定】
- ❑ 外力の周期　：1.0 [Hz]　　【固定】
- ❑ 外力の強度　：ランダム [N]
  （中央値1.0，分散0.1の正規分布）
  ※ 実測データには測定誤差や
  ノイズなどが含まれることを想定



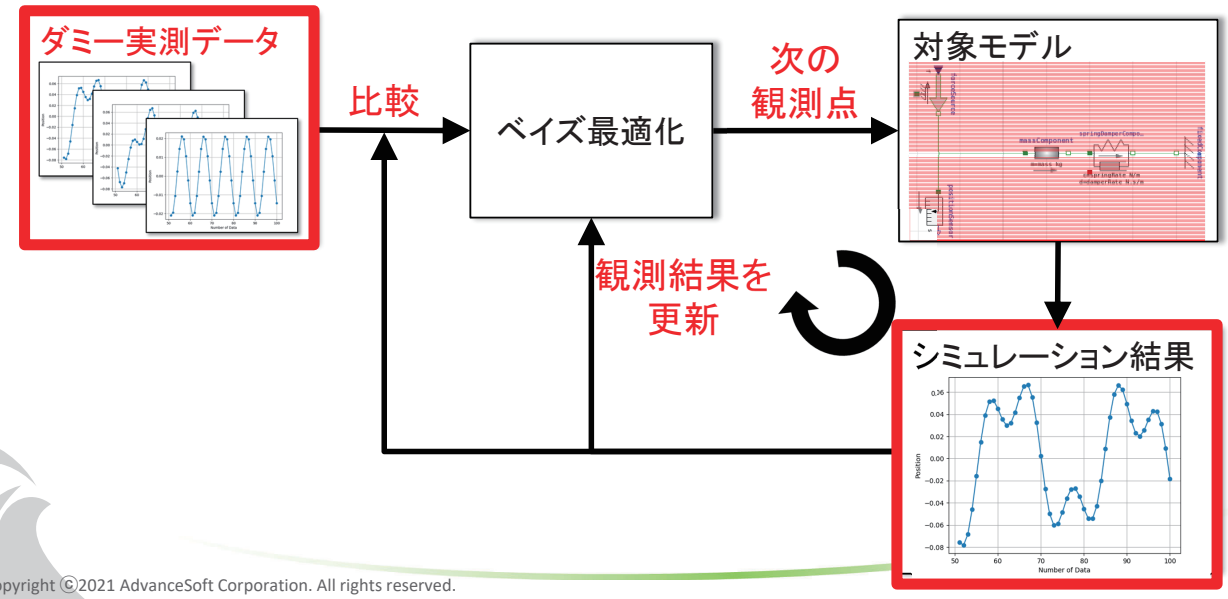
シミュレーション
（100回）



ダミー実測データ
（100波形）

適用テストの概要（ベイズ最適化）

- ダミー実測データ と シミュレーション結果 が一致するように、ベイズ最適化を
  用いて バネ定数 と ダンパー定数 を同定する。
- 最適化パラメータは、バネ定数とダンパー定数である。



ベイズ最適化の大まかな適用フロー

① 最初にランダムに２点 の観測を行う（初期観測）

② ①の測定結果から確率分布を求め、次の測定点
（バネ定数とダンパー定数）を決定

③ ②で決定したバネ定数とダンパー定数を指定して
OpenModelicaのシミュレーションを実行
（計算時間間隔：0.1 sec.、計算時間：0 — 10 sec.）

④ 物体（massComponent）の位置の時間変化を取得

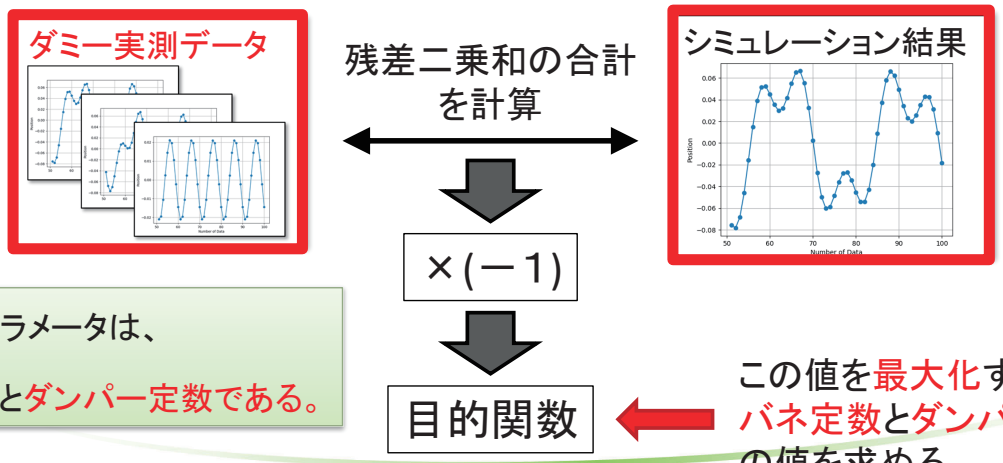
⑤ ダミー実測データ（100波形）とシミュレーション結果の
残差二乗和の合計を求め、その値に －１ を乗じて
目的関数の値を算出

⑥ 上記を256回繰り返し、より目的関数の値が高い
バネ定数とダンパー定数を探索

目的関数

❑ ベイズ最適化において次の観測点を求める指標

❑ 適用テストの目的関数：
ダミー実測データ（100波形）とシミュレーション結果の
残差二乗和の合計を計算し、その符号を負にした値
（一致しているほど値は0に近くなる。完全一致は0になる。）



最適化パラメータは、
バネ定数とダンパー定数である。

実行環境

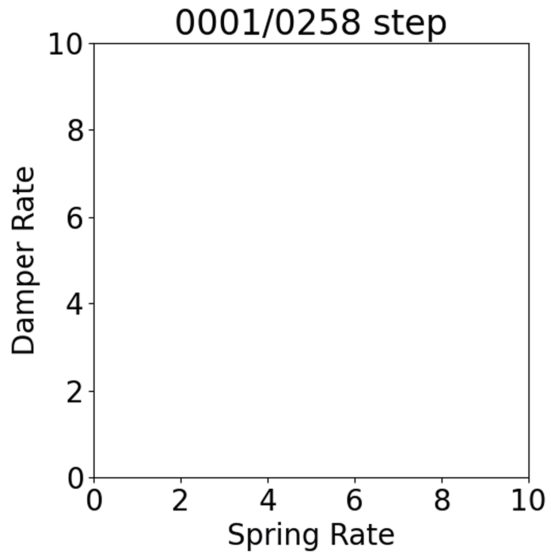
デスクトップで普段使いも兼ねる安価なワークステーション

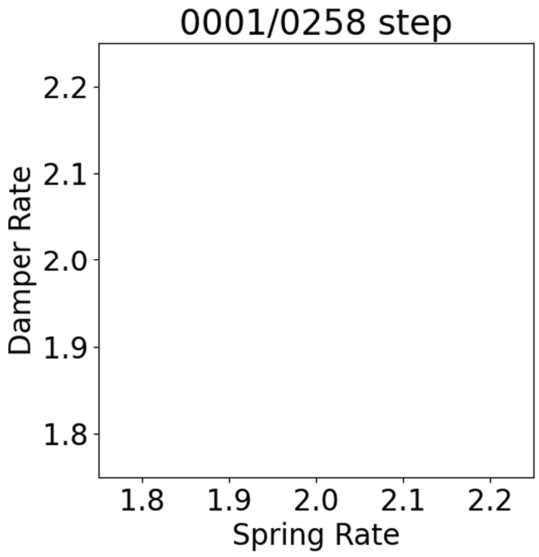
- OS： Windows Subsystem for Linux (Ubuntu-18.04)
  @ Windows 10 Professional 64bit
- CPU： Intel Core i7-8700 CPU @ 3.20GHz
- メモリ： 32GB
- GPU：NVIDIA GeForce GTX 1050 Ti（適用テストでは未使用）

- 適用テストの所要時間： 約２分

適用テストの結果①

- ベイズ最適化を用いた結果、ダミー実測データ作成時の真値である
  バネ定数 ＝ 2.0、ダンパー定数 ＝ 2.0 の近傍が重点的に探索される。

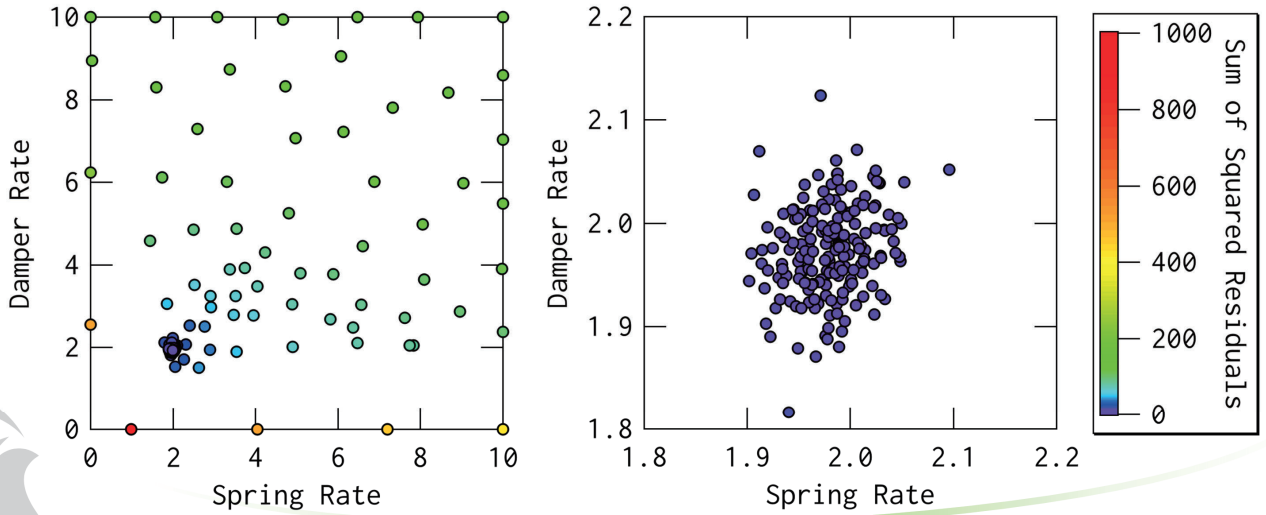




適用テストの結果②

- ベイズ最適化を用いた結果、ダミー実測データ作成時の真値である
  バネ定数 ＝ 2.0、ダンパー定数 ＝ 2.0 の近傍が重点的に探索される。

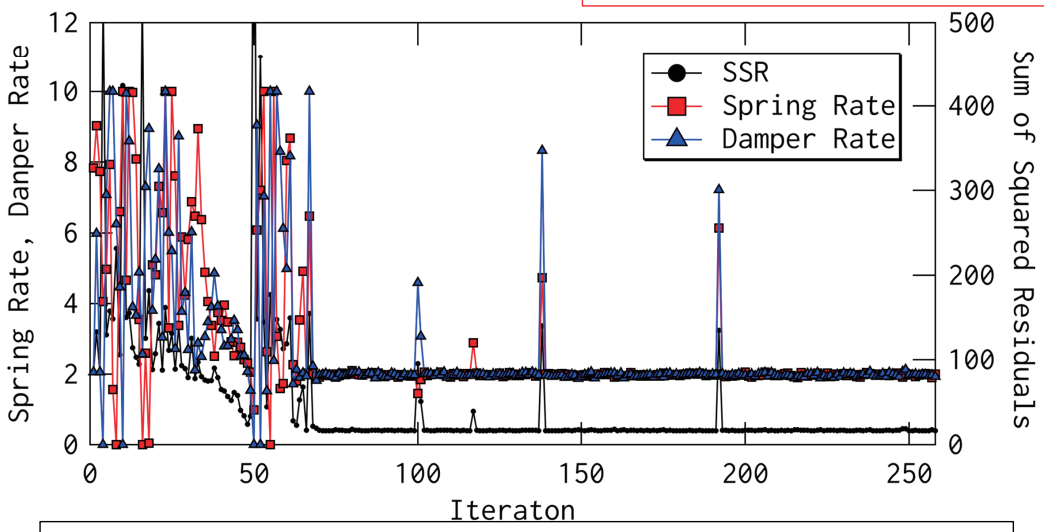
【注意】カラーマップは数値が低いほど良い。



適用テストの結果③

- 初期は、広い範囲のバネ定数とダンパー定数を探索
- 繰り返し回数 70 回 以降は、バネ定数 ＝ 2.0、ダンパー定数 ＝ 2.0 の近傍を重点的に探索

【注意】右縦軸は0に近いほど良い。



バネ定数、ダンパー定数、および残差二乗和（合計）の推移

まとめ

③ 機械学習プログラムとの連携によるパラメータ同定の事例

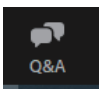
- 「OpenModelica のシミュレーション機能と機械学習アルゴリズムの連携」という汎用性の高い新たな枠組みを開発し、適用テストを行った。
- 適用テストでは、機械学習アルゴリズムとしてベイズ最適化 を採用し、対象モデルは、OpenModelicaで記述した単純な バネ・マス・ダンパーモデル とした。

ダミー実測データ	ベイズ最適化（258 step）
❑ バネ定数： 2.0 [N/m] 【固定】 ❑ ダンパー定数： 2.0 [N・s/m]【固定】	❑ バネ定数： 1.98593 [N/m] ❑ ダンパー定数： 1.97402 [N・s/m]

- ベイズ最適化で求めた値は、ダミー実測データ作成時の値と良く一致した。

只今ご質問を受け付けております

- Q&A機能、マイクを使った質疑応答を受け付けております。
- マイクでのご質問をご希望される方は「手を挙げる」でお知らせください。



Q&A
講演中でもご質問を受け付けております。

手を挙げる
質疑応答の際にマイクでのご質問を希望される方はクリックしてください。

手を降ろす
ご質問いただきましたら、クリックして手を降ろしてください。

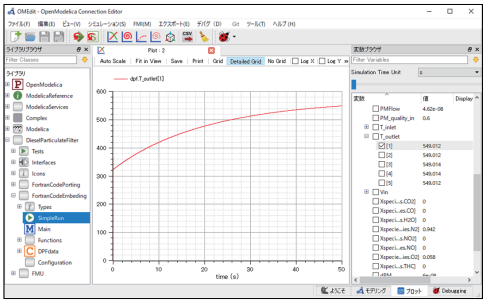
[OpenModelicaサポートサービスと関連サービスのご紹介]

営業部　田口　浩一

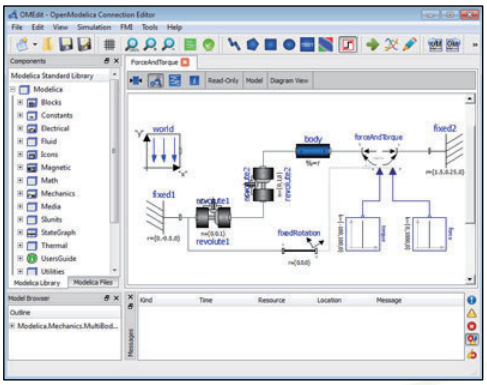
オンラインセミナー「OpenModelica の実践的活用と展望
（機械学習とモデル流通）」
2021年7月9日（金）
アドバンスソフト株式会社

OpenModelica のようなオープンソース・ソフトウェア（OSS）は、ソースが公開されており、無償で商用利用が可能なため、コストダウンが期待できる一方で、サポートがないことで業務が進まなくなる懸念があり、導入が避けられるという傾向があります。アドバンスソフトでは、秘密保持契約を結んだ上で、お客様のサポート依頼に対して具体的な回答を行います。OpenModelica や OSS 全般に関連する技術サポートをご検討中のお客様は、是非ともアドバンスソフトにご相談ください

サポートの内容



- **サポートサービスの対象例**
  - OpenModelica の使用法に関する説明
  - Modelica 言語の文法に関する説明
  - 計算実行時のエラーなどの問題改善に対する技術サポート
  - モデル作成や計算結果の振る舞いに対する技術サポート



- **講習会の実施 (年間契約のみ)**
  - OpenModelica に関連する内容の講習会を実施します。
  - 内容はお客様とのご相談により決定します。
    (※3時間程度を想定しています。)
- **お問合せの方法**
  - E-mail による問合せにご対応いたします。

サポートの種別

質問件数と工数には契約種別および契約期間ごとに下表のような上限があります。

契約種別	年間契約 (1年間)			月間契約 (1カ月間)		
	質問件数	工数	価格	質問件数	工数	価格
ゴールド	50件 (*1)	25人日	(*2)	5件	2.5人日	(*2)
シルバー	20件 (*1)	10人日	(*2)	—	—	—

(*1) 月間件数は5件までとします。

- **サポートサービスの上限に到達した場合**
  - 契約期間の満了、または質問件数もしくは工数が上限に到達した場合は、契約は終了となります。
  - さらにサポートサービスの継続が必要な場合は、新規にサポートサービス契約を締結する必要があります。
- **工数がかかることが予想されるご依頼の場合**
  - 実用的なモデル開発や定量的な合わせ込みなど、あらかじめ工数がかかることが予想されるご依頼の場合は、受託開発でのご発注をお勧めすることがあります。

アドバンスソフトの開発・解析サービス

お客さまのご要望に応じて科学技術計算ソフトウェアの新規開発、機能追加、受託解析等のサービスをおこないます。

1. 流体・構造・ナノ関連など幅広い分野のソフトウェアを開発し、解析経験がある技術者がお客様のご要望をお伺いいたします。

2. 最適な解析方法をご提案いたします。

3. お客さまのご了解が得られましたら、モデリングを行い、解析を実施いたします。

4. 解析結果を可視化し、解析結果の評価や考察を行なって報告書を作成いたします。

MBD関連の受託業務例

[1] **MBD統合プラットフォームの構築支援**
FMU （Functional Mockup Unit）エクスポート機能を持つ既存のシミュレーションソフトウェアや，開発したコンポーネントモジュールを統合して，モジュール毎に差し替え可能なMDB統合プラットフォームの構築を行います．統合プラットフォームには，MATLAB/Simulink†1やOpenModelicaを使用します．

[2] **既存プログラムのFMUモジュールへの移植作業**
既存プログラム（言語は問わず）からFMUモジュールへの移植を行います．移植に際して，入出力ファイルの定義やグローバル変数の取り扱い，ログ出力の管理などを考慮しつつ，OpenModelicaのOMEdit機能を利用して，Model-ExchangeまたはCo-SimulationのFMUモジュールを作成します．

機械学習関連

MBDに限らずあらゆる分野全般において、検討しております。
制御の最適化・自動化・省力化、異常検知、情報予測、定常的な
監視データ取得、実験や実測データ等を研究開発に利用検討の際に
機械学習による課題解決をご検討でしたら是非ともお声がけください。

ご清聴ありがとうございました。

アドバンスソフトは、高度な技術力、開発力、人材を武器に、最先端理論を応用した解析シミュレーションソフトウェアを開発・販売しています。受託解析、受託開発、パッケージソフトウェア、コンサルティング等多様なソリューションを通じて、お客様の問題解決に即戦力として貢献します。

お問い合わせ先：ご担当営業まで
TEL:03-6826-3971 FAX:03-5283-6580
E-mail:office@advancesoft.jp