

OtFMI, an OpenURNS module for uncertainties analysis with 0D/1D system models

Michaël Baudin ¹ Audrey Jardin ¹ Mathias Bouquerel ¹
Anne-Laure Popelin ¹ Audrey Jardin ¹
Julien Schueller ² Sylvain Girard ²

¹EDF R&D. 6, quai Watier, 78401, Chatou Cedex - France, michael.baudin@edf.fr

²Phimeca Engineering. 18/20 boulevard de Reuilly, 75012 Paris - France,
girard@phimeca.com

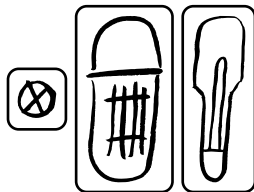
October 19th 2017

Industrial issue

- ▶ EDF uses 0D/1D system models programmed in Modelica as decision support for the conception and operation of its industrial assets.
- ▶ How to apply OpenTURNS' panoply of methods to these models?

“Regular” models vs 0D/1D system models

“Regular”
modelling



“Packing”

$$\begin{cases} \dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t), t) \\ \mathbf{y}(t) = g(\mathbf{x}(t), \mathbf{u}(t), t) \end{cases}$$

Equation formulation



Solver
programming

"Regular" models vs 0D/1D system models

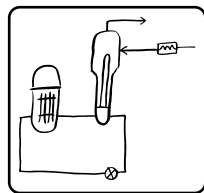
"Regular"
modelling



0D/1D
system modelling



"Packing"



Equation formulation

$$\begin{cases} \dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t), t) \\ \mathbf{y}(t) = g(\mathbf{x}(t), \mathbf{u}(t), t) \end{cases}$$

$$\begin{cases} \dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t), t) \\ \mathbf{y}(t) = g(\mathbf{x}(t), \mathbf{u}(t), t) \end{cases}$$



Solver
programming



“Regular” models vs 0D/1D system models

“Regular”
modelling



0D/1D
system modelling

Long (minute → hour)

CPU time

Short (second → minute)

2D, 3D

Dimensions

0D, 1D

Discrete versioning

Development rate

Continuous mutation

Numerical analyst, physicist

Actors

Design, process, operation...
engineer

CFD, finite elements,
C++, FORTRAN...

Tools

Modelica, Simulink...

Modelica programming language

- ▶ Modelica is an open language for programming models based on **differential algebraic systems of equations**



- ▶ Equations are written in almost **natural language**, and solved by a multipurpose third party tool.
- ▶ It is object-oriented: available **module libraries** cover most applications
 - ▶ Complex models can be achieved simply by combining this modules using a graphical interface!

Modelica tools

- ▶ Main tools :
 - ▶ Dymola (Dassault Systèmes, proprietary)
 - ▶ OpenModelica (Open Source Modelica Consortium, open source)

- ▶ Functions
 - ▶ Flatten equation systems
 - ▶ Compile to machine code after including a solver
 - ▶ Development environment
 - ▶ Model graphical interface
 - ▶ Basic post-processing...

OpenModelica, model graphical view



Piloting models

- ▶ Most OpenTURNS methods apply to functional **black boxes**
 - ▶ Uncertainty propagation and reliability analysis
 - ▶ Sensitivity analysis
 - ▶ Emulation
 - ▶ Parameter estimation



- ▶ We need efficient **input-output data interfaces**,
a.k.a. *wrappers* in OpenTURNS jargon.

Functional mock-up interface (FMI)

- ▶ FMI is a standard for input–output data interface for numerical model.



- ▶ A **functional mock-up unit (FMU)** is a black box following the standard.



OtFMI: integrating FMI support into OpenTURNS

- ▶ The new open source module OtFMI allows transparent use of FMU with OpenTURNS methods.
- ▶ It provides high level classes derived from `ot.PythonFunction`: running an FMU instead of a Python function only requires to **change a single code line!**

```
import otfmi
otfmi.FMUFunction("path/to/fmu",
                  inputs_fmu=["x_1", "x_2", "x_3"],
                  outputs_fmu=["y"])
```

Implementation overview



OtFMI graphical interface

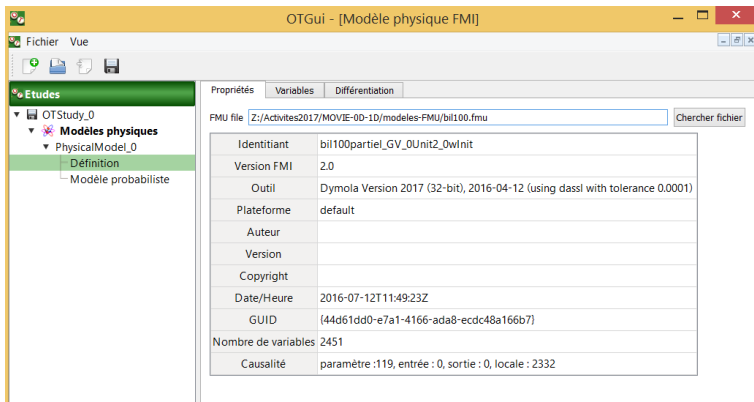
Motivations

- ▶ Provide access to OpenTURNS' methods for Modelica users unfamiliar with Python
- ▶ Considerably ease simple studies

Issues

- ▶ Modelica models often define **hundreds or thousands of variables**

OtFMI graphical interface, FMU overview



OtFMI graphical interface, picking inputs and outputs



Perspectives

- ▶ Most 0D/1D system model are dynamical.
We need methods for sensitivity analysis and emulation of model with **time series inputs or outputs**.
- ▶ EDF is interested into **data assimilation** with its Modelica models.
- ▶ What are the opportunities of **extending the Modelica language** to support stochastic description of variables?

Thank you for your attention.