WHY DO WE WANT TO DEVELOP A TOOLBOX?

We want a toolbox to:

- **accesibilize** the use of techniques
- develop and try new ideas **faster**
- **colaborate** better together.

A good tool:

- solves one specific problem
- uses a software-independent standard for its inputs and outputs
- has enough documentation and examples.

EXAMPLE OF A REALLY GOOD TOOL: STRIKE-GOLDD

The STRIKE-GOLDD toolbox:

- is a free and open-source
- does identifiability and observability analysis
- avoids bad surprises

Systems biology

STRIKE-GOLDD 4.0: user-friendly, efficient analysis of structural identifiability and observability

Sandra Díaz-Seoane [1] **Xabier Rey Barreiro** [1] **and Alejandro F. Villaverde** [1,2,*]

[1]Department of Systems and Control Engineering, Universidade de Vigo, Galicia, 36310 Vigo, Spain and [2]CITMAga, Galicia, 15782 Santiago de Compostela, Spain

*To whom correspondence should be addressed.

Associate Editor: Pier Luigi Martelli

Received on July 19, 2022; revised on November 15, 2022; editorial decision on November 16, 2022; accepted on November 17, 2022

Abstract

Motivation: STRIKE-GOLDD is a toolbox that analyses the structural identifiability and observability of possibly non-linear, non-rational ODE models that may have known and unknown inputs. Its broad applicability comes at the expense of a lower computational efficiency than other tools.
Results: STRIKE-GOLDD 4.0 includes a new algorithm, ProbObsTest, specifically designed for the analysis of rational models. ProbObsTest is significantly faster than the previously available FISPO algorithm when applied to computationally expensive models. Providing both algorithms in the same toolbox allows combining generality and computational efficiency. STRIKE-GOLDD 4.0 is implemented as a Matlab toolbox with a user-friendly graphical interface.
Availability and implementation: STRIKE-GOLDD 4.0 is a free and open-source tool available under a GPLv3 license. It can be downloaded from GitHub at https://github.com/afvillaverde/strike-goldd.
Contact: afvillaverde@uvigo.gal
Supplementary information: Supplementary data are available at Bioinformatics online.

Do you always use STRIKE-GOLDD

before running an identification? Why?

Not using STRIKE-GOLDD is a huge opportunity cost!

Not using software-independent standards leads to…

- ineficient workflows that promote reinventing the wheel
- the Python vs. MATLAB dilemma

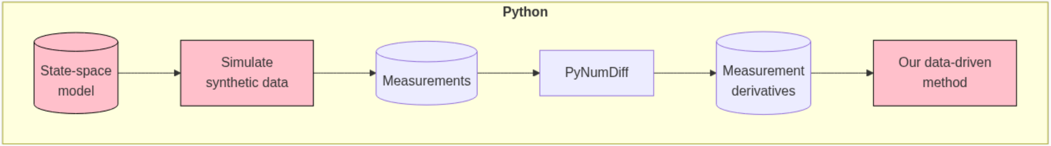
We developed a data-driven method which:

- infers the structure of a model from experimental data
- but requires the measurement derivatives

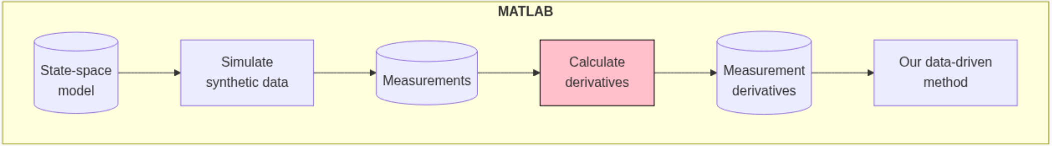
There are many methods to approximate derivatives:

- PyNumDiff is a Python package that implements many of them

Option 1: Do everything in Pyton



Option 2: Do everything in MATLAB



What would you do?

Option 1: Do everything in Python

Option 2: Do everything in MATLAB

What would you do?

Option 1: Do everything in Python $\rightarrow$ not use MATLAB

Option 2: Do everything in MATLAB $\rightarrow$ not use PyNumDiff

THERE IS A THIRD OPTION

Option 3: Use MATLAB and Python

Use a file that can be read and written by:

- MATLAB
- Python

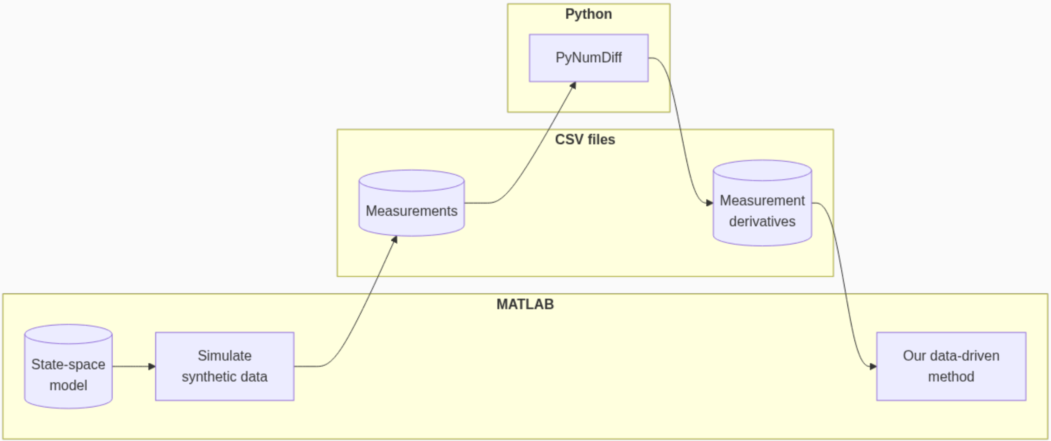
We used CSV (comma-separated value)

Editor - /home/nobel/Sync/01_projects/202405161757 =P= T...

measurements.csv

```
time,    S
0,       9.77647226334696
1,       10.052929466753
2,       9.69446932028831
3,       9.26396045149152
4,       8.89834079526365
5,       8.488393763393
6,       7.98900315436451
7,       7.48229372443103
8,       6.81250854721334
9,       6.17441818357695
10,      5.39309966164657
11,      4.12607487888249
12,      2.82217592629237
13,      1.79117503806433
14,      0.680007877525141
15,      -0.0783014096252128
16,      0.205818536922681
17,      0.0632117190556865
18,      0.0272348018776476
19,      0.103071491534313
20,      0.0522836633408699
```

THE WORKFLOW WE USED



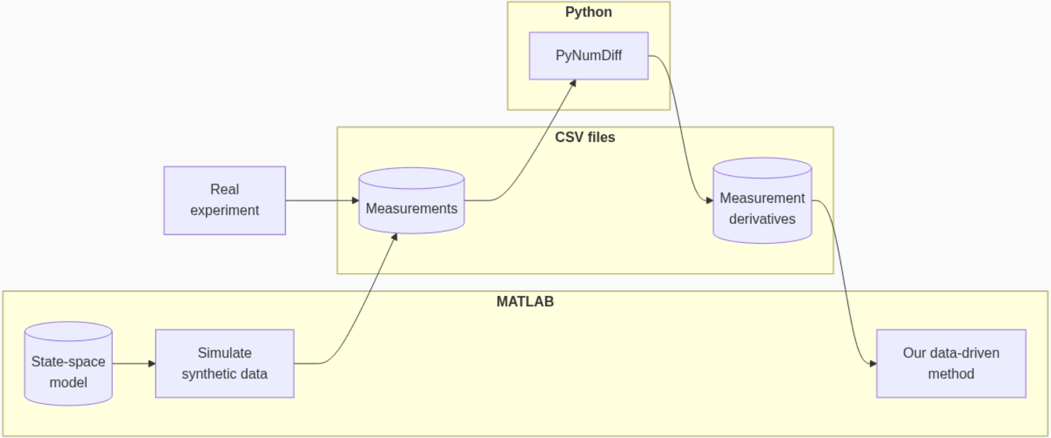
With this workflow, we:

- use MATLAB for (almost) everything
- only need to know the bare minimum of Python:
  - how to load and save a CSV
  - how to use PyNumDiff

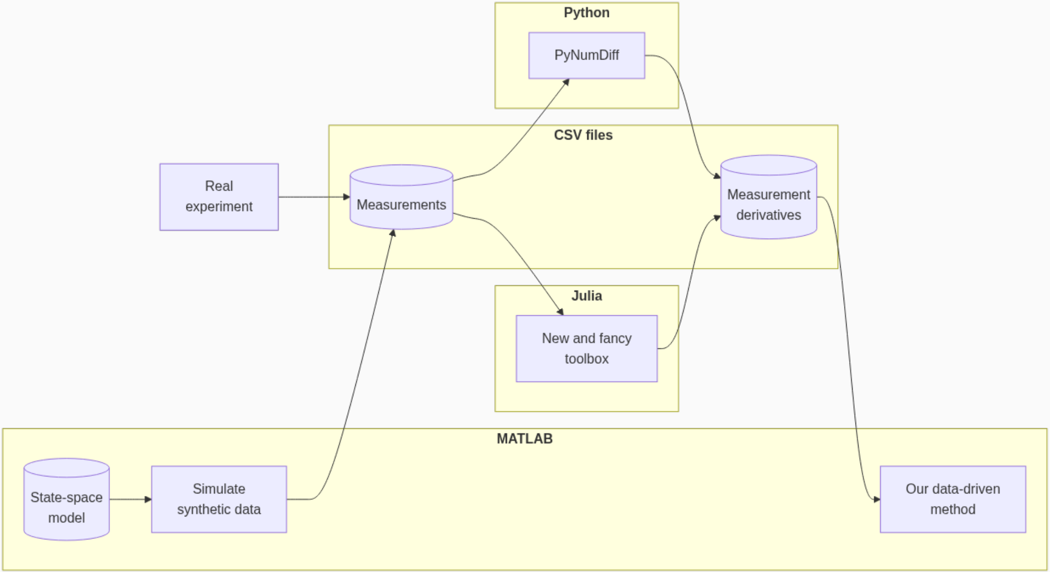
The advantages:

- we avoid the opportunity cost
- we solve the Python vs. MATLAB dilemma
- the workflow is flexible

WE CAN EASILY USE REAL EXPERIMENT DATA



WE CAN EASILY ADD MORE TOOLS



The weakness of this workflow is that:

we need software-independent standards