# 00-data-emoprox2_stim-model-SINDy

August 9, 2023

## 1 Aug 8-9, 2023: trials on fitting SINDy on stimulus signal

[1]: <IPython.core.display.HTML object>

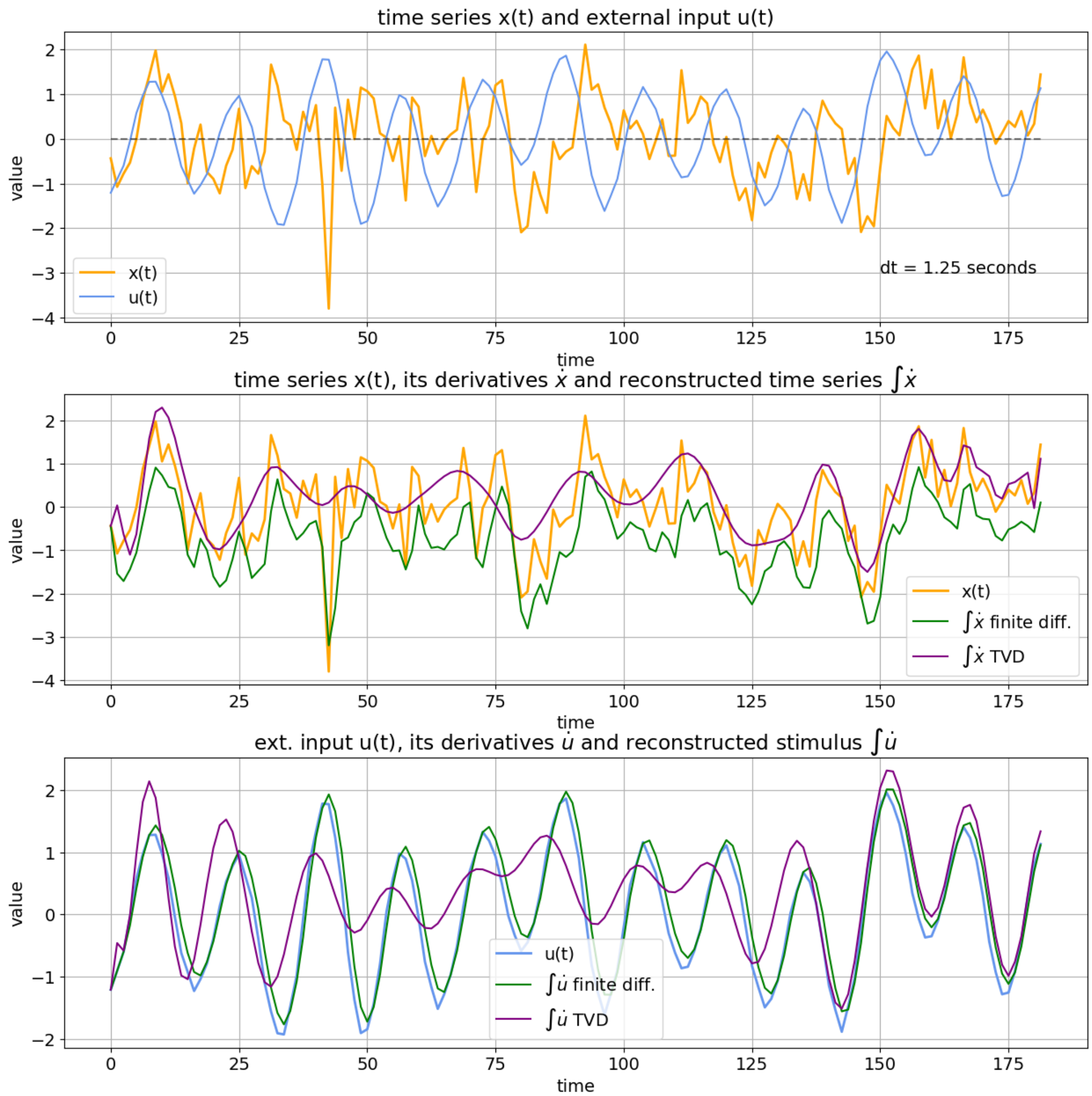
```
(CVXPY) Aug 09 01:18:27 PM: Encountered unexpected exception importing solver
GLOP:
RuntimeError('Unrecognized new version of ortools (9.6.2534). Expected <
9.5.0.Please open a feature request on cvxpy to enable support for this
version.')
(CVXPY) Aug 09 01:18:27 PM: Encountered unexpected exception importing solver
PDLP:
RuntimeError('Unrecognized new version of ortools (9.6.2534). Expected <
9.5.0.Please open a feature request on cvxpy to enable support for this
version.')
```

### 1.1 dataset

```
   Hemi                ROI  Index  Voxels  \
49    R  ant. dorsal Insula     50     235

                                    File_Name
49  Hammers-gm-0.5-2mm-AntDorsal-INS-r.nii.gz
```

[4]: <matplotlib.legend.Legend at 0x7f691de33e50>




## 1.2 SINDy on fMRI signal

SINDy model:

$$\dot{x} = f(x, u)$$

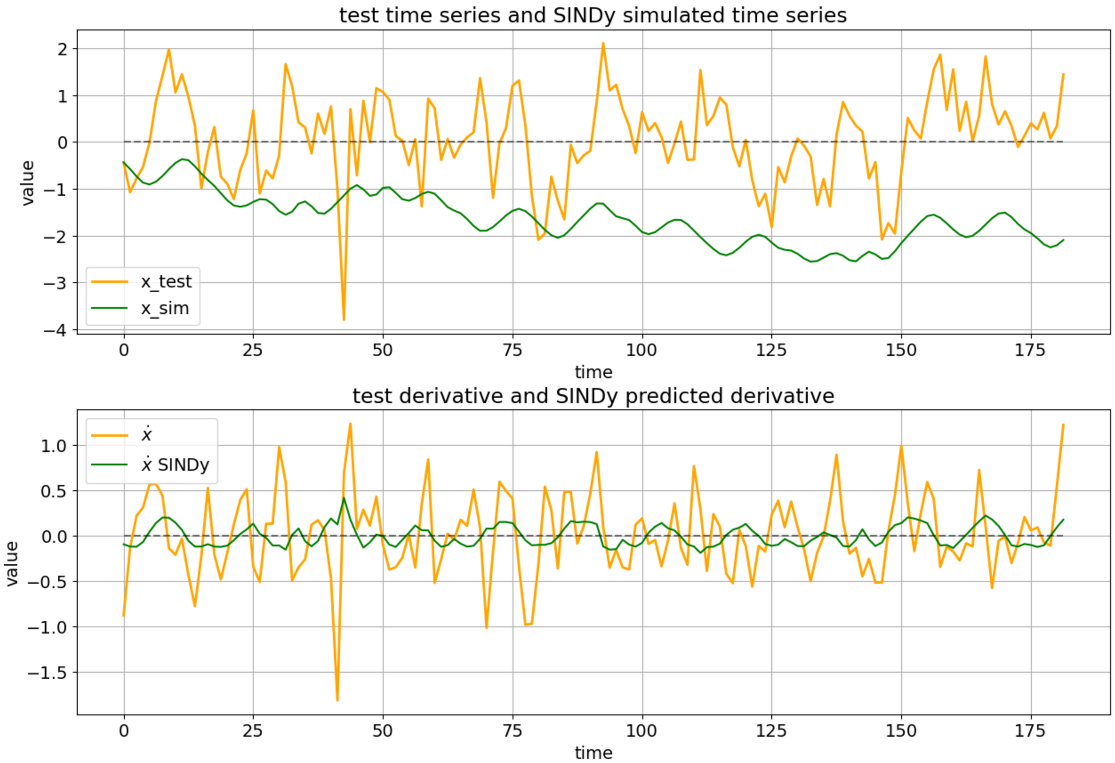
```
model = ps.SINDy(
    optimizer=ps.STLSQ(threshold=0.0),
    feature_library=ps.PolynomialLibrary(degree=3, include_bias=True),
    differentiation_method=ps.FiniteDifference(),
    feature_names=['x', 'u0'],
```

```
    discrete_time=False,
)
```

```
(x)' = -0.056 1 + 0.016 x + 0.151 u0 + -0.007 x^2 + 0.026 x u0 + 0.050 u0^2 + -0.007 x^3 + 0.00
```

```
(x)' = -0.056 1 + 0.016 x + 0.151 u0 + -0.007 x^2 + 0.026 x u0 + 0.050 u0^2 +
-0.007 x^3 + 0.007 x^2 u0 + -0.036 u0^3
```

```
[6]: <matplotlib.legend.Legend at 0x7f691ccdb280>
```




modeling the stimulus:

$$\dot{u} = f(u)$$

## 1.3 SINDy on stimulus signal

### 1.3.1 time-delay coordinates

time-delay coordinates

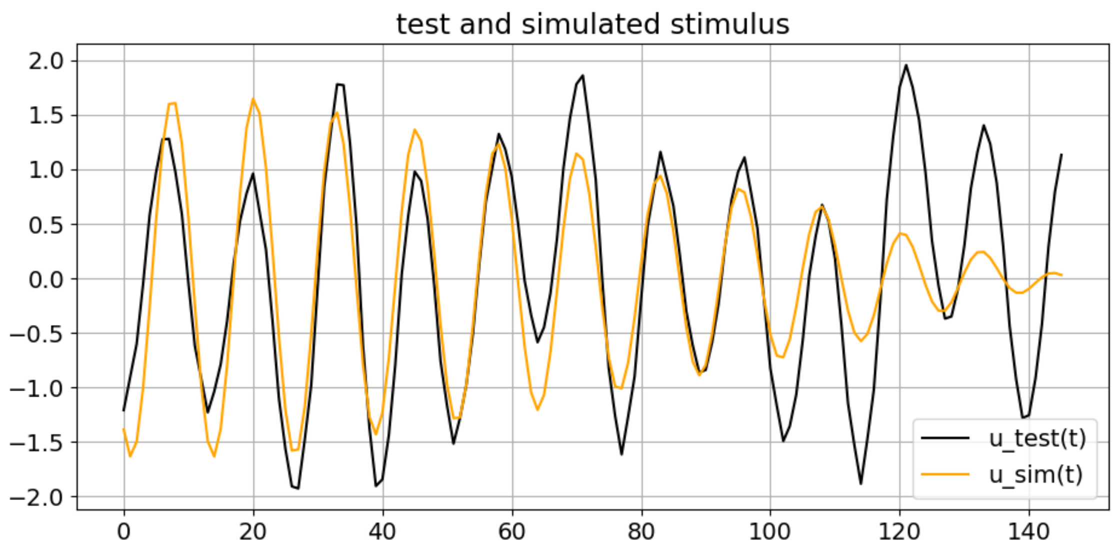
```
model = ps.SINDy(
    optimizer=ps.STLSQ(0.01),
    feature_library=ps.PolynomialLibrary(degree=10),
    differentiation_method=ps.SINDyDerivative(kind='finite_difference', k=4),
```

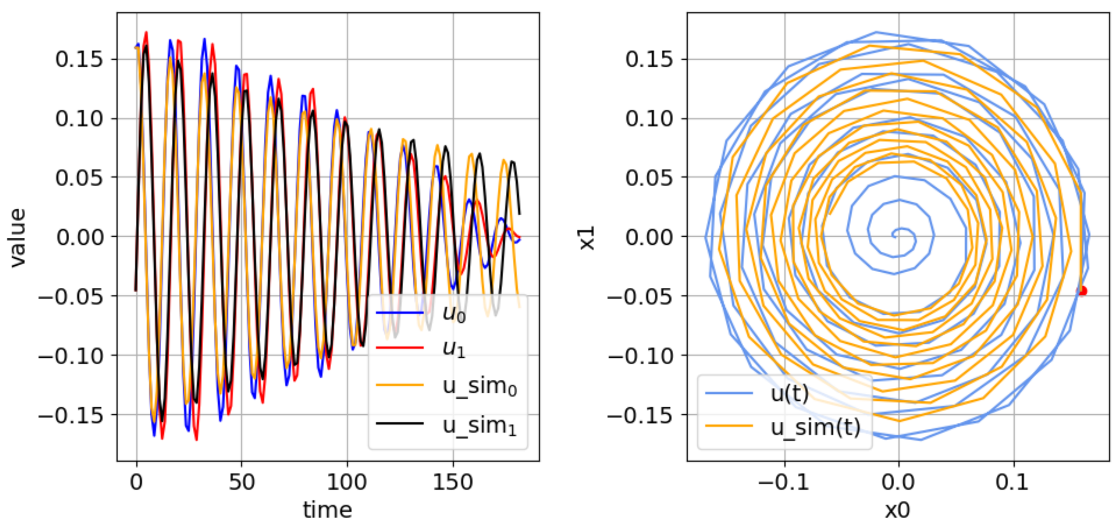
```
    feature_names=['u0', 'u1'],
    discrete_time=False
)
```

```
(u0)' = -0.011 u0 + -0.393 u1
(u1)' = 0.404 u0
```


```
(u0)' = -0.011 u0 + -0.393 u1
(u1)' = 0.404 u0
```


[8]: <matplotlib.legend.Legend at 0x7f691cb49fd0>



issues:
- the simulated signals decay with time.
- the oscillations also do not match those in the test signal.

==technical question==:
how do we map back the simulated signals from time-delay coordinates to the 1D space?
we map the 1D time series `x_train` onto time-delay coordinates using SVD of Hankel matrix.

$USV^* = H$

and use columns of `V` as time-delay coords. we can map back from time-delay coords to 1D spac
but how do we do so for another signal, say `x_test`?

### 1.3.2 2D ODE

#### typical 2D system

2D ODE.
maybe 1D dynamical system cannot generate oscillations (observed in the stimulus).
try modeling a second order system.

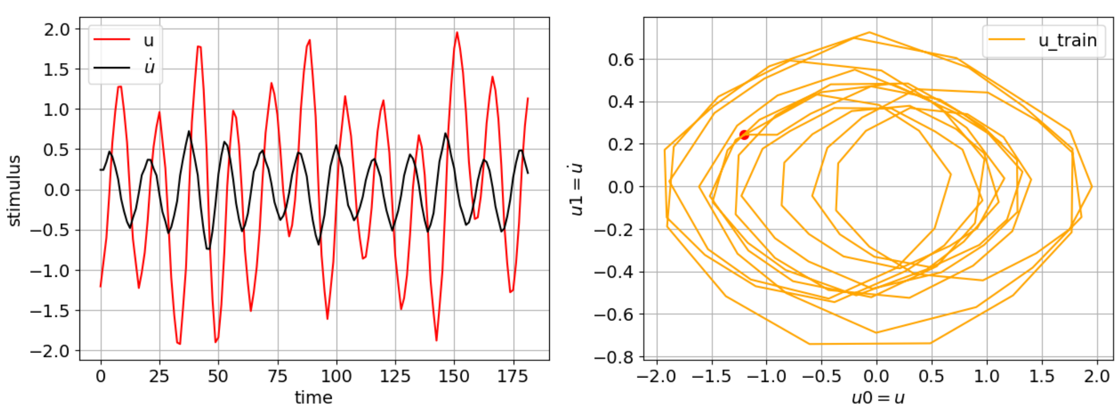
$$\frac{d}{dt}\begin{bmatrix} u \\ \dot{u} \end{bmatrix} = f\left(\begin{bmatrix} u \\ \dot{u} \end{bmatrix}\right)$$

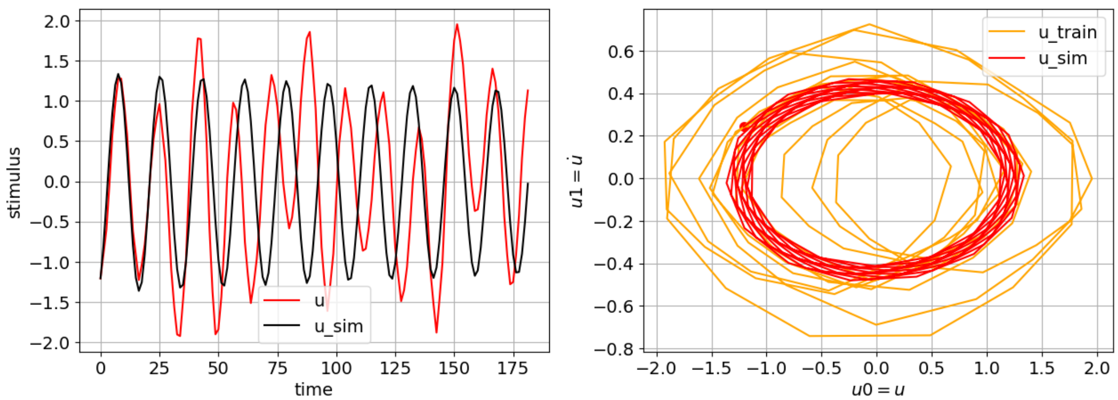
```
model = ps.SINDy(
    optimizer=ps.STLSQ(threshold=0.002, normalize_columns=True),
    feature_library=ps.PolynomialLibrary(degree=2),
    differentiation_method=ps.FiniteDifference(),
    feature_names=['u0', 'u1'],
)
```

```
(u0)' = 1.000 u1
(u1)' = 0.002 1 + -0.124 u0 + -0.002 u1 + 0.002 u0^2 + 0.015 u0 u1 + -0.037 u1^2
```


```
(u0)' = 1.000 u1
(u1)' = 0.002 1 + -0.124 u0 + -0.002 u1 + 0.002 u0^2 + 0.015 u0 u1 + -0.037 u1^2
```


[9]: <matplotlib.legend.Legend at 0x7f691c941d30>






clearly there is not fit.

### SINDyPI formulation

```
library_functions = [
    lambda u: u,
    lambda u: u*u,
]
library = ps.PDELibrary(
    library_functions=library_functions,
    temporal_grid=t,
    function_names=library_function_names,
    include_bias=True,
    implicit_terms=True,
    derivative_order=2,
).fit(x)

optimizer = ps.SINDyPI(
    threshold=0.1,
    tol=1e-5,
    thresholder="l1",
    max_iter=6000,
    # normalize_columns=True
)

model = ps.SINDy(
    optimizer=optimizer,
    feature_library=library,
    feature_names=['u'],
)
```

```
u_tt = -0.003 1 + -0.075 u + -0.001 u^2 + 0.006 uu_t + -0.001 u^2u_t + 0.230 u^2u_tt
```

$\implies$

$$\frac{d}{dt}\begin{bmatrix} u \\ \dot{u} \end{bmatrix} = \begin{bmatrix} \dot{u} \\ \frac{-0.075u}{1-0.23u^2} \end{bmatrix}$$

```
Model  0
Model  1
Model  2
Model  3
Model  4
Model  5
Model  6
Model  7
Model  8
1 = -0.031 u + 0.472 u^2 + 0.086 u_t + -0.163 u_tt + 0.006 uu_t + -0.008 u^2u_t
+ -0.320 uu_tt + -0.151 u^2u_tt
u = -0.025 1 + -0.094 u^2 + -0.004 u_t + -2.992 u_tt + 0.153 uu_t + 0.014 u^2u_t
+ -0.942 uu_tt + -1.055 u^2u_tt
u^2 = 0.409 1 + -0.127 u + -0.052 u_t + 0.008 u_tt + 0.155 u^2u_t + -4.431 uu_tt
+ -0.478 u^2u_tt
u_t = 0.013 1 + -0.005 u^2 + 0.010 uu_t + 0.689 u^2u_t + -0.003 uu_tt
u_tt = -0.003 1 + -0.075 u + -0.001 u^2 + 0.006 uu_t + -0.001 u^2u_t + 0.230
u^2u_tt
uu_t = 0.031 u + -0.002 u^2 + 0.009 u_t + 0.059 u_tt + -0.088 u^2u_t + 0.064
u^2u_tt
u^2u_t = -0.001 1 + 0.003 u + 0.020 u^2 + 0.614 u_t + -0.076 uu_t + 0.148 uu_tt
+ 0.018 u^2u_tt
uu_tt = -0.008 1 + -0.034 u + -0.124 u^2 + 0.026 u^2u_t + -0.151 u^2u_tt
u^2u_tt = -0.011 1 + -0.117 u + -0.039 u^2 + 1.024 u_tt + 0.027 uu_t + 0.014
u^2u_t + -0.468 uu_tt
```