Machine learning to efficiently linearize system for control

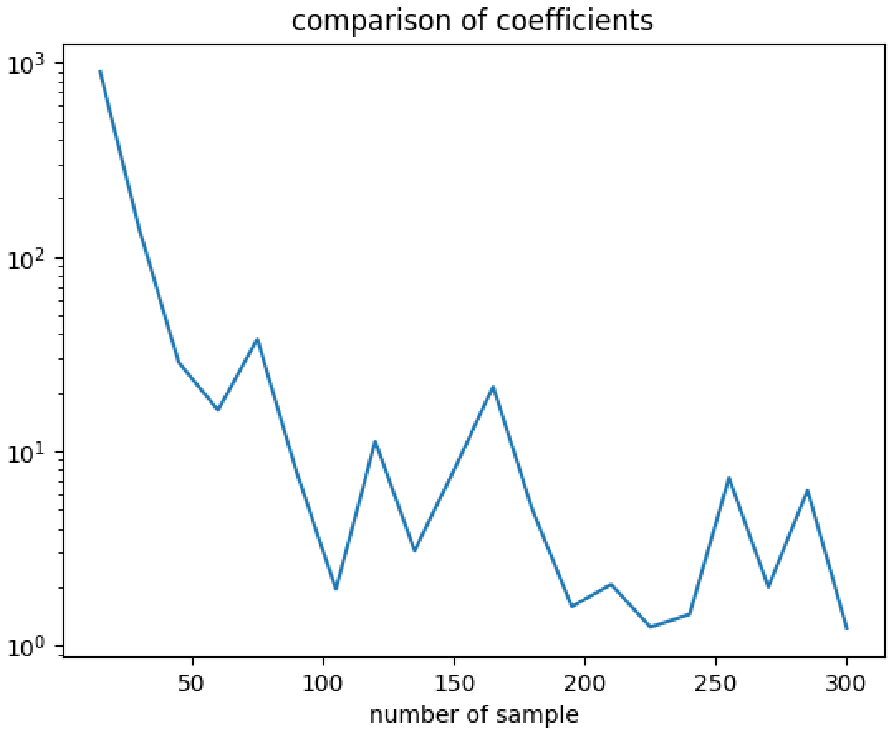
Outline

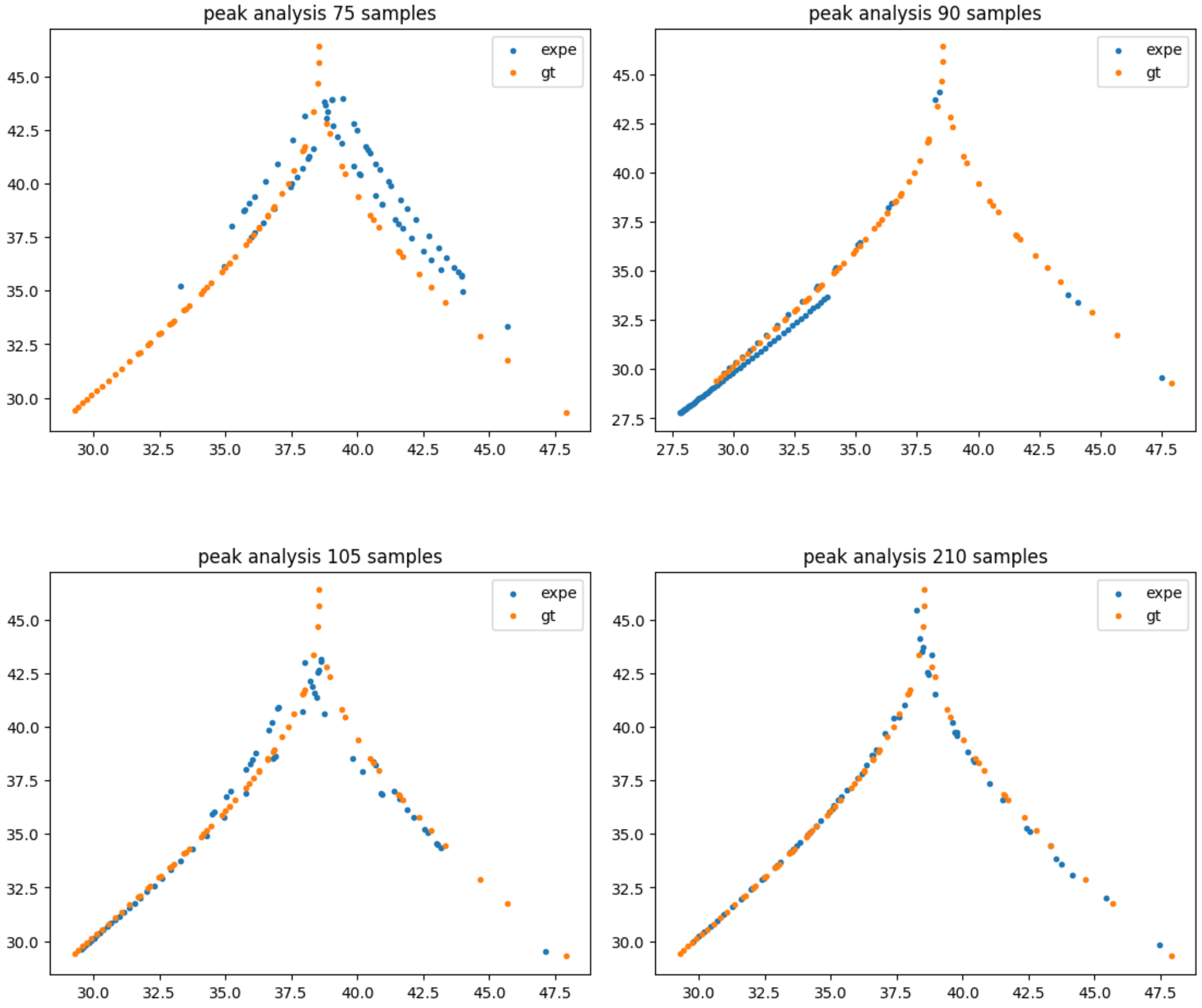
- Efficiency
- Dual method
- Nengo questions

Efficiency in data

- Lorentz system

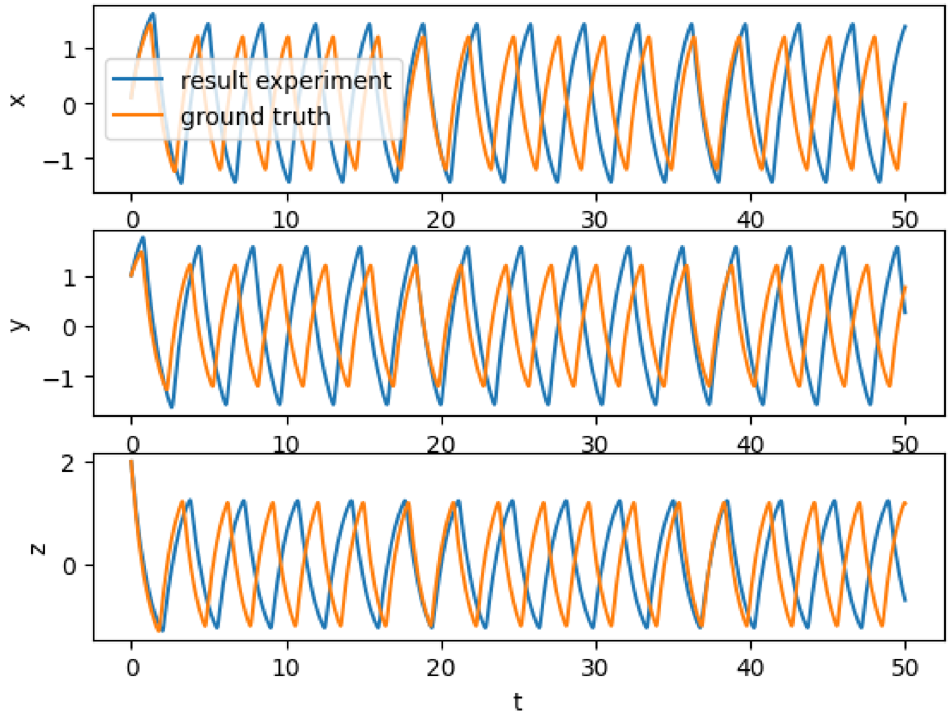
$$||W_{true} - W_{expe}||$$

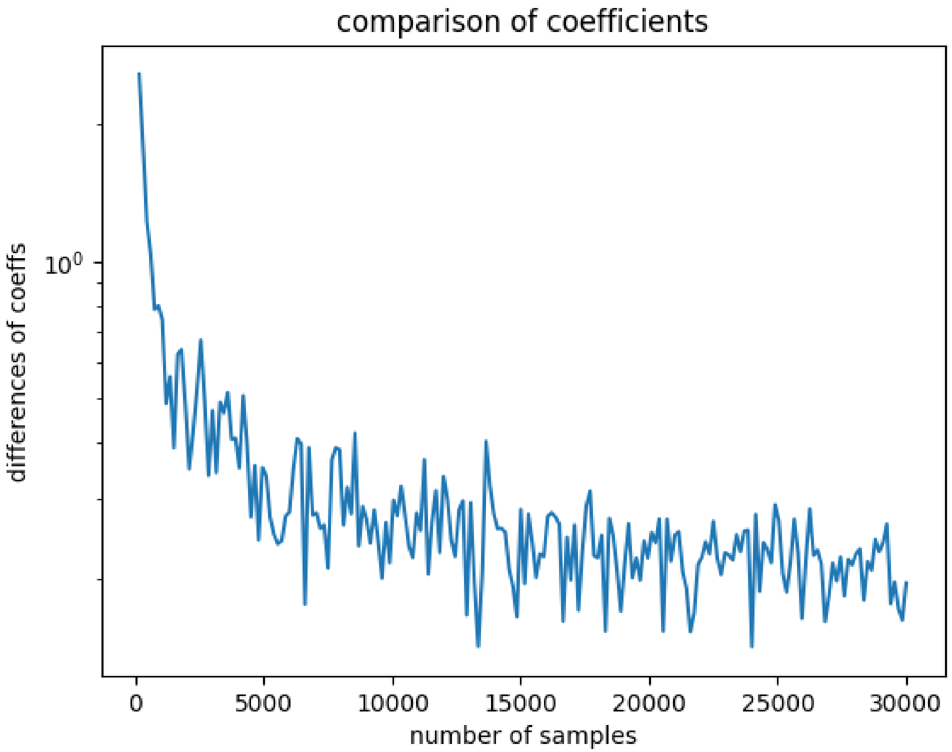




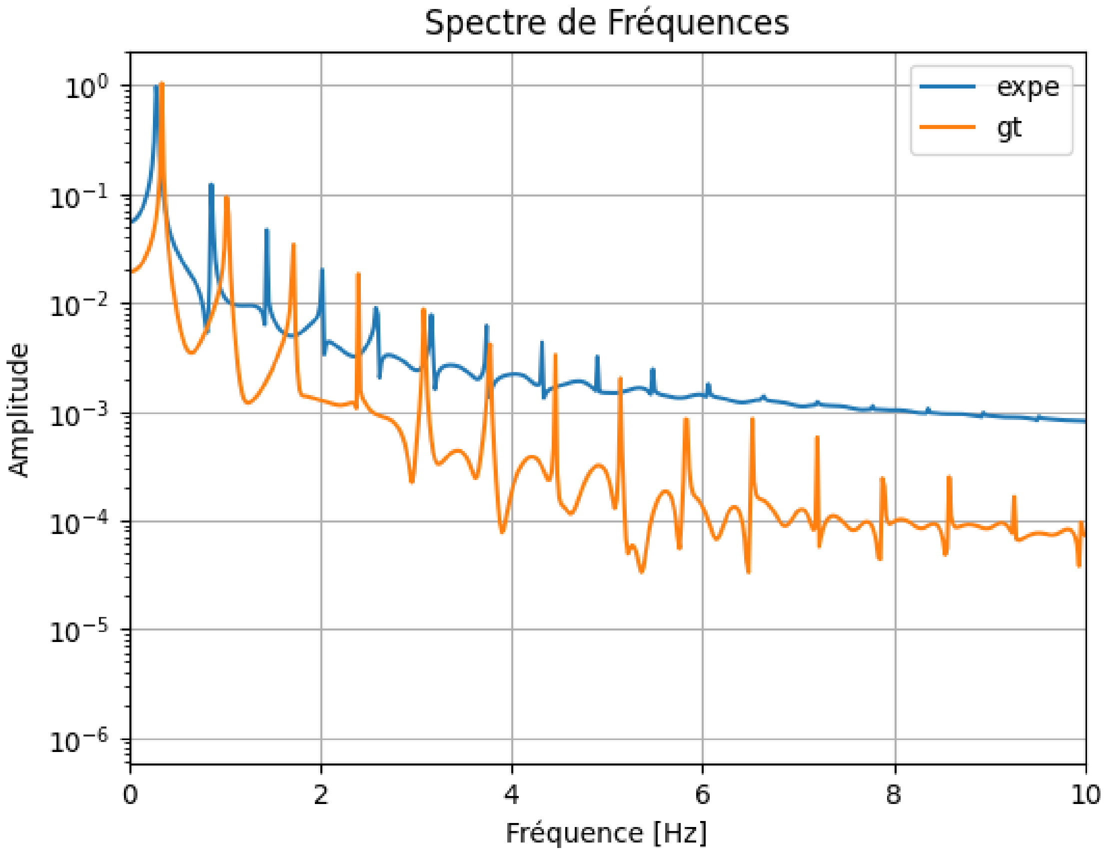
Efficiency in data

- Tanh system

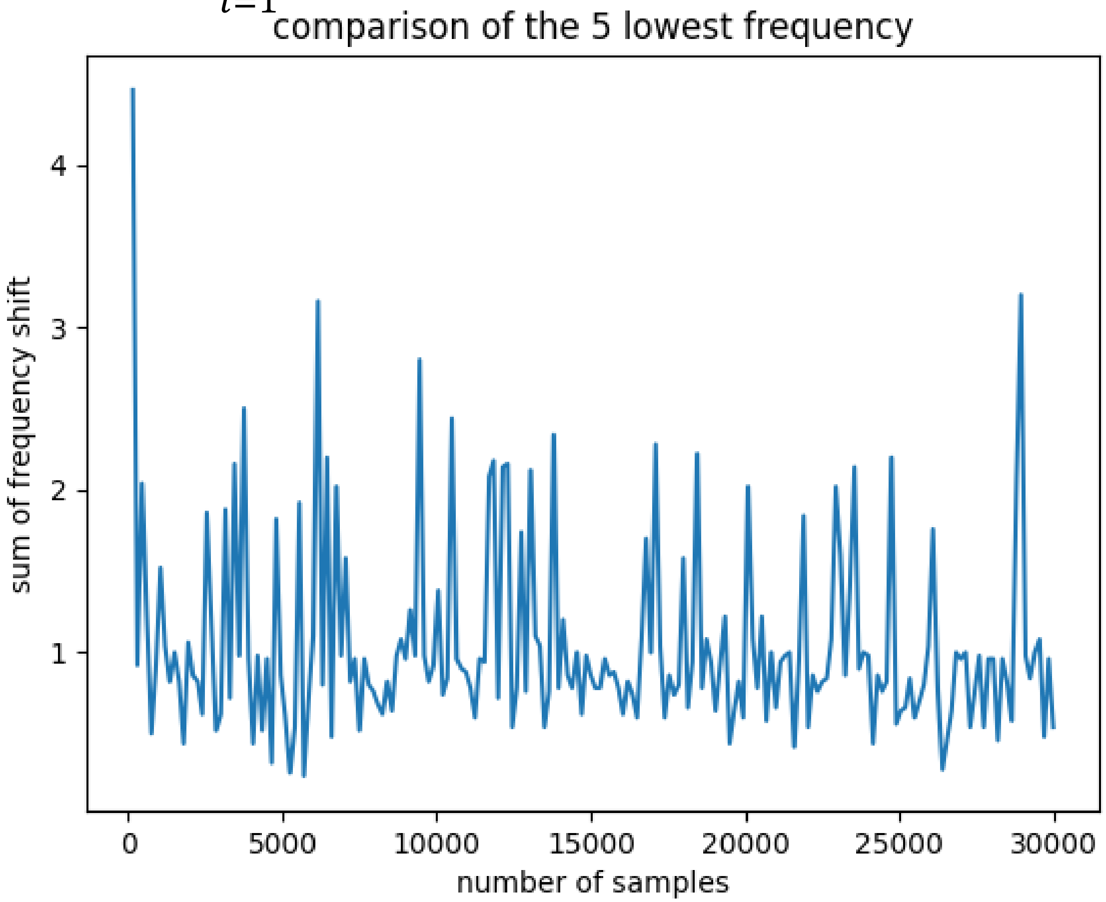




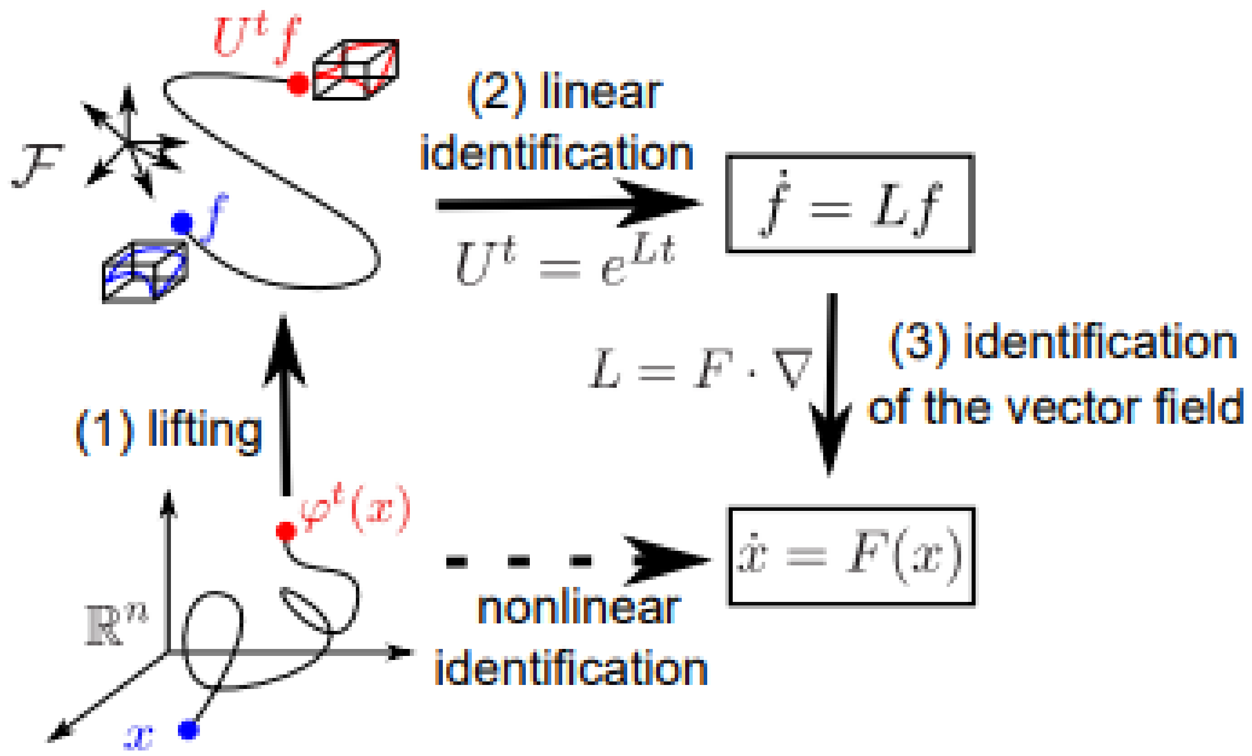
Efficiency in data



$$\sum_{i=1}^{5} |f_{true} - f_{expe}|$$



Dual method



$U^t f$
$\mathcal{F}$
f
(2) linear
identification
$U^t = e^{Lt}$
$\dot{f} = Lf$
$L = F \cdot \nabla$
(3) identification
of the vector field
(1) lifting
$\varphi^t(x)$
$\mathbb{R}^n$
x
nonlinear
identification
$\dot{x} = F(x)$

Dual method

- => Partial observation : (x_k, y_k) K samples
- Real number of dimension: n
- Library functions : (h_k) N functions
- Assume $\mathrm{K} \leq N$

Dual method

Gaussian radial basis function $$g_k(\mathbf{x}) = e^{-\gamma\|\mathbf{x}-\mathbf{x}_k\|^2}$$

$$P_X, P_Y = g_k(x), g_k(y)$$

$$U = P_Y P_X^{\dagger}$$

$$\widetilde{\mathbf{L}}_{\mathbf{data}} = \frac{1}{T_s}\log(\mathbf{P_y}\,\mathbf{P_x^{\dagger}})$$

Dual method

Gaussian radial basis function $$g_k(\mathbf{x}) = e^{-\gamma\|\mathbf{x}-\mathbf{x}_k\|^2}$$

$$P_X, P_Y = g_k(x), g_k(y)$$

$$U = P_Y P_X^{\dagger}$$

$$\widetilde{\mathbf{L}}_{\mathbf{data}} = \frac{1}{T_s} \log(\mathbf{P_y} \, \mathbf{P_x^{\dagger}})$$

Dual Method

$$\begin{pmatrix} \mathbf{F}(\mathbf{x}_1) \cdot \nabla f(\mathbf{x}_1) \\ \vdots \\ \mathbf{F}(\mathbf{x}_K) \cdot \nabla f(\mathbf{x}_K) \end{pmatrix} = \begin{pmatrix} Lf(\mathbf{x}_1) \\ \vdots \\ Lf(\mathbf{x}_K) \end{pmatrix} \approx \widetilde{\mathbf{L}}_{\mathbf{data}} \begin{pmatrix} f(\mathbf{x}_1) \\ \vdots \\ f(\mathbf{x}_K) \end{pmatrix}$$

$$\begin{pmatrix} \hat{\mathbf{F}}(\mathbf{x}_1)^T \\ \vdots \\ \hat{\mathbf{F}}(\mathbf{x}_K)^T \end{pmatrix} \approx \widetilde{\mathbf{L}}_{\mathbf{data}} \begin{pmatrix} \mathbf{x}_1^T \\ \vdots \\ \mathbf{x}_K^T \end{pmatrix}$$

Dual method

- Assume

$$\hat{F}_j(\mathbf{x}_k) = \sum_{l=1}^{N_F} \hat{w}_j^l \, h_l(\mathbf{x}_k)$$

j dimension of x

N_F new N

$$\begin{pmatrix} \hat{F}_j(\mathbf{x}_1) \\ \vdots \\ \hat{F}_j(\mathbf{x}_K) \end{pmatrix} = \mathbf{H}_\mathbf{x} \begin{pmatrix} \hat{w}_1^j \\ \vdots \\ \hat{w}_{N_F}^j \end{pmatrix}$$

$$\mathbf{H}_\mathbf{x} = \begin{pmatrix} \mathbf{h}(\mathbf{x}_1)^T \\ \vdots \\ \mathbf{h}(\mathbf{x}_K)^T \end{pmatrix}$$

Dual method

$$\min_{\mathbf{w}\in\mathbb{R}^{N_F}} \left\| \mathbf{H}_{\mathbf{x}}\mathbf{w} - \begin{pmatrix} \hat{F}_j(\mathbf{x}_1) \\ \vdots \\ \hat{F}_j(\mathbf{x}_K) \end{pmatrix} \right\|_2^2 + \boxed{\rho\|\mathbf{w}\|_1}$$

Solutions :

$$w = H_x^{\dagger} F_j$$

Regression pytorch

Partial observation

- Real state X = (x,y,z) measure $\tilde{X} = (x_0 \ \ldots x_{-12})$
- Assume $\exists T$, $X = T(\tilde{X})$
- Objective : learning T with the dual method

Learning transformation T

$$\widetilde{\mathbf{L}}_{\mathbf{data}} = \frac{1}{T_s} \log(\mathbf{P_y}\,\mathbf{P_x^\dagger})$$

$$\min_{T,\mathbf{w}\in\mathbb{R}^{N_F}} \left\| \mathbf{H_x w} - \begin{pmatrix} \hat{F}_j(\mathbf{x}_1) \\ \vdots \\ \hat{F}_j(\mathbf{x}_K) \end{pmatrix} \right\|_2^2 + \quad \ldots\ldots$$

$$F = L.T(\tilde{X})$$

$$H = h(T(\tilde{X})$$

Results

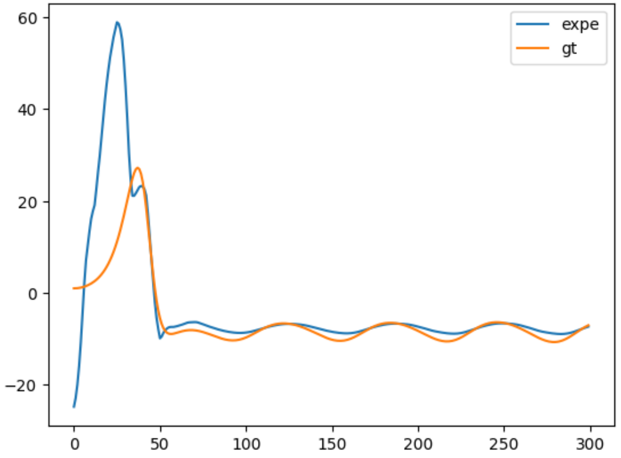
Learning T and

$$w = H_x^{\dagger} F_j$$

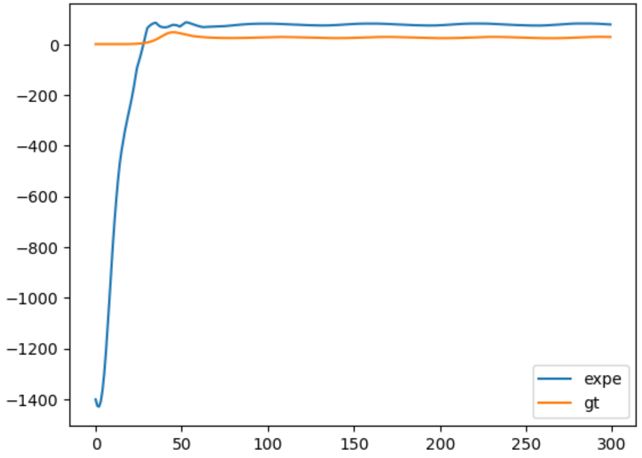
```
tensor([[ 1.0588, -0.2513,  0.0000],
        [-9.7960, 27.8927, -0.2003],
        [ 9.8177, -1.1031, -0.2747],
        [ 0.0000,  0.0000,  0.1146],
        [ 0.0000,  0.0000,  0.0000],
        [ 0.0000,  0.0000,  0.0000],
        [ 0.0000, -0.3808,  0.0000],
        [-0.1320,  0.0000,  0.0000],
        [ 0.0000,  0.0000,  0.0000],
        [ 0.0000,  0.0000,  0.0000]])
```

```
[[  0.        ,  0.        ,  0.        ],
 [-10.        , 28.        ,  0.        ],
 [ 10.        , -1.        ,  0.        ],
 [  0.        ,  0.        , -2.6666667],
 [  0.        ,  0.        ,  0.        ],
 [  0.        ,  0.        ,  1.        ],
 [  0.        , -1.        ,  0.        ],
 [  0.        ,  0.        ,  0.        ],
 [  0.        ,  0.        ,  0.        ],
 [  0.        ,  0.        ,  0.        ]]
```

Giving true coefficients ,



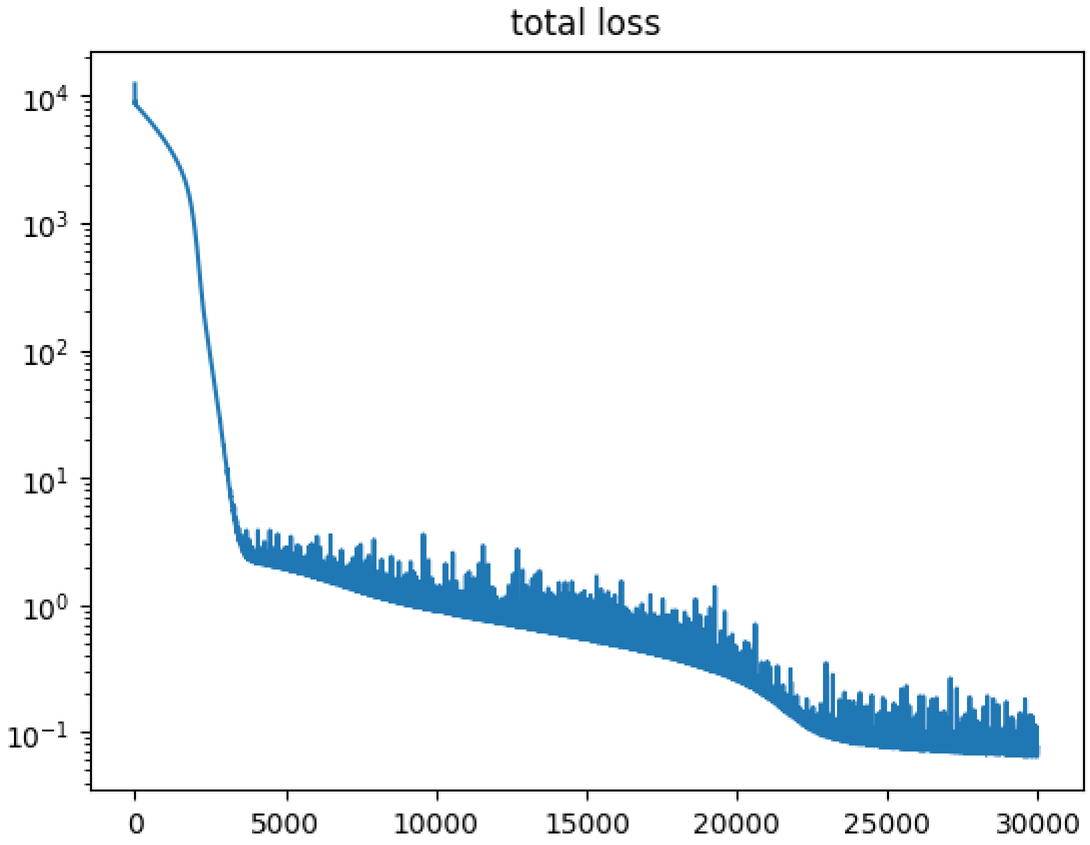
Loss : $||W_{true} - W_{expe}||$



Results

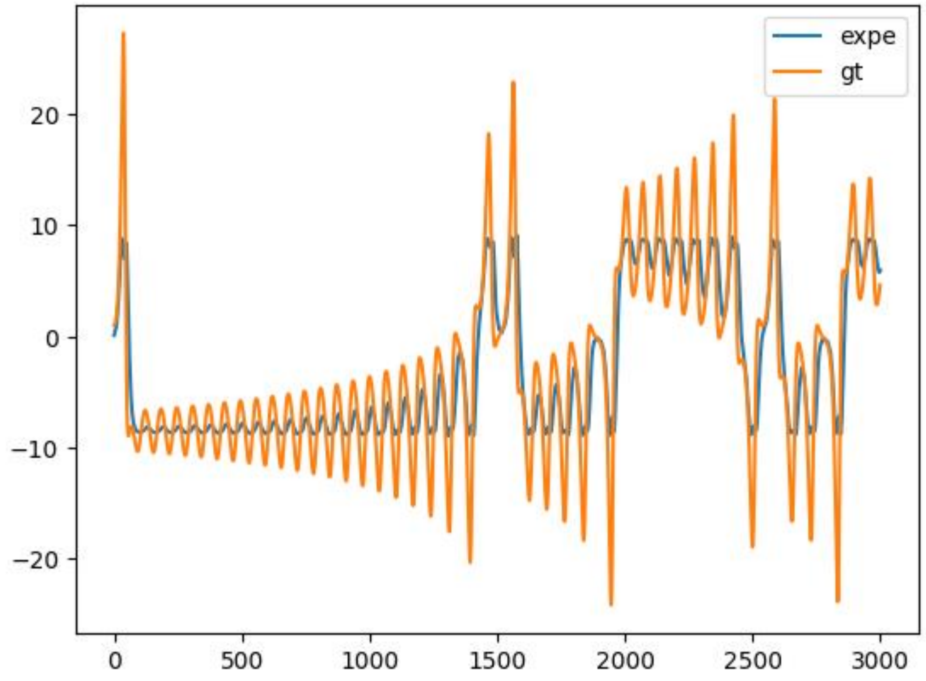
No information on coefficients, loss from the paper

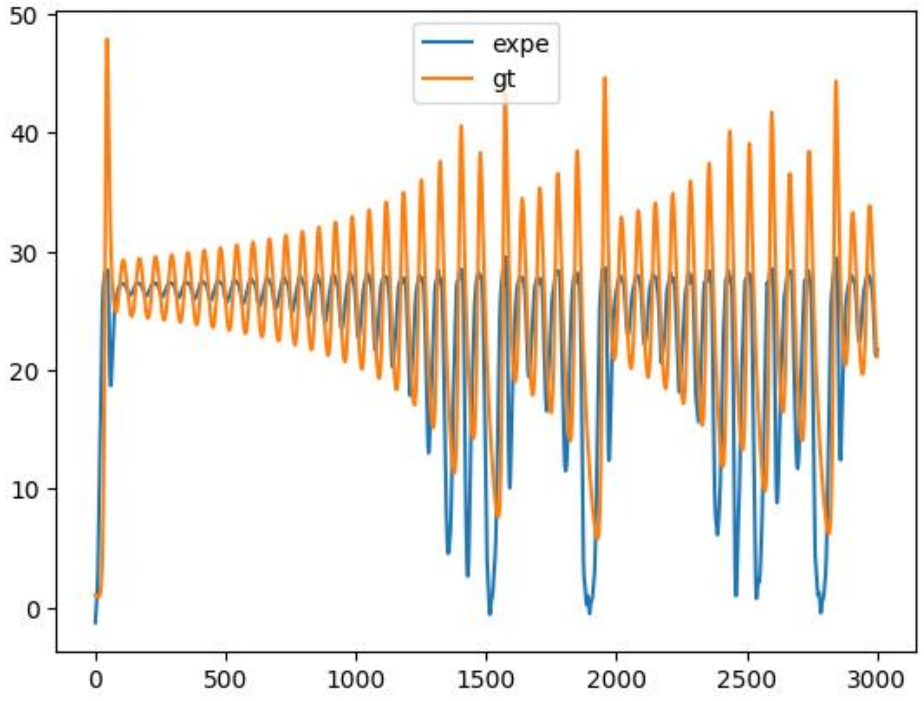
```
[[-2.5555,  8.1897,  6.9786],
 [ 5.9662,  7.6933,  7.4589],
 [ 4.9569,  8.3858,  6.5323],
 [ 7.6605,  2.9042,  4.9652],
 [ 6.4618,  6.2735,  3.6411],
 [ 9.9235,  5.9405,  4.6084],
 [ 7.4591,  8.0647,  6.2848],
 [ 3.7401,  0.9815,  1.3724],
 [ 5.2372,  7.3788,  5.1739],
 [-1.2567,  8.1022,  1.3520]]
```



Results

- Learning knowing w_{true}





Nengo

- Weights ?
- Decoder ?

Questions ?