



HAL
open science

CD-ROM – Complementary Deep - Reduced Order Model

Emmanuel Menier, Michele Alessandro Bucci, Mouadh Yagoubi, Marc Schoenauer, Lionel Mathelin

► **To cite this version:**

Emmanuel Menier, Michele Alessandro Bucci, Mouadh Yagoubi, Marc Schoenauer, Lionel Mathelin. CD-ROM – Complementary Deep - Reduced Order Model. IUTAM Symposium on Data-driven modeling and optimization in fluid mechanics 2022, Jun 2022, Aarhus, Denmark. hal-04405482

HAL Id: hal-04405482

<https://hal.science/hal-04405482v1>

Submitted on 19 Jan 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution 4.0 International License

CD-ROM

Complementary Deep - Reduced Order Model

E. Menier^{1,2} M.A. Bucci² M. Yagoubi¹ L. Mathelin³
M. Schoenauer²

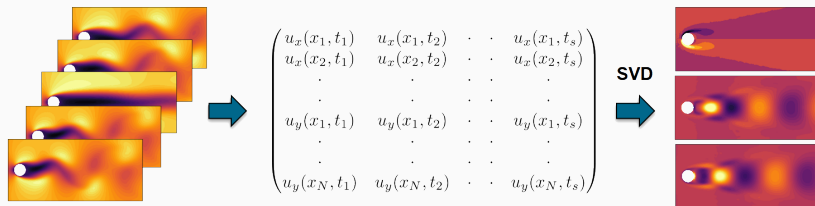


¹IRT - SystemX, Palaiseau, France

²TAU, Inria / LISN, Université Paris-Saclay / CNRS, Orsay, France

³LISN, Université Paris-Saclay & CNRS, Orsay, France

- Proper Orthogonal Decomposition can be used to extract the principal modes (V) from a physical simulation :



- The flow approximates as a linear combination of a reduced number (r) of modes :

$$u(x, t) \approx \bar{u} = V(x)\alpha(t)$$

$$V \in \mathbb{R}^{n_x \times r}, \alpha \in \mathbb{R}^r$$

Galerkin Projection

- An equation for the coefficients of this simplified form is obtained through Galerkin Projection of the original system :

$$\frac{\partial u(x, t)}{\partial t} = g(u(x, t)), \quad V^T u = \alpha$$

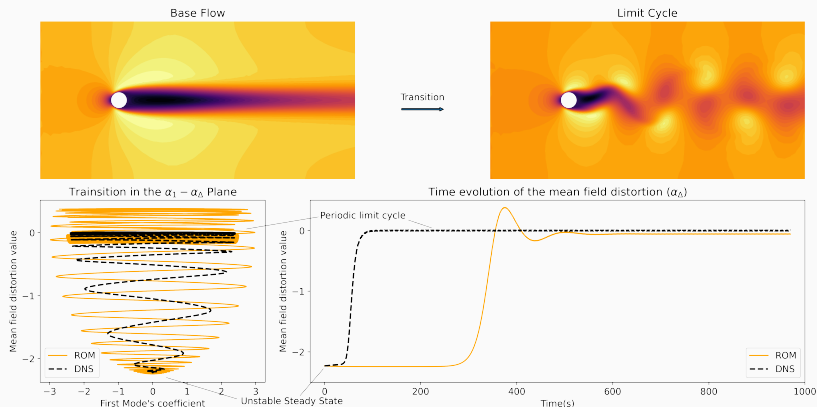
$$\implies \frac{d\alpha}{dt} = V^T g(u(x, t))$$

- The dynamics $g()$ of the reduced model are computed from the approximate solution, which introduces error :

$$\frac{d\alpha}{dt} \approx V^T g(\bar{u}) \neq V^T g(u(x, t))$$

ROM Errors

- In the case of a laminar 2-D cylinder flow, a 3 equation model can be used to capture the transient dynamics¹ :



¹ B. R. Noack et al., "A hierarchy of low-dimensional models for the transient and post-transient cylinder wake", *Journal of Fluid Mechanics* **497**, 335–363 (2003).

- To address classical POD-Galerkin model's shortcomings, we propose to add a correction term to their dynamics :

$$\frac{d\alpha}{dt} = V^T g(u) = V^T g(\bar{u}) + f$$

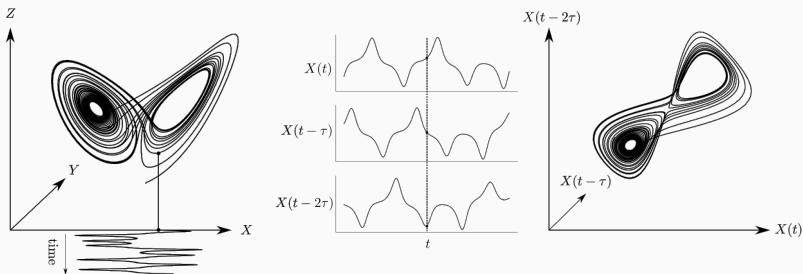
- This correction term depends on information outside of the POD basis:

Corrected ROM:

$$g(u) = g(VV^T u + (I - VV^T)u) \approx g(V\alpha) + g'_{V\alpha}(I - VV^T)u$$

Takens Theorem

- Following Takens's theorem, we can retrieve the information lost during the projection on the POD basis by considering the past states of the system :



- Delay Differential Equations can be used to aggregate information from the past in a time continuous manner :

$$\frac{dx}{dt} = f(t, x, y), \quad y(t) = \int_{-\infty}^t x(\tau) e^{\lambda(\tau-t)} d\tau, \quad \lambda \in \mathbb{R}_+$$

- These equations are solved as an augmented ODE system :

$$\begin{array}{l} \frac{dx}{dt} \\ \frac{dy}{dt} \end{array} = \begin{array}{l} f(t, x, y) \\ x - \lambda y \end{array}$$

Neural Correction Term

- We use Neural Networks to learn the ROM's correction from the memory y :

$$\begin{array}{rcl} \frac{d\alpha}{dt} & = & V^T g(V\alpha) + \mathcal{NN}(y) \\ \frac{dy}{dt} & & \alpha - \lambda y \end{array}$$

- The information available in memory y is limited by its dimension :

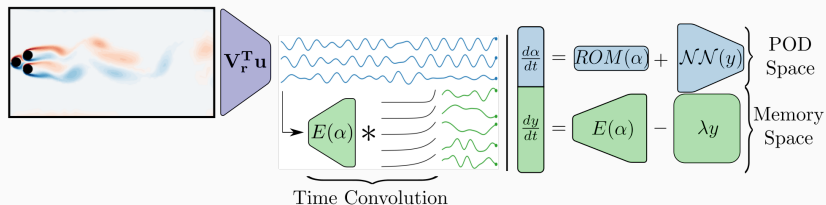
$$y(t) = \int_{-\infty}^t \alpha(\tau) e^{\lambda(\tau-t)} d\tau \implies \dim(y) = \dim(\alpha)$$

Encoders

- To address the bottleneck in the memory, we propose to add an encoder network to expand the dimension of the memory :

$$\begin{array}{l} \frac{d\alpha}{dt} \\ \frac{dy}{dt} \end{array} = \begin{array}{l} V^T g(V\alpha) \\ E(\alpha) \end{array} + \begin{array}{l} \mathcal{NN}(y) \\ - \lambda y \end{array}$$

- Leading to the final CD-ROM formulation :



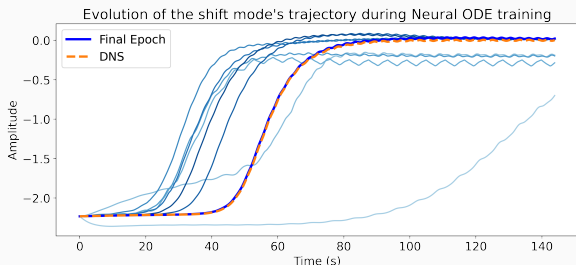
Training Data and Neural ODE

- Training data for the true ROM trajectories can be obtained from the snapshot data :

$$\hat{\alpha}_{t_i} = V^T u_{DNS,t_i}$$

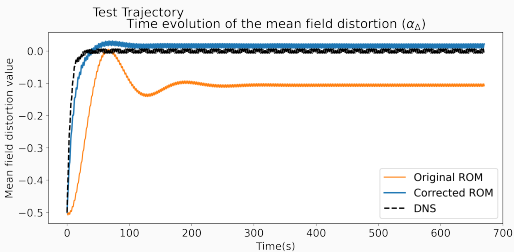
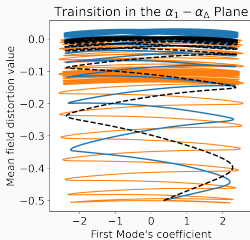
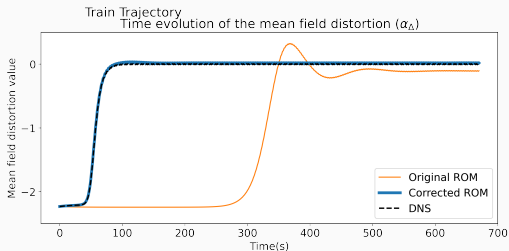
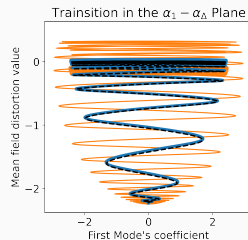
- The model can then be trained through adjoint backpropagation² :

$$\mathcal{L}(\alpha) = \sum_{t_0}^{t_n} \|\alpha_{t_i} - \hat{\alpha}_{t_i}\|_2$$



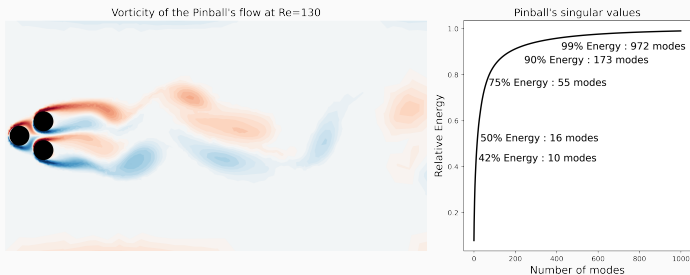
² R. T. Q. Chen et al., "Neural ordinary differential equations", (2019).

Cylinder Results



Chaotic Pinball

- We tested the approach on a more challenging case, the chaotic pinball³:

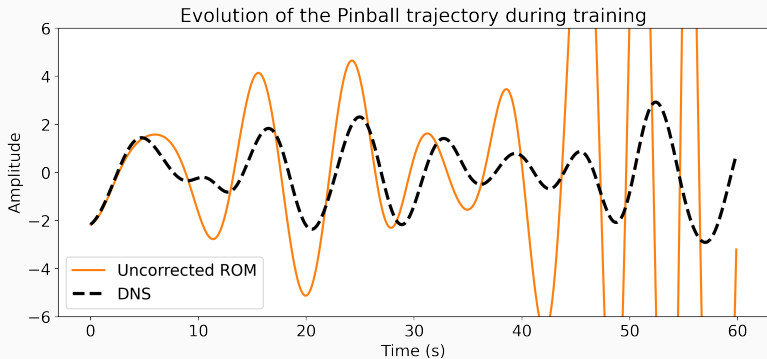


- Reminder : 8 modes are enough to capture 99 % of the cylinder flow's energy.

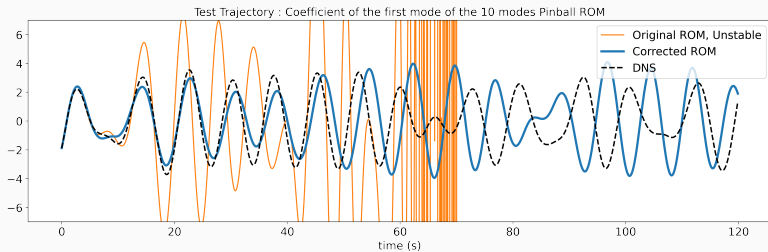
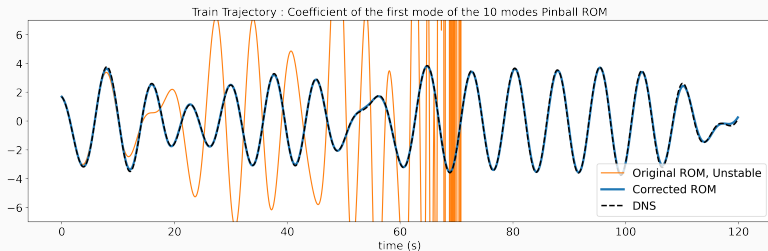
³ N. Deng et al., "Low-order model for successive bifurcations of the fluidic pinball", *Journal of Fluid Mechanics* 884, 10.1017/jfm.2019.959 (2019).

Unstable ROM

- Taking a very reduced number ($r = 10$) of modes for the pinball flow yields a very unstable model.

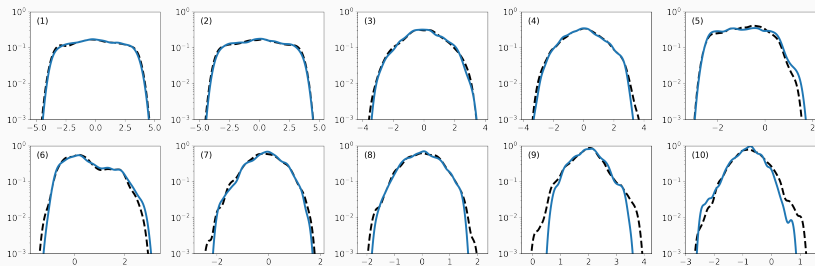


Pinball Results



Attractor Statistics

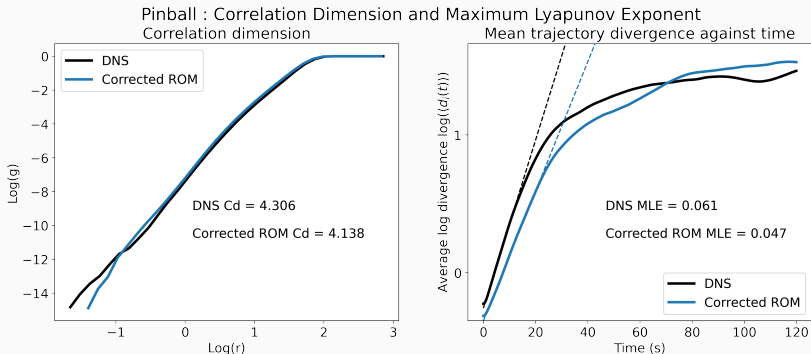
- The test trajectory presents statistics similar to the original attractor :



Probability density function of the amplitude of each mode over a trajectory of ≈ 1000 seconds. The CD-ROM results are presented in blue while the DNS is in black.

Attractor Statistics

- The test trajectory presents statistics similar to the original attractor :



Time Horizons

- We can learn a different λ for each dimension of the memory :

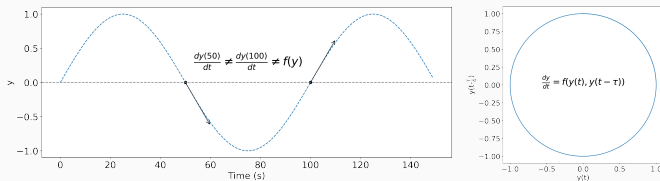
$$y(t) = \int_{-\infty}^t E(x(s))e^{\Lambda(s-t)} ds, \Lambda = \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_N \end{bmatrix}, N = \dim(y)$$

- Each λ_i defines the time horizon of each memory dimension, which can be computed as :

$$\tau_i = \frac{1}{\lambda_i}$$

Synthetic Experiment

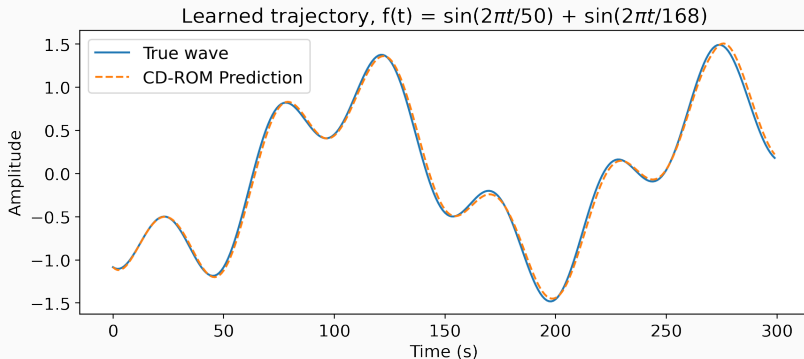
- Sine waves are a good test-case to assess the ability of the CD-ROM to learn the time-horizons :



- With a wave of period T , using a delay of $\tau = \frac{T}{4}$ yields a perfect circle in phase space.

Trajectory Fit

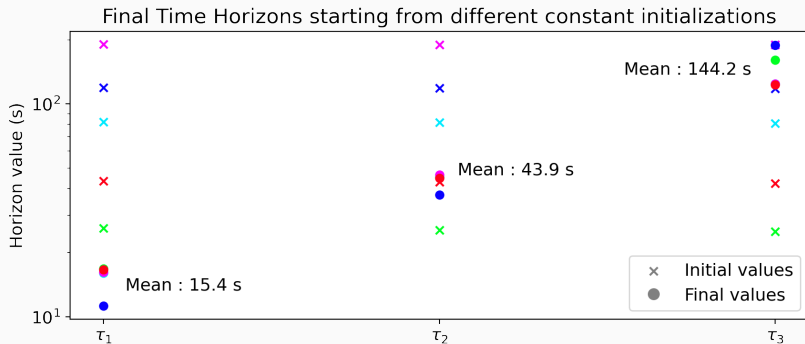
- We use a superposition of two waves of periods $T_1 = 50$, $T_2 = 168$:



- We need a memory of dimension 3 to represent the dynamics of the wave.

Learned Horizons

- Starting from varying initial conditions, the model learns similar horizons.



- $\tau_1 \approx 50/4$, $\tau_2 \approx 168/4$, $\tau_3 \approx 168$

Summary :

- We propose a novel, time-continuous, neural architecture for modeling partially observable systems.
- It provides a computationally inexpensive and data-efficient correction to POD-Galerkin models, illustrated here in different flow configurations.
- Relevant and physically consistent time-constants are extracted from observations of the system.

- CD-ROM: Complementary Deep - Reduced Order model, Menier *et al.*, <https://arxiv.org/abs/2202.10746>

References



B. R. Noack, K. Afanasiev, M. Morzyński, et al., "A hierarchy of low-dimensional models for the transient and post-transient cylinder wake", *Journal of Fluid Mechanics* **497**, 335–363 (2003).



R. T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud, "Neural ordinary differential equations", (2019).

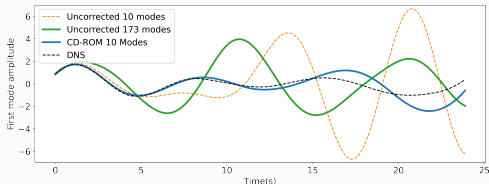


N. Deng, B. R. Noack, M. Morzynski, et al., "Low-order model for successive bifurcations of the fluidic pinball", *Journal of Fluid Mechanics* **884**, 10.1017/jfm.2019.959 (2019).

Annex Slides

Computational Costs

- Although the CD-ROM increases the computational costs, it remains less expensive than bigger models :

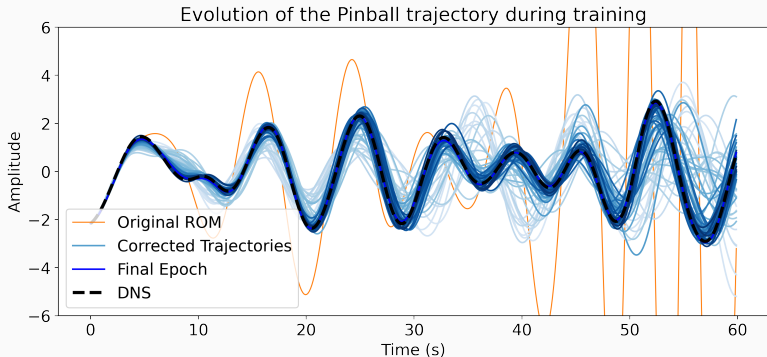


Integration times	
Uncorrected 10 modes	30 ms
Uncorrected 173 modes	550 ms
CD-ROM 10 modes	330 ms

Performance and computational cost of different models on a point outside the CD-ROM training basis.

Pinball Training

- This model can then be progressively corrected and stabilised through our approach :



Comparison with Reservoir Computing

- Echo State Networks are based on random matrices W_{in} and W :

$$Y_{t+1} = \tanh(W_{in} \cdot x_t + W \cdot Y_t)$$

$$s.t. \rho(W) < 1$$

- Only the output operator is optimised to map the output sequence (O) to the memories :

$$x_t = W_{out} \cdot Y_t$$

$$W_{out} = Y^\dagger O$$

Comparison with RC

- Ignoring the non-linearity, the ESN becomes a linear dynamical system :

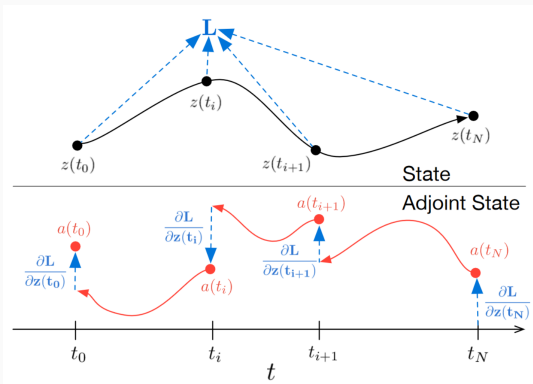
$$\begin{aligned} Y_{t+1} &= W_{in} \cdot x_t + W \cdot Y_t \\ &= Y_t + \Delta_t \frac{dY}{dt} \end{aligned} \implies \frac{dY}{dt} = \frac{W_{in}}{\Delta_t} \cdot x(t) + \frac{W - I}{\Delta_t} Y(t)$$

- The operator $\bar{W} = \frac{W-I}{\Delta_t}$ eigenvalues have negative real part. It plays the same memory dissipation role as the Λ matrix in our architecture :

$$\begin{aligned} \text{ESN :} & \quad \frac{dY}{dt} = \frac{W_{in}}{\Delta_t} \cdot x + \frac{W-I}{\Delta_t} Y \\ \text{Deep ROM :} & \quad \frac{dY}{dt} = \text{Enc}(x) - \Lambda Y \end{aligned}$$

Neural ODE

- With Neural ODEs⁴, we can back-propagate gradients through the simulation steps :



$$\frac{dz(t)}{dt} = f(z, t; \theta)$$

$$a(t) = \left. \frac{dL}{dz} \right|_t$$

$$\frac{da(t)}{dt} = -a(t) \frac{\partial f(z(t), t; \theta)}{\partial z}$$

$$\frac{dL}{d\theta} = - \int_{t_0}^{t_1} a(t) \frac{\partial f(z(t), t; \theta)}{\partial \theta}$$

⁴ R. T. Q. Chen et al., "Neural ordinary differential equations", (2019).

- In our case, the Neural ODE corresponds to the complete ROM :

$$z = \begin{pmatrix} \alpha \\ y \end{pmatrix}$$

$$f(z, t; \theta) = \begin{array}{cc} \frac{1}{Re} \bar{K} \alpha - \alpha^T \bar{N} \alpha & + \mathcal{NN}(y; \theta_1) \\ E(\alpha; \theta_2) & - \Lambda y \end{array}$$

- Because the model is fully expressed with differentiable objects, the required jacobians can be easily evaluated through automatic differentiation.

Cylinder Hyperparameters

ReLU activation	
<hr/>	
Encoder Network	
<hr/>	
Layer 1	3 <i>neurons</i>
Layer 2	11 <i>neurons</i>
Layer 3	19 <i>neurons</i>
Layer 4	27 <i>neurons</i>
<hr/>	
Correction Network	
<hr/>	
Layer 1	30 <i>neurons</i>
Layer 2	30 <i>neurons</i>
Layer 3	30 <i>neurons</i>
Layer 4	3 <i>neurons</i>

Pinball Hyperparameters

Swish activation

Encoder Network

Layer 1 10 *neurons*

Layer 2 36 *neurons*

Layer 3 63 *neurons*

Layer 4 90 *neurons*

Correction Network

Layer 1 100 *neurons*

Layer 2 500 *neurons*

Layer 3 500 *neurons*

Layer 4 10 *neurons*

Takens Theorem

- Discrete time delays are unsuitable for classical time integrators :

