Interpretable Recurrent Neural Networks for Reconstructing Nonlinear Dynamical Systems from Time Series Observations

Daniel Durstewitz

Dept. of Theoretical Neuroscience

Central Institute for Mental Health/ Heidelberg University

Problem Setting

$X = \{x_t\},\ x_t \in \mathbb{R}^N,\ t = 1 \ldots T$

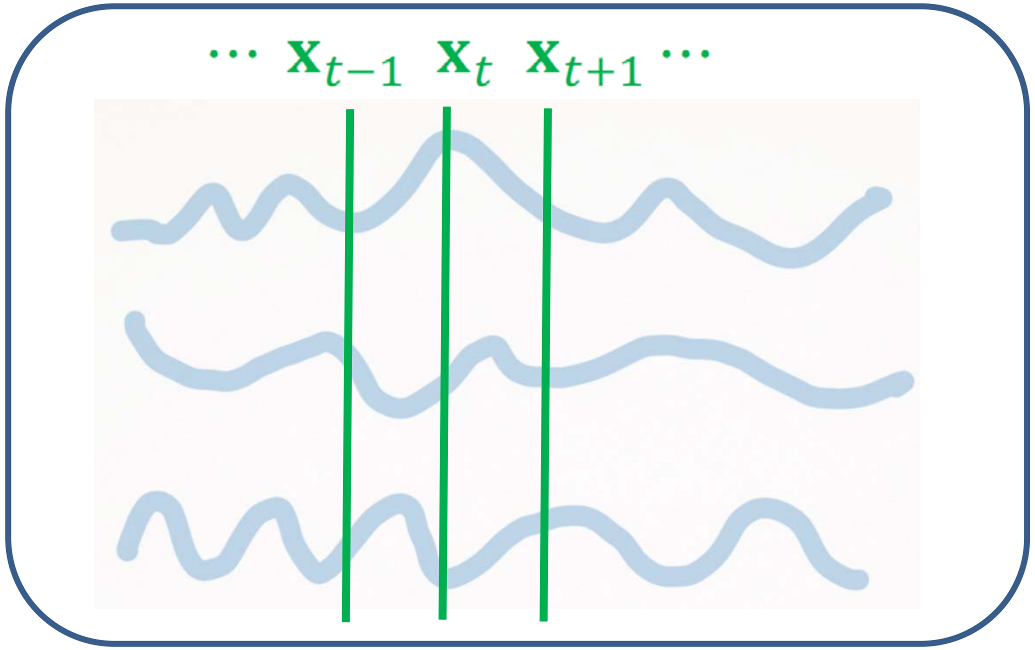
Generation $p(x_t \mid z_t)$ ↑

$\dot{z} = f(z, t)$

$z_t = F(z_{t-1}, s_t)$

$\overset{?}{\Longleftrightarrow}$

computation



What do we mean by reconstruction?

Systems identification

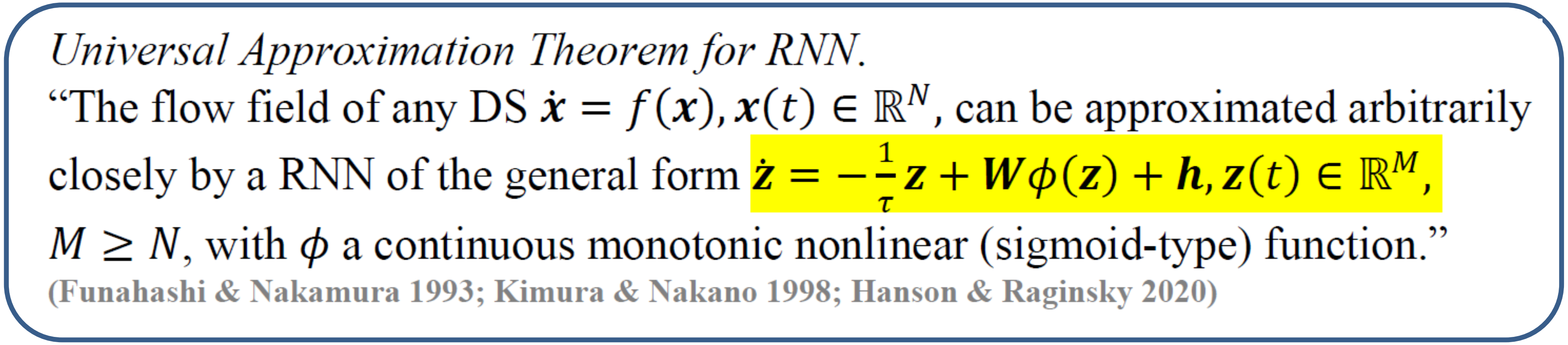
$$\dot{x} = f(x) = \sum_{i=0}^{p} \hat{\alpha}_i \, x^i$$

Posterior inference

$$\dot{x} = f(x) = \sum_{i=0}^{p} \alpha_i \, x^i , \quad p(x \mid \text{"obs."})$$

DS reconstruction/ approximation

$$\dot{x} = f(x) \approx \hat{f}_\theta(x) = A\phi(Wx+h)$$

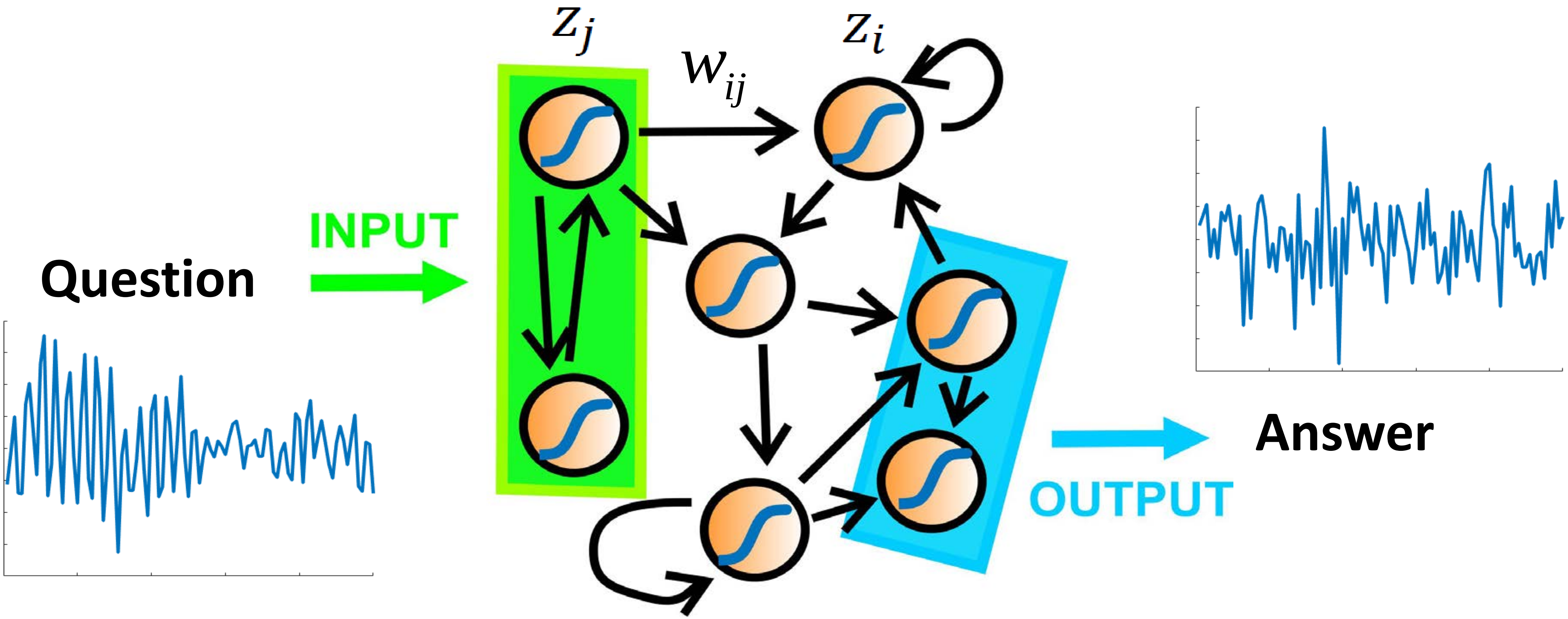


Universal Approximation Theorem for RNN.

"The flow field of any DS $\dot{\boldsymbol{x}} = f(\boldsymbol{x}), \boldsymbol{x}(t) \in \mathbb{R}^N$, can be approximated arbitrarily closely by a RNN of the general form $\dot{\boldsymbol{z}} = -\frac{1}{\tau}\boldsymbol{z} + \boldsymbol{W}\phi(\boldsymbol{z}) + \boldsymbol{h}, \boldsymbol{z}(t) \in \mathbb{R}^M$, $M \geq N$, with ϕ a continuous monotonic nonlinear (sigmoid-type) function."

(Funahashi & Nakamura 1993; Kimura & Nakano 1998; Hanson & Raginsky 2020)

What is a Recurrent Neural Network?



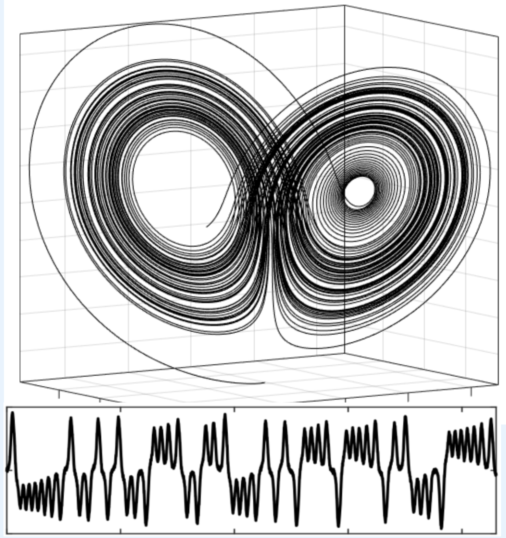
$$\boldsymbol{z}_t = \phi(\boldsymbol{W}\boldsymbol{z}_{t-1} + \boldsymbol{h} + \boldsymbol{C}\boldsymbol{s}_t)$$, e.g. $\phi(\boldsymbol{z}) = \tanh(\boldsymbol{z})$

external input

Nonlinear dynamical systems reconstruction

dynamical system

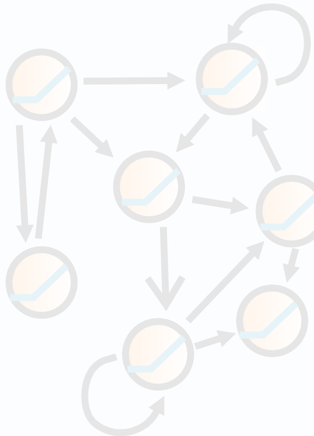
$$\dot{x}_1 = a(x_2 - x_1)$$
$$\dot{x}_2 = x_1(b - x_3) - x_2$$
$$\dot{x}_3 = x_1 x_2 - c x_3$$



train

RNN

$$z_t = F_\theta(z_{t-1}, s_t)$$



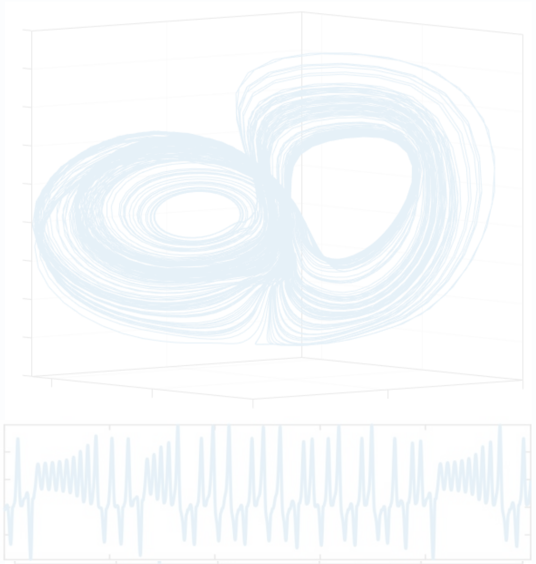
universal approximator of dynamical systems

(Hanson & Raginsky, 2020)

generate

simulated trajectory

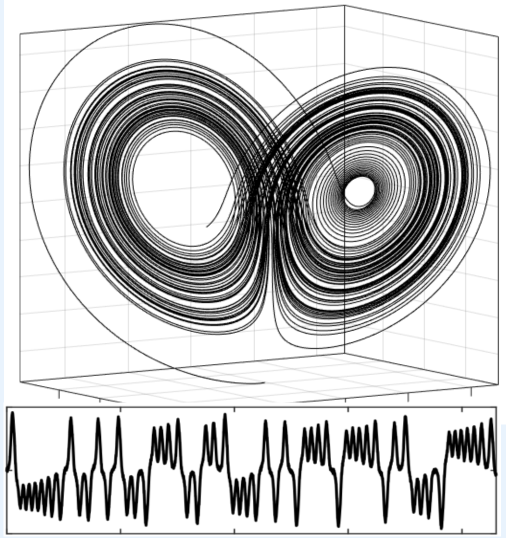
$$z_t = A z_{t-1} + W \max(z_{t-1}, 0) + \mathrm{h}$$



Nonlinear dynamical systems reconstruction

dynamical system

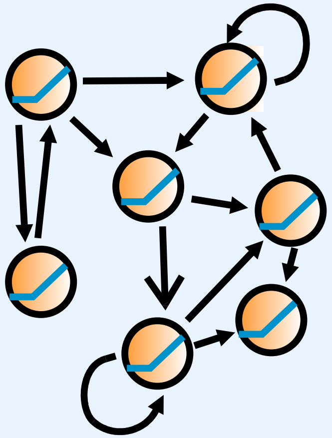
$$\dot{x}_1 = a(x_2 - x_1)$$
$$\dot{x}_2 = x_1(b - x_3) - x_2$$
$$\dot{x}_3 = x_1 x_2 - c x_3$$



train

RNN

$$\boldsymbol{z}_t = F_\theta(\boldsymbol{z}_{t-1}, \boldsymbol{s}_t)$$



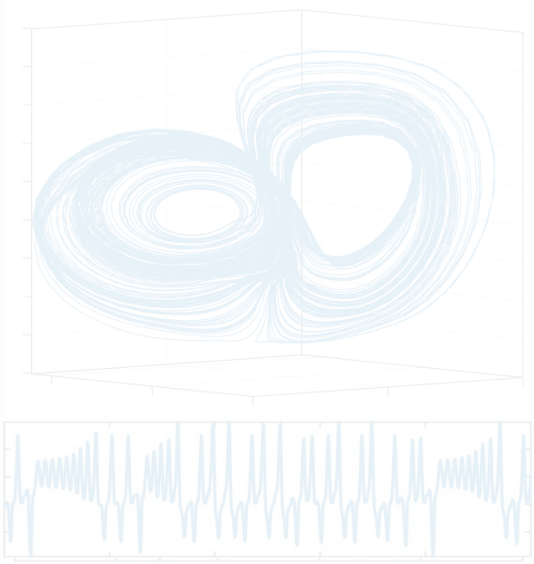
universal approximator of dynamical systems

(Hanson & Raginsky, 2020)

generate

simulated trajectory

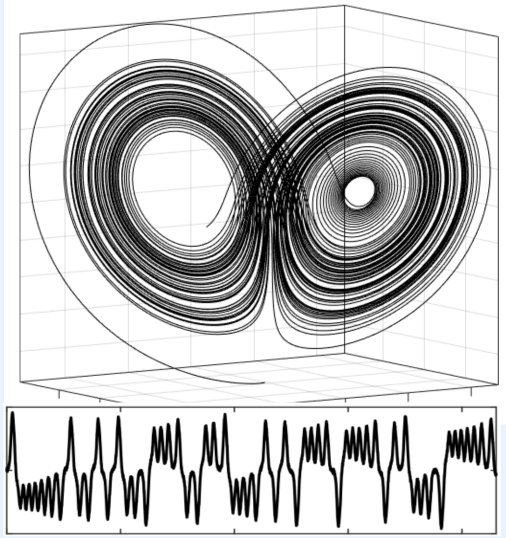
$$z_t = \boldsymbol{A}z_{t-1} + \boldsymbol{W}\max(\boldsymbol{z}_{t-1}, \mathbf{0}) + \mathrm{h}$$



Nonlinear dynamical systems reconstruction

dynamical system

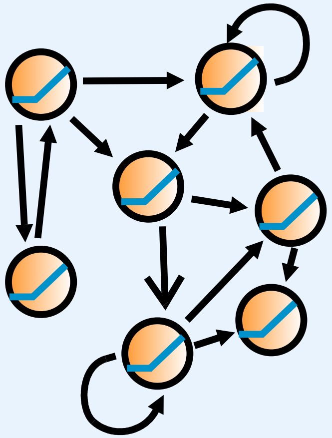
$$\begin{aligned}\dot{x}_1 &= a(x_2 - x_1)\\ \dot{x}_2 &= x_1(b - x_3) - x_2\\ \dot{x}_3 &= x_1 x_2 - c x_3\end{aligned}$$



train

RNN

$$\boldsymbol{z}_t = F_\theta(\boldsymbol{z}_{t-1}, \boldsymbol{s}_t)$$



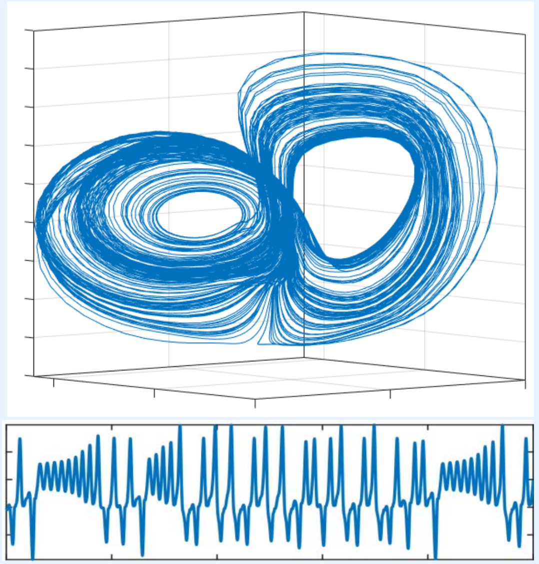
universal approximator of dynamical systems

(Hanson & Raginsky, 2020)

generate

simulated trajectory

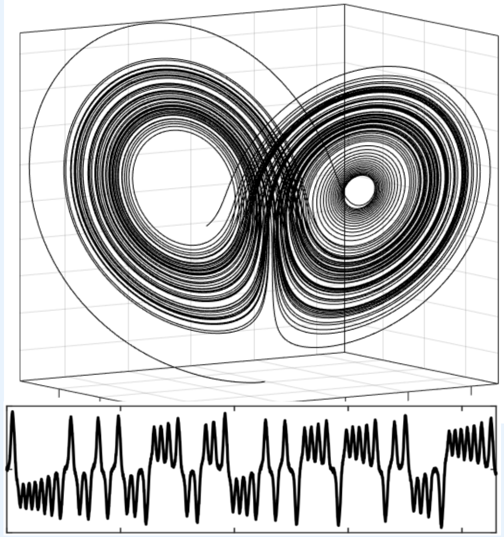
$$\boldsymbol{z}_t = \boldsymbol{A}\boldsymbol{z}_{t-1} + \boldsymbol{W}\max(\boldsymbol{z}_{t-1}, \boldsymbol{0}) + \mathbf{h}$$



Nonlinear dynamical systems reconstruction

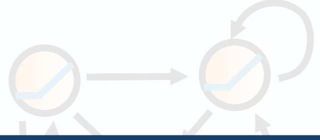
dynamical system

$$\dot{x}_1 = a(x_2 - x_1)$$
$$\dot{x}_2 = x_1(b - x_3) - x_2$$
$$\dot{x}_3 = x_1 x_2 - c x_3$$



RNN

$$z_t = F_\theta(z_{t-1}, s_t)$$



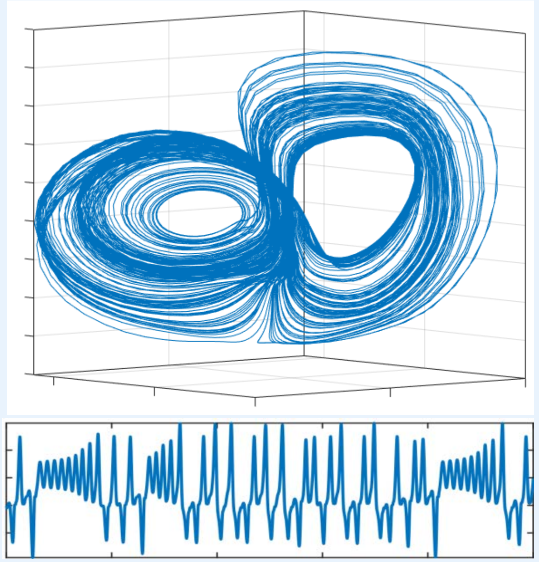
universal approximator of dynamical systems

(Hanson & Raginsky, 2020)

Same geometrical and temporal structure

simulated trajectory

$$\boldsymbol{z}_t = \boldsymbol{A}\boldsymbol{z}_{t-1} + \boldsymbol{W}\max(\boldsymbol{z}_{t-1}, \boldsymbol{0}) + \mathbf{h}$$



How do we train Recurrent Neural Networks?

Given data: $\mathbf{X} = \{\mathbf{x}_t\},\ \mathbf{x}_t \in \mathbb{R}^N, t = 1 \ldots T$

model: $\mathbf{z}_t = F_\theta(\mathbf{z}_{t-1}, \mathbf{s}_t), \hat{\mathbf{x}}_t = g_\theta(\mathbf{z}_t)$

- **Mean Squared Error (MSE) loss:** $\boldsymbol{\theta}^* := \arg\min_{\boldsymbol{\theta}} \left\{ \sum_{t=1}^{T} \left\| \boldsymbol{x}_t - g_{\boldsymbol{\theta}}(\boldsymbol{z}_t) \right\|_2^2 \right\}$
- **Maximum Likelihood:** $\boldsymbol{\theta}^* := \arg\max_{\boldsymbol{\theta}} \{\log p(\mathbf{X}|\boldsymbol{\theta})\}$
- **Bayesian inference:** $p(\boldsymbol{\theta}, \mathbf{Z}|\mathbf{X}) = \dfrac{p(\mathbf{X}|\boldsymbol{\theta}, \mathbf{Z}) p(\boldsymbol{\theta}, \mathbf{Z})}{\int p(\boldsymbol{\theta}, \mathbf{Z}, \mathbf{X}) d\boldsymbol{\theta} d\mathbf{Z}}$

- **Deterministic models: Gradient Descent (RTRL, BPTT), 2nd-order methods**
- **Probabilistic models: EM, Variational Inference, MCMC**

How do we train Recurrent Neural Networks?

Given data: $\mathbf{X} = \{\mathbf{x}_t\},\ \mathbf{x}_t \in \mathbb{R}^N, t = 1 \ldots T$

model: $\mathbf{z}_t = F_\theta(\mathbf{z}_{t-1}, \mathbf{s}_t), \hat{\mathbf{x}}_t = g_\theta(\mathbf{z}_t)$

- **Mean Squared Error (MSE) loss:** $\boldsymbol{\theta}^* := \arg\min_{\boldsymbol{\theta}} \left\{ \sum_{t=1}^{T} \left\| \boldsymbol{x}_t - g_{\boldsymbol{\theta}}(\boldsymbol{z}_t) \right\|_2^2 \right\}$
- **Maximum Likelihood:** $\boldsymbol{\theta}^* := \arg\max_{\boldsymbol{\theta}} \{\log p(\mathbf{X}|\boldsymbol{\theta})\}$
- **Bayesian inference:** $p(\boldsymbol{\theta}, \mathbf{Z}|\mathbf{X}) = \dfrac{p(\mathbf{X}|\boldsymbol{\theta}, \mathbf{Z})p(\boldsymbol{\theta}, \mathbf{Z})}{\int p(\boldsymbol{\theta}, \mathbf{Z}, \mathbf{X})d\boldsymbol{\theta}d\mathbf{Z}}$

- **Deterministic models: Gradient Descent (RTRL, BPTT), 2nd-order methods**
- **Probabilistic models: EM, Variational Inference, MCMC**

“Vanishing/ exploding gradient problem”

“Inputs”: $\boldsymbol{S} = \{\boldsymbol{s}_t\}$ “Outputs”: $\widetilde{\boldsymbol{Z}} = \{\tilde{\boldsymbol{z}}_t\}, t = 1 \ldots T$

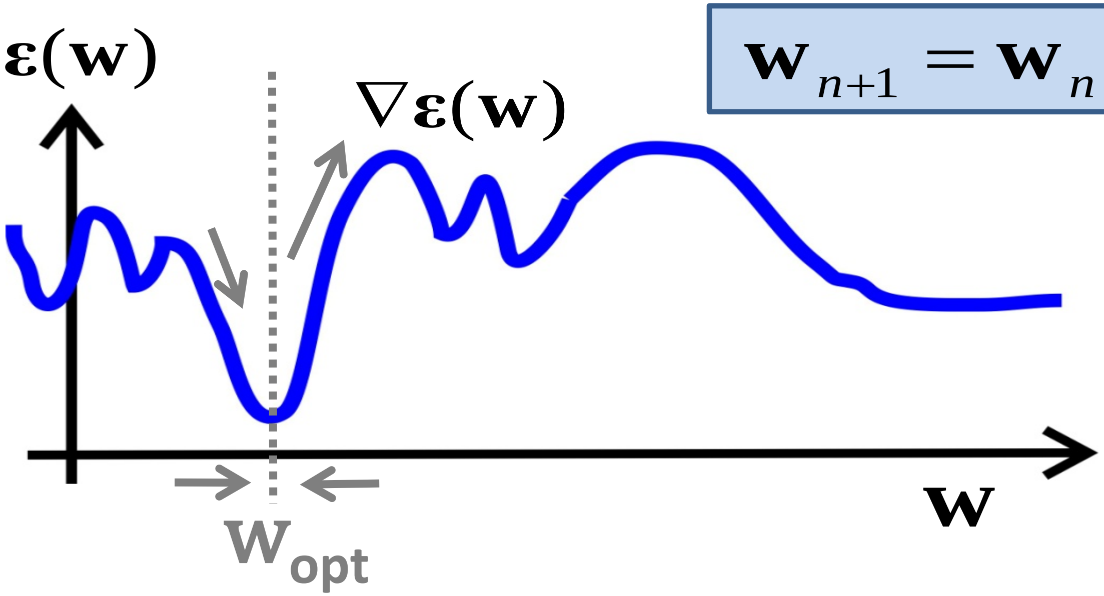
RNN $\boldsymbol{z}_t \in \mathbb{R}^M$

$$\boldsymbol{z}_t = \phi(\boldsymbol{W}\boldsymbol{z}_{t-1} + \boldsymbol{h} + \boldsymbol{C}\boldsymbol{s}_t)$$

Loss function

$$\boldsymbol{\varepsilon} = \sum_{r=1}^{R} \sum_{t=1}^{T} \|\tilde{\boldsymbol{z}}_t - \boldsymbol{z}_t\|_2^2$$

$$\mathbf{w}_{n+1} = \mathbf{w}_n - \gamma \nabla \boldsymbol{\varepsilon}(\mathbf{w}_n)$$



“Vanishing/ exploding gradient problem”

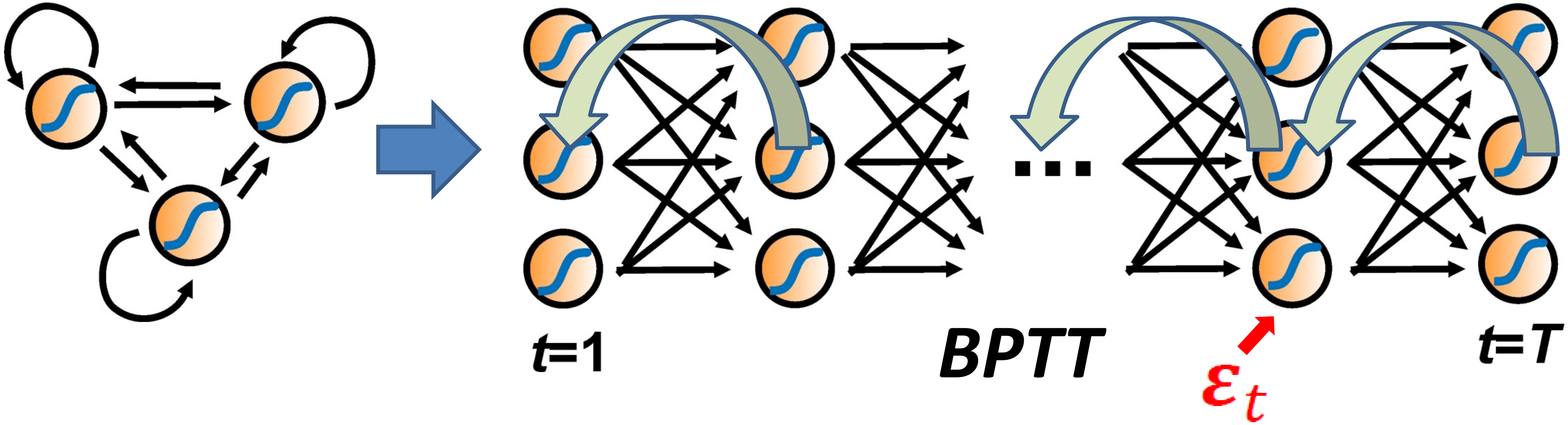
“Inputs”: $\boldsymbol{S} = \{\boldsymbol{s}_t\}$ “Outputs”: $\widetilde{\boldsymbol{Z}} = \{\tilde{\boldsymbol{z}}_t\}\,, t = 1 \dots T$

RNN $\boldsymbol{z}_t \in \mathbb{R}^M$

$$\boldsymbol{z}_t = \phi(\boldsymbol{W}\boldsymbol{z}_{t-1} + \boldsymbol{h} + \boldsymbol{C}\boldsymbol{s}_t)$$

Loss function

$$\varepsilon = \sum_{r=1}^{R} \sum_{t=1}^{T} \|\tilde{\boldsymbol{z}}_t - \boldsymbol{z}_t\|_2^2$$



“Vanishing/ exploding gradient problem”

“Inputs”: $\boldsymbol{S} = \{\boldsymbol{s}_t\}$ “Outputs”: $\widetilde{\boldsymbol{Z}} = \{\tilde{\boldsymbol{z}}_t\}\,, t = 1 \ldots T$

RNN $\boldsymbol{z}_t \in \mathbb{R}^M$

$$\boldsymbol{z}_t = \phi(\boldsymbol{W}\boldsymbol{z}_{t-1} + \boldsymbol{h} + \boldsymbol{C}\boldsymbol{s}_t)$$

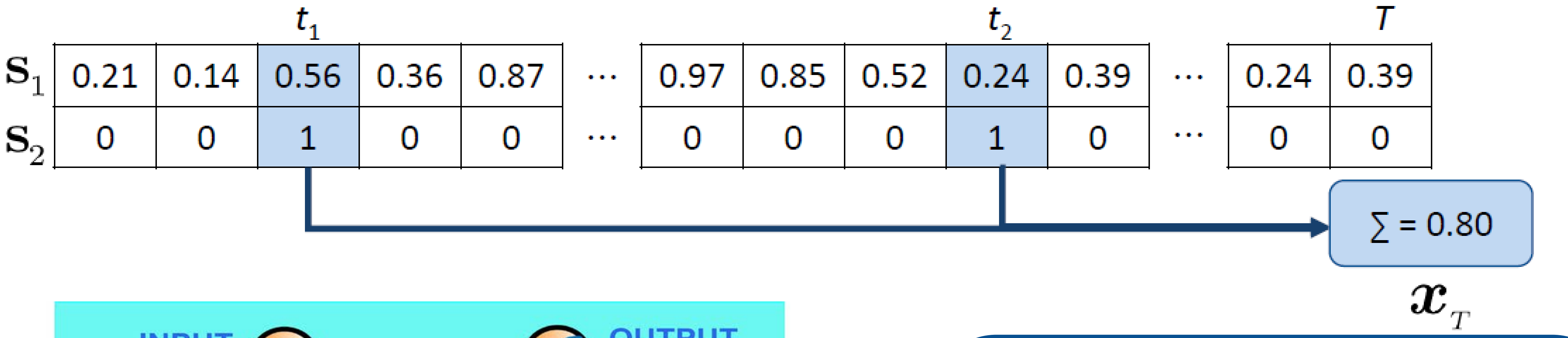
Loss function

$$\boldsymbol{\varepsilon} = \sum_{r=1}^{R} \sum_{t=1}^{T} ||\tilde{\boldsymbol{z}}_t - \boldsymbol{z}_t||_2^2$$

$$\frac{\partial \varepsilon}{\partial W_{ij}} = \sum_{1 \leq t \leq T} \frac{\partial \varepsilon_t}{\partial W_{ij}}$$

$$\overset{?}{\lessgtr} 1$$

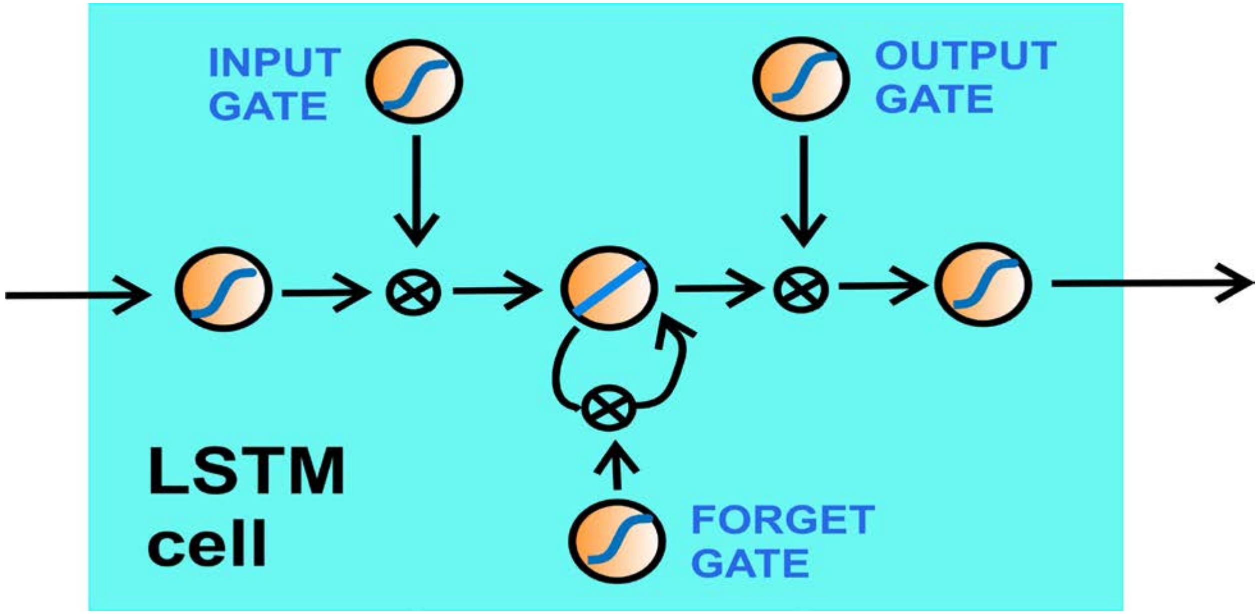
Benchmark example: Addition problem



			t_1							t_2				T
$\mathbf{S}_1$	0.21	0.14	0.56	0.36	0.87	⋯	0.97	0.85	0.52	0.24	0.39	⋯	0.24	0.39
$\mathbf{S}_2$	0	0	1	0	0	⋯	0	0	0	1	0	⋯	0	0

$\sum = 0.80$

$\boldsymbol{x}_T$



Hochreiter & Schmidhuber (1997)

$$
\begin{aligned}
\boldsymbol{i}_t^l &= \sigma(W_{\boldsymbol{i}}^l[\chi_t; \boldsymbol{h}_{t-1}^l; \boldsymbol{h}_t^{l-1}] + \boldsymbol{b}_{\boldsymbol{i}}^l) \\
\boldsymbol{f}_t^l &= \sigma(W_{\boldsymbol{f}}^l[\chi_t; \boldsymbol{h}_{t-1}^l; \boldsymbol{h}_t^{l-1}] + \boldsymbol{b}_{\boldsymbol{f}}^l) \\
\boldsymbol{s}_t^l &= \boldsymbol{f}_t^l \boldsymbol{s}_{t-1}^l + \boldsymbol{i}_t^l \tanh(W_{\boldsymbol{s}}^l[\chi_t; \boldsymbol{h}_{t-1}^l; \boldsymbol{h}_t^{l-1}] + \boldsymbol{b}_{\boldsymbol{s}}^l) \\
\boldsymbol{o}_t^l &= \sigma(W_{\boldsymbol{o}}^l[\chi_t; \boldsymbol{h}_{t-1}^l; \boldsymbol{h}_t^{l-1}] + \boldsymbol{b}_{\boldsymbol{o}}^l) \\
\boldsymbol{h}_t^l &= \boldsymbol{o}_t^l \tanh(\boldsymbol{s}_t^l)
\end{aligned}
$$

Our approach: Piecewise-Linear (PL) RNN

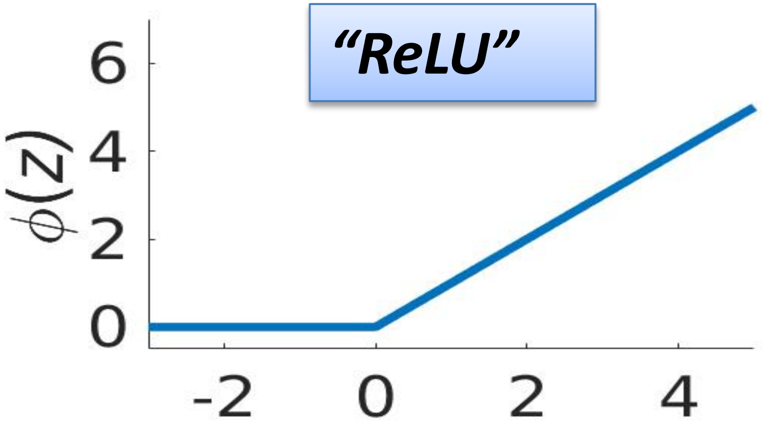
$$\boldsymbol{z}_t = \boldsymbol{A}\boldsymbol{z}_{t-1} + \boldsymbol{W}\max(\boldsymbol{z}_{t-1}, \mathbf{0}) + \mathbf{C}\mathbf{s_t} + \mathbf{h} + \varepsilon_{\mathbf{t}}$$

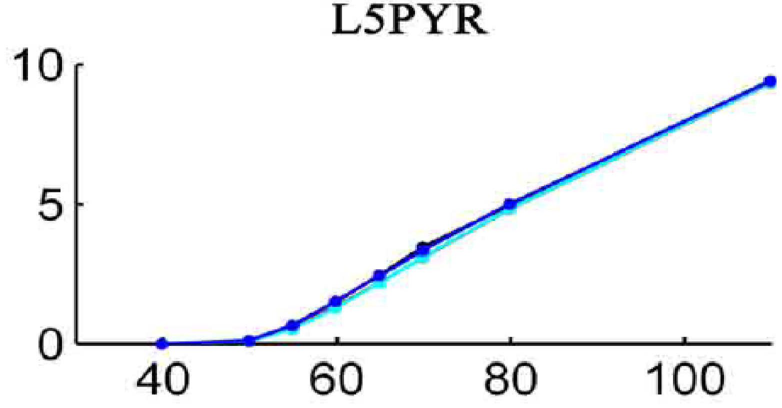
latent states	$\boldsymbol{z}_t \in \mathbb{R}^{M \times 1}$
diag (auto-regression)	$\boldsymbol{A} \in \mathbb{R}^{M \times M}$
connection wgt.s	$\boldsymbol{W} \in \mathbb{R}^{M \times M}$
input wgt.s	$\boldsymbol{C} \in \mathbb{R}^{M \times K}$
bias term	$\boldsymbol{h} \in \mathbb{R}^{M \times 1}$
external inputs	$\boldsymbol{s}_t \in \mathbb{R}^{K \times 1}$
i.i.d. Gaussian noise	$\varepsilon_t \sim \mathcal{N}(0, \boldsymbol{\Sigma})$

Our approach: Piecewise-Linear (PL) RNN

$$\boldsymbol{z}_t = \boldsymbol{A}\boldsymbol{z}_{t-1} + \boldsymbol{W}\max(\boldsymbol{z}_{t-1}, \mathbf{0}) + \mathbf{C}\mathbf{s}_\mathbf{t} + \mathbf{h} + \varepsilon_\mathbf{t}$$

- **Physiological motivation**





- **Universal approximation properties still hold**

(Koiran et al. 1994, *Theo Comp Sci*; Lu et al. 2017, *NeurIPS*; Storace et al. 2003, *Intl J Circuit Theo Appl*)

- **PL structure allows for efficient inference/ training**

(Durstewitz 2017, *PLoS Comp Biol*; Koppe et al. 2019, *PLoS Comp Biol*)

PLRNN is a PWL map: DS properties accessible

- **Fixed points, cycles, and their stability analytically accessible**

(Schmidt et al. 2021, *ICLR*)

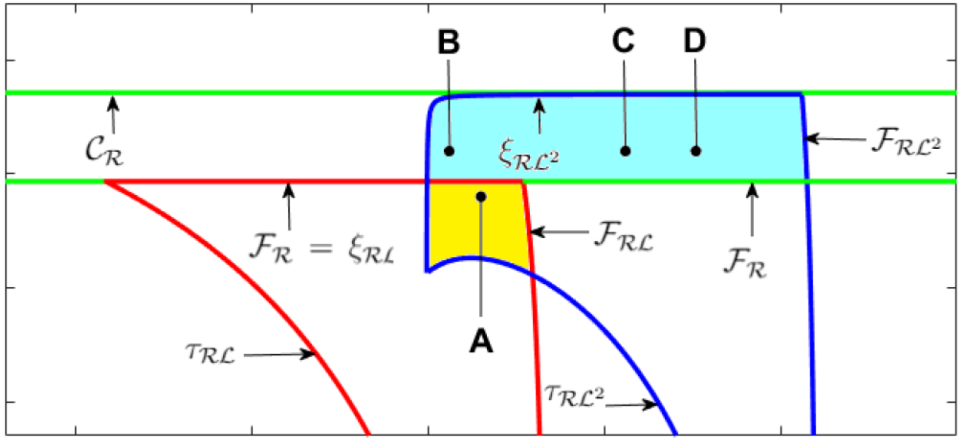
$$\boldsymbol{z}^{*1} = \left(\boldsymbol{I} - \boldsymbol{W}_{\Omega(t^{*1})}\right)^{-1} \boldsymbol{h}$$

$$\boldsymbol{z}^{*k} = \left(\boldsymbol{I} - \prod_{i=1}^{k} \boldsymbol{W}_{\Omega(t^{*k}-i)}\right)^{-1} \times \left[\sum_{j=2}^{k} \prod_{i=1}^{k-j+1} \boldsymbol{W}_{\Omega(t^{*k}-i)} + \boldsymbol{I}\right] \boldsymbol{h}$$

with Jacobians $\boldsymbol{W}_{\Omega(\mathrm{t})} := \boldsymbol{A} + \boldsymbol{W}\boldsymbol{D}_{\Omega(\mathrm{t})}$

- **Important classes of bifurcations relatively well described (in low dim)**

(Monfared & Durstewitz 2020, *Nonlinear Dynamics*; Monfared, Patra, Durstewitz, *in prep*)



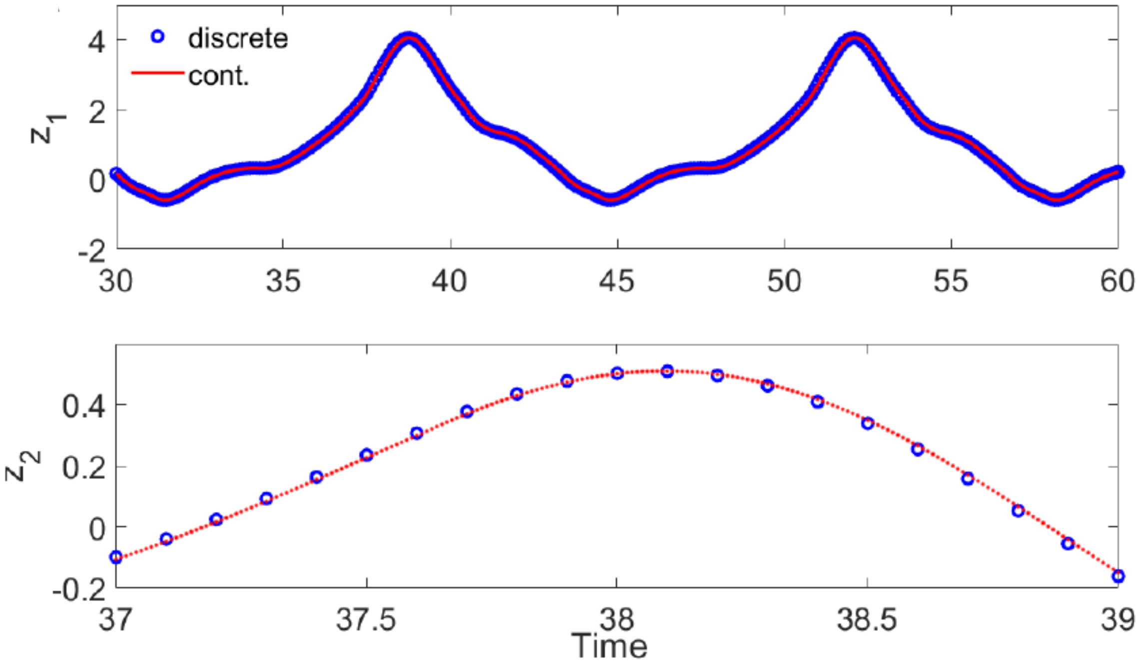
PLRNN is a PWL map: DS properties accessible

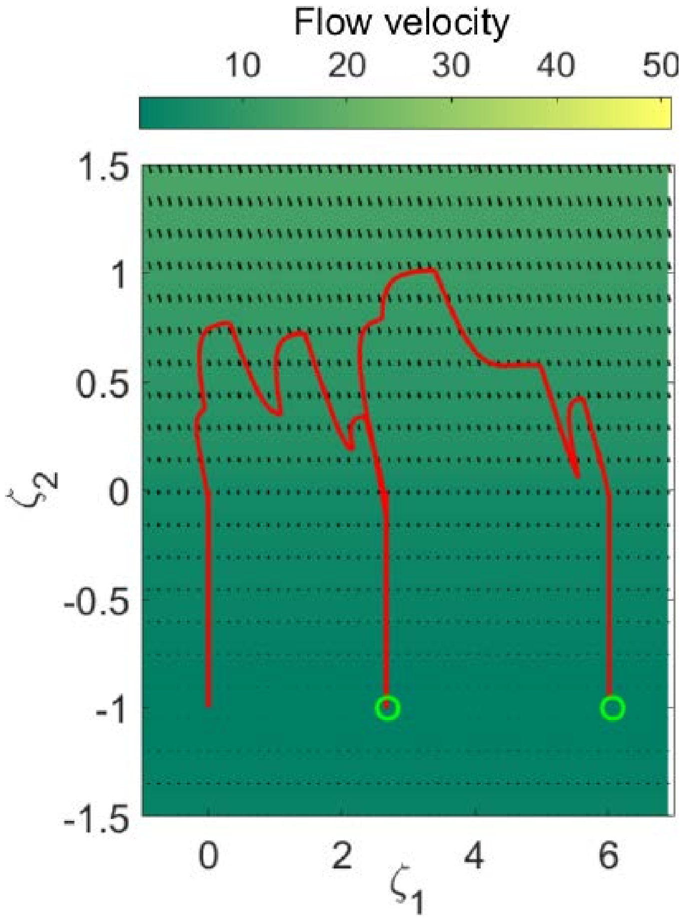
- **Discrete-time PLRNN can be converted into mathematically equivalent continuous-time PLRNN**

(Monfared & Durstewitz 2020, *ICML*)

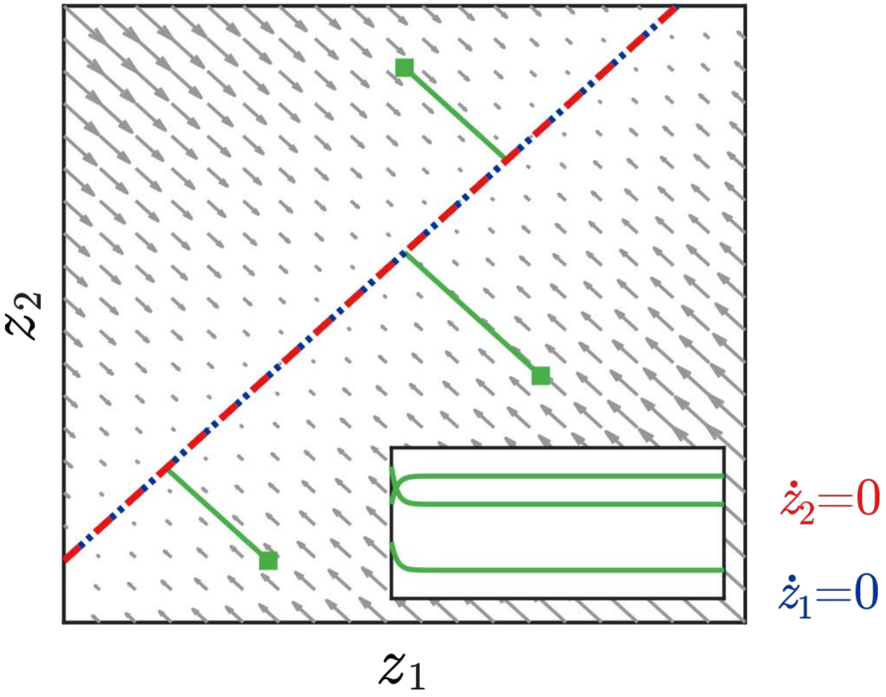
$$\mathbf{z}_{t+1} = F(\mathbf{z}_t) \ , \ \dot{\boldsymbol{\zeta}} = G(\boldsymbol{\zeta})$$

$$\mathbf{z}_0 = \boldsymbol{\zeta}(0) \ , \ \mathbf{z}_1 = F(\mathbf{z}_0) = \boldsymbol{\zeta}(\Delta t)$$





How can we solve issues in training while keeping the model simple?

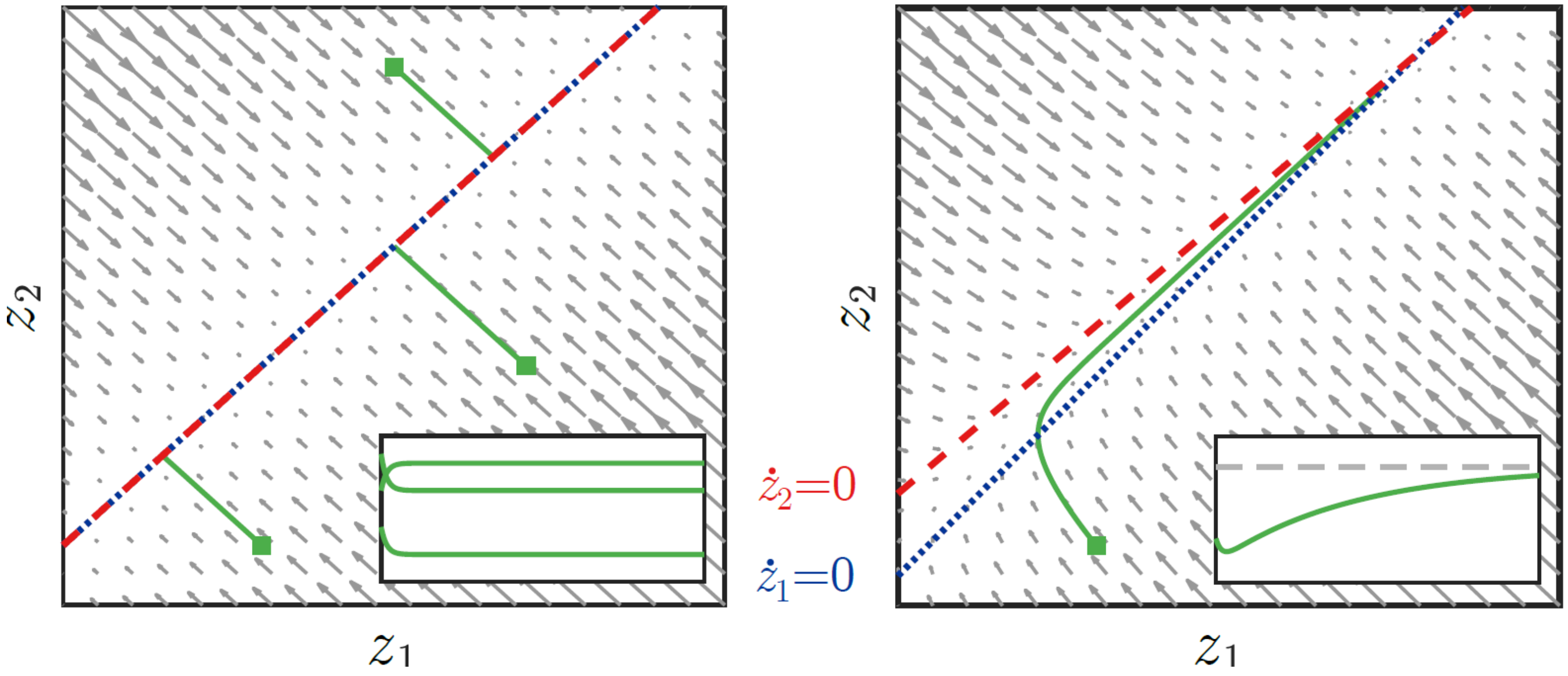


Perfect line attractor retains states indefinitely

Slightly ‘detuned’ line attractor enables arbitrary time constants

(Schmidt et al. 2021, *ICLR*)

Manifold-attractors for long memory & slow time scales

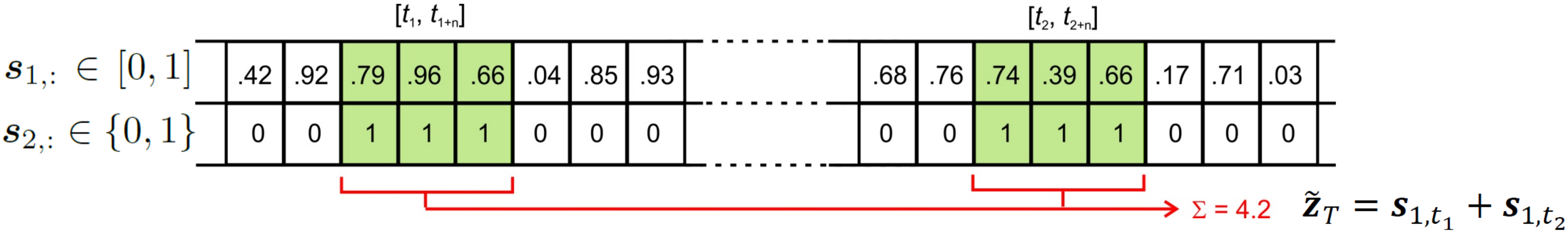


Perfect line attractor retains states indefinitely

Slightly ‘detuned’ line attractor enables arbitrary time constants

(Schmidt et al. 2021, *ICLR*)

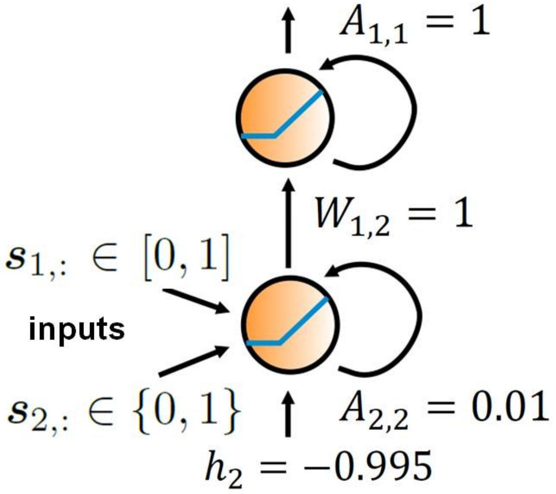
Manifold-attractors for long memory & slow time scales

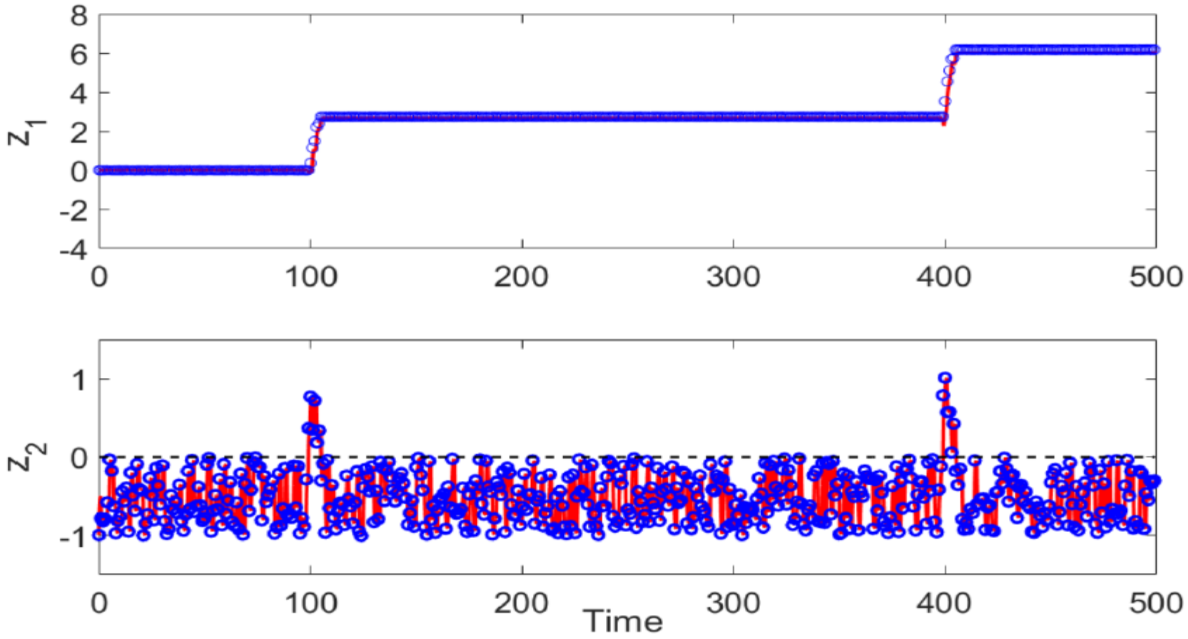


$[t_1, t_{1+n}]$ $[t_2, t_{2+n}]$

$s_{1,:} \in [0,1]$	.42	.92	.79	.96	.66	.04	.85	.93	- - - - - - - -	.68	.76	.74	.39	.66	.17	.71	.03
$s_{2,:} \in \{0,1\}$	0	0	1	1	1	0	0	0	- - - - - - - -	0	0	1	1	1	0	0	0

Σ = 4.2 $\tilde{z}_T = s_{1,t_1} + s_{1,t_2}$



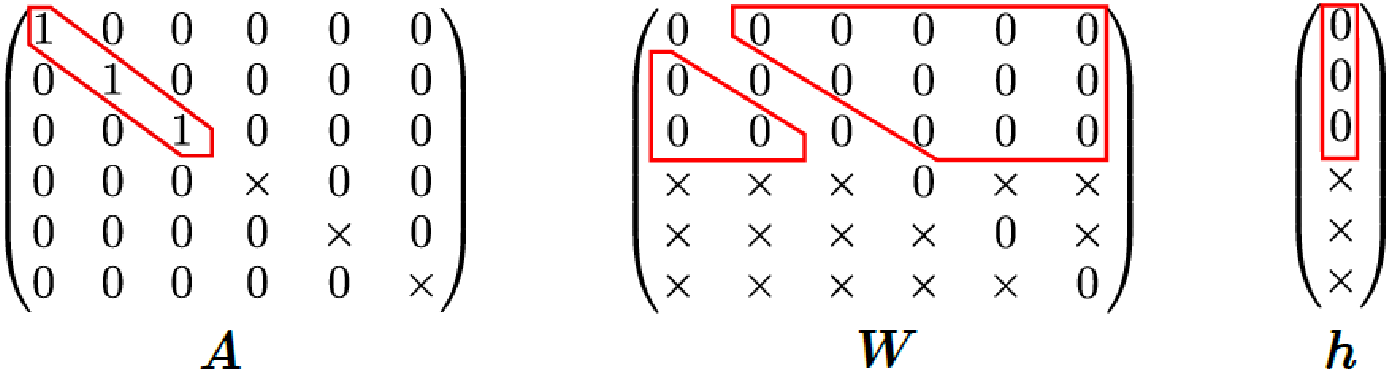


(Monfared & Durstewitz 2020, *ICML*)

Manifold-Attractor regularization

$$\boldsymbol{z}_t = \boldsymbol{A}\boldsymbol{z}_{t-1} + \boldsymbol{W}\max(\boldsymbol{z}_{t-1}, \mathbf{0}) + \mathbf{C}\mathbf{s_t} + \mathbf{h} + \boldsymbol{\varepsilon}_\mathbf{t}$$

$$\mathrm{L}_{\mathrm{reg}} = \tau_A \sum_{i=1}^{M_{\mathrm{reg}}} (A_{i,i} - 1)^2 + \tau_W \sum_{i=1}^{M_{\mathrm{reg}}} \sum_{\substack{j=1 \\ j \neq i}}^{M} W_{i,j}^2 + \tau_h \sum_{i=1}^{M_{\mathrm{reg}}} h_i^2$$



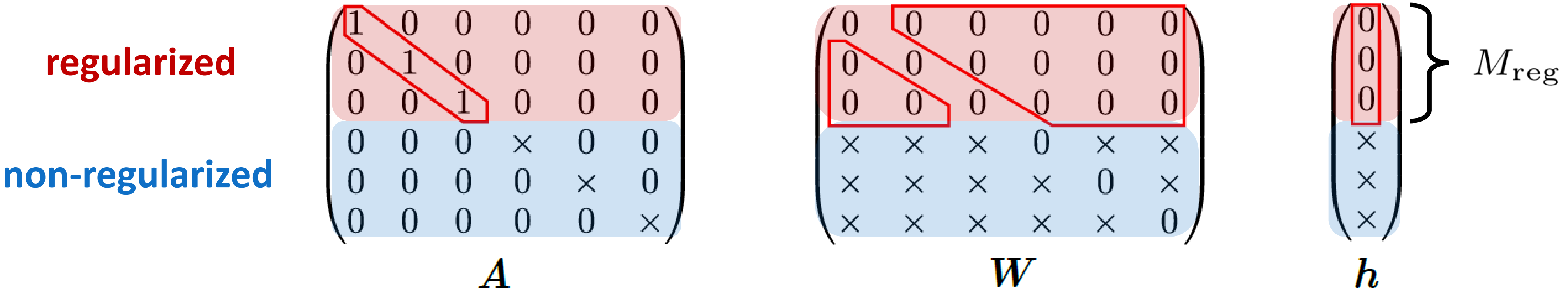
$$\boldsymbol{A} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & \times & 0 & 0 \\ 0 & 0 & 0 & 0 & \times & 0 \\ 0 & 0 & 0 & 0 & 0 & \times \end{pmatrix} \quad \boldsymbol{W} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \times & \times & \times & 0 & \times & \times \\ \times & \times & \times & \times & 0 & \times \\ \times & \times & \times & \times & \times & 0 \end{pmatrix} \quad \boldsymbol{h} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ \times \\ \times \\ \times \end{pmatrix}$$

(Schmidt et al. 2021, *ICLR*)

Manifold-Attractor regularization

$$\boldsymbol{z}_t = \boldsymbol{A}\boldsymbol{z}_{t-1} + \boldsymbol{W}\max(\boldsymbol{z}_{t-1}, \boldsymbol{0}) + \mathbf{C}\mathbf{s}_{\mathbf{t}} + \mathbf{h} + \boldsymbol{\varepsilon}_{\mathbf{t}}$$

$$\mathrm{L}_{\mathrm{reg}} = \tau_A \sum_{i=1}^{M_{\mathrm{reg}}} (A_{i,i} - 1)^2 + \tau_W \sum_{i=1}^{M_{\mathrm{reg}}} \sum_{\substack{j=1 \\ j \neq i}}^{M} W_{i,j}^2 + \tau_h \sum_{i=1}^{M_{\mathrm{reg}}} h_i^2$$

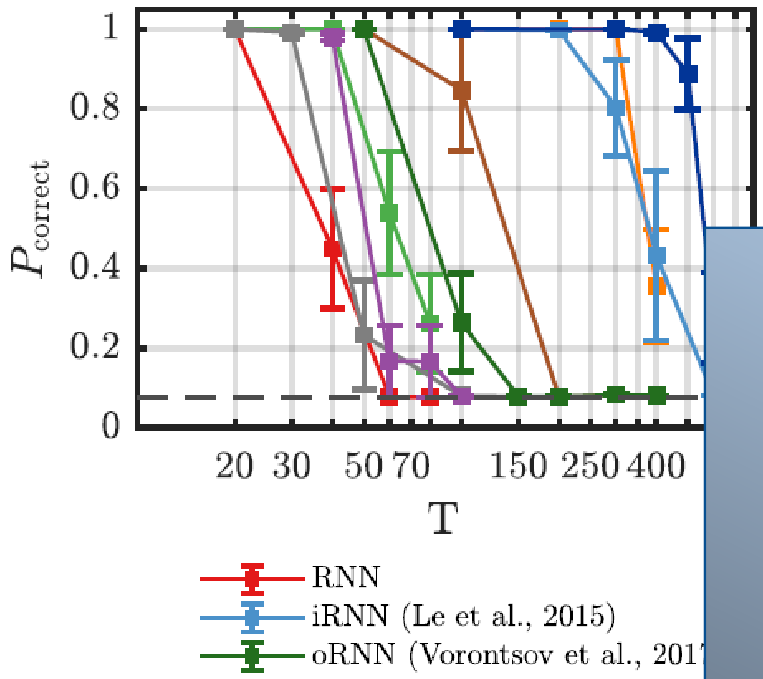


$$\boldsymbol{A} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & \times & 0 & 0 \\ 0 & 0 & 0 & 0 & \times & 0 \\ 0 & 0 & 0 & 0 & 0 & \times \end{pmatrix} \quad \boldsymbol{W} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \times & \times & \times & 0 & \times & \times \\ \times & \times & \times & \times & 0 & \times \\ \times & \times & \times & \times & \times & 0 \end{pmatrix} \quad \boldsymbol{h} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ \times \\ \times \\ \times \end{pmatrix}$$

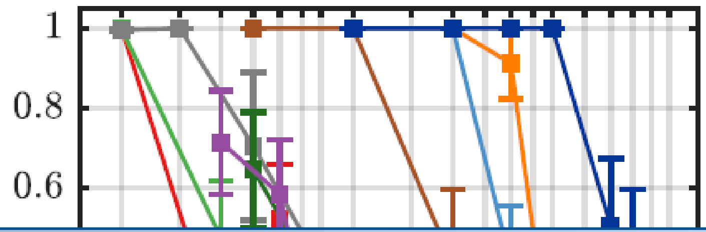
(Schmidt et al. 2021, *ICLR*)

Supervised: Classical ML benchmarks

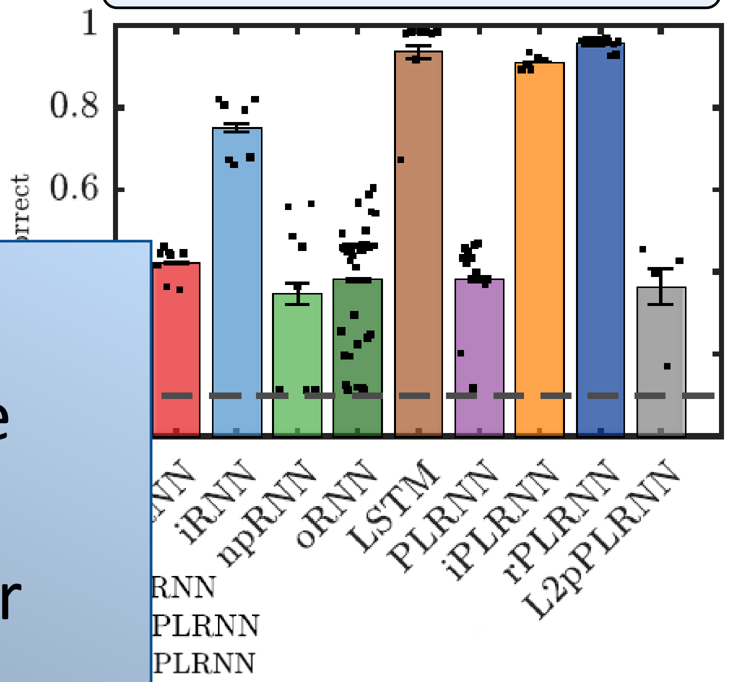
Addition task



Multiplication task



Sequential MNIST task



Bottom line

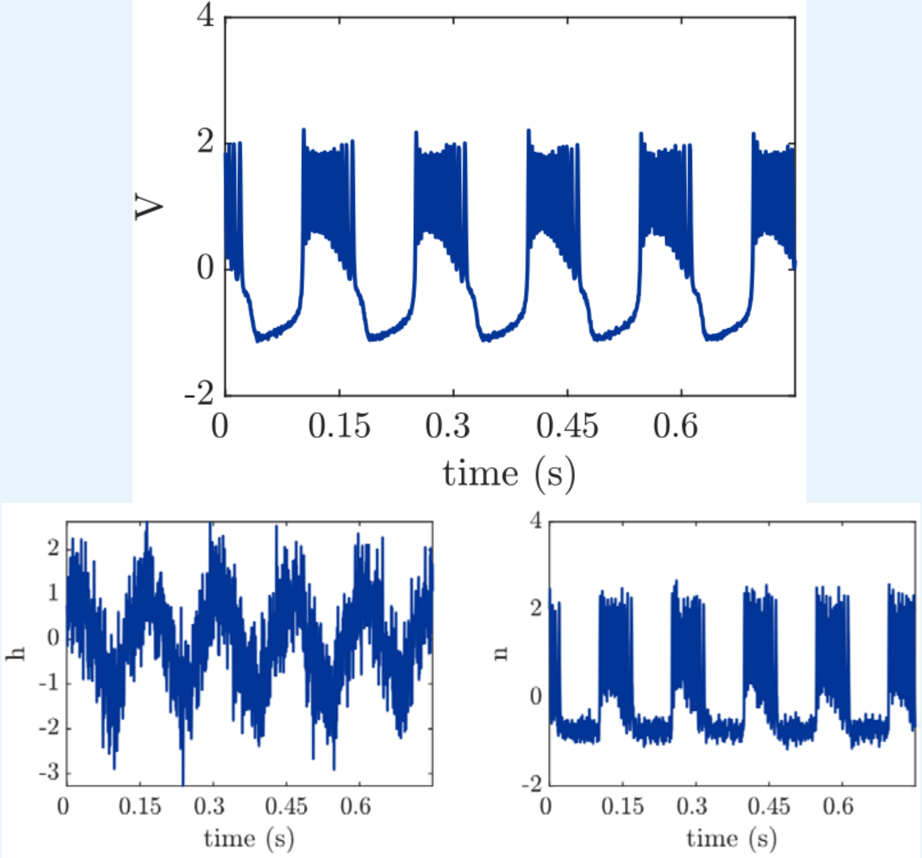
RNN with simple, tractable mathematical structure

→ yet performance en par with LSTM, orth-RNN etc.

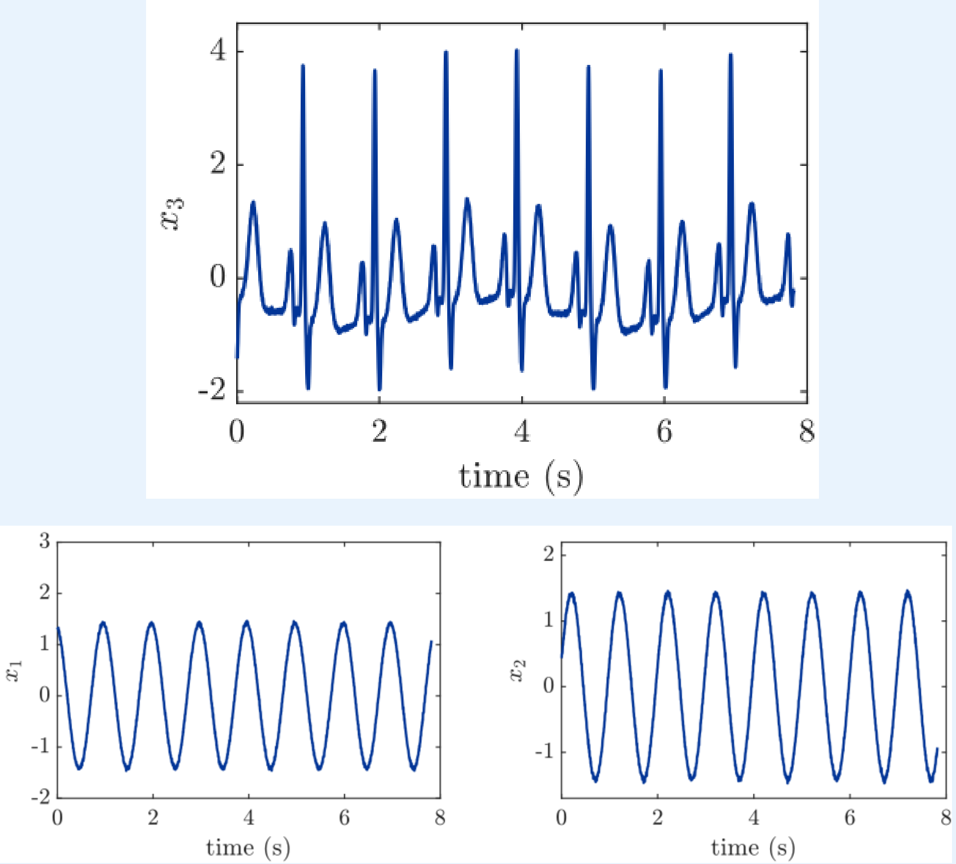
(Schmidt et al. 2021, *ICLR*)

Unsupervised dynamical systems reconstruction

Bursting neuron model



Simulated electrocardiogram (ECG)

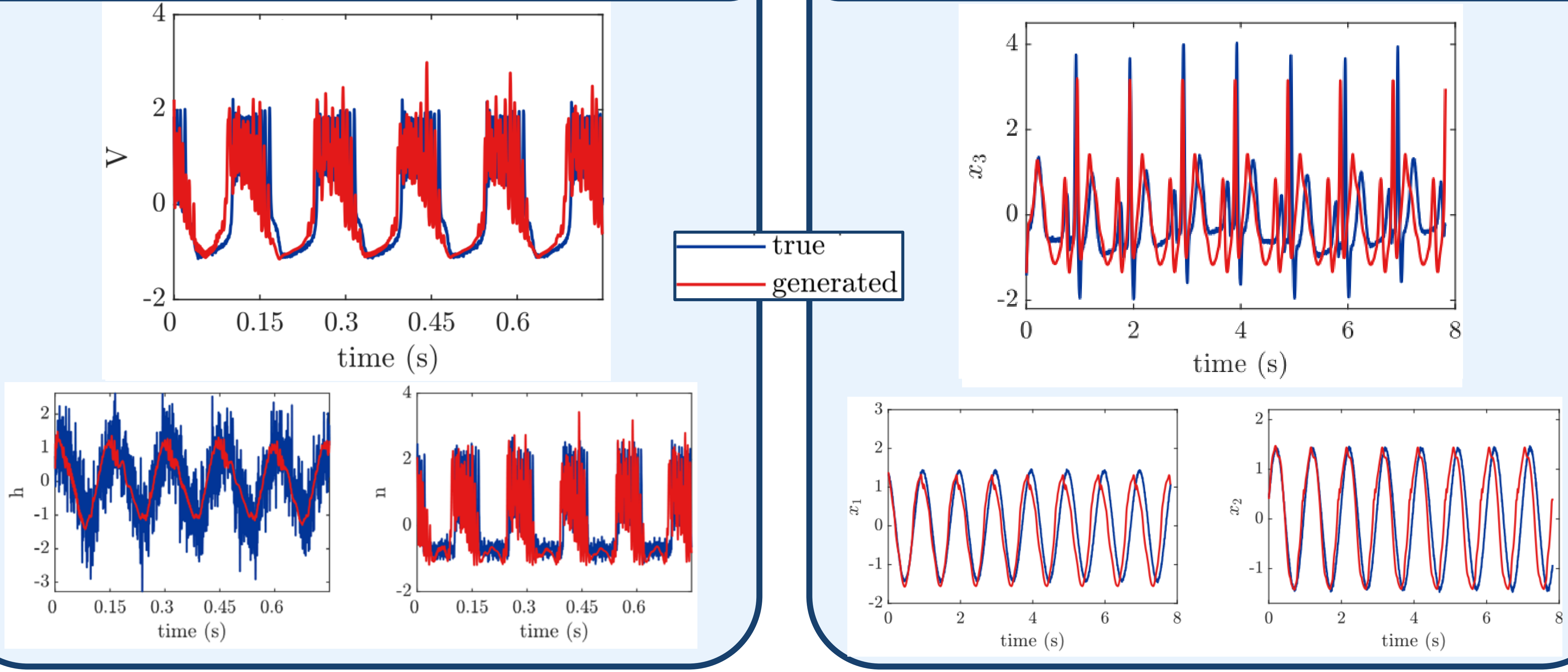


(Schmidt et al. 2021, *ICLR*)

Unsupervised dynamical systems reconstruction

Bursting neuron model

Simulated electrocardiogram (ECG)

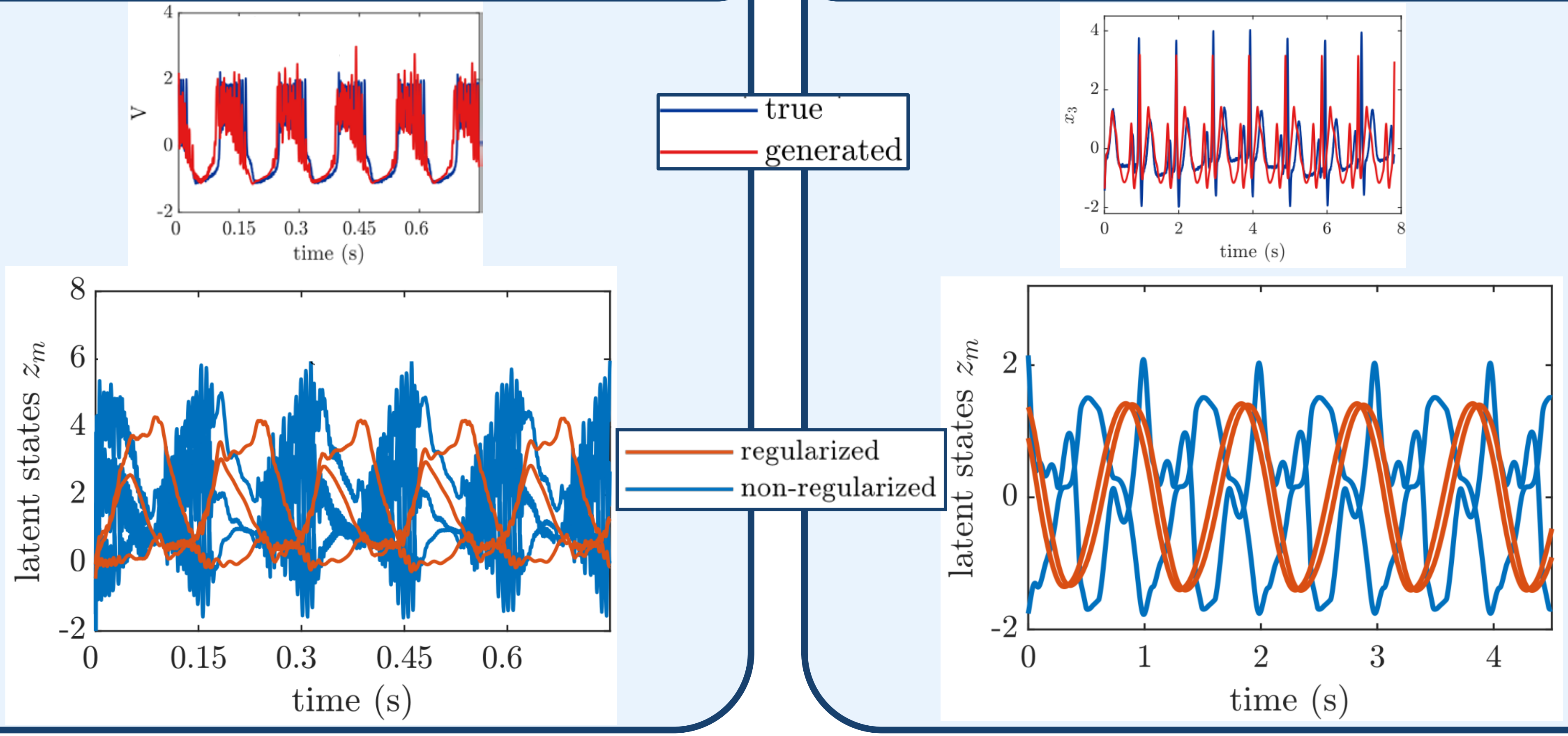


(Schmidt et al. 2021, *ICLR*)

Unsupervised dynamical systems reconstruction

Bursting neuron model

Simulated electrocardiogram (ECG)



(Schmidt et al. 2021, *ICLR*)

Theoretical insights: Link between dynamics & loss gradients

Rewrite

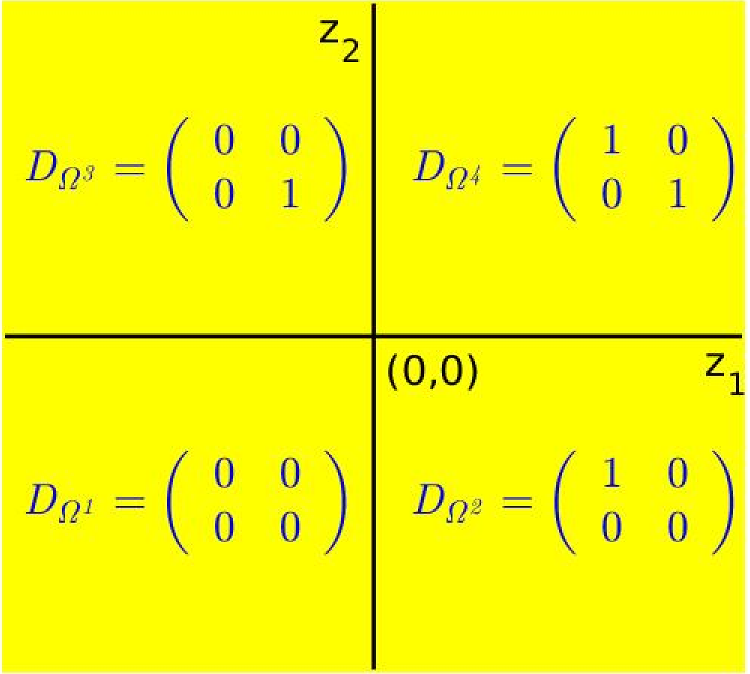
$$\boldsymbol{z}_t = F(\boldsymbol{z}_{t-1}) = \boldsymbol{A}\boldsymbol{z}_{t-1} + \boldsymbol{W}\max(0, \boldsymbol{z}_{t-1}) + \boldsymbol{h}$$

$$=: \big(\boldsymbol{A} + \boldsymbol{W}\boldsymbol{D}_{\Omega(t-1)}\big)\boldsymbol{z}_{t-1} + \boldsymbol{h}$$

$$=: \boxed{\boldsymbol{W}_{\Omega(t-1)}\boldsymbol{z}_{t-1} + \boldsymbol{h}}$$

where $\boldsymbol{D}_{\Omega(t)} := \mathrm{diag}(\boldsymbol{d}_{\Omega(t)})$, $\boldsymbol{d}_{\Omega(t)} := (d_1, d_2, \cdots, d_M)$

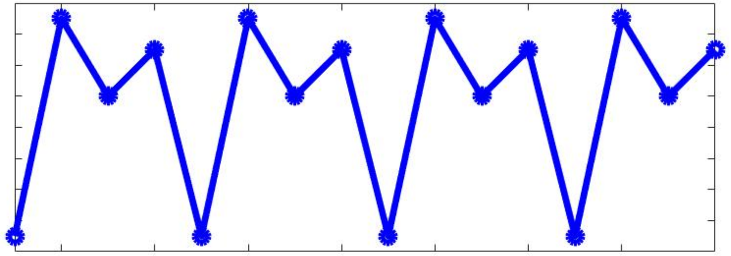
with $d_m(z_{m,t}) := d_m = 1$ iff $z_{m,t} > 0$ and 0 else



Fixed point $\boldsymbol{z}^* = F(\boldsymbol{z}^*)$

***k*-cycle** $\{\boldsymbol{z}^{*1}, \ldots, \boldsymbol{z}^{*k}\} = \{\boldsymbol{z}^{*1}, F(\boldsymbol{z}^{*1}), \ldots, F^k(\boldsymbol{z}^{*1})\}$,

$$\boldsymbol{z}^{*r} = F\Big(F\big(\ldots F(\boldsymbol{z}^{*r})\big)\Big) = F^k(\boldsymbol{z}^{*r}) \text{ for } 1 \leq r \leq k$$



Theoretical insights: Link between dynamics & loss gradients

$$\boldsymbol{z}_2 = F(\boldsymbol{z}_1) = \boldsymbol{W}_{\Omega(1)}\, \boldsymbol{z}_1 + \boldsymbol{h},$$

$$\boldsymbol{z}_3 = F^2(\boldsymbol{z}_1) = F(\boldsymbol{z}_2) = \boldsymbol{W}_{\Omega(2)}\, \boldsymbol{W}_{\Omega(1)}\, \boldsymbol{z}_1 + \big(\boldsymbol{W}_{\Omega(2)} + \boldsymbol{I}\big)\boldsymbol{h},$$

$$\vdots$$

$$\begin{aligned}\boldsymbol{z}_T = F^{T-1}(\boldsymbol{z}_1) = F(\boldsymbol{z}_{T-1}) &= \boldsymbol{W}_{\Omega(T-1)}\, \boldsymbol{W}_{\Omega(T-2)} \cdots \boldsymbol{W}_{\Omega(1)}\, \boldsymbol{z}_1 \\ &+ \big(\boldsymbol{W}_{\Omega(T-1)}\, \boldsymbol{W}_{\Omega(T-2)} \cdots \boldsymbol{W}_{\Omega(2)} \\ &+ \boldsymbol{W}_{\Omega(T-1)}\, \boldsymbol{W}_{\Omega(T-2)} \cdots \boldsymbol{W}_{\Omega(3)} + \cdots + \boldsymbol{W}_{\Omega(T-1)} + \boldsymbol{I}\big)\boldsymbol{h}\end{aligned}$$

$$= \boxed{\prod_{i=1}^{T-1} \boldsymbol{W}_{\Omega(T-i)}}\, \boldsymbol{z}_1 + \Big[\sum_{j=2}^{T-1} \boxed{\prod_{i=1}^{j-1} \boldsymbol{W}_{\Omega(T-i)}} + \boldsymbol{I}\Big]\boldsymbol{h}$$

(Schmidt et al. 2021, *ICLR*)

Theoretical insights: Link between dynamics & loss gradients

dynamics

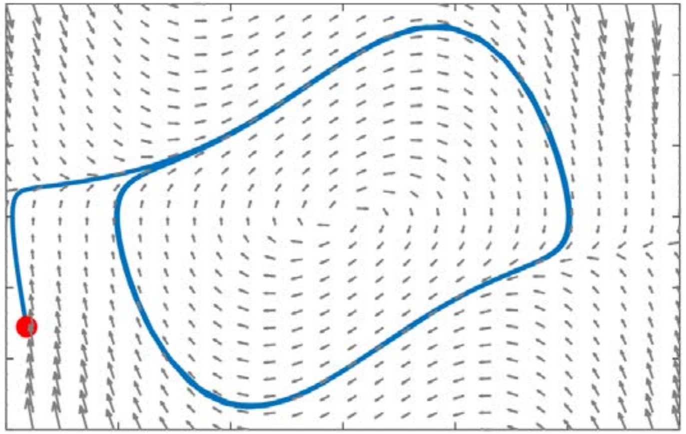
$$\boldsymbol{z}_T = \prod_{i=1}^{T-1} \boldsymbol{W}_{\Omega(T-i)}\, \boldsymbol{z}_1 + \left[\sum_{j=2}^{T-1} \prod_{i=1}^{j-1} \boldsymbol{W}_{\Omega(T-i)} + \boldsymbol{I}\right] \boldsymbol{h}$$

“gradients” (loss tensors)

$$\frac{\partial \mathcal{L}}{\partial w_{mk}} = \sum_{t=2}^{T} \frac{\partial \mathcal{L}_t}{\partial \boldsymbol{z}_t} \frac{\partial \boldsymbol{z}_t}{\partial w_{mk}}, \qquad \text{with } \frac{\partial \boldsymbol{z}_t}{\partial w_{mk}} = \mathbf{1}_{(m,k)} \boldsymbol{D}_{\Omega(t-1)}\, \boldsymbol{z}_{t-1}$$

$$+ \sum_{j=2}^{t-2} \left(\prod_{i=1}^{j-1} \boldsymbol{W}_{\Omega(t-i)} \right) \mathbf{1}_{(m,k)} \boldsymbol{D}_{\Omega(t-j)} \boldsymbol{z}_{t-j} + \prod_{i=1}^{t-2} \boldsymbol{W}_{\Omega(t-i)} \frac{\partial \boldsymbol{z}_2}{\partial w_{mk}}$$

***Theorem*.** PLRNN dynamics converges to fixed point or *k*-cycle
→ Norms of loss “gradients” (tensors) bounded from above



$$\left\| \frac{\partial \boldsymbol{z}_t}{\partial \boldsymbol{A}} \right\|_2, \left\| \frac{\partial \boldsymbol{z}_t}{\partial \boldsymbol{h}} \right\|_2, \left\| \frac{\partial \boldsymbol{z}_t}{\partial \boldsymbol{W}} \right\|_2 < \lambda$$

(Schmidt et al. 2021, *ICLR*)

Theoretical insights: Link between dynamics & loss gradients

dynamics

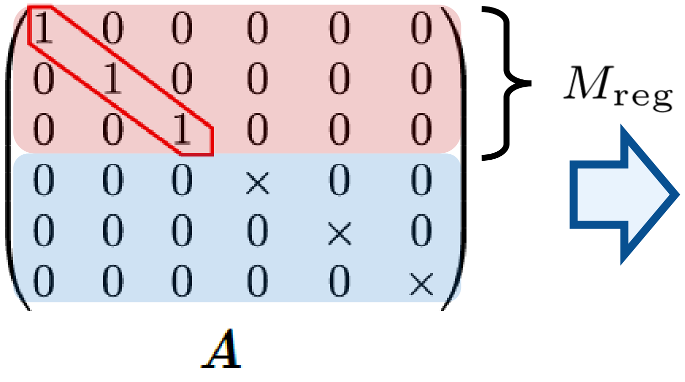
$$\boldsymbol{z}_T = \prod_{i=1}^{T-1} \boldsymbol{W}_{\Omega(T-i)}\, \boldsymbol{z}_1 + \left[\sum_{j=2}^{T-1} \prod_{i=1}^{j-1} \boldsymbol{W}_{\Omega(T-i)} + \boldsymbol{I}\right] \boldsymbol{h}$$

“gradients” (loss tensors)

$$\frac{\partial \mathcal{L}}{\partial w_{mk}} = \sum_{t=2}^{T} \frac{\partial \mathcal{L}_t}{\partial \boldsymbol{z}_t} \frac{\partial \boldsymbol{z}_t}{\partial w_{mk}}, \qquad \text{with } \frac{\partial \boldsymbol{z}_t}{\partial w_{mk}} = \boldsymbol{1}_{(m,k)} \boldsymbol{D}_{\Omega(t-1)}\, \boldsymbol{z}_{t-1}$$

$$+ \sum_{j=2}^{t-2} \left(\prod_{i=1}^{j-1} \boldsymbol{W}_{\Omega(t-i)} \right) \boldsymbol{1}_{(m,k)} \boldsymbol{D}_{\Omega(t-j)} \boldsymbol{z}_{t-j} + \prod_{i=1}^{t-2} \boldsymbol{W}_{\Omega(t-i)} \frac{\partial \boldsymbol{z}_2}{\partial w_{mk}}$$

Theorem**.** PLRNN + MAR. Non-regularized subsystem converges to FP or *k*-cycle → Norms of loss “gradients” bounded from above *and* below



$$\underbrace{\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & \times & 0 & 0 \\ 0 & 0 & 0 & 0 & \times & 0 \\ 0 & 0 & 0 & 0 & 0 & \times \end{pmatrix}}_{\boldsymbol{A}} \quad \left.\right\} M_{\text{reg}}$$

$$\rho_{up} \geq \left\| \frac{\partial \boldsymbol{z}_T}{\partial \boldsymbol{z}_t} \right\|_2 = \left\| \prod_{t<k\leq T} \boldsymbol{W}_{\Omega(k)} \right\|_2 \geq \rho_{low} > 0$$

(Schmidt et al. 2021, *ICLR*)

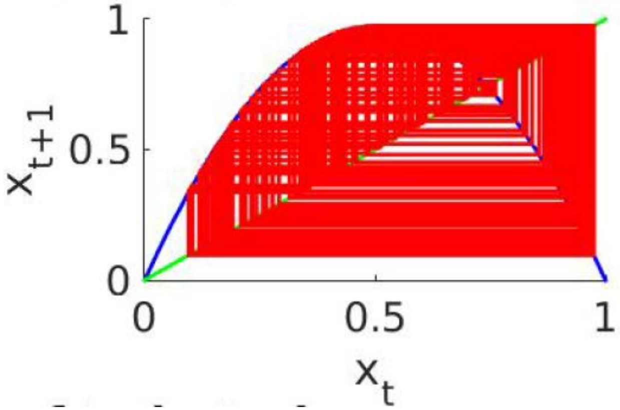
What about chaotic systems?

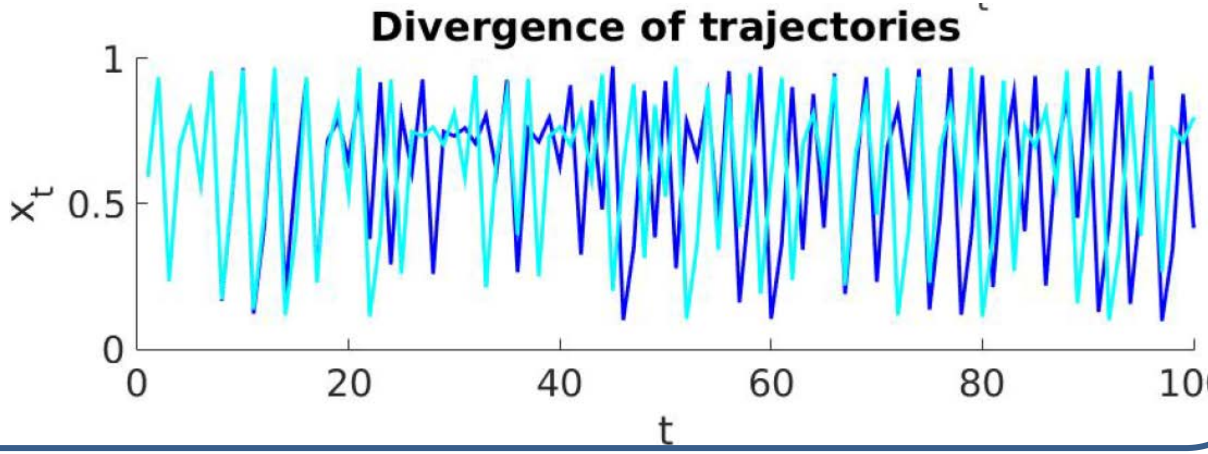
***Definition*.** We call a (deterministic) DS chaotic if
- long-term behavior is aperiodic (irregular)
- Max. Lyapunov exponent > 0 (nearby trajectories diverge expon. fast)

Logistic map

$$x_{t+1} = \alpha x_t(1 - x_t)$$

$$x_0 \in [0,1]\,,\ \alpha \in [0,4]$$



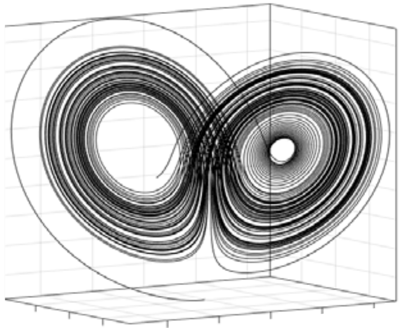


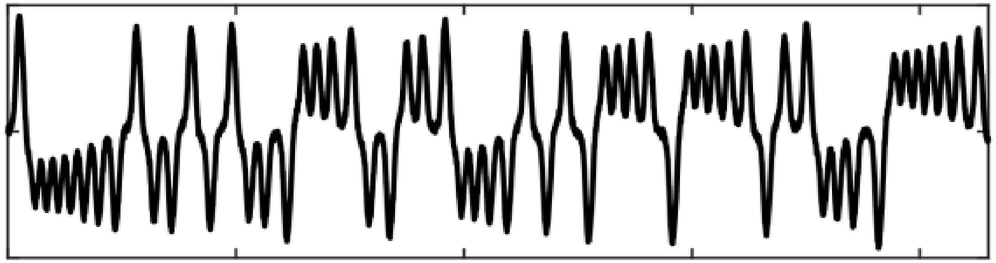
Lorenz system

$$\dot{x}_1 = a(x_2 - x_1)$$

$$\dot{x}_2 = x_1(b - x_3) - x_2$$

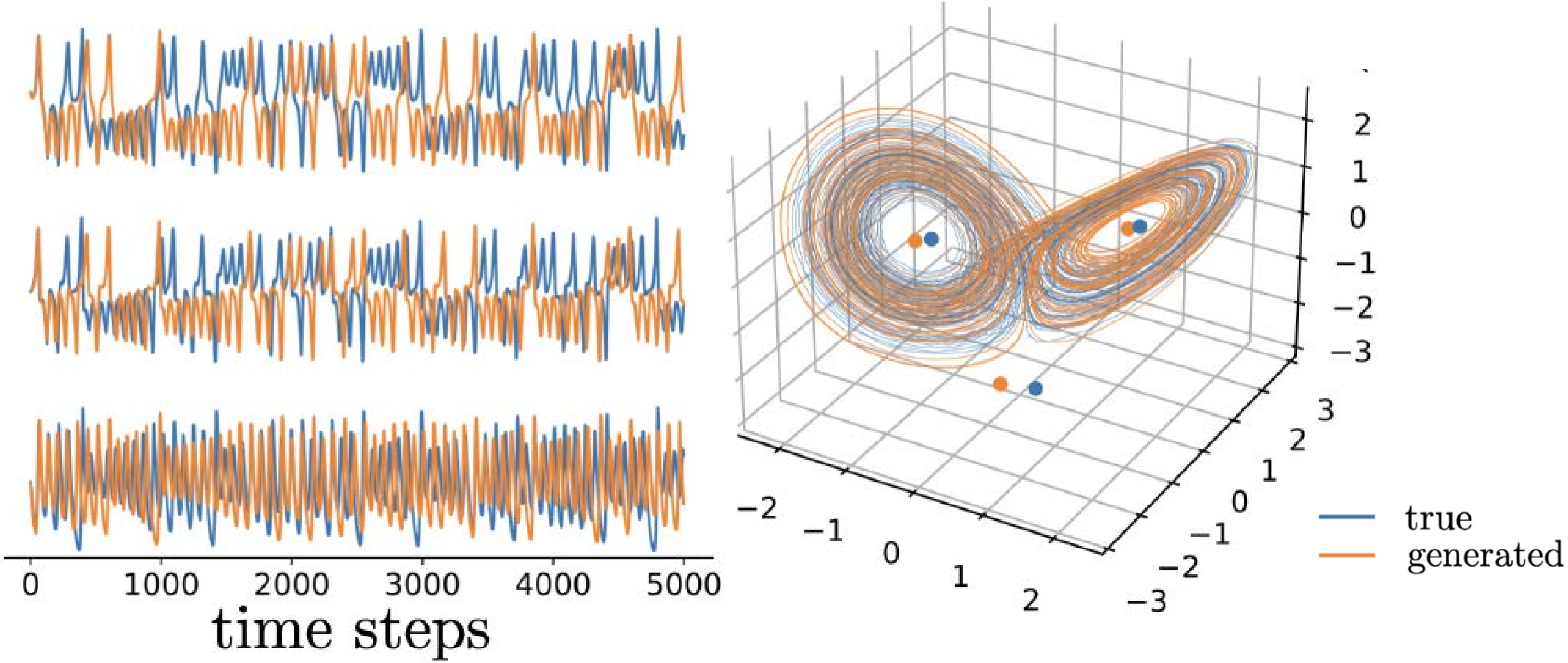
$$\dot{x}_3 = x_1 x_2 - c x_3$$





Example: Lorenz-63 system

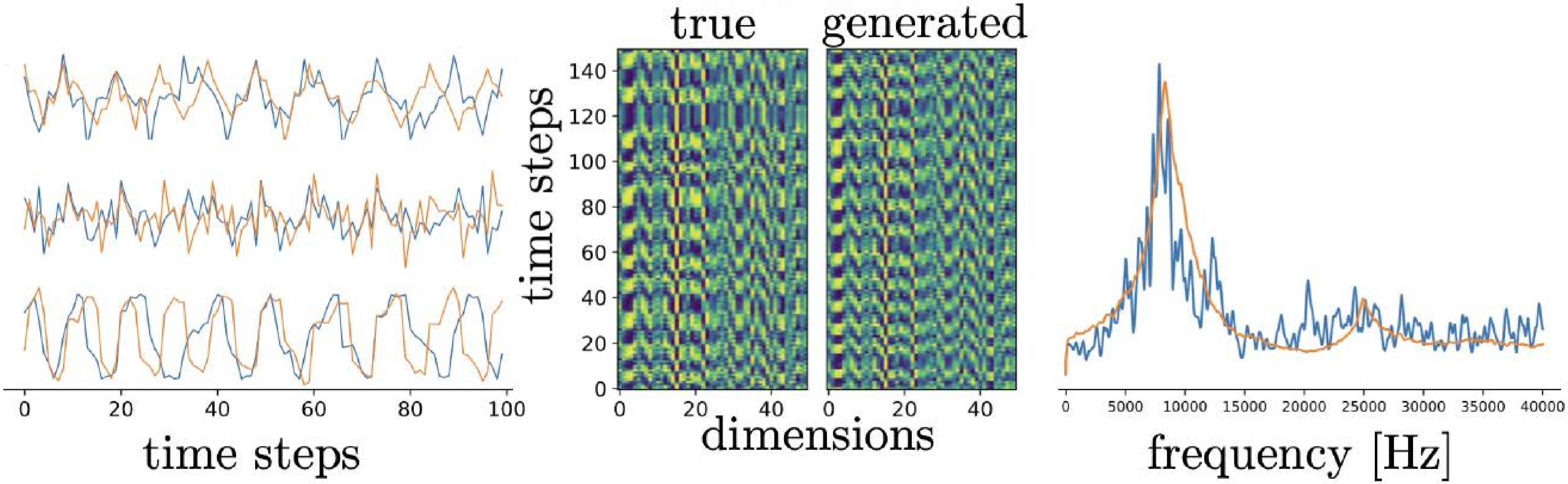
$$\begin{aligned}\dot{x}_1 &= a(x_2 - x_1)\\ \dot{x}_2 &= x_1(b - x_3) - x_2\\ \dot{x}_3 &= x_1 x_2 - c x_3\end{aligned}$$



(Bereska, Brenner et al., *in prep*)

Example: Lorenz-96 system (spatio-temporal chaos)

$$\frac{dx_i}{dt} = (x_{i+1} - x_{i-2})x_{i-1} - x_i + F,\ i = 1 \ldots N$$



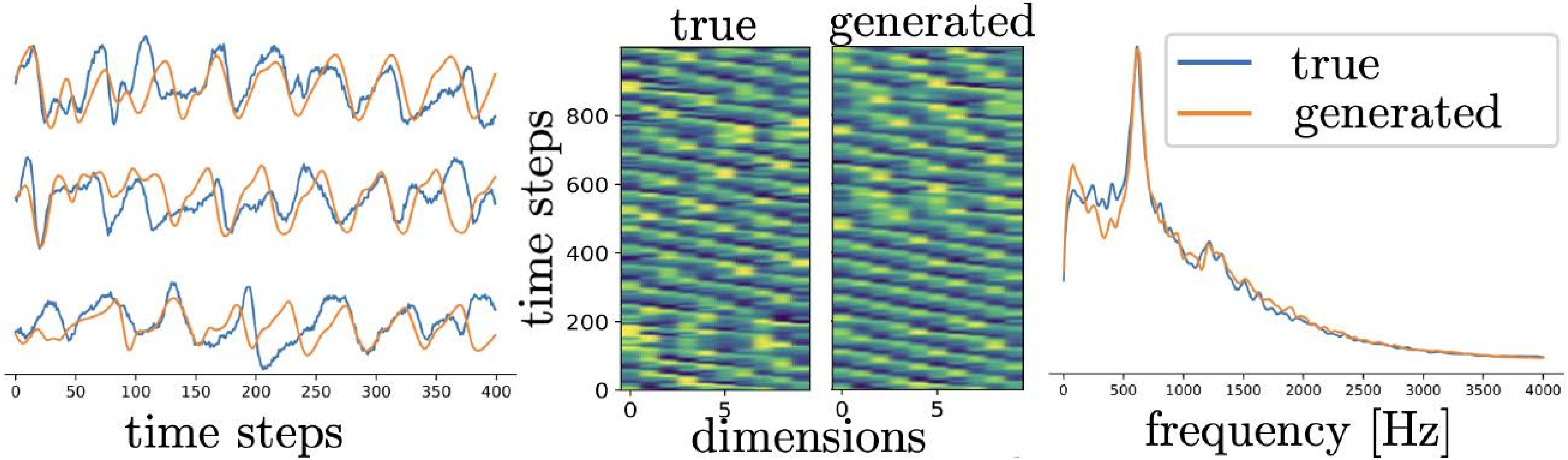
(Bereska, Brenner et al., *in prep*)

Example: Neural population model (“structured chaos”)

$$\frac{d\mathbf{h}}{dt} = -\mathbf{h} + \mathrm{J}\phi(\mathbf{h}) + \frac{J_1}{\sqrt{N}}\xi \upsilon^T \phi$$

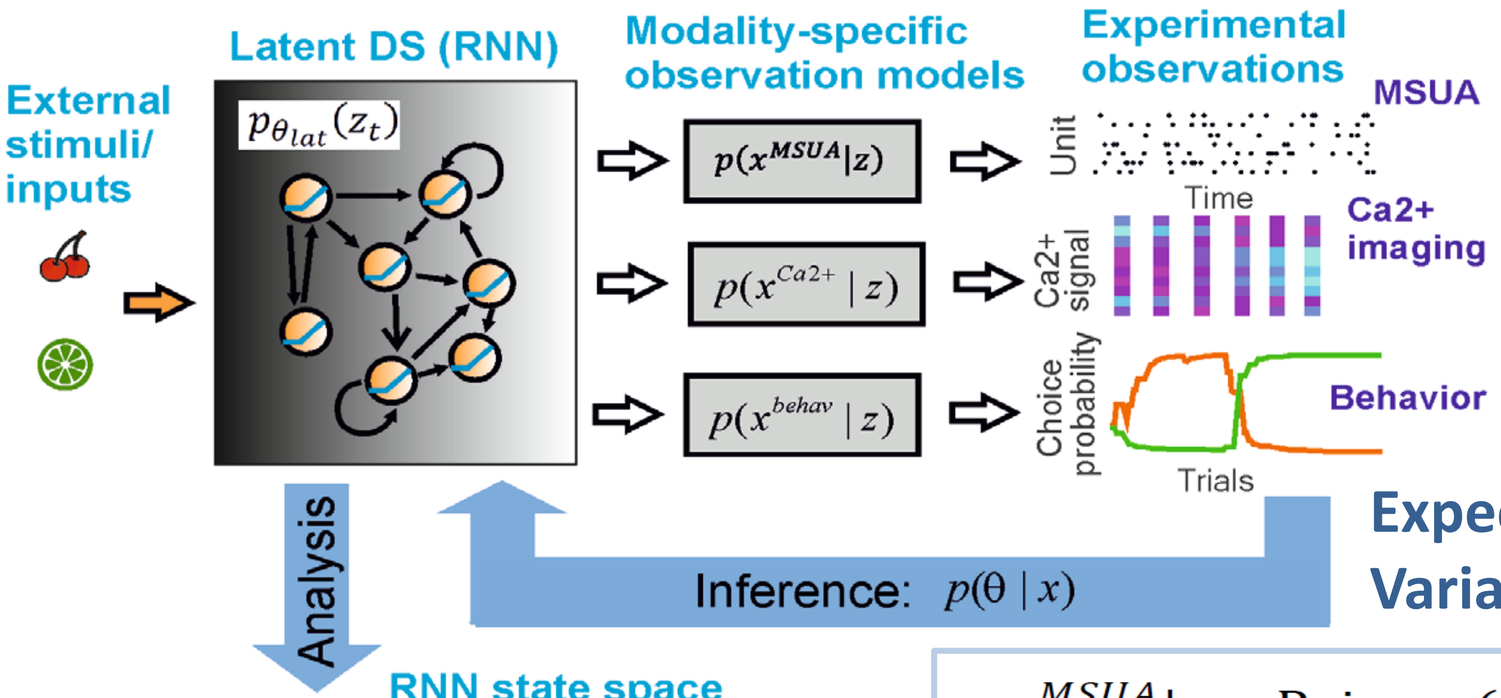
$$\phi(\mathbf{h}) = \tanh(\mathbf{h}(t))$$

(Landau & Sompolinsky 2018, *PLoS Comp Biol*)



(Bereska, Brenner et al., *in prep*)

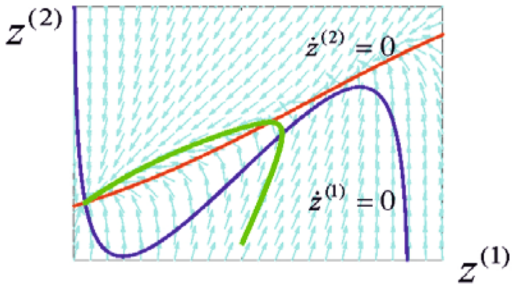
Probabilistic framework: Multi-modal PLRNN



Expectation Maximization

Variational Inference (SVAE)

RNN state space



$$\boldsymbol{z}_t|\boldsymbol{z}_{t-1}\sim N(\boldsymbol{\mu}_t^{(z)},\boldsymbol{\Sigma}_t^{(z)})$$

$$\boldsymbol{\mu}_t^{(z)}=\boldsymbol{A}\boldsymbol{z}_{t-1}+\boldsymbol{W}\max[0,\boldsymbol{z}_{t-1}]+\boldsymbol{h}+\boldsymbol{C}\boldsymbol{s}_t$$

$$\boldsymbol{x}_t^{MSUA}|\boldsymbol{z}_t\sim\text{Poisson}(\boldsymbol{\lambda}_t=g_{\boldsymbol{\theta}}(\boldsymbol{z}_t))$$

$$\boldsymbol{x}_t^{Ca2+}|\boldsymbol{z}_t\ldots\boldsymbol{z}_{t-\tau}\sim N(\boldsymbol{\mu}_t^{(x)},\boldsymbol{\Sigma}_t^{(x)})$$

$$\boldsymbol{\mu}_t^{(x)}=f_{\boldsymbol{\theta}}^{(\mu)}(\boldsymbol{z}_t\ldots\boldsymbol{z}_{t-\tau})$$

$$\boldsymbol{\Sigma}_t^{(x)}=f_{\boldsymbol{\theta}}^{(\Sigma)}(\boldsymbol{z}_t\ldots\boldsymbol{z}_{t-\tau})$$

$$\boldsymbol{x}_t^{behavior}|\boldsymbol{z}_t\sim\text{Catgorical}(\boldsymbol{\pi}_t=f_{\boldsymbol{\theta}}^{(\pi)}(\boldsymbol{z}_t))$$

Application: Inferring PLRNN from human fMRI

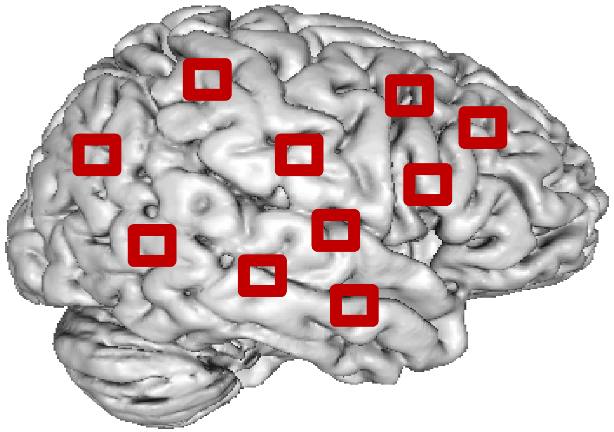
process model

$$\mathbf{z}_t = \mathbf{A}\mathbf{z}_{t-1} + \mathbf{W}\phi(\mathbf{z}_{t-1}) + \mathbf{h} + \mathbf{C}\mathbf{u}_t + \boldsymbol{\varepsilon}_t$$

$$\boldsymbol{\varepsilon}_t \sim N(\mathbf{0}, \boldsymbol{\Sigma})$$

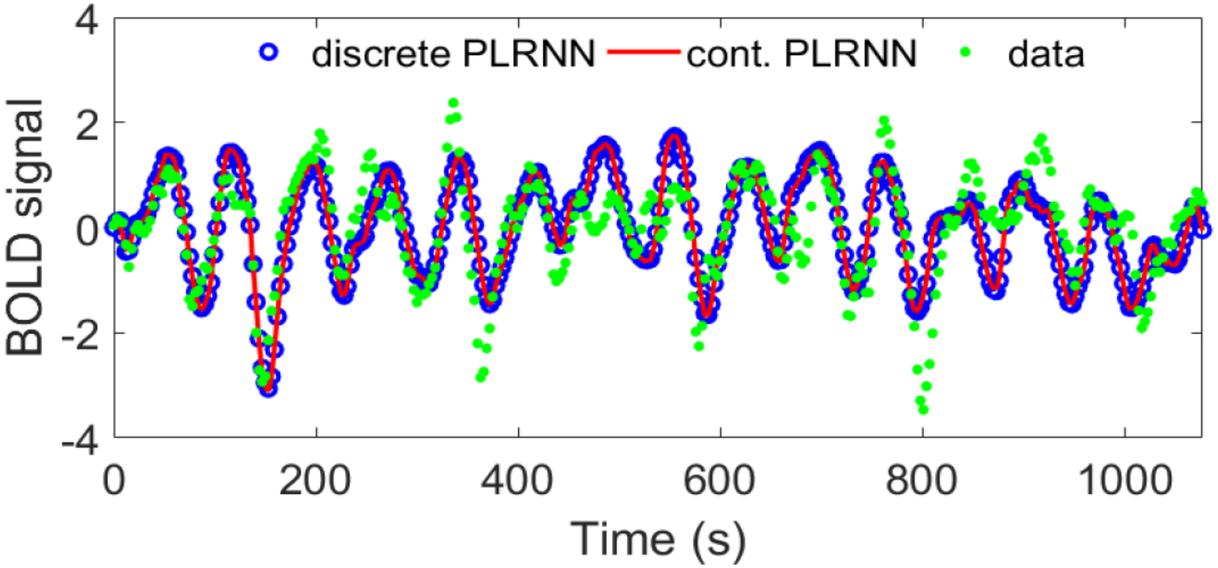
fMRI observation model

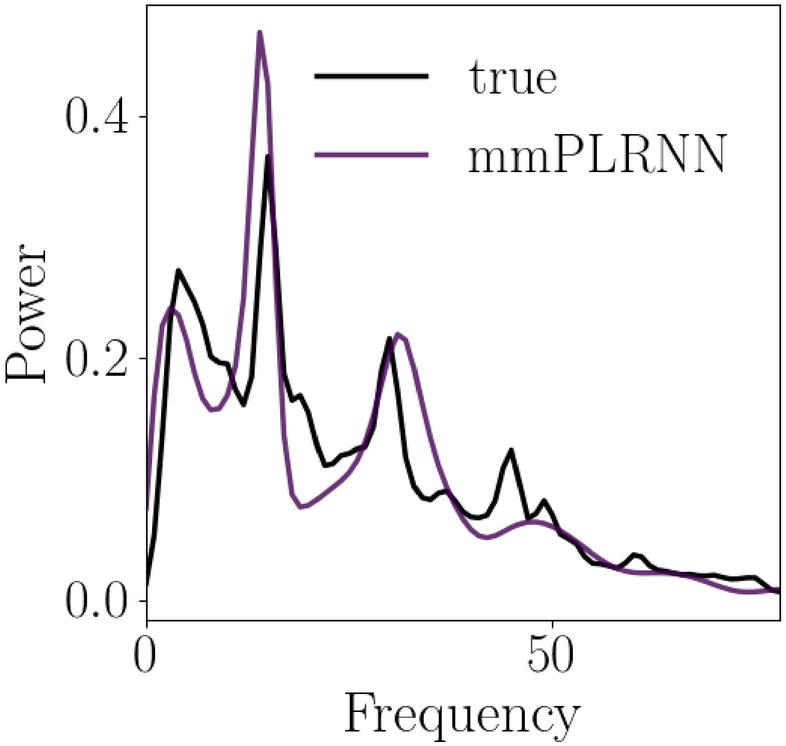
$$\mathbf{x}_t \,|\, \mathbf{z}_t \sim \mathrm{N}(\mathbf{B}[hrf * \mathbf{z}_{t-\tau:t}] + \mathbf{M}\mathbf{r}_t, \boldsymbol{\Gamma})$$



(Koppe et al. 2019, *PLoS Comp Biol*)

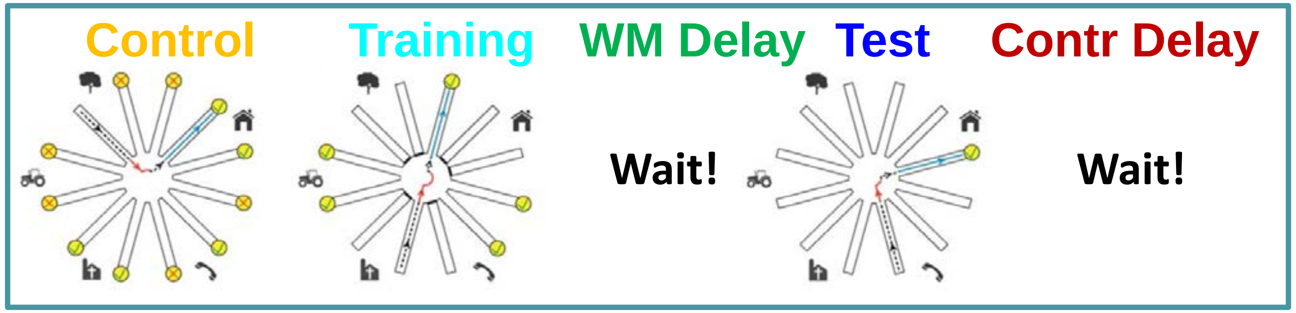
Application: Inferring PLRNN from human fMRI



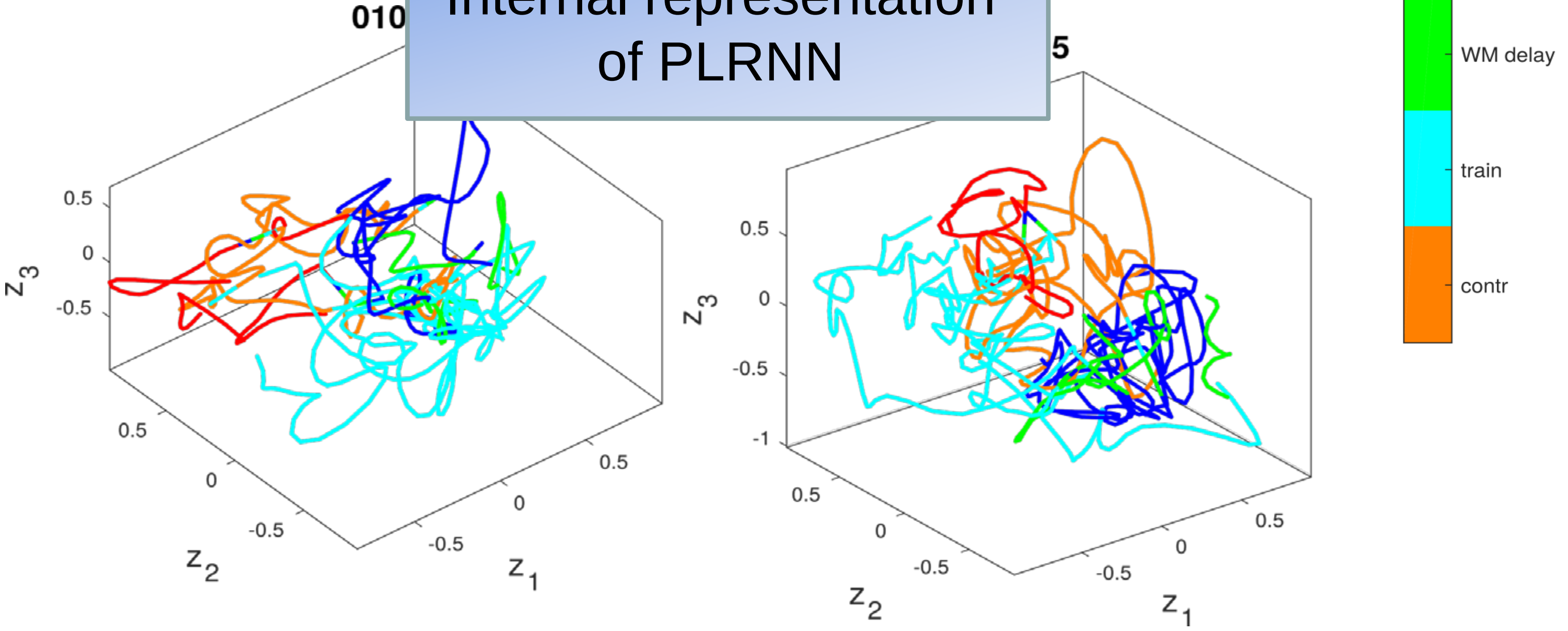


(Koppe et al. 2019, *PLoS Comp Biol*; Monfared & Durstewitz 2021, *ICML*; Bommer, Kramer et al., *in prep.*)

Reconstructed state spaces from fMRI

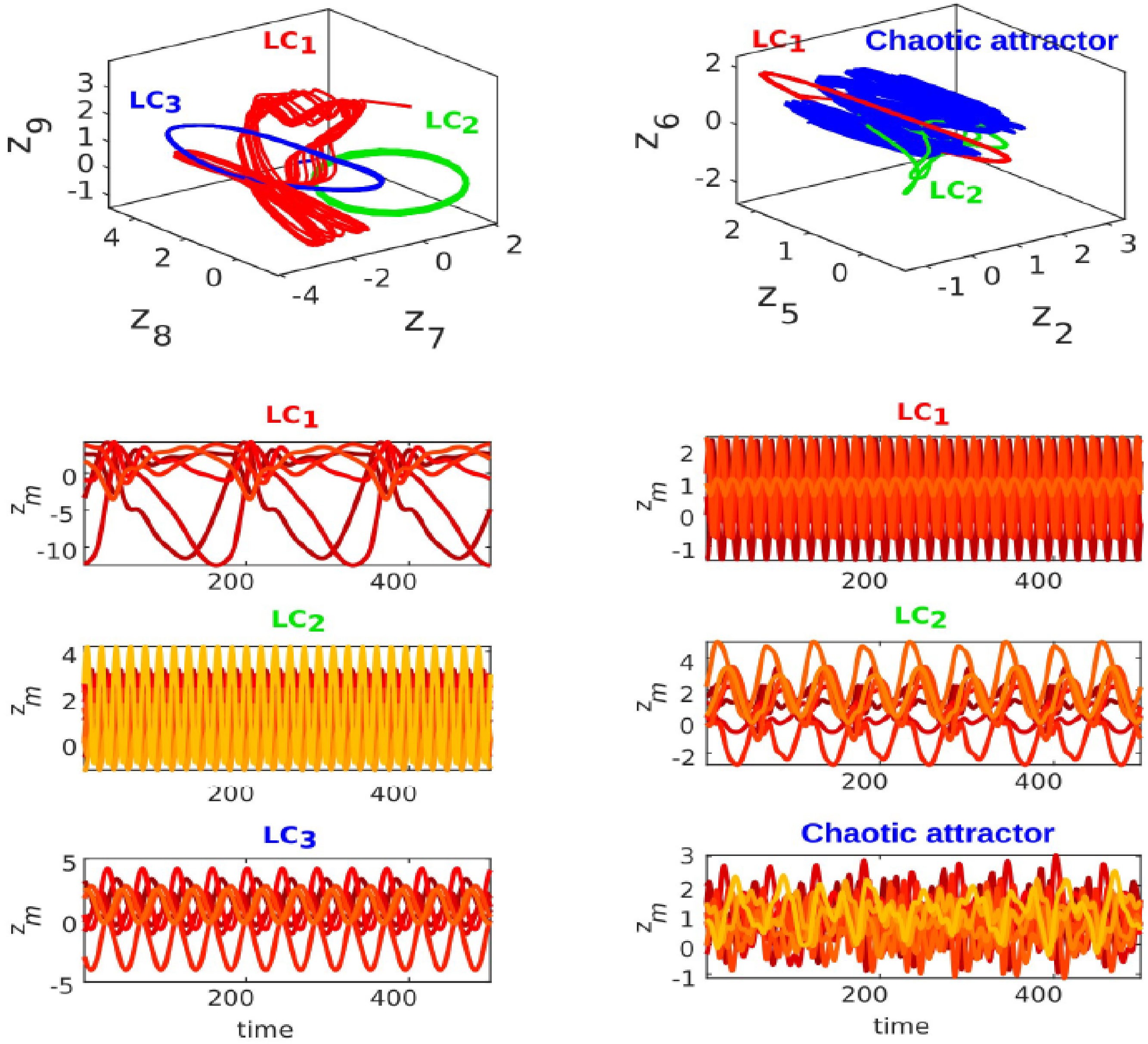


Internal representation of PLRNN



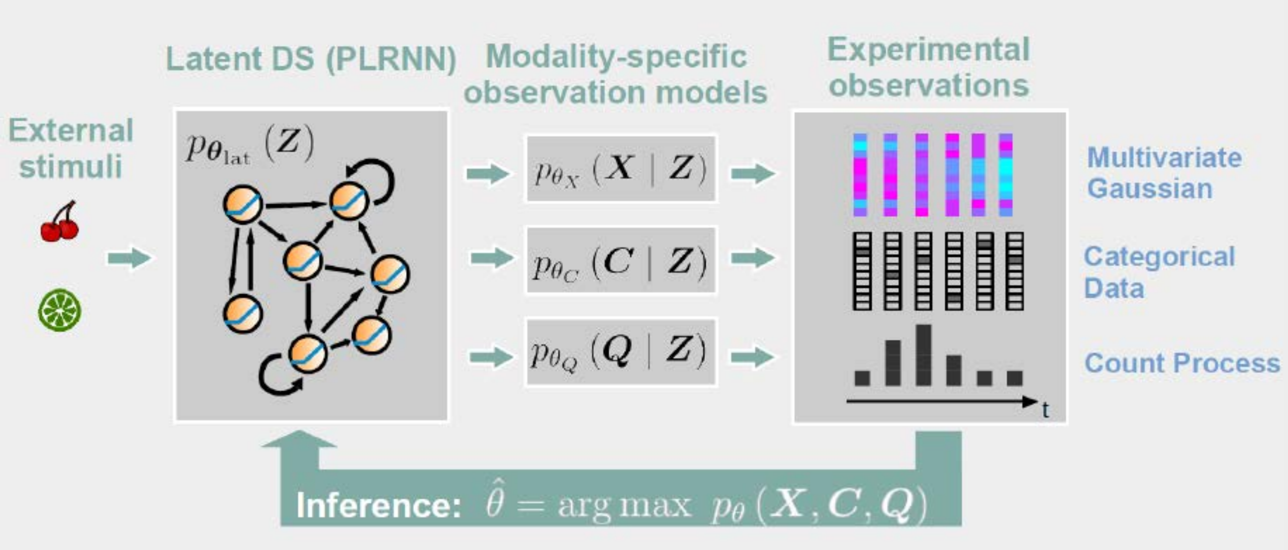
(data: Bähner et al. 2015; Koppe et al. 2019, *PLoS Comp Biol*)

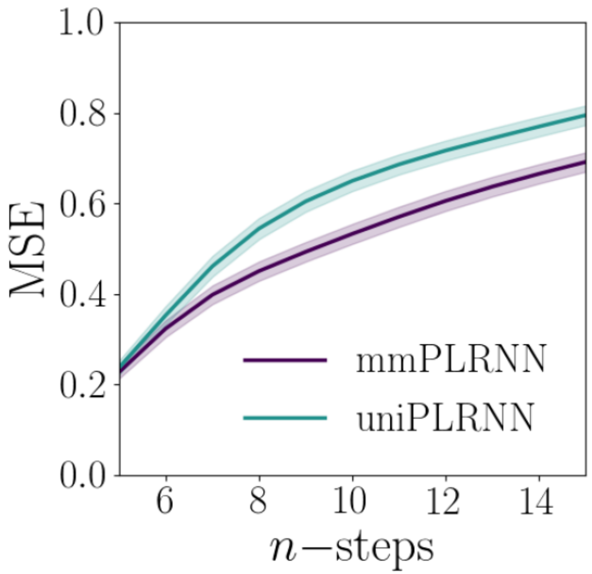
Nonlinear phenomena from fMRI data

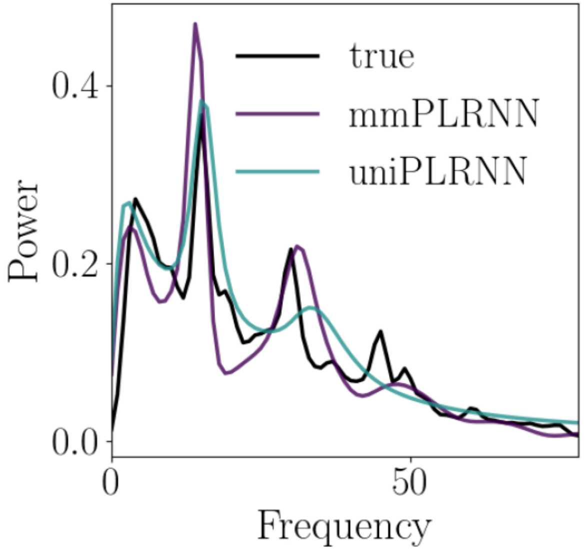


(Koppe et al. 2019, *PLoS Comp Biol*)

Multi-modal integration improves DS reconstruction



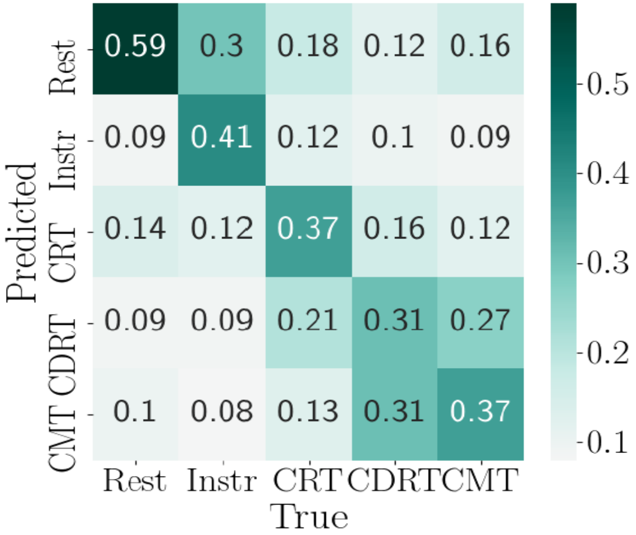




Cross-modal prediction ("decoding thoughts")

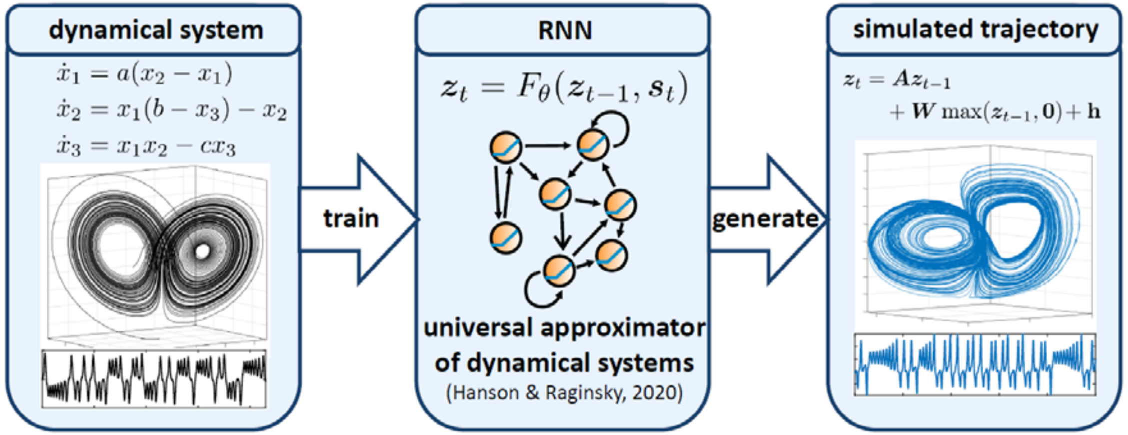
$$\hat{\mathbf{Z}} = \mathrm{E}(\mathbf{Z}|\mathbf{X})$$

$$\rightarrow \hat{\mathbf{C}} = \mathrm{Mode}(\mathbf{C}|\hat{\mathbf{Z}})$$



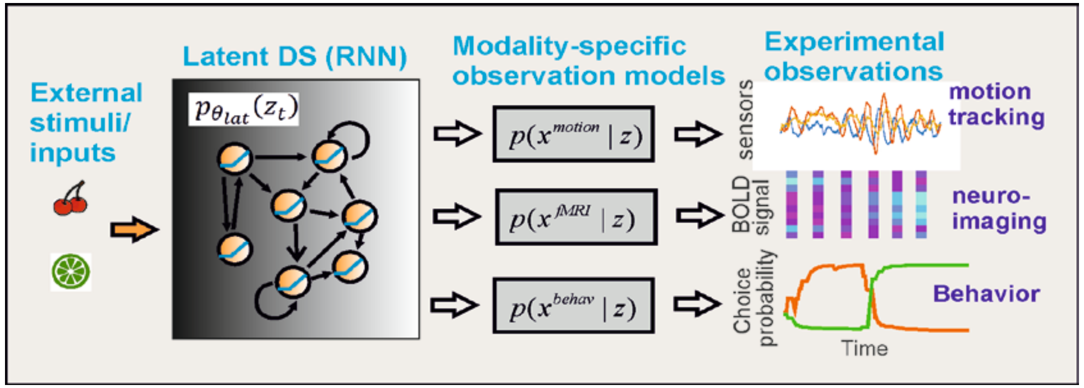
(Bommer, Kramer et al., *submitted*)

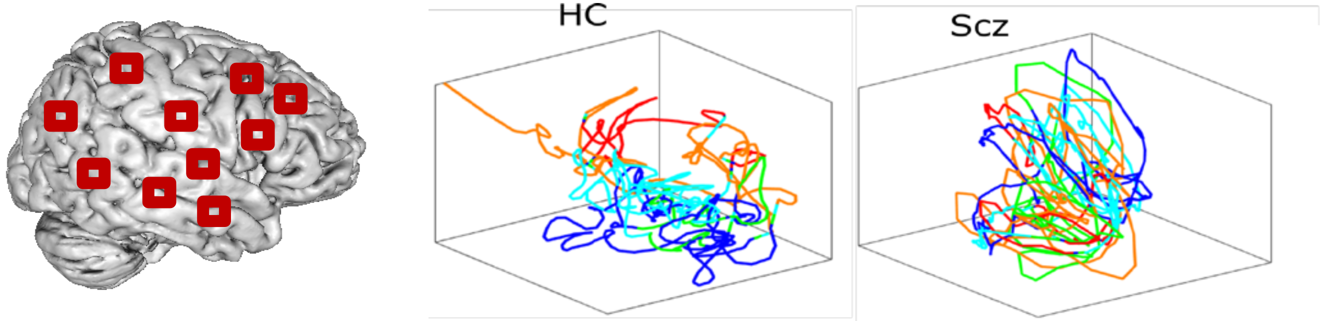
Wrap-up



Tractable PLRNN for reconstructing DS from time series

Probabilistic and multimodal framework for PLRNNs





Useful technique to get insight into (malfunctioning) brain dynamics

github.com/DurstewitzLab

Many thanks for your attention!

Job openings!

Collaborations:

Heike Tost
Florian Bähner
Andreas Meyer-Lindenberg
Chris Lapish
Jeremy Seamans
James Hyman

Bundesministerium für Bildung und Forschung
bccn heidelberg-mannheim
e:Med SYSTEMS MEDICINE
STRUCTURES CLUSTER OF EXCELLENCE
Deutsche Forschungsgemeinschaft DFG