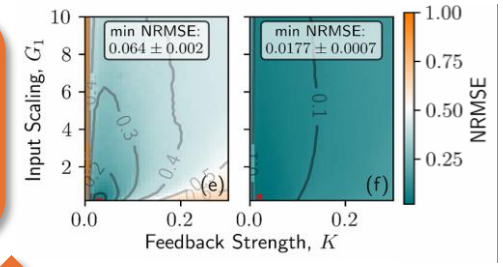
Benchmarking and Optimizing Quantum Reservoirs

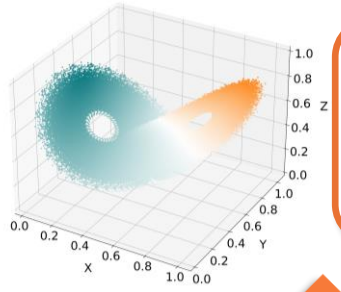
Dr. Lina Jaurigue

CZS Junior Research Group: Interpretable surrogates for efficient analog timeseries forecasting

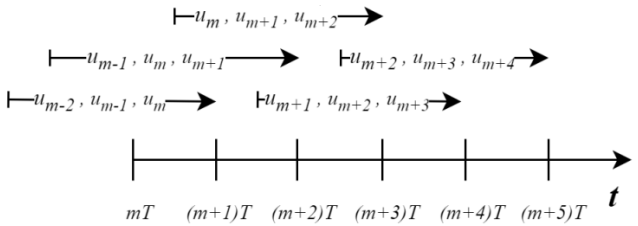
Research interests

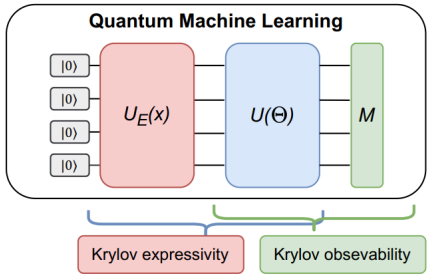
Reservoir computing optimisation
- Hyperparameter optimization
- Quantum reservoir algorithms



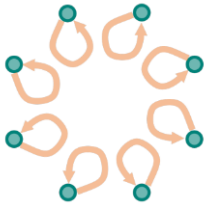


Metrics
- Error measures
- Expressivity and observability

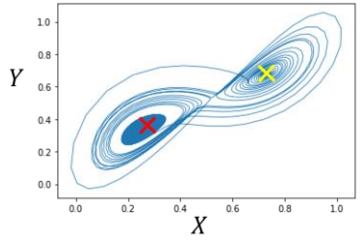


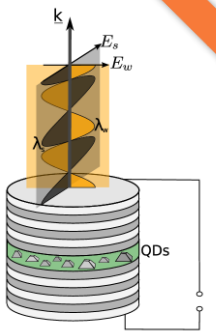


Algorithms for energy efficient edge devices for timeseries forecasting applications

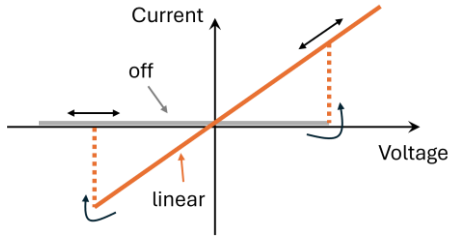


Echo-state networks
- Network size and topology
- Dynamics of trained systems





Physical nonlinearities
- Spin-VCSEL
- Memristors



Outline

- Reservoir computing and related methods
  - Training a reservoir computer
  - Comparison with nonlinear vector regression
  - Quantum reservoir computing – Ising model
- Benchmarking tasks – what are they testing?
  - Lorenz 63 timeseries prediction
  - Information processing capacity (IPC)
- Performance of a quantum reservoir
  - Lorenz timeseries tasks and informations processing capacity
  - Performance tuning via memory restriction
- Memory augmentation methods
  - Input delay and delayed statematrix concatenation

Outline

- **Reservoir computing and related methods**
  - **Training a reservoir computer**
  - **Comparison with nonlinear vector regression**
  - **Quantum reservoir computing – Ising model**

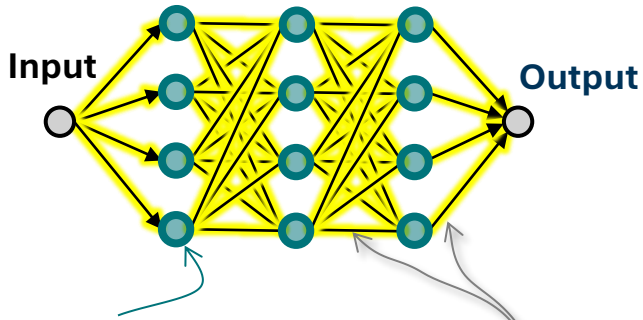
- Benchmarking tasks – what are they testing?
  - Lorenz 63 timeseries prediction
  - Information processing capacity (IPC)

- Performance of a quantum reservoir
  - Lorenz timeseries tasks and informations processing capacity
  - Performance tuning via memory restriction

- Memory augmentation methods
  - Input delay and delayed statematrix concatenation

Artificial Neural Networks

Deep Neural Networks

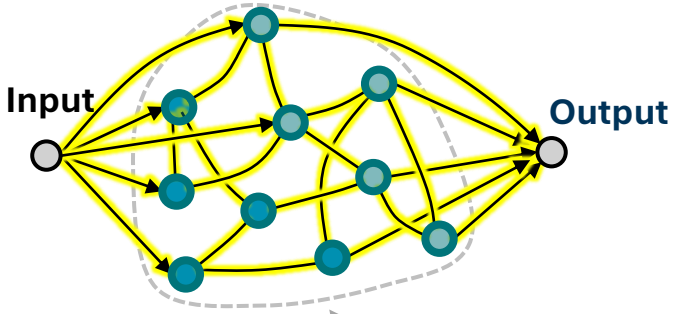


Linear/nonlinear transform of incoming signal

Linear connections that are trained

Network is optimized via gradient decent algorithms

Recurrent Neural Network

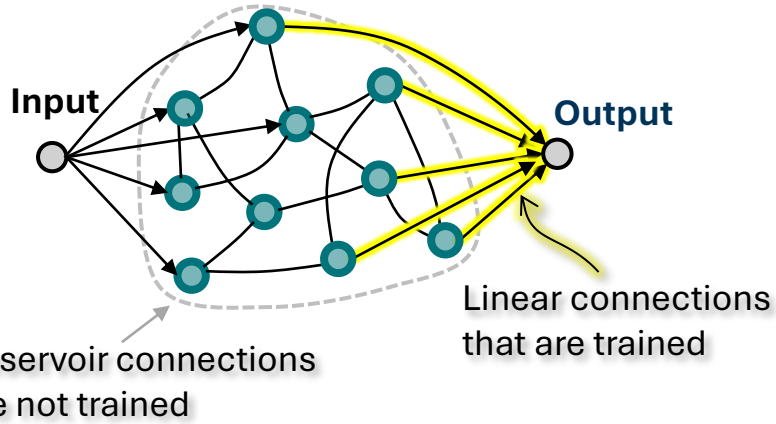


All connections are trained

Reservoir Computing

- Only a final linear layer is trained
- **Time multiplexing** can increase the readout dimension and thus the performance

First introduced by: Appeltant et al., Nat. Commun. **2**, 468 (2011)



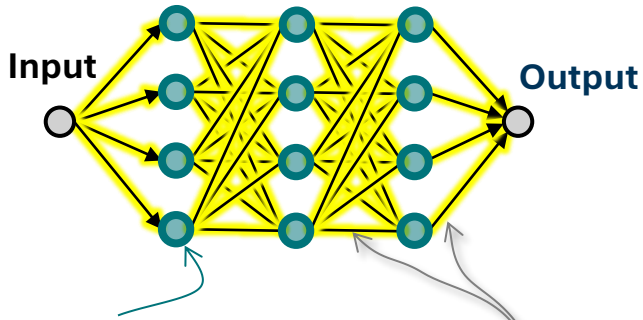
Reservoir connections are not trained

Linear connections that are trained

📖**L. C. Jaurigue** and K. Lüdge, *Nat. Commun.* **13**, 227 (2022).

Artificial Neural Networks

Deep Neural Networks

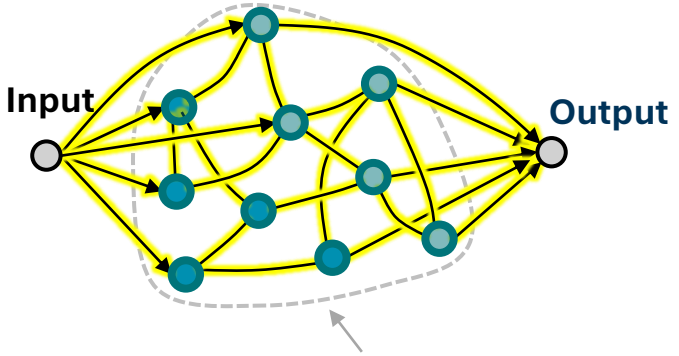


Linear/nonlinear transform of incoming signal

Linear connections that are trained

Network is optimized via gradient decent algorithms

Recurrent Neural Network

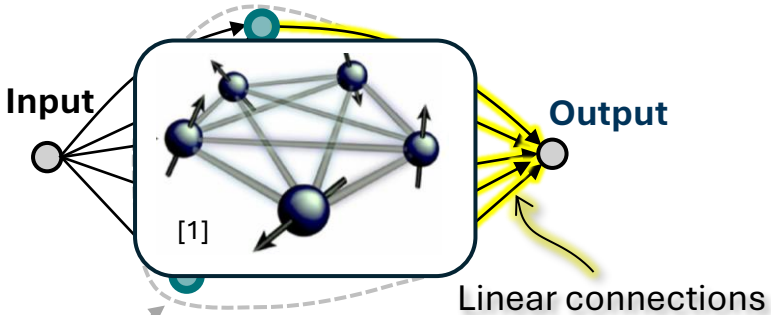


All connections are trained

Reservoir Computing

- Only a final linear layer is trained
- **Time multiplexing** can increase the readout dimension and thus the performance

First introduced by: Appeltant et al., Nat. Commun. **2**, 468 (2011)



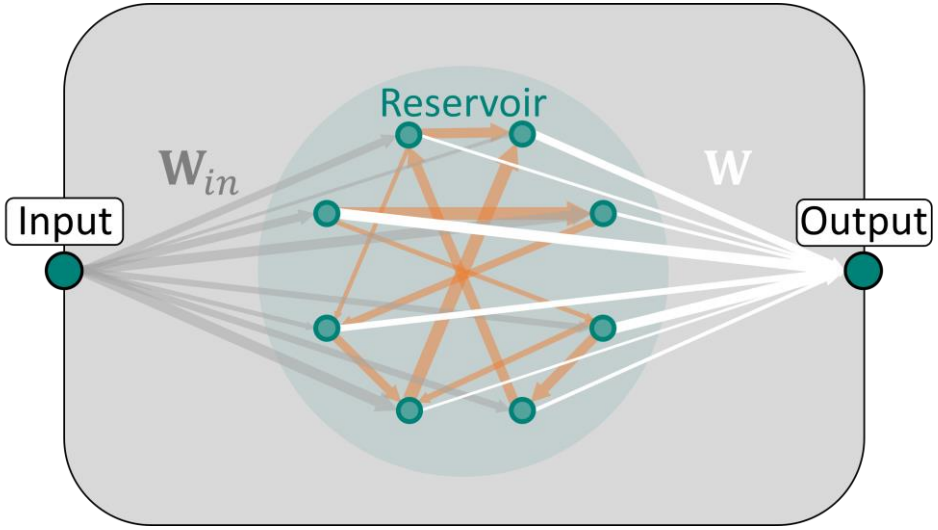
Reservoir connections are not trained

Linear connections that are trained

📖**L. C. Jaurigue** and K. Lüdge, *Nat. Commun.* **13**, 227 (2022).

[1] Image source: K. Fujii et al., Phys. Rev. Applied 8, (2017)

Reservoir training



$$[x_1 \quad \cdots \quad x_N] \begin{bmatrix} W_1 \\ \vdots \\ W_N \end{bmatrix} \rightarrow \textbf{Output}$$

Sampled nodes states

Trained output Weights W

Training:

→ Feed input sequence $I(k)$ of length K_{tr} into the reservoir

→ Collect sampled reservoir responses into a state matrix

$$\mathbf{S} = \begin{bmatrix} x_1(1) & \cdots & x_N(1) & 1 \\ \vdots & \ddots & \vdots & \vdots \\ x_1(k) & \cdots & x_N(k) & 1 \\ \vdots & \ddots & \vdots & \vdots \\ x_1(K_{tr}) & \cdots & x_N(K_{tr}) & 1 \end{bmatrix}$$

→ Minimize the distance to the target output $\boldsymbol{y}$

$$\min_{W} |\boldsymbol{y} - \mathbf{S}W|^2$$

→ $$W = [(\mathbf{S}^T\mathbf{S} + \lambda\mathbf{1})^{-1}\mathbf{S}^T\boldsymbol{y}]^T$$

Example reservoir: Echo State Network

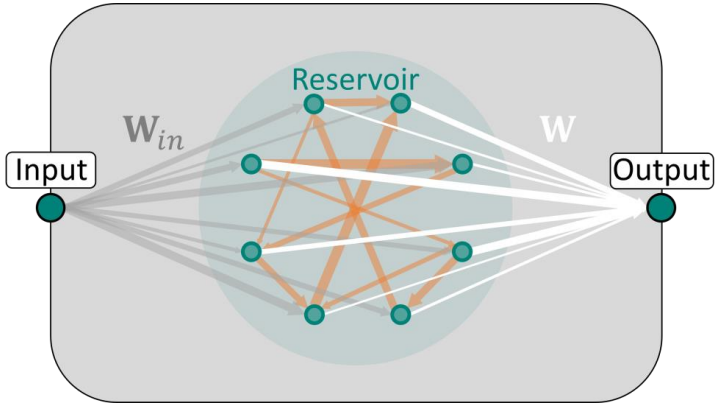
Internal coupling matrix
(adjacency matrix)

$$x(k+1) = \tanh(\boldsymbol{W_{int}} x(k) + \boldsymbol{w_{in}} I(k+1))$$

Vector of node states $x_i(k)$

Input weights vector

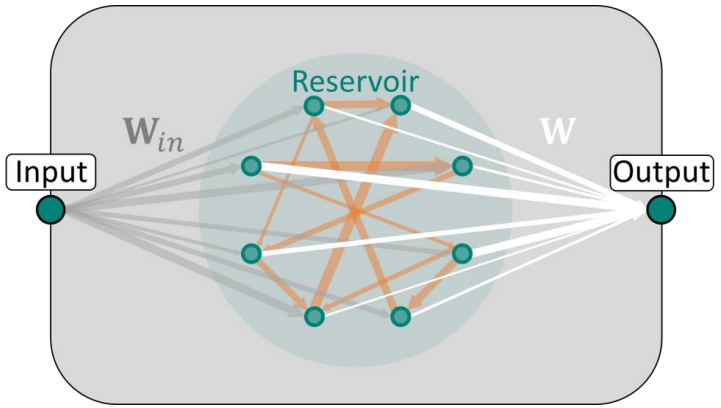
Input sequence



- Nonlinear transform performed by the reservoir depends on the input strength and previous states of the reservoir.

$$\boldsymbol{S} = \begin{bmatrix} x_1(1) & \cdots & x_N(1) & 1 \\ \vdots & \ddots & \vdots & \vdots \\ x_1(k) & \cdots & x_N(k) & 1 \\ \vdots & \ddots & \vdots & \vdots \\ x_1(K_{tr}) & \cdots & x_N(K_{tr}) & 1 \end{bmatrix}$$

Example reservoir: Echo State Network



Internal coupling matrix (adjacency matrix)

$$x(k+1) = \tanh(\boldsymbol{W_{int}} x(k) + \boldsymbol{w_{in}} I(k+1))$$

Vector of node states $x_i(k)$

Input weights vector

Input sequence

$$\tanh(x) \approx x - \frac{x^3}{3} + \frac{2x^5}{15} + \cdots$$ ← Expansion about $x_0 = 0$

$$\begin{aligned}\tanh(x) \approx\ & 0.76159 + 0.41997(x-1) \\ & - 0.31985(x-1)^2 \\ & + 0.10360(x-1)^3 \\ & + 0.02771(x-1)^4 + \cdots\end{aligned}$$ ← Expansion about $x_0 = 1$

$$\boldsymbol{S} = \begin{bmatrix} x_1(1) & \cdots & x_N(1) & 1 \\ \vdots & \ddots & \vdots & \vdots \\ x_1(k) & \cdots & x_N(k) & 1 \\ \vdots & \ddots & \vdots & \vdots \\ x_1(K_{tr}) & \cdots & x_N(K_{tr}) & 1 \end{bmatrix}$$

Extreme learning machine

$\boldsymbol{W_{int}} = \mathbf{0}$ - no internal coupling

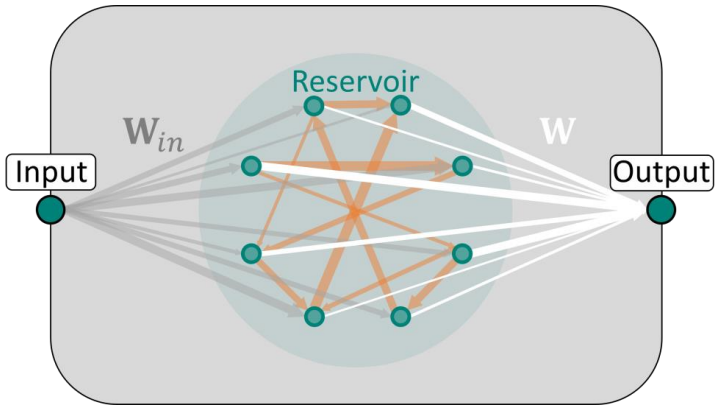
Internal coupling matrix (adjacency matrix)

$$\boldsymbol{x}(k+1) = \tanh(\boldsymbol{W_{int}}\boldsymbol{x}(k) + \boldsymbol{w_{in}}I(k+1))$$

Vector of node states $x_i(k)$

Input weights vector

Input sequence

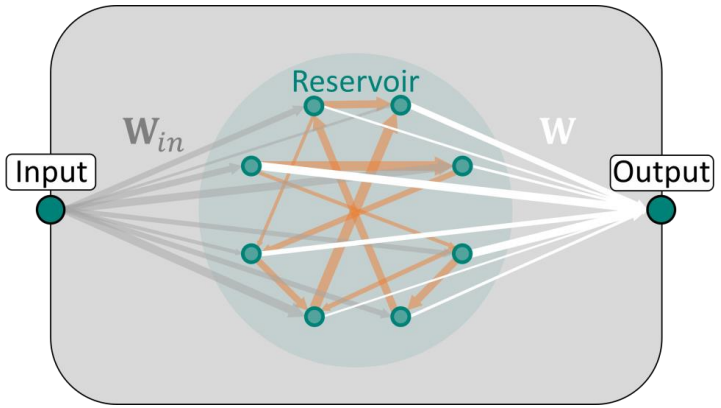


$$\boldsymbol{S} = \begin{bmatrix} x_1(1) & \cdots & x_N(1) & 1 \\ \vdots & \ddots & \vdots & \vdots \\ x_1(k) & \cdots & x_N(k) & 1 \\ \vdots & \ddots & \vdots & \vdots \\ x_1(K_{tr}) & \cdots & x_N(K_{tr}) & 1 \end{bmatrix}$$

📖 GB Huang, QY Zhu and CK Siew, *IEEE Proc.*. **2**, 985-90 (2004).

Extreme learning machine

$\boldsymbol{W_{int}} = \mathbf{0}$ - no internal coupling



$$x(k+1) = \tanh(\boldsymbol{w_{in}} I(k+1))$$

Input weights vector

Input sequence

$$\mathbf{S} = \begin{bmatrix} x_1(1) & \cdots & x_N(1) & 1 \\ \vdots & \ddots & \vdots & \vdots \\ x_1(k) & \cdots & x_N(k) & 1 \\ \vdots & \ddots & \vdots & \vdots \\ x_1(K_{tr}) & \cdots & x_N(K_{tr}) & 1 \end{bmatrix}$$

- Nonlinear transform performed by the extreme learning machine depends on the input strength
- No memory of past inputs

📖 GB Huang, QY Zhu and CK Siew, *IEEE Proc.*. **2**, 985-90 (2004).

Nonlinear Vector Regression

Input sequence $I(k)$
→ Feature vector with chosen nonlinearities and past inputs:

$$[1 \quad I(k) \quad I(k-1) \quad I(k)^2 \quad I(k-1)^2 \quad I(k)I(k-1) \quad \cdots]$$

→ Create state matrix out of sequential feature vector for the training phase:

$$\mathbf{S} = \begin{bmatrix} 1 & I(1) & I(0) & I(1)^2 & I(0)^2 & I(1)I(0) & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & I(k) & I(k-1) & I(k)^2 & I(k-1)^2 & I(k)I(k-1) & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & I(K_{tr}) & I(K_{tr}-1) & I(K_{tr})^2 & I(K_{tr}-1)^2 & I(K_{tr})I(K_{tr}-1) & \cdots \end{bmatrix}$$

→ Solve of the output weights via linear (Ridge) regression:

$$\mathbb{W} = [(\mathbf{S}^T\mathbf{S} + \lambda\mathbf{1})^{-1}\mathbf{S}^T\boldsymbol{o}]^T$$

"Next generation Reservoir Computing" 🕮 D. J. Gauthier, et al., *Nat. Commun.* **12(1)**, 5564 (2021).

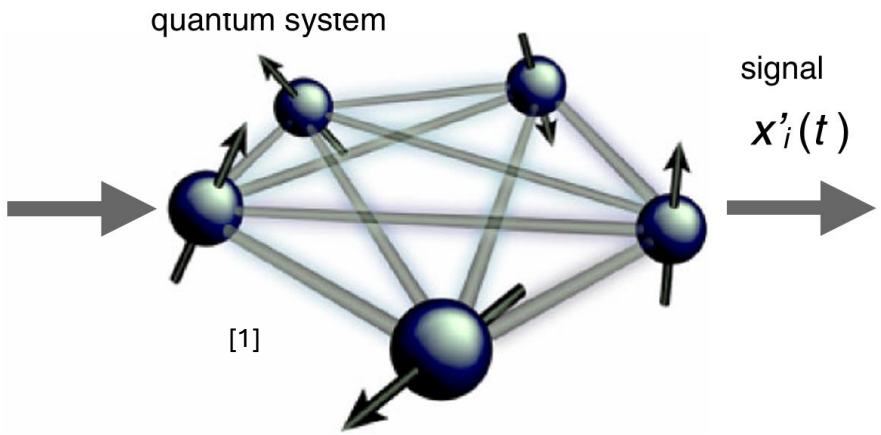
Quantum Reservoir Computer

[1] Image source: K. Fujii et al., Phys. Rev. Applied 8, (2017)

Fully connected transverse field Ising model

Hamiltonian:

$$H = \sum_{i=1, j>i}^{N_S} J_{ij} X_i X_j + \sum_{i=1}^{N_S} h Z_i$$



Time evolution of closed quantum system:

$$|\psi(t)\rangle = e^{-\frac{i}{\hbar} H t} |\psi_0\rangle$$

How do observables, like the Pauli *Z* operator, evolve in time?

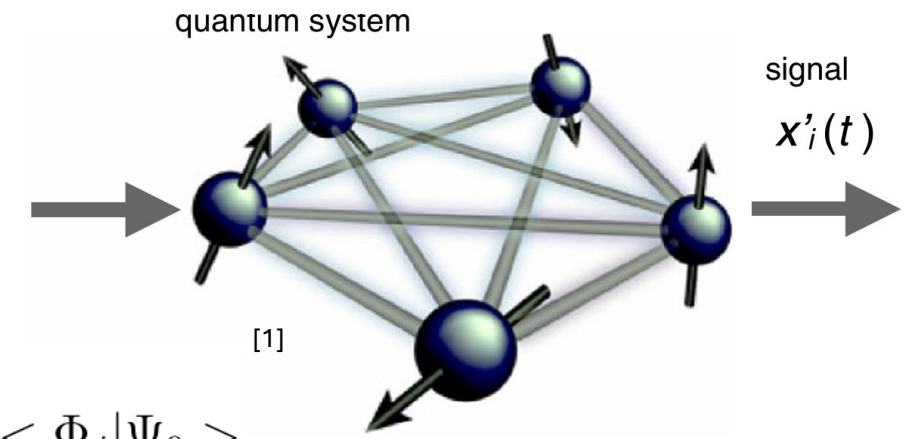
How can these dynamics be utilized for computation?

📖 Saud Čindrak, Master Thesis, TU Berlin (2022)

Quantum Reservoir Computer

[1] Image source: K. Fujii et al., Phys. Rev. Applied 8, (2017)

Fully connected transverse field Ising model



Example: 3 spins

Initial state

$$< Z_p > (t) =< \Psi|Z_p|\Psi > = \sum_{i=1,j=1}^{8} e^{i(\varepsilon_i-\varepsilon_j)t} < \Psi_0|\Phi_i >< \Phi_i|Z_p|\Phi_j >< \Phi_j|\Psi_0 >$$

Eigenstates

$$= \sum_{i=1,j>i}^{8} 2\Big(\mathrm{Re}(A_{i,j}^{(p)})\cos(\omega_{i,j}t) - \mathrm{Im}(A_{i,j}^{(p)})\sin(\omega_{i,j}t)\Big)$$

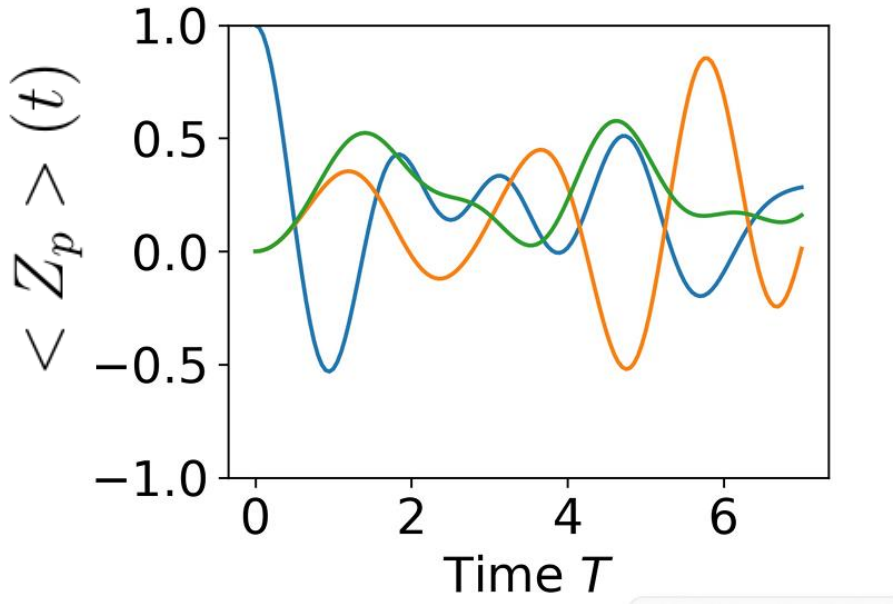
📖 Saud Čindrak, Master Thesis, TU Berlin (2022)

Quantum Reservoir Computer

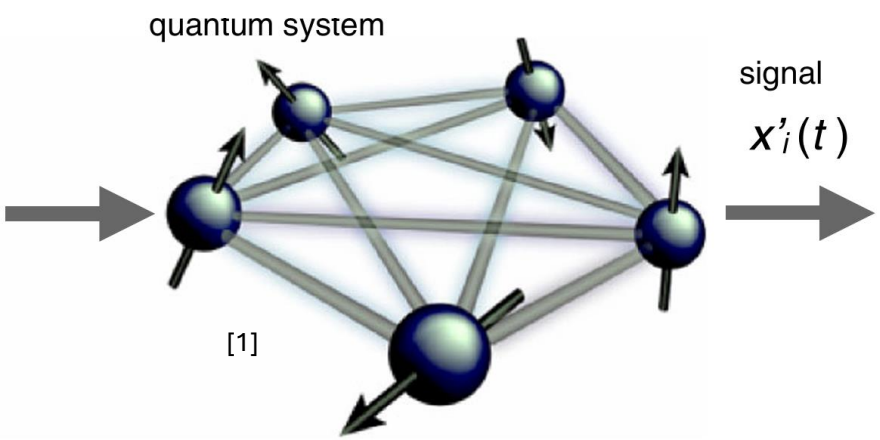
Fully connected transverse field Ising model

Example: 3 spins

$$< Z_p > (t) \; = \sum_{i=1, j>i}^{8} 2\Big(\mathrm{Re}(A_{i,j}^{(p)}) \cos(\omega_{i,j} t) - \mathrm{Im}(A_{i,j}^{(p)}) \sin(\omega_{i,j} t) \Big)$$



[1] Image source: K. Fujii et al., Phys. Rev. Applied 8, (2017)



- Amplitude depends on the initial state of the system
- Dynamics are periodic on long time scales

📖 Saud Čindrak, Master Thesis, TU Berlin (2022)

Quantum Reservoir Computer

Fully connected transverse field Ising model

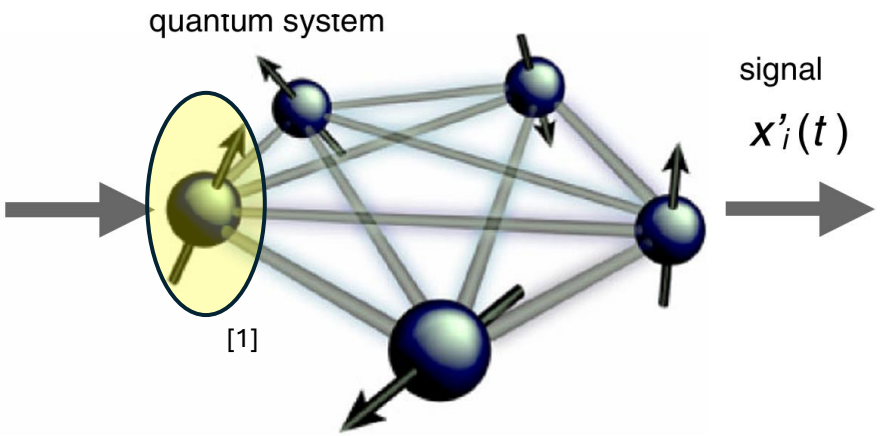
Input encoding:

Input encoded in the state of spin 1

$$|\Psi_E(u_i)\rangle = \sqrt{\frac{1-u_i}{2}}|0\rangle + \sqrt{\frac{1+u_i}{2}}|1\rangle$$

u_i - input at time step i

[1] Image source: K. Fujii et al., Phys. Rev. Applied 8, (2017)



Remaining spins hold information about previous inputs

Over writing of the time evolved state of spin 1 induces fading memory

📖 S. Čindrak, B. Donvil, K. Lüdge, **L.C. Jaurigue**, *Phys. Rev. Research* **6** (1): 013051 (2024).

Quantum Reservoir Computer

Fully connected transverse field Ising model

Input encoding:

Input encoded in the state of spin 1

$$|\Psi_E(u_i)\rangle = \sqrt{\frac{1-u_i}{2}}|0\rangle + \sqrt{\frac{1+u_i}{2}}|1\rangle$$

u_i - input at time step i

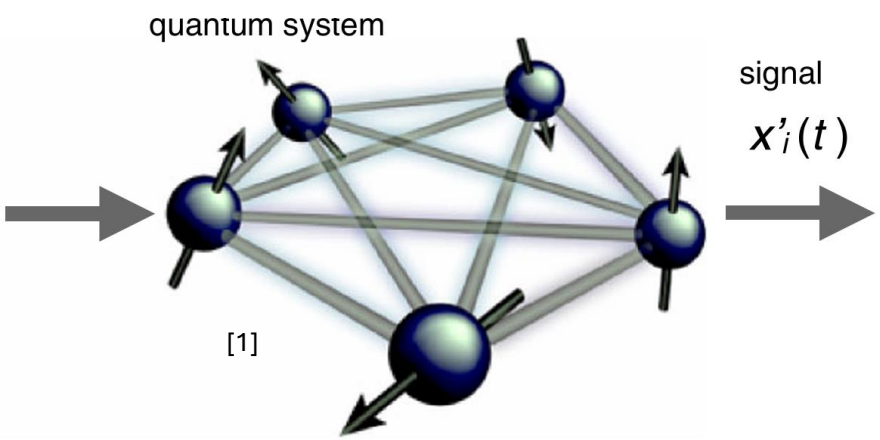
Sampled reservoir responses:

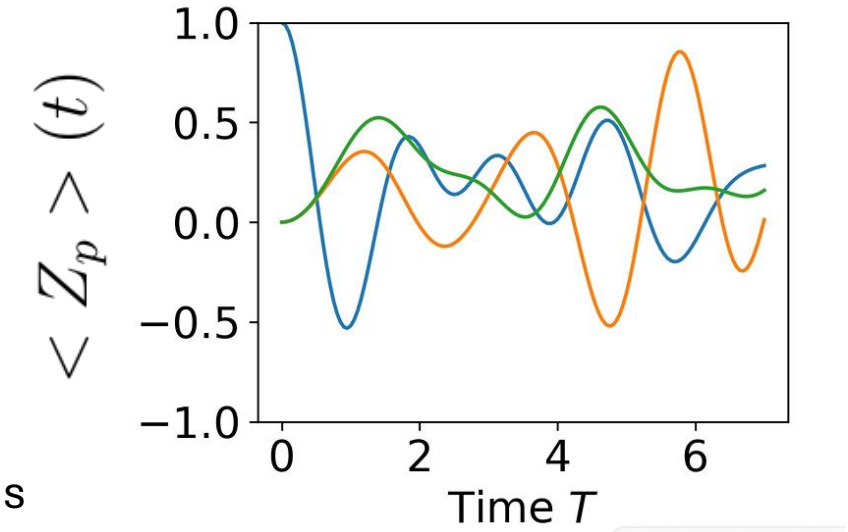
Time-multiplexed expectation values of Pauli-Z operator

- For each input the system is evolved for time T. The expectation value for each spin is measured N_v during this time interval

→ number of state matrix features is $N \times N_v$, where N is the number of spins

[1] Image source: K. Fujii et al., Phys. Rev. Applied 8, (2017)





📖 S. Čindrak, B. Donvil, K. Lüdge, **L.C. Jaurigue**, *Phys. Rev. Research* **6** (1): 013051 (2024).

📖 Saud Čindrak, Master Thesis, TU Berlin (2022)

Outline

- Reservoir computing and related methods
  - Training a reservoir computer
  - Comparison with nonlinear vector regression
  - Quantum reservoir computing – Ising model
- **Benchmarking tasks – what are they testing?**
  - **Lorenz 63 timeseries prediction**
  - **Information processing capacity (IPC)**
- Performance of a quantum reservoir
  - Lorenz timeseries tasks and informations processing capacity
  - Performance tuning via memory restriction
- Memory augmentation methods
  - Input delay and delayed statematrix concatenation

Benchmarking tasks

How can the performance be assessed/benchmarked?

→ image classification tasks

→ timeseries forecasting tasks

→ expressivity/information processing measures

Common timeseries tasks:

- NARMA
- Santa Fe (chaotic laser)
- Lorenz 63
- Maskey-Glass

Task independent measures:

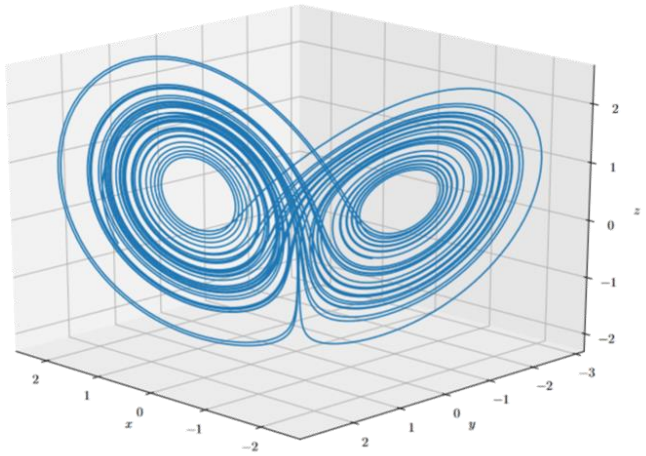
- Information processing capacity (IPC)
- Covariance rank

Timeseries forecasting task – Lorenz 63

Lorenz system

$$\frac{dx}{dt} = c_1 y - c_1 x, \quad \frac{dy}{dt} = x(c_2 - z) - y, \quad \text{and} \quad \frac{dz}{dt} = xy - c_3 z$$

$$c_1 = 10$$
$$c_2 = 28$$
$$c_3 = 8/3$$



Chaotic dynamics:

- Trajectories are sensitivity to the initial conditions
- Timeseries are difficult to predict

What is the Lorenz timeseries prediction task?

→ There are many and they all require different reservoir properties

Task options: $(t = k\Delta t)$

- Temporal discretization Δt
- Input variables, e.g. only $x(k)$, or a combination of $x(k), y(k), z(k)$, etc.
- Target variables, e.g. $x(k+1)$, $x(k+n)$ or $z(k)$, etc.
- Prediction mode, e.g. open-loop or closed loop (autonomous)

Timeseries forecasting task – Lorenz 63

Let us consider the cases:

$$x(k) \rightarrow x(k+1)$$

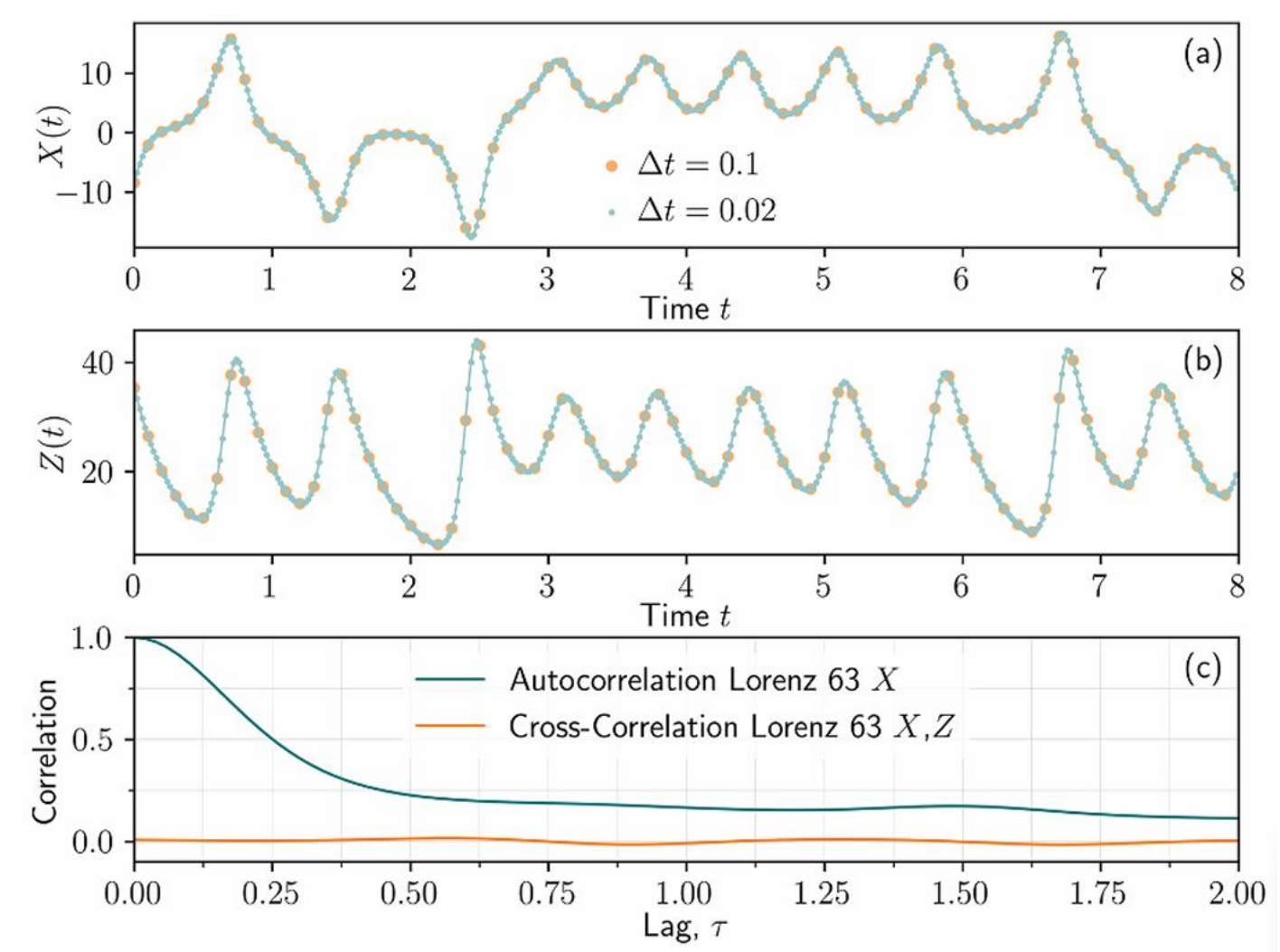
and

$$x(k) \rightarrow z(k)$$

Input

Target

Partial information tasks – the reservoir is not provided with full information about the system of the target system
→ delay embedding may be required
→ memory is required



- **L.C. Jaurigue**, K. Lüdge, *Neuromorph. Comput. Eng.* **4**, 014001 (2024).
- H. Kantz and T. Schreiber, *Nonlinear Time Series Analysis*, Cambridge University Press (2003).
- L. Storm, K. Gustavsson, B. Mehlig, *Mach. Learn.: Sci. Technol.* **3**, 045021 (2022).

Timeseries forecasting task – Lorenz 63

Let us consider the cases:

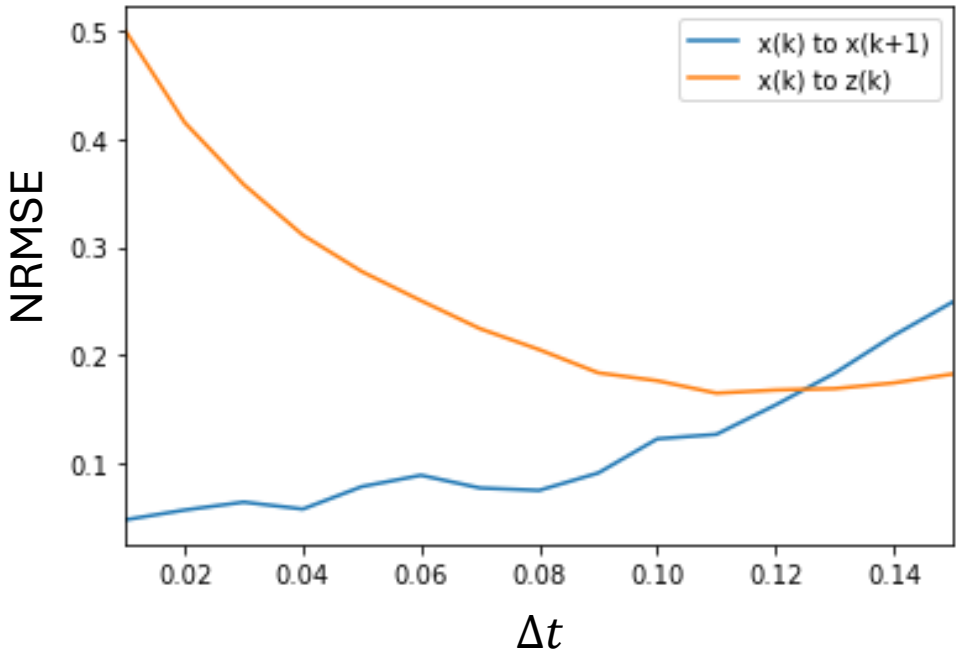
$$x(k) \rightarrow x(k+1)$$

and

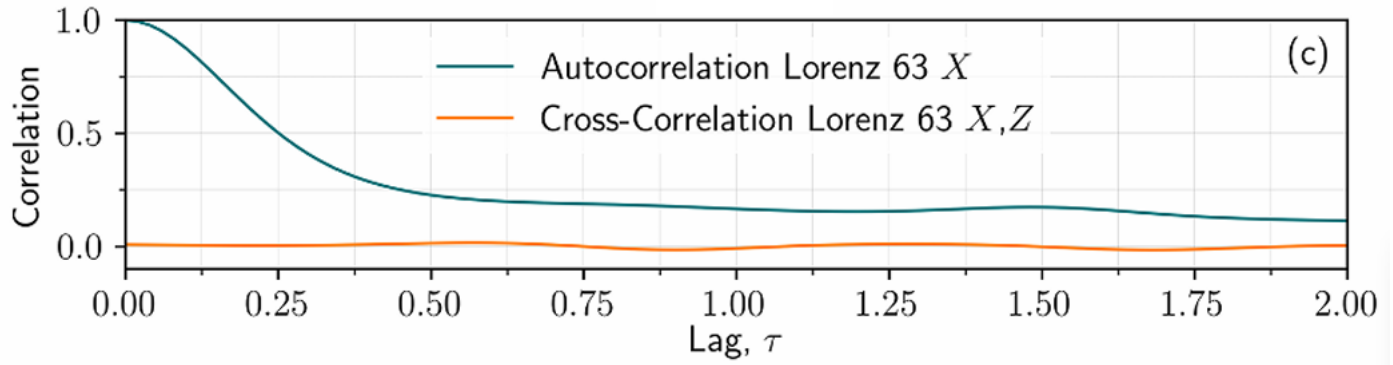
$$x(k) \rightarrow z(k)$$

Input

Target



Partial information tasks – the reservoir is not provided with full information about the system of the target system
→ delay embedding may be required
→ memory is required



- **L.C. Jaurigue**, K. Lüdge, *Neuromorph. Comput. Eng.* **4**, 014001 (2024).
- H. Kantz and T. Schreiber, *Nonlinear Time Series Analysis*, Cambridge University Press (2003).
- L. Storm, K. Gustavsson, B. Mehlig, *Mach. Learn.: Sci. Technol.* **3**, 045021 (2022).

Timeseries forecasting task – Lorenz 63

Let us consider the cases:

$$x(k) \rightarrow x(k+1)$$

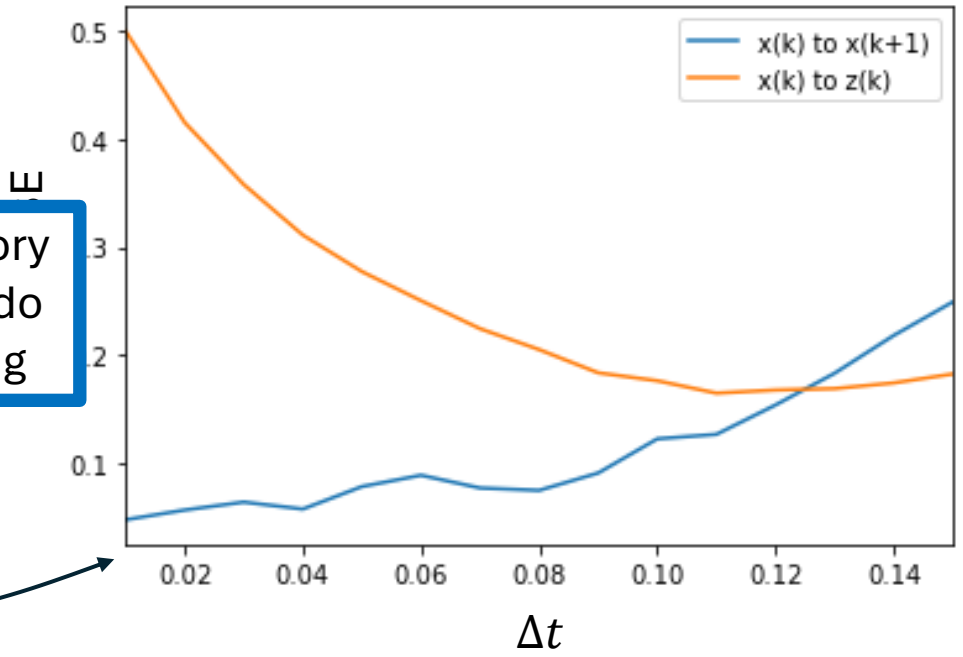
and

$$x(k) \rightarrow z(k)$$

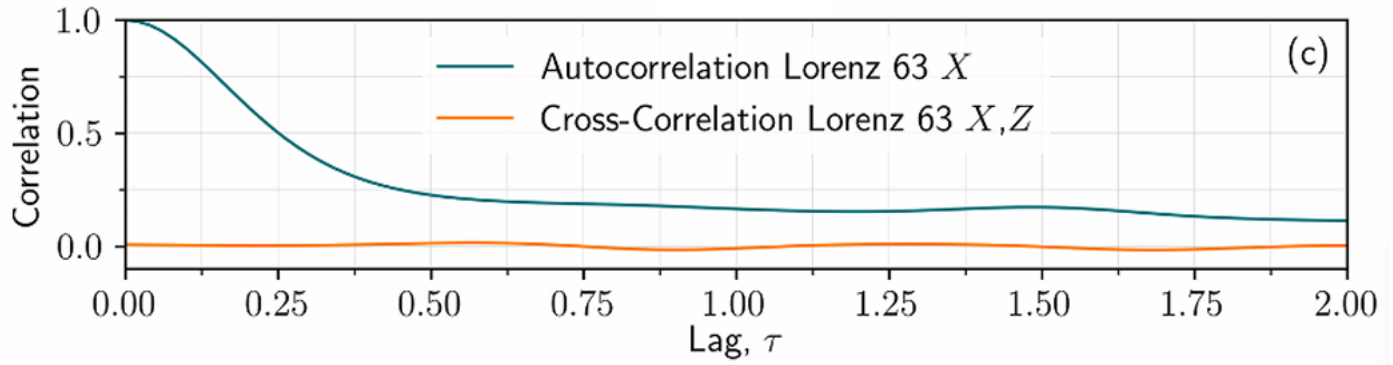
Input

Target

Reservoir requires memory on longer timescales to do a good delay-embedding



Partial information tasks – the reservoir is not provided with full information about the system of the target system
→ delay embedding may be required
→ memory is required



- **L.C. Jaurigue**, K. Lüdge, *Neuromorph. Comput. Eng.* **4**, 014001 (2024).
- H. Kantz and T. Schreiber, *Nonlinear Time Series Analysis*, Cambridge University Press (2003).
- L. Storm, K. Gustavsson, B. Mehlig, *Mach. Learn.: Sci. Technol.* **3**, 045021 (2022).

Information Processing capacity (IPC)

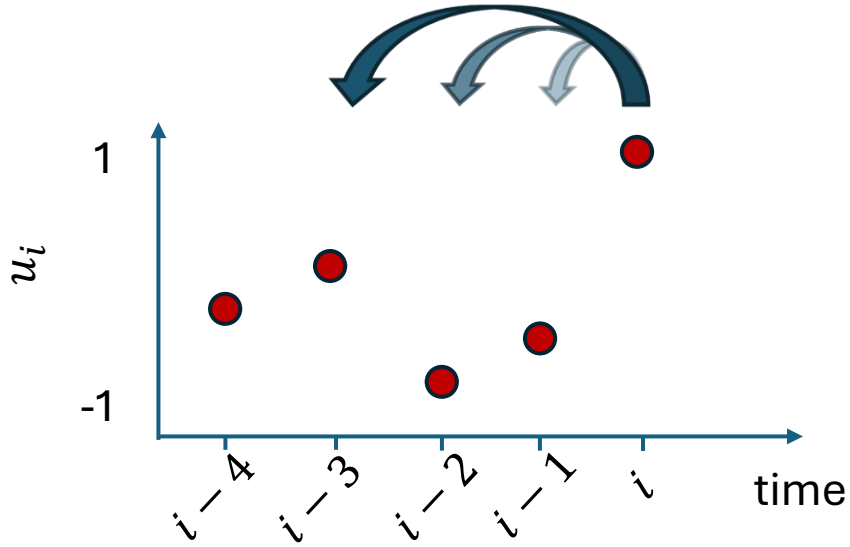
Measure of the nonlinear transform and memory capabilities of a reservoir

📖 J Dambre, D Verstraeten, B Schrauwen and S Massar, *Scientific Reports*, 2:514 (2012).

Input sequence:
Sequence of u_i drawn from a uniform distribution on the interval $[-1,1]$

Task:
Construct all possible combinations of past inputs

Example: linear components (i.e. memory capacity)
- recall of input m steps in the past



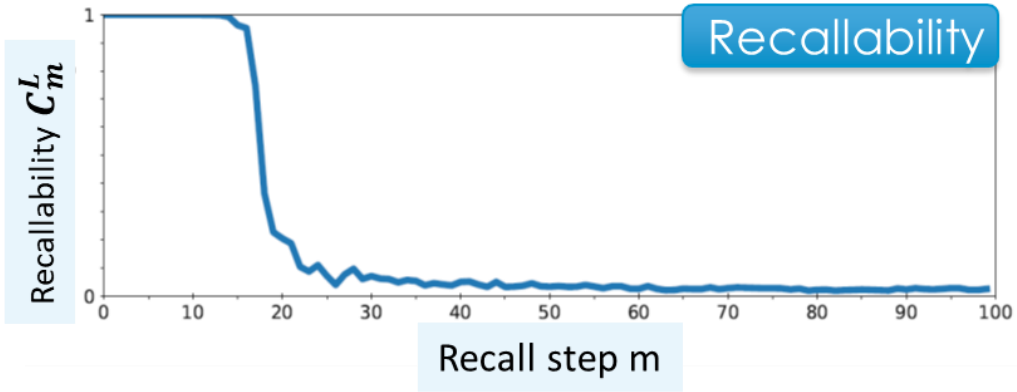
Recallability

$$C_m^L = 1 - \mathrm{NMSE}(\hat{o}^m)$$

Target: $\hat{o}_k^m = u_{k-m}$ input m steps ago

Linear memory capacity

$$C^L = \sum_{m=1}^{\infty} C_m^L$$



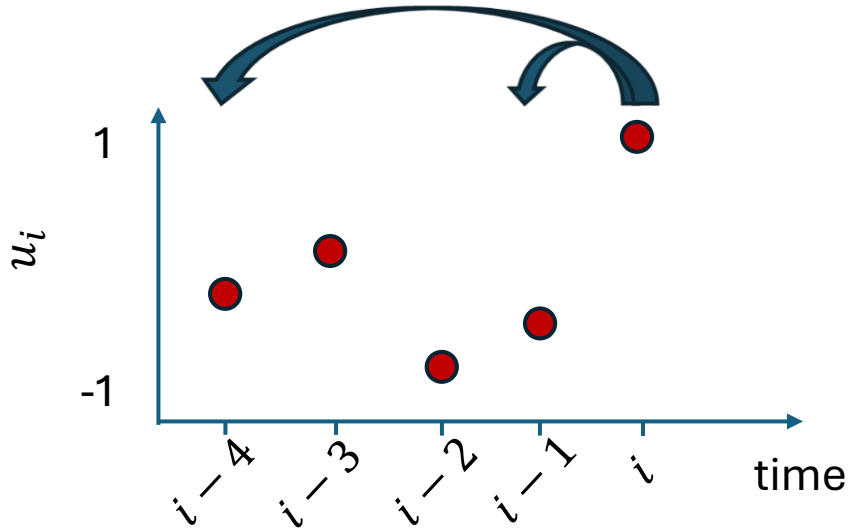
Information Processing capacity (IPC)

Measure of the nonlinear transform and memory capabilities of a reservoir

📖 J Dambre, D Verstraeten, B Schrauwen and S Massar, *Scientific Reports*, 2:514 (2012).

Input sequence:
Sequence of u_i drawn from a uniform distribution on the interval $[-1,1]$

Task:
Construct all possible combinations of past inputs



For higher order capacities the Legendre Polynomials which form a complete orthogonal basis when the input distribution is uniform on the interval $[-1,1]$

$$l_0(x) = 1$$
$$l_1(x) = x$$
$$l_2(x) = \frac{1}{2}(3x^2 - 1)$$
$$l_{n+1}(x) = \frac{2n+1}{n+1} x l_n(x) - \frac{n}{n+1} l_{n-1}(x), \text{ for } n \geq 2$$

IPC of degree k is the sum over all capacites of degree k:

$$\mathrm{IPC}_k = \sum_l C_{k,l}$$

Total IPC is the sum over all degrees:

$$\mathrm{IPC} = \sum_k \mathrm{IPC}_k$$

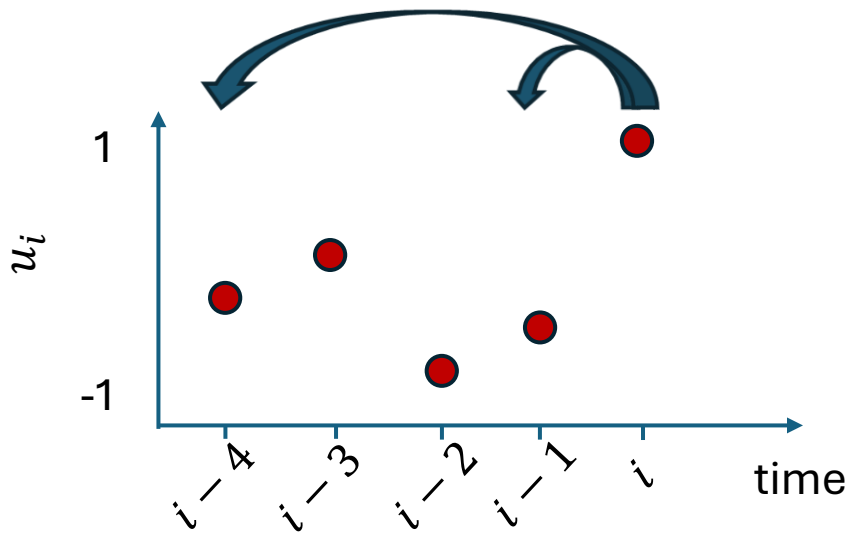
Information Processing capacity (IPC)

Measure of the nonlinear transform and memory capabilities of a reservoir

📖 J Dambre, D Verstraeten, B Schrauwen and S Massar, *Scientific Reports*, 2:514 (2012).

Input sequence:
Sequence of u_i drawn from a uniform distribution on the interval $[-1,1]$

Task:
Construct all possible combinations of past inputs



For higher order capacities the Legendre Polynomials which form a complete orthogonal basis when the input distribution is uniform on the interval $[-1,1]$

$$l_0(x) = 1$$
$$l_1(x) = x$$
$$l_2(x) = \frac{1}{2}(3x^2$$
$$l_{n+1}(x) = \frac{2n+$$
$$n+1 \qquad\qquad n+1$$

Upper bounded by the feature dimension of the state matrix

IPC of degree k is the sum over all capacites of degree k:

$$\mathrm{IPC}_k = \sum_l C_{k,l}$$

Total IPC is the sum over all degrees:

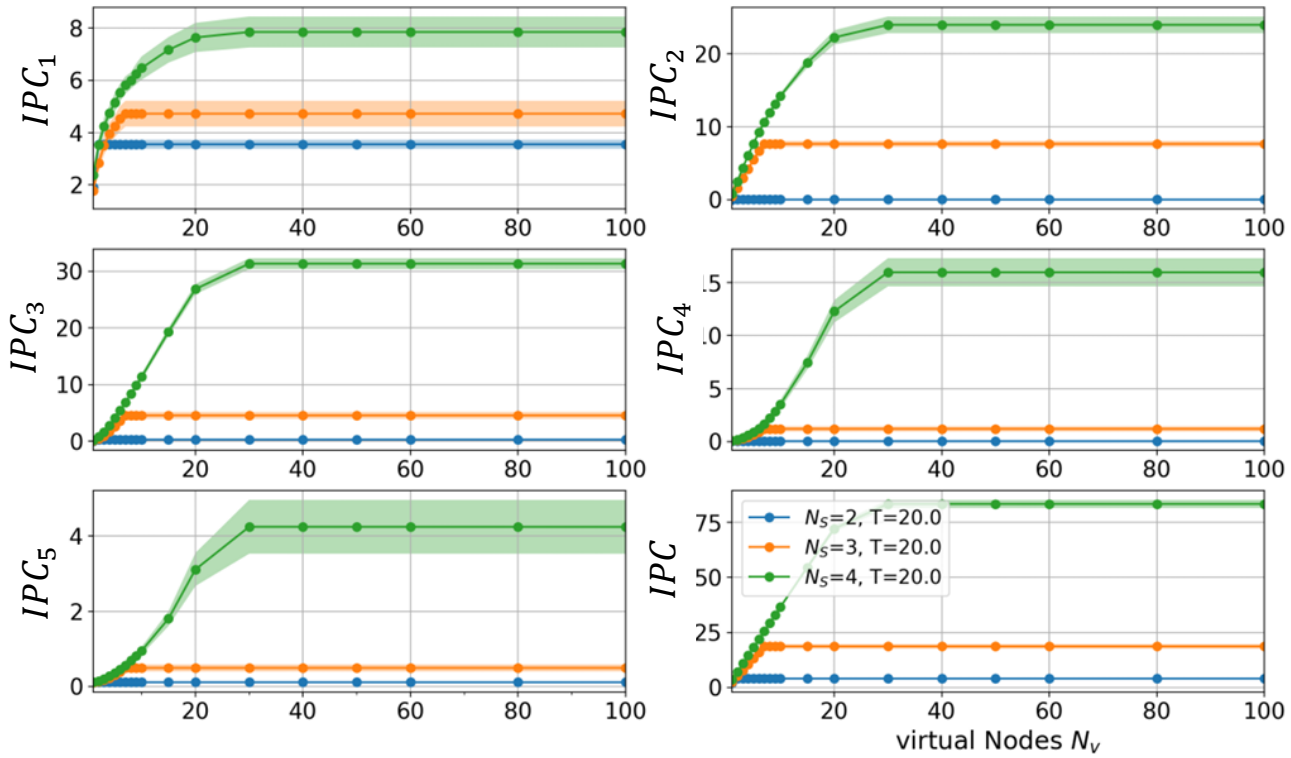
$$\mathrm{IPC} = \sum_k \mathrm{IPC}_k$$

Outline

- Reservoir computing and related methods
  - Training a reservoir computer
  - Comparison with nonlinear vector regression
  - Quantum reservoir computing – Ising model
- Benchmarking tasks – what are they testing?
  - Lorenz 63 timeseries prediction
  - Information processing capacity (IPC)
- **Performance of a quantum reservoir**
  - **Lorenz timeseries tasks and informations processing capacity**
  - **Performance tuning via memory restriction**
- Memory augmentation methods
  - Input delay and delayed statematrix concatenation

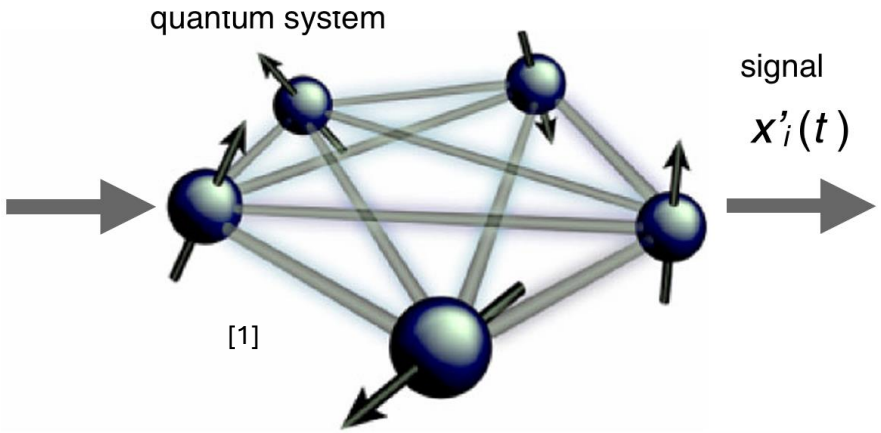
Quantum Reservoir Computer

IPCs of Ising model reservoirs



Maximum possible IPC is $N_s \times N_v$

[1] Image source: K. Fujii et al., Phys. Rev. Applied 8, (2017)



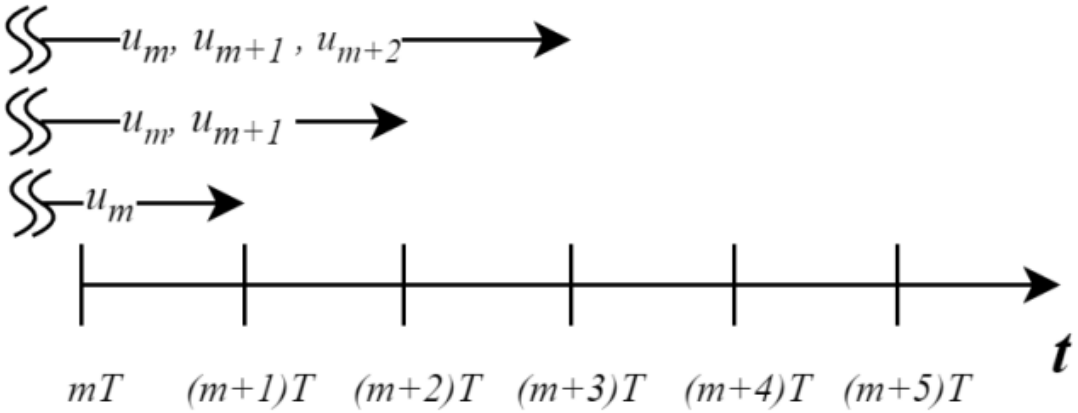
IPC does not generally correlation with task performance

📖 T. Hülser, F. Köster, K. Lüdge, and L. Jaurigue, *Nanophotonics*, 0415 (2022).

📖 Saud Čindrak, Master Thesis, TU Berlin (2022)

Quantum reservoir computing optimisation

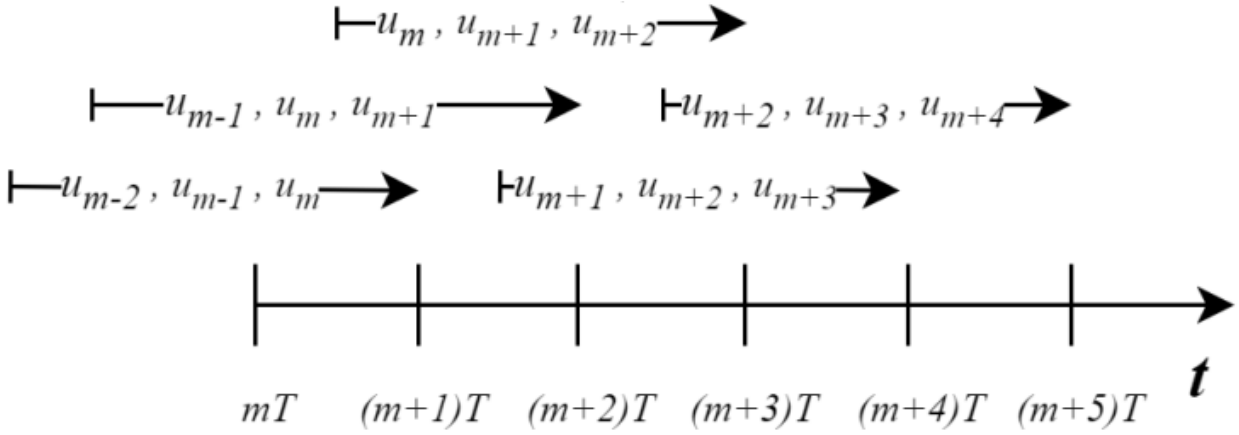
Time-complexity problem:
measurement collapse of the quantum states
→ system must be reinitialized from beginning after each measurement



Number of operations scale with M^2, where M is the length of the timeseries

Solution: Fixed reset length

→ The entire history isn't needed due to fading memory
→ reduced time-complexity



Number of operations scale with M

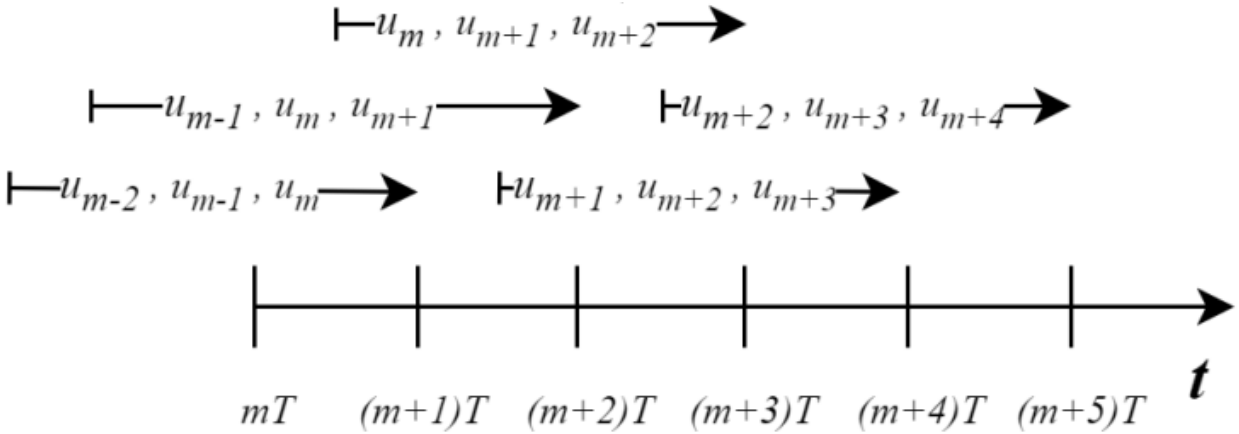
📖 S. Čindrak, B. Donvil, K. Lüdge, **L.C. Jaurigue**, *Phys. Rev. Research* **6** (1): 013051 (2024).

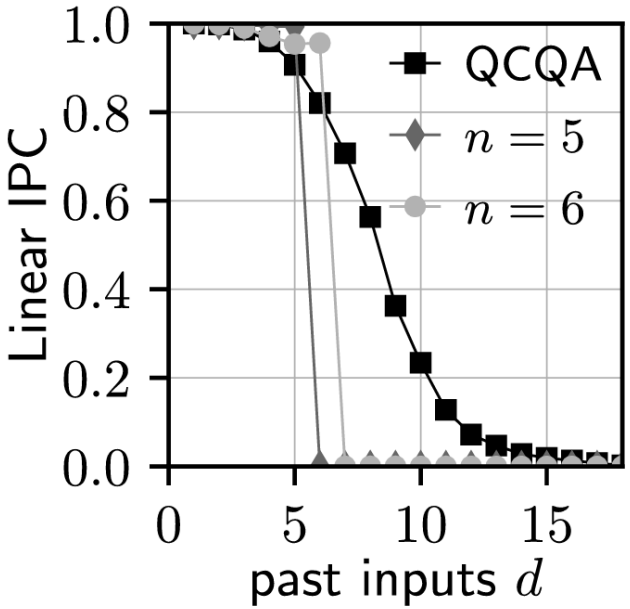
Quantum reservoir computing optimisation

Solution: Fixed reset length

→ The entire history isn't needed due to fading memory

→ reduced time-complexity





Number of operations scale with M

📖 S. Čindrak, B. Donvil, K. Lüdge, **L.C. Jaurigue**, *Phys. Rev. Research* **6** (1): 013051 (2024).

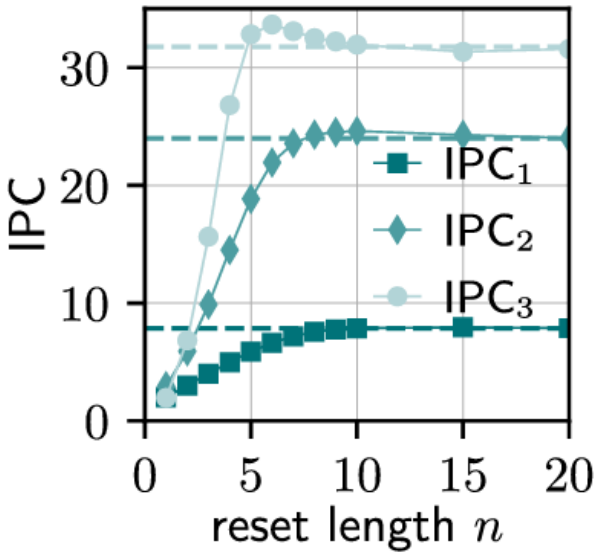
Quantum reservoir computing optimisation

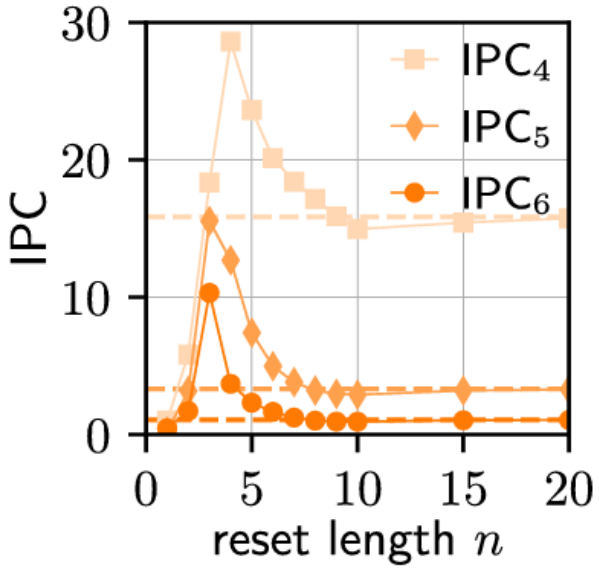
Solution: Fixed reset length

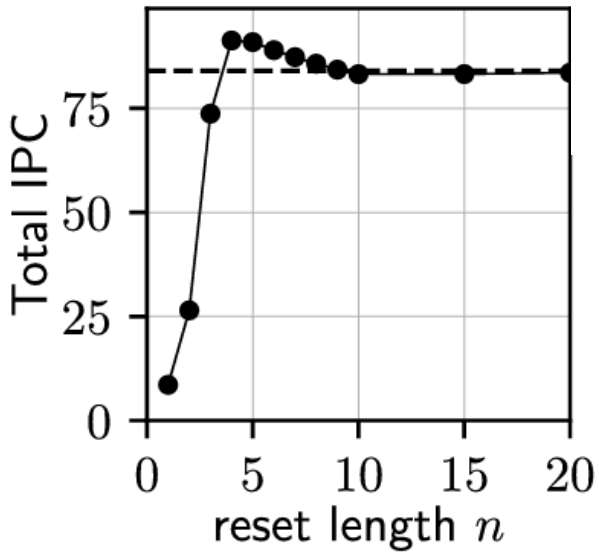
→ The entire history isn't needed due to fading memory

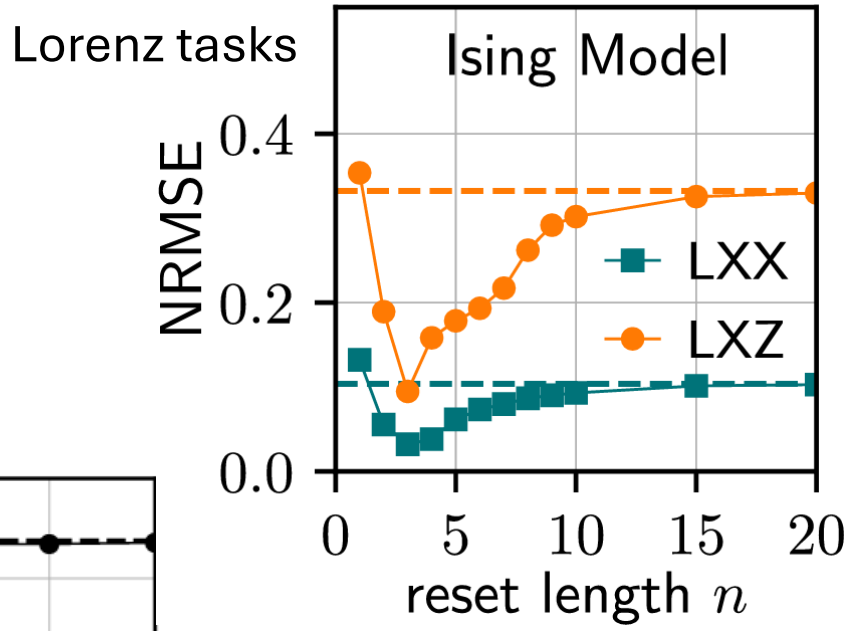
→ reduced time-complexity

→ **Improved performance**









Memory-nonlinearity trade-off

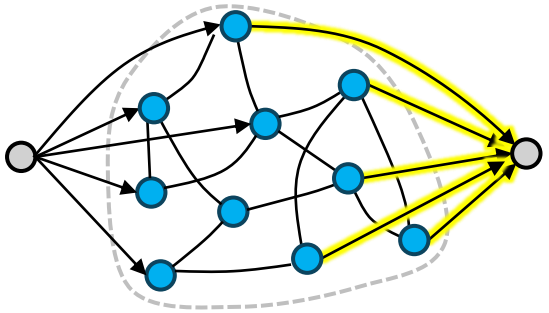
📖 S. Čindrak, B. Donvil, K. Lüdge, **L.C. Jaurigue**, *Phys. Rev. Research* **6** (1): 013051 (2024).

Outline

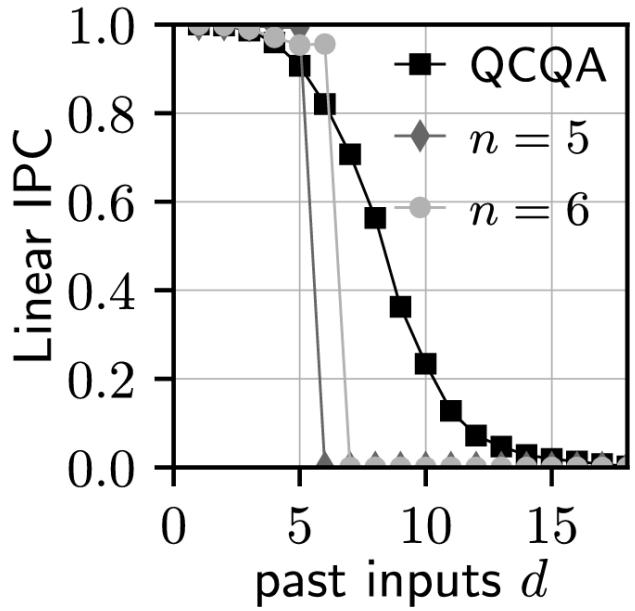
- Reservoir computing and related methods
  - Training a reservoir computer
  - Comparison with nonlinear vector regression
  - Quantum reservoir computing – Ising model
- Benchmarking tasks – what are they testing?
  - Lorenz 63 timeseries prediction
  - Information processing capacity (IPC)
- Performance of a quantum reservoir
  - Lorenz timeseries tasks and informations processing capacity
  - Performance tuning via memory restriction
- **Memory augmentation methods**
  - **Input delay and delayed statematrix concatenation**

Hybrid Reservoir Computing

The memory and nonlinearity of reservoir computing approaches can be augmented at the input and output layers



- Concatenation of the state matrix with
  - → nonlinear transforms of the state matrix, e.g. $[\mathbf{S}, \mathbf{S}^2]$
  - → delayed versions of the state matrix, e.g. $[\boldsymbol{s}_k, \boldsymbol{s}_{k-d}]$
- Preprocessing of the input sequence by
  - → addition of delayed inputs, e.g. $I(k) + I(k-d)$
  - → addition of nonlinear transforms of the input, e.g. $I(k) + I(k)^2$



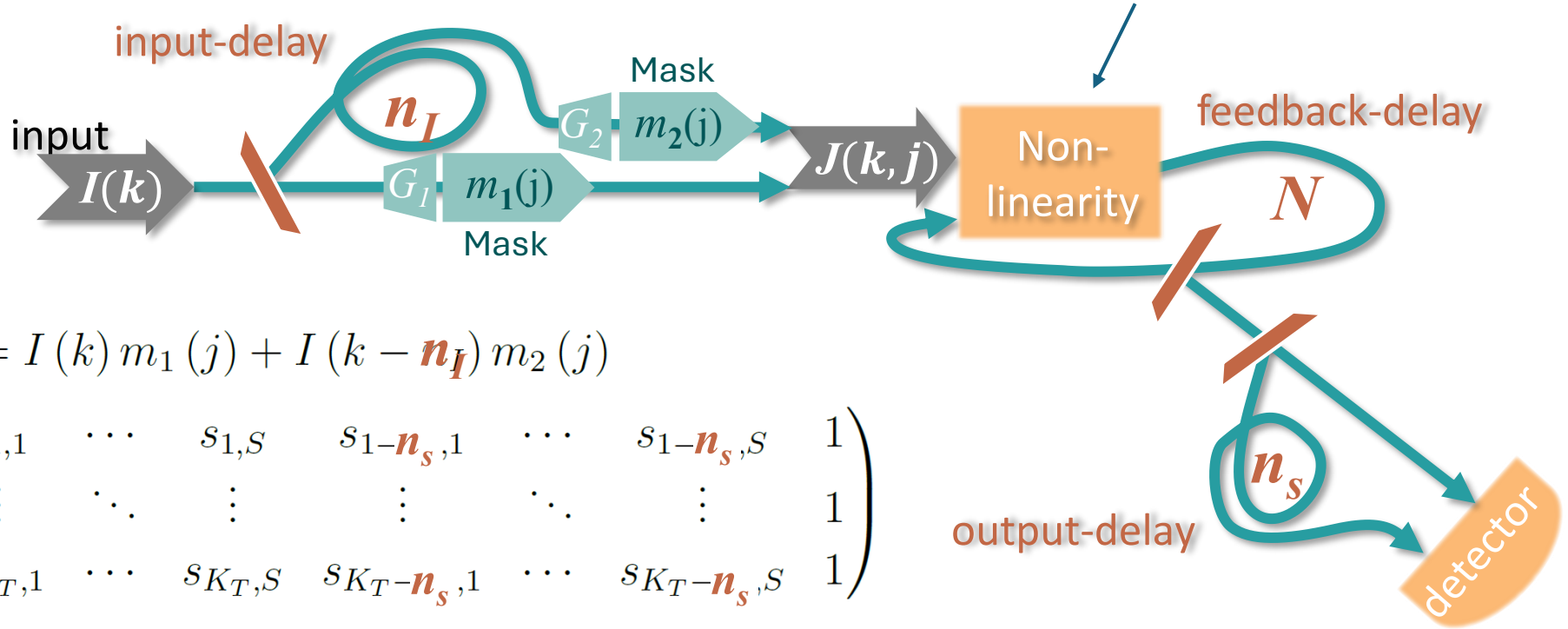
📖 K. Takano, C. Sugano … A. Uchida et al. , *Opt. Express* **26**, 29424 (2018).
📖 Y. Sakemi, K. Morino, T. Leleu and K. Aihara, *Scientific Reports* **10**, 21794 (2020).
📖 B. A. Marquez, J. Suarez-Vargas and B. J. Shastri Phys. Rev. Res. **1**, 033030, (2019).
📖 L.C. Jaurigue, L. Robertson, J. Wolters and K. Lüdge, Entropy **23**, 1560 (2021).
📖 L.C. Jaurigue, K. Lüdge, *Neuromorph. Comput. Eng.* **4**, 014001 (2024).

Memory augmentation methods

$$x(i) = \mathcal{G}\left(Kx(i-\boldsymbol{N}) + J(i)\right)$$

Semiconductor Optical Amplifier

$$\mathcal{G}(x) = \frac{g_0 x}{1 + x/a}$$



$$\tilde{J}(i) = I(k)\, m_1(j) + I(k - \boldsymbol{n_I})\, m_2(j)$$

$$\boldsymbol{S} = \begin{pmatrix} s_{1,1} & \cdots & s_{1,S} & s_{1-\boldsymbol{n_s},1} & \cdots & s_{1-\boldsymbol{n_s},S} & 1 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & 1 \\ s_{K_T,1} & \cdots & s_{K_T,S} & s_{K_T-\boldsymbol{n_s},1} & \cdots & s_{K_T-\boldsymbol{n_s},S} & 1 \end{pmatrix}$$

📖 L.C. Jaurigue, L. Robertson, J. Wolters and K. Lüdge, Entropy **23**, 1560 (2021).
📖 L.C. Jaurigue, K. Lüdge, *Neuromorph. Comput. Eng.* **4**, 014001 (2024).
📖 J Jaurigue, J Robertson, A Hurtado, L Jaurigue, K Lüdge, *Communications Engineering* **4** (1), 10 (2025)

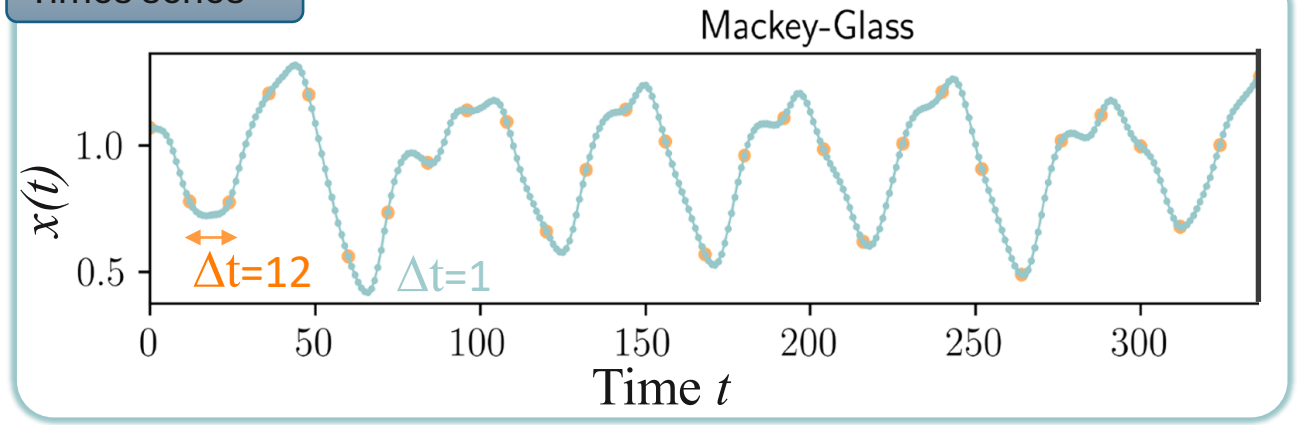
Mackey-Glass timeseries prediction

$$\frac{dx}{dt} = \beta \frac{x(t-\tau)}{1+x(t-\tau)^n} - \gamma x$$

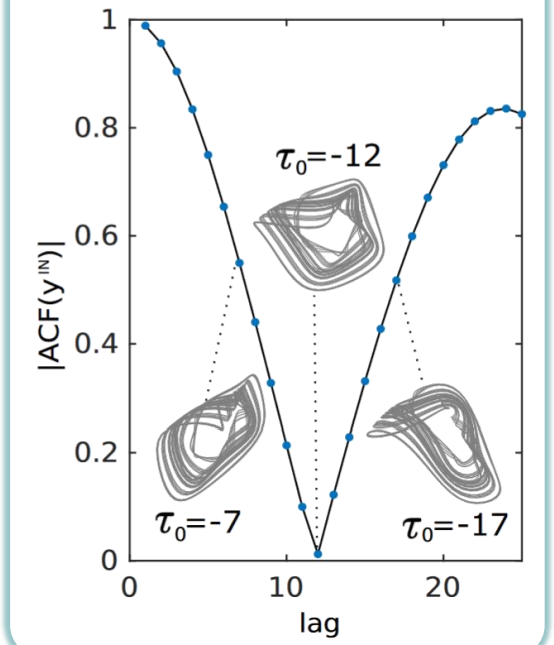
$\tau = 17,\ n = 10,\ \beta = 0.2$ and $\gamma = 0.1$

Time series prediction task defined by discretization Δt (sampling step size) and number of predicted steps s (prediction horizon $= s\Delta t$)

Times series



Auto-correlation



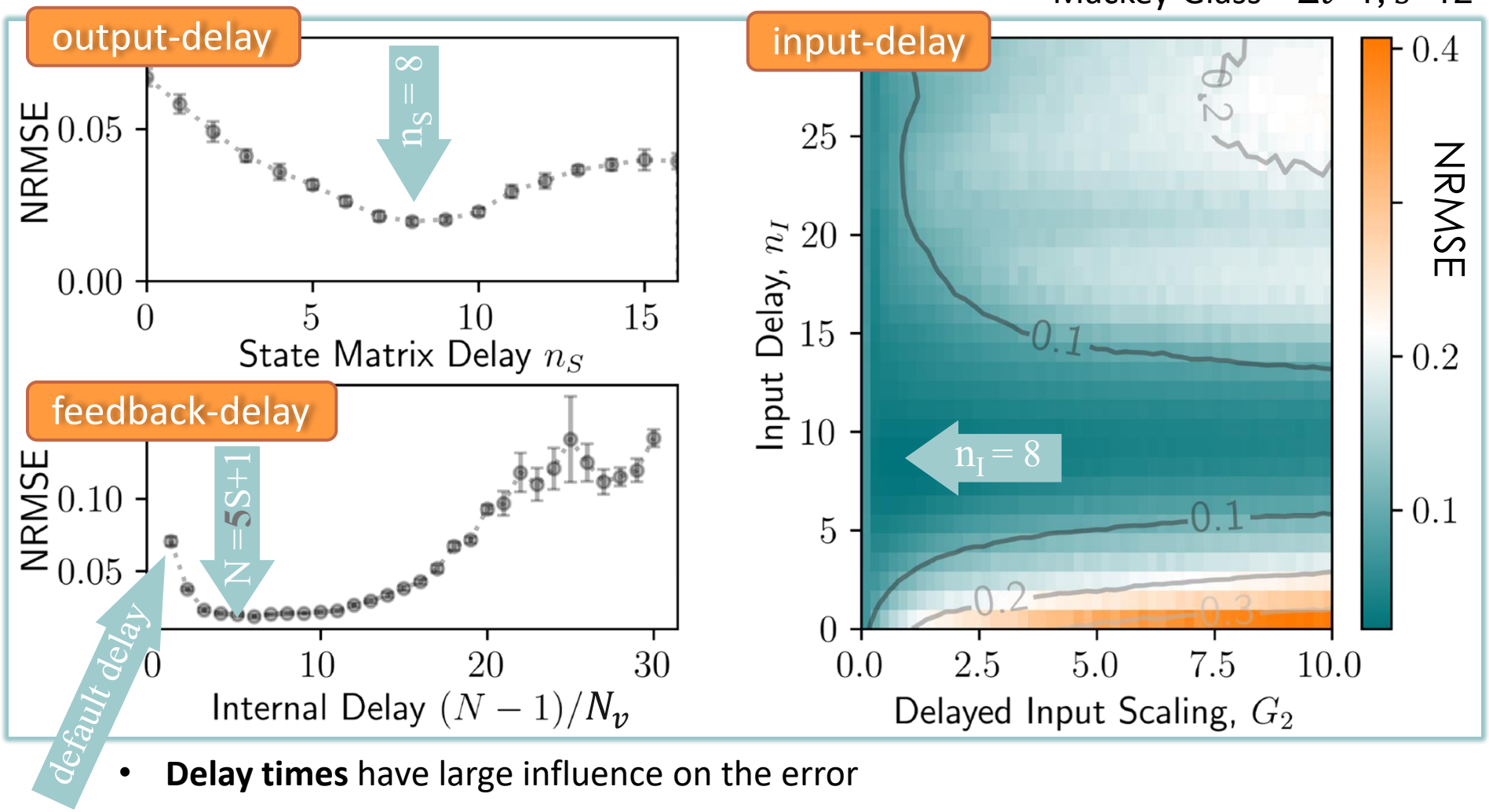
- Optimal embedding step (Takens) is Δt=12
- For small Δt the reservoir requires longer memory

📖 F. Takens, *Detecting strange attractors in turbulence, Springer* (1981).

📖 B. A. Marquez, J. Suarez-Vargas and B. J. Shastri Phys. Rev. Res. **1**, 033030, (2019).

Memory augmentation methods

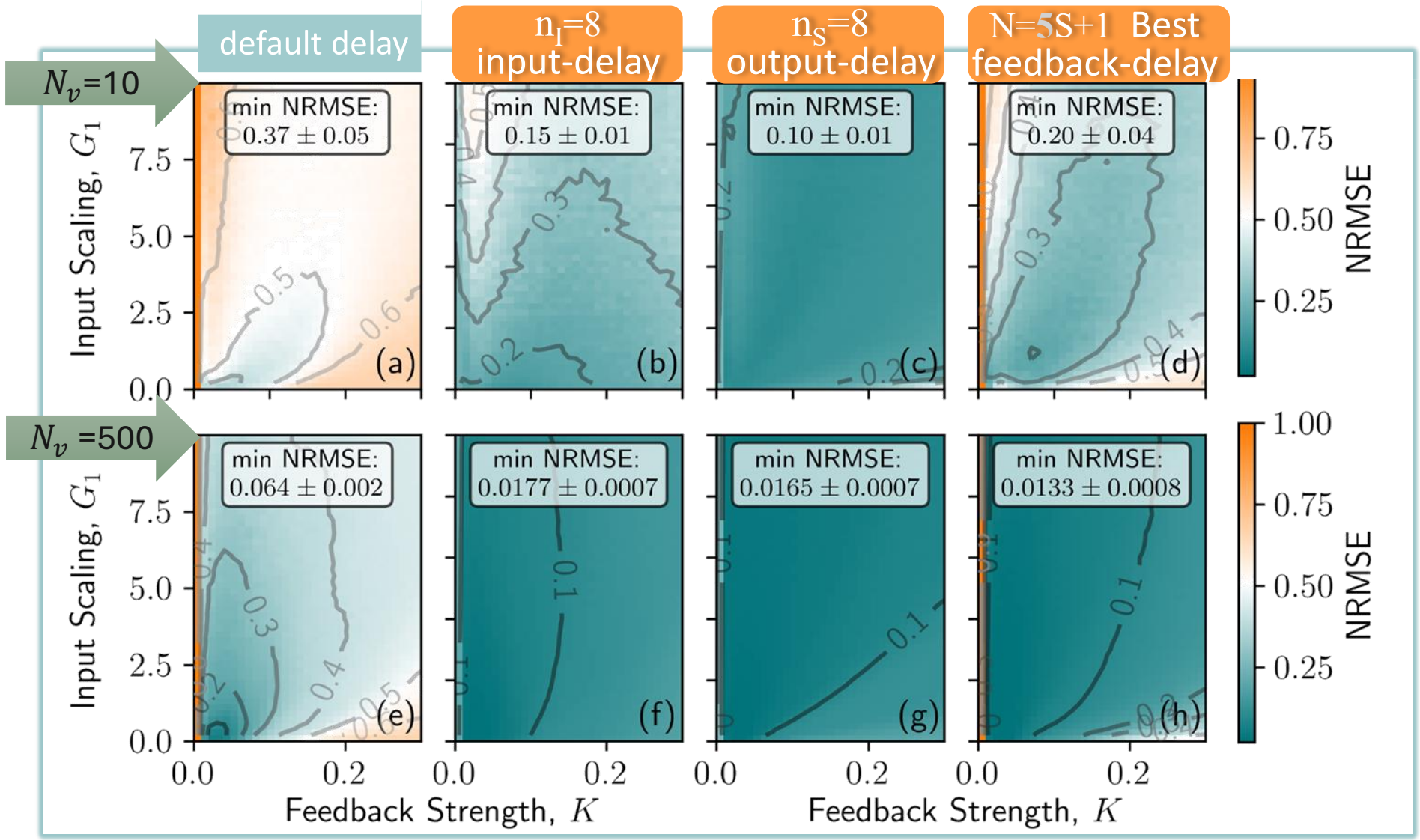
Mackey-Glass – Δt=1, s=12



- **Delay times** have large influence on the error
- **Optimal values** exist for each task (not identical for output, input or internal delay)

📖 L.C. Jaurigue, K. Lüdge, *Neuromorph. Comput. Eng.* **4**, 014001 (2024)

Memory augmentation methods



- Reduced hyperparameter sensitivity when the correct timescales are introduced

📖 L.C. Jaurigue, K. Lüdge, *Neuromorph. Comput. Eng.* **4**, 014001 (2024)

Krylov expressivity and Krylov observability

Can we quantify the computational potential of quantum systems?

→ Measurable Krylov expressivity calculated from time evolved quantum states

Can we predict the relative performance of quantum reservoir computers?

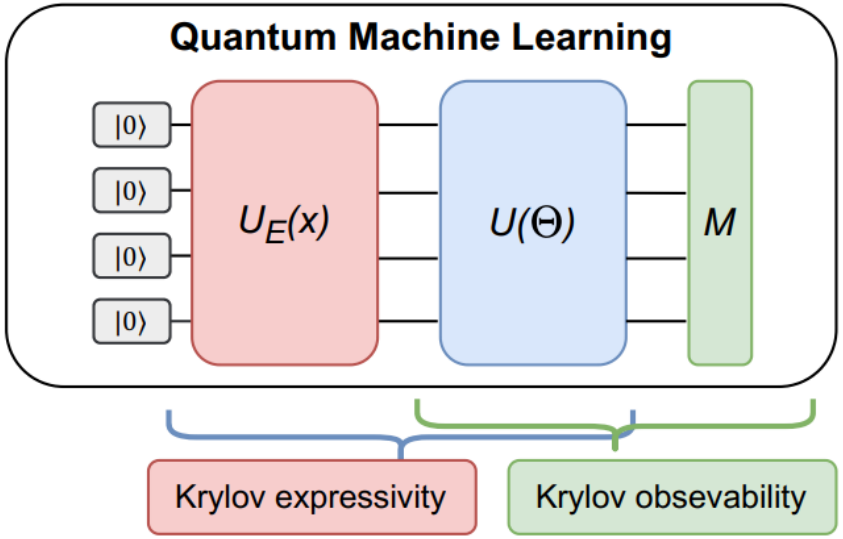
→ Krylov observability calculated from time evolved quantum observables

Doctoral Student
Saud Čindrak

Krylov space: smallest basis in which all time evolved states lie

Krylov expressivity:
→ explains how encoded states map to the Hilbert space of the trained unitary.

Krylov observability:
→ a measure of how much of the Hilbert space of the trained unitary can be sampled.

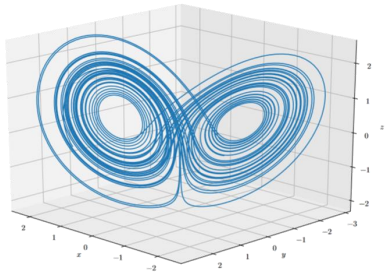


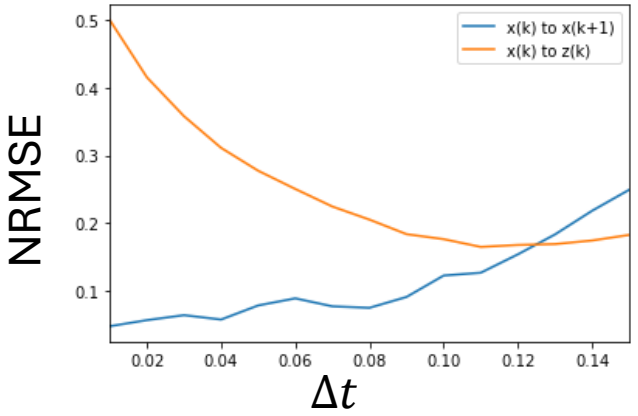
📖 S. Čindrak, A. Paschke, **L.C. Jaurigue**, K. Lüdge, *Journal of High Energy Physics* **10**, 1-21 (2024)
📖 S Čindrak, L Jaurigue, K Lüdge, arXiv preprint arXiv:2409.12079
📖 S Čindrak, K Lüdge, L Jaurigue, arXiv preprint arXiv:2502.12157

Summary

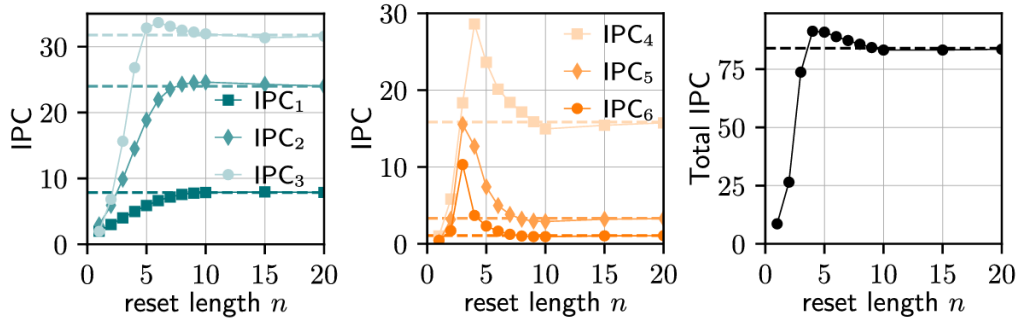
Timeseries prediction tasks – Details matter!

Input?
Target?
Discretization?





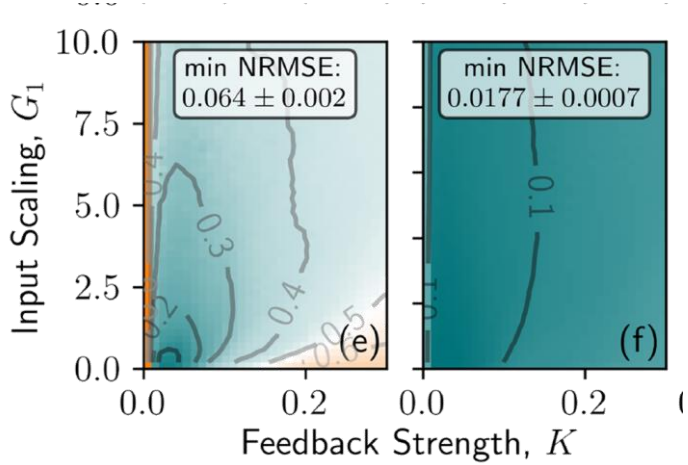
Memory – nonlinearity trade off in QRC which can be controlled by the reset length



S. Čindrak, B. Donvil, K. Lüdge, **L.C. Jaurigue**, *Phys. Rev. Research* **6** (1): 013051 (2024).

The memory of a reservoir can be augmented on the input and output layers

→ reduced hyperparameter sensitivity
→ improved performance



L.C. Jaurigue, K. Lüdge, *Neuromorph. Comput. Eng.* **4**, 014001 (2024).

Echo State Network

Internal coupling matrix (adjacency matrix)

Vector of node states $x_i(k)$

Input sequence

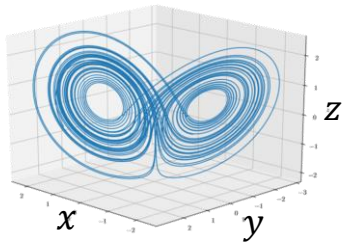
$$x(k+1) = \tanh(\boldsymbol{W_{int}} x(k) + \boldsymbol{w_{in}} I(k))$$

Nonlinear activation function

Input weights vector

Lorenz system:

$$\frac{dx}{dt} = c_1 y - c_1 x, \quad \frac{dy}{dt} = x(c_2 - z) - y, \quad \text{and} \quad \frac{dz}{dt} = xy - c_3 z$$



Sampling step size: $\Delta t = 0.1$

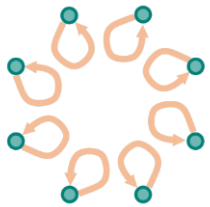
Reservoir is trained for one step ahead prediction:

$X(t) \rightarrow X(t + \Delta t)$
$Y(t) \rightarrow Y(t + \Delta t)$
$Z(t) \rightarrow Z(t + \Delta t)$

Can we reconstruct a chaotic attractor using a simple coupling topology and few nodes?

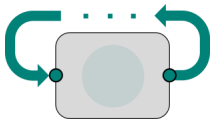
Internal coupling:

$\boldsymbol{W_{int}}$ → Uncoupled

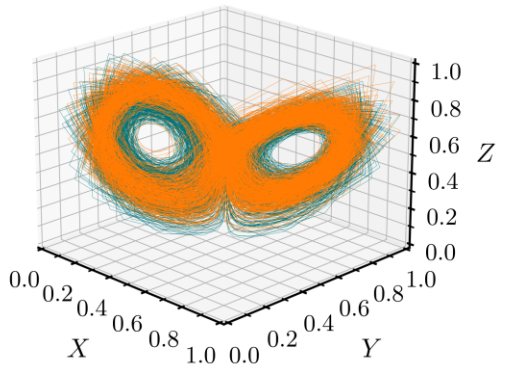


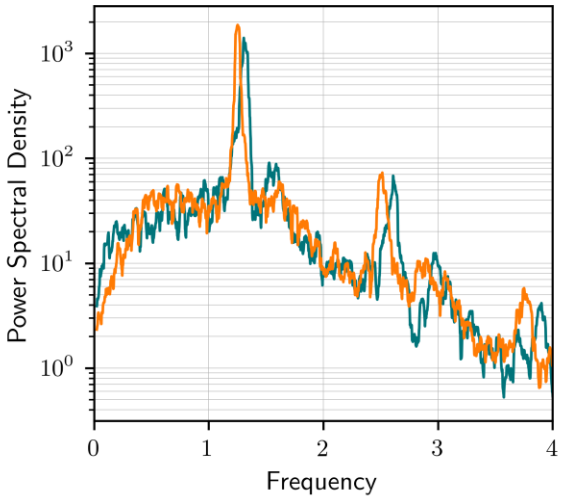
- Easier to implement experimentally
- Less resources required

20 nodes
(typically 100s-1000s are used)



Trained trajectory — True trajectory

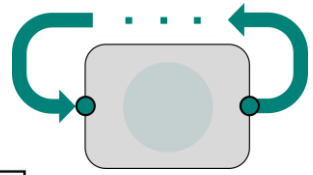


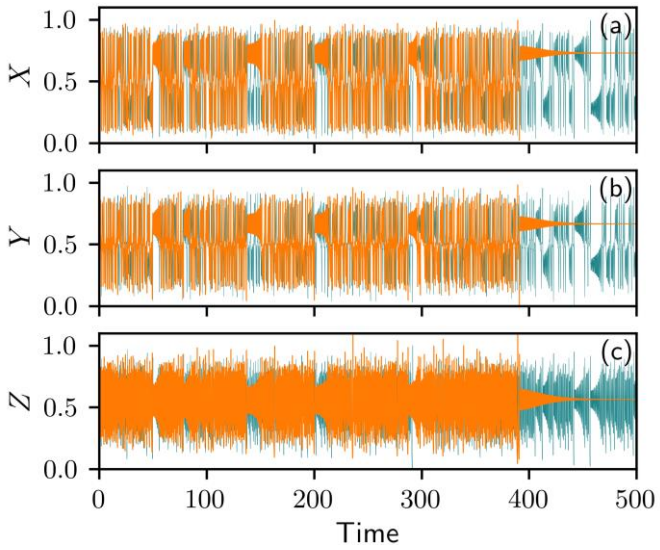


📖 **L.C. Jaurigue**, *Mach. Learn.: Sci. Technol.* **5**(3), 035058 (2024).

Dynamics of the closed system:

- Diverging (large excursions in phase space)
- Oscillatory
- Steady state





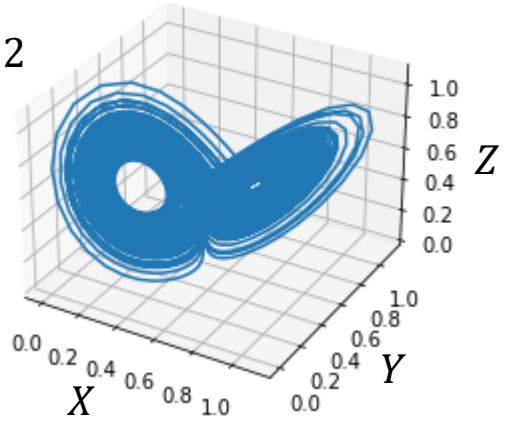
Dynamics depend on:

- Random input weights
- Starting point of the prediction phase

Can we learn more by analysing the dynamics of the trained systems?

Further reduction network size → 10 nodes

Task: Lorenz reconstruction with $\Delta t = 0.02$



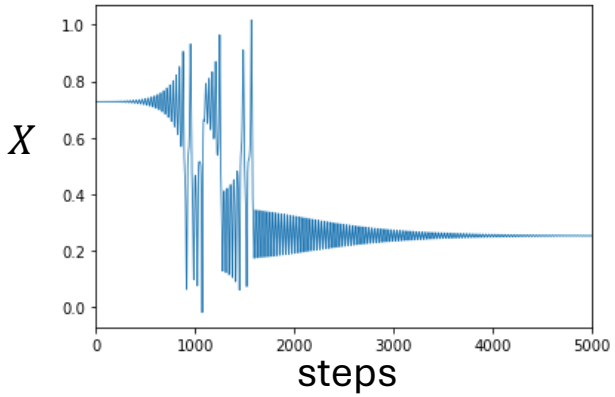
Correlation dimension:

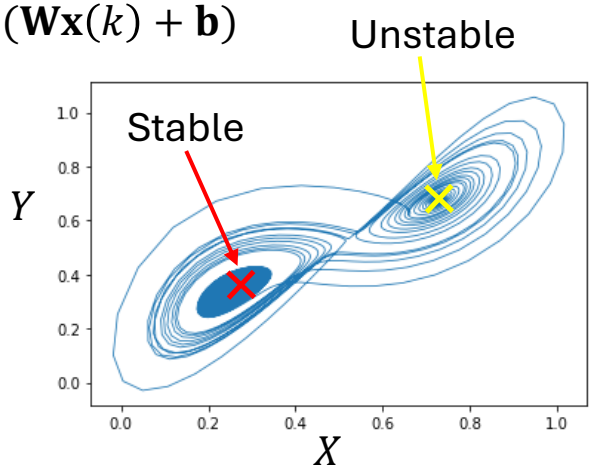
Trained system → ≈2.02

True Lorenz attractor → 2.05±0.01[1]

Fixed point stability analysis:

$$\mathbf{x}(k+1) = \tanh(\mathbf{W}\mathbf{x}(k) + \mathbf{b})$$





[1] P. Grassberger, I. Procaccia, *Physica D: Nonlinear Phenomena* **9** (1–2), 189-208 (1983).