

---

---

ECML-PKDD 2024 tutorial on

# Sustainable Deep Learning for Time-series Reservoir Computing

---

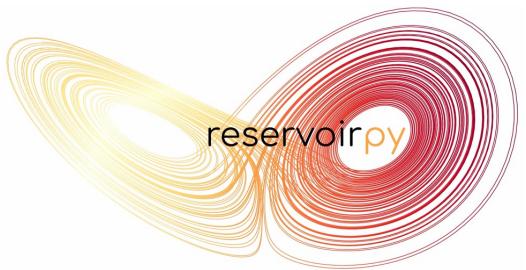
---

— A. Ceni, P.F. Dominey , C. Gallicchio,  
X. Hinaut, G. Milano —



# Chapter 3: Software libraries and applications

How to create and tune reservoirs easily?



# Xavier Hinaut

Research Scientist

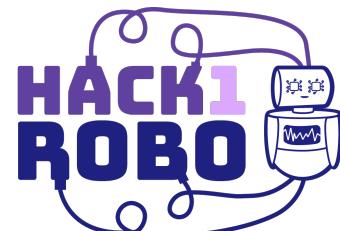
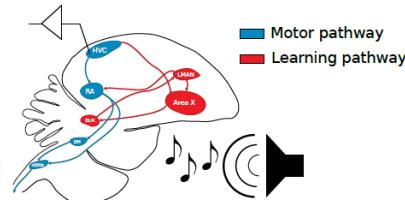
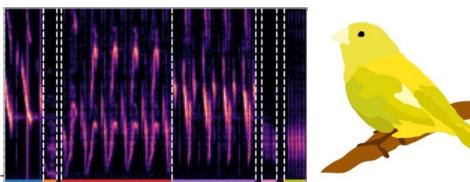
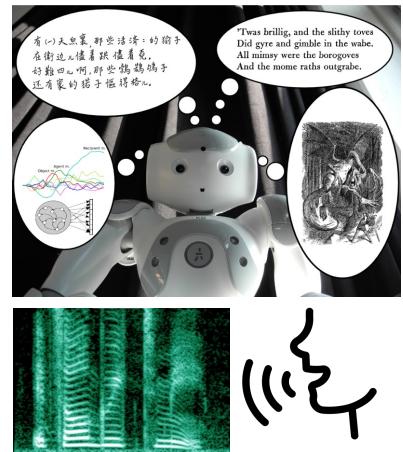
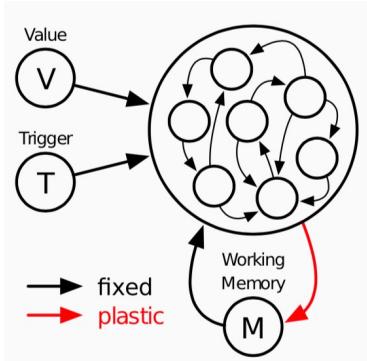
+ Paul Bernard

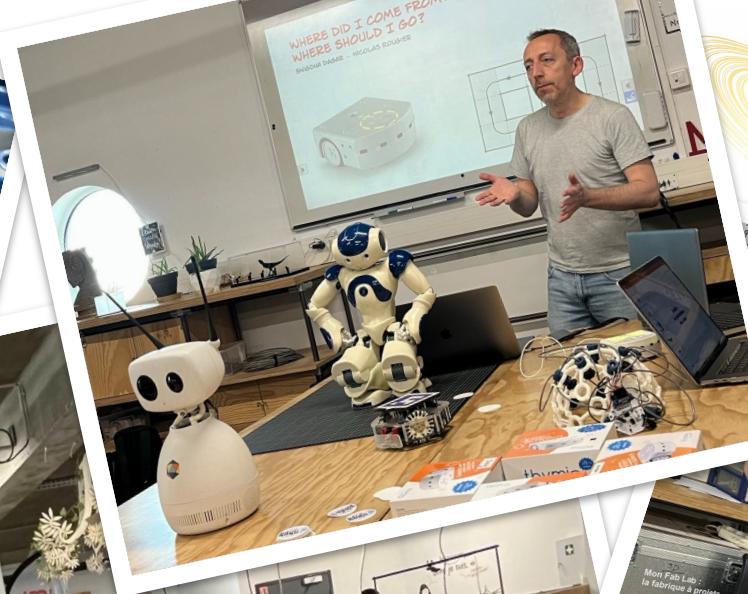
Research Engineer  
Inria, Bordeaux, France

Inria

université  
de BORDEAUX

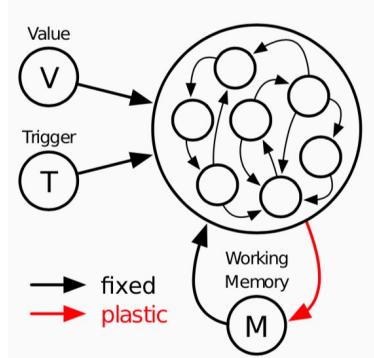
LaBRI



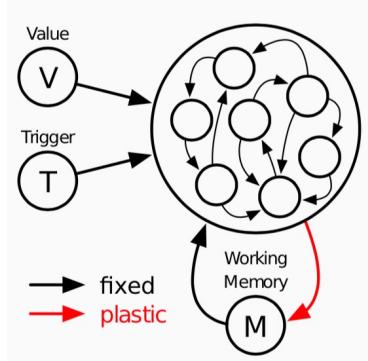


# Have you ever dreamed of being a DJ?





Alison Wonderland



```

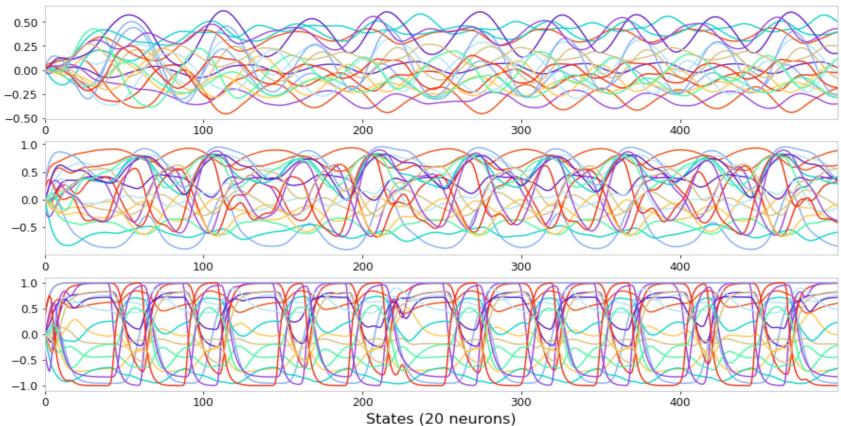
from reservoirpy.nodes import Reservoir, Ridge, Input

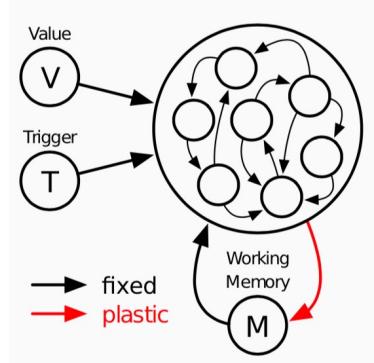
data = Input(input_dim=1)
reservoir = Reservoir(100, lr=0.3, sr=1.1)
readout = Ridge(ridge=1e-6)

esn = data >> reservoir >> readout

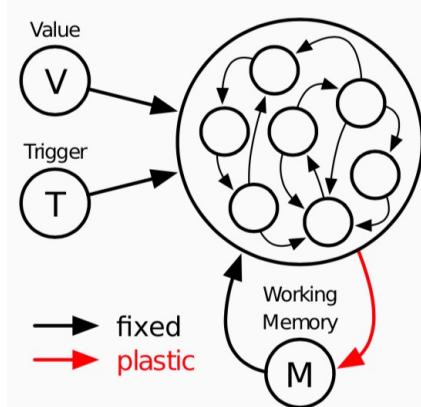
forecast = esn.fit(X, y).run(timeseries)

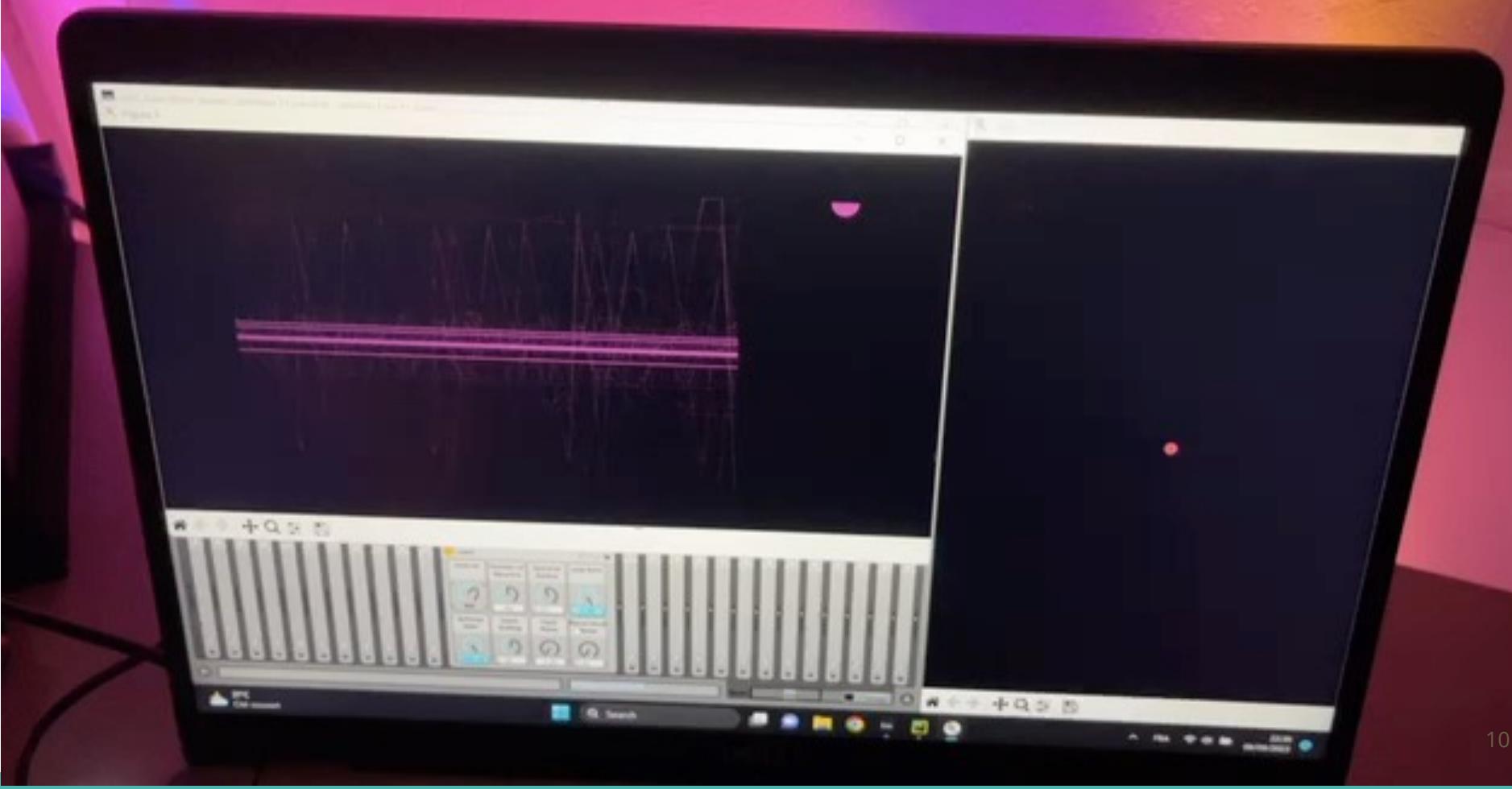
```





# Have you ever dreamed of playing with a reservoir?





# Outline

- Reservoir Computing librairies
  - ReservoirPy
  - Hands-on reservoir computing
  - Hyperparameter exploration
  - Quick demo
-

## Quick installation for ReservoirPy demo



<https://github.com/reservoirpy/reservoirpy>

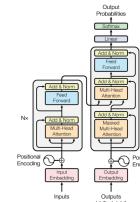
**pip install reservoirpy[hypers]**

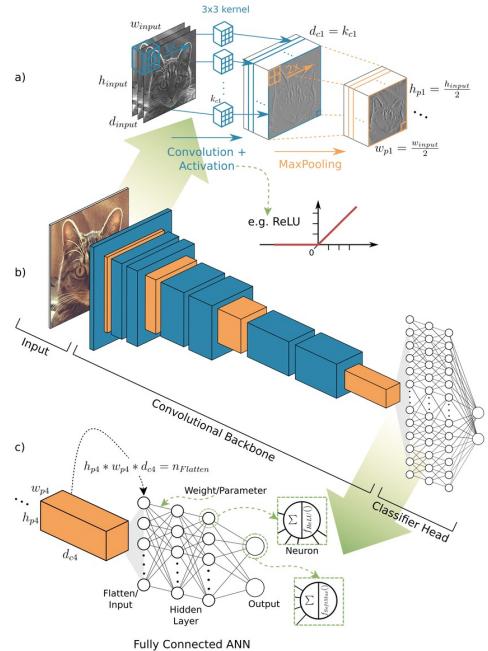
# Reservoir Computing Libraries Overview

**Why is a library important to  
strengthens a community?**

# Why a library is important to strengthens a community?

- What enabled Deep Learning community to grow so big since 2010's?
- The first well known ingredients:
  - Massive amount of data
  - Massive computations available with GPUs
- The additional ingredients:
  - Several libraries backed by big institutions / companies
  - Platforms to share datasets and models 😊 **Hugging Face**
- Scikit-learn had a similar effect on the ML community
- ➔ Enabled to build complex architectures!

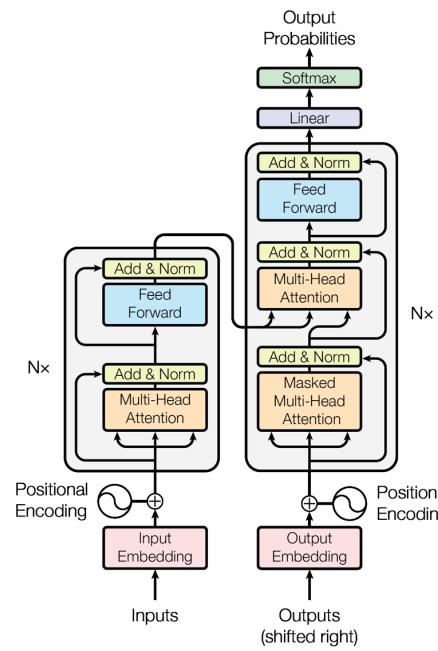




## Convolutional Neural Networks (CNNs)

### → Aims of ReservoirPy

- Help to **share and reuse models** in the RC community
- Empower to **build complex architectures** such as Transformers



## Transformers

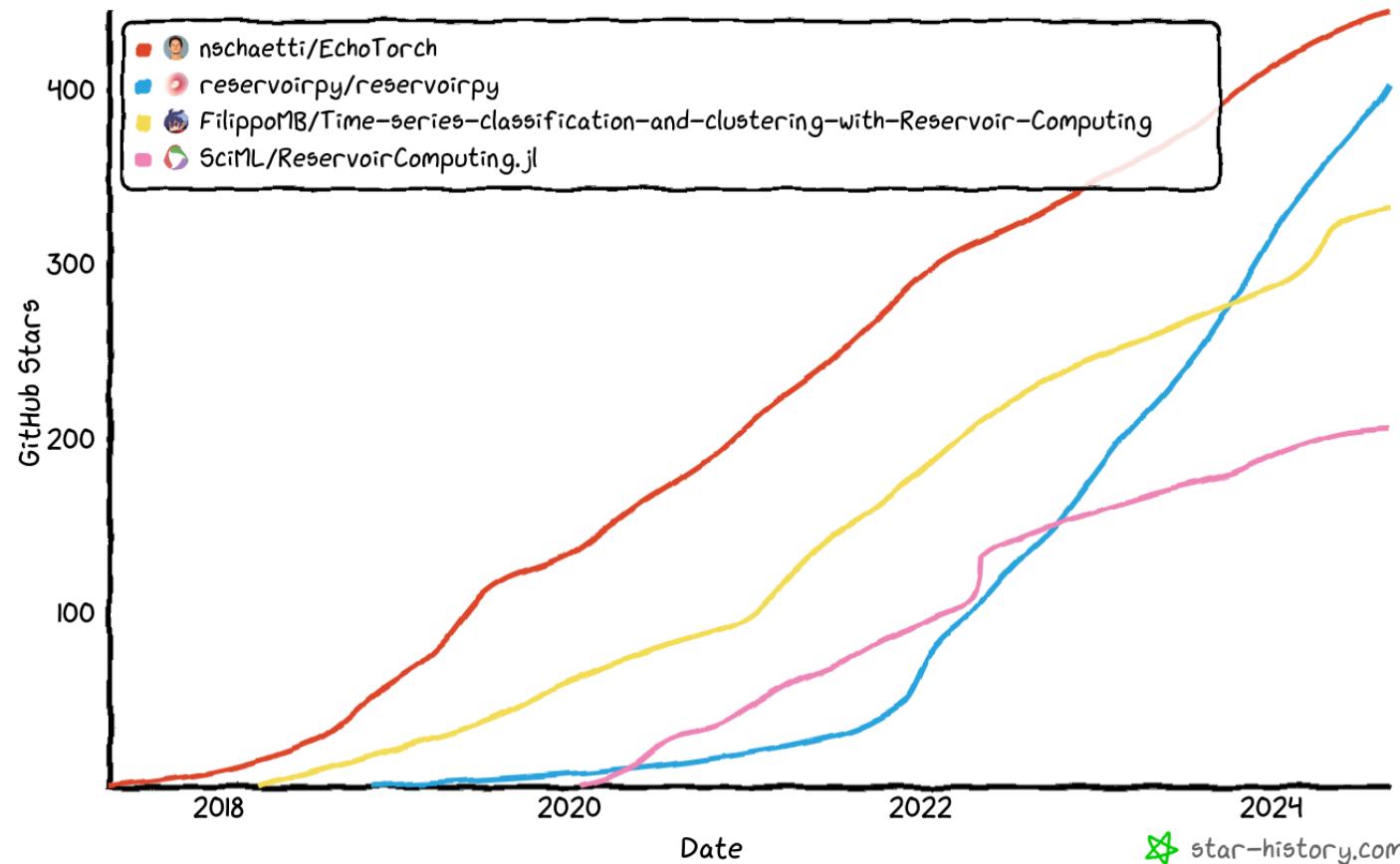


## Some Reservoir Computing Libraries

Library	Language	Main dependency	Last activity	Last release	Package	Doc.	Tests	On.	Fb.	Model type	Deep
★ PyRCN	Python 3	Scikit-learn	Jul. 2024	2024	pip	✓	✓	✗	✗	ESN	✓
EchoTorch	Python 3	PyTorch	May 2024	2021	pip	✓	✓	✗	✗	ESN, Conceptor	✓
★ Res.Comp.jl <sup>3</sup>	Julia	Julia	May 2024	2024	Pkg	✓	✓	✗	✗	ESN	✗
Pytorch-esn	Python 3	Pytorch	Feb. 2023	2021	✗	✗	✗	✗	✗	ESN	✓
DeepESN	Matlab	Matlab	Nov. 2019	2018	Matlab	✓	?	✗	✗	ESN	✓
EasyML	C#	C#	June 2023	2023	✗	~	✗	✗	✗	ESN, LSM	✗
LSM	Python 3	Nest	Nov. 2020	2020	✗	✗	✗	✗	✗	LSM	✗
Oger	Python 2	mdp	2012	2012	✗	✗	✓	✓	✓	LSM, ESN	✓
★ reservoirpy (this package)	Python 3	Numpy	Sep. 2024	2024	pip	✓	✓	✓	✓	ESN, LSM	✓

Table 1: Comparative table of some open source software for Reservoir Computing. This table might not be exhaustive. **Doc.** Complete documentation. **On.** Online learning strategies included. **Fb.** Feedback and delayed connections. **Deep** The software allows the design of complex models where basic RC elements such as reservoirs and readouts can be stacked to form so-called “deep” networks.

## Star History



# ReservoirPy

## Pure Python scientific libraries

No dependencies on frameworks (PyTorch, Tensorflow, ...)



[github.com/reservoirpy/reservoirpy](https://github.com/reservoirpy/reservoirpy)

reservoirpy Public

Edit Pins Unwatch Fork Starred

master Go to file + <> Code About

neuronalX and PAUL-BERNARD ad for reservoirchat on read... 169d9ab · 5 days ago

.github/workflows Test: Add ignored path for gh tests 3 months ago

docs Doc: partial\_fit documentation last week

examples Doc: ES2N, MSO8 on different seeds 2 months ago

reservoirpy Doc: examples for various nodes a... last week

static Moved deep-reservoir gif to static/ 3 months ago

tutorials Correction of some minor mistake... 3 months ago

README MIT license

python 3.8 | 3.9 | 3.10 | 3.11 | 3.12 pypi package 0.3.11 docs passing Testing failing

codecov 91% downloads 1.4k/month

ReservoirPy (v0.3.11) 🧠

A simple and flexible code for Reservoir Computing architectures like Echo State Networks

python machine-learning timeseries neural-network machine-learning-algorithms esn recurrent-neural-networks artificial-intelligence reservoir echo-state-networks reservoir-computing

Contributors 16

+ 2 contributors

Languages

Python 100.0%



## Project philosophy

- Python 3.8+ "scikit" tools: only based on **Python standard scientific stack**.
  - No dependency on other « heavy » frameworks not relevant for reservoirs (TensorFlow, Torch, ...)
- Create **complex models** with **simple building blocks**.
- Reach **high computational efficiency**.
- **Community driven developement** (tests, documentation, tutorials...).

## Targeted users:

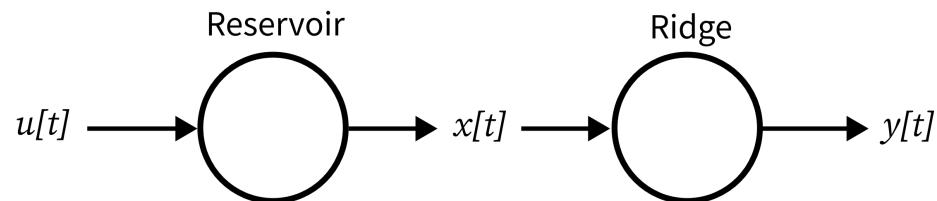
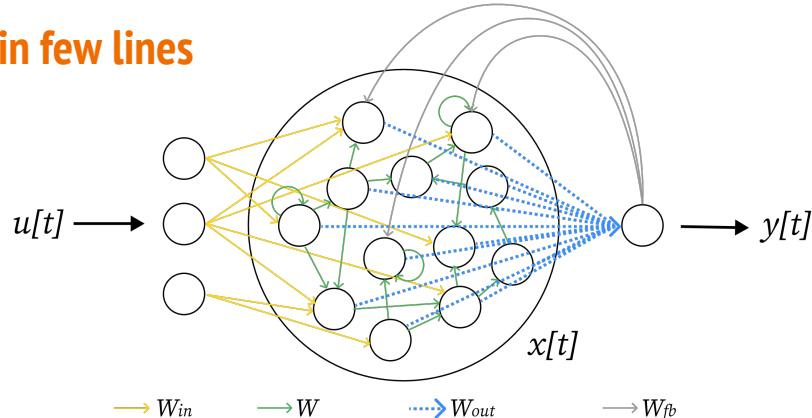
- ➔ Academic researchers
- ➔ **Industry**
- ➔ *Machine learning students/beginners*

# ReservoirPy

## ReservoirPy in few lines

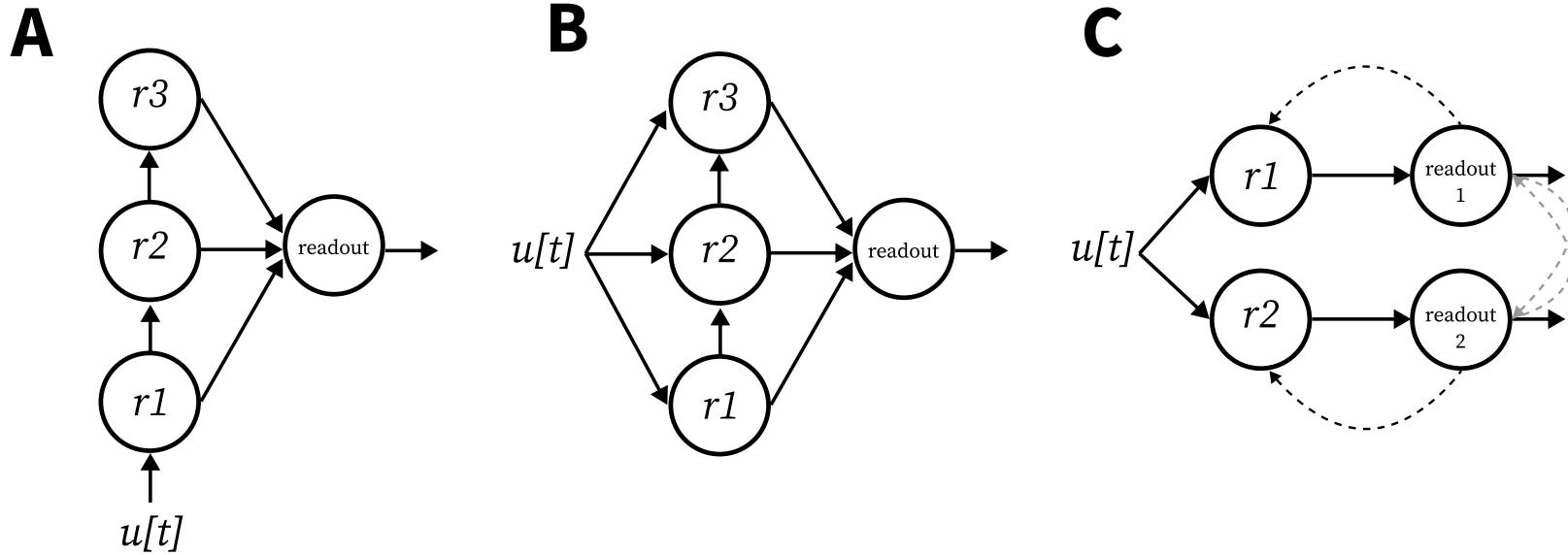
### Node creation

A *node* is an independent operator, that applies a function on some data, potentially in a recurrent way.

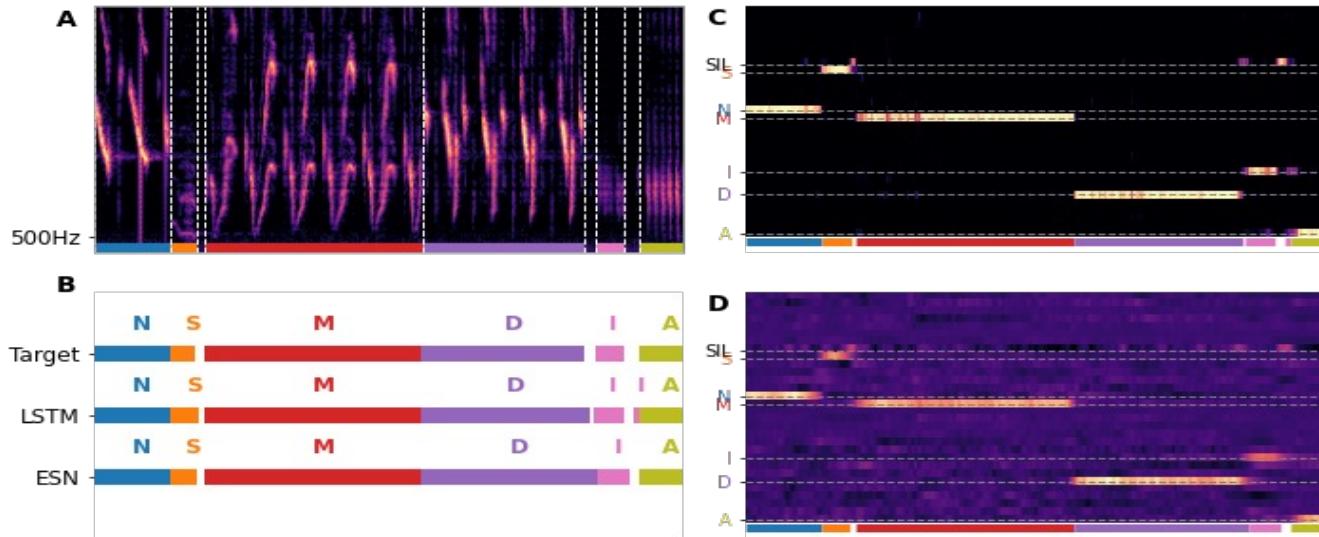


```
1 from reservoirpy.nodes import Reservoir, Ridge  
2  
3 reservoir = Reservoir(100, lr=0.3, sr=1.25, input_scaling=0.1)  
4 readout = Ridge(ridge=1e-5)
```

## Enable complex architectures



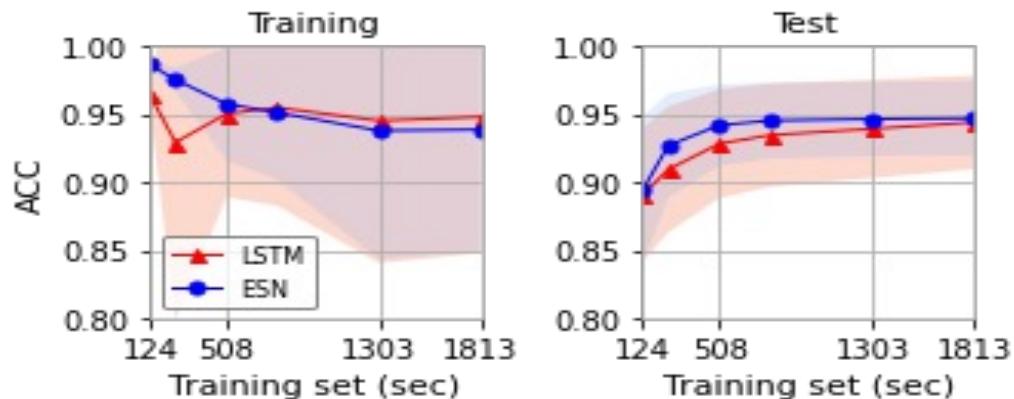
## Automatic canary songs annotation



- 459 songs of a canary individual
- 40 different song motifs in the songs.

## Automatic canary songs annotation

Score: **95% (accuracy)**  
Benchmarked against LSTMs



### Performances

Benchmark on CPU with  
**distributed** computation

Model	LSTM	ESN
Average training time (s)	$2930 \pm 222$	<b><math>35 \pm 1</math></b>

# Hands-on ReservoirPy

# Follow with the Notebook



[github.com/reservoirpy/workshops/...](https://github.com/reservoirpy/workshops/)

The screenshot shows a GitHub repository interface for 'reservoirpy / workshops'. The 'Code' tab is selected. Below it, a file named 'workshops / ECML-PKDD-2024 / Introduction.ipynb' is listed. The notebook was created by 'PAUL-BERNARD' and updated '2 days ago'. It contains 556 lines (556 loc) and is 341 KB in size. The notebook's content starts with a title and imports for numpy, matplotlib, and reservoirpy.

```
# Imports
import numpy as np
import matplotlib.pyplot as plt

import reservoirpy as rpy
from reservoirpy.nodes import Reservoir, Ridge

rpy.verbosity(0)
# Make everything deterministic
rpy.set_seed(2341)
```

## Data format

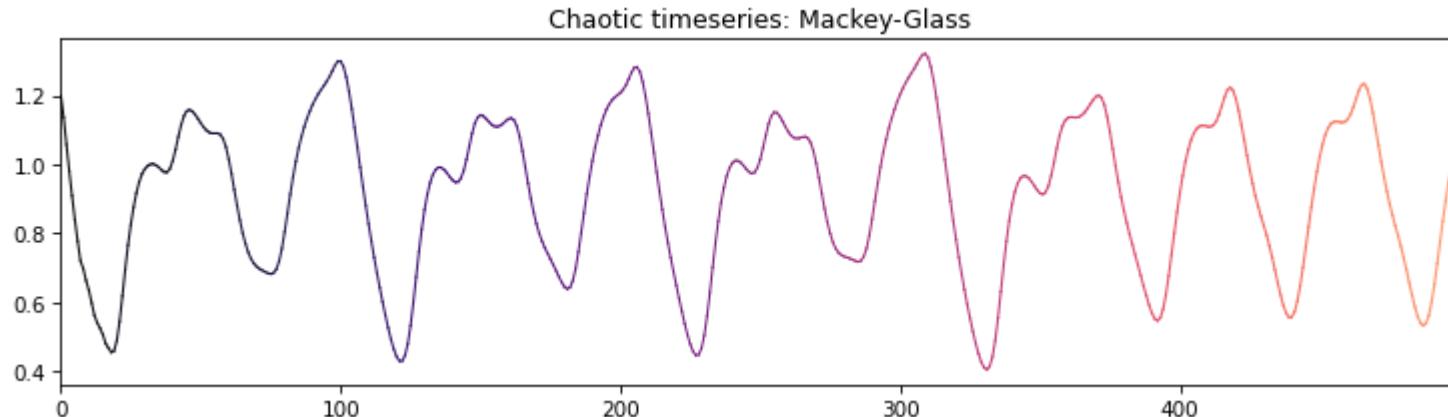
Everything is a numpy array.  
shape = (timesteps, features)

```
from reservoirpy.datasets import mackey_glass  
  
timeseries = mackey_glass(n_timesteps=2_000)  
timeseries.shape
```

✓ 0.0s

Python

(2000, 1)

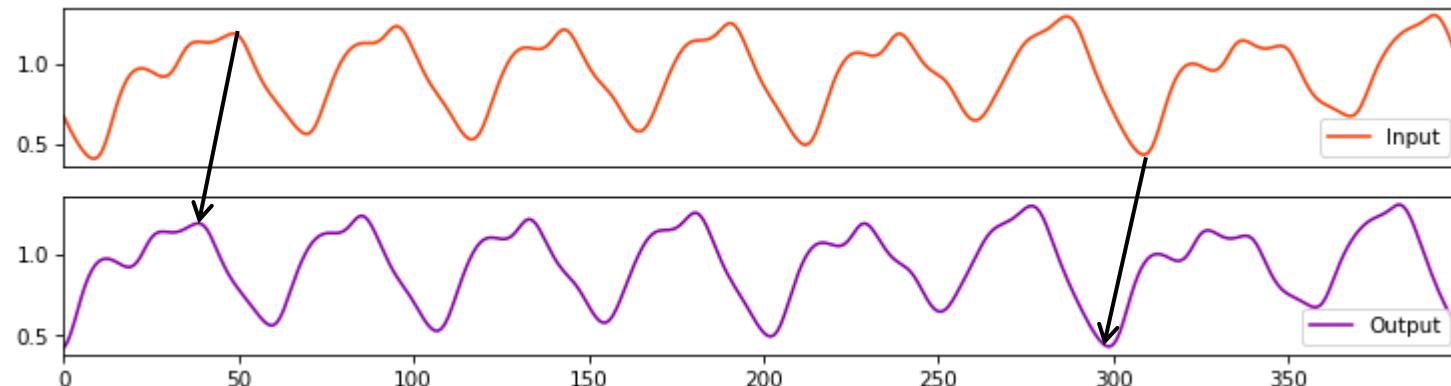


## Data pre-processing

```
from reservoirpy.datasets import to_forecasting  
  
dataset = to_forecasting(X, forecast=10, test_size=0.2)  
x_train, x_test, y_train, y_test = dataset
```

✓ 0.0s

Python

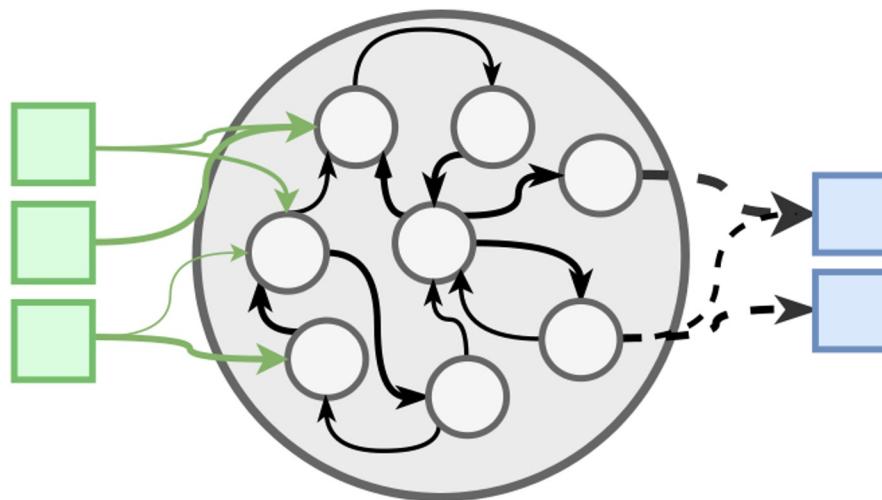


# Echo State Network creation

```
model = ESN(  
    units=100, # neurons inside the reservoir  
    lr=0.3, # leak rate  
    sr=1.25, # spectral radius of the weight matrix  
    ridge=1e-8) # Regularization parameter
```

✓ 0.0s

Python

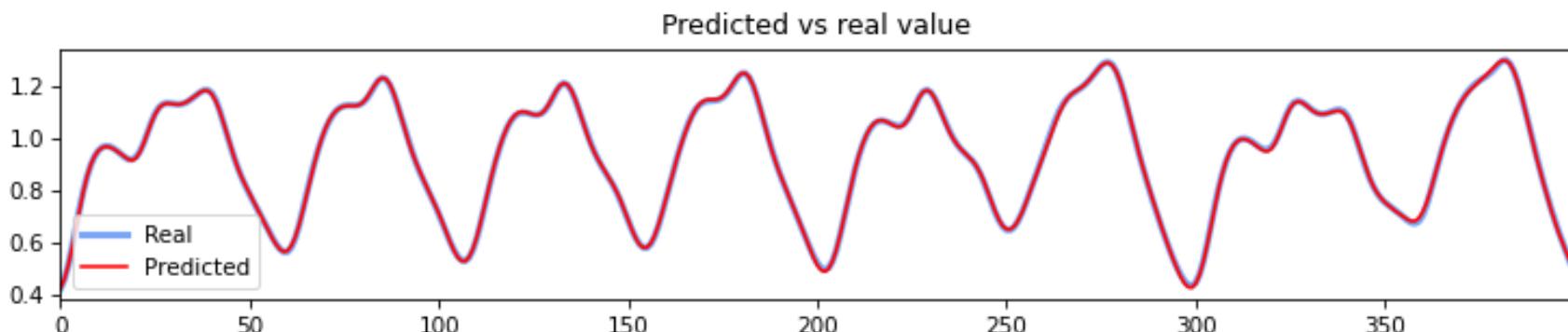


## Fit and run the model

```
model.fit(x_train, y_train) # Learning phase  
y_pred = model.run(x_test) # Prediction
```

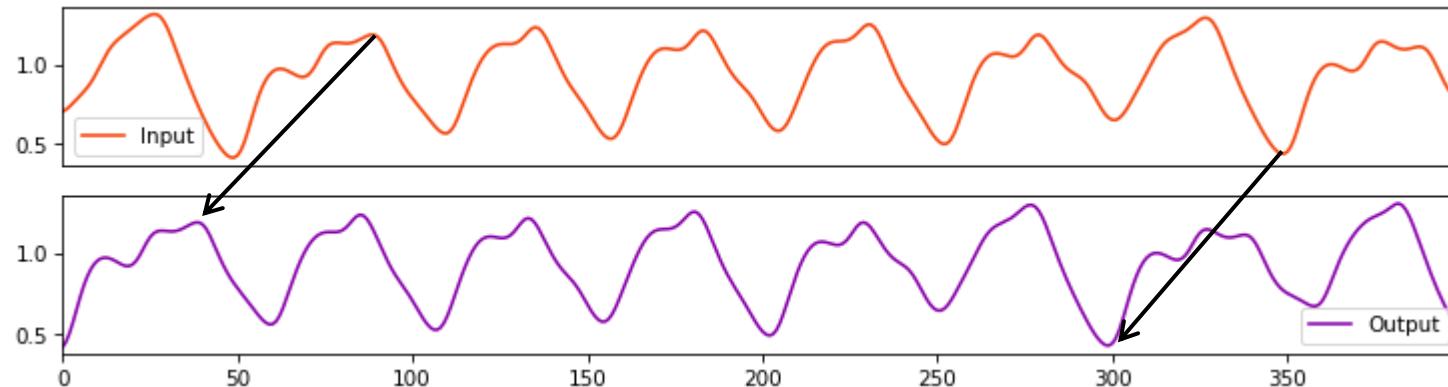
✓ 0.5s

Python



## ... what about a t+50 prediction?

```
dataset = to_forecasting(X, forecast=50, test_size=0.2)  
x_train, x_test, y_train, y_test = dataset
```



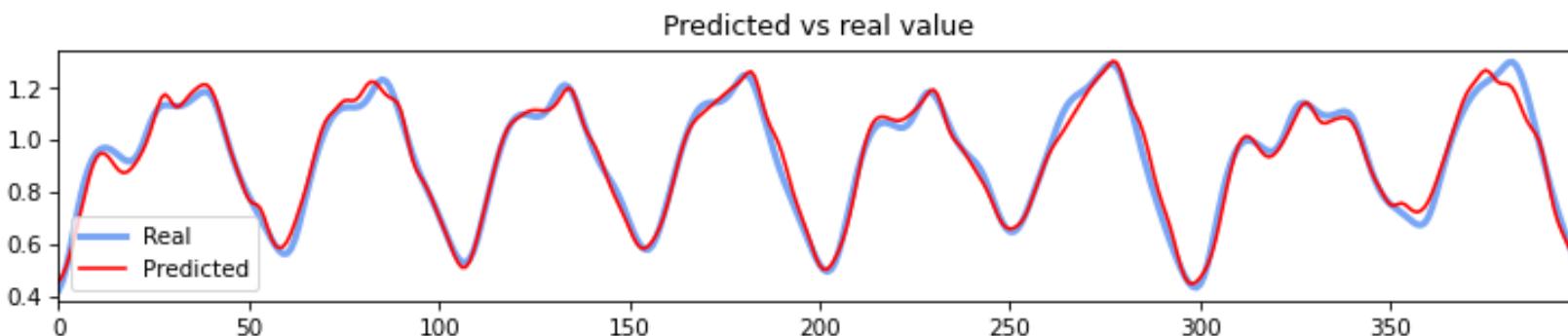
# ... what about a t+50 prediction?

## The result

```
# Fitting the model  
model.fit(x_train, y_train)  
# Testing the model  
y_pred = model.run(x_test)
```

✓ 0.4s

Python



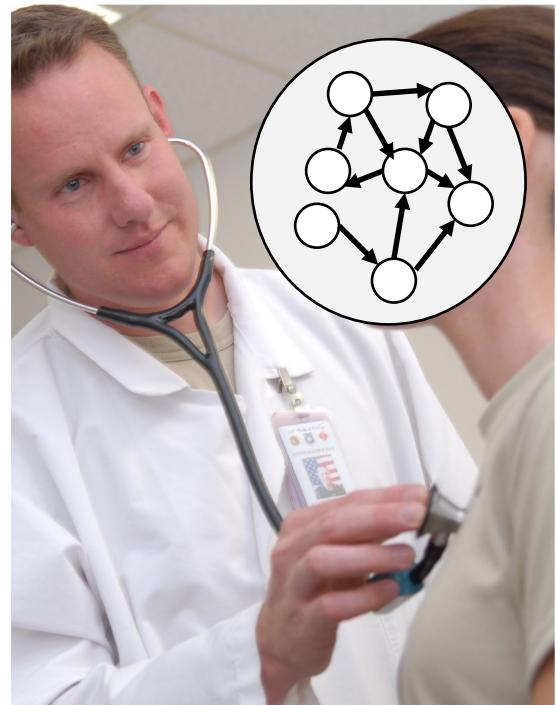
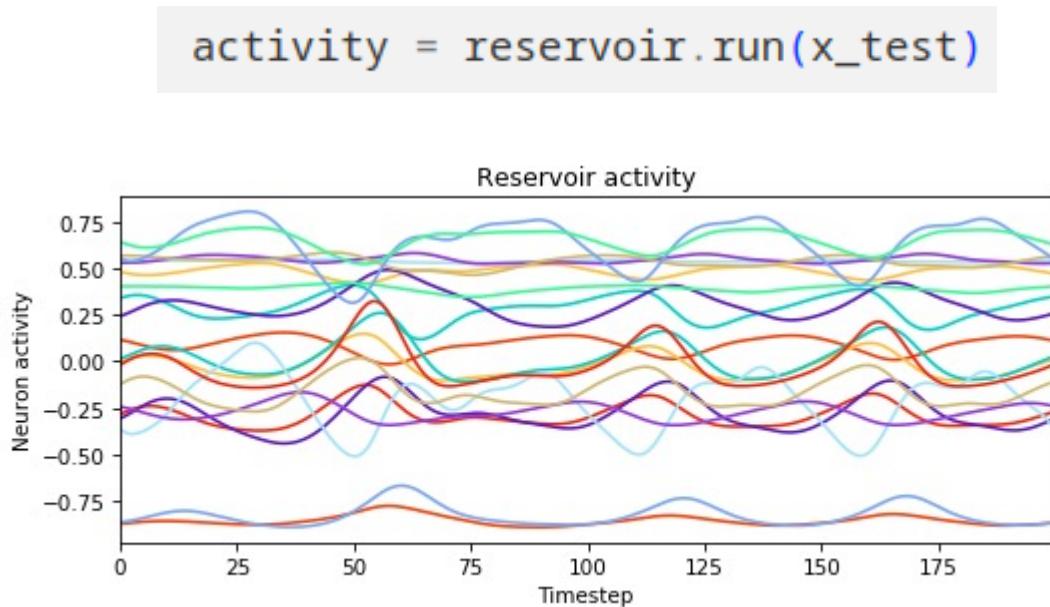
# Summary

```
model = ESN(  
    units=100, # neurons inside the reservoir  
    lr=0.3, # leak rate  
    sr=1.25, # spectral radius of the weight matrix  
    ridge=1e-8, # Regularization parameter  
)  
  
# Fitting the model  
model.fit(x_train, y_train)  
# Testing the model  
y_pred = model.run(x_test)
```

✓ 0.3s

Python

# What is happening inside the reservoir?



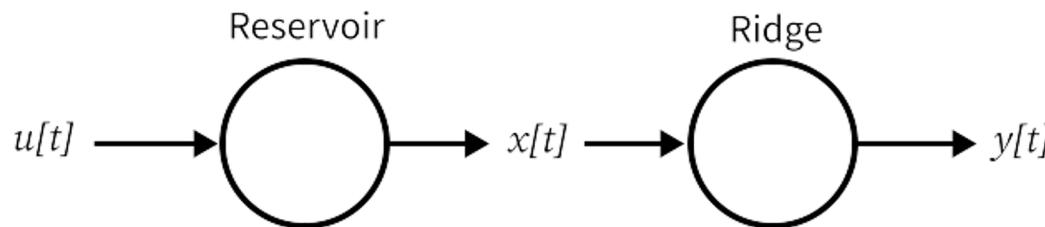
*Image credits : Wikimedia*

# The nodes

```
from reservoirpy.nodes import Reservoir, Ridge  
  
reservoir = Reservoir(units=10)  
readout = Ridge(ridge=1e-4)  
  
model = reservoir >> readout
```

✓ 0.0s

Python



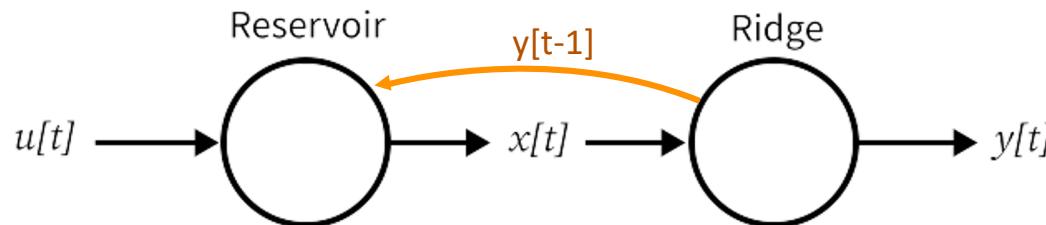
## Feedback connections

```
reservoir = Reservoir(units=10)
readout = Ridge(ridge=1e-4)

reservoir <=> readout # create feedback
model = reservoir >> readout
```

✓ 0.0s

Python



# The scikit-learn node

```
from reservoirpy.nodes import ScikitLearnNode
from sklearn.linear_model import (
    RidgeClassifier,
    LogisticRegression,
)
reservoir = Reservoir(units=10)

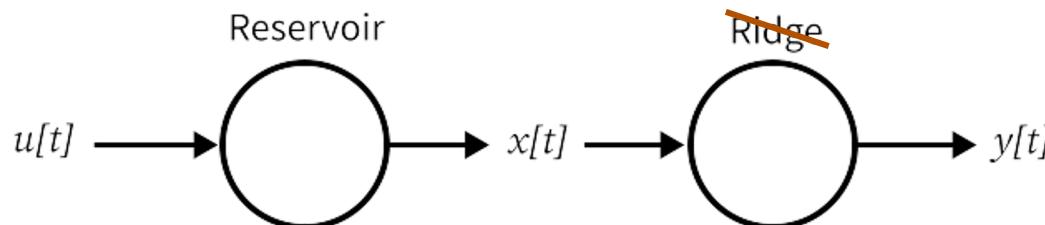
classifier = ScikitLearnNode(RidgeClassifier)
logistic_regressor = ScikitLearnNode(LogisticRegression)

model1 = reservoir >> classifier
model2 = reservoir >> logistic_regressor
```

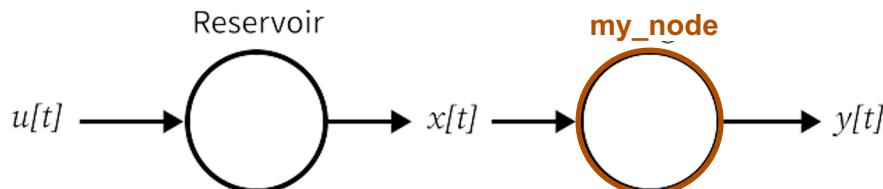
✓ 0.0s

Python

Any scikit-learn model



# Create custom node



```
from reservoir.node import Node

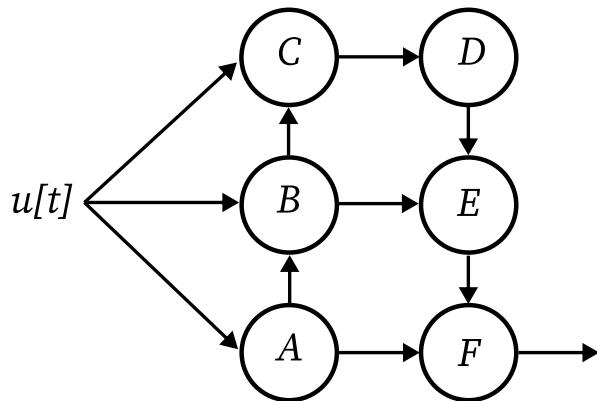
def initialize(node, x, y):
    ...

def forward(node, x):
    ...

my_node = Node(
    initializer=initialize,
    forward=forward,
    params={"param1": None},
    hypers={"const2": -1},
)
```

[https://reservoirpy.readthedocs.io/en/latest/user\\_guide/create\\_new\\_node.html](https://reservoirpy.readthedocs.io/en/latest/user_guide/create_new_node.html)

## Enable complex architectures



```
from reservoirpy.nodes import Input, Output  
  
A, B, C, D, E, F = (Node() for _ in range(6))  
  
path1, path2 = A >> F, B >> E  
# One-to-many connection using a list  
path3 = Input() >> [A, B, C]  
# Chain of connections  
path4 = A >> B >> C >> D >> E >> F >> Output()  
# Merge all pathways to create the full computational graph  
model = path1 & path2 & path3 & path4
```

# Tools for Hyperparameter (HP) optimization

# Which Hyperparameters (HP)?

- Important HP to keep in mind
  - number of neurons
  - input scaling
  - spectral radius
  - (Feedback scaling)
  - Leak-rate (= 1 / time constant)
  - Ridge (regularization)
  - Warm-up (= « transitory phase »)

➡ See Tutorial 4 on GitHub

[https://github.com/reservoirpy/reservoirpy/blob/master/tutorials/4-Understand\\_and\\_optimize\\_hyperparameters.ipynb](https://github.com/reservoirpy/reservoirpy/blob/master/tutorials/4-Understand_and_optimize_hyperparameters.ipynb)

reservoirpy / tutorials / 4-Understand\_and\_optimize\_hyperparameters.ipynb □



VirgileBoraud and PAUL-BERNARD Correction of some minor mistakes on the tutorials (#161) □

78b89ab · 2 we

2.29 MB

## Understand and optimize ESN hyperparameters

This chapter gives very basic clues on how to interpret the action of some first importance hyperparameters of an Echo state networks.

It also present a very basic example of optimization using [hyperopt](#) and [reservoirpy.hyper](#) tools.

### Understand ESN hyperparameters

In [1]:

```
UNITS = 100          # - number of neurons
LEAK_RATE = 0.3      # - leaking rate
SPECTRAL_RADIUS = 1.25 # - spectral radius of W
INPUT_SCALING = 1.0    # - input scaling
RC_CONNECTIVITY = 0.1   # - density of reservoir internal matrix
INPUT_CONNECTIVITY = 0.2 # and of reservoir input matrix
REGULARIZATION = 1e-8    # - regularization coefficient for ridge regression
SEED = 1234           # for reproducibility
```

# What is the spectral radius?



# What is the spectral radius?

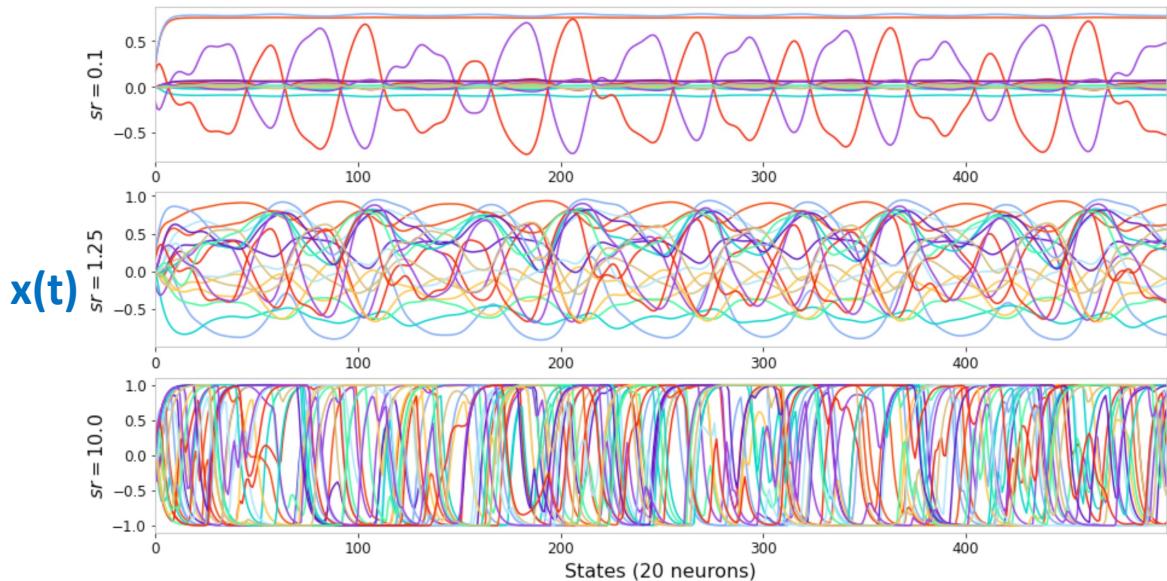


# What is the spectral radius?



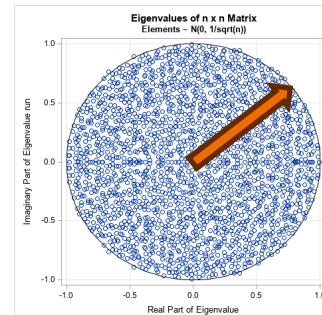
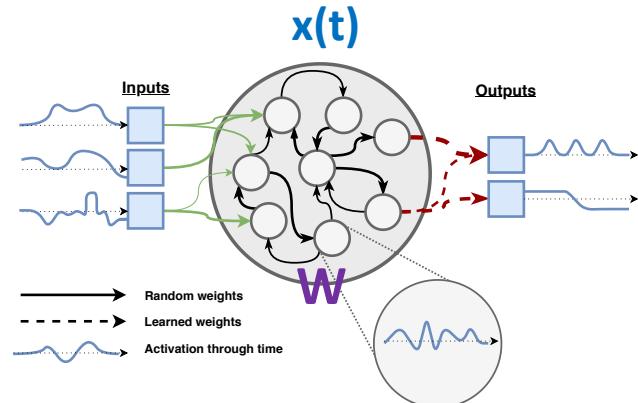
# Spectral Radius

(= recurrent connections scaling)



The spectral radius is defined as the maximum eigenvalue of the reservoir matrix.

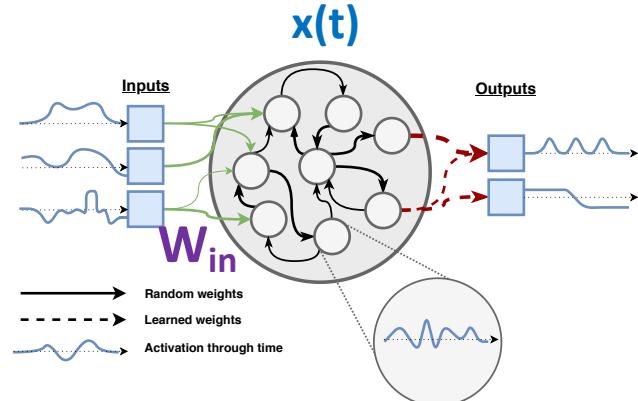
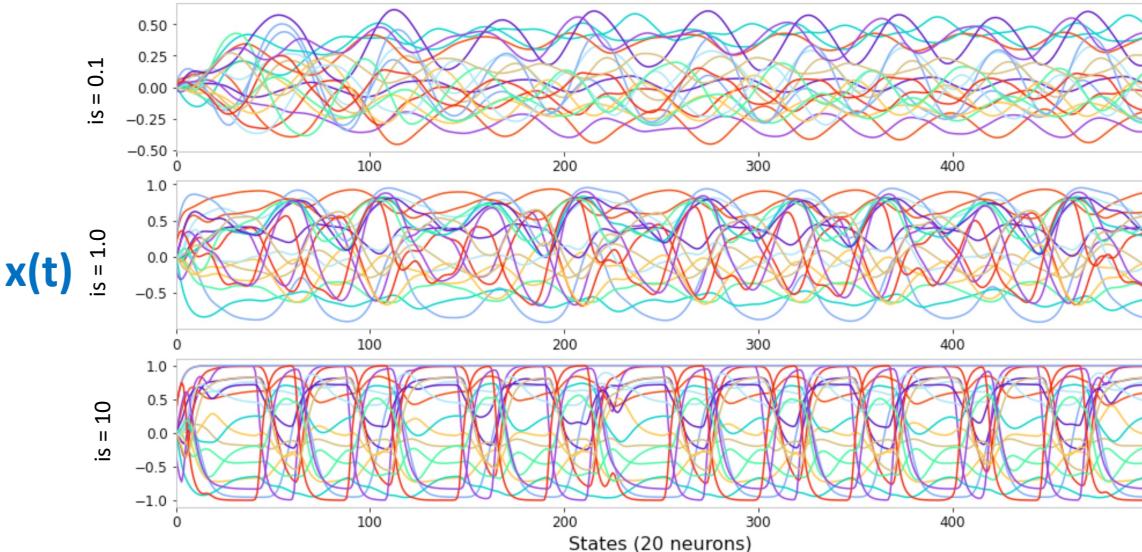
- spectral radius → **stable** dynamics
- + spectral radius → **chaotic** dynamics



Spectral radius  
of matrix  $W$

# Input Scaling

= scaling of input connections



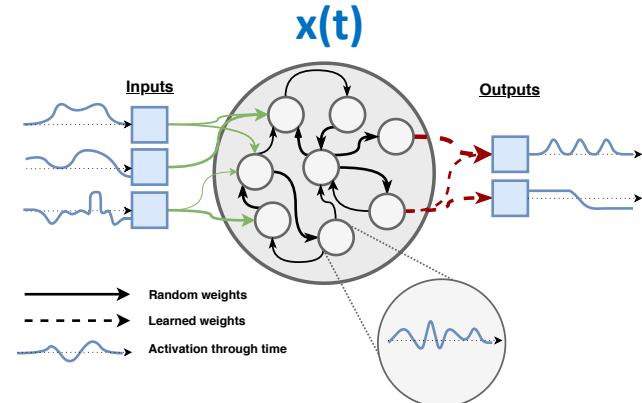
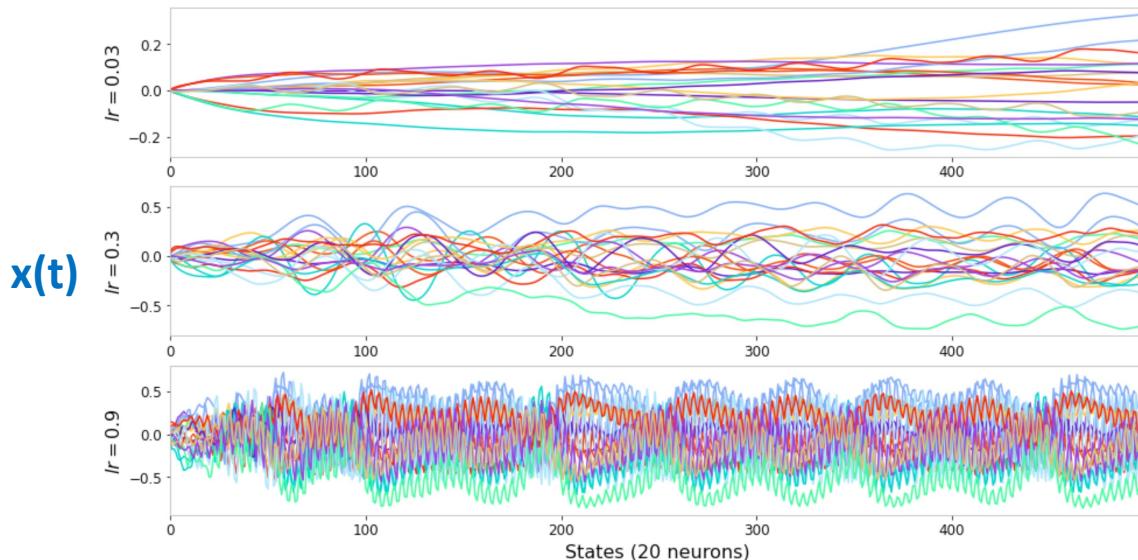
The input scaling controls how the ESN interact with the inputs. It is a coefficient applied to the input matrix  $W_{in}$ .

- + input scaling → **input-driven** activities
- - input scaling → **free** activities

The input scaling can also be used to rescale the inputs and adjust their influences.

# Leak-rate (« leaking rate »)

= inverse of time constant



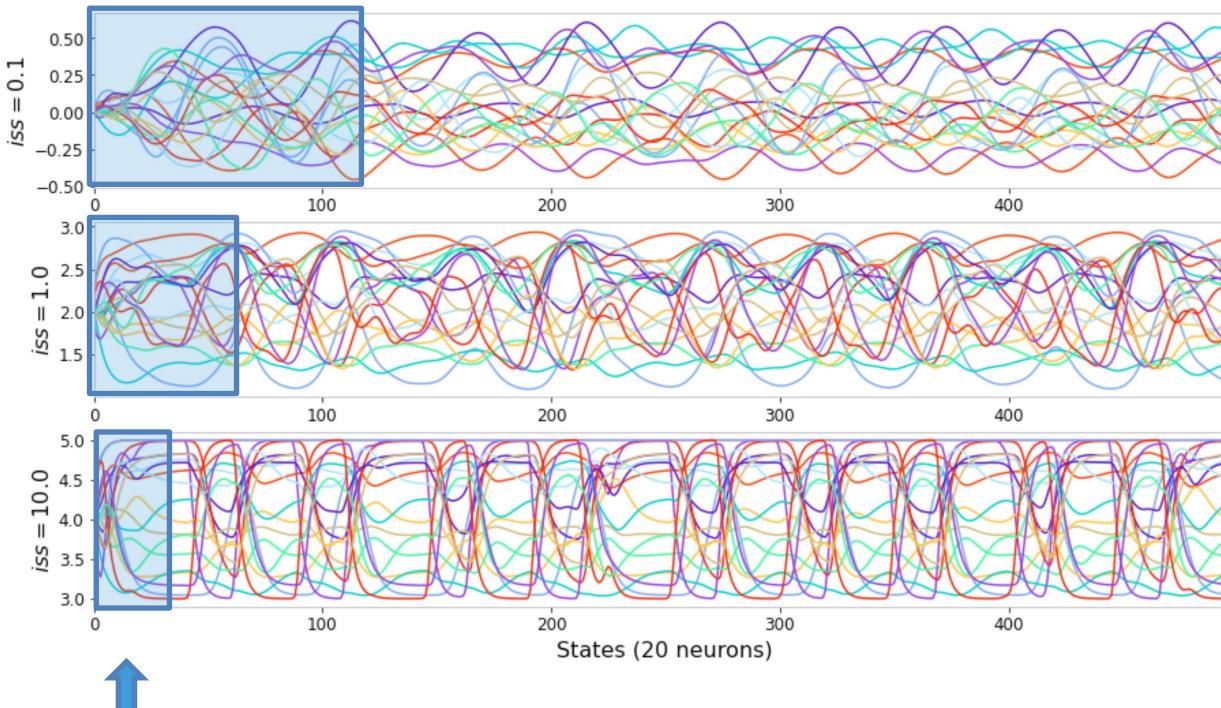
The leaking rate ( $\alpha$ ) controls the "memory feedback" of the ESN. The ESN states are indeed computed as:

$$s(t+1) = \underbrace{(1 - \alpha)s(t)}_{\text{previous states}} + \underbrace{\alpha f(u(t+1), s(t))}_{\text{new states}}$$

- + leaking rate → **low inertia**, little memory of previous states
- - leaking rate → **high inertia**, big memory of previous states

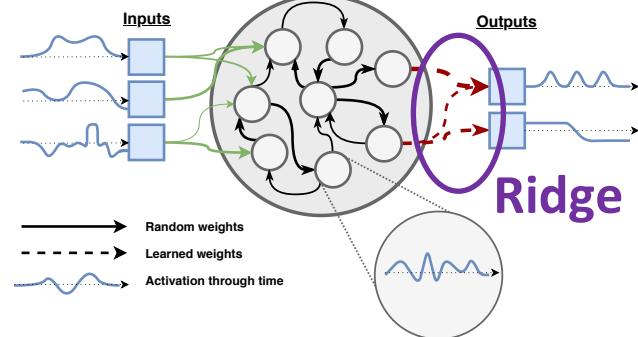
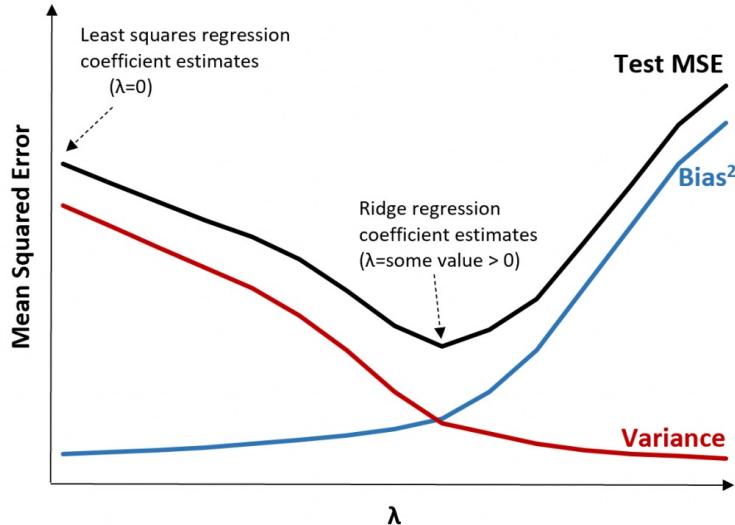
# Warm-up (« warming-up »)

= transitory period during the « heating » of the system



# Ridge

= regularisation of linear regression



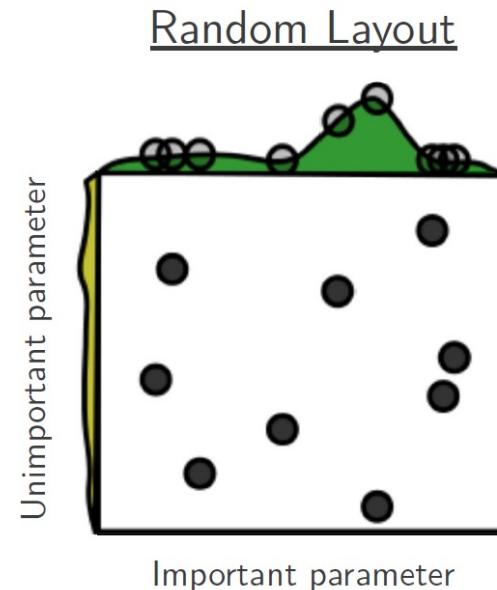
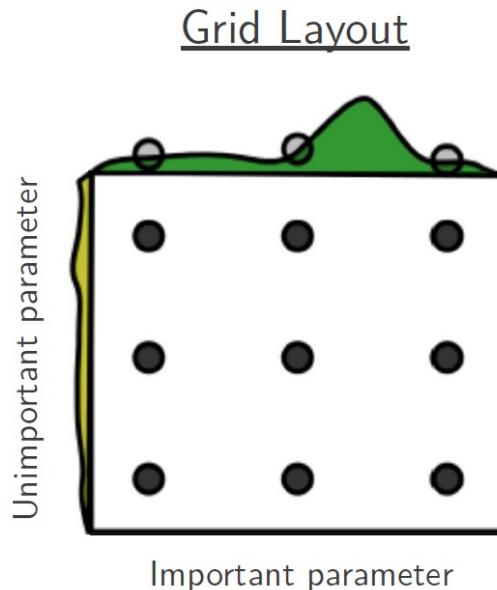
# Let's make hyperparameter search easy and reproducible!



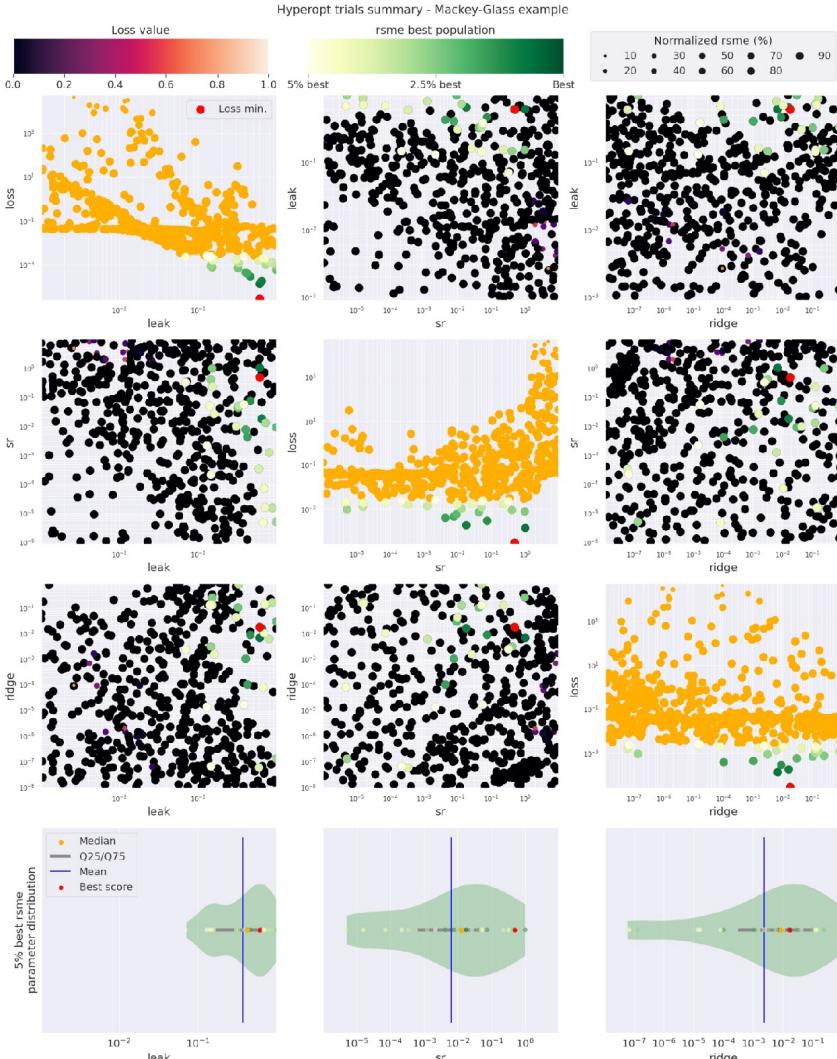
Don't try at home without proper tools!  
( = don't try with grid search!)

## Practical advices for parameter optimization

- ➡ Grid search often fails in high dimension spaces
  - « Grid search allocate **too many trials** to the exploration of **dimensions that do not matter ...**
  - ... and suffer from **poor coverage in dimensions that are important.** » (Bergstra, 2012)

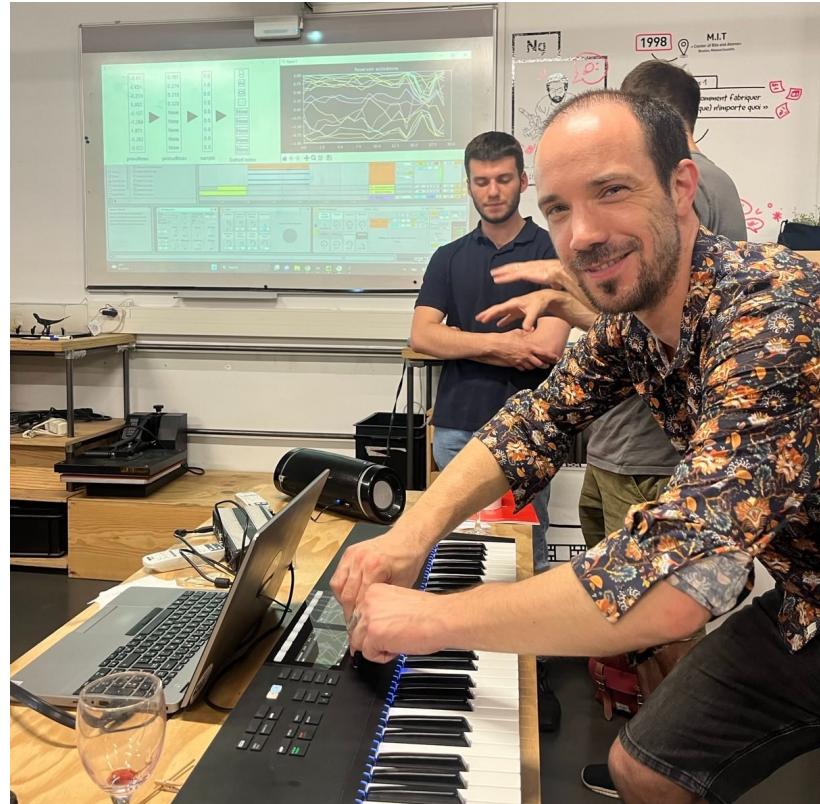
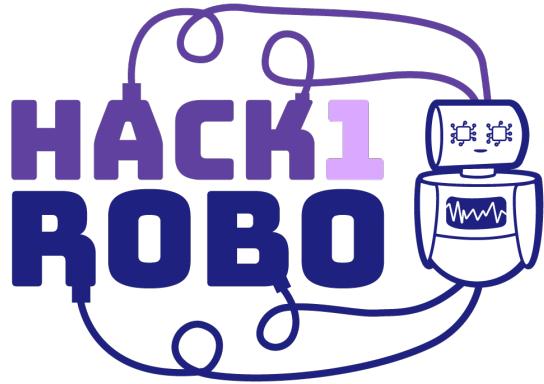


# Graphical view of hyperparameter optimization

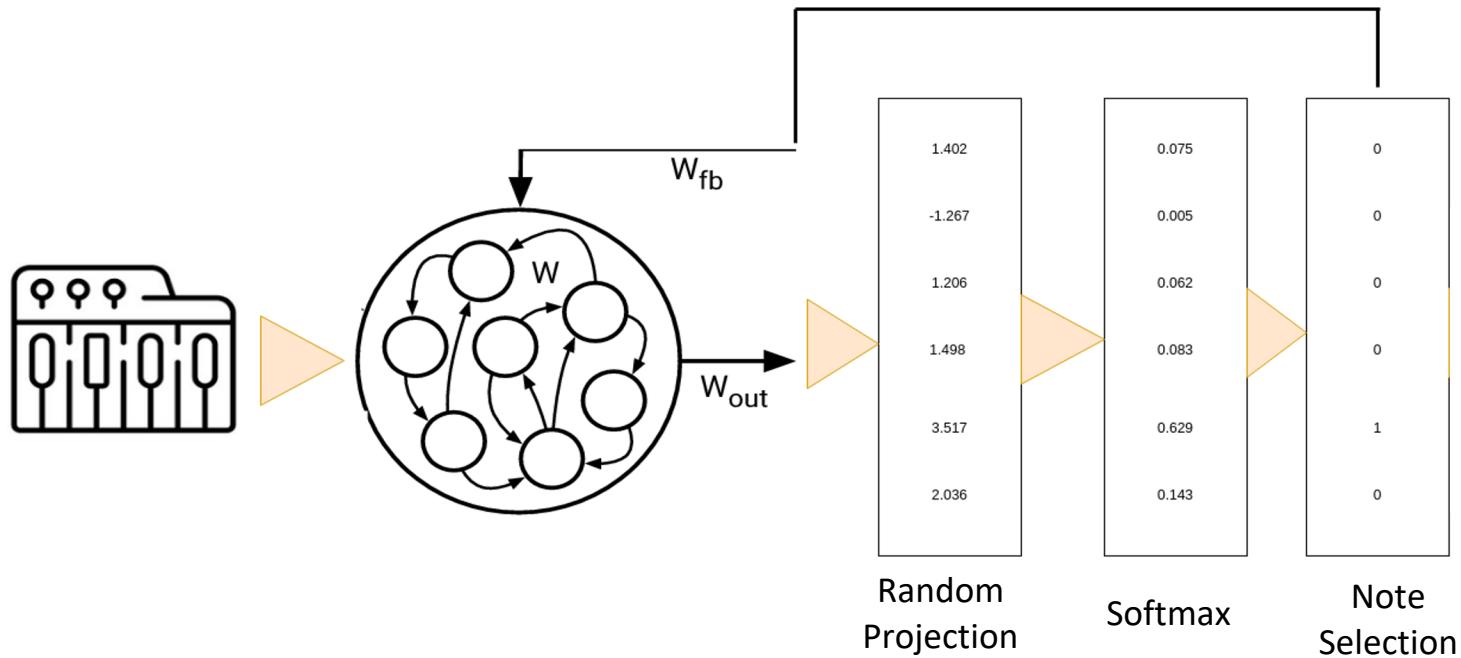


# Application examples

# “RéMi” the Reservoir MIDI



# “RéMi” the Reservoir MIDI



## Classical arpegiators

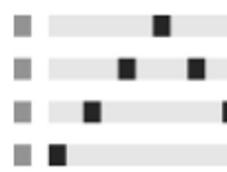
**Up**



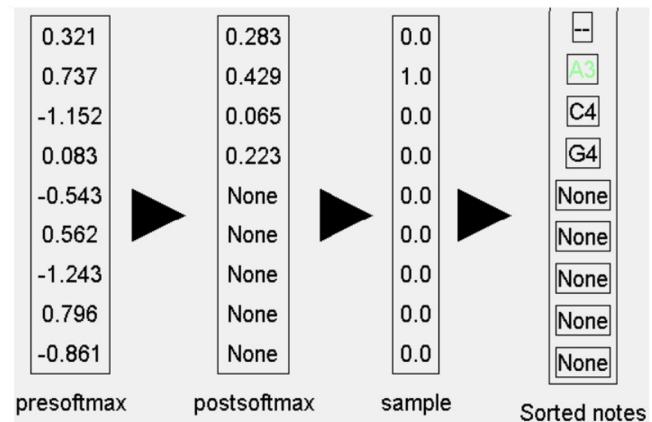
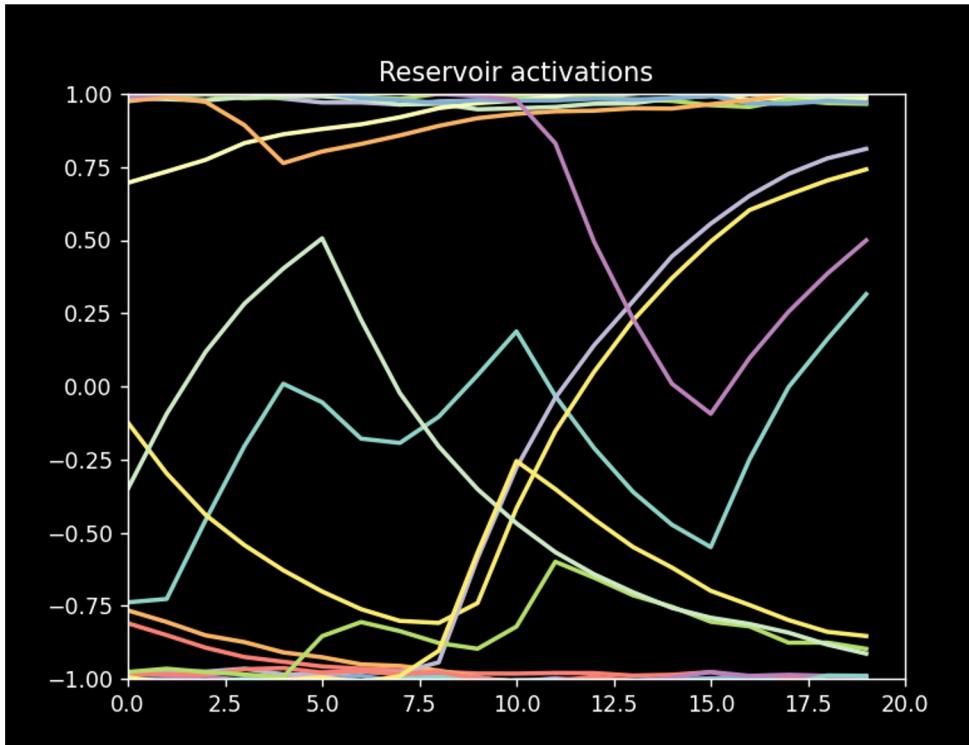
**Down**



**Up&Down**



# Interface





PLUG-IN

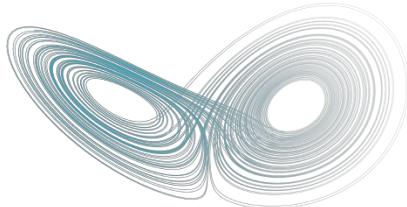
MIXER

INSTANCE

MIDI

SETUP

# ReservoirChat

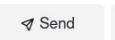


<https://chat.reservoirpy.inria.fr>

ReservoirChat  

User  What's the point of ridge regression in reservoir computing ?   
10:32

ReservoirChat  Ridge regression is used in reservoir computing to prevent overfitting and improve the generalization of the model. It does this by adding a penalty to the regression equation to reduce high-value coefficients, making the model more stable. This technique is particularly useful when dealing with multicollinearity, where two or more independent variables are highly correlated. By adding a regularization term to the loss function, ridge regression helps to mitigate the risk of overfitting and improve the model's performance on new, unseen data.  
  
In the context of reservoir computing, ridge regression is used to compute the readout map from reservoir states to the output signal. This is typically done by linear regression using the MSE loss, but ridge regression can be used instead to add a penalty to the loss function and prevent overfitting. The regularization term in ridge regression 

Send a message    

# Discussion

## Reservoir dedicated framework

- **Online/Offline** learning
- **Feedback loops**
- **Hierarchical** models
  - e.g. *Deep Reservoir Computing*
- **Parallel** computing
- R language, (interface)
- Datasets, metrics, ...
  - e.g. Memory Capacity
- Hyperoptimization tools
- **ReservoirR**: interface in R language
- Tutorials, examples, documentation, ...

## Reservoir models

- Echo State Networks (ESN)
- Autoregressive reservoir (NVAR)
- Ridge regression
- FORCE learning
- Intrinsic Plasticity
- New** • Reservoir topology (ring, orthogonal, ...)
- New** • Delay nodes
- New** • Scikit-learn models, (interface)
- New** • ES<sup>2</sup>N

➡ Share your models in [ReservoirPy](#)

## Perspectives

### Upgrade framework capabilities

- **Spiking neural networks**
  - *Liquid State Machines*
- **JAX back-end** (late 2024)
  - Extended distributed computation
  - GPU acceleration

### Sharing within RC community

- Replicate scientific results
- Implement new tools
- Increase user and expert community
- **Find new use cases**

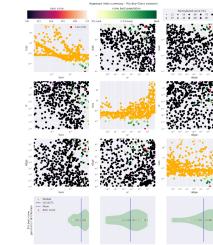
# Summary on reservoirpy



- Easy and quick to use



- Rich documentation and many tutorials (e.g. HPs optim.)



- Rich set of features + Upcoming new features



- Possible to interface with other languages



- We can build advanced and robust architectures by sharing models



<https://github.com/reservoirpy/reservoirpy>

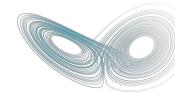


Looking for a post-doc and engineer!



Many thanks to  
Nathan Trouvain  
+ all developers

# Thank you!



- ➡ Share your models in [ReservoirPy](#)
  - ➡ Ideas of new features to implement?
- 
- ➡ Add your best papers to the **Reservoir “Awesome list”** ➡

[github.com/reservoirpy/awesome-reservoir-computing](https://github.com/reservoirpy/awesome-reservoir-computing)



# Some references using ReservoirPy

HAL id 	HAL citation
hal-03699931	Nathan Trouvain, Xavier Hinaut. reservoirpy: A Simple and Flexible Reservoir Computing Tool in Python. 2022. <a href="#">(hal-03699931)</a>
hal-02595026	Nathan Trouvain, Luca Pedrelli, Thanh Trung Dinh, Xavier Hinaut. ReservoirPy: an Efficient and User-Friendly Library to Design Echo State Networks. <i>ICANN 2020 - 29th International Conference on Artificial Neural Networks</i> , Sep 2020, Bratislava, Slovakia. <a href="#">(hal-02595026v2)</a>
hal-03533731	Nathan Trouvain, Xavier Hinaut. Reservoir Computing : théorie, intuitions et applications avec ReservoirPy. <i>Plate-Forme Intelligence Artificielle (PfIA)</i> , Jun 2021, Bordeaux, France. <a href="#">(hal-03533731)</a>
hal-03203318	Xavier Hinaut, Nathan Trouvain. Which Hype for my New Task? Hints and Random Search for Reservoir Computing Hyperparameters. <i>ICANN 2021 - 30th International Conference on Artificial Neural Networks</i> , Sep 2021, Bratislava, Slovakia. <a href="#">(hal-03203318v2)</a>
hal-03482372	Silvia Pagliarini, Arthur Leblois, Xavier Hinaut. Canary Vocal Sensorimotor Model with RNN Decoder and Low-dimensional GAN Generator. <i>ICDL 2021- IEEE International Conference on Development and Learning</i> , Aug 2021, Beijing, China. <a href="#">(hal-03482372)</a>
hal-03203374	Nathan Trouvain, Xavier Hinaut. Canary Song Decoder: Transduction and Implicit Segmentation with ESNs and LTSMs. <i>ICANN 2021 - 30th International Conference on Artificial Neural Networks</i> , Sep 2021, Bratislava, Slovakia. pp.71--82, <a href="#">(10.1007/978-3-030-86383-8_6)</a> . <a href="#">(hal-03203374v2)</a>
hal-03761440	Nathan Trouvain, Nicolas P. Rougier, Xavier Hinaut. Create Efficient and Complex Reservoir Computing Architectures with ReservoirPy. <i>SAB 2022 - FROM ANIMALS TO ANIMATS 16: The 16th International Conference on the Simulation of Adaptive Behavior</i> , Sep 2022, Cergy-Pontoise / Hybrid, France. <a href="#">(hal-03761440)</a>
tel-03946773	Xavier Hinaut. Reservoir SMILES: Towards SensoriMotor Interaction of Language and Embodiment of Symbols with Reservoir Architectures. <i>Artificial Intelligence [cs.AI]</i> . Université de Bordeaux (UB), France, 2022. <a href="#">(tel-03946773)</a>
hal-03628290	Subba Reddy Oota, Frédéric Alexandre, Xavier Hinaut. Cross-Situational Learning Towards Robot Grounding. 2022. <a href="#">(hal-03628290v2)</a>
hal-03780006	Xavier Hinaut, Nathan Trouvain. ReservoirPy: Efficient Training of Recurrent Neural Networks for Timeseries Processing. <i>EuroSciPy 2022 - 14th European Conference on Python in Science</i> , Aug 2022, Basel, Switzerland. <a href="#">(hal-03780006)</a>
hal-03945994	Nathan Trouvain, Xavier Hinaut. Reservoir Computing : traitement efficace de séries temporelles avec ReservoirPy. <i>Dataquitaine 2022</i> , Feb 2022, Bordeaux, France. <a href="#">(hal-03945994)</a>