

Mardi 21 Janvier 2025 – workshop AI4industry



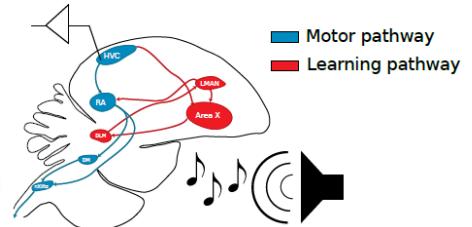
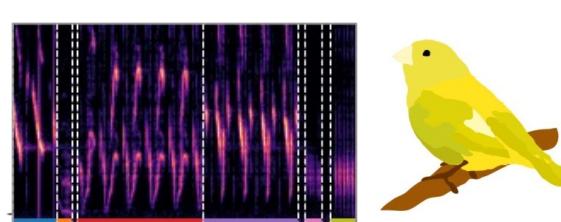
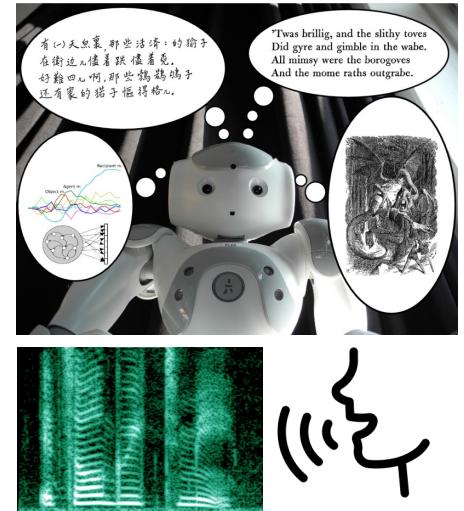
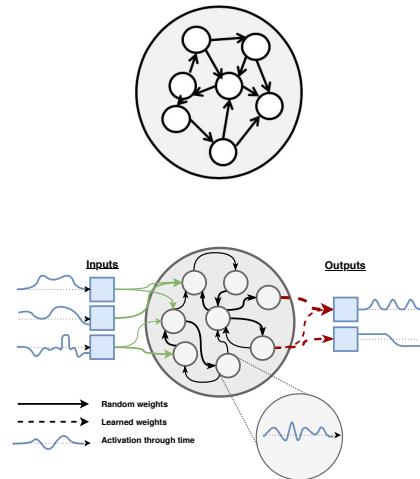
Le Reservoir Computing

Prédire, générer, discriminer des séquences ...

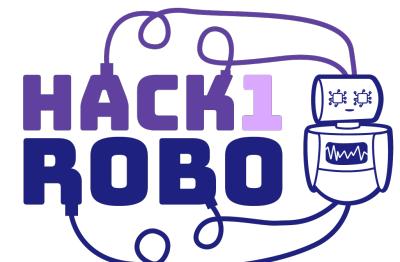


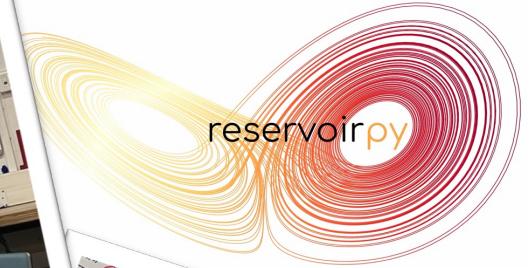
Computational Neuroscience
Cognition
Mnemosyne
Brain & Body Complex

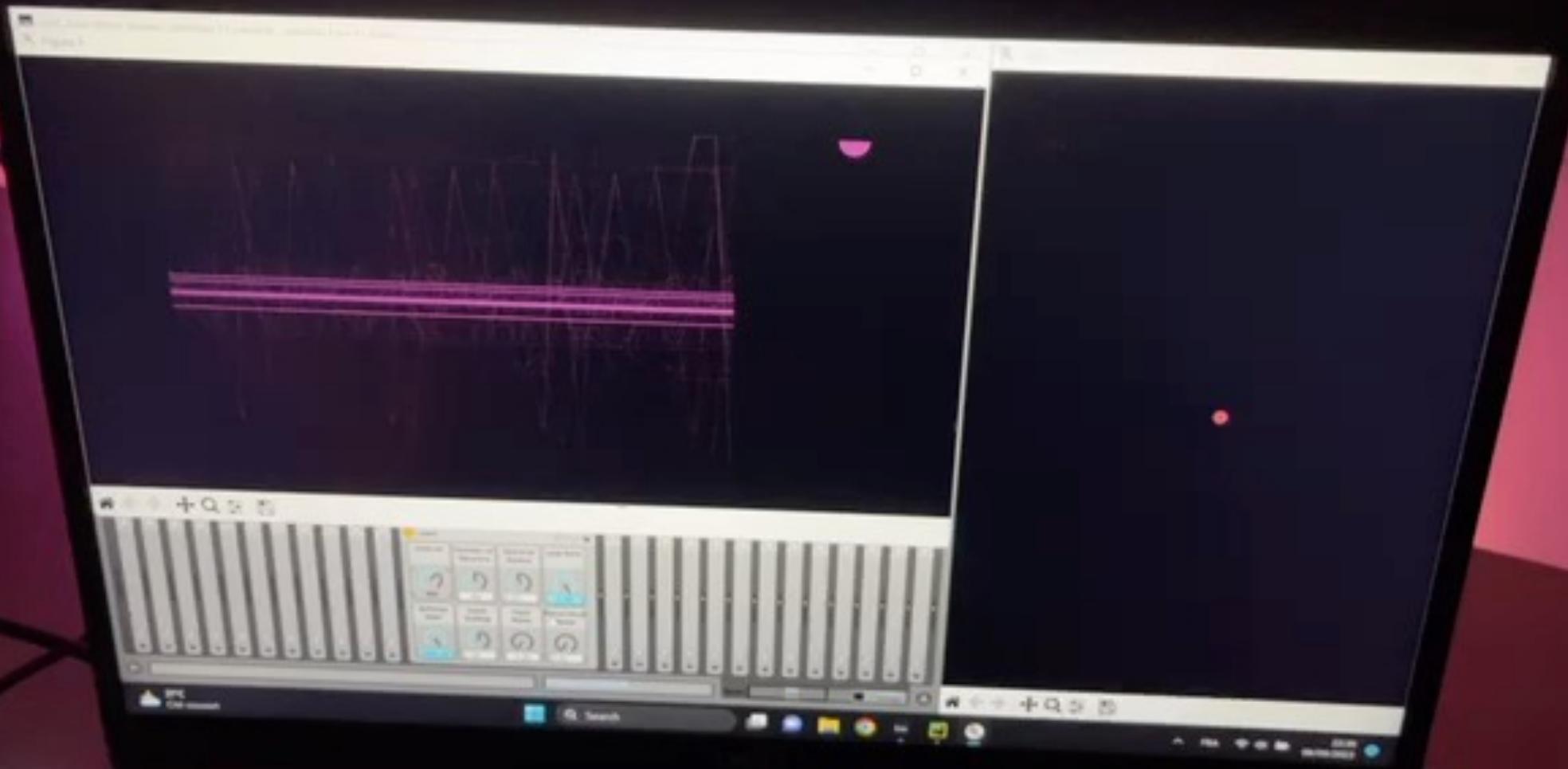
LaBRI



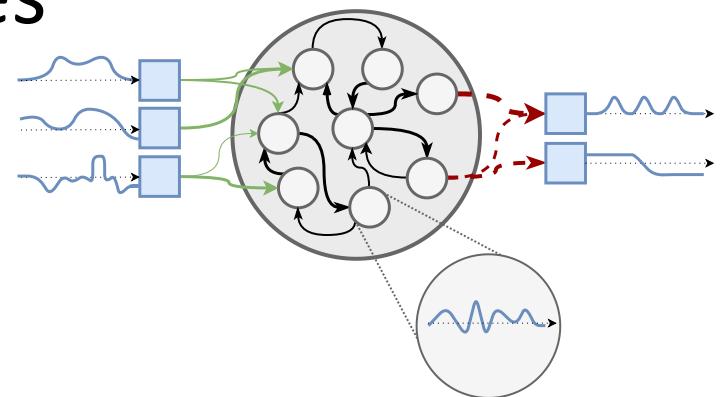
xavier.hinaut@inria.fr
 @neuronalX



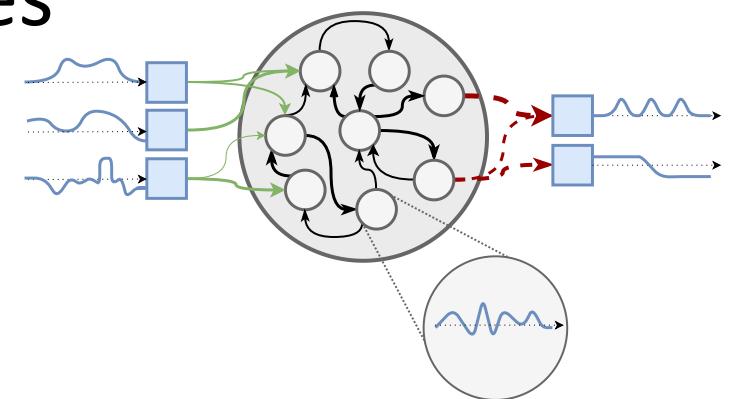




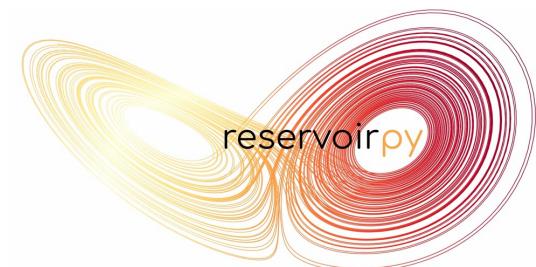
Les **reseaux aléatoires** peuvent
générer des dynamiques fascinantes



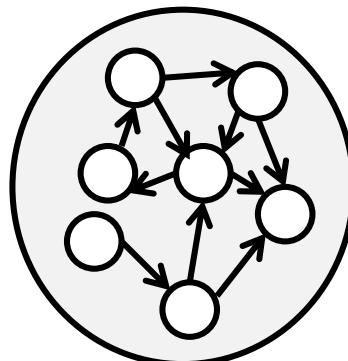
Les **reseaux aléatoires** peuvent
générer des dynamiques fascinantes



Le **reservoir computing** exploite
les **dynamiques naturelles** d'un réseau aléatoire



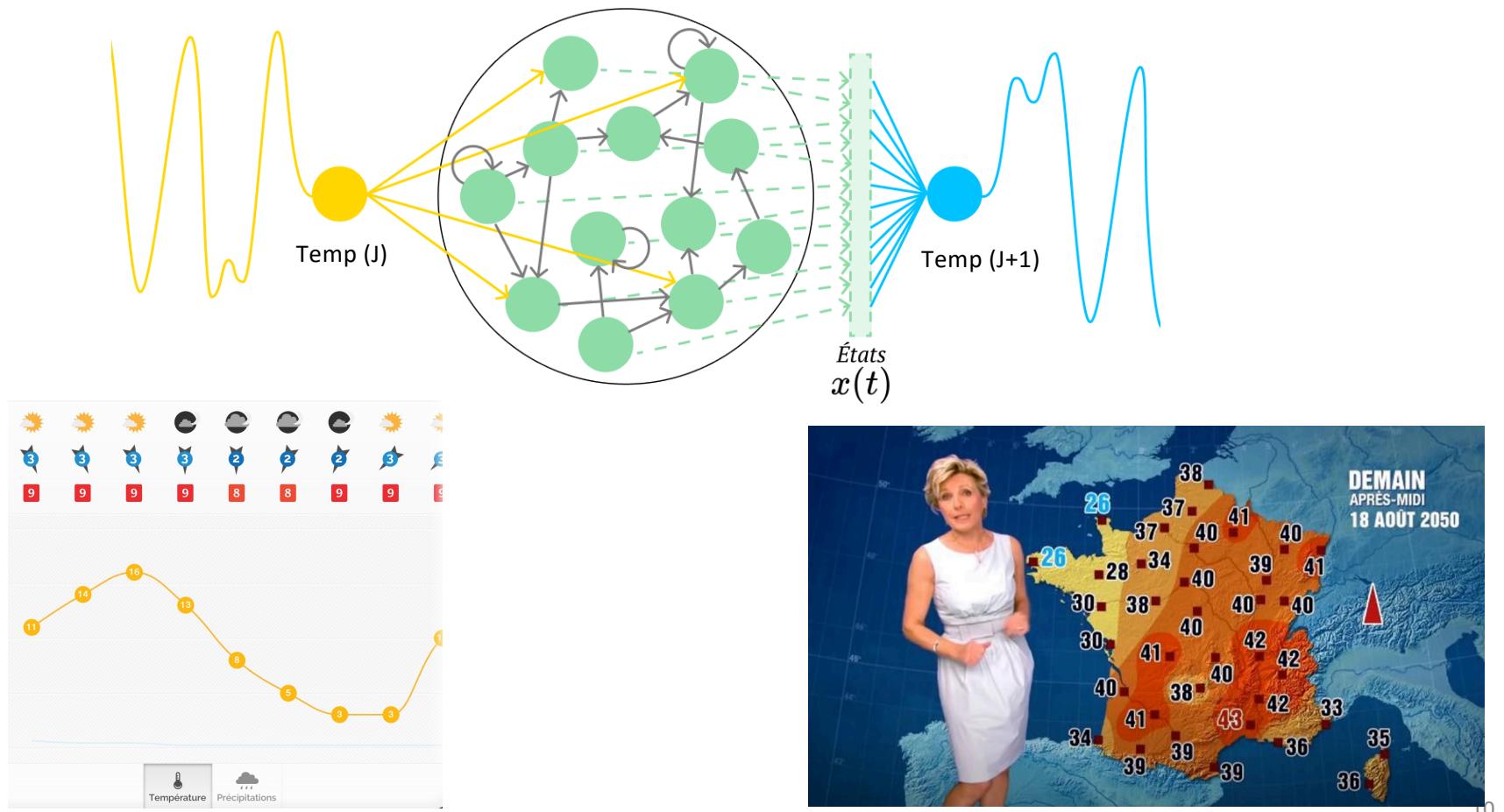
A quoi servent ces Réseaux de Neurones Récursifs ?

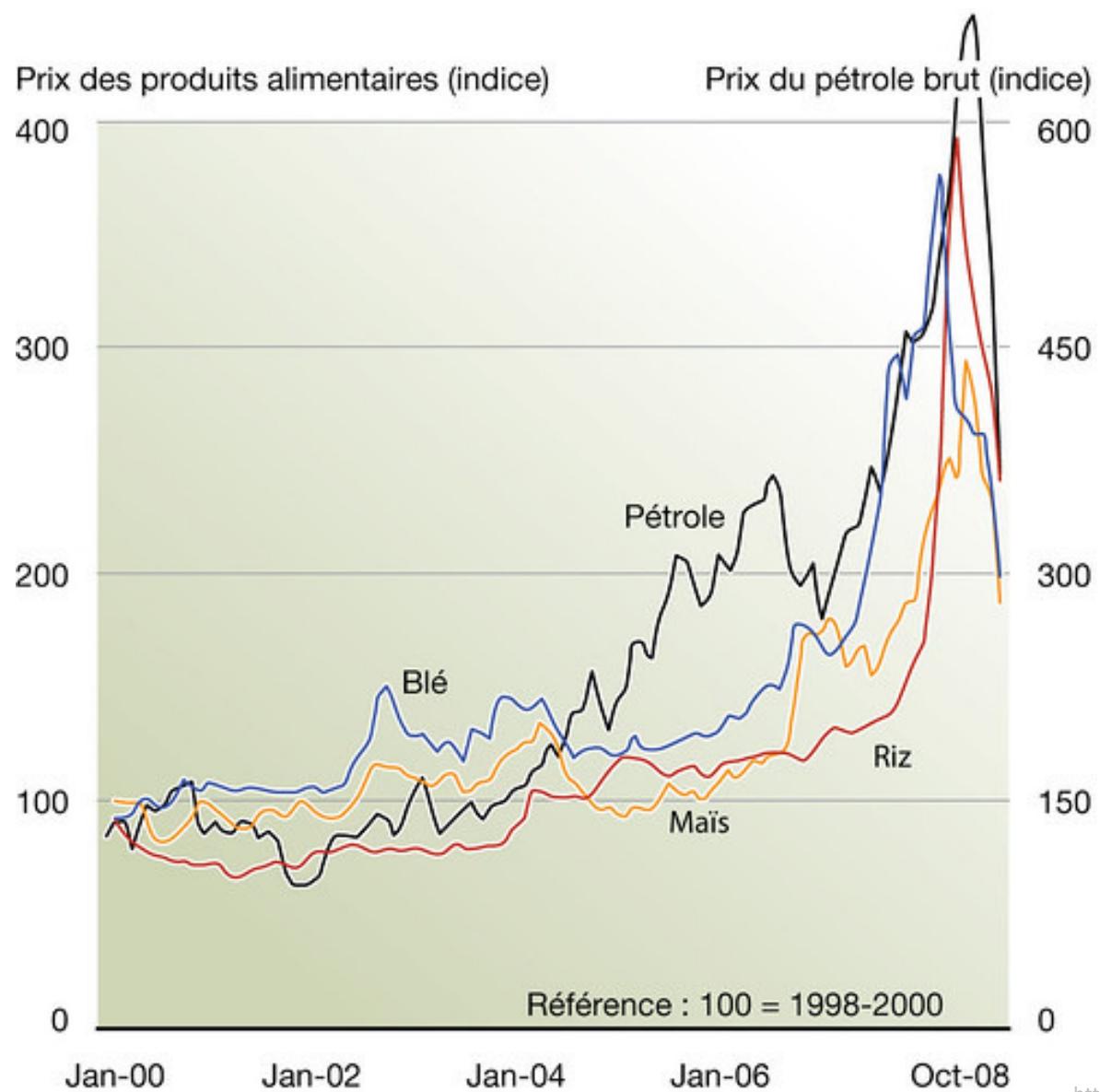


○ Neurone

→ Connexion entre deux neurones

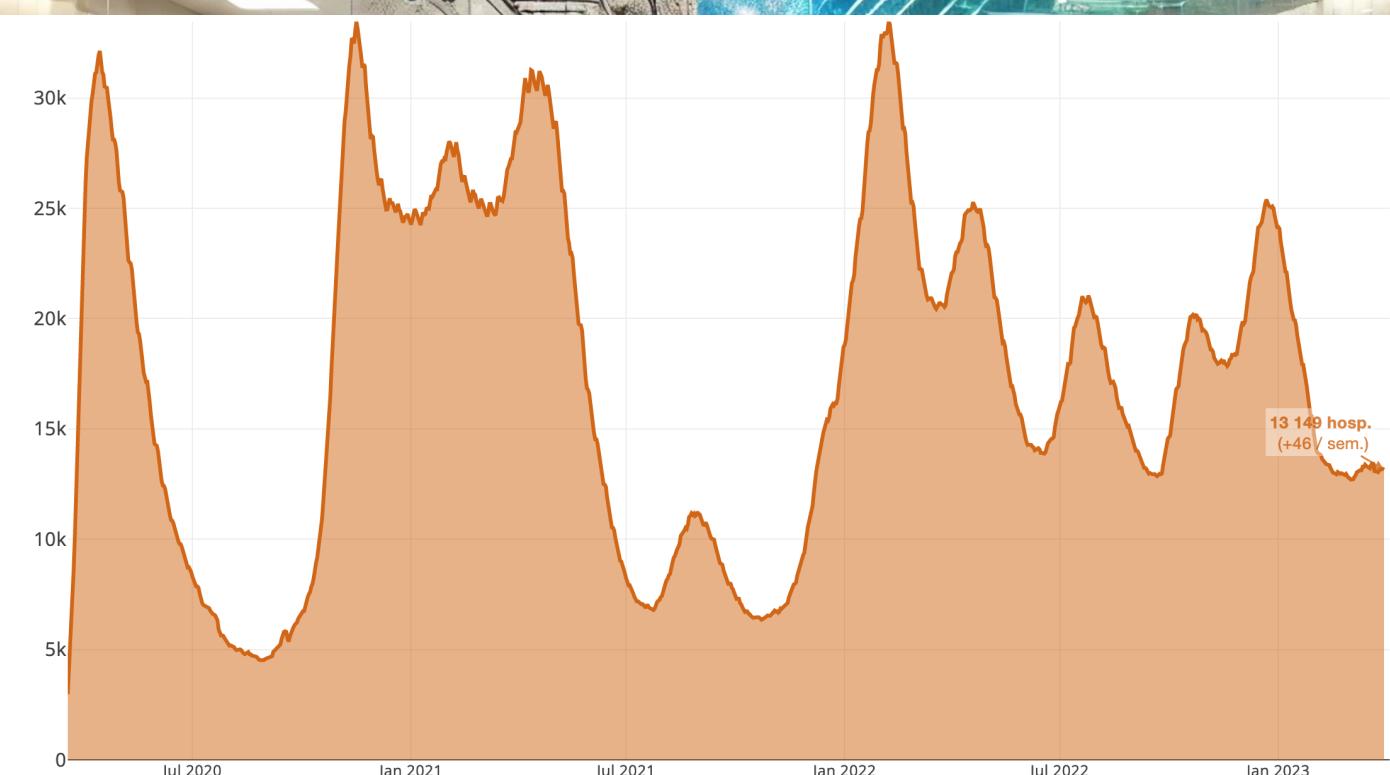
Prédiction de séries temporelles





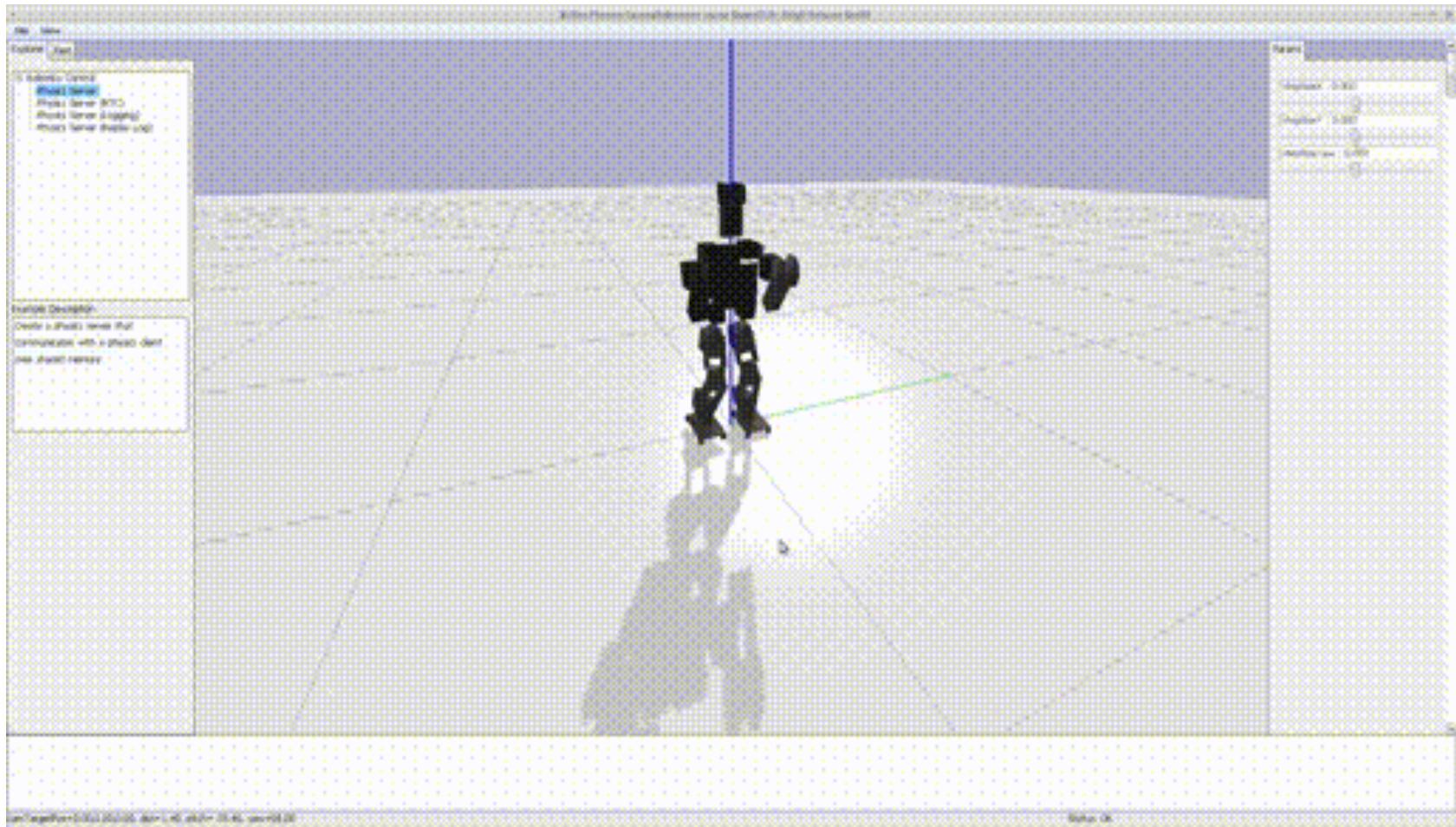
<https://www.flickr.com/photos/gridarendal/32211647472> (by-nc-sa/2.0/)

Prédiction d'hospitalisations COVID



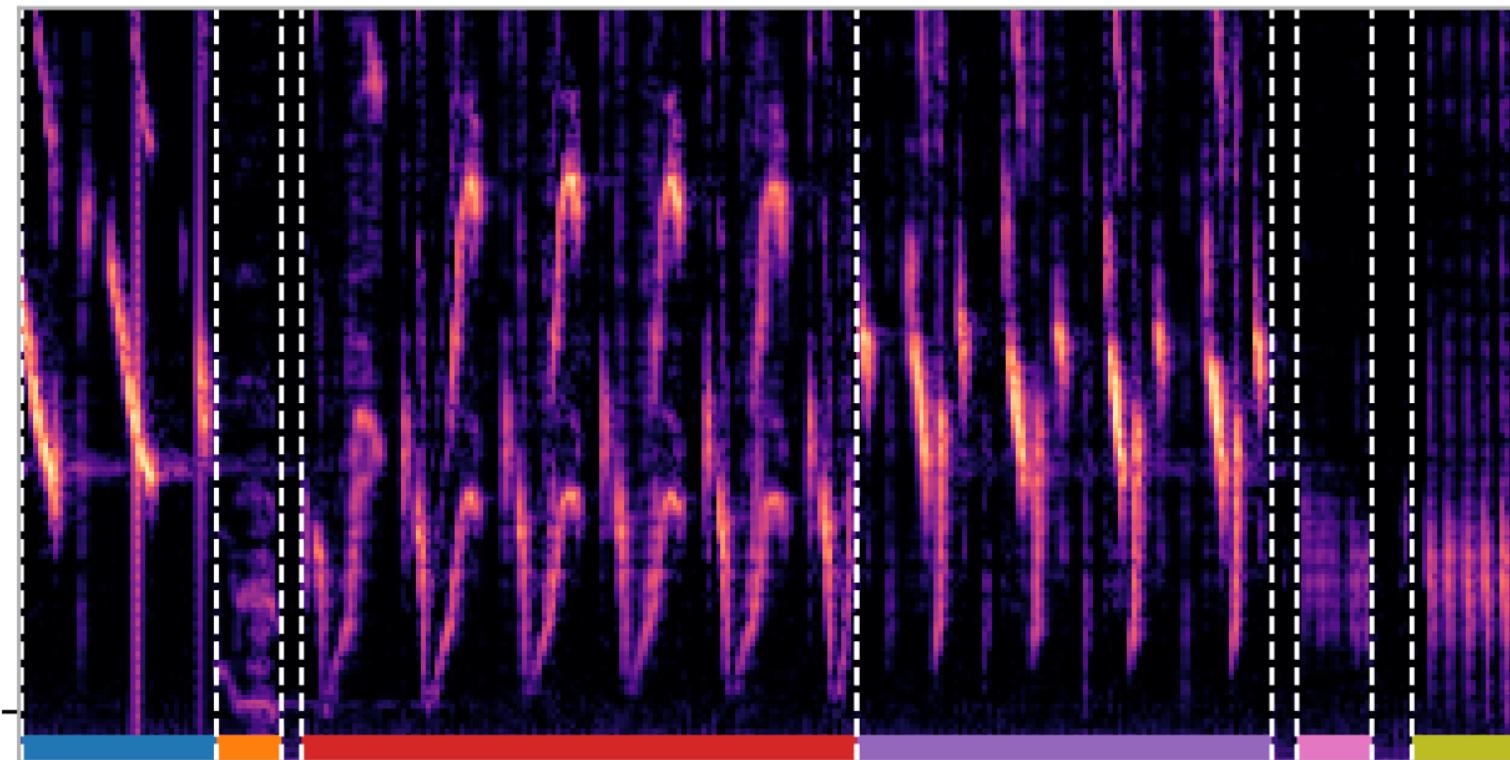
<https://covidtracker.fr/france/#hospitalisations>

Prédiction d'événements (chute, panne, ...)

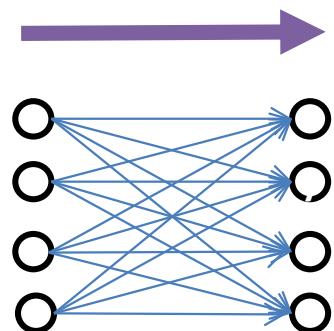


Classifier des sons

(syllabes de canari)



2 types de réseaux



- Réseau de neurones en couches
 - Connexions *d'une couche à la suivante*
 - ex : Faire des associations (Image → « Chat »)

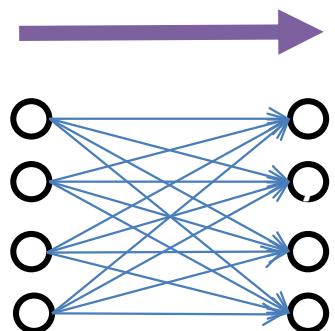


Neurone



Connexion entre neurones

2 types de réseaux



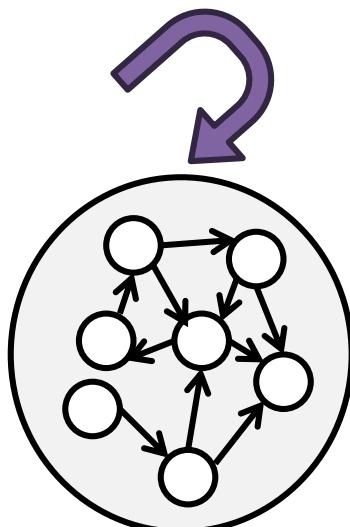
- Réseau de neurones en couches
 - Connexions *d'une couche à la suivante*
 - ex : Faire des associations (Image → « Chat »)



Neurone

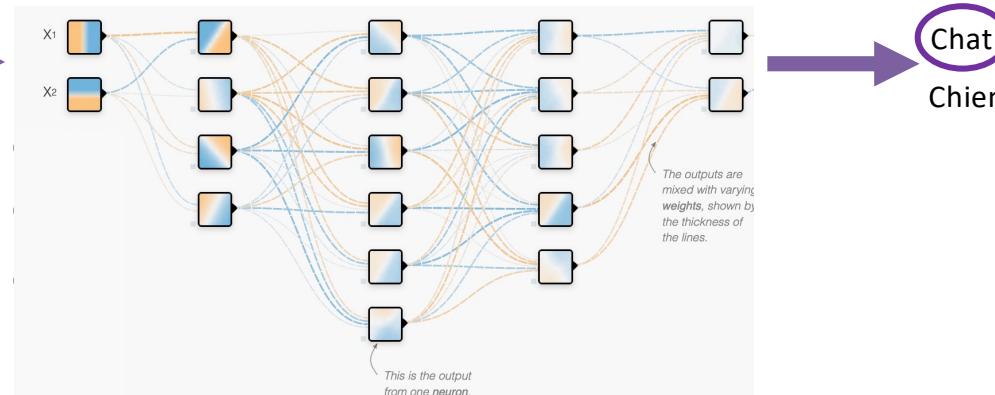


Connexion entre neurones

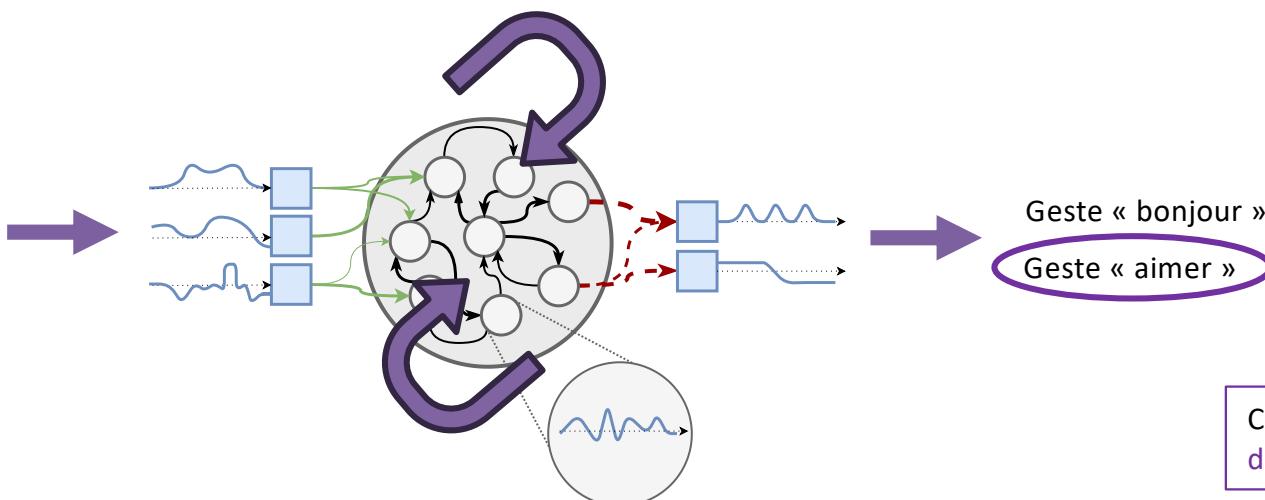


- Réseau de neurones récurrent
 - Connexions *au sein d'une même couche*
 - Mémoire dynamique du contexte temporel
 - ex : reconnaître une séquence (mouvement)

2 types de réseaux

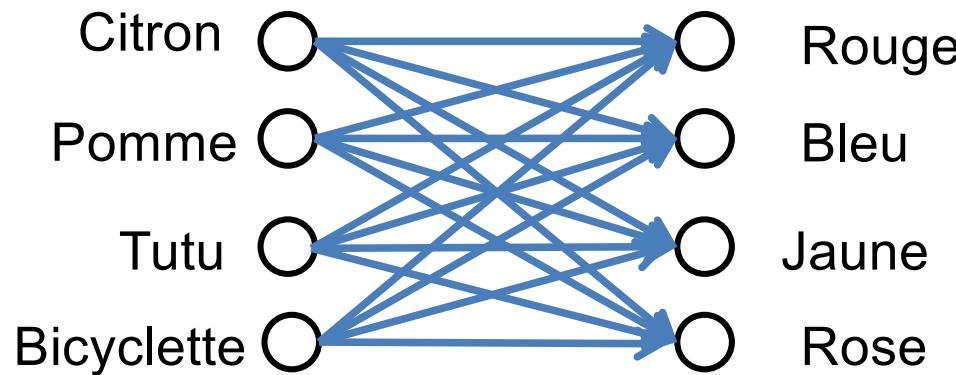


Chaque stimulus est indépendant du précédent



Chaque stimulus est dépendant du précédent

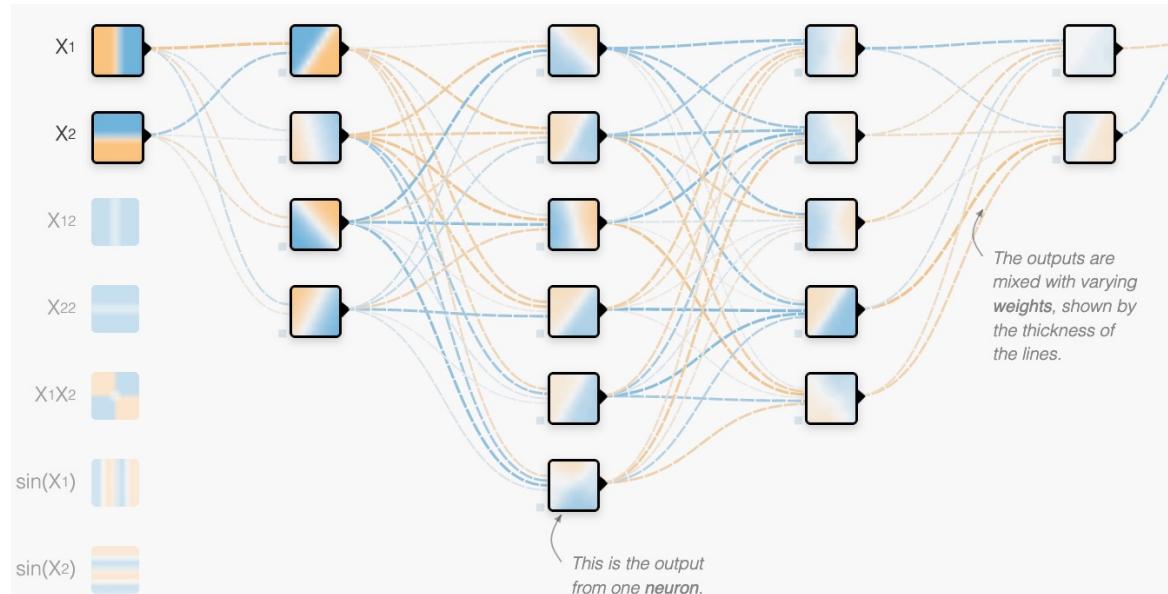
Un réseau de neurones en couches



- Loi de Hebb (1949)
 - lorsque 2 neurones sont actifs en même temps
leurs connexions sont renforcés
- « Neurons that fire together, wire together »

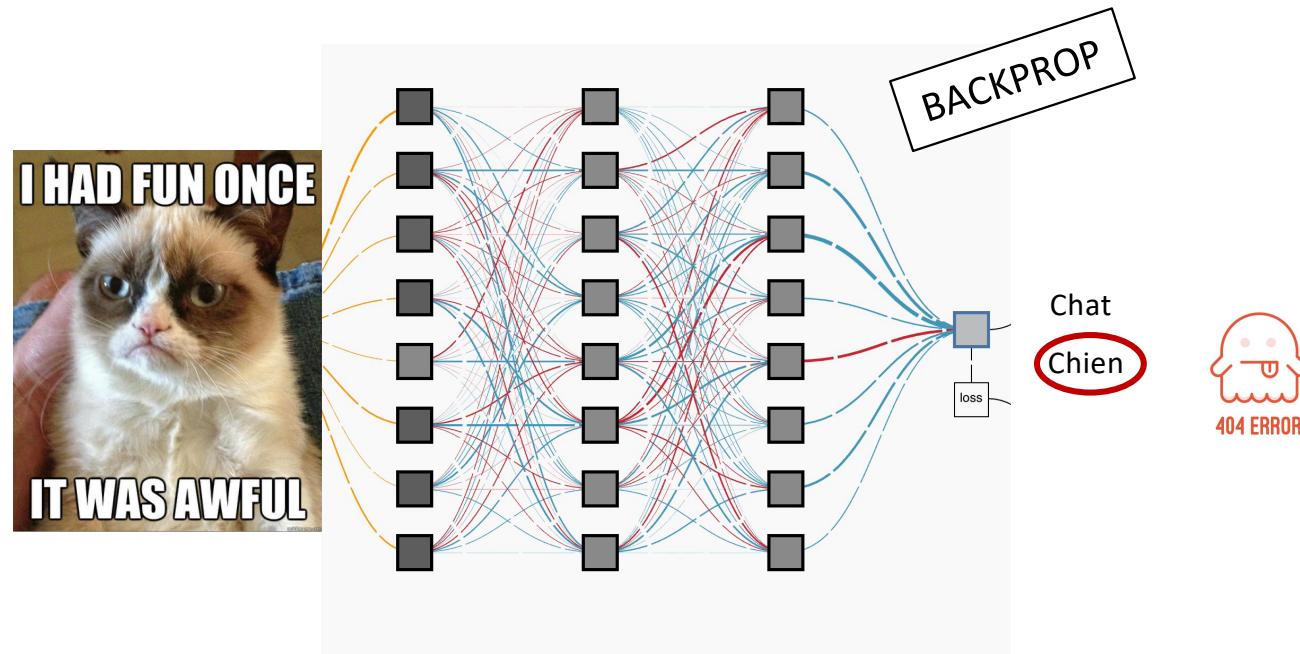
Un réseau de neurones en couches

L'algorithme de rétro-propagation du gradient permet d'apprendre avec plusieurs couches
→ Deep Learning

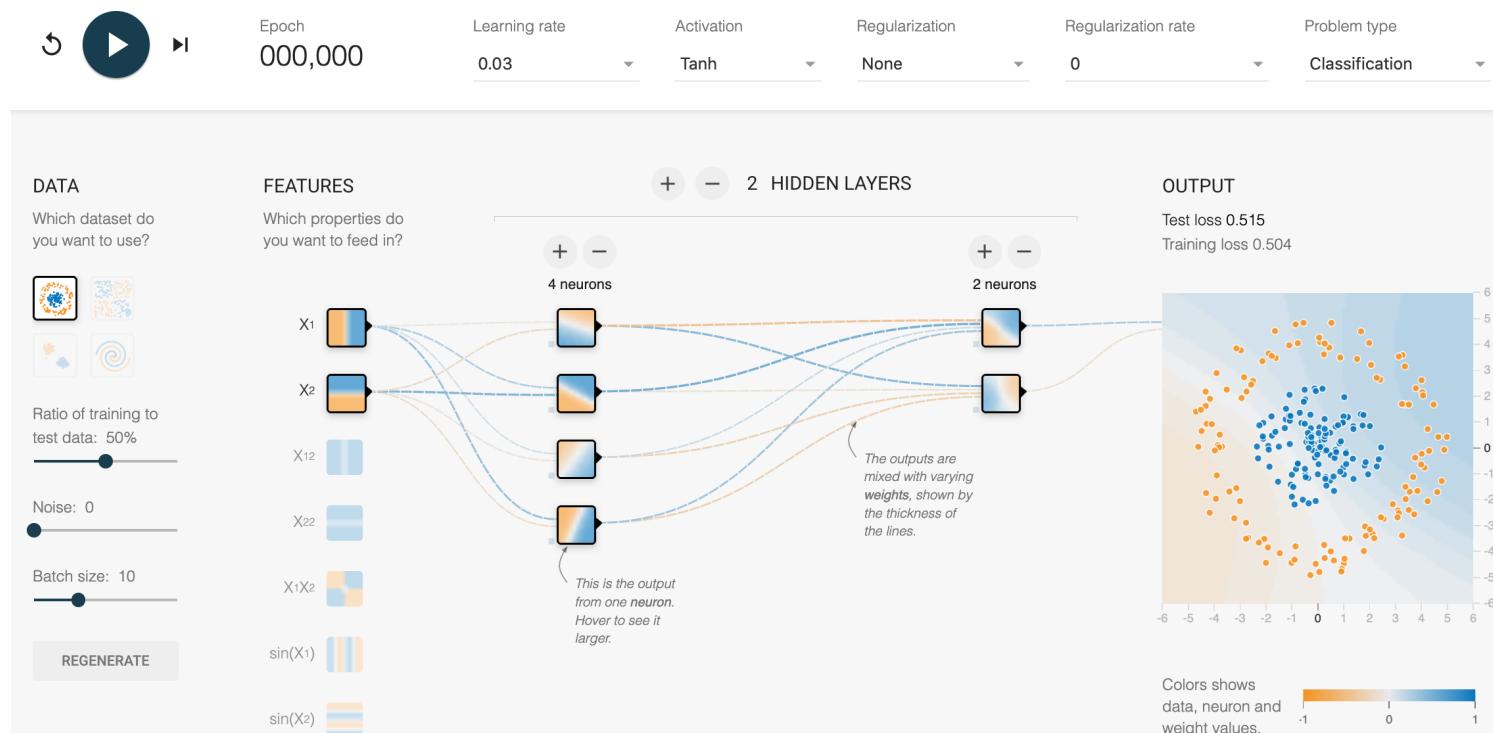


Un réseau de neurones en couches

L'algorithme de rétro-propagation du gradient permet d'apprendre avec plusieurs couches
→ Deep Learning



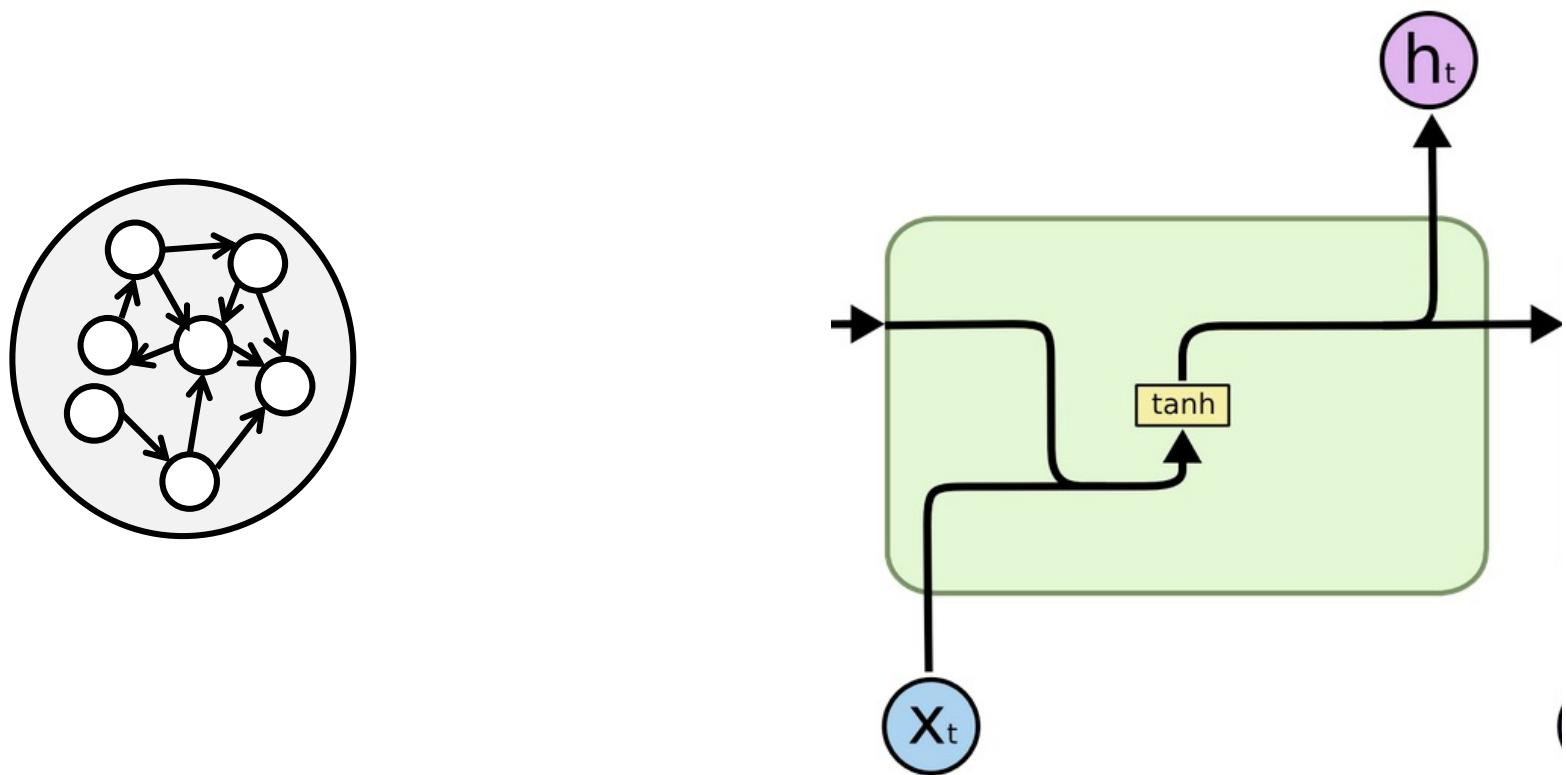
Testez votre intuition des réseaux de neurones



playground.tensorflow.org

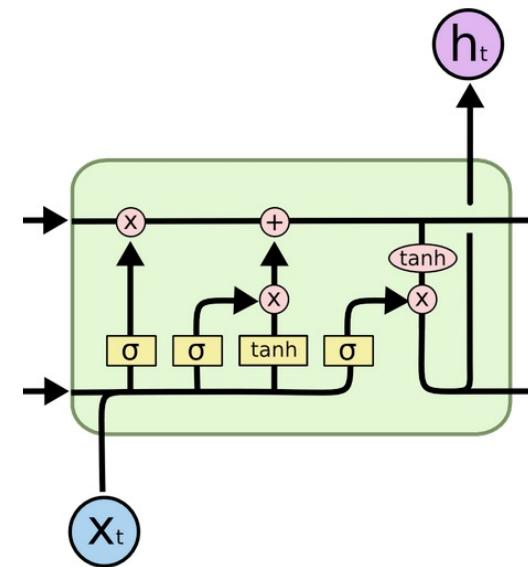
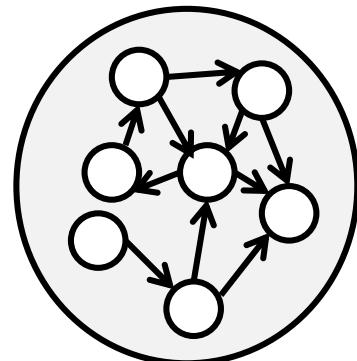
RNN

(Recurrent Neural Network)



LSTM

(Long Short Term Memory network)



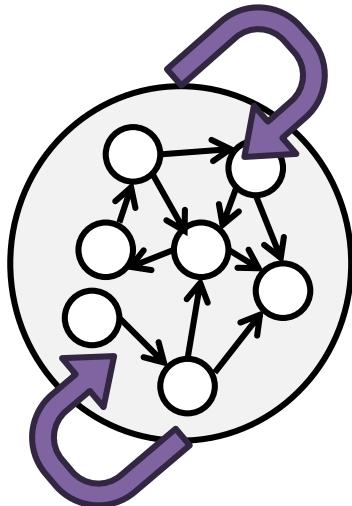
Entraîner un RNN/LSTM avec la backprop

= virtualiser le temps

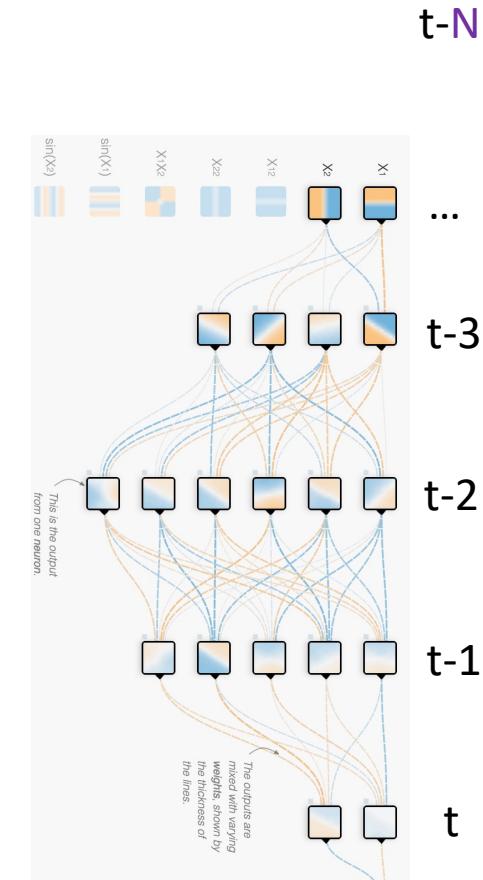
= déplier les récurrences

→ les mettre à plat comme un réseau en couches

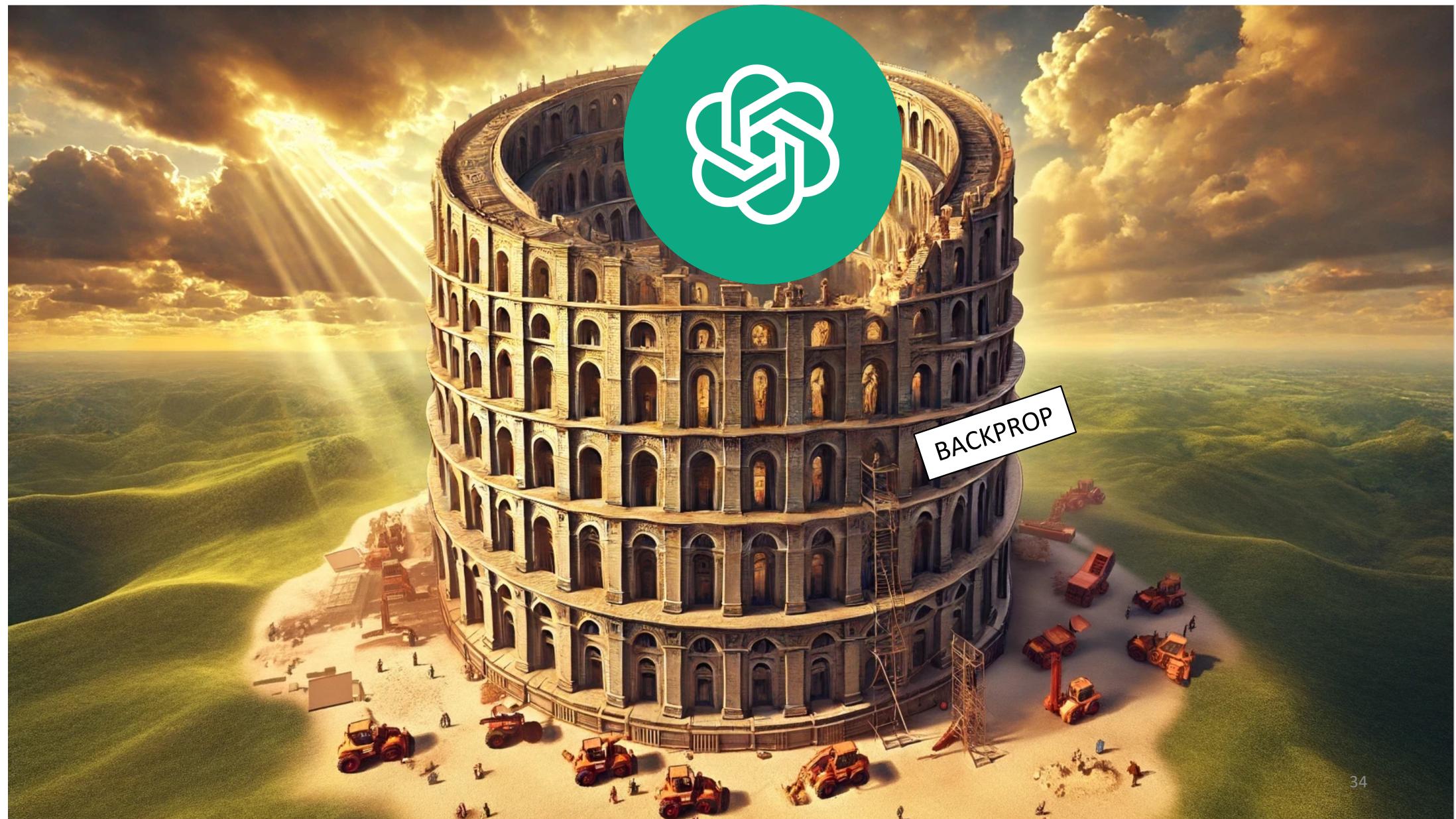
$t-N, \dots, t-2, t-1, t, \dots$

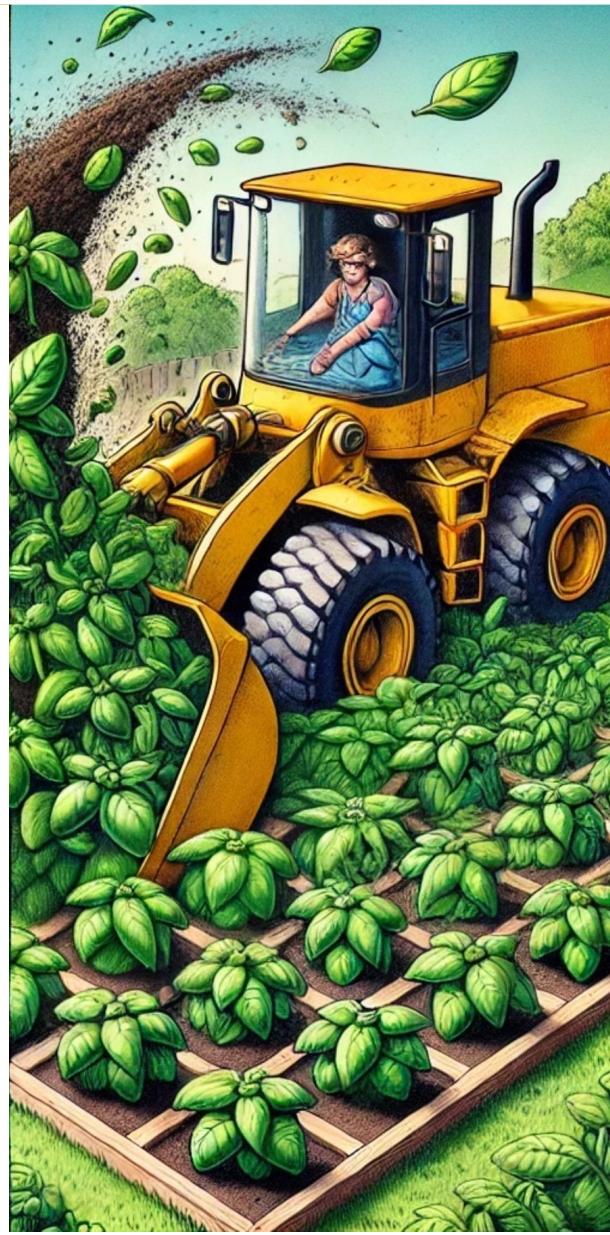


BACKPROP



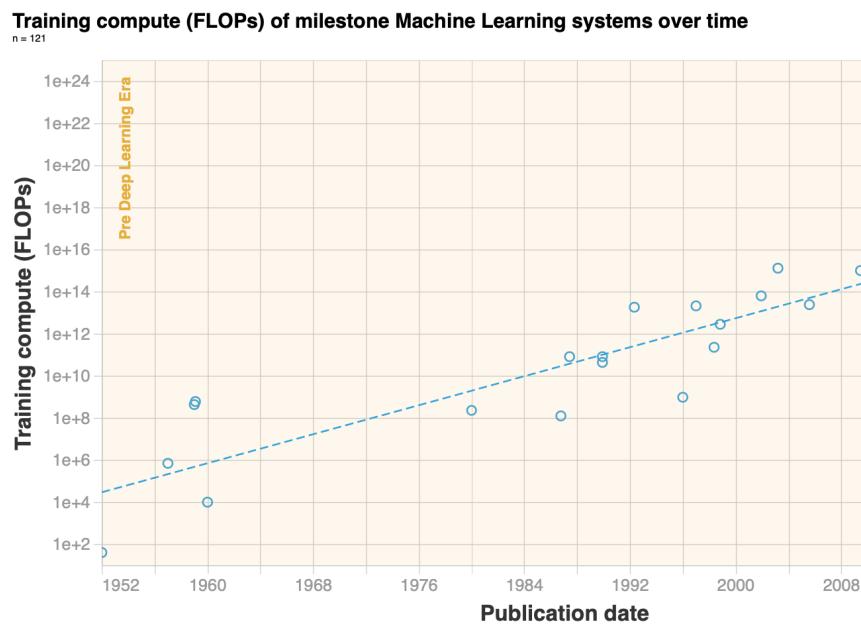
Nombre de couches = taille du contexte temporel → N











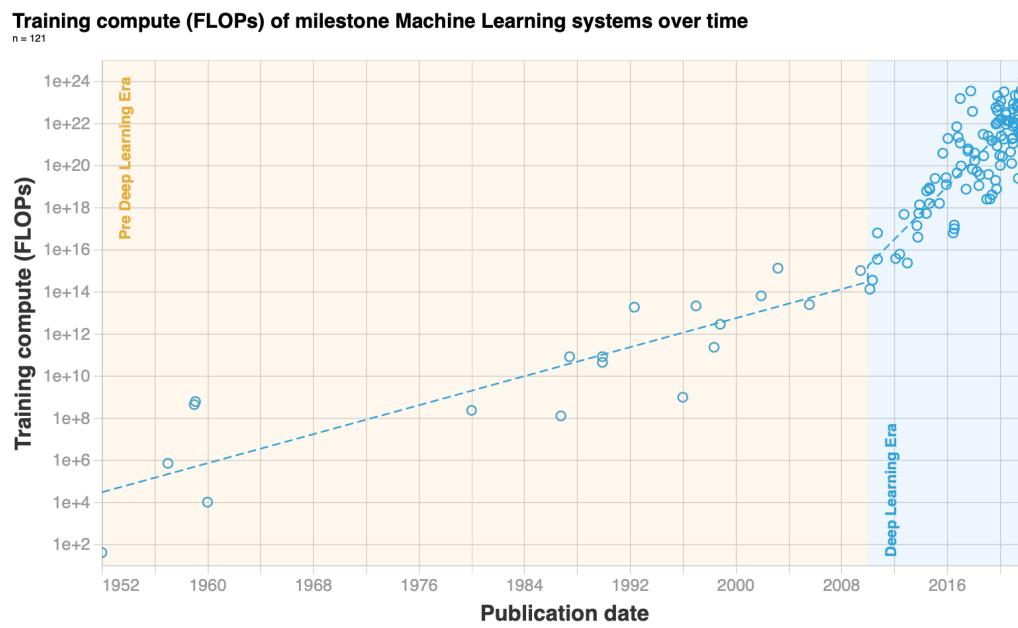
Doublement de la puissance de calcul

Loi de Moore
 $\approx 2 \text{ ans}$

Algo. de DL
 $\approx 3 \text{ mois}$

Sevilla, Jaime, et al. "Compute Trends Across Three Eras of Machine Learning." *arXiv e-prints* (2022): arXiv-2202.

Thanks to Claudio Gallicchio for the slide



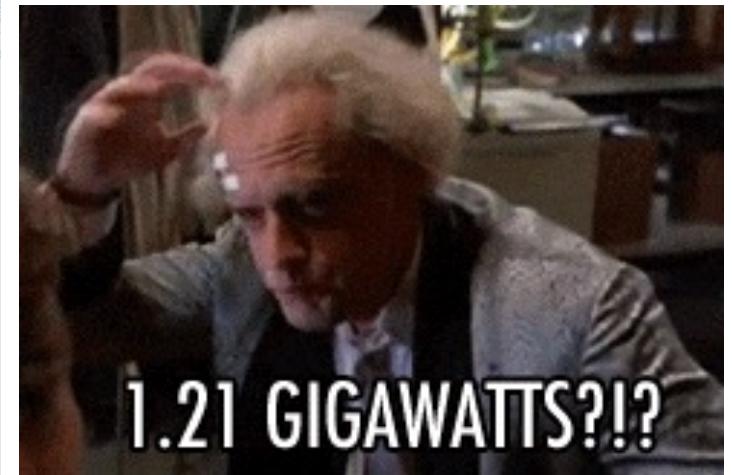
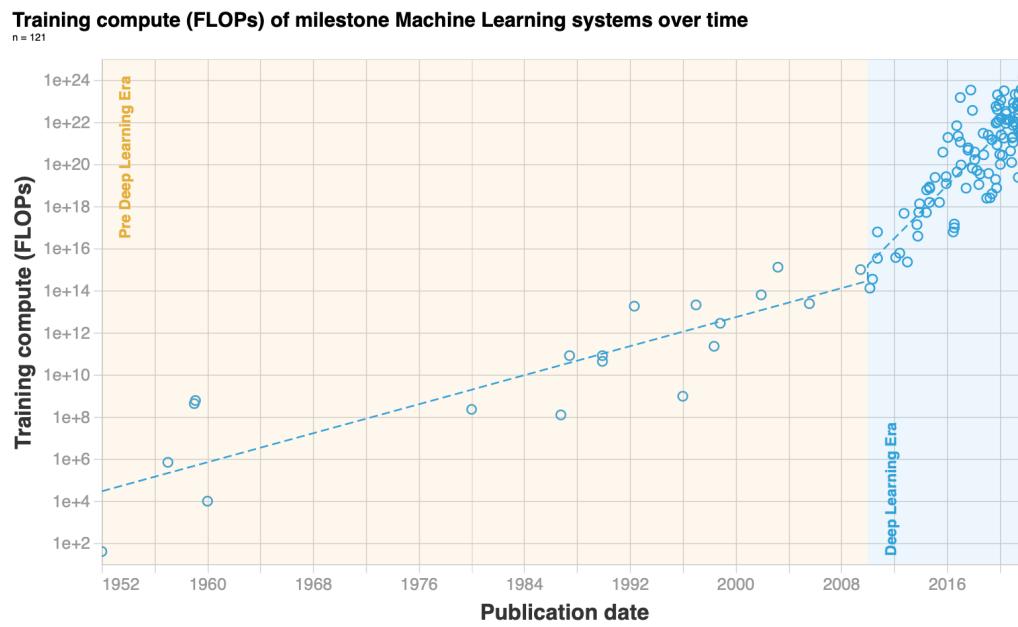
Doublement de la puissance de calcul

Loi de Moore
 $\approx 2 \text{ ans}$

Algo. de DL
 $\approx 3 \text{ mois}$

Sevilla, Jaime, et al. "Compute Trends Across Three Eras of Machine Learning." *arXiv e-prints* (2022): arXiv-2202.

Thanks to Claudio Gallicchio for the slide

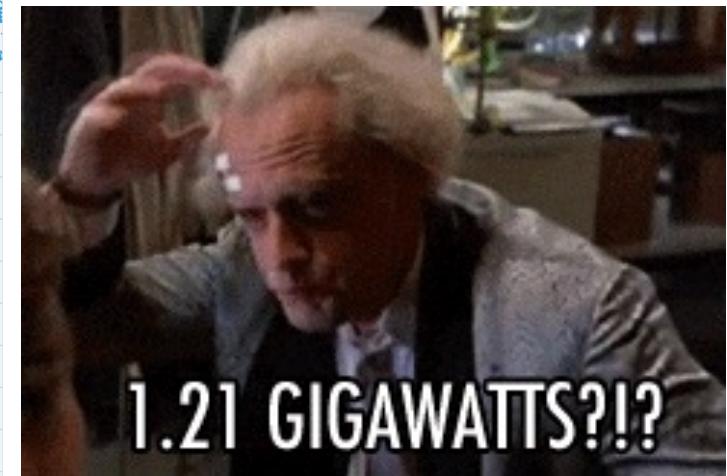
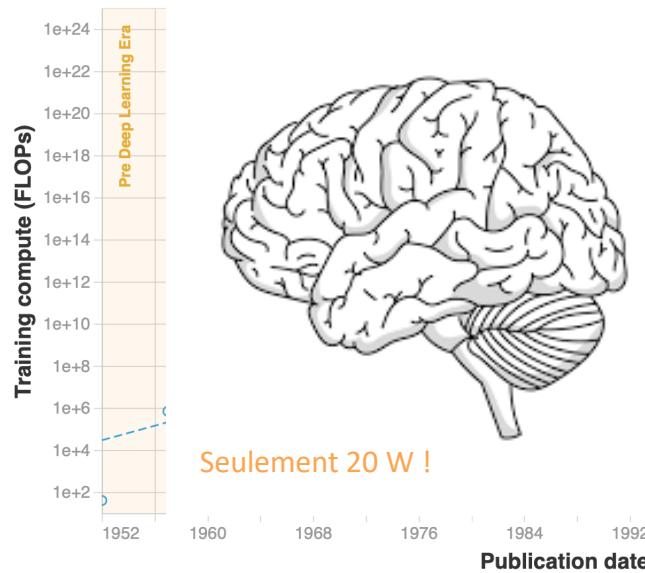


Sevilla, Jaime, et al. "Compute Trends Across Three Eras of Machine Learning." *arXiv e-prints* (2022): arXiv-2202.

Thanks to Claudio Gallicchio for the slide

Training compute (FLOPs) of milestone Machine Learning systems over time

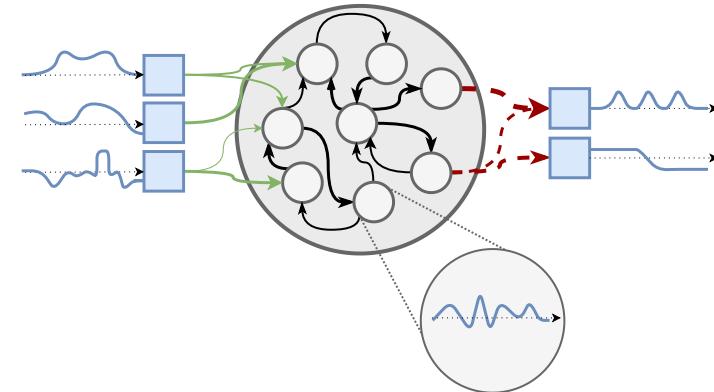
n = 121

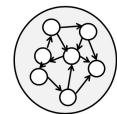


Sevilla, Jaime, et al. "Compute Trends Across Three Eras of Machine Learning." *arXiv e-prints* (2022): arXiv-2202.

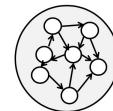
Thanks to Claudio Gallicchio for the slide

Peut-on avoir un bon équilibre entre performances
et
efficacité énergétique
+ temps de calcul ?

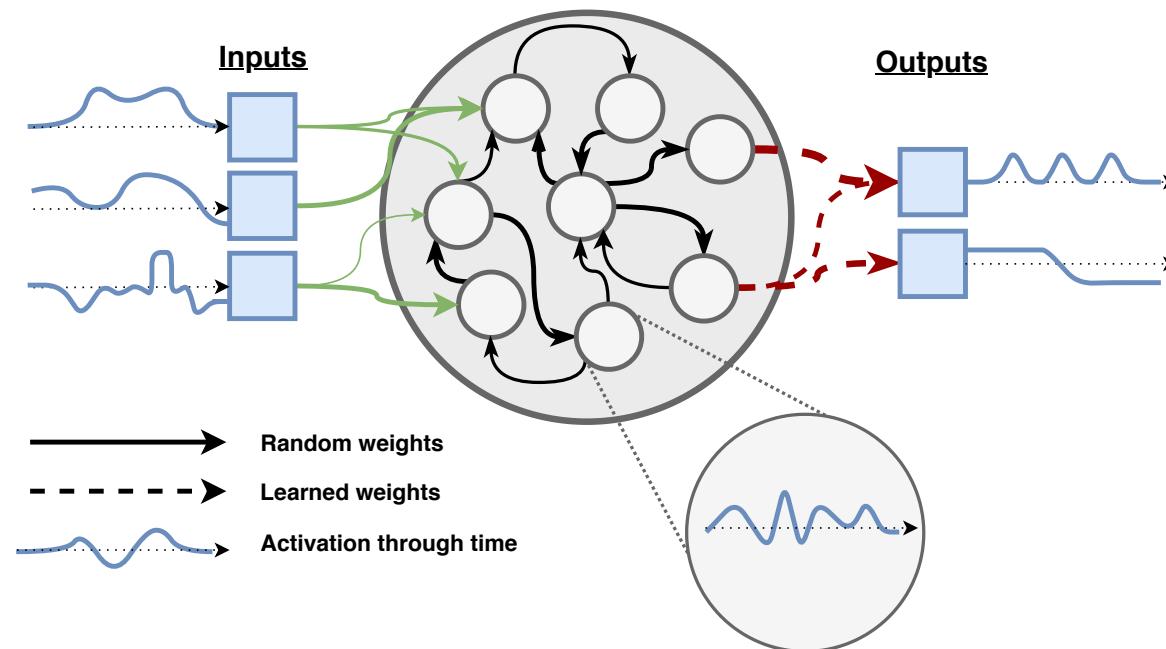




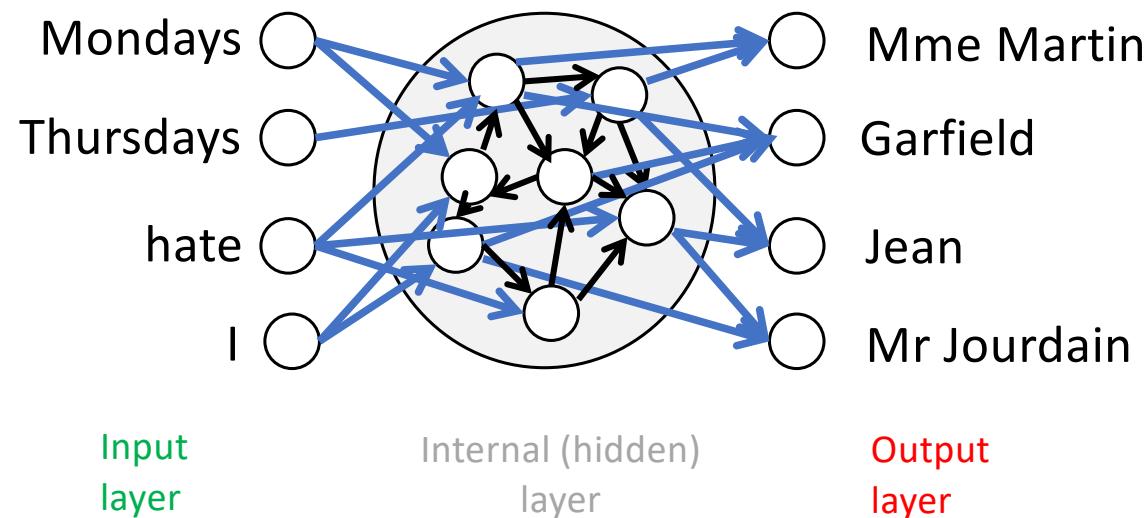
Reservoir Computing



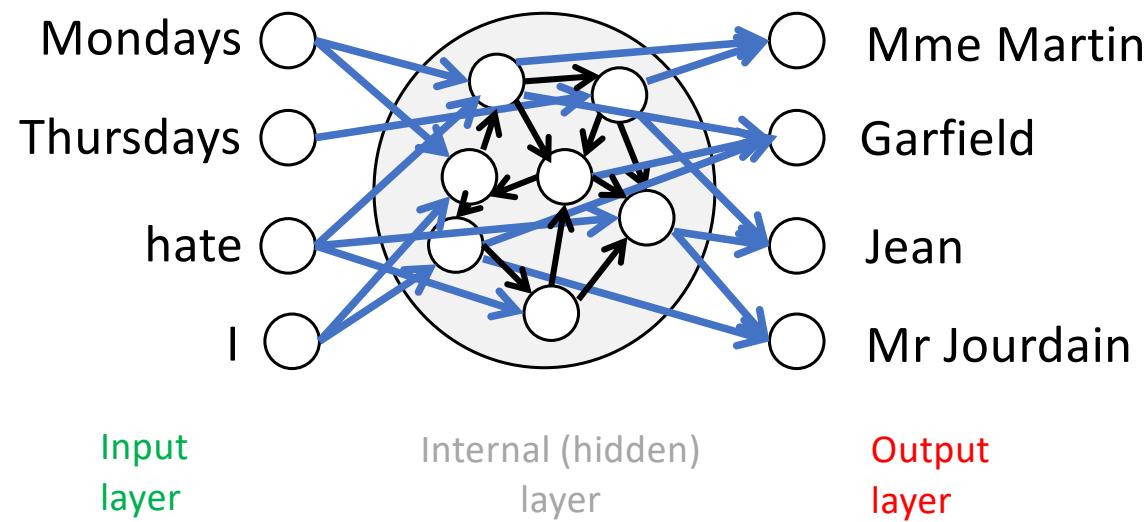
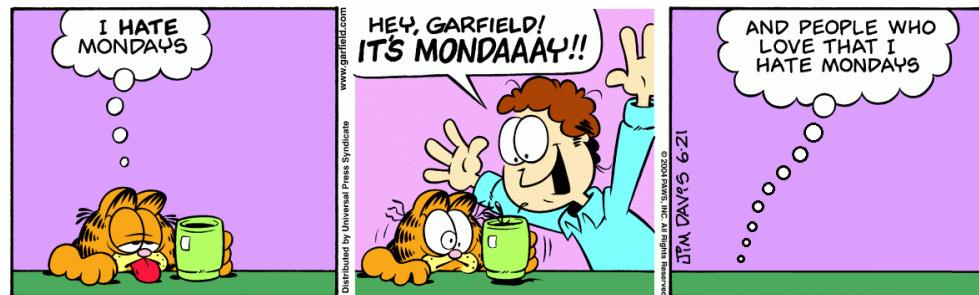
Ceci n'est pas un réseau de neurones récurrent (RNN) classique !

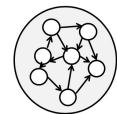


Le **Reservoir Computing** exploite les dynamiques naturelles d'un réseau récurrent aléatoire

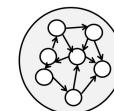


Le Reservoir Computing exploite les dynamiques naturelles d'un réseau récurrent aléatoire

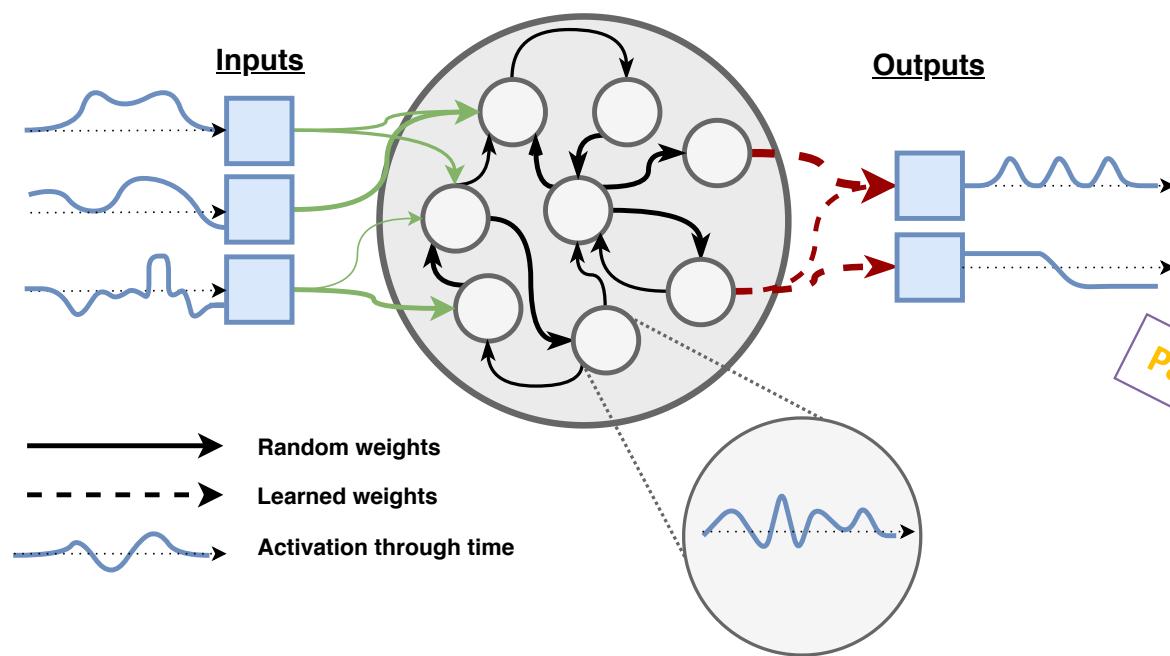




Reservoir Computing



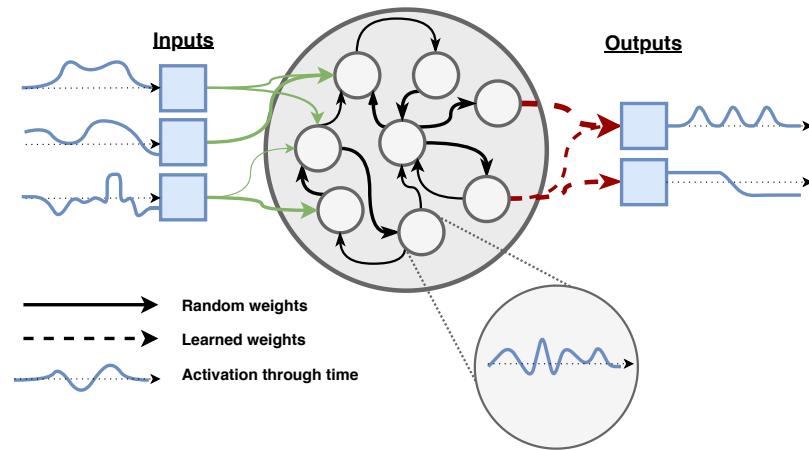
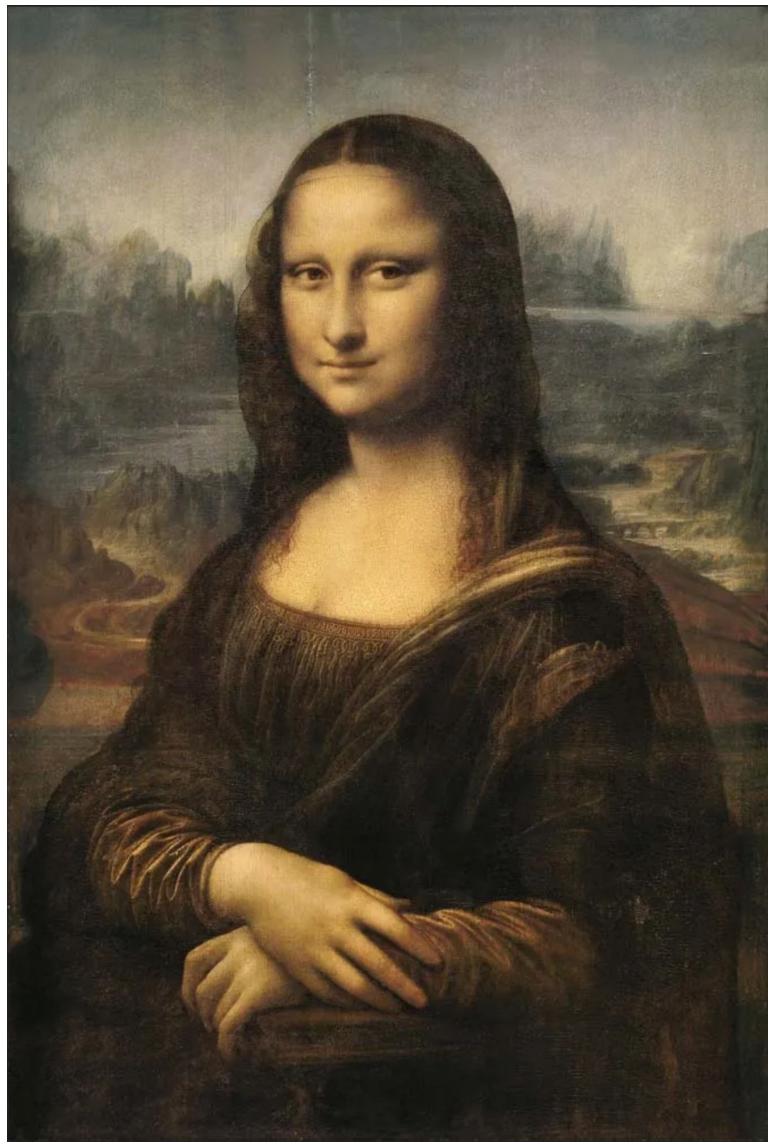
Poids aléatoires, MAIS dynamiques déterministes !



Poids aléatoires = pas d'a priori sur les connexions

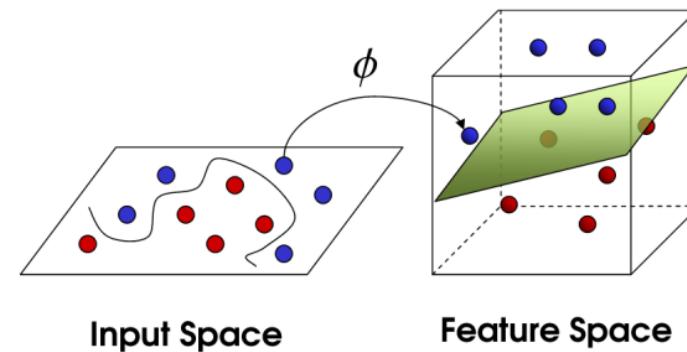
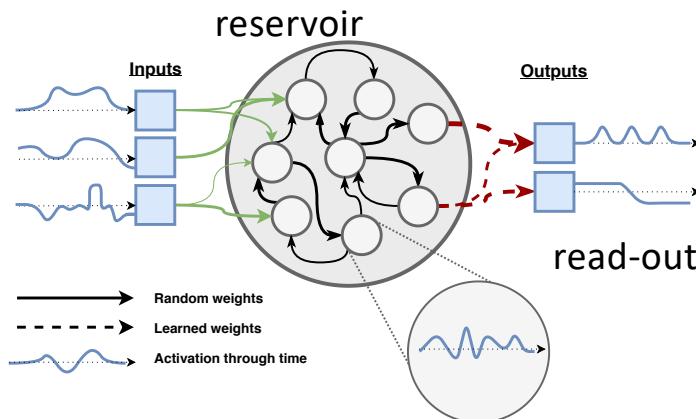


Seulement 20 W !

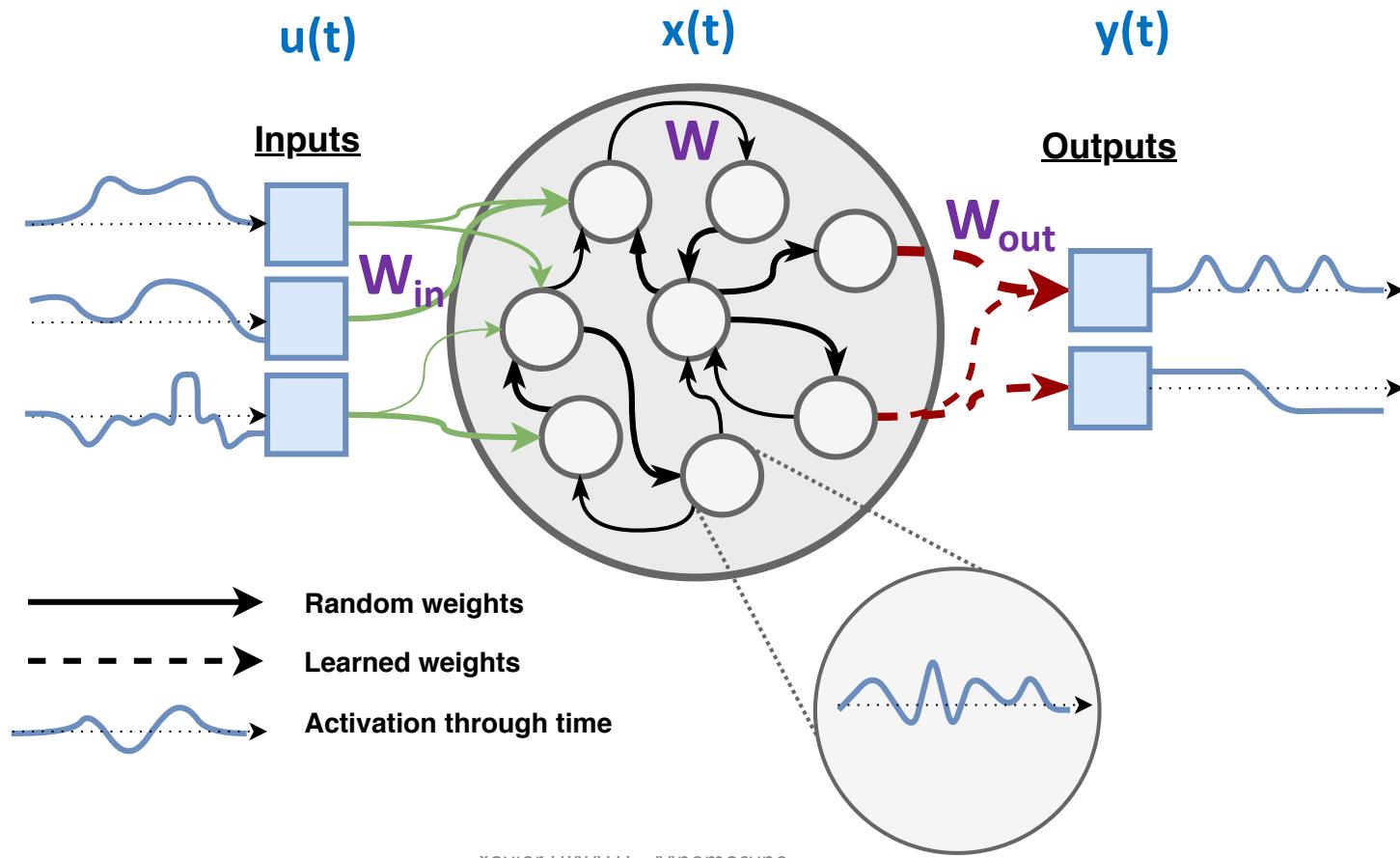


Pourquoi ce nom ?

- Réservoir = couche récurrente
- read-out = couche de sortie
- beaucoup de combinaisons des entrées sont faites dans le reservoir
 - grâce aux projections aléatoires
- “réservoir” est littéralement un réservoir de calculs (= “reservoir computing”)
 - calculs non-linéaires
- dans lequel on vient “piocher”/décoder (= “read-out”) linéairement
 - les combinaisons qui vont être utiles pour le problème



Reservoir Computing



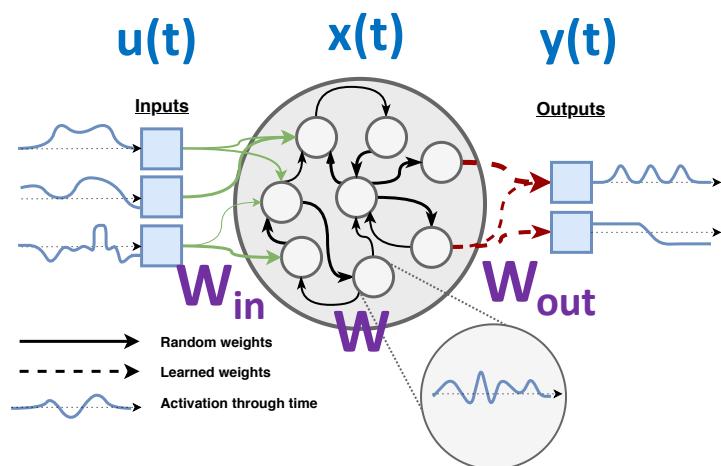
NE REGARDEZ PAS LA PROCHAINE DIAPO !

Reservoir Computing: Echo State Network (ESN)

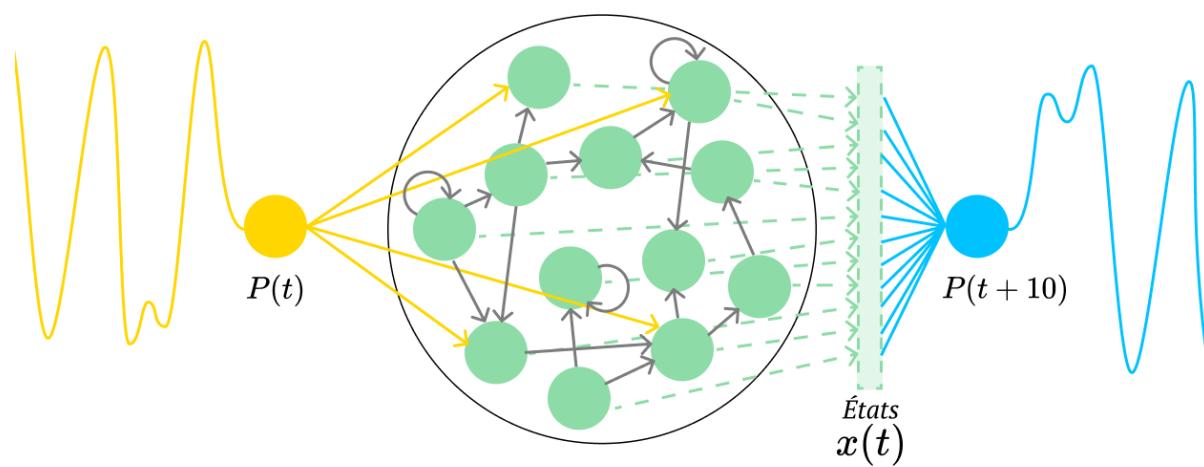
Reservoir state update

$$x(t) = \left(1 - \frac{1}{\tau}\right)x(t-1) + \frac{1}{\tau}f(W^{in}u(t) + Wx(t-1))$$

Output (« read-out ») update

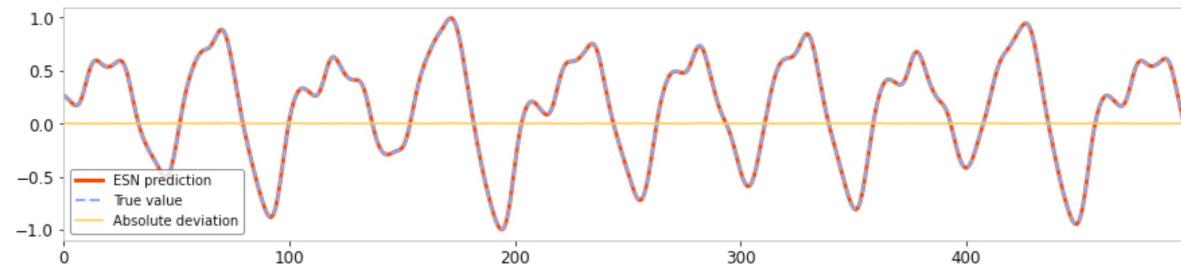


Prédiction de séries temporelles

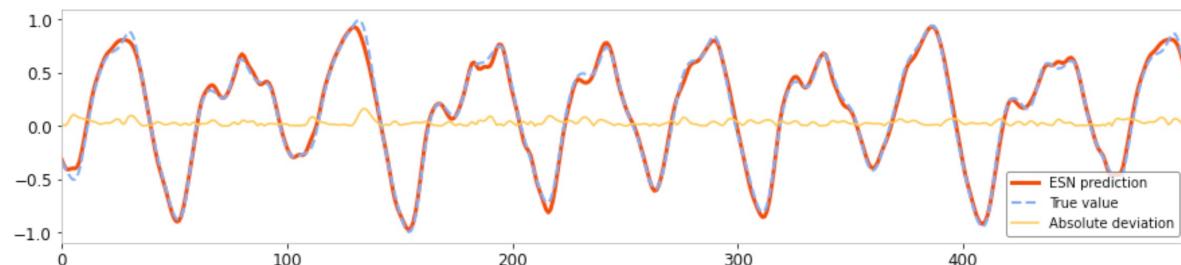


Prédiction de séries temporelles

- Ex : série chaotique de Mackey & Glass
 - Prédictions obtenues pour 10 pas de temps dans le futur ($T+10$)



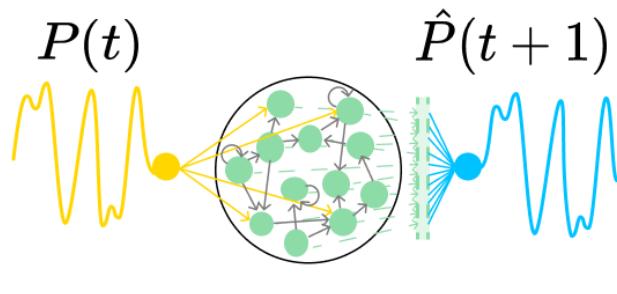
- Prédictions obtenues pour 50 pas de temps dans le futur ($T+50$)



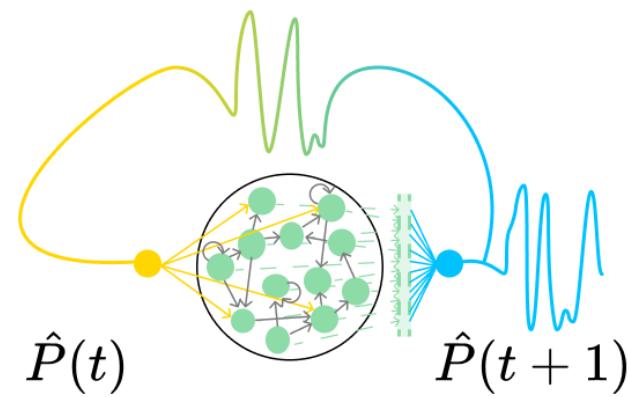
Exemple tiré de github.com/reservoirpy/reservoirpy

Génération de séries temporelles

Génération de séquences en « faisant boucler » la sortie sur les entrées
= Mode « Autorégressif » comme ChatGPT



Mode Prédiction

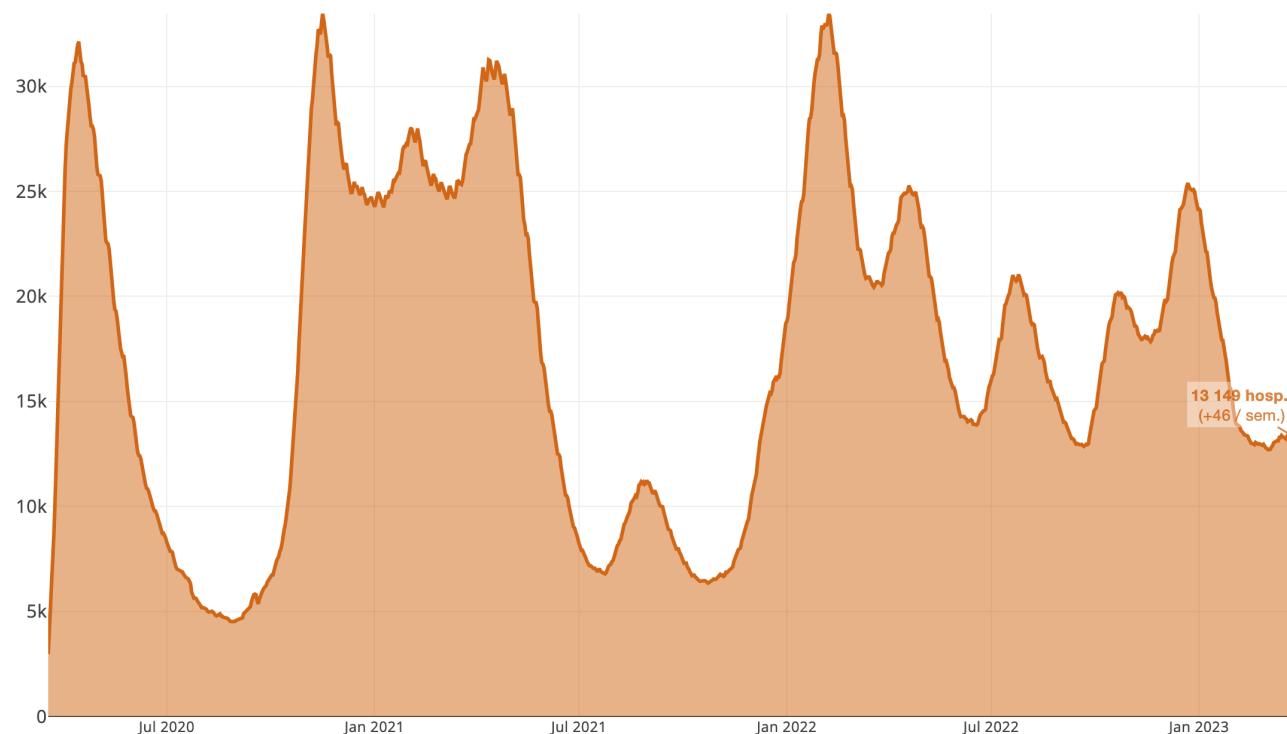


Mode Génération



Prédiction hospitalisation COVID à t+14

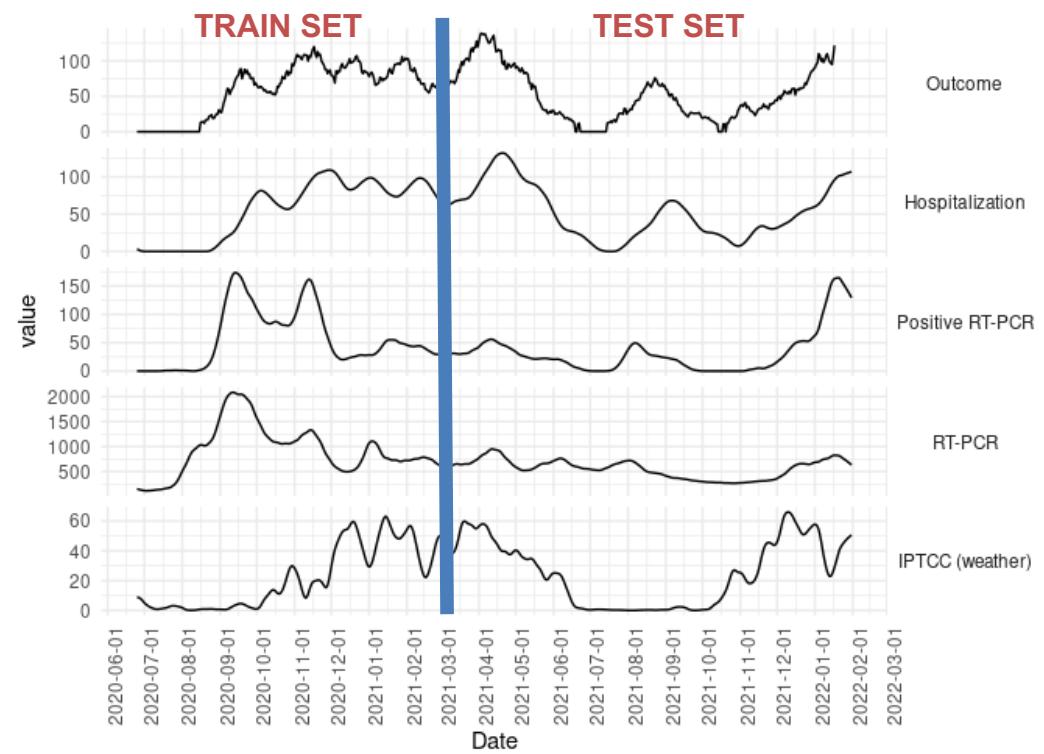
Nombre de personnes hospitalisées
Nombre de personnes hospitalisées avec Covid19.



CovidTracker.fr • Données : Santé publique France • Dernière donnée : 31 / 03 / 2023

Reservoir Computing + Algorithme Génétique

- **Données** : Données agrégées provenant des dossiers médicaux électroniques + SARS-CoV-2 publics + météo = 409 *features* sur 586 jours.
- **Haute dimension** : 409 *features* \simeq nombre de variables observées chaque jour.
- **Résultat** : Hospitalisations à 14 jours au CHU de Bordeaux.
- **Métrique** : Erreur Absolue Moyenne (MAE)
- **Modèle** : Reservoir Computing (médiane de prédictions de 40 réservoirs)



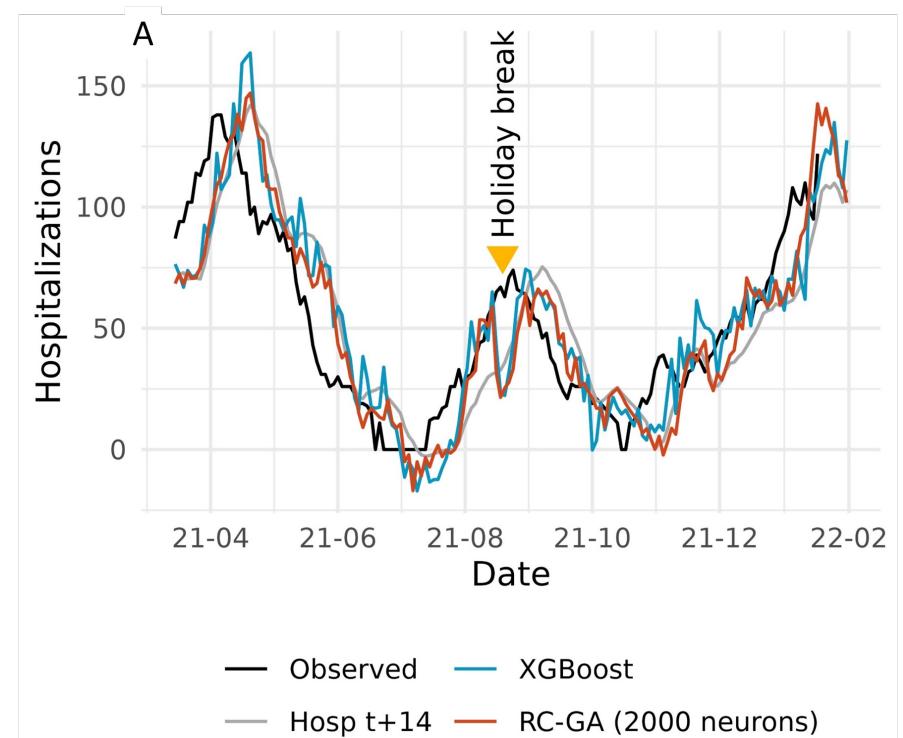
Prédiction hospitalisation COVID à t+14

Model	Mean Absolute Error (\pm std)
RC-GA (500 neurons)	15.27 (\pm 12.88)
RC-GA (2000 neurons)	14.66 (\pm 12.63)
RC-RS (500 neurons)	18.45 (\pm 13.27)
RC-GA-no Feature selection (500 neurons)	17.59 (\pm 13.35)
Elastic-net	15.83 (\pm 12.41)
XGBoost	15.45 (\pm 13.91)
LSTM-PCA	15.74 (\pm 13.98)
Transformers-PCA	19.28 (\pm 16.80)
Informer-PCA	18.70 (\pm 14.11)
PatchTST-PCA	18.34 (\pm 18.14)
Prophet	21.30 (\pm 16.33)

Notre approche

Prédiction hospitalisation COVID à t+14

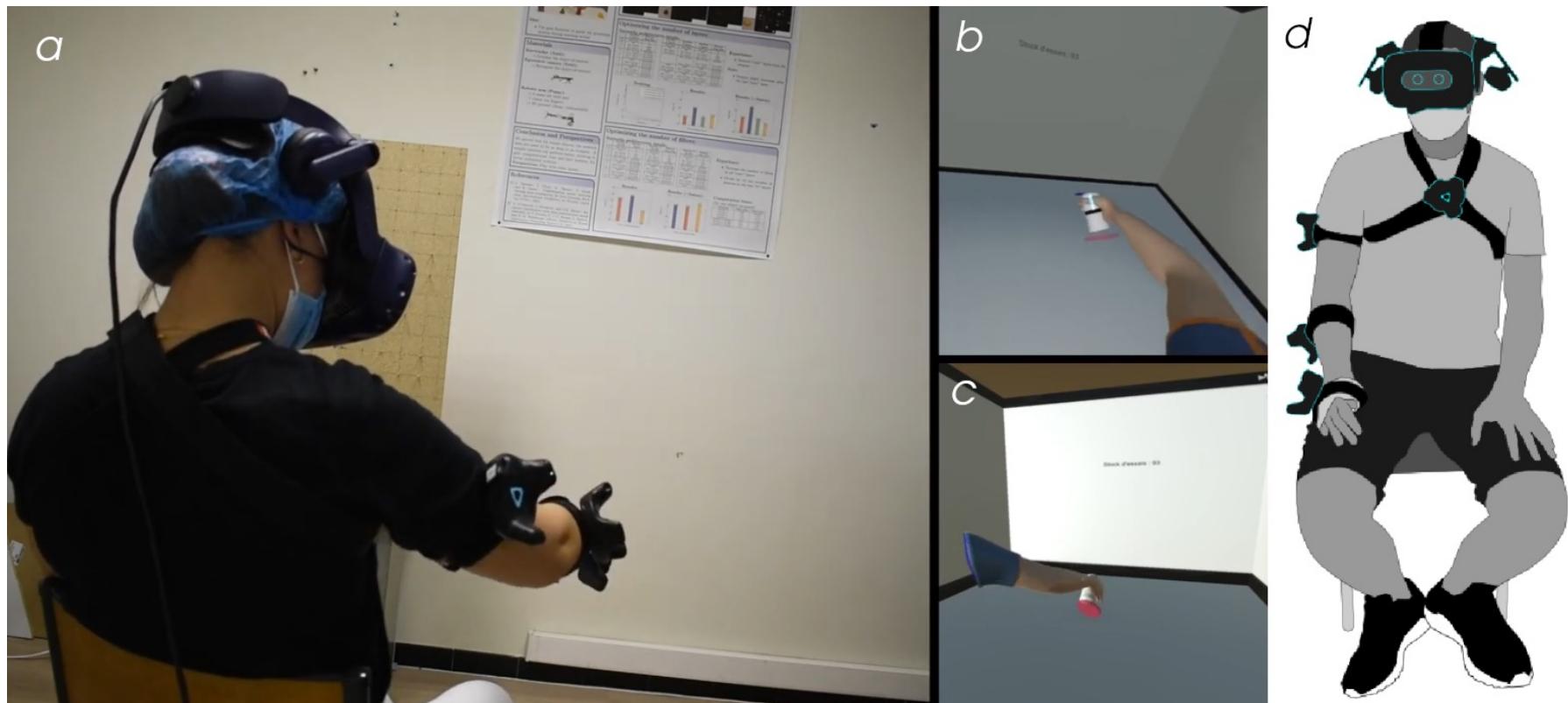
Model	Mean Absolute Error (\pm std)
RC-GA (500 neurons)	15.27 (\pm 12.88)
RC-GA (2000 neurons)	14.66 (\pm 12.63)
RC-RS (500 neurons)	18.45 (\pm 13.27)
RC-GA-no Feature selection (500 neurons)	17.59 (\pm 13.35)
Elastic-net	15.83 (\pm 12.41)
XGBoost	15.45 (\pm 13.91)
LSTM-PCA	15.74 (\pm 13.98)
Transformers-PCA	19.28 (\pm 16.80)
Informer-PCA	18.70 (\pm 14.11)
PatchTST-PCA	18.34 (\pm 18.14)
Prophet	21.30 (\pm 16.33)



EXEMPLE D'APPLICATION

CONTROLE DE PROTHÈSE BIOMIMÉTIQUE

Controle de prothèse biomimétique

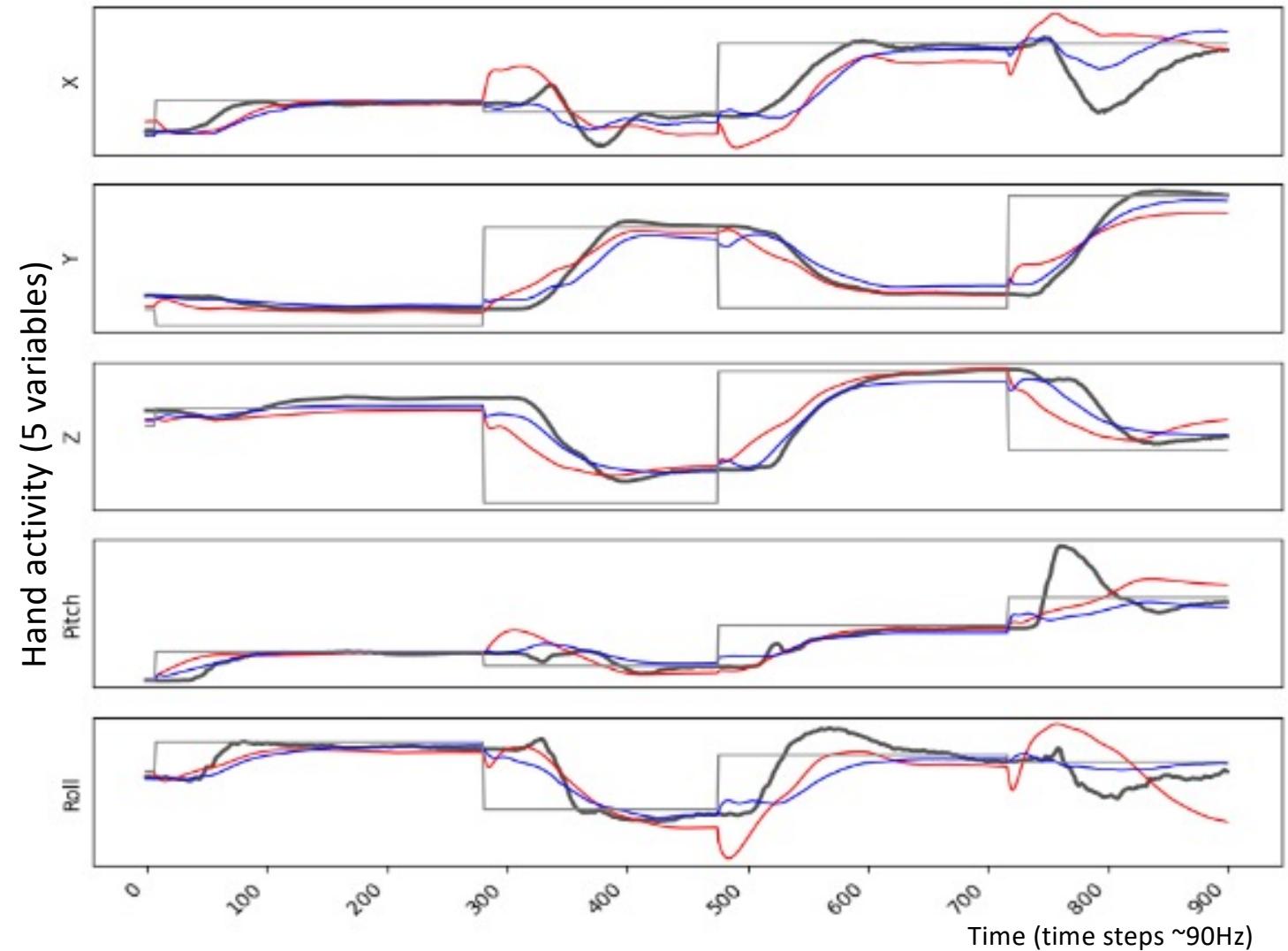
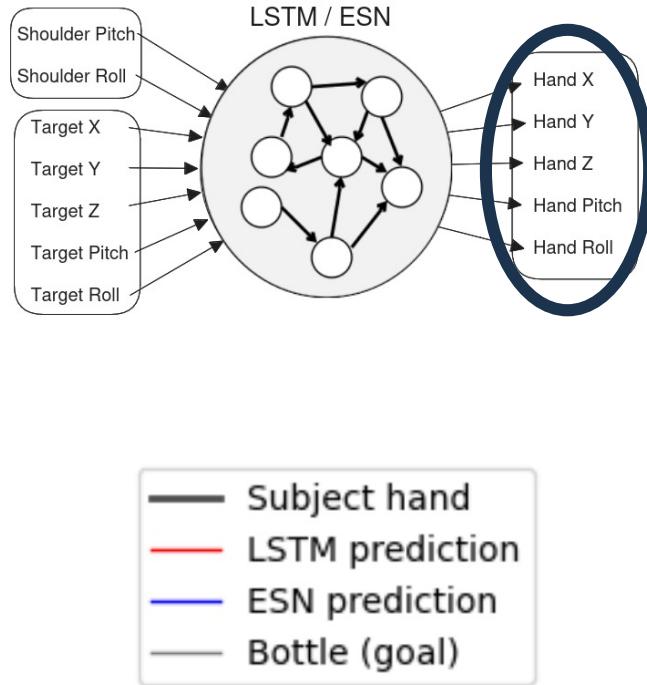


Segas, 2023

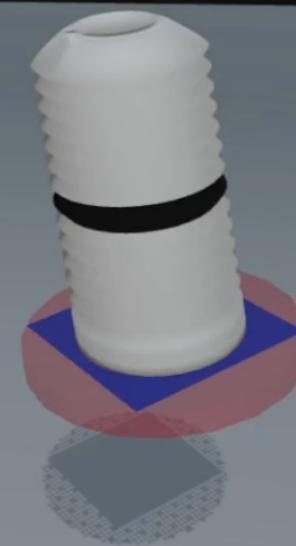


P. Bernard - ICANN 2024

Résultats de l'apprentissage sur un seul sujet



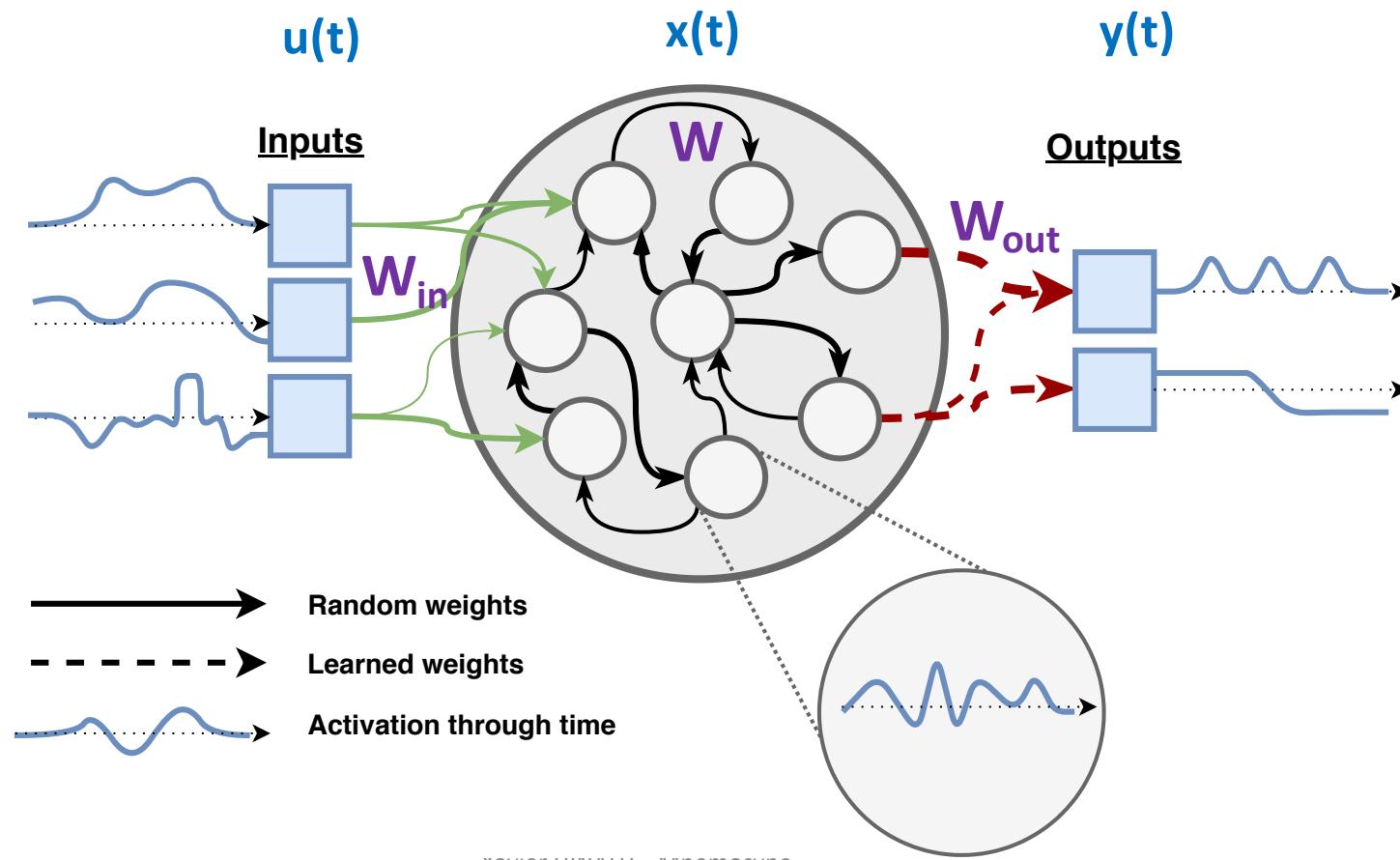
Stock d'essais: 49



Plongeons dans les profondeurs du Reservoir Computing



Reservoir Computing



Pourquoi est-ce que ça marche ?



Pourquoi est-ce que ça marche ?

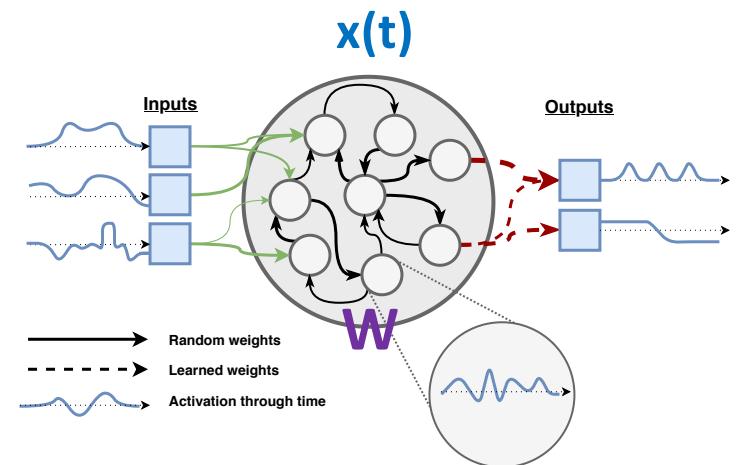


Pourquoi est-ce que ça marche ?



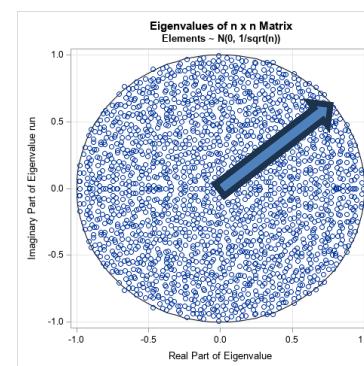
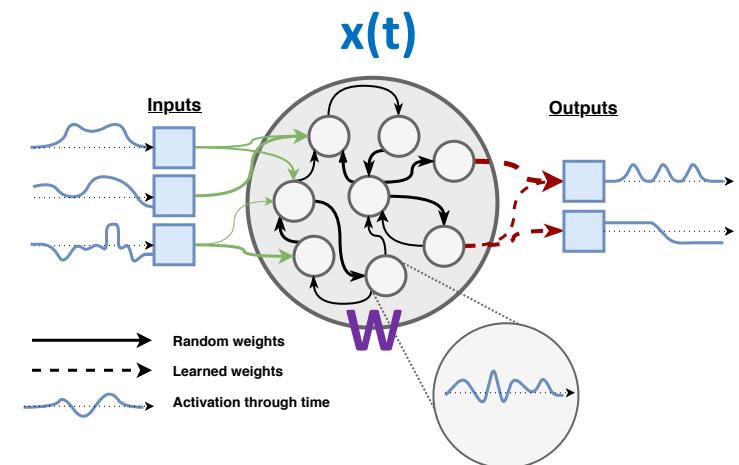
Spectral Radius (rayon spectral)

= scaling des connexions récurrentes



Spectral Radius (rayon spectral)

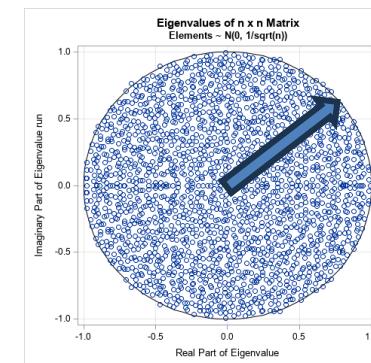
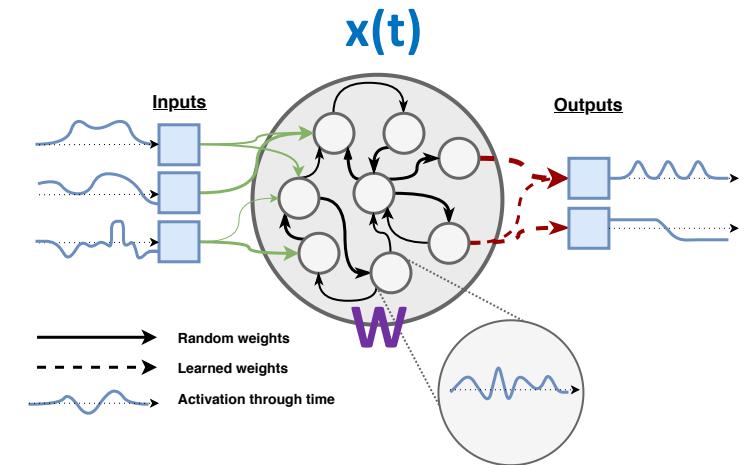
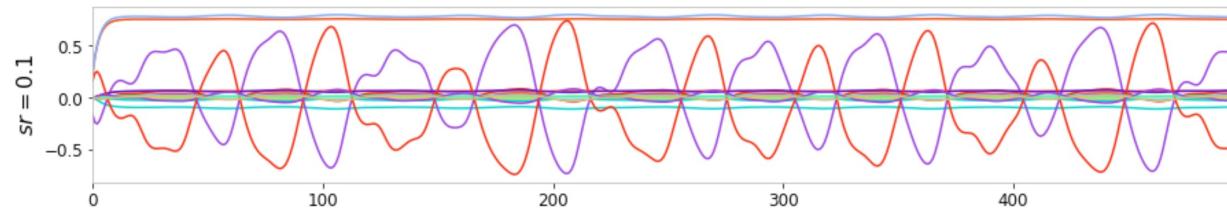
= scaling des connexions récurrentes



Rayon spectral de la matrice W

Spectral Radius (rayon spectral)

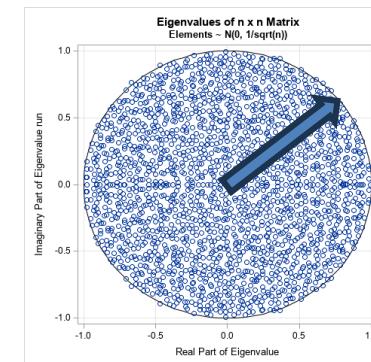
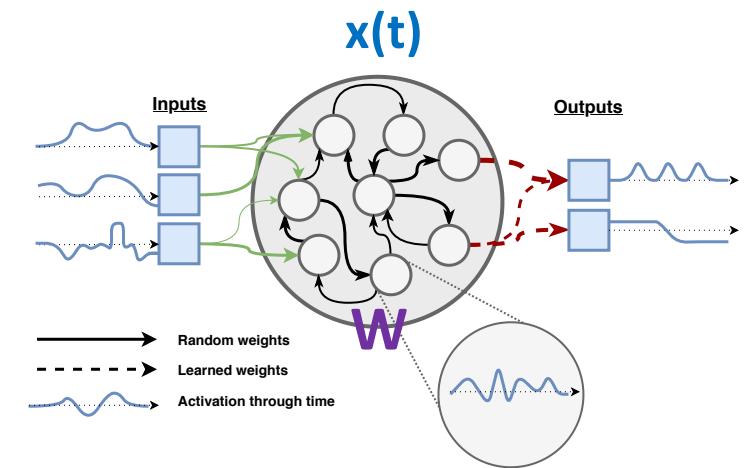
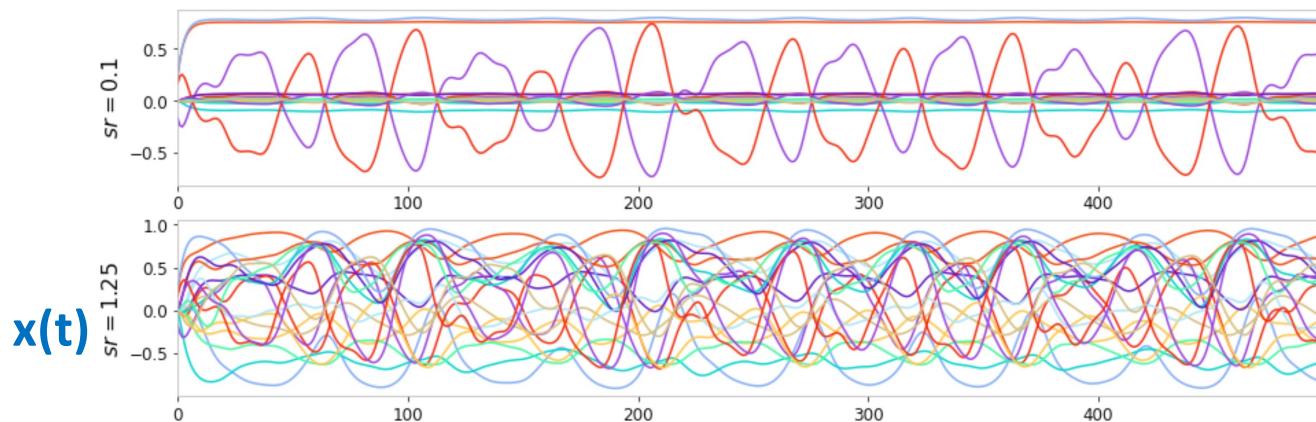
= scaling des connexions récurrentes



Rayon spectral de la matrice W

Spectral Radius (rayon spectral)

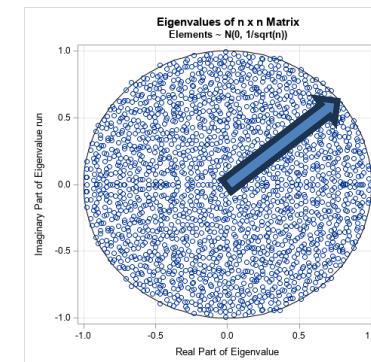
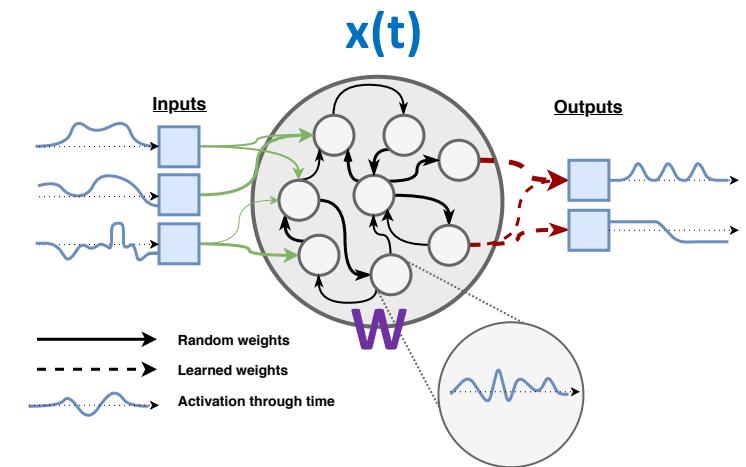
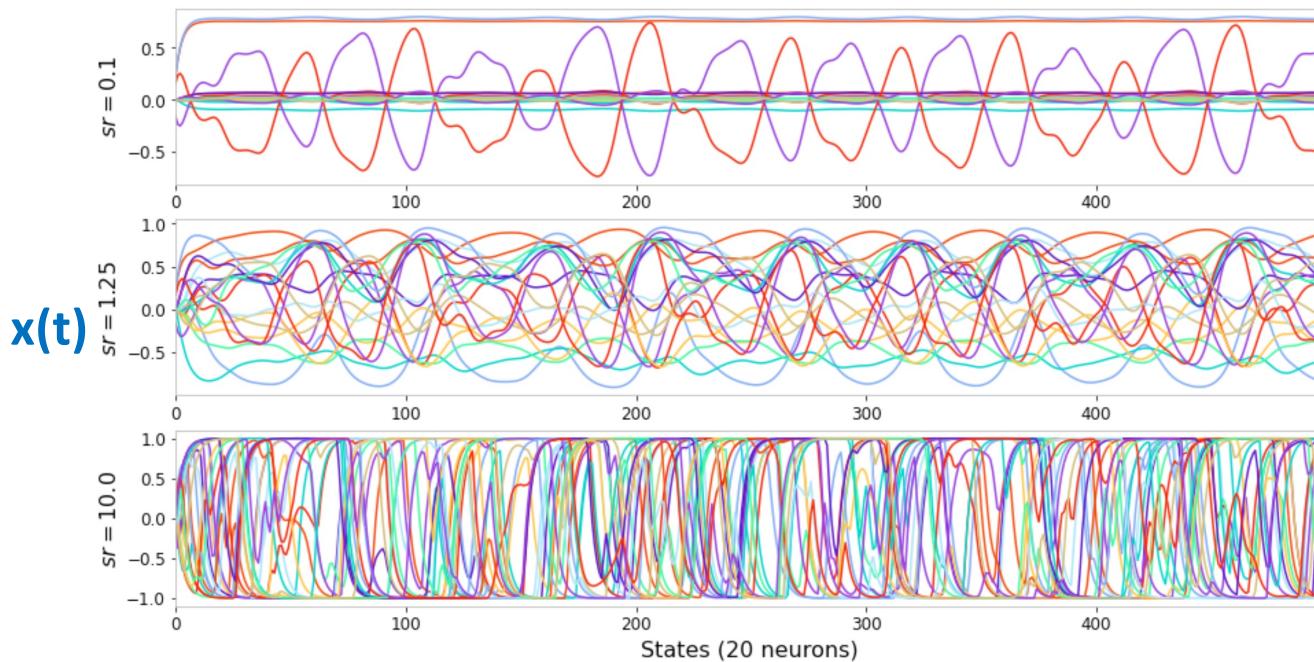
= scaling des connexions récurrentes



Rayon spectral de la matrice W

Spectral Radius (rayon spectral)

= scaling des connexions récurrentes

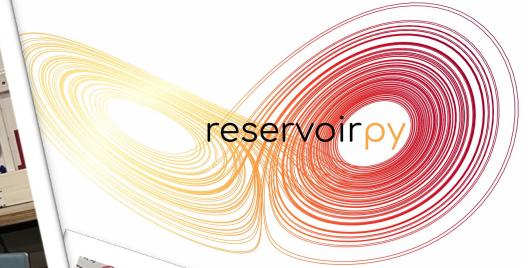
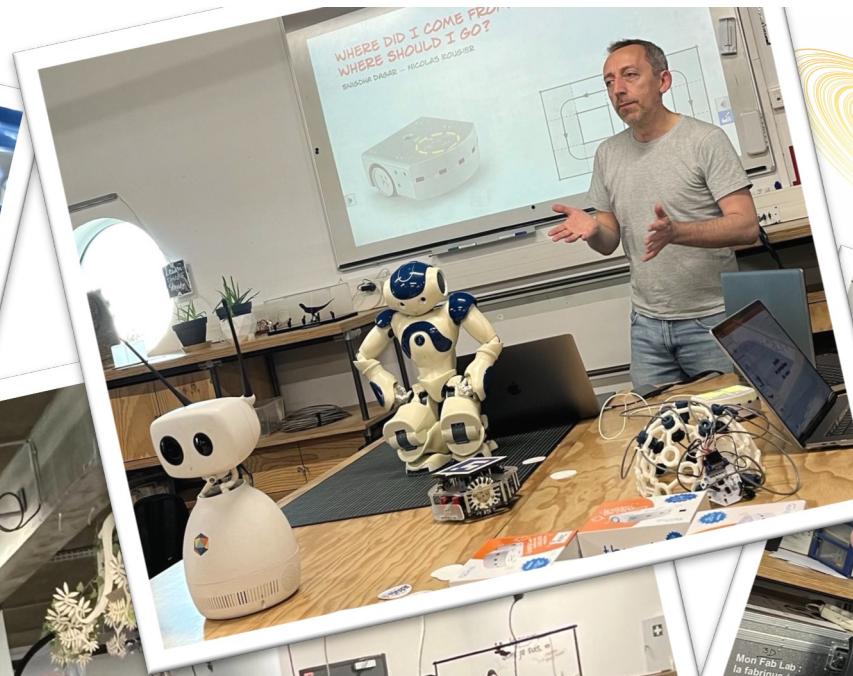


Rayon spectral de la matrice W

RESERVOIRPY ET EXEMPLE D'APPLICATION

GÉNÉRATION DE MUSIQUE + INFLUENCE DES HYPERPARAMÈTRES







github.com/reservoirpy/reservoirpy

reservoirpy / reservoirpy

Code Issues Pull requests Discussions Actions Projects Wiki

reservoirpy Public Edit Pins Unwatch Fork Starred

v0.3.11 Go to file + Code

README MIT license

python 3.8 | 3.9 | 3.10 | 3.11 | 3.12 | 3.13 pypi package 0.3.12 docs passing Testing passing codecov 93%
downloads 1.9k/month downloads 57k

ReservoirPy (v0.3.12) 🤖

Simple and flexible code for Reservoir Computing architectures like Echo State Networks (ESN).

launch binder

🎉 Exciting News! We just launched a new beta tool based on a Large Language Model! 🚀 You can chat with our "ReservoirChat" and ask anything about Reservoir Computing or coding reservoirs! 🤖💡 Don't miss out, it's available for a limited time! 🕒 <https://chat.reservoirpy.inria.fr>

```
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)
```

banner_bw_small-size.jpg >> reservoir >> readout

About

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

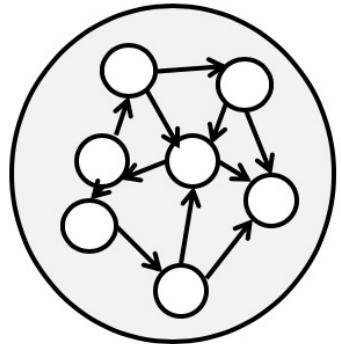
Readme MIT license Activity Custom properties 449 stars 17 watching 110 forks Report repository

Releases 26

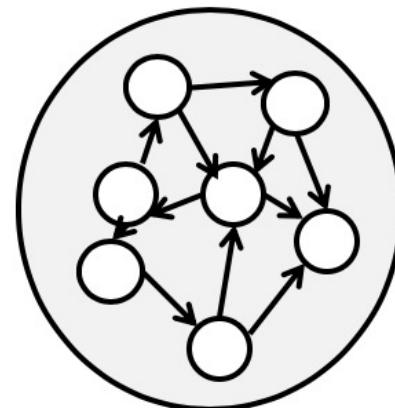
ReservoirPy v0.3.12 (Latest) on Nov 12, 2024 + 25 releases

Avez-vous déjà rêvé d'être DJ ?

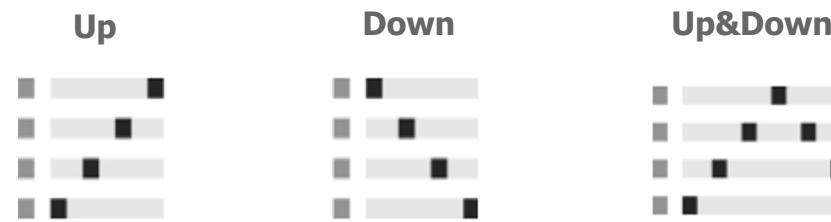




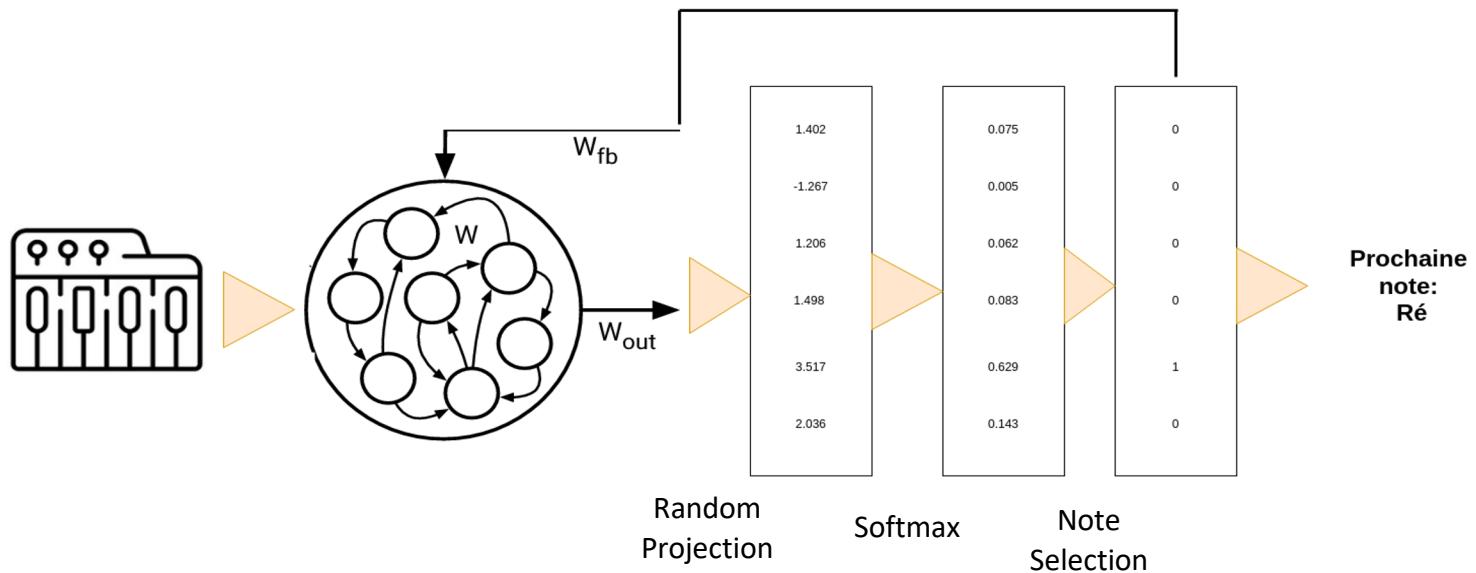
Avez-vous déjà rêvé de jouer d'un réservoir ?

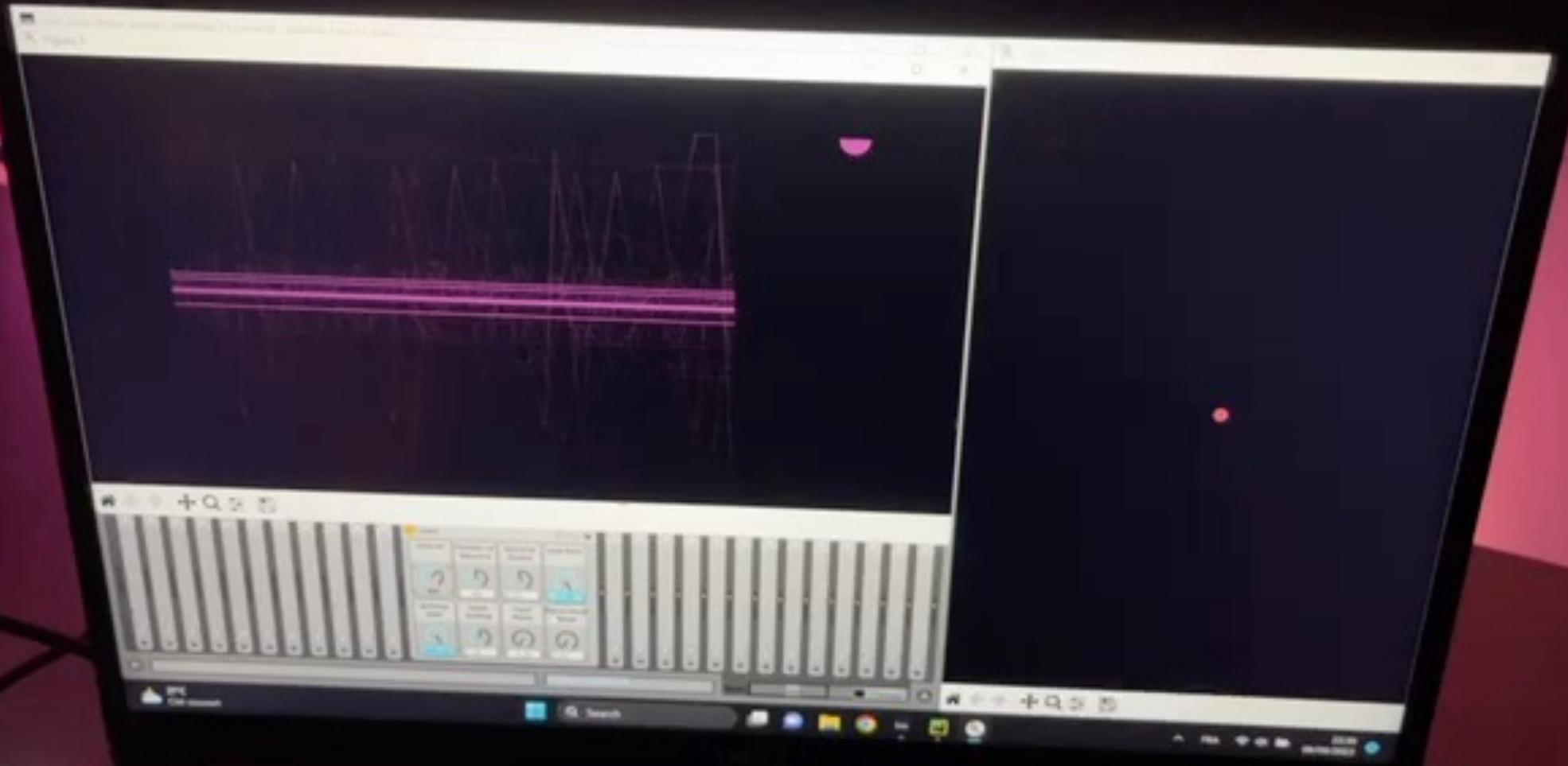


Classical arpegiators

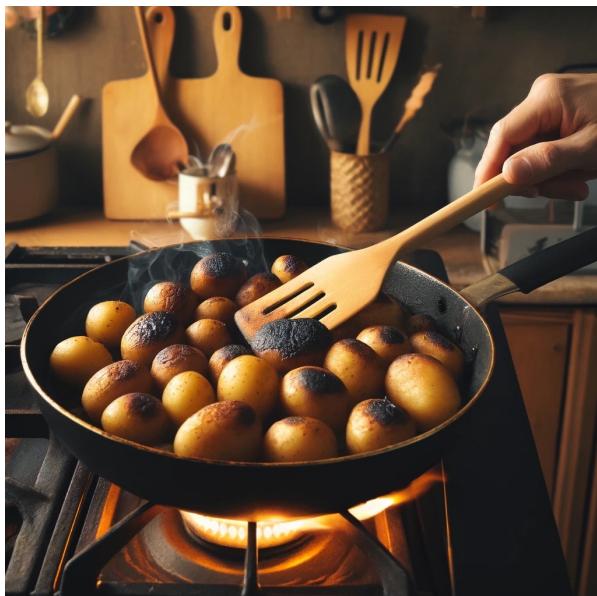


“RéMi”: the Reservoir MIDI



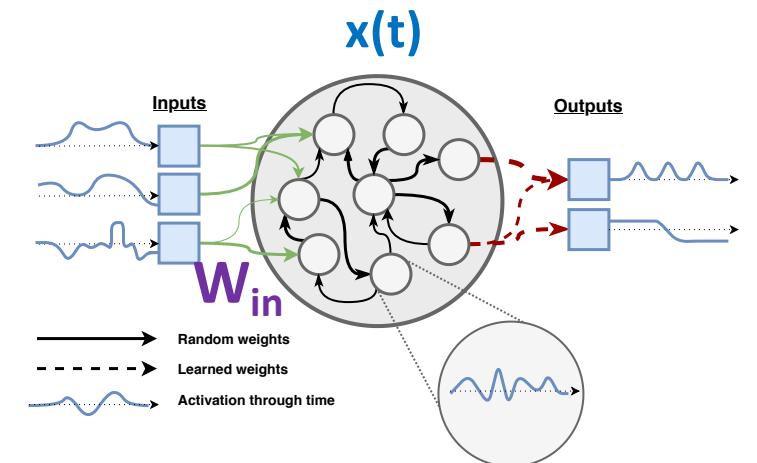
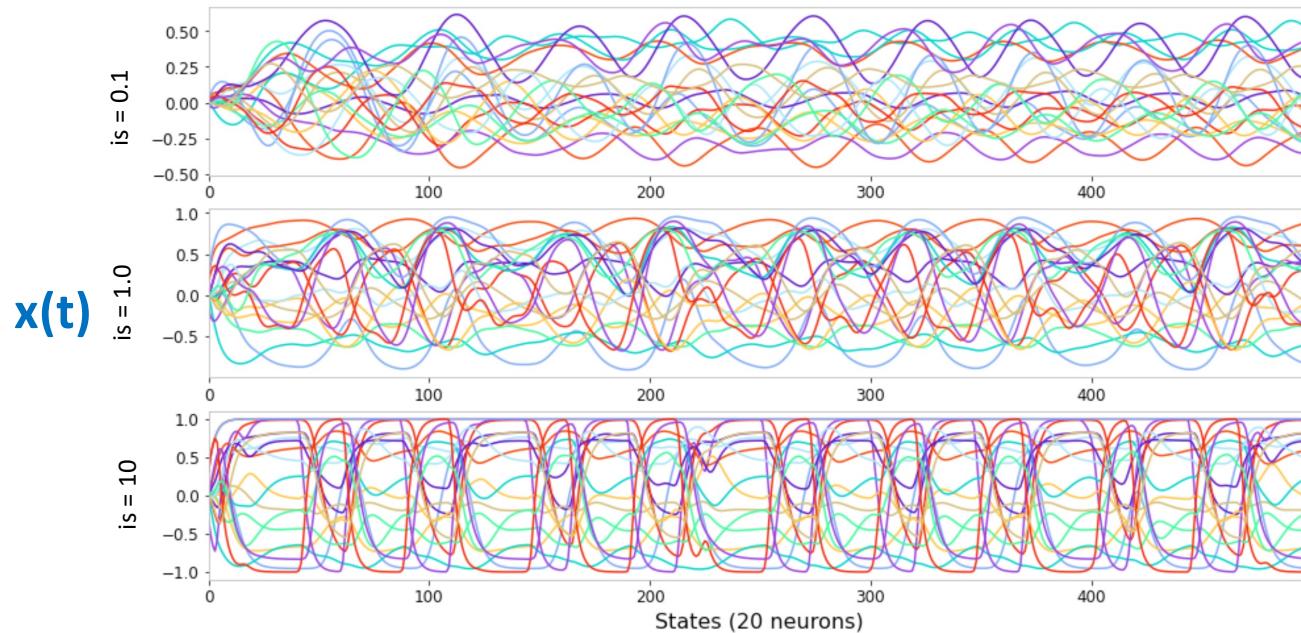


En pratique ? (Hypster-paramètres)



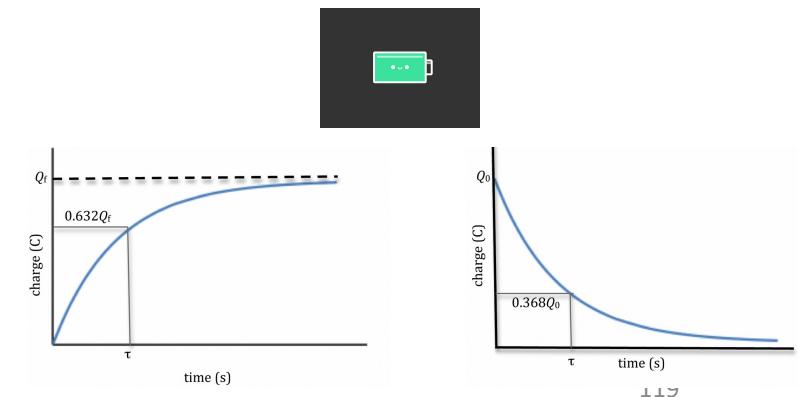
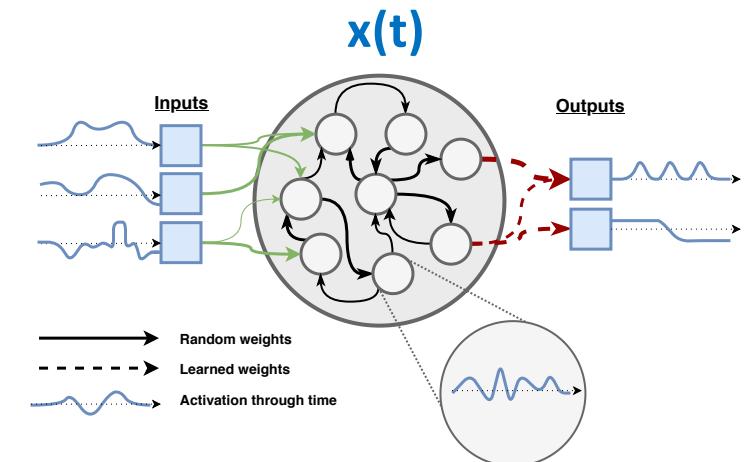
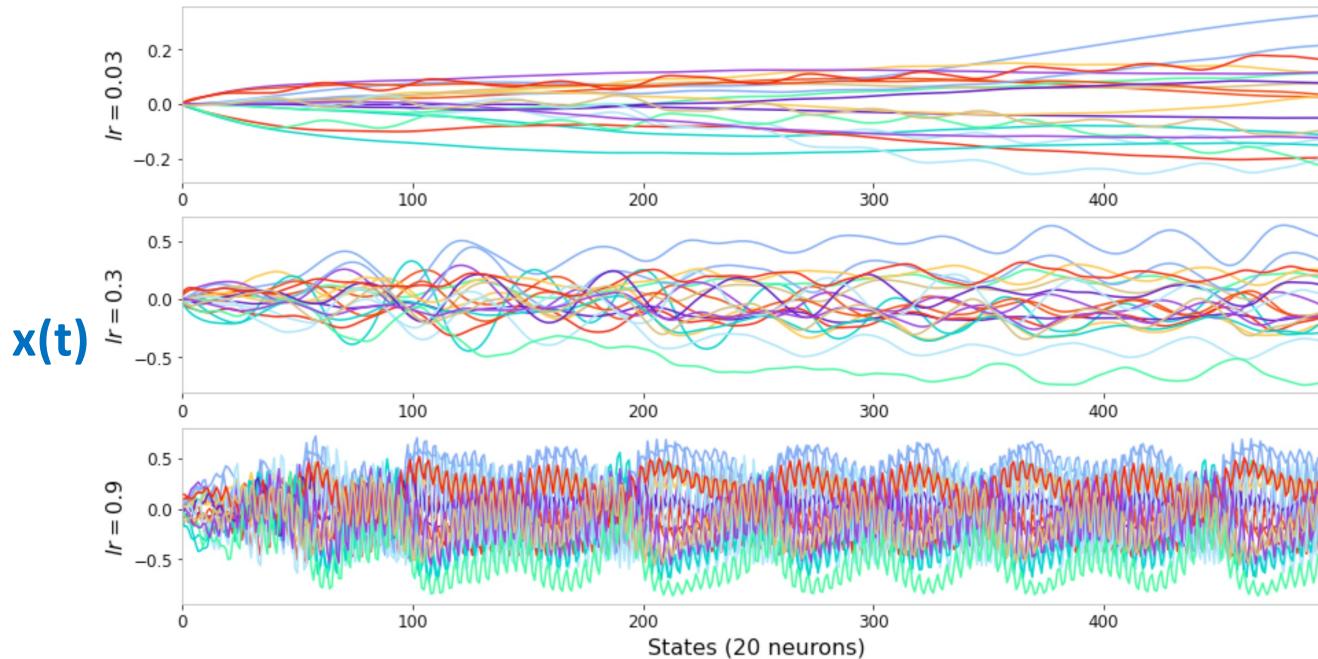
Input Scaling

= scaling des connexions entrantes



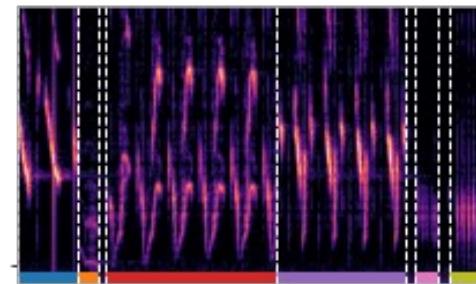
Leak-rate (=taux de fuite)

= inverse de la constante de temps des neurones

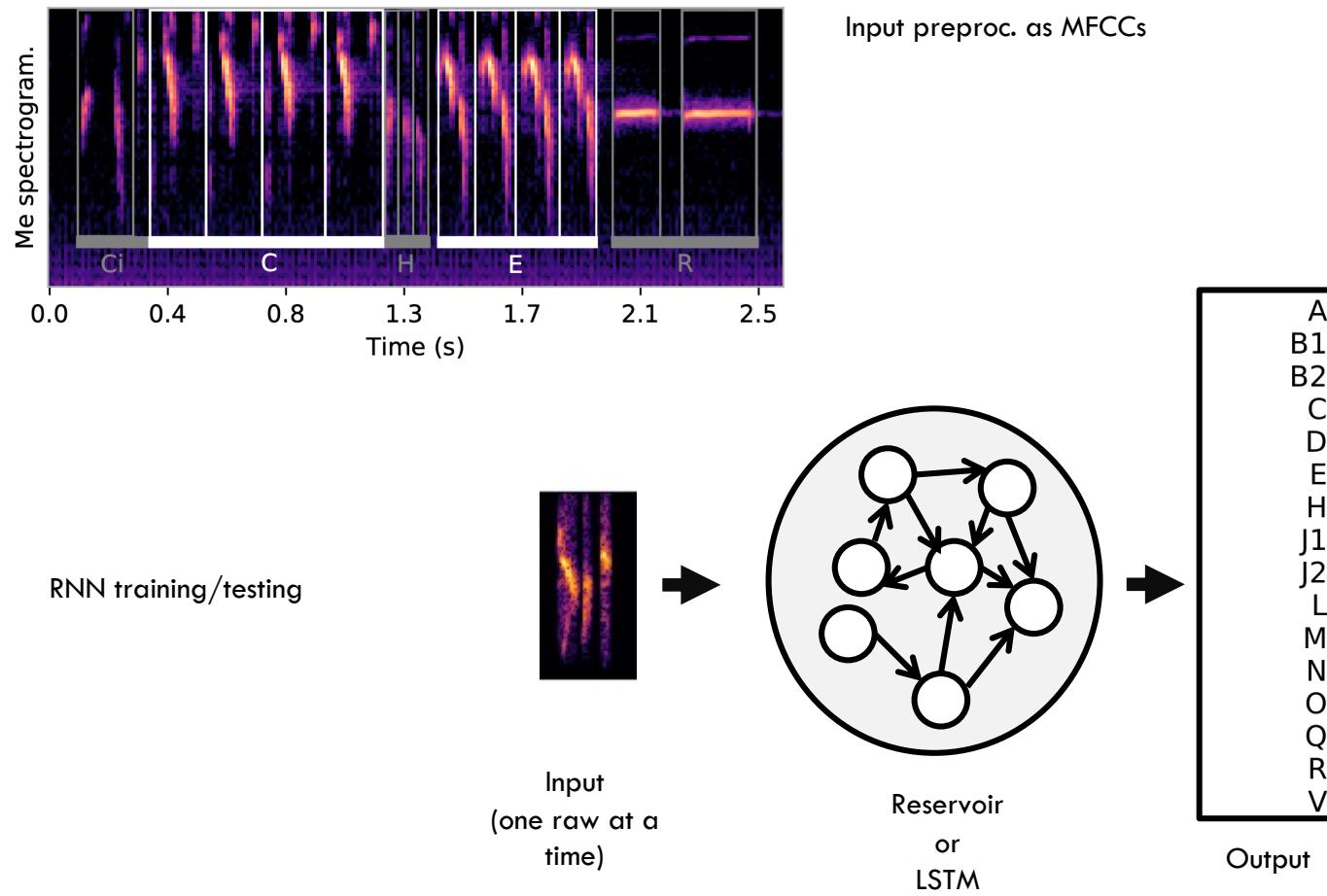


EXEMPLE D'APPLICATION

DISCRIMINATION CHANT CANARIS RESERVOIR VS. LSTM

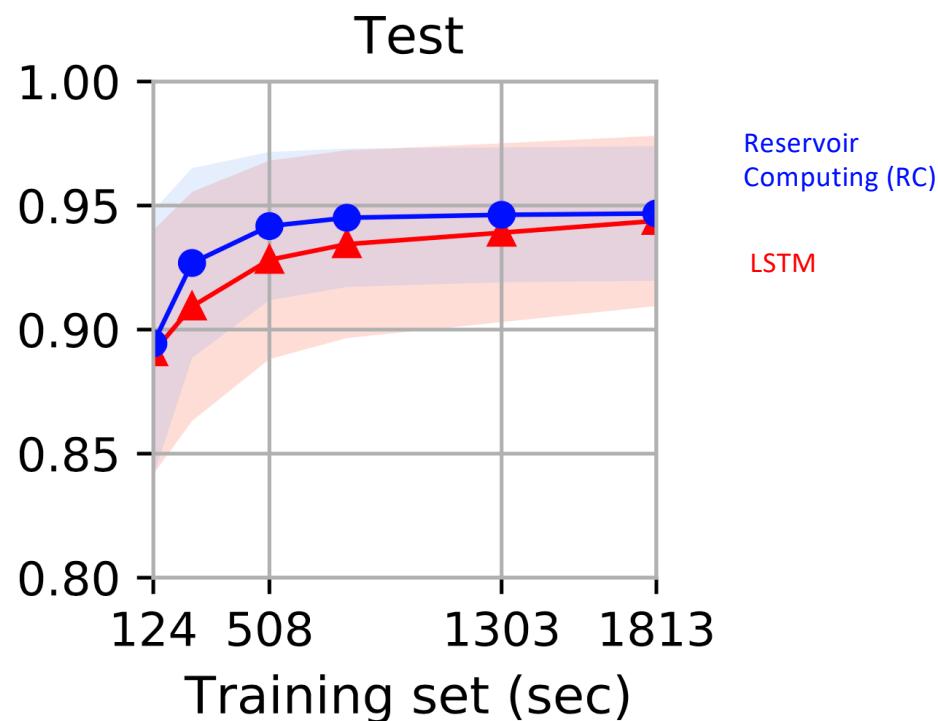


Discrimination de phrases de canaris



Discrimination de phrases de canaris

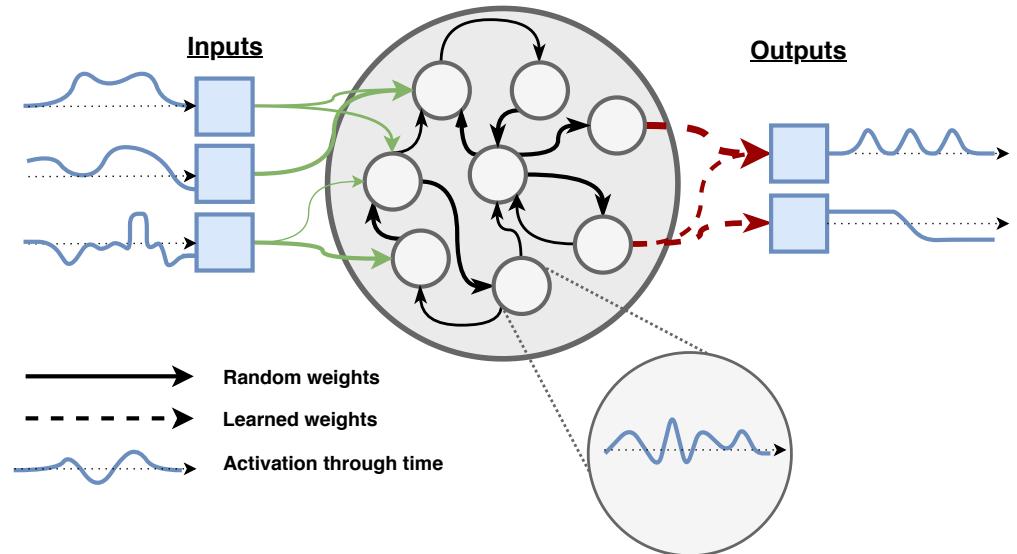
Taux de précision vs. taille des données d'entraînement



Temps moyen pour l'entraînement sur CPU (10-fold CV)

Model	Average training time (s)
LSTM	2930 ± 222
RC	35 ± 1







Choisissez le bon outil pour chaque problème



Si on a un marteau
tous les problèmes
sont des clous !

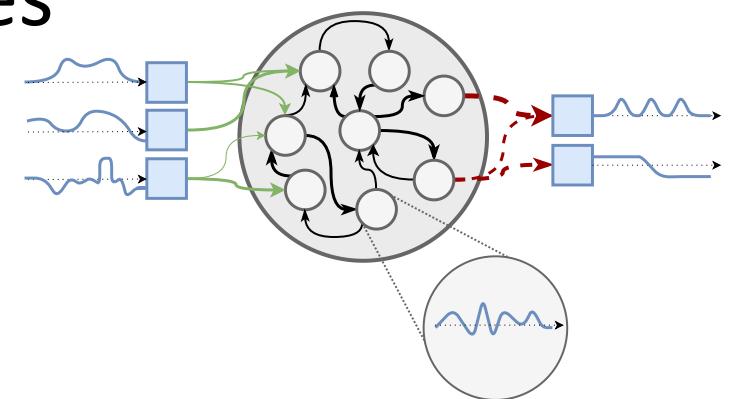
En Résumé

- Le **Reservoir Computing** permet de
 - Prédire / Générer / Classifier / Simuler
 - Entraînement *online* et *offline*
 - → données temporelles (et non-temporelles)
 - Se combine avec CNN, RL, ...
- **ReservoirPy** propose une IA frugale
 - Bonnes performances même avec **peu de données nécessaires**
 - Economie de calculs / temps
 - → Economie d'énergie !

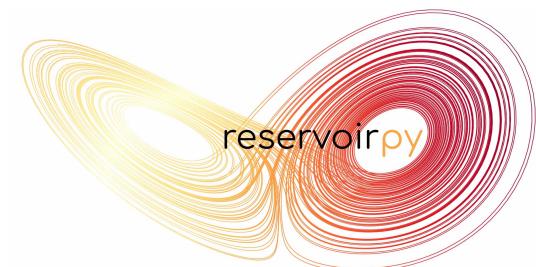
- Transférable sur différents types de *hardware*
 - systèmes physiques optiques, magnétiques, biologiques, etc.



Les **reseaux aléatoires** peuvent
générer des dynamiques fascinantes



Le **reservoir computing** exploite
les **dynamiques naturelles** d'un réseau aléatoire



Les réseaux aléatoires peuvent générer des dynamiques fascinantes

- Ça ne coûte pas cher !



- Et ça permet d'avoir une grande palette



Le reservoir computing exploite les dynamiques naturelles d'un réseau aléatoire



C'est juste une question de bonnes dynamiques !



github.com/reservoirpy/reservoirpy

reservoirpy / reservoirpy

Code Issues Pull requests Discussions Actions Projects Wiki

reservoirpy Public Edit Pins Unwatch Fork Starred 457

v0.3.11 Go to file + Code

README MIT license

python 3.8 | 3.9 | 3.10 | 3.11 | 3.12 | 3.13 pypi package 0.3.12 docs passing Testing passing codecov 93%
downloads 1.9k/month downloads 57k

ReservoirPy (v0.3.12) 🧠

Simple and flexible code for Reservoir Computing architectures like Echo State Networks (ESN).

launch binder

🎉 Exciting News! We just launched a new beta tool based on a Large Language Model! 💫 You can chat with our "ReservoirChat" and ask anything about Reservoir Computing or coding reservoirs! 🤖💡 Don't miss out, it's available for a limited time! 🕒 <https://chat.reservoirpy.inria.fr>

```
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)
```

banner_bw_small-size.jpg >> reservoir >> readout

About

A simple and flexible code to 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

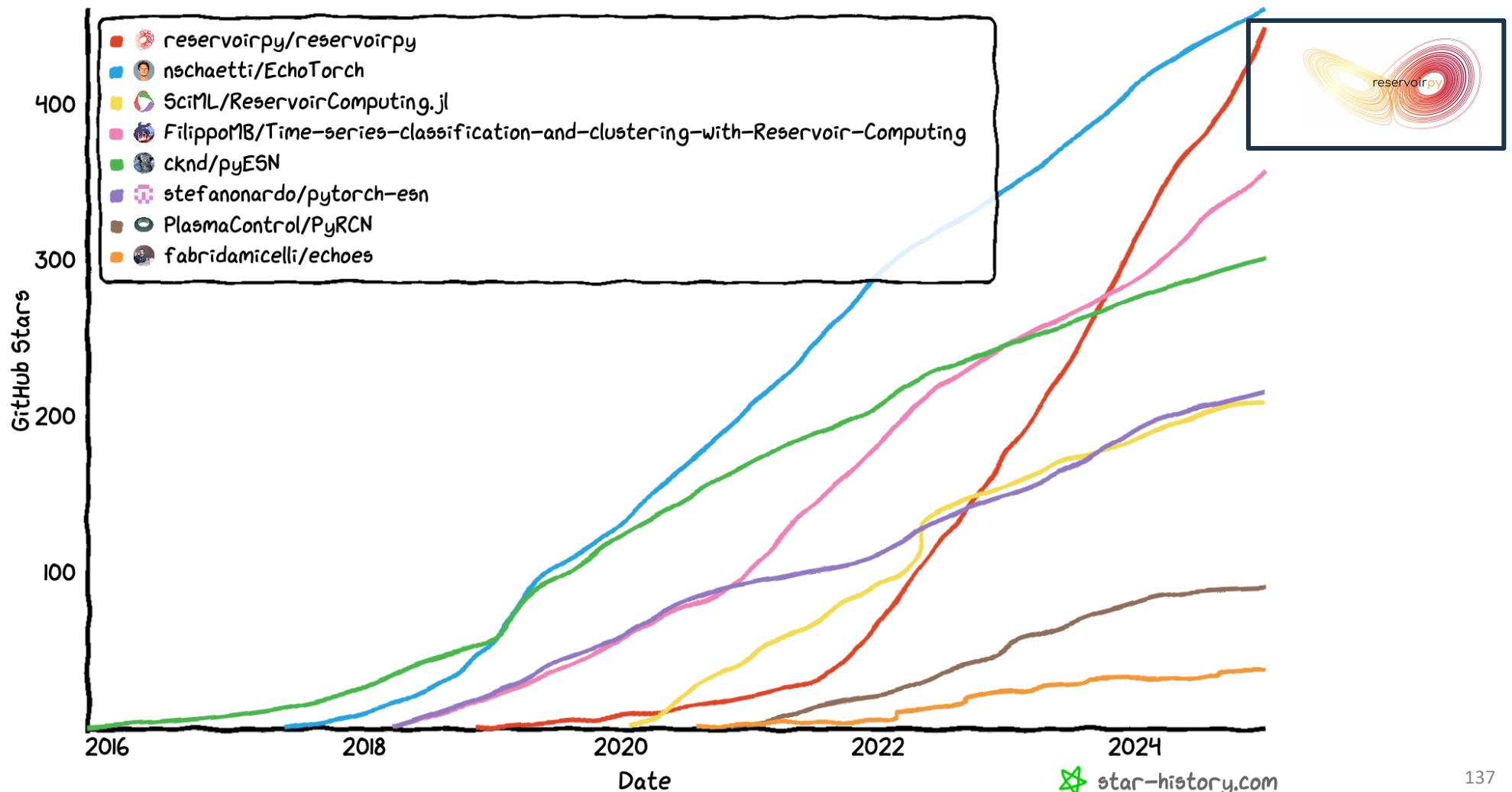
Readme MIT license Activity Custom properties 449 stars 17 watching 110 forks Report repository

Releases 26

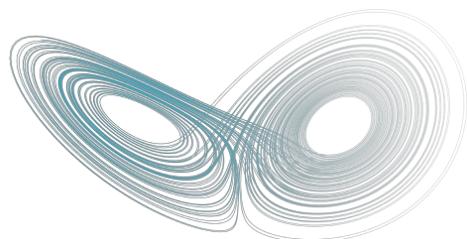
ReservoirPy v0.3.12 (Latest) on Nov 12, 2024 + 25 releases

A purple arrow points to the 'Starred' button in the top right corner of the GitHub interface.

Star History



ReservoirChat



chat.reservoirpy.inria.fr

ReservoirChat



Inria

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

 Send

 Rerun

 Undo

 Clear

Merci pour votre attention !

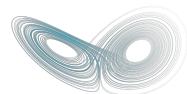
Ingénieurs dév. ReservoirPy

Actuel : Paul Bernard

Précédent : Nathan Trouvain



github.com/reservoirpy/reservoirpy

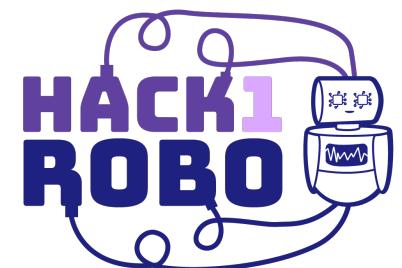


chat.reservoirpy.inria.fr

xavier.hinaut@inria.fr



@neuronalX



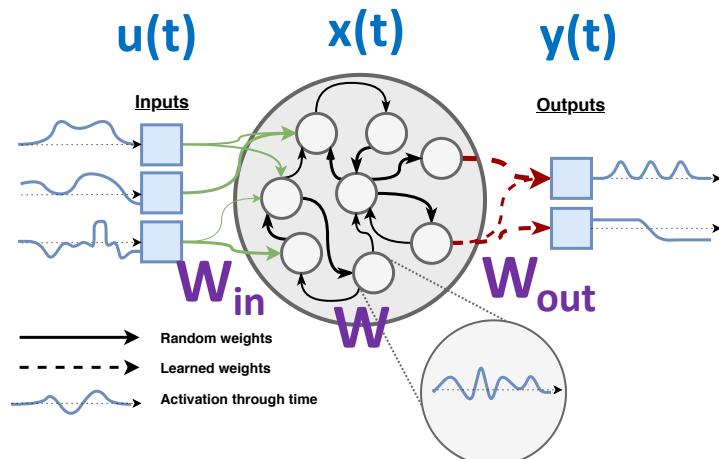
Reservoir Computing: Echo State Network (ESN)

Reservoir state update

$$x(t) = \left(1 - \frac{1}{\tau}\right)x(t-1) + \frac{1}{\tau}f(W^{in}u(t) + Wx(t-1))$$

Output (« read-out ») update

$$y(t) = W^{out}x(t)$$



- $u(t)$: inputs
- W^{in}, W, W^{out} : input, recurrent, output matrices
- W^{in} and W matrices are kept random
- Only W^{out} is trained (e.g. ridge regression)
- τ : time constant of reservoir units
 - *leak rate (LR)* is often used instead of $\frac{1}{\tau}$
- f : activation function (usually $tanh$)

En pratique ? (Hypster-paramètres)

- Les **hyperparamètres** importants à régler
 - nombre de neurones
 - Input scaling
 - Spectral radius (= rayon spectral)
 - (Feedback scaling si présence de feedback)
 - Leak-rate (= taux de fuite = 1 / constance de temps)
 - Ridge (régularisation de la régression)
 - Warm-up (= « temps de chauffe »)

→ Voir le tutoriel 4 de ReservoirPy sur GitHub

[reservoirpy / tutorials / 4-Understand_and_optimize_hyperparameters.ipynb](#)

VirgileBoraud and PAUL-BERNARD Correction of some minor mistakes on the tutorials (#161) 78b89al

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 networks.

It also present a very basic example of optimization using `hyperopt` and `reservoirpy.hyper` tools.

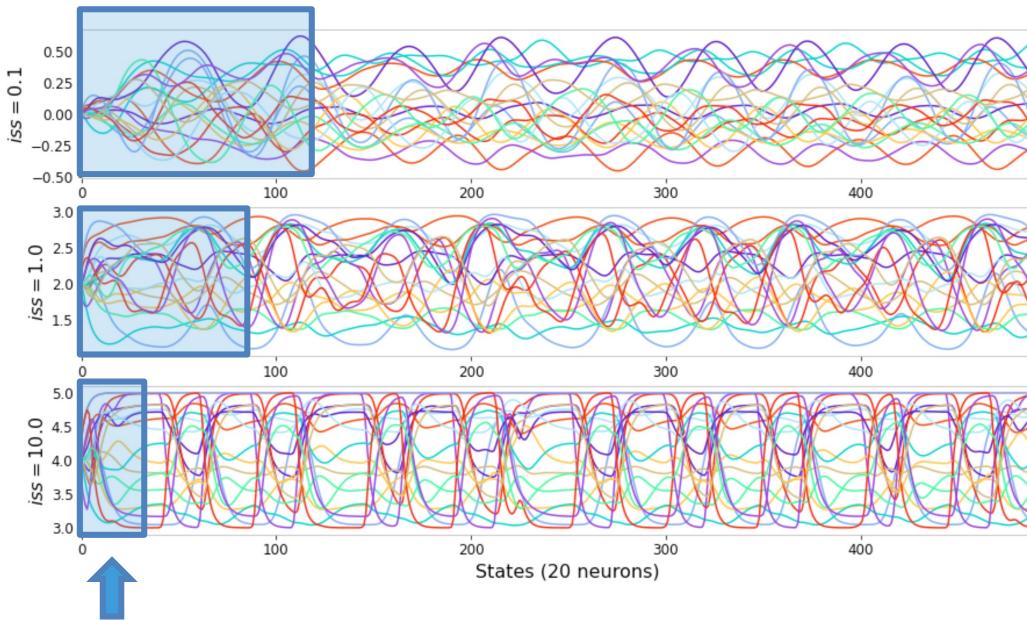
Understand ESN hyperparameters

```
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
```

https://github.com/reservoirpy/reservoirpy/blob/master/tutorials/4-Understand_and_optimize_hyperparameters.ipynb

« Warming up »

warmup
ou
warming up

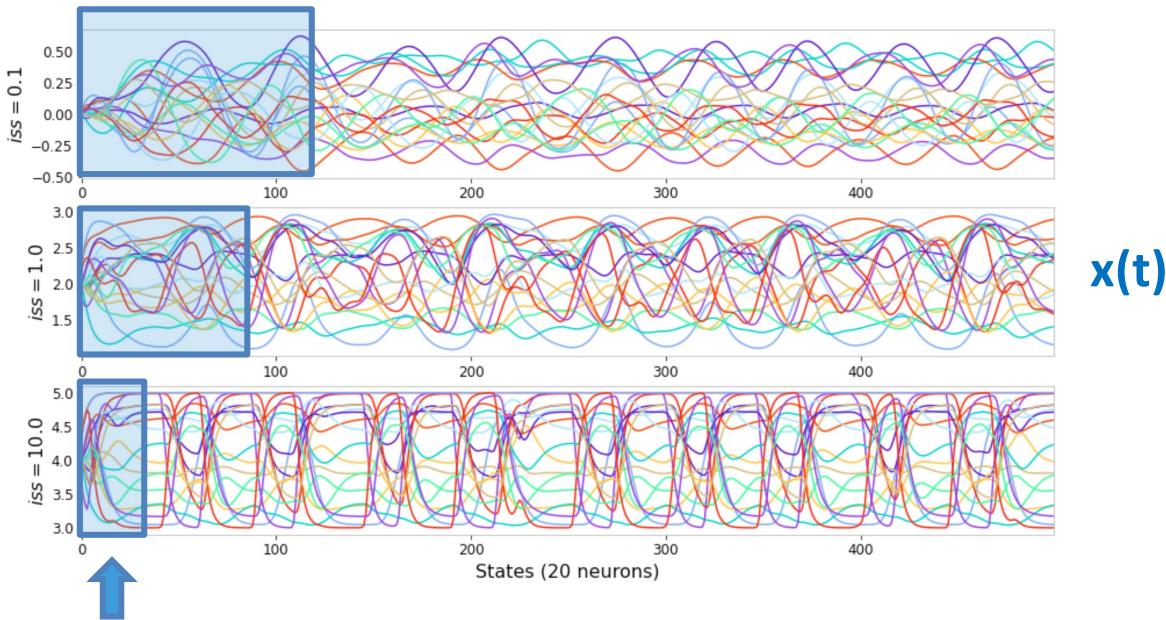


$x(t)$

- Le *warmup* est le « temps de chauffe » nécessaire pour que les dynamiques soient dans un rythme de « croisière ».
 - = zone bleue sur chaque figure
- Cela correspond au temps nécessaire pour que le réseau ne fasse plus vraiment la différence entre le début ou le milieu de la série temporelle (c-à-d après la zone bleue).
- = si je copie/colle un bout de graphique après le *warmup* vous ne serez pas capable de distinguer s'il s'agit du début/milieu/fin du graphique.

« Warming up »

warmup
ou
warming up



$x(t)$

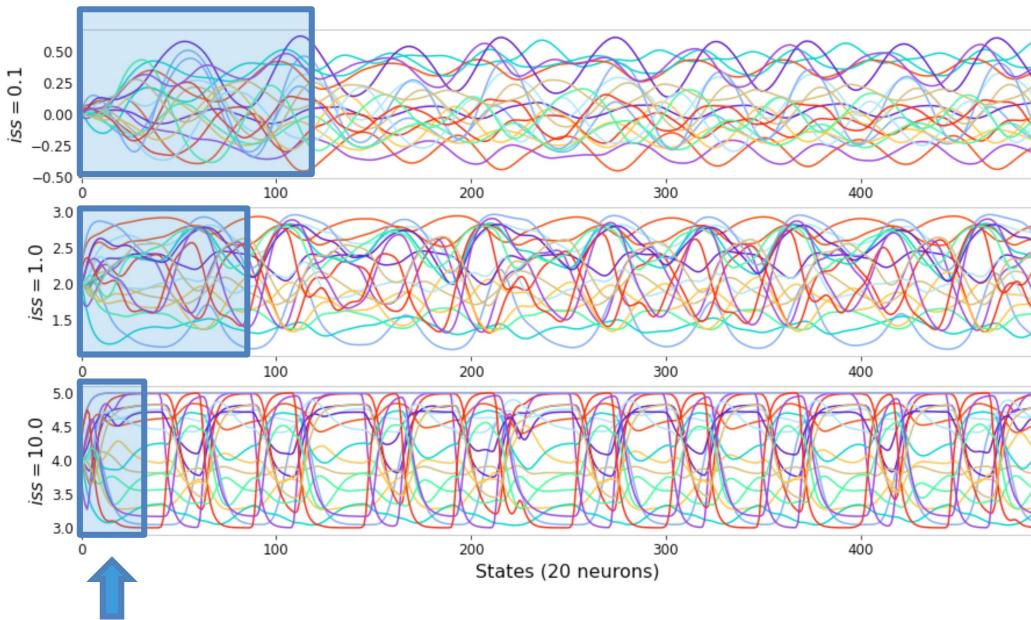
A priori, vous ne voulez pas utiliser ces dynamiques initiales du réservoir qui peuvent perturber l'apprentissage de votre tâche.

→ En conséquence, il faut définir l'hyperparamètre *warmup*

- « à l'œil » par rapport aux activités internes du réservoir
- et/ou en testant par essai/erreur pour quel *warmup* minimal vous avez des performances qui ne changent plus

« Warming up »

warmup
ou
warming up



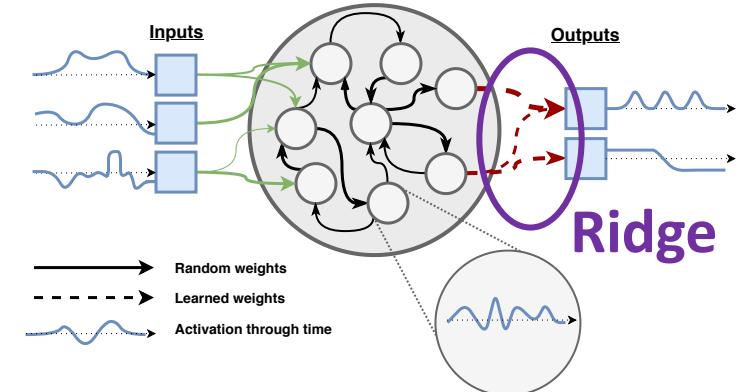
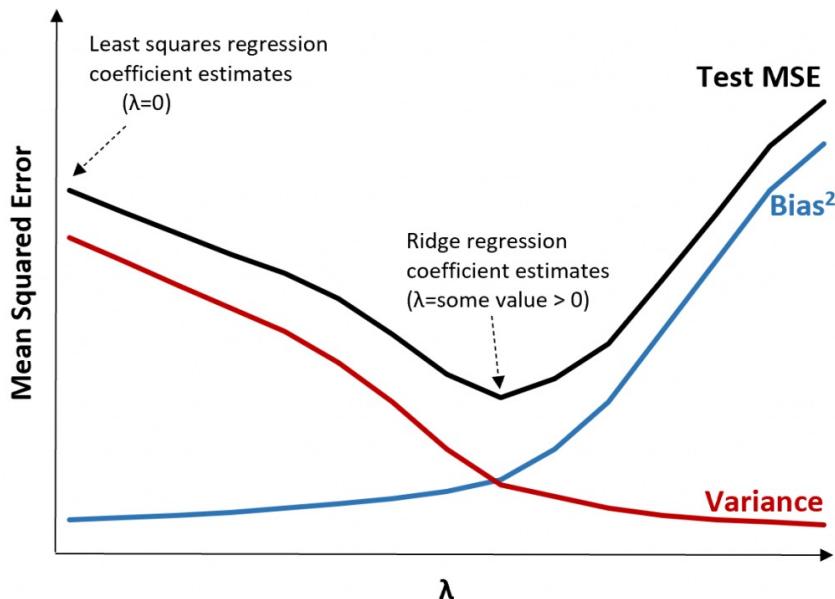
$x(t)$

Ce *warmup* / « temps de chauffe » dépend

- des hyperparamètres du réservoir
 - (ici l'input scaling iss)
- des entrées / des données que vous donnez au réservoir
 - (par ex. si le taux d'échantillonnage change)

Ridge

= régularisation de la régression linéaire



- Modèle avec **variance élevée** est très flexible
 - et s'adapte trop bien aux données d'entraînement, **y compris le bruit**
- Modèle avec **biais élevé** simplifie trop le problème
 - le modèle est **trop simple** pour capturer les relations sous-jacentes dans les données

Méthodes pour la recherche d'hyperparamètres

- « Random search » répétés et adaptés
 - Hinaut & Trouvain, ICANN 2021
 - Plus efficace que « grid search »
- Algorithmes génétiques avec 400 variables d'entrées
 - Ferte, ..., Hinaut, ICML 2024
 - Prédiction hospitalisations COVID à +14 jours

Which Hype for my New Task?
Hints and Random Search for
Echo State Networks Hyperparameters

Xavier Hinaut^{1,2,3,*} [0000-0002-1924-1184] and Nathan Trouvain^{1,2,3} [0000-0003-2121-7826]

¹ INRIA Bordeaux Sud-Ouest, France.
² LaBRI, Bordeaux INP, CNRS, UMR 5800.
³ Institut des Maladies Neurodégénératives,
Université de Bordeaux, CNRS, UMR 5293.

*Corresponding author: xavier.hinaut@inria.fr

Abstract. In learning systems, hyperparameters are parameters that are not learned but need to be set a priori. In Reservoir Computing, there are several parameters that need to be set a priori depending on the task. Newcomers to Reservoir Computing cannot have a good intuition on which hyperparameters to tune and how to tune them. For instance, beginners often explore the reservoir sparsity, but in practice this parameter is not of high influence on performance for ESNs. Most importantly, many authors keep doing suboptimal hyperparameter searches: using

<https://inria.hal.science/hal-03203318>

Reservoir Computing for Short High-Dimensional Time Series: an Application to SARS-CoV-2 Hospitalization Forecast



Thomas Ferté, Dan Dutartre, Boris P Hejblum, Romain Griffier, Vianney Jouhet, Rodolphe Thiébaut, Pierrick Legrand, Xavier Hinaut

Published: 02 May 2024, Last Modified: 25 Jun 2024 [ICML 2024 Poster](#) [Everyone](#) [Revisions](#) [BibTeX](#)
CC BY-NC 4.0

Abstract:

In this work, we aimed at forecasting the number of SARS-CoV-2 hospitalized patients at 14 days to help anticipate the bed requirements of a large scale hospital using public data and electronic health records data. Previous attempts led to mitigated performance in this high-dimension setting: we introduce a novel approach to time series forecasting by providing an alternative to conventional methods to deal with high number of potential features of interest (409 predictors). We integrate Reservoir Computing (RC) with feature selection using a genetic algorithm (GA) to gather optimal non-linear combinations of inputs to improve prediction in sample-efficient context. We illustrate that the RC-GA combination exhibits excellent performance in forecasting SARS-CoV-2 hospitalizations. This approach outperformed the use of RC alone and other conventional methods: LSTM, Transformers, Elastic-Net, XGBoost. Notably, this work marks the pioneering use of RC (along with GA) in the realm of short and high-dimensional time series, positioning it as a competitive and innovative approach in comparison to standard methods.

<https://openreview.net/forum?id=CY0lFwD4qx>

