ReservoirPy: The Only Thing More Chaotic Than My High School

(Yes, my high school was called Reservoir High School)

Author: Bella Pereira

Date: 05/10/2025

[1] Name of the package, describe what is the basic aim of what the package does or solve?

The ReservoirPy package is a Python library designed for building, training, and using Echo State Networks(ESNs), a type of recurrent neural network used in Reservoir Computing. The package provides a flexible, modular framework for time series prediction, signal processing, and other sequential data problems. ReservoirPy aims to simplify the implementation of ESNs while maintaining performance and efficiency. It also includes tools to generate synthetic datasets from well-known systems like the Lorenz-system.

[2] Why/how did you select this package?

I chose to work with this package because I was interested in exploring alternatives to traditional recurrent neural networks for modelling chaotic systems. Reservoir computing presents an interesting alternative to learning architectures by offering faster training and better stability from time-dependent tasks.

[3] How old is the package? does it have a geneology, i.e. what related codes came before or after. are there other codes you can find that solve the same problem? Can you figure out which version you installed?

ReserviorPy was first released on GitHub in 2021 and developed by researchers at INRIA. Other libraries that are similar are Oger, pyESN, and EasyESN, however, ReserviorPy is maintained more frequently. The version I have installed is 0.3.13post1.

[4] Is it still maintained, and by the original author(s)? Are there instructions how to contribute to this project?

It is currently maintained on GitHub, mainly by nTrouvain and PAUL-BERNARD, affiliated with the original authors from INRIA, France. Instructions to contribute to ReservoirPy as well as future update plans are clearly stated on GitHub:
https://github.com/reservoirpy/reservoirpy

[5] Evaluate how easy it was to install and use. What commands did you use to install?

It's installation is very simply using: “pip install reservoirpy”.

[6] Does it install via the "standard" pip/conda, or is it more complex?

It is extremely easy to install via pip. You don't need any Conda or manual build to install it.

[7] Is the source code available? For example, "pip install galpy" may get it to you, but where can you inspect the code?

Full source code is hosted on GitHub and can be inspected by browsing the repository at the GitHub previously stated. The code includes modules, examples, and developer documentation.

[8] Is the code used by other packages (if so, give one or two examples). ASCL codes have citations via their ADS link. See also 22.

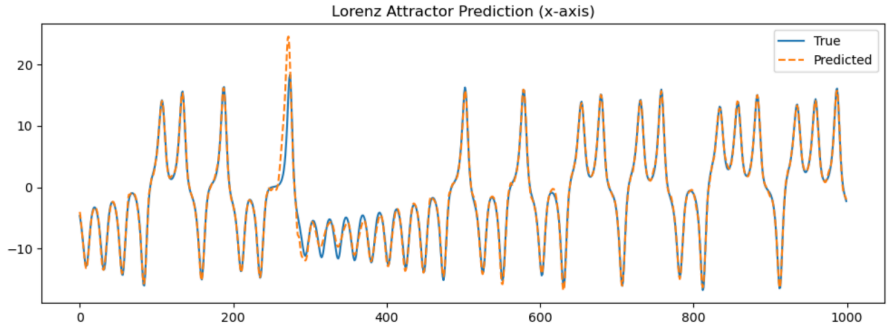
ReservoirPy primarily depends on research and academic packages. For example, “Conn2Res”, which models neural connectivity using ESNs, depends on ReservoirPy and includes it in its “requirements.txt”.

[9] How is the code used. Is it commandline, python script, or a jupyter notebook, or even a web interface?

The code is used primarily through Python scripts and Jupyter notebooks. However, there is no command-line or web interface for ReservoirPy.

[10] provide examples using the code. if you prefer to use a jupyter notebook instead of a python script, that's ok. See also 12

```python
from reservoirpy.datasets import lorenz
from reservoirpy.nodes import Reservoir, Ridge
import numpy as np
import matplotlib.pyplot as plt

# Load Lorenz data
timeseries = lorenz(10000) # from example
data = timeseries[:, 0].reshape(-1, 1)

trainLen = 8000
xTrain = data[:trainLen]
yTrain = data[1:trainLen + 1]
xTest = data[trainLen:-1]
yTest = data[trainLen + 1:]

# Define ESN NO noise
reservoir = Reservoir(units=300, sr=1.25, lr=0.3, input_scaling=0.5)
readout = Ridge(ridge=1e-6)

esn = reservoir >> readout

# Train and predict
esn = esn.fit(xTrain, yTrain)
yPred = esn.run(xTest)

# Plot
fig, ax = plt.subplots(figsize =(12,4))
ax.plot(yTest[:1000], label = "True")
ax.plot(yPred[:1000], label="Predicted", linestyle="--")
ax.set_title("Lorenz Attractor Prediction (x-axis)")
ax.legend()
```

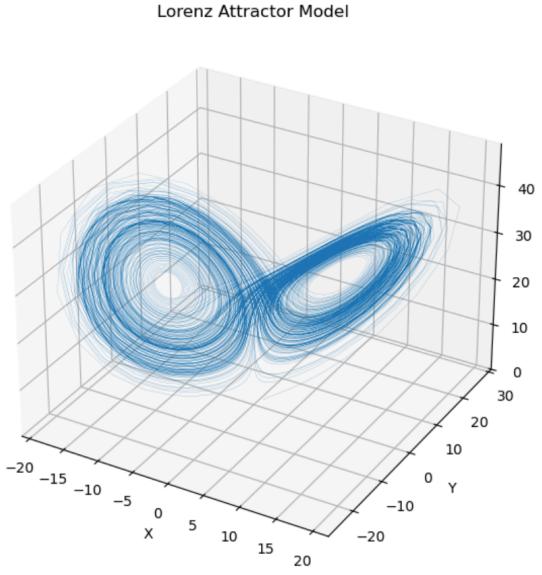


This figure compared the true and predicted trajectories of the x-component of the Lorenz attractor over time. The blue solid line represents the ground truth data generated using the Lorenz differential equations, while the orange dashed line shows the prediction produced by the ESN trained by ReserviorPy. There is a high degree of overlap which indicates that the ESN effectively learned the dynamics of the Lorenz system.

[11] Does the package produce figures, or are you on your own? Is matplotlib used?

The package is unable to produce figures on its own, but supports integration with mainly matplotlib to visualize results.

[12] Your code and report should show at least one figure, and create a nice figure caption explaining what it shows. You notebook should show how the figure was made (i.e. be reproducable). Second figure is optional, but only use it when you need to illustrate something extra. This implies you may need to add any data you need to your repository!!!



This is a 3D figure that shows the Lorenz attractor trajectory in the (X, Y, Z) space, plotted using data generated from the Lorenz system. The attractor demonstrates the system's chaotic behavior through its butterfly-shaped structure. The visualization helps illustrate the complexity of the Lorenz system and is used to verify the quality of the time series predictions from the models.

[13] Is the package pure python? or does it need accompanying C/C++/Fortran code?

ReservoirPy is mainly Python based on NumPy and SciPy. There is no C/C++/Fortran extensions required at all.

[14] What is the input to the package? Just parameters, or dataset(s), or can they be generated from scratch?

Various inputs to the package include time series data, typically NumPy arrays. Another input is model parameters to adjust reservoir size, input scaling, and spectral radius.

[15] What is the output of the package? Just parameters, or dataset(s)?, or just a screen output you would need to capture

The main output is a predicted time series or numeric matrix(numpy.ndarray). The package may also return metrics like mean squared error when evaluating model performance.

[16] Does the code provide any unit tests, regression or benchmarking?

ReservoirPy provides built-in tools for evaluations, such as the previous metric, R-squared score, and memory capacity measurement. These help assess model quality and are listed in the API guide under "observable" methods at https://reservoirpy.readthedocs.io/.

[17] How can you feel confident the code produce a reliable result?

This code is kept up to date and tested often on GitHub. Additionally, its Lorenz dataset generation matches the results from my test, seen above.

[18] What (main) python package(s) does it use or depend on (e.g. numpy, curve_fit, solve_ivp) - how did you find this out?

ReservoirPy depends on different Python libraries like numpy, scipy, sklearn, and matplotlib. You can check what Python packages ReservoirPy depends on via "pip show reservoirpy" or examining "setup.py"

[19] What kind of documentation does the package provide? was it sufficient for you?

The main documentation for this package is found at https://reservoirpy.readthedocs.io/. This website is essential for understanding ReservoirPy as it includes in-depth tutorials, theory explanations, examples, and API references. It was more than sufficient for me to help complete this project.

[20] If you use this code in a paper, do they give a preferred citation method?

The preferred method for citation is in the "README" in the GitHub. They prefer citing work in the code in this format:

```
Python
@incollection{Trouvain2020,
  doi = {10.1007/978-3-030-61616-8_40},
  url = {https://doi.org/10.1007/978-3-030-61616-8_40},
  year = {2020},
  publisher = {Springer International Publishing},
```

```
  pages = {494--505},
  author = {Nathan Trouvain and Luca Pedrelli and Thanh Trung Dinh and Xavier
Hinaut},
  title = {{ReservoirPy}: An Efficient and User-Friendly Library to Design Echo State
Networks},
  booktitle = {Artificial Neural Networks and Machine Learning {\textendash} {ICANN}
2020}
}
```

[21] Provide any other references you used in your report.

The main resources I used were from https://reservoirpy.readthedocs.io/ and the GitHub repository. I found that they had ample information for me to understand the package.

[22] Can you find two other papers that used this package. E.g. use ADS citations for ASCL based code. See also 8.

Two papers I found that use ReserviorPy:

1. Enhanced multi-step streamflow series forecasting using hybrid signal decomposition and optimized reservoir computing models
   https://www.sciencedirect.com/science/article/pii/S0957417424017238
2. Isovector electromagnetic form factors of the nucleon from lattice QCD and the proton radius puzzle
   https://arxiv.org/abs/2102.07460

[23] Did you have to learn new python methods to use this package? Or was the class good enough to get you through this project.

The class helped me get through some of the project since the package wasn't too hard for me to understand. The new methods I learned were about Echo State Networks and Reservoir computing. Learning those required me to expand my understanding of time series modelling concepts.

[24] Final Disclaimer: you need to state if you have prior experience in using the package or the data, or this is all new to you. In addition, if you collaborated in a group, as long as this is your work

I have no prior experience with ReservoirPy before this project. I collaborated with another student, Kanjonavo Sabud, to strengthen our understanding of the package, however, all of this report work was done independently.