Predicting Oscillatory Systems with Machine Learning

The College at BROCKPORT
STATE UNIVERSITY OF NEW YORK

Nolan J. Coble

Summary

- Predicting dynamically evolving systems is important for applications in meteorology, biology, etc.
- Academic groups will often approach this problem from different perspectives
- Reservoir computing, a machine learning technique, is a promising candidate for predicting chaotic systems
- In this poster, the focus is on predicting the time evolution of mass-spring systems
- A thorough treatment of reservoir computing is provided, and prediction results are explained

Reservoir Computing

- Machine learning technique that uses a recurrent neural network, known as a "reservoir", to predict the evolution of a dynamical system
  - Neural network: sparsely connected nodes that evolve over time, similar to neural activity
- Reservoir has no knowledge of system dynamics; it adapts to known training data for the system
- Method:
  1) Listening: input system states and generate reservoir states
  2) Training: adjust output layer such that reservoir output closely approximates reservoir input
  3) Prediction: use reservoir output as the next input

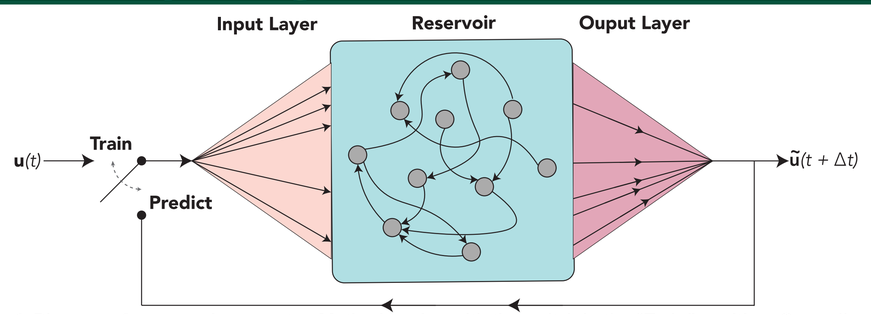

Figure 1. Diagram of a reservoir computer. Method begins with the switch in the "Train" position. Once all training data has been processed and output weights are adjusted, switch is flipped to "Predict" position. Now, output from the reservoir is used as input, and the system evolves without any data from the true system state.

Prediction Strategies

Predicting dynamics is important, but how is it done? People from various academic departments will approach this problem in different ways. Here is an outline of some pros and cons of each approach.

Area	Pros	Cons
Physicist **Develop Theory**	• Exact solution • Provides a physical understanding of the problem	• Impractical for certain systems • Impossible for others
Applied Mathematician **Create Model**	• Can give very accurate results	•Typically need some knowledge of solution form • Impractical for complex systems
Computer Scientist **Use Machine Learning**	• Good at predicting all types of systems • No physical knowledge required	• Prevides no physical understanding of the problem •Time required varies greatly

Next Steps

The predictions for these small mass-spring systems were good, but not perfect. If we were to look at systems with hundreds or even thousands of components, we would see that **the error would quickly grow** out of control.

How do we fix this? One way is to take advantage of the fact that **the components are not independent, they are coupled together**. By using this, we can develop a parallel reservoir scheme that will drastically reduce error in large sytems.

Mass-Spring Systems

Imagine two frictionless carts on a track, attached to each other by a spring, and each attached to the wall by a spring. This would be a 2-mass, 3-spring system.

If the two carts are pulled apart then allowed to move freely, their motion would be oscillatory. **Figure 2(a)** to the right shows this motion. A natural question may arise: how do we predict the future evolution of this system given this data?

Reservoir computing is the answer! By generating reservoir states for each training datum and training the reservoir output layer, we can predict this system.

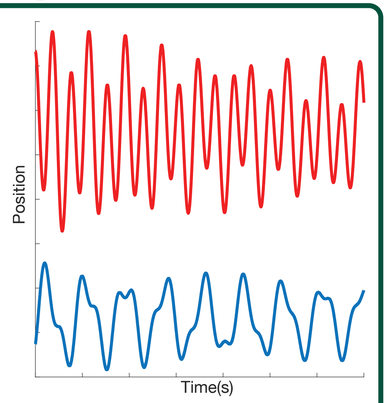

Figure 2(a). Position vs. time plot for a 2-mass, 3-spring system.

Instead of only two carts, imagine now five. **Figure 2(b)** shows a possible motion of this 5-mass, 6-spring system.

Again, we can use reservoir computing to predict the future evolution of this system. You will notice, however, that the prediction results are not quite as perfect as the previous example.

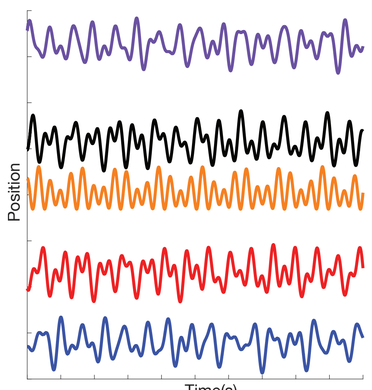

Figure 2(b). Position vs. time plot for a 5-mass, 5-spring system.

Prediction Results

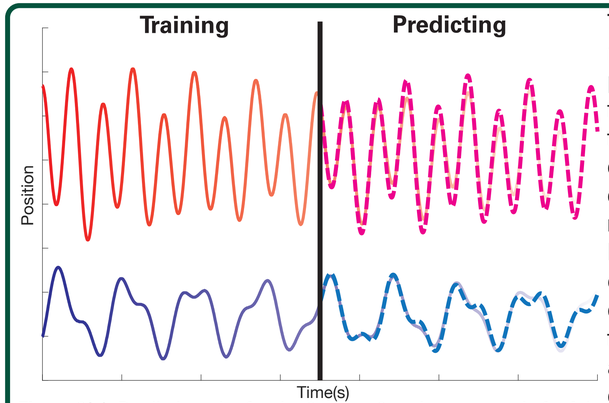

Figure 3(a). Prediction plot for the 2-mass, 3-spring system. Left of the black line the system is training. The prediction occurs to the right.

The dotted line represents the reservoir prediction, and the faded line behind it is the true data. In this case, the reservoir computer predicts **nearly perfectly**. Notice, however, that the blue dotted line does not exactly match up with the faded line behind it at times. This is an example of **error** in the prediction.

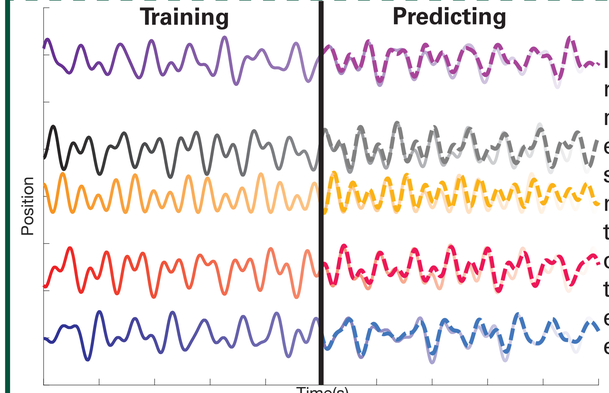

Figure 3(b). Prediction plot for the 5-mass, 6-spring system. Left of the black line the system is training. The prediction occurs to the right.

In this case, the reservoir computer makes many more errors than the 2-mass system. In particular, notice how the peaks of the prediction never quite reach as far as the true data. Although the error may be small, it is error nonetheless.