Machine Learning Approach to the n-Body Problem

Luca Lavezzo and Brandon Manley, Department of Physics

Spring Undergraduate Research Festival, The Ohio State University

Background

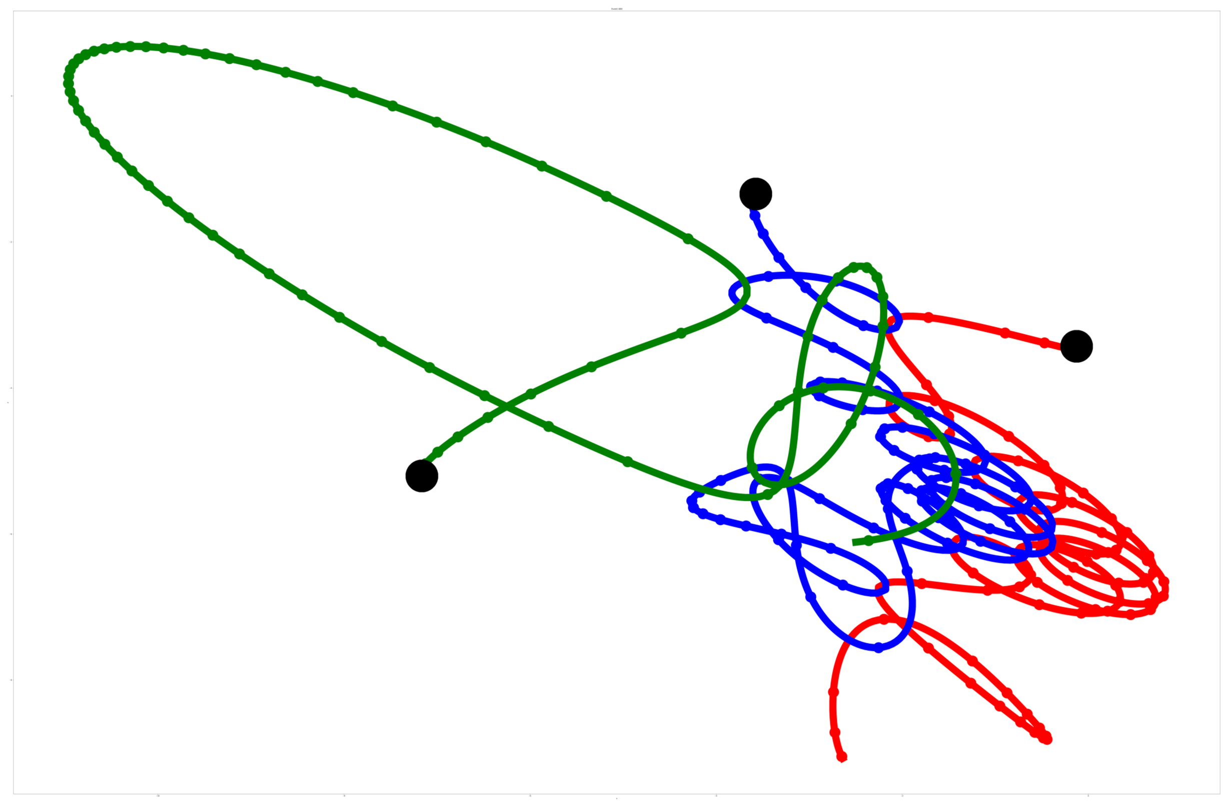
The n-body Problem

- Classically, seek to calculate motion of n bodies under the action of Newtonian gravity, or other interactive forces
- Seek to model *small-scale* interactions (1 – 100 bodies)
- No closed form, mathematical solution to the coupled differential equations that describe the interactions

$$m_i \frac{d^2 \mathbf{q}_i}{dt^2} = \sum_{\substack{j=1 \\ j \neq i}}^{n} \frac{G m_i m_j \left(\mathbf{q}_j - \mathbf{q}_i\right)}{\left\|\mathbf{q}_j - \mathbf{q}_i\right\|^3} = -\frac{\partial U}{\partial \mathbf{q}_i} \qquad (1)$$

- The problem must be approached through numerical solutions of the differential equations
- The problem is *chaotic* – extreme sensitivity to initial conditions, impossible to predict accurately after a long period of time

Fig. 1. Path of n particles interacting through gravity, simulated using Brutus [1].



Simulation and Previous Approaches

Brutus simulation originally developed by Protegies, Zwart & Boheckolt [1] and adapted by Breene et al. [2].
The Brutus simulation is an arbitrary precision brute force simulation that loops over each combination of particles and directly calculates the acceleration from eq (1).

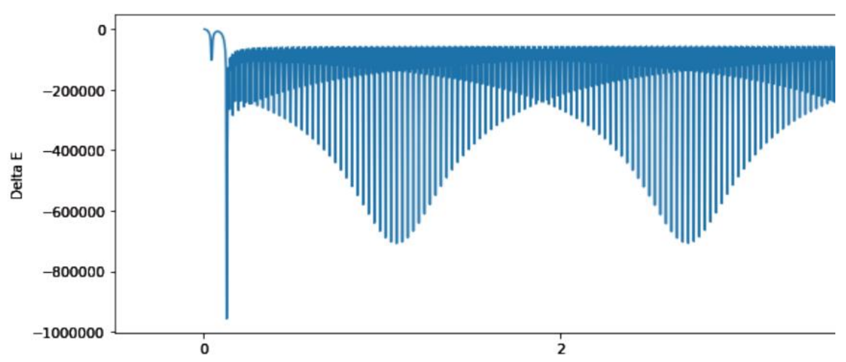


Fig. 2. An alternative simulation we built in Julia was found to violate conservation of energy in events that had close encounters. A sample event's change in energy over time is pictured.

Breene et al. [2] present the original idea of this approach in a simplified version of the problem by setting initial velocities to 0 and keeping mass constant. They achieve about an order of magnitude better than our results of a more complex system. We achieved similar results for the same problem.

Method

Simulation parameters:

- Simulate n bodies interacting through gravity, G=1, for N bodies = 3, 4, 5.
- Masses vary between [1, 100]
- Initial velocities varying between [-1 , 1]
- 1 event = 2560 steps per unit time *t,* simulated until *t* = 10.

Results:

- 3 bodies (4000 events) - 4 bodies (500) - 5 bodies (500)
- 24 hours to produce ~500 events for 3 body simulation, ~200 events for 4 and 5 body simulation

- Neural network chosen to solve coupled differential equations due to ability to approximate function that describe the relationship between an outcome and a set of covariates.
- Many have already demonstrated ability of neural networks to predict the states of dynamical systems in time (See [3] [4] [5]).

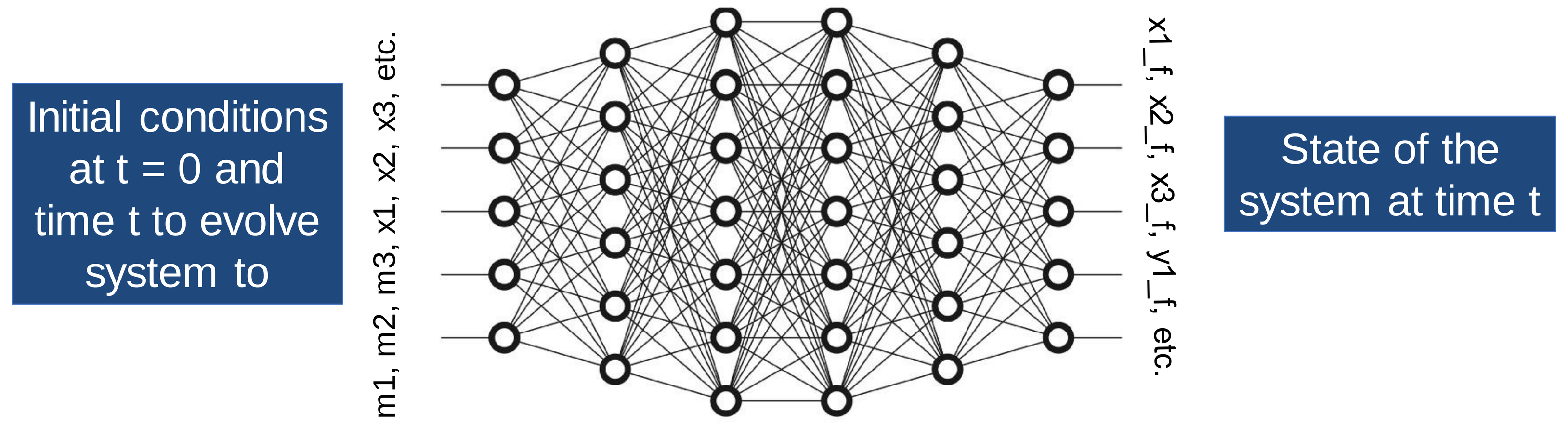


Figure 3. Schematic of the network, drawing from [6]

Optimization

- Need different neural network for each n-body simulation, since each n represents different computational complexity
- Each is optimized by conducting a **Grid Search** over a set of parameters to determine which yields the best performing network (evaluated using validation MAE): the parameters optimized are the *number of hidden layers* of the network and the *number of nodes per layer*.
- At each combination of parameters, a **5-fold validation** is conducted. K-Fold validation consists of randomly splitting the data in K batches to train K-1 batches and validate on the last, which gives an unbiased estimate of the network's performance on unseen data.

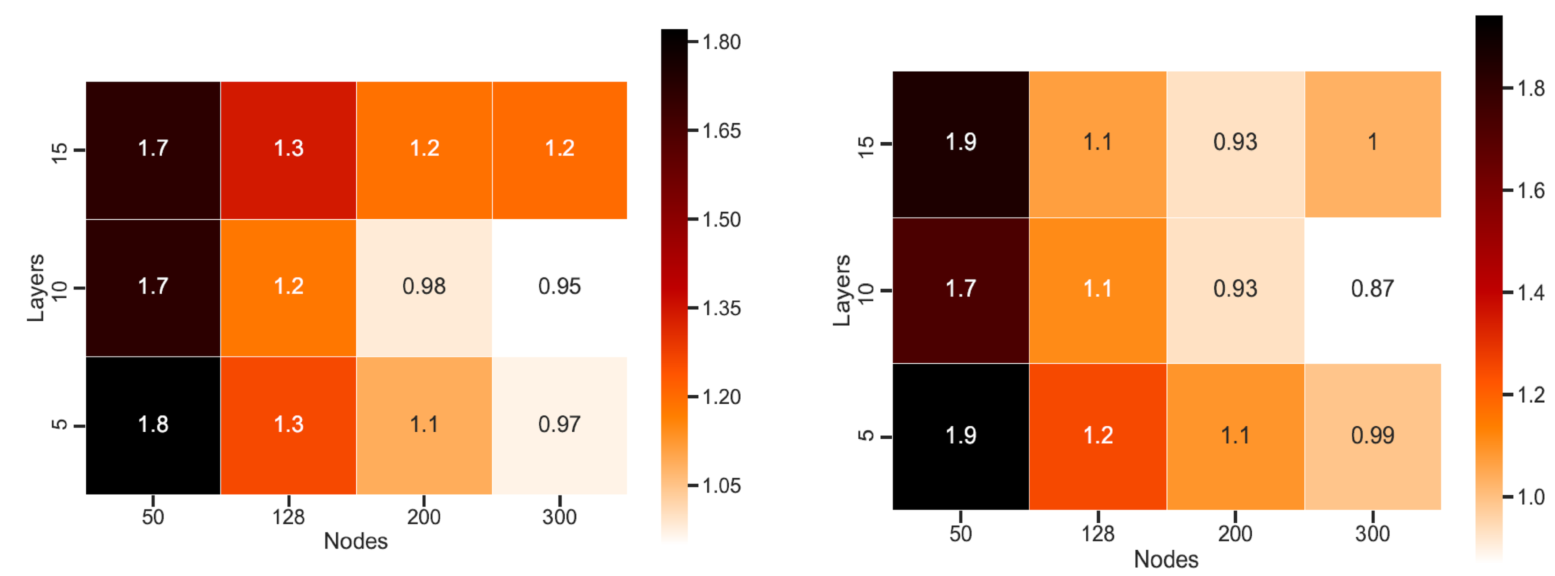


Figure 4. Grid search MAE results for 4 and 5 bodies.

Results

Number of bodies	MAE (Test)
3	1.00
4	0.78
5	0.83

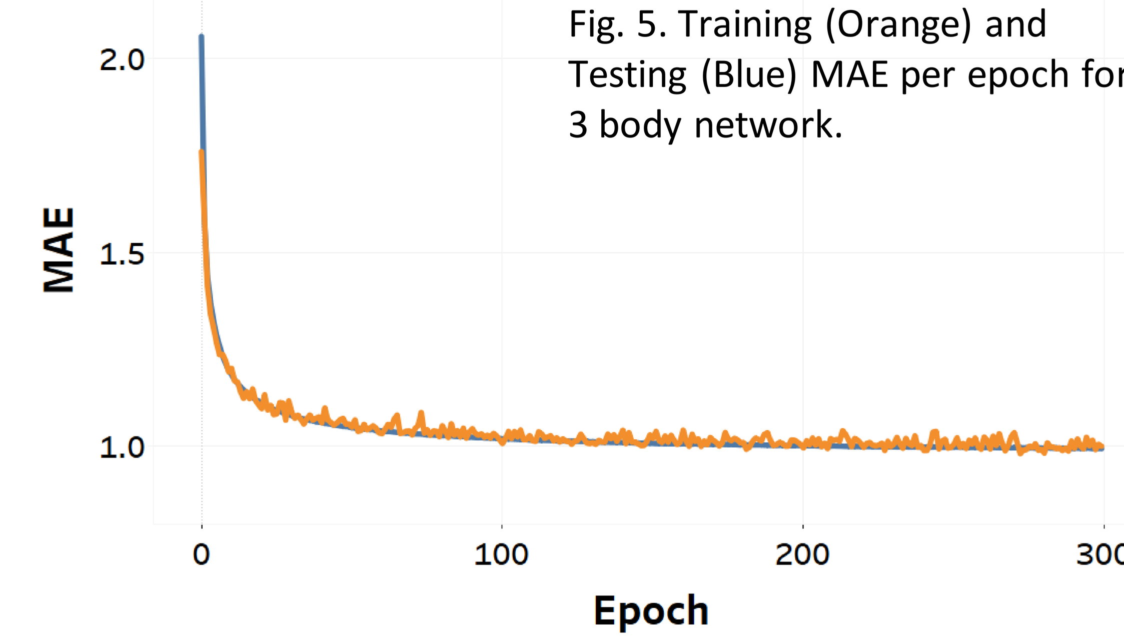
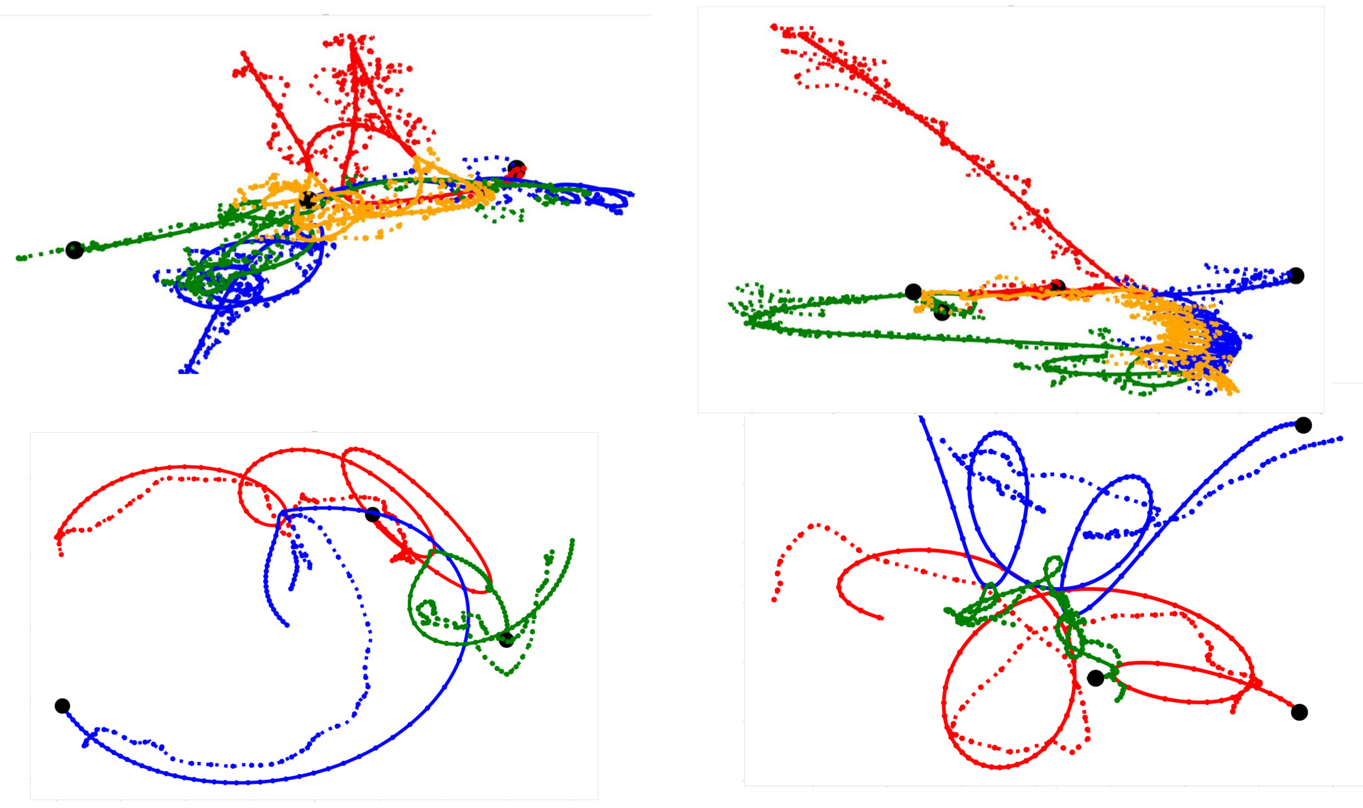


Fig. 5. Training (Orange) and Testing (Blue) MAE per epoch for 3 body network.

- Higher MAE than Breene at al reflects increased dimensionality of problem (2 -> 5*n)
- Comparable performance between 3, 4, and 5 bodies -> early indication of the scalability
- Predicted results do not observe conservation laws as well as simulation. Paths also don't observe same smooth path as simulation
- Networks seemed to have achieved a certain amount of accuracy, but need more work to achieve greater accuracy akin to the result in Breene et al.



Future Studies

- Seek to generalize to higher dimensional problems
- Consider alternative data driven methods, starting from new architectures of neural networks, LSTM, CNN, etc.
- Develop way to incorporate physics into the neural network (e.g. via Physics Informed Neural Networks, ChaosNet etc.)
- Apply method to other chaotic systems that obey physical laws (atoms, turbelent flow, etc.)
- Seek to increase predictability of physical chaotic systems

References

[1] Boekholt T., Portegies Zwart S., 2015, Computational Astrophysics and Cosmology 2, 2
[2] Breen P.G., Foley C.N., Boekholt T, and Portegies Zwart S., 2019, Newton vs the machine: solving the chaotic three-body problem using deep neural networks, arXiv: 1910.07291
[3] Pathak et. Al., 2018, Hybrid Forecasting of Chaotic Processes: Using Machine Learning in Conjunction with a Knowledge-Based Model, arXiv:1803.04779
[4] Yi Wan et. Al., 2018, arXiv:1803.03365
[5] Gately E., 1995, Neural Networks for Financial Forecasting, John Wiley & Sons, Inc. New York, NY, United States
[6] Raicea R. "Want to Know How Deep Learning Works? Here's a Quick Guide for Everyone." FreeCodeCamp.org, FreeCodeCamp.org, 31 Mar. 2020