Analyse a set of outputs of ARWV with machine learning:
Can Deep Learning help us see order in chaos?

Luca Negri

Physics and Astronomy department
University of Padua

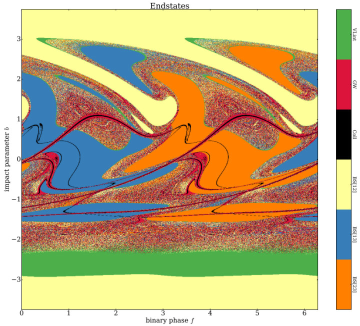
January 25, 2022

Sections

1. Introduction
2. Data analysis and manipulation
3. Neural Network and training
4. Output analysis

- The 3 body problem has puzzled physicists and mathematicians for centuries.
- Differential equations that describe gravity have no general analytical solutions.
- Numerical integration is required to solve the equations
- **High dependence on initial condition** leads to chaotic behaviours.

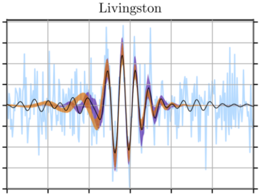
- Computational Complexity rises fast.

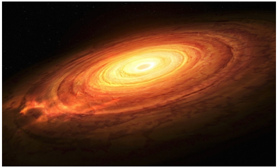


Introduction : Modern Challanges in science

Gravitational Waves astronomy

- The LIGO-VIRGO collaboration has observed many Binary Black Holes (BBH) mergers for years
- The formation of these massive BBH system (like GW190521) is theorized to occur mainly in the densest parts of Star Clusters (SC)
- **Many close gravitational and chaotic interactions** occur in SCs
- Single-binary BHs scattering can lead to formations of tightly bounded high mass systems.
- But are **Hard to integrate**





Introduction : When does ML comes in

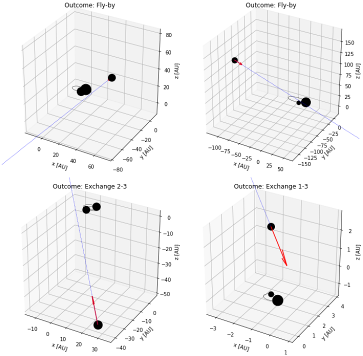
The aim of this project is to build a **Fully connected Deep NN** that can **predict the topology** (general outcome) of an encounter.

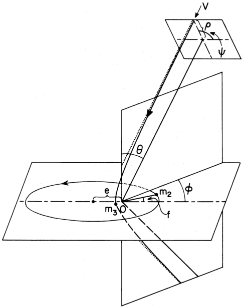
- By taking the output of a large set of 3-body interactions (BBH-BH) computed by ARWV (a higly accurate nbody relativistic integrator) we can **train a neural network on it**
- once trained(and if it works) it computes the outcome in a fraction of the time reqiured by ARWV, skipping the numerical integration alltoghether

This network can be used:

- to study the probability distribution of the outcomes in general
- To predict the outcome of just close encounters in larger SCs simulations, by trading precision with less computational requirements.

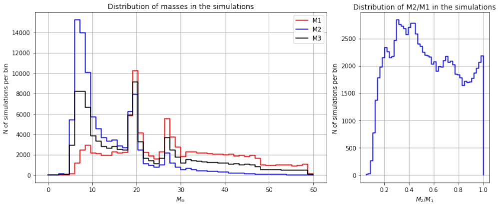
Main focus on a dataset of 1e5 simulations performed with ARWV , with 18 columns that describe initial conditions of each simulations and 4 columns that descibe the output.

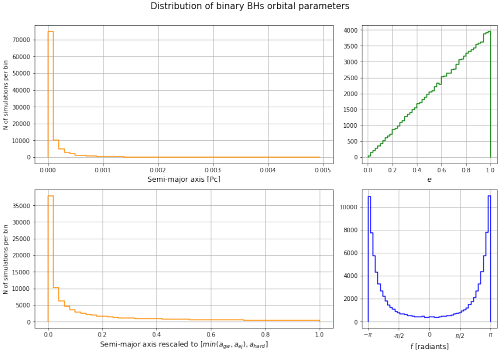




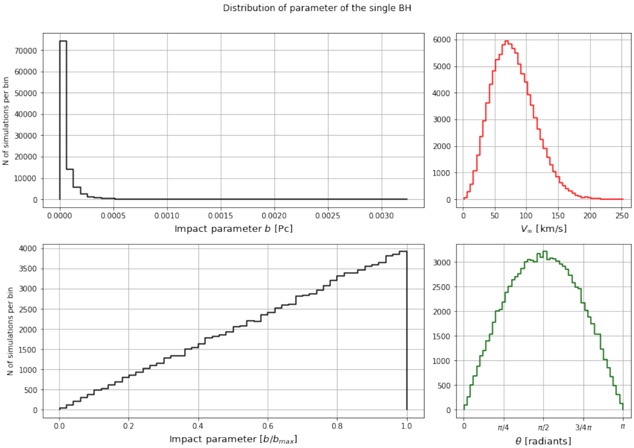
Università degli Studi di Padova

Dataset: Masses and BBH orbital parameters





Dataset: Single BH parameters



Problem: Neural networks work best when data is roughtly in the range $[-1, 1]$:

- Solution for $\{m_1, m_3, a, e, v_\infty, \theta\}$:

$$\mathrm{x_{NN}} = \frac{\mathrm{x_0} - \overline{x_0}}{std(\mathrm{x_0})}$$

- Solution for $\{\psi, \phi, f\}$

$$\mathrm{x_{NN}} = \{sin(x_0), cos(x_0)\}$$

(To impose periodicity of the output function)

- Solution for b:

$$\mathrm{b_{NN}} = \frac{\mathrm{b_0}}{\mathrm{b_{max}}} - \overline{\left(\frac{\mathrm{b_0}}{\mathrm{b_{max}}}\right)}$$

Dataset manipulation

- Solution for a :

$$a_{NN} = \frac{a - min(a_{gw}, a_{ej})}{a_{hard} - min(a_{gw}, a_{ej})},$$

- Solution for m2 :

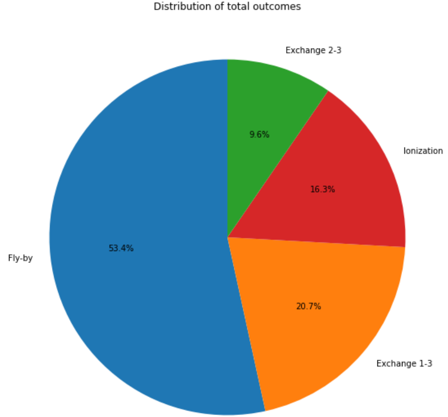
$$M_{2NN} = \frac{M_2}{M_1}$$

- Additional parameter: v_{crit} indicator is 1 if

$$v_{\infty} > v_{crit} = \sqrt{\frac{m_1 m_2 (m_1 + m_2 + m_3)}{m_3 a (m_1 + m_2)}}$$

Dataset: Outcomes

Index	Outcome	Description
0	Fly-by	The binary BHs remain as a binary at the end of the simulation
1	Exchange 1-3	The smaller BH in the binary is replaced by the single BH
2	Exchange 2-3	The bigger BH in the binary is replaced by the single BH
3	Ionization	No binaries remain at the end of the simulation



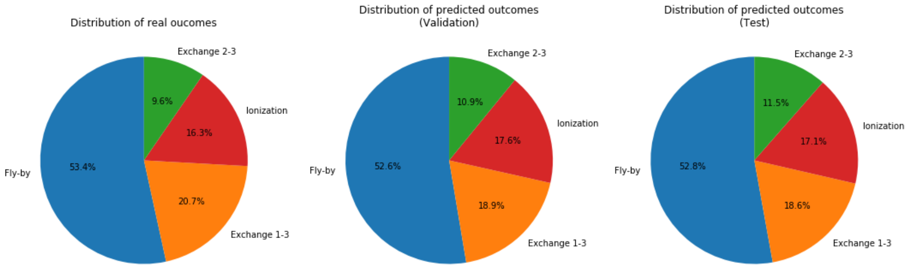
Neural network

Neural network constructed with `pytorch`

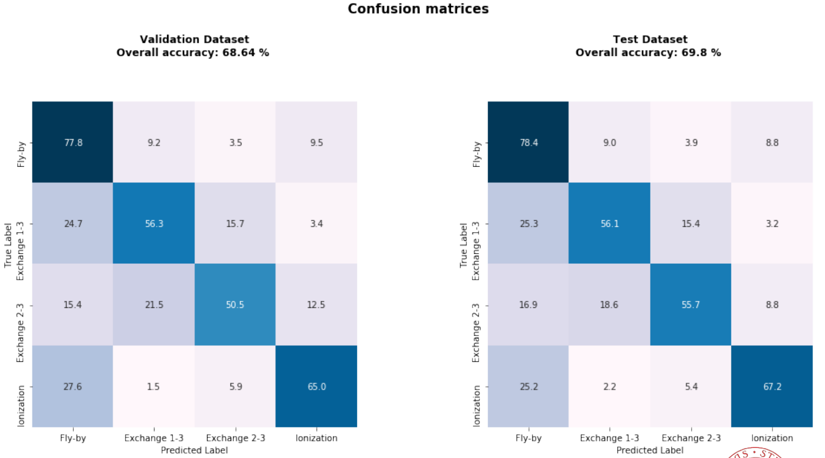
- 4 hidden layers, with 256 nodes each.
- 15 nodes as inputs (8 "normal" parameters + 3 sines + 3 cosines + v_{crit}), 4 nodes as outputs (4 classes)
- ReLU as activation function
- Optimizer: Adamax with lr = 2e-3
- minibatch learning with sample shullfe and 256 as batch size
- Dropout 0.3 and L2 regularization used to avoid overfitting (weight decay = 2e-5)
- loss function: Cross entropy (weighted classes: [0.21 , 0.21 , 0.31 , 0.27])

Neural Network: Training for 60 epochs

Network Output Distribution

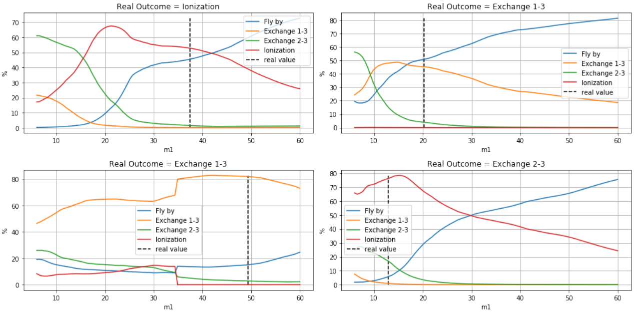


Network output: Confusion matrices



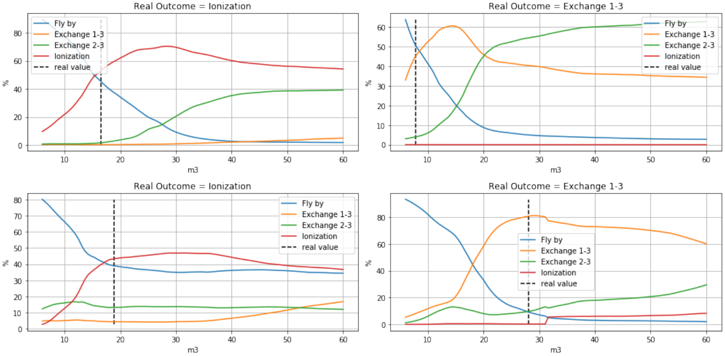
Network output: 1D analysis of M_1

Here we analyze the network output in more specific detail, choosing a random simulation and looking at how the output function changes by varing just one parameter



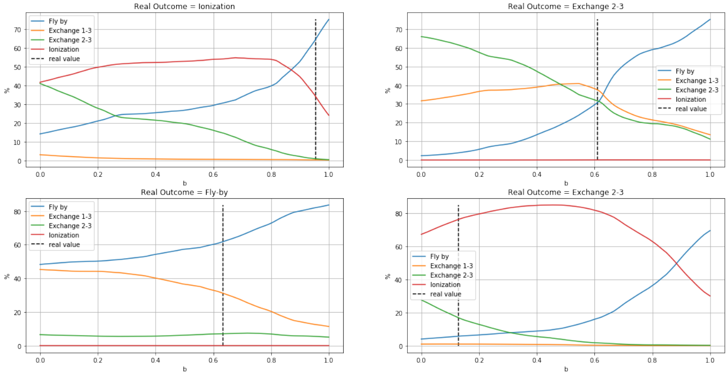
- High M_1 -> high fly-by
- Interesting behaviour with ionization in some cases

Network output: 1D analysis of M_3



- High M_3 -> Low fly-by
- some "trading" exists between exchange probabilities

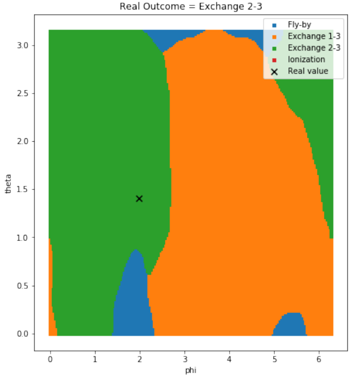
Network output: 1D analysis of b/b_{max}

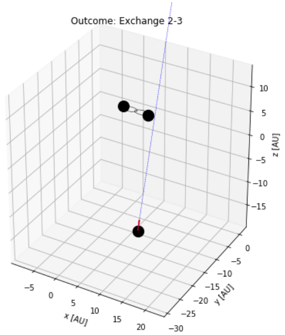


- High b/b_{max} -> high fly-by
- Exchanges follow similar patterns

Network output: 2D analysis of ϕ and θ

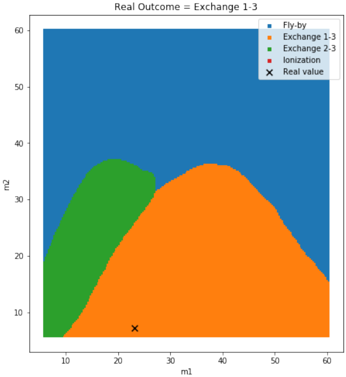
Same concept as before applies, but now 2 parameters are changed at the same time.

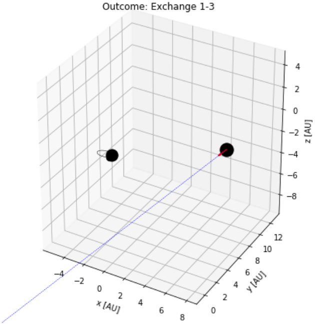




- Highly simmetrical system leads to highly simmetrical output
- fly-by probabilities: net artifacts? Actual property?

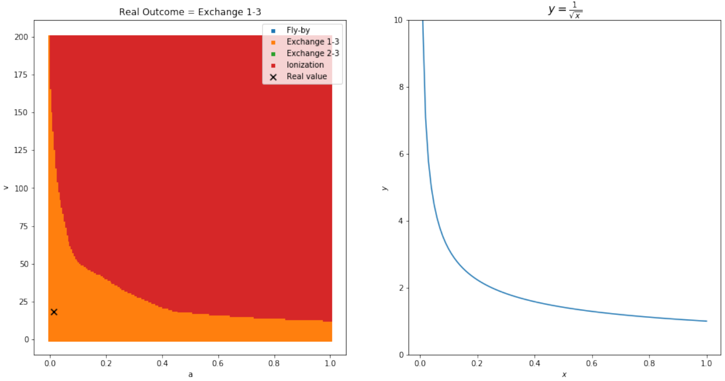
Network output: 2D analysis of M_1 and M_2





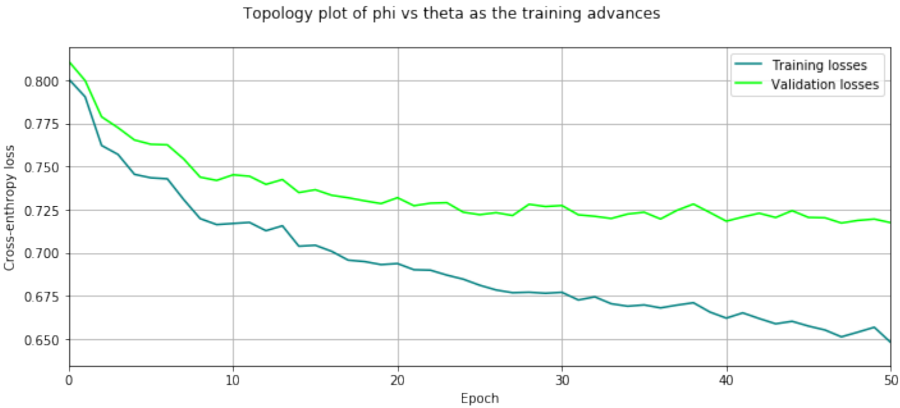
- Expected Simmetry in $M_1 - M_2$ is weakly present, even if the network was not trained on recognising it
- moving far away form actual training data distr makes the accuracy go down

Network output: 2D analysis of v_∞ and Semi-major axis

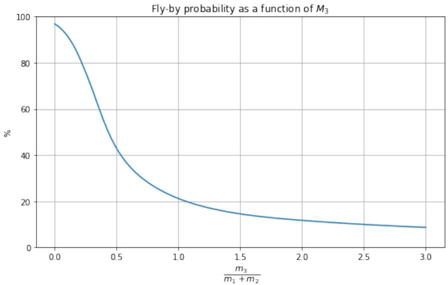


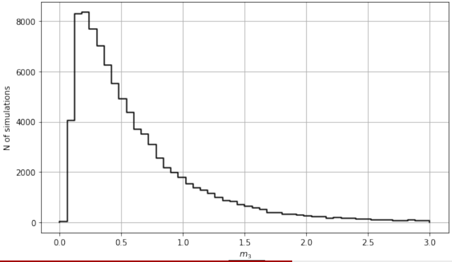
- Expected behaviour from v_{crit} formula

Network output: Complexity as a function of training



Epochs = 10 Epochs = 20 Epochs = 30 Epochs = 40 Epochs = 50





- We know that this relation should hold up in theory, since fly-by should go down as this ratio increases and probability of interaction become more common
- Prediction breaks down when the data lacks (around 0), where we expect 100 % fly by

Conclusions

- A Deep Neural Network can approximate the outcome of 3-body encounters whith $\sim$ 70 % accuracy.
- really good for a chaotic system
- if it can be proven that it does not introduce any bias, it could be possibly implemented in large cluster simulations to cut down time.
- It could be used to study outcome probability distribution in more details, to help us understand chaos of 3-body interactions

Can we improve it?

- More simulations -> possible to expand the network
- Experiment with data augmentation
- More accurate Gridsearch or bayesian search of hyperparameters
- Experiment with letting the network overfit
- Weighting loss function over chaoticness of interaction.

Thanks for your attention!