Detecting Orbital Instabilities with Deep Learning

Liz Ellithorpe

Fall 2020

Abstract

A field that is fundamentally connected to how we interpret our own place in the solar system is the study of planetary formation. While there are strong theories concerning the general formation of planetary bodies that condense out of a circumstellar disk, the question of what happens to these planets next is immediately complicated by their presence in a system with greater than 2 bodies. Given the dynamic complexity of planetary systems both old and new, numerical integration of the long-lifetime behavior of planetary systems have become a crucial part of the study of planetary astrophysics. These computer simulations are used to study everything from the evolution of our own solar system to the evolution of large-scale stellar clusters.

Of particular interest is the criterion for the *stability* of planetary systems in the presence of a binary star pair. As the majority of stars spend some of their lives in a binary pair (or a higher order multiple), the study of the stability of planets in the presence of a binary star is an important factor in establishing a holistic model for planet formation.

One solution for sorting through the large amounts of data produced by N-body integration of this dynamically complex, 3+ body problem is the use of a neural net to identify systems as stable or unstable. The goal of this project is to recreate the result from the paper *A machine learns to predict the stability of circumbinary planets* by Lam & Kipping, 2018. This project generated numerical simulations of circumbinary planets using REBOUND and then classified them with a neural net as either 'stable' or 'unstable'. I have access to their fully-trained neural net, and my goal will be to train and then compare my own neural net's capability to theirs.

1 Introduction

Machine Learning techniques are becoming increasingly popular in modern Astrophysics, as very large data sets become the norm. Brute force techniques to analyze and produce these data sets are often computationally expensive, sometimes prohibitively so. One such influx of data comes from exoplanet surveys (Kepler, K2, TESS). These catalogs are giving us a great opportunity for planetary population studies, but characterizing hundreds of thousands exoplanets with finite data is a tall order. A 2018 paper by Christopher Lam and David Kipping titled *A machine learns to predict the stability of circumbinary planets*(3) proposes the use of neural nets to better inform our analysis of the stability of circumbinary planets. This analysis can possibly be used to better inform orbital fitting analysis from exoplanet light curves. That is, the stability of a system calculated by the neural can be used as a bayesian prior for fitting an exoplanet's orbit.

As a foray into my own experiments with neural net architecture, I will seek to recreate the results from Lam & Kipping 2018. While the systems analyzed by this paper are relatively simple and idealized, I will also discuss the pertinence of neural nets to my own work concerning the stability of planets in stellar cluster environments.

2 Planetary Dynamics

2.1 Existing Mysteries

Within our own solar system and outside, the exact cause of disparate orbital dynamics remain somewhat of a mystery. Topics of interest that intersect with planetary dynamics include the origin of so-called 'hot jupiters' (gas giants that orbit extremely close to their host star, closer than they likely formed within a circumstellar disk), the origin of circumbinary planets (i.e., a planet that orbits a binary pair of stars), the effect of wide binaries on planetary systems (the opposite case, where a planetary systems orbits a star within the orbit of an outside binary), the misalignment between the spin axis of a star and the orbital plane of its planets (such as the Kepler-11 system (7)), and the distribution of eccentricities of observed exoplanets (i.e., what causes a planet to become eccentric after it condenses out of a circumstellar disk?). These problems all present the possibility of an elegant union between observational astronomy and numerical simulations to investigate the cause of these dynamics.

2.2 Circumbinary Planets

A particularly interesting orbital configuration is a circumbinary or P-type orbit. The difference between P-type and S-type orbits are shown in Figure 7. Approximately 20 such planets have been confirmed observationally, making them a relatively common phenomenon. The vast majority of confirmed circumbinary planets have been found via the transit method, although at lease one has been found via a gravitational microlensing event. It is thought that the planet and the binary companion condense out of the same circumstellar disk. This is because many of these planets orbit in a configuration very close to being coplanar with the binary companion, suggesting that the planet and binary form in a single disk and migrate out to their current positions (6). This migration is due largely to frictional effects within the disk, stripping the bodies of angular momentum and widening their orbits. In addition to being largely coplanar, circumbinary planets often have extremely tight orbits, keeping close to the central binary. This particular kind of tight, coplanar orbit was studied in depth in the paper *Long-Term Stability of Planets in Binary Systems* by M. Holman and P. Wiegert, and makes up the motivation for the neural network study I emulated in this project.

2.3 Orbital Stability

Holman and Wiegert 1999 is the seminal paper on the stability criterion for circumbinary planets. Similar to what I will do in this work, the authors generated a large parameter space of circumbinary orbits via numerical integration of the three bodies over time, exploring the spacing of the planet and the eccentricity and mass ratio of the central binary star pair. They fit an analytic function to numerical experiment results to obtain a critical semi-major axis, referred to in this work as a_{HW99}. Within this value, planets are unstable and cannot survive on a long term orbit. Outside the orbit, planets can remain stable. However, there are islands of instability close to the critical value not predicted by the Holman and Wiegert study that indicate this stability criterion is not quite cut and dried.

The critical orbit was determined numerically by Holman and Weigert 1999 as a function of the mass ratio of the binary (μ) and the eccentricity of the binary (e). The error bars on each term for the best fitting polynomial are included.(2)

$$\begin{aligned} a_{HW99} = (1.6 \pm 0.04) + (5.1 \pm 0.05)e + (\pm 0.11 - 2.22)e + (4.12 \pm 0.09)\mu \\ (\pm 0.17 - 4.27)e\mu + (\pm 0.11 - 5.09)\mu^2 + (4.61 \pm 0.36)e^2\mu^2 \end{aligned} \tag{1}$$

This value a_{HW99} is actually a ratio of the circumbinary planet's semimajor axis to the interior binary companions semi-major axis, so it is a dimensionless ratio.

2.4 Numerical Integration

The existence of the 3-body problem, the simplest example of many-body physics, already requires the use of numerical integration to solve for long-term evolution of the gravitationally interacting bodies. The very basic ways to consider numerical integration is the solving of the 2nd order differential equation $F = m\frac{d^2x}{dt^2}$ for each body. The more complicated, but commonly used, solution is the use of symplectic integrator to solve Hamilton's equations for the entire system. This symplectic integrator requires the use of a heliocentric frame, where one body plays the role of the central body. Some integrators that are simply using the force of gravity to advance the positions of massive bodies can be accomplished with only a few lines of code, while some that account for close encounters, collisions, tidal effects, etc. are much more complex. To generate the training set for the neural net in question, I emulated the work of Lam and Kipping and used the *Rebound* integration package in Python (5) .

2.5 Rebound & IAS15

The IAS15 integrator(4), available in the Rebound package, is a 15th order numerical solver. Rebound is available for use in both Python and C. In Rebound, the user specifies the integrator (in this case IAS15), the mass and initial positions of the bodies in the system, and the total time of integration. It is by no means the fastest integrator available to Rebound, making for a non-trivial runtime to generate an appropriate amount of traning data, but it does feature adaptive timesteps to select the optimum timestep and handles high eccentricity orbits and close encounters well. For a circumbinary planet at the critical radius, so necessarily close to the central binary, this is needed. IAS15 was developed to address issues faced by symplectic integration schemes that utilize a system's Hamiltonian to solve for the motion of bodies. In particular, symplectic integrators cannot use adaptive timesteps well, as they are designed to work in a heliocentric frame where a single star plays the role of the central coordinate. For systems that include a binary star, like the ones we will look at in this paper, this is problematic. Thus, this integrator was developed for Rebound to evolve N-body systems to very high precision without the use of symplectic (Hamiltonian scheme) integrators.

3 Lam & Kipping 2018

The paper that is the focus of this project (Lam & Kipping 2018) sought to work from the Holman-Wiegert criterion as a jumping off point, but to look more closely at the region of instability located very near to the critical orbit a_{HW99} Thus, the region of parameter space to explore to pin down circumbinary planet stability was a grid of the parameters μ,the binary mass ratio, e, the binary pair's eccentricity and

Parameter	Range
binary mass ratio (μ)	$0.1 \rightarrow 0.5$
binary eccentricity(e)	$0 \rightarrow 0.99$
planetary semi-major axis(a_p)	-$0.77a_{HW99} \rightarrow +0.77a_{HW99}$
binary true anomaly(Φ)	$0 \rightarrow 2\pi$

Table 1: Rebound orbit parameter ranges, evenly sampled to generate a grid of 1 million different integrations.

a_p, the initial semi-major axis of the planet. With rebound, they generated 10 million orbits that sampled all combinations of the range of parameters laid out in Table 4.1. With those integrations completed, the orbits were marked as stable or unstable to be used as the labels in a binary classifier neural net that took in user defined features from the orbits and assessed whether they were likely to be stable or unstable. Lam & Kipping utilized keras(1) to construct, train, and test their neural net. They then tested their net's performance at accurately classifying orbits versus how the Holman-Wiegert criterion would classify them, and found that, especially near the critical orbit, their neural net performed far better ($> 90\%$accuracy vs $< 50\%$ accuracy) than the Holman-Wiegert polynomial. Luckily for me, they also uploaded their trained 'best' neural net to github, with a python wrapper to use it to evaluate the stability of different combinations of orbital parameters. This python code took in a sample set of features: the semimajor axis ratio of the planet compared to the binary, the mass ratio of the binary, and the eccentricity of the binary, and output a single value between 0 and 1. If the value rounded down to 0, the code reported the orbit as unstable. If the value rounded up to 1, the orbit was said to be stable. This ended up being very useful for debugging how I was constructing and evaluating my feature data, as the Lam and Kipping net performance and evaluation of my data should be the baseline to compare the success of my own neural net.

4 My Neural Network

4.1 Training Data

My training set consisted of 1 million rebound orbits, as generating 10 million would have taken on the order of 10+ weeks, which was too long to finish this project on time. Even with the reduced training set size, I had to split the runtime among 2 computers, being my personal laptop and my linux network machine, in order to finish on time. The grid of models was laid out to explore the range of parameter space found in Table 1. The planet begins on a circular ($e_p = 0$), coplanar orbit with a variable initial semi-major axis and was given a mass of $1M_{Jupiter}$. The primary star has a mass of $1M_{\odot}$, and the binary's mass and eccentricity are varied.

As Rebound is written to normalize all input units to make the value of the gravitational constant $G = 1$,

(and gravity is of course independent of scale), I can input the initial parameters of mass of the stars and binary semimajor axis as dimensionless constants. To simplify the analysis in particular I assigned the binary's semimajor axis as 1, as the Holman Wiegert criterion is based on the ratio of the planet's semimajor axis to the binary's. The central star also had a mass of 1, and the binary companion's mass was determined by the desired value of μ. That is, if I was analyzing a system where the binary mass ratio was $\mu = 0.2$, the binary would have a mass of 0.2. Very convenient!

The orbits were run in Rebound for 1000 binary periods. The true anomaly of the binary was varied between 0 and 2π to attempt to account for any resonance effects. If any of the choices of true anomaly resulted in an unstable planetary orbit, that combination of parameters (μ, e, a_p) is marked as unstable in the training data. There are two conditions for the planet which, if met, I marked it as unstable. Firstly, if the radial velocity of the planet with respect to the binary pair is greater than the escape velocity, the planet is considered to leave its orbit and is unstable. Secondly, if the pericenter of the planet becomes less that the pericenter of the binary (i.e., the planet crosses the orbit of the binary), the planet is considered unstable. If either instability condition is met at any point in the integration, the integration is stopped and the next grid cell is evaluated. This is purely to save time so that systems are not continued to evolve once the planet has been lost. Some example finished orbits are shown in Figure 1. The Rebound integrations were run on both the Nielsen network and my own machine to save time, and took a total of 12 days to complete all 1 million.

4.2 Errors in the Training Data

In comparing the stability map of my integrations to the results of Lam and Kipping's neural net, there was a consistent region of parameter space that was being flagged as stable by the Lam and Kipping net, whereas I had recorded those orbits as unstable. The erroneous region is shown in Fig 4 at very high eccentricities. My guess is that my timestep was not as fine as the training data used by Lam and Kipping, and, due to the high eccentricity of the binary, this caused the planet to be flagged as instantaneously inside the pericenter of the binary, where it may not have remained for long. Thus, I labeled some orbits as unstable that were actually stable. Very eccentric orbits have larger pericenter passages compared to a circular orbit with the same semimajor axis, so that is my best guess for why that section of parameter space was erroneously labeled in my analysis. I chose to cut out high eccentricity orbits from the data set to train my neural net, as it takes a non-trivial amount of time to re-generate these orbits and there was a good deal

of interesting structure in the region of $e_{bin} = [0 : 0.8]$ to analyze regardless.

4.3 Architecture

Lam and Kipping include somewhat detailed notes on the construction of their neural net, which I replicated for an initial look at its performance with my own training data (errors and all!)

- 4 input neurons for 4 orbital features
- 6 hidden layers with 48 neurons each
- a one neuron output layer, where 0 is unstable and 1 is stable
- relu activation function for the hidden layers
- sigmoid activation function for the output layer
- keras: binary_crossentropy loss function
- keras: RMSprop optimizer
- 100 training epochs
- a dropout layer(s) with undisclosed parameters

The authors note that they utilized a dropout layer to avoid overfitting. This layer randomly weights a subset of the training data to zero and re-weights the rest of the training data to compensate for the 'lost' data. They did not specify how much of the data was dropped out during each training epoch, so I tried several amounts $(30\%, 20\%, 10\%, 0\%)$ of dropout and used the built in keras.layers.Dropout function to achieve it. Additionally important as the architecture of the network is the features used by the author to train the network. Three of these features are straightforward and are direction informed by the Holman-Weigert polynomial: the mass ratio (μ) of the central binary, the eccentricity (e) of the central binary, and the semimajor axis a_p of the orbiting planet. The fourth feature is based on the occurence of mean motion resonance effects in the parameter space. These can be seen in the example slice of stability space in Fig 3 as jagged triangles that fall outside the Holman-Weigert polynomial. These structures within the parameter space clearly show why the Holman-Weigert polynomial is less reliable as we near the critical orbit. As these structure result from a mean-motion resonance between the planet and the binary companion, the authors parametrized each integration with a resonance proxy, ξ. ξ is a measure of how close the planet and binary are to being on orbits with an integer period ratio between them. The 'period ratio' $((a_{bin}/a_p)^{3/2})$ for each orbit is calculated from Kepler's Laws ($P^2 \propto a^3$), and then the 'floor' of that period ratio is subtracted to

determine how close the period ratio is to an integer value. A resonant configuration is more likely to be unstable as the planet and binary are very close to one another on repeated orbits, leading to more frequent, strong exchange of angular momentum that can destabilize the circumbinary planet. Lam and Kipping note that without the use of the resonance proxy feature, their neural net did not pick up the islands of instability at all (which makes sense, as that resonant behavior was what was left out of the original Holman and Wiegert criterion).

$$\xi = \frac{(a_{bin}/a_p)^{3/2} - \text{floor}(a_{bin}/a_p)^{3/2}}{2} \tag{2}$$

4.4 Performance

The ultimate goal will be to recreate this figure(Fig 2) from Lam & Kipping 2018. The 'slice' they displace in their paper is the subset of orbits with binary mass ratio of $\mu = 0.1$, so that was the subset I used as my toy model to tweak parameters. I began with an exact copy of Lam and Kipping's neural net used on my entire data set, including the erroneously labeled high eccentricity orbits. The initial accuracy on the training set was 90.91%. This accuracy is quite good to begin with, but a huge portion of the mislabelled orbits fell in the most interesting region of parameter space very close to a_{HW99}, making the neural net a very complicated work around to arrive at essentially the exact same conclusion as Holman & Weigert 1999. Accuracy alone therefore is not the most helpful metric to optimize, as I am really working to pick up the 'islands of instability' cause by mean motion resonances. This swath of mislabelled orbits near the critical semimajor axis can be seen in Fig 4. To improve this, I first varied the amount of neurons in each hidden layer. Increased the neurons to 64 tanked the accuracy down to 59%. Decreasing the neurons to 32 per layer resulted in an accuracy of 84%, which isn't terrible, but I decided to stick with the prescribed amount of 48 neurons.

The training set explores a range of planet semi-major axis that is $\pm 33\%$ greater/lesser than a_{HW99}. This is a rather wide range, and a lot of the orbits falls well outside the region of interest where the substructures that are not as well categorized by the Holman-Weigert polynomial fall. Essentially, I decided that orbits at $\pm 20\% a_{HW99}$ are clearly (un)stable and are not helpful in training the neural net, as I believe it was overtraining on the large amounts of orbits way outside the islands of instability. I believe that there could maybe be a more clever way of handling this by perhaps a adding custom weight to orbits well outside the area of interest, but I simply eliminated all training data for orbits at $> \pm 20\% a_{HW99}$. Thus, on subsequent trainings, I 'zoomed' my data set in on the more interesting region of parameter space in hopes the neural net would begin picking up more of those resonance structures. I also cut out the my data that was

labeled incorrectly in the first place (orbits with binary eccentricity greater than 0.8). I played with both the dropout rate and the number of training epochs, but stuck with the basic architecture (i.e., number of neurons and hidden layers) from Lam and Kipping from then on. I found that with my zoomed and trimmed data set using 100 training epochs was a bit too much and accuracy began dropping off with too many training epochs. A plot of the number of epochs vs accuracy is shown in Fig 5. I also played with the dropout rate, as Lam and Kipping imply that they used some amount of dropout during training. In actuality, I found that 0% dropout gave me the best accuracy. I may very well be in danger of overfitting my neural net to my comparatively smaller dataset, but seeing as this project was largely to recreate Lam and Kipping's work in improving on the Holman-Wiegert criterion, that was not something I investigated in depth. I settled on 25 training epochs, did not use a dropout layer, and used a 'zoomed' data set which looked at binary eccentricity from $0 \rightarrow 0.8$ and $\pm 20\%$ of the Holman-Weigert critical orbit. This gave me an accuracy of 96% and recreated the mean motion resonance substructures seen in both my training set and Lam and Kipping's set (see Fig 6 for a slice of parameter space with false negatives and positives labeled).

In playing with the parameters of the net, I was looking to maximize overall accuracy. However, I also recorded the neural net's recall. In all cases, the recall percentage was higher than the accuracy. I believe that this is due to a high rate of false positives, in that the net is picking up all stable configurations and then some. For this reason, I chose accuracy as the parameter of choice to maximize in order to damp down some of the false positives (although getting a high recall was nice too!).

4.5 Further Work

My own research also takes a look at planetary stability. However, as I am focusing on the long lifetime dynamics of planetary systems within a binary pair that is in turn embedded in stellar cluster environments, the parameter space for stability is much more chaotic. It is difficult to pin down one parameter that would guarantee a stellar passage near the planetary system, for instance. While I can make these interactions more likely by creating a denser stellar cluster, in general I think the inherent chaos of predicting such destabilizations from initial conditions make this particular problem unsuitable for neural networks. That being said, I can see a use for classifying destabilizations when they do happen. That is, there is likely a disparate timescale for planetary systems being made unstable by perturbation from a distant binary companion versus sudden, violent perturbations from a close stellar passage of a cluster member. Features surrounding the destabilization, like the change in angular momentum of the planet and the timescale, could be used as features to classify the source of a destabilization.

In a more practical sense, such stability studies as the one conducted by Lam and Kipping are important for establishing a realistic initial parameter space. Starting a simulation with an inherently unstable planet/star configuration can pollute the results of the trial if we are looking at how stellar clusters affect planets, for instance. This indicates an obvious follow-up to their project which would be training a neural net on different orbital arrangements of planets, such as a more eccentric circumbinary planet. While more orbital arrangements lack the obvious comparison to the Holman-Weigert criterion, with a sufficiently sampled parameter space, I can envision a neural net trained on such data would be helpful in establishing a quick initial stability check for very eccentric planets. Very eccentric orbits are a possible precursor to hot jupiters and I think therefore would be a worthwhile set of orbits to explore.

Finally, and this is an application hinted at by Lam and Kipping, I think this work and analysis similar to it could be very useful in augmenting orbital fitting studies from observational data. The stability of a potential orbit as determined by a trained neural net could be used as a Bayesian prior to more quickly and efficiently evaluate a possible orbital configuration for an observed exoplanet. That is, you do not want to 'guess' a set of orbital elements for system that is inherently unstable, but you also don't want to spend the time or computing resources to have to numerically integrate every possible orbital fit. As I mentioned previously, even for such a simple system, this is a nontrivial undertaking.

5 Conclusions

With some tweaking and exclusion of erroneous training data, I achieved a similar amount of accuracy to Lam and Kipping's net. This is very much due to ease of generating data with the 'off-the-shelf' Rebound integration package. That such a powerful and user-friendly N-body integrator is available for free is a great benefit to students and researchers alike. Crafting, training, and evaluating the net with Keras was quite straightforward as it was a relatively simple fully-connected network. This project also highlighted the importance of choosing good features to train your network. The bare minimum orbital elements (e and a_p) were not sufficient to extract the islands of instability in the parameter space, and the network would not have been any more useful than the original Holman-Weigert criterion without the inclusion of the 'resonance proxy' ξ. However, with all four features, and after eliminating some of my erroneous training data, building a neural network to analyze orbital stability was doable in a short time and with the computing resources available to me. I believe that machine learning techniques can be very useful in numerical planetary studies, even though many techniques we discussed this semester are largely designed for datasets with some amount

of error. With an influx of data from both sides, observation and simulation, planetary astrophysics will likely have to adapt these modern computational methods to delve into the complicated mysteries presented by the field.

In the future, two things I would like to optimize in a neural net would be avoiding 'overfitting' to the data and creating a finalized, trained neural net that classifies samples more quickly. Creating the graphs with false negatives and positives involved assessing every point in a slice of parameter space and comparing the neural network output to its actual label, and this took probably 20 minutes to get through about 100,000 orbits. If it is possible to craft a more efficient neural network for such an analysis, I think that would be worthwhile.

6 References

(1) Chollet, F. et al. Keras. https://keras.io. 2015.

(2) Holman, M. & Weigert, P. Long-Term Stability of Planets in Binary Systems. AJ. 1999.

(3) Lam, C. & Kipping, D. A machine learns to predict the stability of circumbinary planets. MNRAS. 2018.

(4) Rein, H. & Spiegel, D. IAS15: a fast, adaptive, high-order integrator for gravitational dynamics, accurate to machine precision over a billion orbits. MNRAS. 2015.

(5) Rein, H. & Liu, S. REBOUND: an open-source multi-purpose N-body code for collisional dynamics. A&A. 2012.

(6) Welsh, W.F. et al. Recent Kepler Results On Circumbinary Planets. IAU Symposium. 2013. **(7)** Spalding, C. and Konstantin, B. Spin-Orbit Misalingment as a driver of the Kepler Dichotomy. ApJ. 2018.

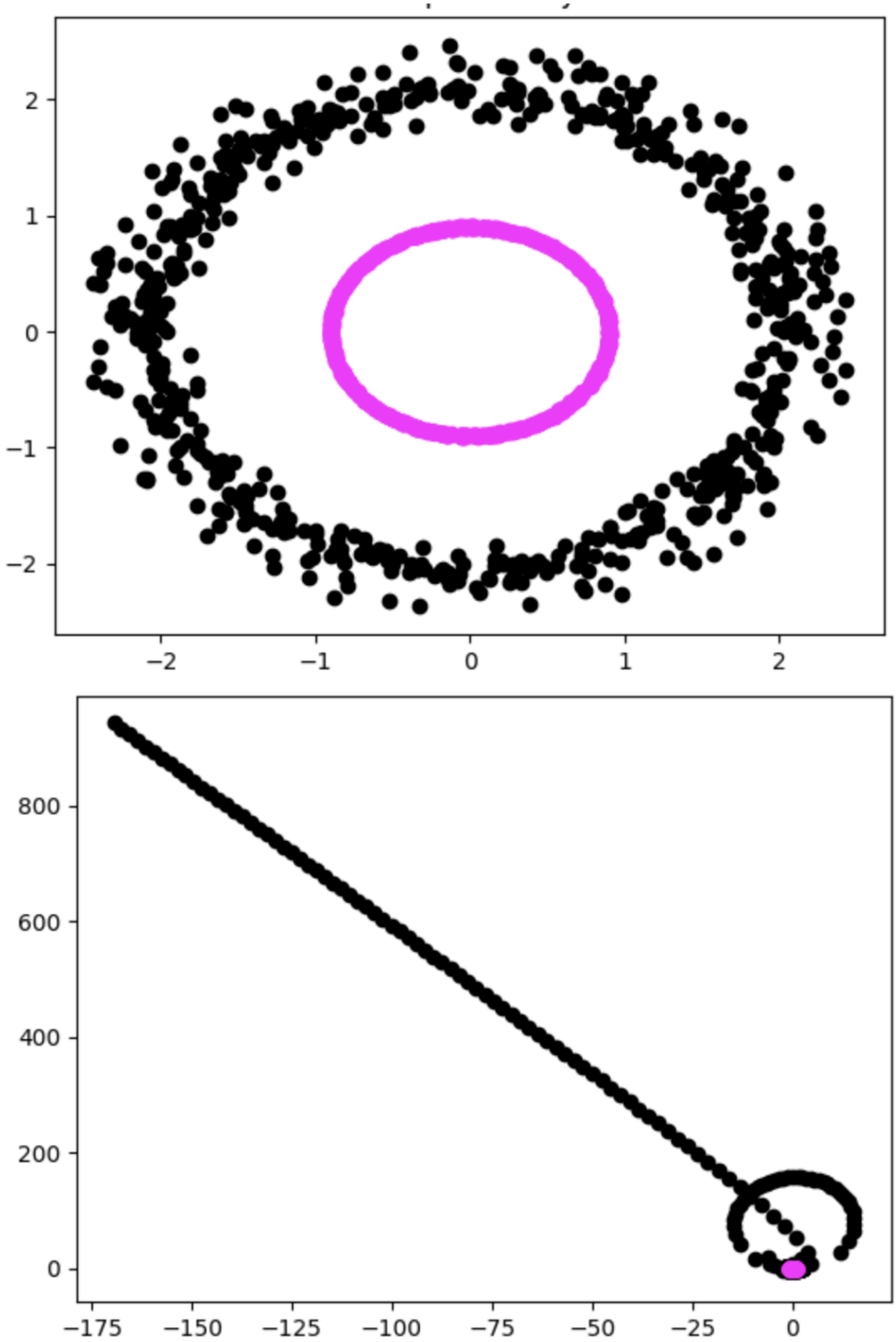

Figure 1: These orbits are an example of a planet near an 'island of instability' that I hope to detect with my neural net. These differ only slightly in initial semi-major axis of the planet, and have the same binary eccentricity and binary mass ratio. The pink dots are the orbit of the binary companion, and the black dots are the planet. One planet, initially slightly closer to the binary, gets ejected (bottom plot). The other remains bound(top). The axes denote a position of the bodies, but are unlabeled as I ultimately used dimensionless quantities to define the semimajor axes of the bodies.

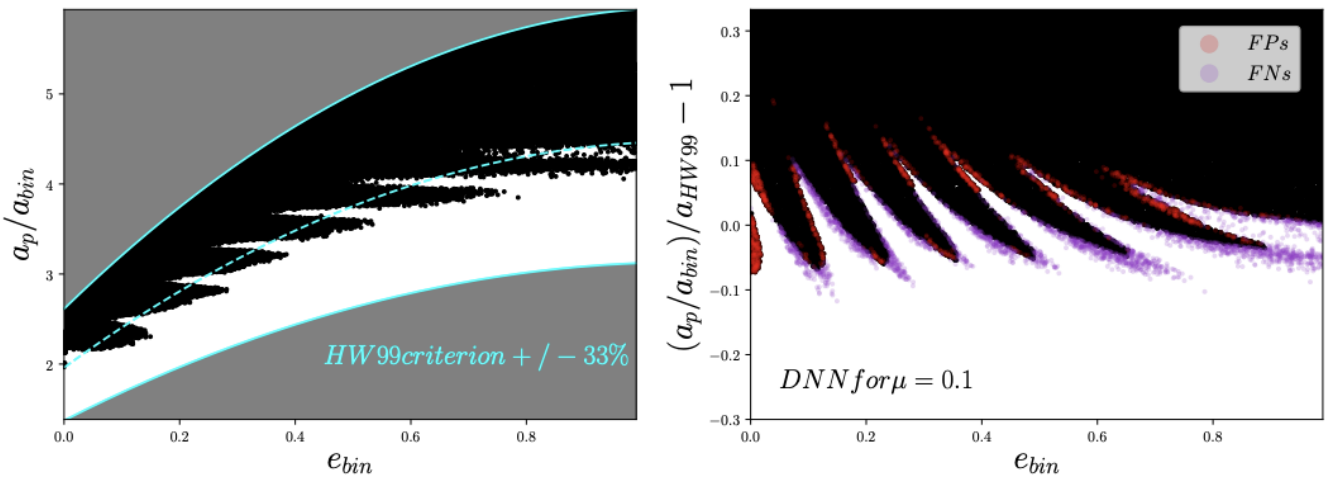

Figure 2: On the left: results on the numerical simulations. Black is unstable, white is stable, and the dash line is the Holman-Weigert critical orbit polynomial for a single value of μ. On the right: predictions of instability from the neural net. The y-axis has been reparametrized with respect to a_{HW99}. This figure was created by Lam and Kipping (3)

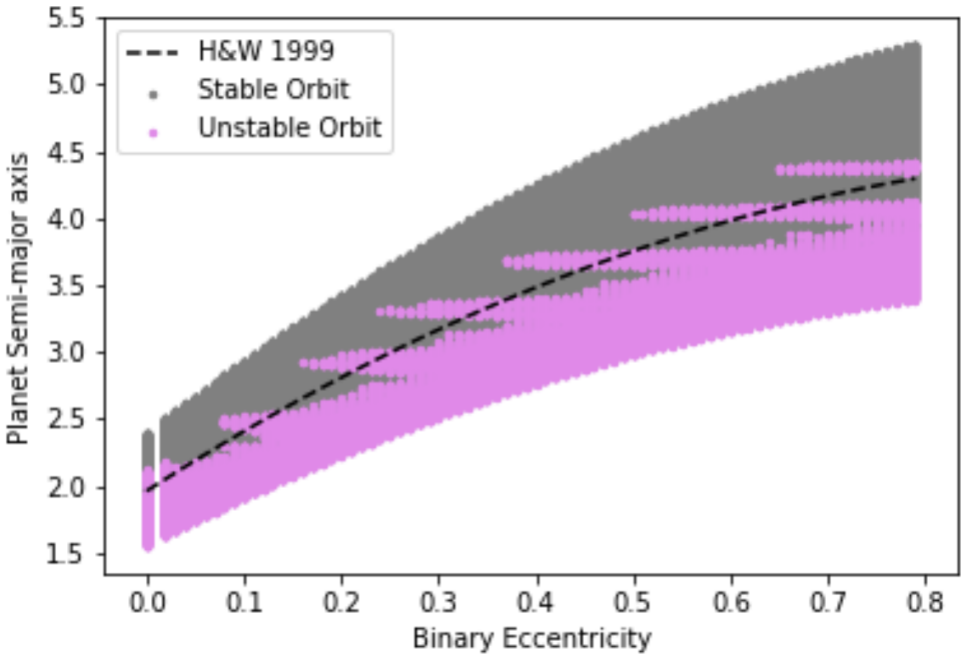

Figure 3: A slice of parameter space for a binary mass ratio of 0.1, depicting stable and unstable planetary systems. The dashed line denotes the Holman-Weigert critical orbit polynomial. Note the 'islands' of stability/instability that violate the Holman-Weigert criterion.

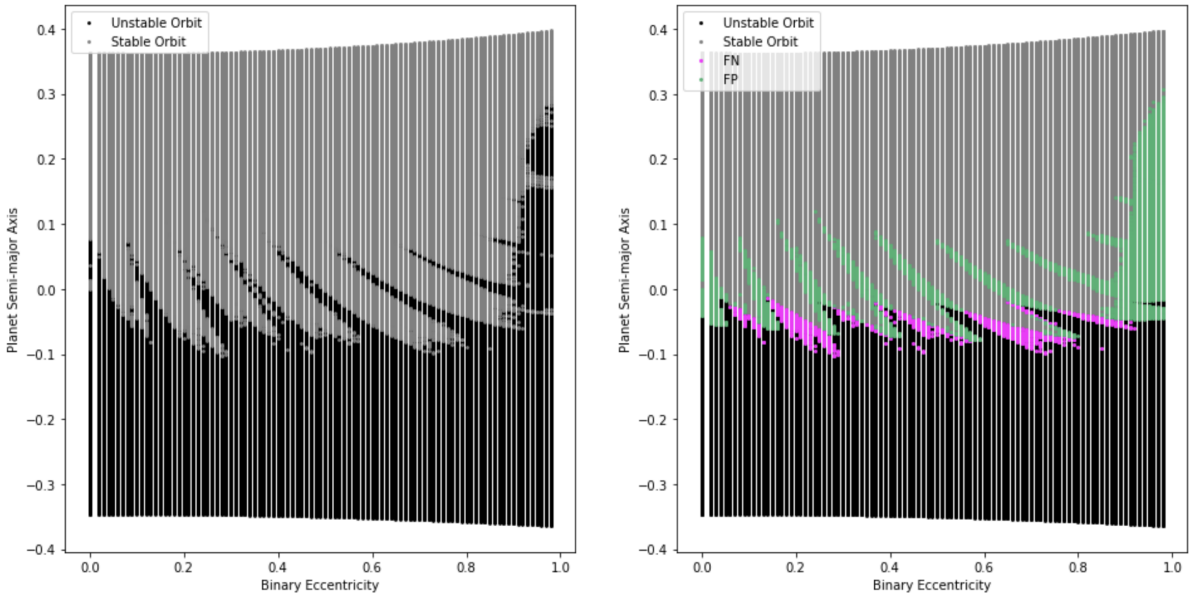

Figure 4: The performance of my first neural net, replicating Lam and Kipping's architecture and using the entirety of parameter space as training data. The green points in the right panel are false positives (i.e., unstable orbits that were characterized as stable by the network) and the pink points are false negatives (the opposite case). While this initial pass picked up some of the interesting structures, the accuracy was significantly diminished in the region of parameter space that we are actually interested in. The planet's semimajor axis has been reparametrized to the ratio $a_p/aHW99$ to better illustrate the substructure. Also note the great deal of mislabelled points at high eccentricity. These points are actually mislabelled in the training data, and were eliminated from further tests.

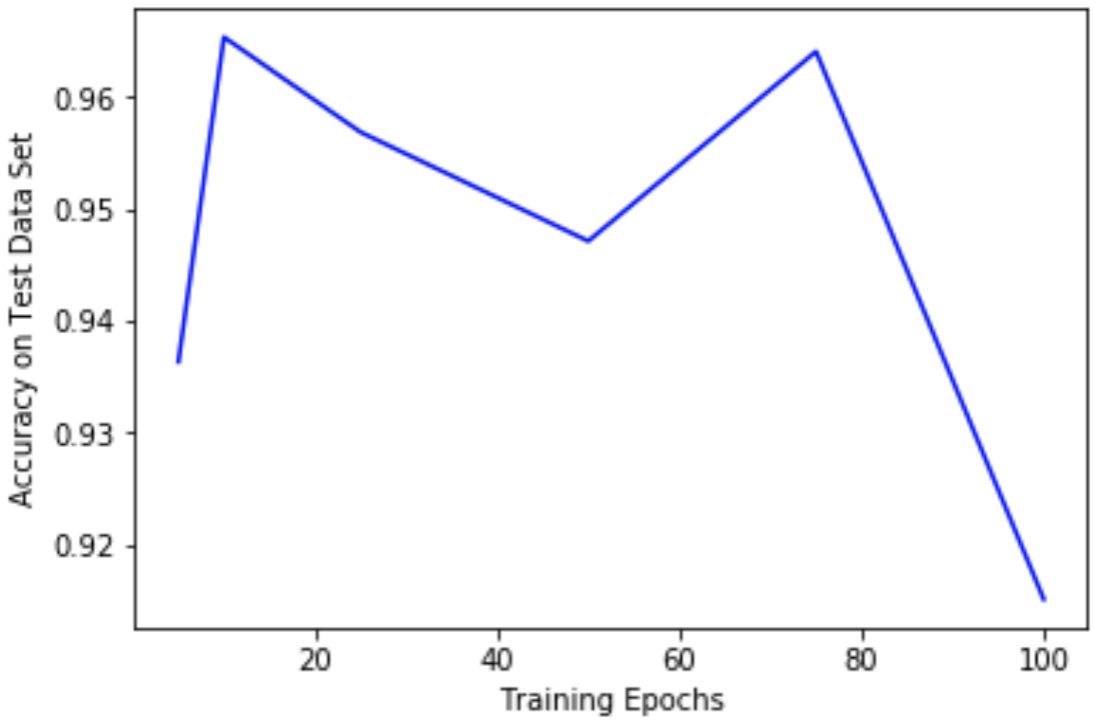

Figure 5: A plot of accuracy vs the number of training epochs for the neural net trained on the 'zoomed' data set.

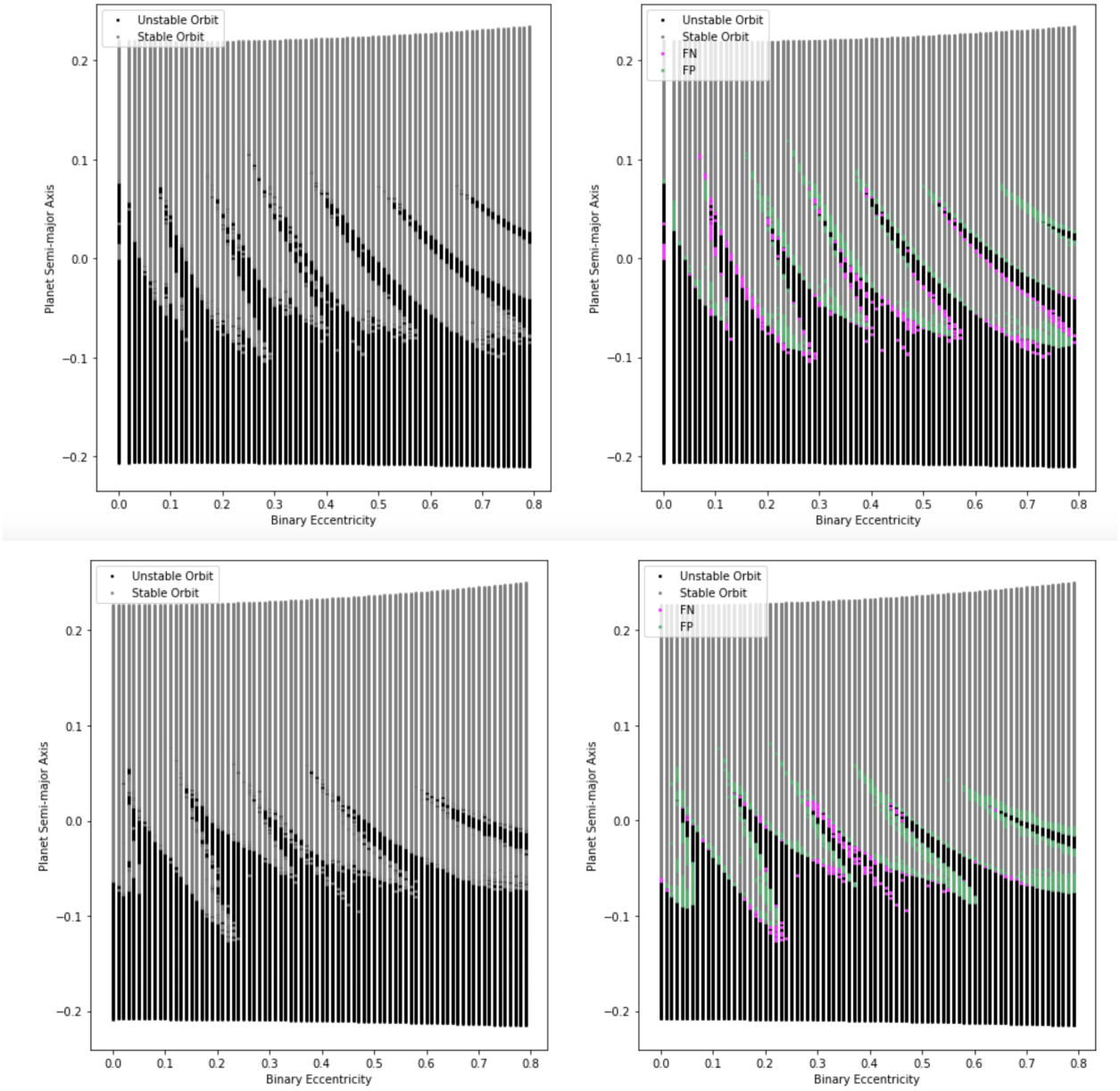

Figure 6: *Top:*The evaluation results on a slice of parameter space with binary mass ratio $\mu = 0.1$. The left panel is the results of the rebound integration, and the right panel is overlaid with the false positives and false negatives generated by the best performing neural network. The y-axis is the planet's semi-major axis reparametrized to its ratio with the Holman-Weigert critical radius (a_p/a_{HW99}).*Bottom:* The same experiment, but with binary mass ratio $\mu = 0.3$. We can see that for both 'slices', a good deal of the structure of the islands of instability is recreated.

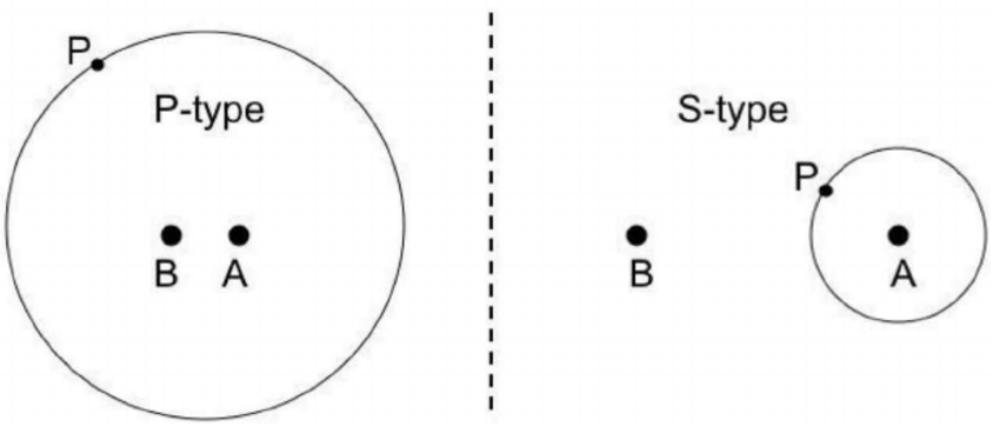

Figure 7: An illustration of P-type and S-type orbits, where A and B are the stars that make up the binary pair, and the point P is the planet.