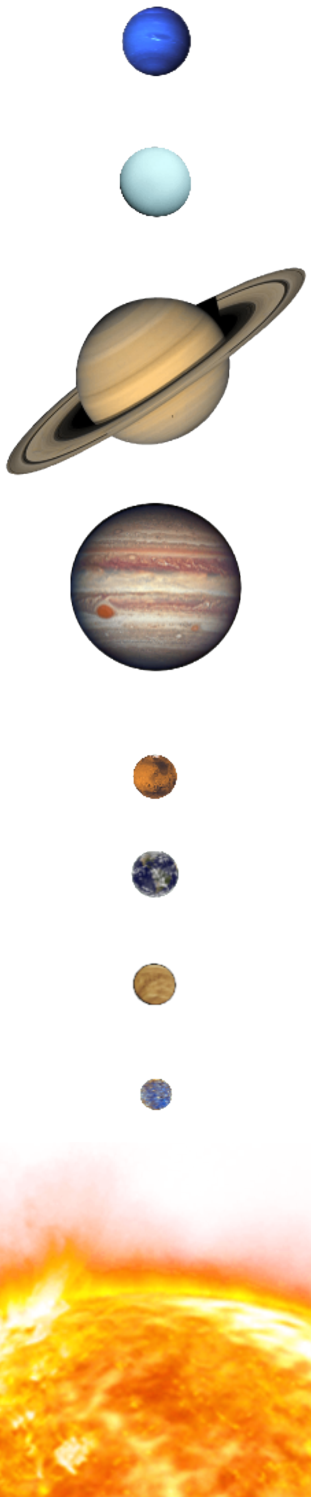
PAI project

Yannick Eyholzer

Physic Application of AI

June 14, 2024



1 Introduction

The goal of the project is to predict the disk features that drives the planet evolutions in the bern model simulations **resume projects - TODO**

2 Theory

2.1 Planet formation and evolution

proof read synthax - TODO Our understanding of things is always limited by how far we can see. This is specially the case in Astrophysics. Hence our comprehension of planets were for a long time based on the solar system planets.

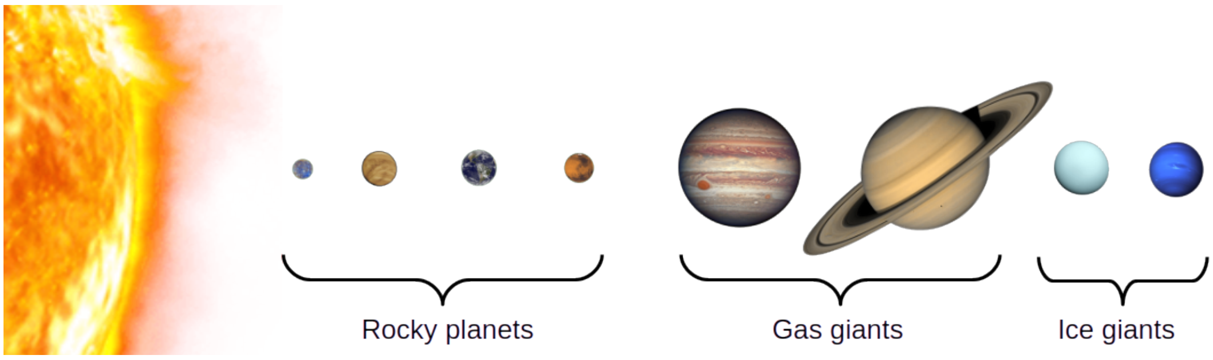


Figure 1: Representation of the solar system showing each planet sorted by distance: *Mercury*, *Venus*, *Earth*, *Mars*, *Jupiter*, *Saturn*, *Uranus* and *Neptune*. Pluto wishes it was here.

From the solar system one can infer three distinct populations: the *rocky planets*, the *gas giants* and the *ice giant*. At the time people started thinking about how would planet outside our solar system - extrasolar planets or exoplanet - look like, it was natural to expect them to look like the one from our solar system. A pletora of different formation models and hypothesis for planets emerged with time. Philosopher, mathematician and astrophysicist worked hard building new theories on top of old ones, discarding proven wrong aspect and keeping the good ones, to result in a theory close to the contemporary one; the nebular hypothesis.

The nebular hypothesis states that solar systems form from the collapse of a giant nebular cloud, being an enormous quantity of gas spanning multiples light years floating in the galaxy. Some external perturbation such as a close supernovae, a close encounter or collision with another cloud would cause the gravitational collapse of the nebular cloud. Most of the mass of the cloud would form the star at the center, and the residual matter would flattened into the protoplanetary disk around the star. Every other celestial bodies such as the planets would form from this disk.

Following this model, gas and ice giant would form in the disk at a distance that is far enough to the star to have solid water. Ice helping the accretion of mass, these planet can acrete more mass thus becoming bigger. This delimitation between solid and liquid water is called the ice line. Before the ice line, the water being liquid and gaseous, the planets forming there are less capable of acreting mass thus being smaller. This is were the rocky planets form. This model fits well the current hierarchical architecture of our solar system, visible in figure 1.

One can imagine the effect on the scientific community of discovering the first exoplanet around a star similar to the sun being a Jupiter-sized planet 10x closer to the sun than Earth, visible as the red dot on the left of figure 2. According to the model at the time, this was impossible. Thus the discovery was first highly criticizing and received with scepticism, but end up rewarding the Nobel prize to Michel Mayor and Didier Queloz for being the first exoplanet around a sun-like star discovered. Later on, a lot more of exoplanets have been detected with different methods, visible on the right of figure 2. These new discoveries, even thought they are bias toward easy discoverable planets, shows that our solar system may not represent the norm of solar systems.

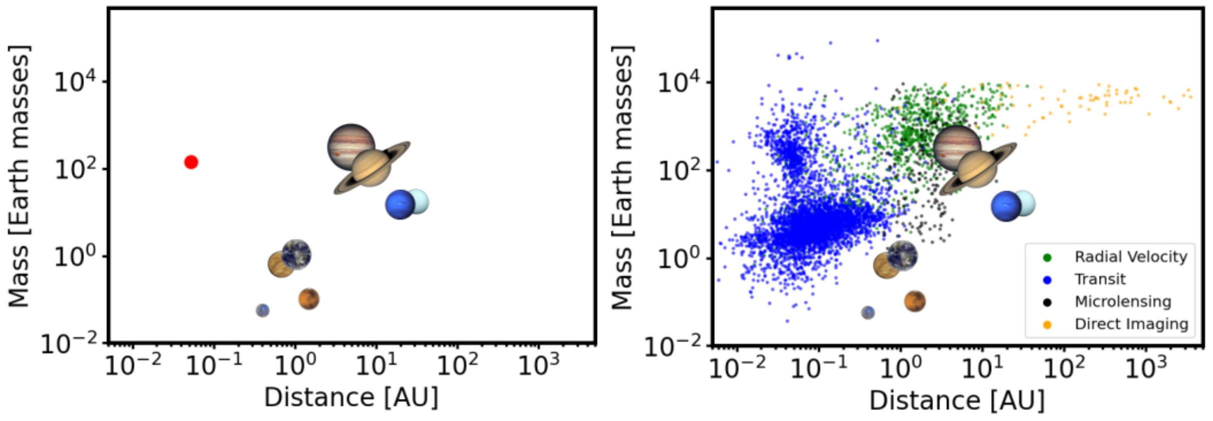


Figure 2: Plots of the distance of the planet to the host star in astronomical unit (AU) (Earth is at 1 AU) as a function of the mass of the planet in Earth mass. The left plot shows the solar system planets with the first discovered Jupiter like planet in red. The right plots adds the current exoplanets detected for a selection of the best 4 methods.

The main drawback in the creation of good planetary models and better understanding in astronomy is the enormous distance and timescale we are dealing with. On the scale of their life, we observe snapshot of systems which light took a few years to hundred of thousand of years to reach us. We cannot observe a system from birth to death, as solar systems probably form for at least Million of year and keep on living until the supernovae of the host star up to Billion of years later. Finally, the relative size and brightness of the planets compared to their host star make them hard to detect, and even harder to completely characterize a solar system.

These obstacles motivated astronomer to develop simulations in addition to the observations and theory already used. These simulations would be based on models, and the results would be compared to observations that would allow to verify how good the model is. They would allow to "observe" a system and their planet during their complete lifespan, to test different formation and evolution mechanism.

2.2 Bern model

The bern model **cite model - TODO** is a simulation process that start from a protoplanetary disk around a star with planetary embryos and simulates the evolutions of the embryos within the disk forming planets. The simulations have been updated multiple times, being more complex each time. It takes into account chemical, gravitational and hydrodynamical process, with N-body interactions. **check bern model paper - TODO**

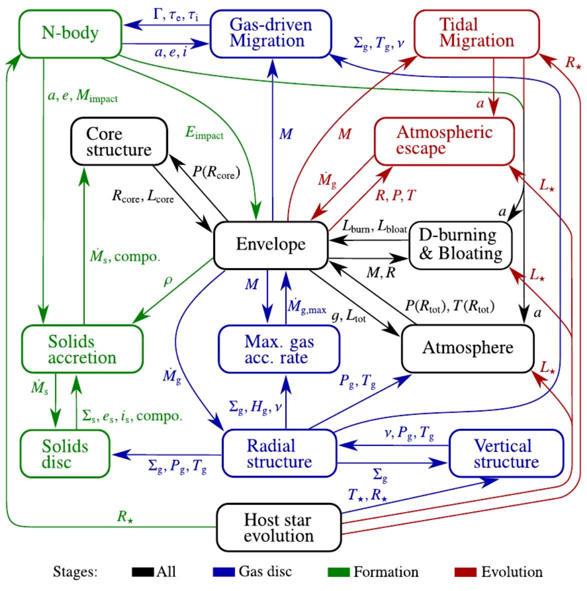


Figure 3: bern model **comment - TODO**

3 Goal & Method

The goal of the project will be to use result of bern model simulations to predict the disk characteristics that are driving the planet formations and evolutions. The approach will be to train a model on the disks feature and to use the resulting formed planet as targets. Then, to determine the features driving the planet formations, the feature importances will be determined.

3.1 The raw dataset

The dataset is the results of bern model simulations for a host star of 1 solar mass, with the evolution of a total of 15'000 disks containing each 20 planetary embryos initially. The dataset posses information on the disk initial condition of the simulation and the characteristics of each planets resulting from the evolution of the disk. that are still in the system.

The disk properties are:

- *Metallicity*
- *Gas disk (Msun)*
- *Solid disk (Mearth)*
- *life time (yr)*
- *luminosity*

The propoerties of each planet are:

- *The total mass (Mearth)*
- *The semi major axis (AU)*
- *the gas to core ratio*
- *the fraction of water*
- *the initial semi major axis* at the beggining of the simulation
- *the radius*

3.2 Processing the dataset

3.2.1 Disk features

Looking at the distribution of the disk features in figure 4, we can see some typical right-skewed distribution. This non-uniformity in the distribution of the features can affect the learning and generalisation capacity of the models, thus a change of representation can be useful. Looking at the figure 5, we can see that the log10 scale is a good choice. This is the representation we will use to train the model with.

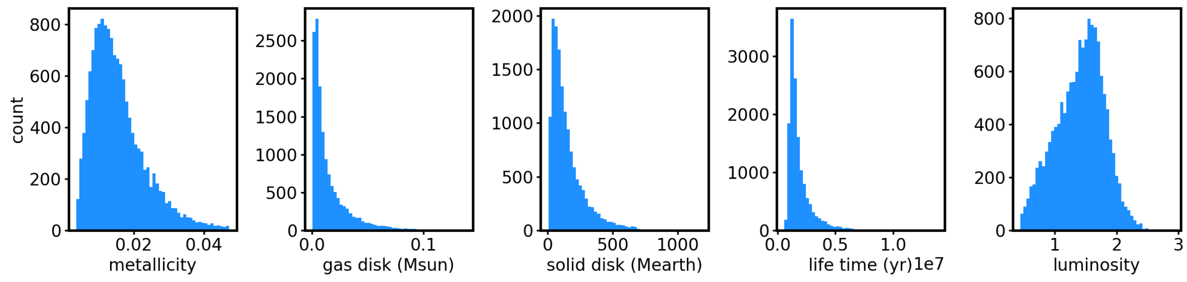


Figure 4: Distribution of the disk features

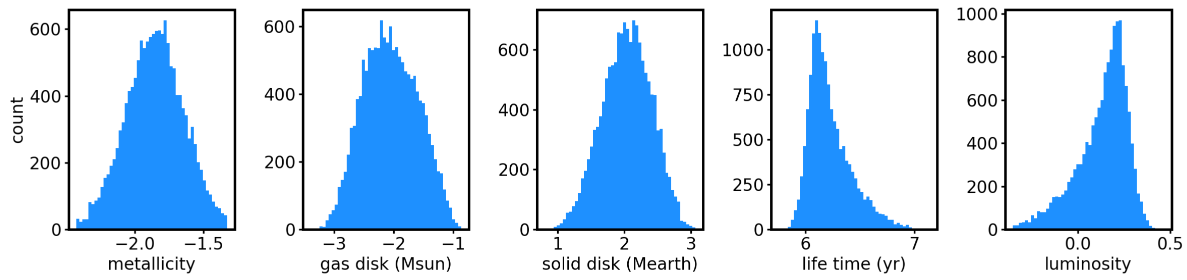


Figure 5: Distribution of the disk features in log10 scale

3.2.2 Planet features

The first step with the planets properties is to address the variable number of planets for different disk. As each disk does not necessarily produce the same number of planet, some collide, merge, or are ejected. In addition to the variable number of planet, their occurrences are not orderer. Hence the input is constant and ordered, as each disk possesses the same number of features, but the target are variable in size and non-ordered. To solve this problem two approachs are possible; either creating a sophisticated model that could deal with theses complex target, or to simplify the target to use much common model.

Model-wise, it would need to be able to generate a dynamical number of planets, and the loss would need to be able to compare each predicted planet to the corresponding true target. There would be a lot of approach, such as the use of padding for the target, capping the number of planet to the maximum one, using Graph Neural Network to solve the ordering issue, or using a recurence system to make the model predict planet one by one, keeping information on the already predicted planet, training a diffusion model on the planets themselves with increasing noise and using the disk feature as a form of seed, etc.. However this task seemed complicated, thus the project will use the target-wise approach.

The second approach would be to generalize the knowledge of the planetary system into fix and ordered information. To start simply, two generalisation of the system will be use:

- *The number of planet* in the system, calculated by simply counting the number of planet left after the disk evolutions.
- *The total mass of the planets*, calculated by summing the mass of each planet present in the system.
- **others if time - TODO**

In that way, whatever the number of planets and their order, the generalisation is always constant and ordered. This allow to use much simpler model such as Neural Network.

3.3 The models

Thanks to the generalisation of the planets informations ,the model used will a simple Neural Network.

loss - TODO **optim - TODO**

3.4 Important features

To retrieve the feature that are important in the evolution of the planets, we can retrieve the feature importances. In this project two methods are used.

The first method is to use feature permutation. The model is normally trained on the data, then it is tested on a test dataset were the feature of the disks have been permuted *between the disks*. Thus the model will have to predict the target based on disks that have have one of their feature exchanged with all the other disks. For each feature, the permutation is applied and then the performance are measured, and finally we can analyse which feature affected the performance the most, which should be the most important feature.

The second method is to add linear layer at the beggining of the neural network that is connected in a 1-to-1 manner, each input connects to one neurone. Thus, this is possible to analyse the weight of this layer to see which input the model weighted the most as "most important".

3.5 Trainings and tests

say train val test etc - TODO s

4 Results

4.1 Architecture and parameters explorations

The results for that part have been produce by training the models multiple times with the same parameters **say n time - TODO** to take an average of the performances

4.1.1 lr test

The first test to do is to see how the learning rate affects the efficiency of the models, as higher learning rate would allow to train the model for a shorter time. This test was done for 3 different numbers of hidden layer, having each 32 neurones, with a learning rate going from $5 \cdot 10^{-5}$ to $1 \cdot 10^{-1}$ with a log-scaled step of $5 \cdot 10^{0}$.

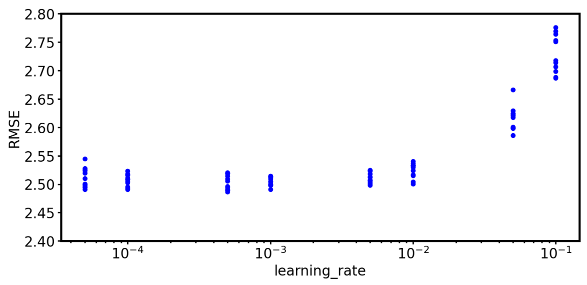


Figure 6: Results of the performances of the models on the test dataset as a function of the learning rate

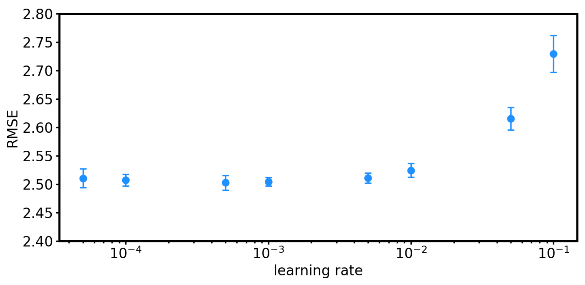


Figure 7: Same plot as the left one, with error bar showing the deviation of the performance from the mean.

Figure 6 shows the Root Mean Squared Error (RMSE) of the model on the test dataset as a function of the learning rate of the training. The figure 7 is the same plot, with the scattering of the plot showed with errorbar, while the point shows the mean. The model were trained for 1000 epochs. We see that for a learning rate of 10^{-3}, the performances do not increase so much anymore, thus we will use that learning rate for the rest of tests. In addition, looking at the loss on figure 8 for the lowest training with a learning rate of $5 \cdot 10^{-5}$, the loss already converge at 100 epochs, thus we will also use that epoch as maximum training epoch.

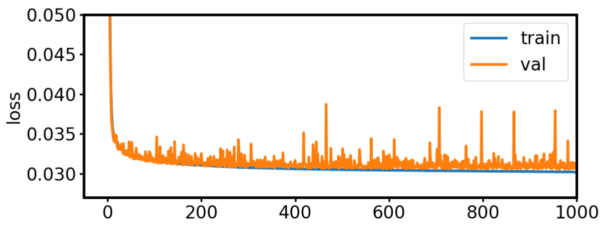


Figure 8: Train and validation loss of a model with two hidden layer of 32 neurones each, with a learning rate of $5 \cdot 10^{-5}$.

4.1.2 batchsize test

We then test the batchsize as bigger batch allows the training to go faster, for the same model used in the previous text, with a learning rate of 10^{-3} and 100 epochs.

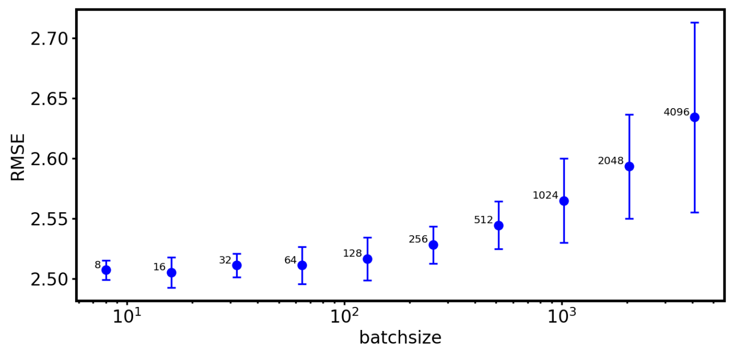


Figure 9: Performances of the models as a function of the batchsize.

Figure 10 shows that too big batchsize actually decrease the performance of the models. The decrease of performance being negligible below a batchsize of 64, we will use a batchsize of 32 for the rest of the tests.

4.1.3 Number of layer tests

We will now test the number of layer. The test are done for 100 epochs, a learning rate of 10^{-3} and a batchsize of 32. The number of tested layer goes between 1 and 8, for neural layers of 32 and 64 neurones (i.e ([32], [32,32], [32,32,32],..) and ([64], [64,64], [64,64,64], ..)).

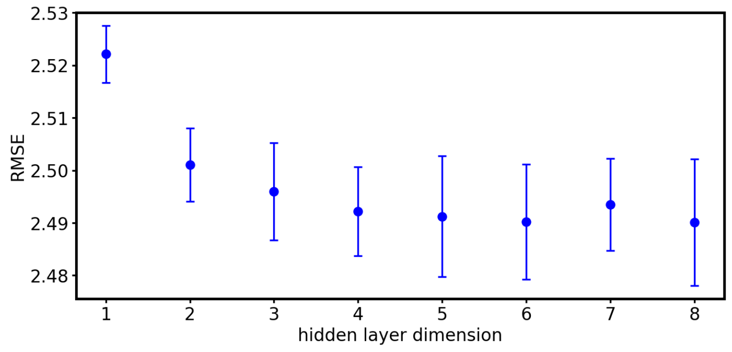


Figure 10: Performances of the models as a function of the batchsize.

4.1.4 depth test

4.2 Feature importances

5 Conclusion