Orbital velocity and distance from sun analysis of planets in solar system

Tony Liao - UQ

Contents

1.Introduction

1.1 Project aims

The aim of this project is to analyse and construct a unique visualisation of orbital velocity and distance of planets from sun in the solar system. To achieve the aim of this project, the three visualised techniques has been included:

- Simple linear regression
- Multiple linear regression
- 3D interactive visualisation

The dataset for this project includes:

- Diameter of planets
- Orbital velocity of planets
- Distance of each planet from sun

1.12 Motivation

This project will incorporate a variety of visualisations with these datasets to provide a better understanding of the relationship between the orbital velocity of planets and the distance from the sun. Unfortunately, very few interactive 3D visualisation and linear regression related to this analysis can be found online. This project will supply simple linear regression and more focus on complex 3D interactive visualisation and multiple linear regression to find the relationship between velocity and distance from the sun of planets. The pattern revealed by these visualisations can provide insight into how the different distances from the sun also affect the speed of planets.

2 Data source

I will primarily access the data from NASA's websites as they appear more reliable. They are gathered by certificated scientists with detailed information on the planets, including their mass, distance from the Sun, and other relevant parameters. For example, to obtain data for a specific planet, I can access the NASA planetary fact sheet to retrieve data on its information of orbital velocity, diameter, and other relevant details of planets, as shown in Figure 1.

Planetary Fact Sheet - Metric

	MERCURY	VENUS	EARTH	MOON	MARS	JUPITER	SATURN	URANUS	NEPTUNE	PLUTO
Mass (10^{24}kg)	0.330	4.87	5.97	0.073	0.642	1898	568	86.8	102	0.0130
Diameter (km)	4879	12,104	12,756	3475	6792	142,984	120,536	51,118	49,528	2376
Density (kg/m^3)	5429	5243	5514	3340	3934	1326	687	1270	1638	1850
Gravity (m/s^2)	3.7	8.9	9.8	1.6	3.7	23.1	9.0	8.7	11.0	0.7
Escape Velocity (km/s)	4.3	10.4	11.2	2.4	5.0	59.5	35.5	21.3	23.5	1.3
Rotation Period (hours)	1407.6	-5832.5	23.9	655.7	24.6	9.9	10.7	-17.2	16.1	-153.3
Length of Day (hours)	4222.6	2802.0	24.0	708.7	24.7	9.9	10.7	17.2	16.1	153.3
Distance from Sun (10^6 km)	57.9	108.2	149.6	0.384*	228.0	778.5	1432.0	2867.0	4515.0	5906.4
Perihelion (10^6 km)	46.0	107.5	147.1	0.363*	206.7	740.6	1357.6	2732.7	4471.1	4436.8
Aphelion (10^6 km)	69.8	108.9	152.1	0.406*	249.3	816.4	1506.5	3001.4	4558.9	7375.9
Orbital Period (days)	88.0	224.7	365.2	27.3*	687.0	4331	10,747	30,589	59,800	90,560
Orbital Velocity (km/s)	47.4	35.0	29.8	1.0*	24.1	13.1	9.7	6.8	5.4	4.7

Figure 1 The planetary fact sheet by NASA

The planetary fact sheet was last updated on the 11th of February 2023 by NASA, which is recent and reliable.

2.1 Data format

Various data formats are used for relevant information on the properties of each planet. While considering the complexity of the data structures. JSON format is more suitable for representing this nested or hierarchical data structure, which is a lightweight, text-based format commonly used for data exchange between web applications and can be easily parsed and edited using programming languages such as Python. Thus, I will mainly use the data in JSON format to achieve these purposes. JSON file format makes storing each planet's data in the solar system simpler. It has the collection of key-value pairs of each property, with the key representing the name of the planet and each associated value containing the properties such as mass and distance.

A JSON Format that contained the planet's information:

```
{
  "Mercury": {
        "mass_10_to_power_24_kg": 0.33,
        "orbital_period_days ": 87.969,
        "surface_gravity": 3.7,
        "distance_from_sun_10_to_power_6_km": 57.9
   }
   "Earth": {
        "mass_10_to_power_24_kg": 5.97,
        "orbital_period_days ": 365.2,
        "surface_gravity": 9.8,
        "distance_from_sun_10_to_power_6_km": 147.1
   }
}
```

The code block that contained planet's information is inside the inner curly brackets with each key value pairs.

3. Data processing

3.1 Tools used

The tools that are used for this project:

- Matplotlib is a standard Python library for plotting.
- Python built-in package json to deal with a file in JSON format.
- Python seaborn library based on matplotlib provides a high-level interface for drawing statistical graphics.
- Python ursina library that provides built-in animation and is easy to use for making 3D visualisation.
- Wolfram Alpha is an answer engine that computes the answer from externally sourced data.
- Python pandas library is a fast, easy to use for open source data manipulation and analysis.
- Python numpy library supports large, multi-dimensional arrays and matrices, along with high level mathematical functions to works on arrays.
- Python scipy library used for scientific computing and easy to use for statistics operations.

3.2 JSON Format Conversion

To convert the data to JSON format, a more efficient way to deal with this conversion is to use `JSON.dump()` function in Python. This was done by opening the file in Python, reading each line in a file, using list sliced to get each key-value pair and saving each key-value pair into a dictionary variable. It then passed that variable as an argument to `JSON.dump()` function.

3.3 Gravitational force of each planet

To accomplish the multiple linear regression, I will need one more independent variable, which is the gravitational force of each planet. To compute the data of gravitational force between planets and the sun in the solar system, I have used Newton's law of universal gravitation formula:

$$F_g = G\frac{m_1 m_2}{r_2}$$

Where F_g is the target planet's gravitational force, G is the gravitational constant, r is the distance from sun. To get the gravitational force between the sun and the planet, m_1 will be present as the mass of planet to be measured and m_2 will be the mass of sun.

3.4 Formula for Regression analysis

This project is aimed to design linear regression for data on the relationship between the planet's orbital velocity and distance from the sun. The linear equation is required to compute and construct both simple and multiple linear regression models. With this linear equation, the model of linear regression should predict the velocity of a planet with minimum error:

$$Y = \beta_0 + \beta_1 X + \epsilon$$

Where Y is the dependent value, β_0 is the constant and β_1 is the first regression coefficient for X and X is the independent variable. ϵ represents as error of estimate.

For multiple linear regression, I just simply add one more regression coefficient β_2 and independent variable X_2:

$$Y = \beta_0 + \beta_1 X + \beta_2 X_2 + \epsilon$$

For each additional variable, the coefficient of each independent variable represents as the degree of change in predict value y.

Then I used least squares method to find the best fitting line for the data of planet's velocity and distance from the sun. It minimises the sum of these squared distances with the line passes closest to the set of points. To determine the best fitting line, the β_0 and β_1 are estimate based on the data at hand. The estimates can be computed by hand with these formulas:

$$\begin{aligned}\widehat{\beta}_1 &= \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2} \\ &= \frac{\left(\sum_{i=1}^{n} x_i y_i\right) - n\bar{x}\bar{y}}{\sum_{i=1}^{n}(x_i - \bar{x})^2}\end{aligned}$$

$$\widehat{\beta}_0 = \bar{y} - \widehat{\beta}_1 \bar{x}$$

Where $\bar{y}$ denoting the sample mean of y and $\bar{x}$ denoting the sample mean of x.

3.5 Data unit down scaling

To get 3D interactive visualization to work on smaller screen space, the data unit length downscaling is one of the solutions. Choosing an appropriate scale for the data unit is extremely important as it is vital for showing valuable information about the different distances between the sun and the planets. To achieve this, I first take Mercury's distance from the sun, 57.9e6 km, as 1, with cm as the downscaled unit. I then calculated the actual differences in distance from the sun of other planets based on the downscale unit length of Mercury's distance from the sun. To calculate the distance from the sun between Earth and Mercury, I divided Mercury's distance from the sun by the Earth's distance from the sun. Then get multiple and multiply it to the Mercury's downscaled unit. For example, Earth's actual distance from the sun is approximately 2.58 times Mercury's exact distance from the sun. Thus, if downscaled Mercury's distance from the sun is 1 cm for the distance from the sun, then for Earth, it will be converted to approximately 2.58 cm on a small screen as shown in figure 2.

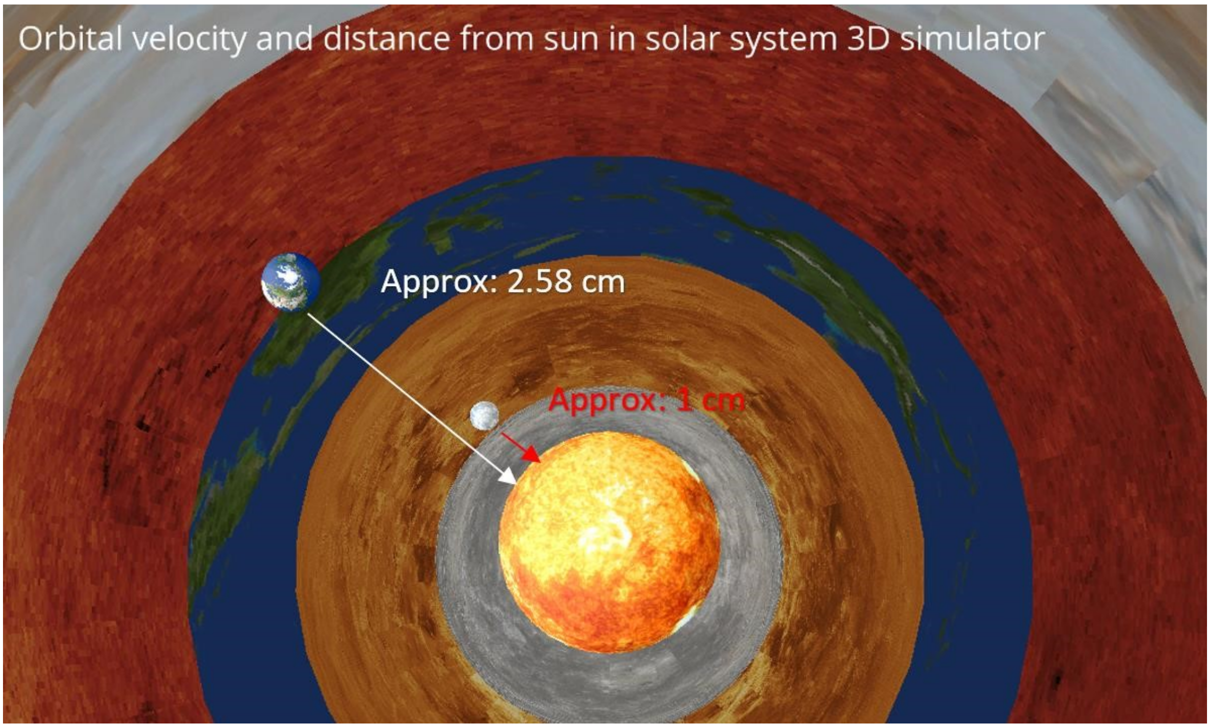


Figure 2 The downscaled unit length of Mercury's distance from the sun and compared to the Earth's distance from the sun

4. Data Analysis

4.1 Direct relationship of velocity and distance

The first visualisation was a simple linear regression showing the direct relationship between the planet's velocity and the distance from the sun. This is useful for finding the relationship between two variables. If the independent variable can accurately express the explanatory variable, thus their relationship will be deterministic. Figure 3 shows that the planet's orbital velocity is the explanatory variable and is accurately expressed by the distance from the sun. This showed that the planet's velocity would be faster at a closer distance from the sun.

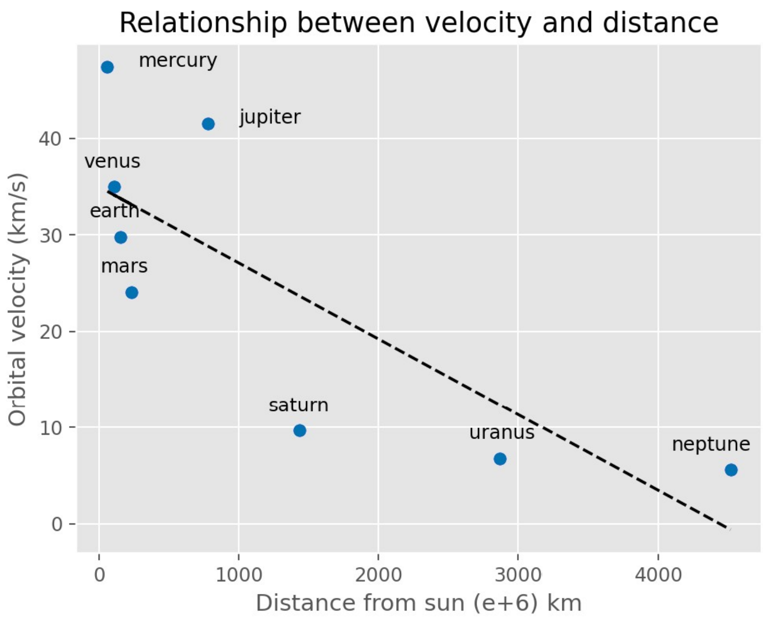


Figure 3 The simple linear regression of relationship between orbital velocity of planets and distance from sun

This is well demonstrated that Mercury will have the fastest speed at approximately 47 km per second and is closest to the sun with approximately 50 (e6) km where (e6) denote the scientific noting of 10 to the power of 6 which is large data unit.

To visualise the line of best fit of simple linear regression, I have added the line with a dashed line. I have also added a name for each point to distinguish each planet from each point better. The figure clearly shows the strong negative relationship from the line of best fit.

4.2 Comparing different variable

As there are only two variables, it is hard to make a more complex visualisation such as multiple linear regression. I added one additional independent variable, which represents the gravitational force of each planet between the sun and compares the relationship between the velocity and distance from the sun.

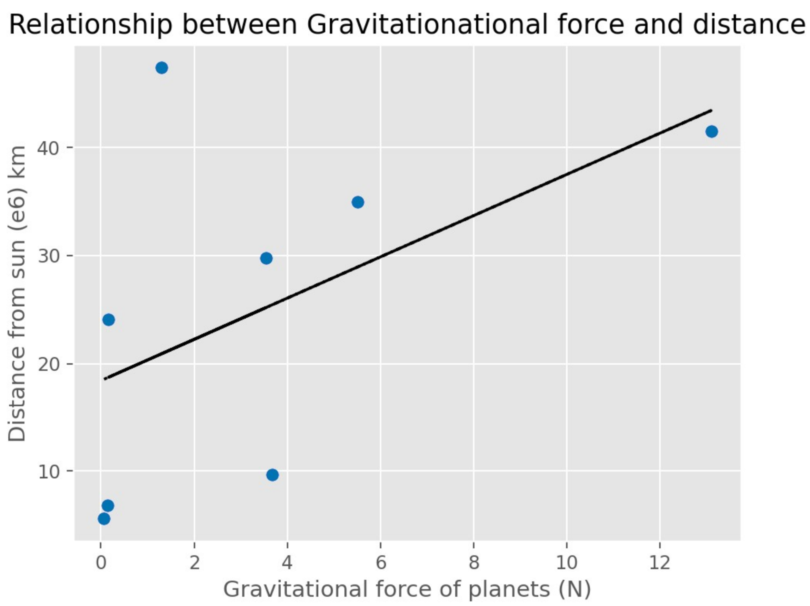


Figure 4 The simple linear regression of relationship between gravitation force of planets and distance from sun

This simple linear regression of gravitational force and planets show a possible strong positive relationship in Figure 4. I then faced a problem in which the relationship did not explain and expressed accurately. Some points have long distances but with less gravitational force. For example, figure 4 shows a point with the furthest distance between 0 and 2 gravitational forces on the x axis. Thus, it is not very accurate to express its strong positive relationship by looking at these two variables individually. To resolve this problem, instead of fitting the straight line through the points on a simple linear regression, it is more suitable to fit a plane through the points with another independent variable and this is where the multiple linear regression comes into place.

4.3 Relationship between each variable

Before going deeper into the multiple linear regression model, it is better to understand the relationship between each variable in the data. To accomplish this, I first used Python's `pairplot` function from the `seaborn` library to plot subplots for each relationship between gravitational force, the velocity of planets, and distance from the sun.

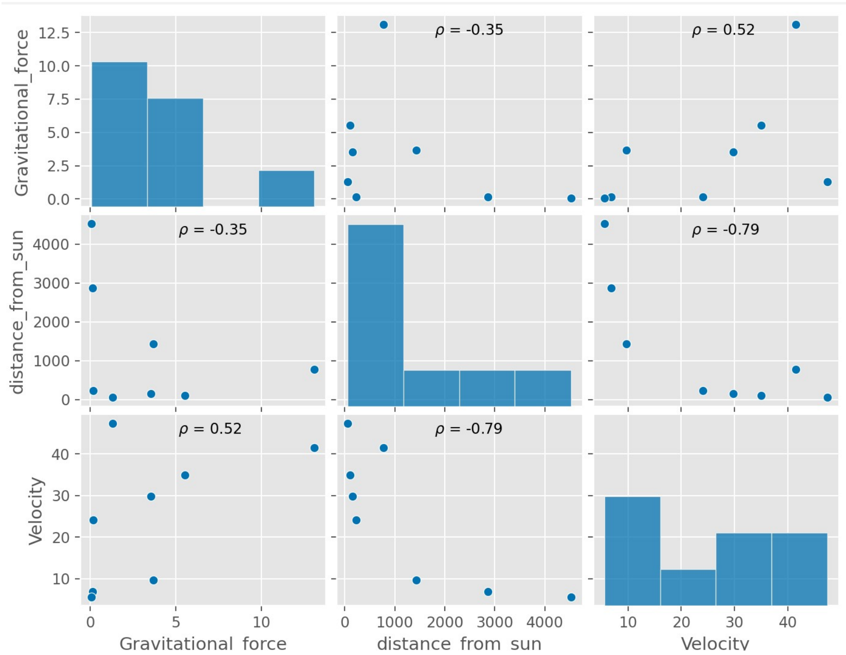


Figure 5 The 2D plots of each relationship between orbital velocity of planets, gravitational force and distance from sun.

The pairs plot by itself immediately shows us the insights of variables. The histogram on the diagonal in the pair plot shows the distribution of a single variable. The scatter plots with labelled Pearson corrections immediately show the relationship between the two variables. From figure 5, it is clear that the second column and third row show the scatter plot of the distance from the sun and the planet's velocity. They also show that it has a strong negative Pearson correction, which is negatively correlated. This is expected that speed will be slower with a long distance from the sun.

4.4 3D visualisation planet surface textures

To provide a better visual, I used eight different colours for each planet in 3D to distinguish between eight planets and the planes to represent their distance from the sun. Unfortunately, there were planets such as Uranus and Neptune. Both have a very close RGB range which is blue and cyan. Given very little difference between these colours, it is hard to distinguish between them. To address this problem, I have utilised the different textures corresponding to each planet and applied them to the circular planes and each planet's sphere model.

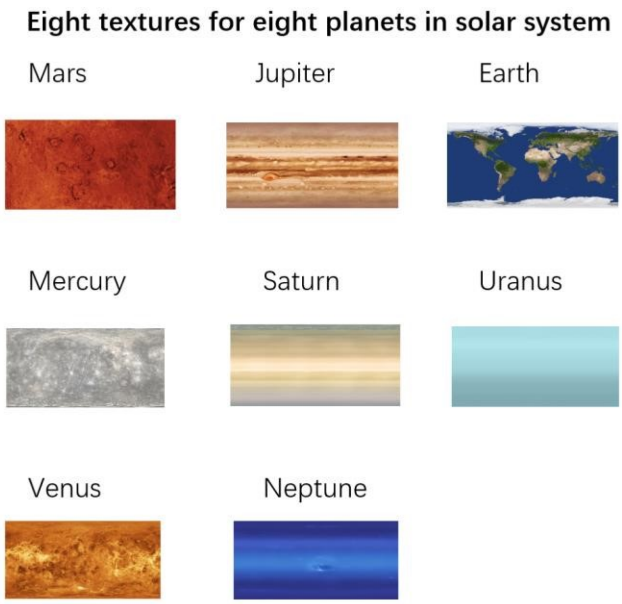


Figure 6 The eight surface textures for eight planets in solar system

This is an appropriate way to distinguish even with colour blindness. This adjustment makes it more comfortable to identify each planet and planet's distance by their surface texture.

4.5 Orbital movement of each planet in 3D visual

To create orbital movement for all planets excluding the sun is simple, I have used the equation of circle as shown in Figure 7 to achieve this creation, the orbital movement basically just a flat circle.

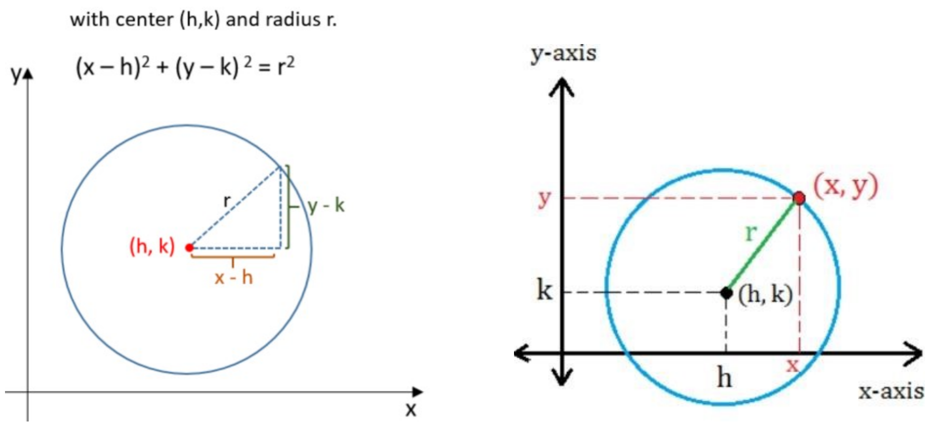


Figure 7 The equation of the circle

Where r is the radius of circle, (h, k) is the coordinates of centre of the circle. I then choose a suitable radius that won't overlap the other planet's orbit and set all centre positions of the rotation as the sun's position.

5. Result

5.1 Multiple Linear Regression in 3D

To accomplish the visualisation for multiple linear regression, I have used several high-level libraries in Python to compute a 3D model of this visualisation with the assistance of the formulas of regression analysis. From Figure 8, the plane of best fit shows the relationship between distance and the other two variables. This plot suggests that when the distance is approximately 1500 km, and the gravitational force is approximately about 12 newtons, the velocity is approximately at around 45 km per second.

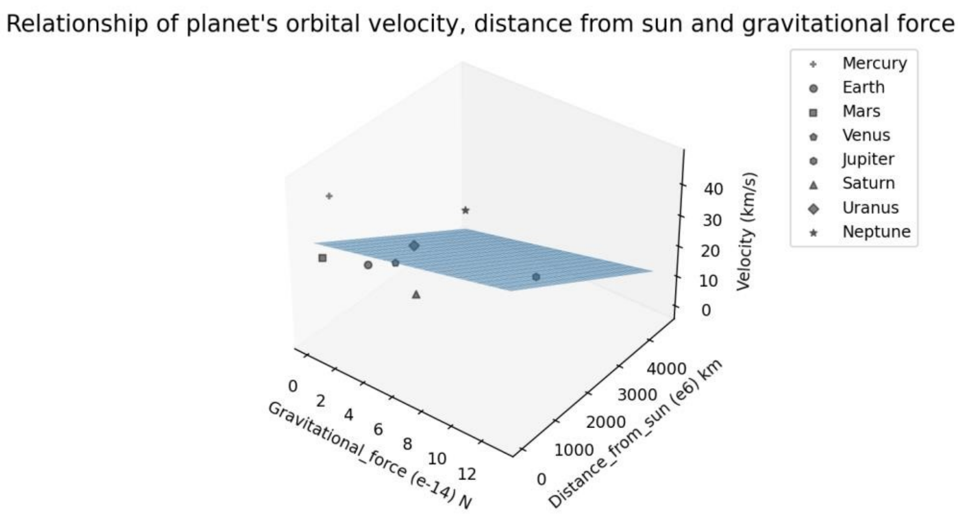
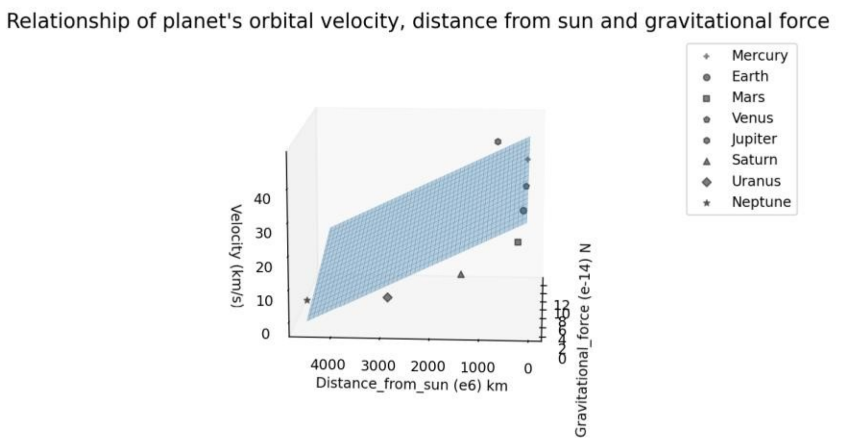


Figure 8 The multiple linear regression of relationship between orbital velocity of planets and distance from sun

The 3D multiple linear regression also improves the visualise of the relationship between three variables. After the view of the plot was switched to the left-hand side, there was a negative slope indicating a robust negative relationship between velocity and distance. This is to be expected as the planet gets closer to the sun. It will need to orbit faster to avoid the gravitational force which pulls the planet into the sun.

Left hand side view:



I have added different markers for each point and a legend to label each point with the planet's name to make it easily viewable and distinguish for colour blindness. This is also designed for broader use, such as print and on-screen viewing.

5.2 3D visual for each planet's distance from sun

A simple circular plane for each planet is used to assist the visual of the planet's distance from the sun and provide better visibility for their orbital movement, as shown in Figure 9 and 10. Using this circular plane with red arrow label, it gives approximation of how far the planet is from the sun. Each red arrow label will have a different length, calculated based on the real data and scaled down to fit a smaller screen.

Figure 9 The red arrow label represents the accurate approximation of distance from sun for each planet in solar system on circular plane

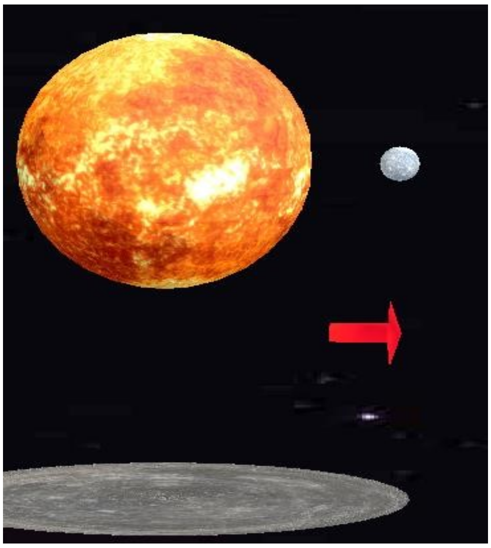


Figure 10 The side view of distance from sun for Mercury with red arrow label

It is important to note that as the planet is further away from the sun, its distance from the sun approximation is close to the circular plane's radius but not the circular plane's actual radius. The circular plane is just for the overall view of approximated different distances from the sun of each planet on its different size of the radius, with the arrow label can provide a more accurate approximated distance from the sun for each planet with the visual assistance of circular planes.

To visualise this, I created the editor camera inside this 3D visual to provide better interaction for different angles that can be viewed on the circular plane. The circular plane also has the texture of a planet shaded on its surface, allowing comparison of the distance of each plane more easily and detailing the surface texture that made it easier to distinguish each planet. The red arrow model for label of distance from the sun of planets was added to this visual.

The comparison of distance from the sun for each planet gives a more precise result with the amazed look of all planes combined next to each other and the red arrow labelled distance from the sun for each planet. Figure 11 suggests Neptune has the furthest distance from the sun, and Uranus is the next planet with a closer distance to the sun than Neptune by just looking at its circular plane with red arrow labels. Fortunately, the zoom-in feature is also applicable in this 3D visual, and it is handy to find the approximate distance and details of the planet with a smaller size and closer to the sun. Sometimes it can be challenging to spot a planet far apart from another planet as this is based on actual data with unit length downscaling and this designed circular plane with labelled distance makes life easier.

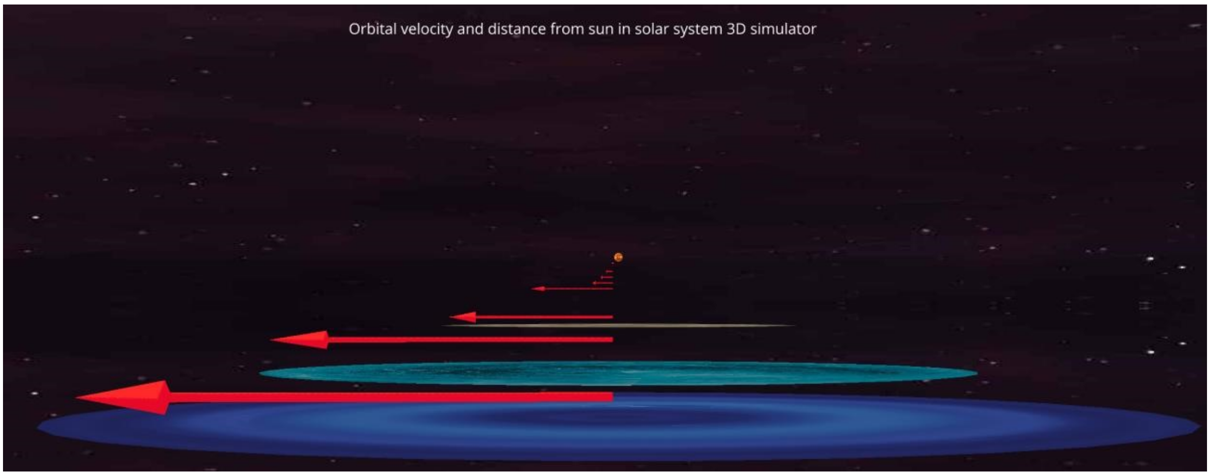


Figure 11 The full 3D visualisation for all eight planets with eight circular planets in solar system and labelled distance from sun for all planets by red arrows

I was able add serval options to make it more interactive for the distance. It provided each distance from sun labelled by red arrow corresponding to each planet and had the functionality to display their distance from sun. The red arrow label and text of real measured distance data will only appear if the distance from sun option was clicked for the corresponding planet as shown in Figure 12. The Venus option button was clicked, the red arrow for the distance appears on the circular plane and the white text shown 108.2 km (e6) for the distance from sun for Venus below the option button. User can click the Venus option button again to cancel the labelling and hide the distance information for the corresponding planet.

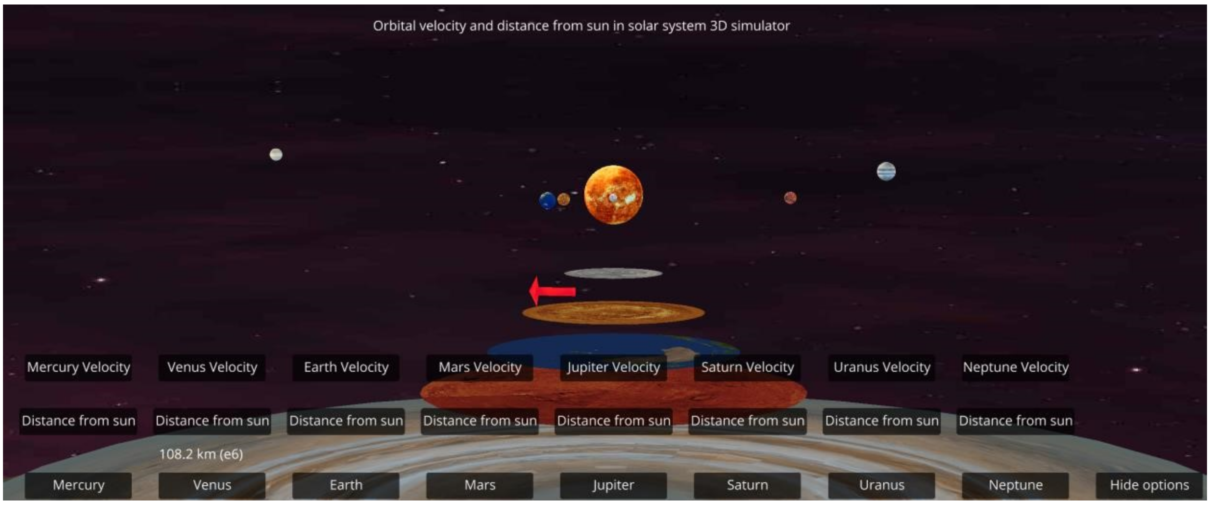


Figure 12 The Venus's distance from sun was clicked and information of distance from sun shows under the option button

After implementing the functionalities for each button, this gave lots of helpful information and labelled for this visual. However, this could have been better as this can be too messy to provide a good visual of the planets and led to undesired effects, such as half of the screen being covered by buttons. To overcome this problem, I have added hide options button to hide all the options and information for a better visual experience.

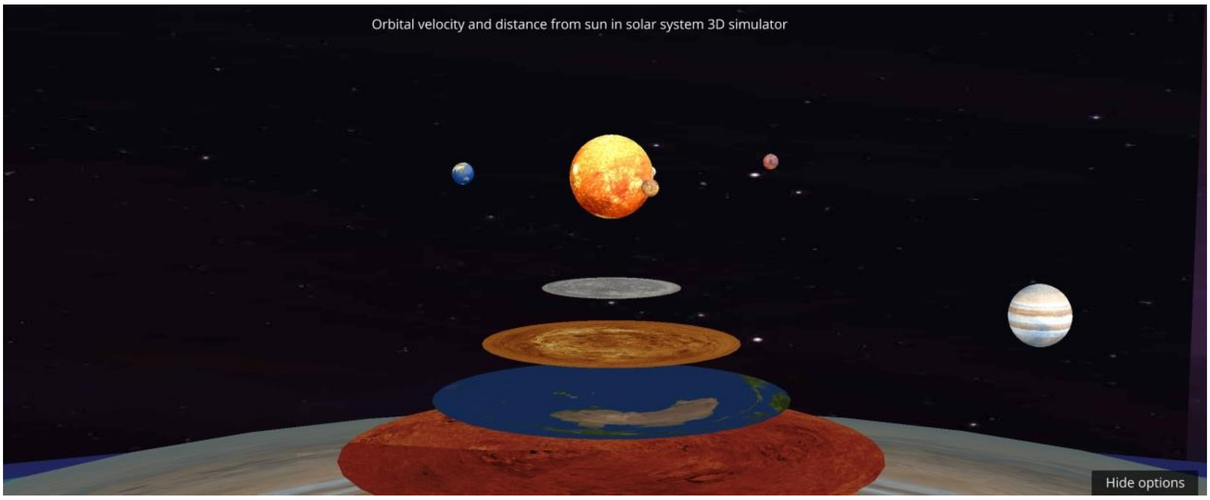


Figure 13 The buttons and texted information were hided when Hide options button was clicked

5.3 3D visual for orbital velocity of planets

To visualise the velocity of planets, I have made an orbital movement animation in this 3D visual with scaled-down speed based on actual data. This 3D visual shows that each planet has a different speed according to its distance from the sun. Mercury has the fastest speed of orbital movement as it’s the closest planet to the sun, and Uranus has the slowest orbital movement as its further away from the sun. However, this is important to note that the velocity unit of each planet is calculated by km per second. It is speedy and challenging to be captured by human eyes. For this reason, I have scaled the planet's orbital movement speed down and shown the different speeds to another planet as accurately as possible. Doing this makes it more comfortable and easier to capture by human eyes.

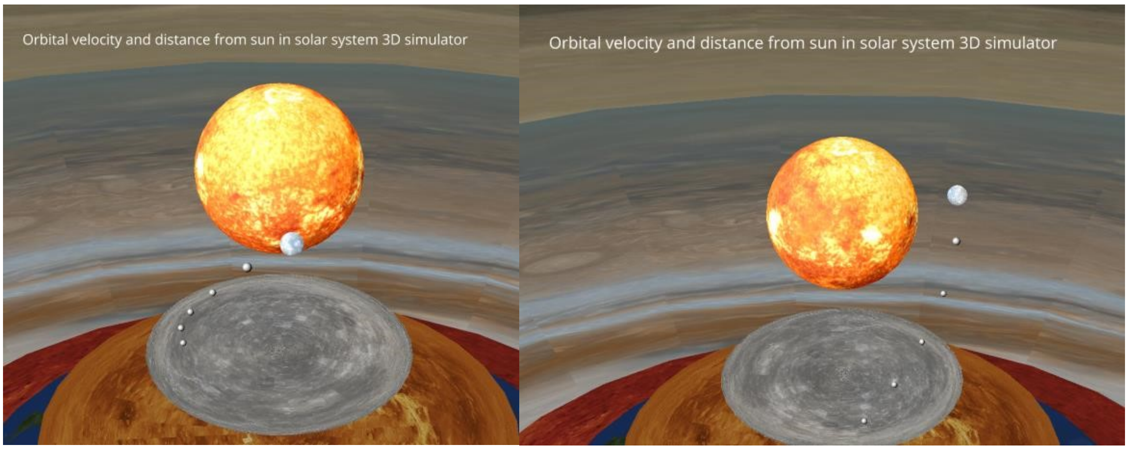


Figure 14 The movement of Mercury around the sun

With the help of Python's `ursina` library, I also added the trail tracking animation for each planet for a better visual of the velocity for each planet. As shown in Figure 15, many small white spheres models at the bottom of planet represented the trail of each planet's orbital movement. The trail is at the bottom of the planet when it pauses and will start following the orbital movement of a planet when it is unpaused. It is clear that I have paused Earth, and the other two planets, except for Mercury, are still orbiting around the sun with approximate downscaling speed from actual data.

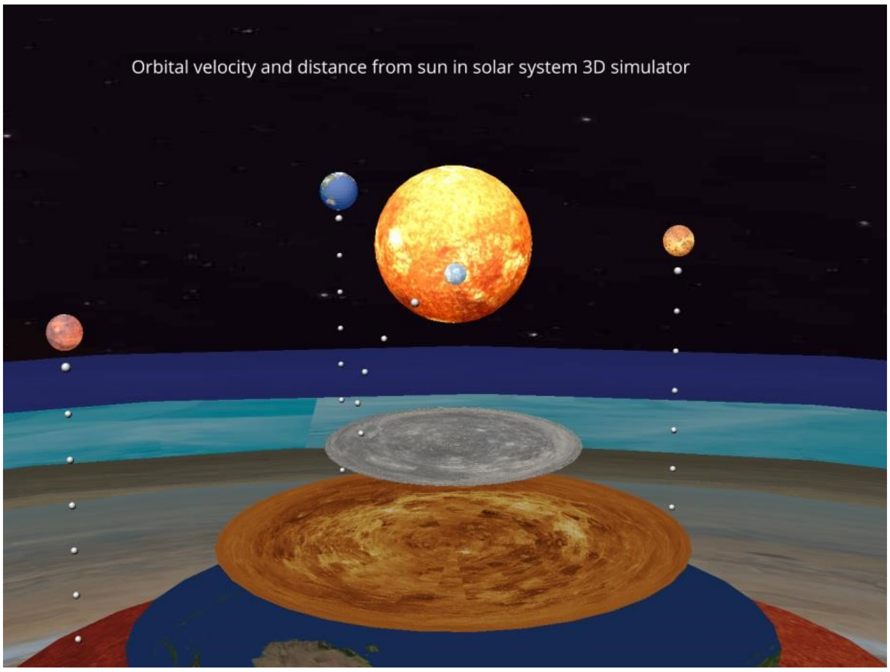


Figure 15 Planet's movement trail in 3D visual

The trail of the planet's orbital movement also attached to the corresponding circular plane and can be disabled by click again on velocity option. This is useful for use to find their corresponding circular plane and viewable on the speed of planets at the same time.

I have also implemented the functionality to show the information of the velocity of corresponding planet if option was first clicked:

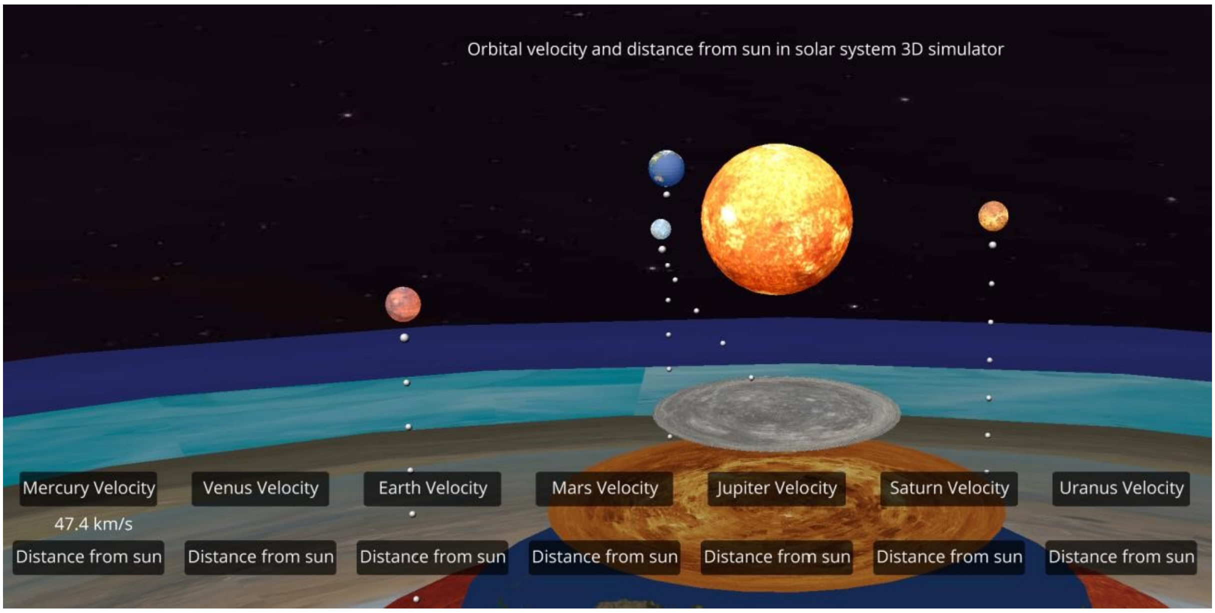


Figure 16 The information of velocity shows under Mercury Velocity option

Figure 16 shows the information on velocity for Mercury as white text under the Mercury Velocity option buttons. This is useful to get the user to understand the planets' actual speed.

6. Conclusion

Overall this project has successfully visualised simple linear regression, multiple linear regression, and 3D visualization with enhanced interactive features for the velocity analysis of each planet in the solar system and the relationship between velocity and the distance of each planet from the sun.

With several powerful Python libraries such as `seaborn` and `matplotlib`, I was able to represent the excellent visual for this project.

6.2 Areas for improvement

The main area for improvement of this project would be the quantity of the dataset of planets.

This project only visualized the orbital velocity of eight main planets and their distance from the sun in the solar system. However, the quantity of the dataset could be improved if the area of the planet's data was not limited to the solar system but to the whole galaxy. This can be done by collecting and creating a dataset of specific galaxies, which prevents the dataset from getting too large or too small. Thus, the current dataset used for velocity and distance analysis could have been improved by increasing the data intensity and providing more flexibility for this project to create a more complex visualization.

More complex visualisation can be used, such as a stacked histogram plot. It may have been helpful to represent it, allowing a more direct vision of the relationship between velocity and distance from the sun. This could achieve quickly using Python's `matplotlib` library.

For 3D visuals, more complex elements such as trend lines, labelled distance measures, and velocity measurements can be added to the visual. Python's `pygame` library could have been used to achieve this.

6.3 Flaws and disappointments

The enormous data unit makes it very difficult to visualise the real-scaled solar system on a smaller screen. I found it was struggling to keep the data unit downscaling and compute results accurately. Especially when calculating the data of the gravitational force of each planet, this is very easy to make a mistake in calculation with this such a large data unit. To mitigate this, I have re-computed the result with Wolfram Alpha's assistance multiple times and ensured that the downscaling results are primarily precise according to the real-life scaling data unit.

Also, this is time-consuming to learn new techniques to visualise the data with several Python libraries. Some implementations of 3D visuals were implemented mainly by my own used math heavily as it required downscaling with enormous data units and computing the planet's orbit with the sun's position. Getting accurate movement of a planet based on real-life data took a lot of work to implement and increased complexity for the current dataset.

Python's ursina library disappointed me as it is hard to use, rarely used by others, and has limited functionality that prevents me from showing a more complex and detailed 3D visual. Only a few tutorials can be found for using this library, and their manual page only gives minimal information about how each function works and their parameters for this library.

7. Reflection

This project was self-learned, and most of my work was my very own implementation. I have spent, on average, more than four hours per day to get it done and success to get it to visualise the data. I enjoyed it when I was about to write such a significant and exciting report and was satisfied that I have made a completion for this project.

This table of content of this project clearly shows all the working processes that I have gone through step by step to complete this project which was a smooth workflow.

The Python program for linear regression was designed with composability and flexibility. It was the most time I have spent on this project. It is my implementation, and I can process any data in json file and implement the visualisation of linear regression. It is reusable for other researchers to only just change the file name that will be retrieved by Python's json package in code and was written mainly by myself. The code can be found in my repository, allowing others to access it when needed. I have increased my time requirements and complexity during this project to achieve this.

Besides, my 3D visual program can give an interesting idea for others to better visualise with limited elements as for the limitation of Python's ursina library. It is a unique 3D visual of the planet's velocity and its relationship of distance from the sun that is designed by my very own, which also can achieve pretty good visuals even with some limitations for this rarely used library.

Overall, it deserves a 7 due to the complexity, time invested, and unique design for 3D visuals.

References

[1] NASA. (2018). *Planetary Fact Sheet*. Nasa.gov. https://nssdc.gsfc.nasa.gov/planetary/factsheet/

[2] Wood, R. (2020, July 27). *Visualizing Multiple Regression in 3D*. Medium. https://towardsdatascience.com/visualizing-multiple-regression-in-3d-a1e7aba7a007

[3] Swaminathan, S. (2018, February 26). *Linear Regression — Detailed View*. Towards Data Science; Towards Data Science. https://towardsdatascience.com/linear-regression-detailed-viewea73175f6e86

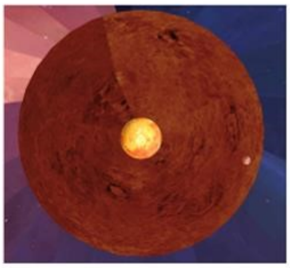
[4] Fitting the Multiple Linear Regression Model. (n.d.). Www.jmp.com. Retrieved April 21, 2023, from https://www.jmp.com/en_au/statistics-knowledge-portal/what-is-multipleregression/fitting-multiple-regression-model.html

8. Appendix

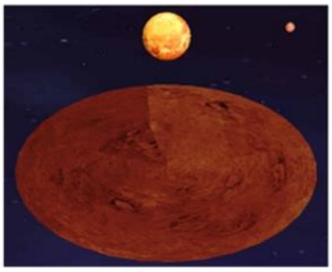
8.1 3D visual of individual planet

Mars

Top view

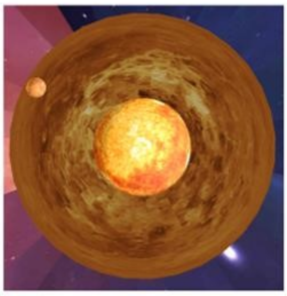


Side view

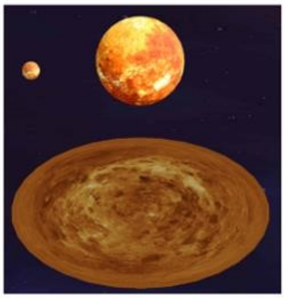


Venus

Top view

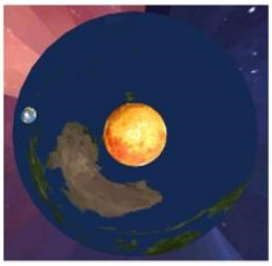


Side view

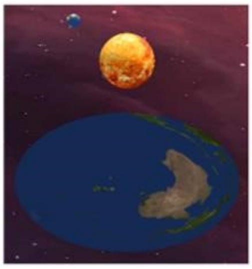


Earth

Top view

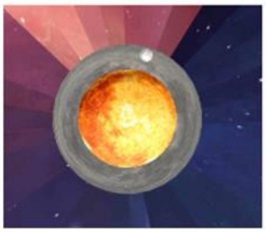


Side view

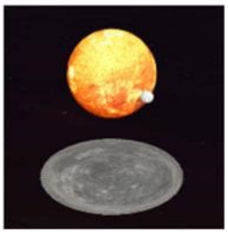


Mercury

Top view

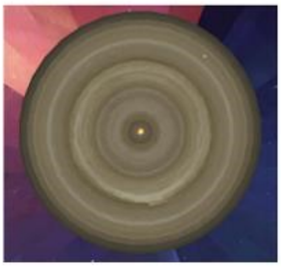


Side view

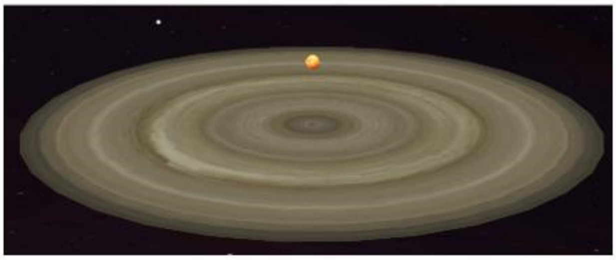


Saturn

Top view

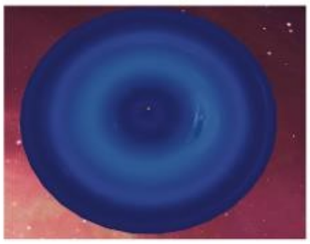


Side view



Neptune

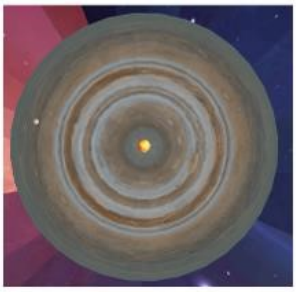
Top view



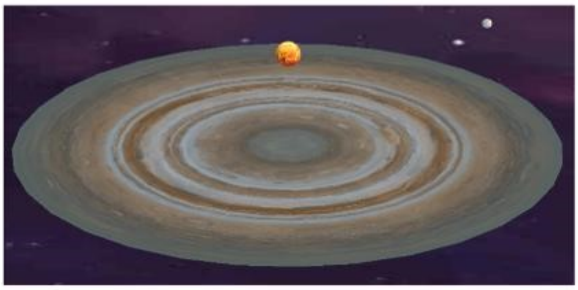
Side view

Jupiter

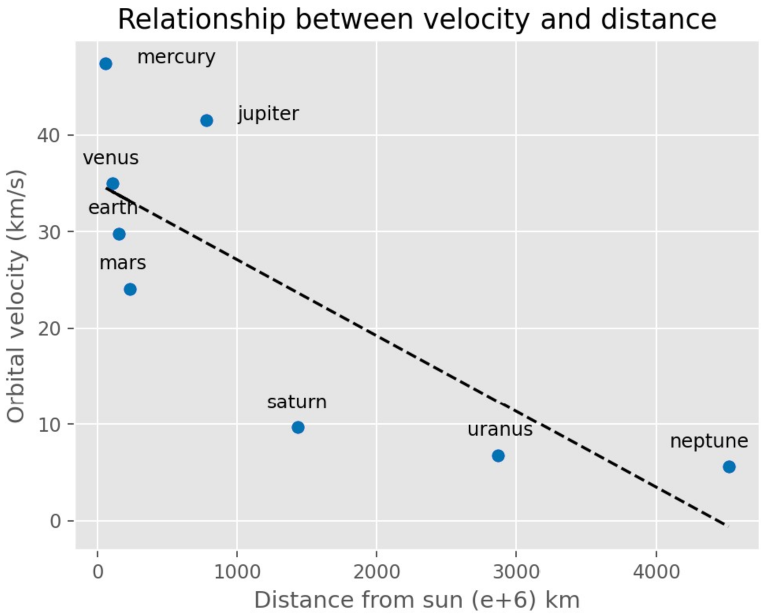
Top view

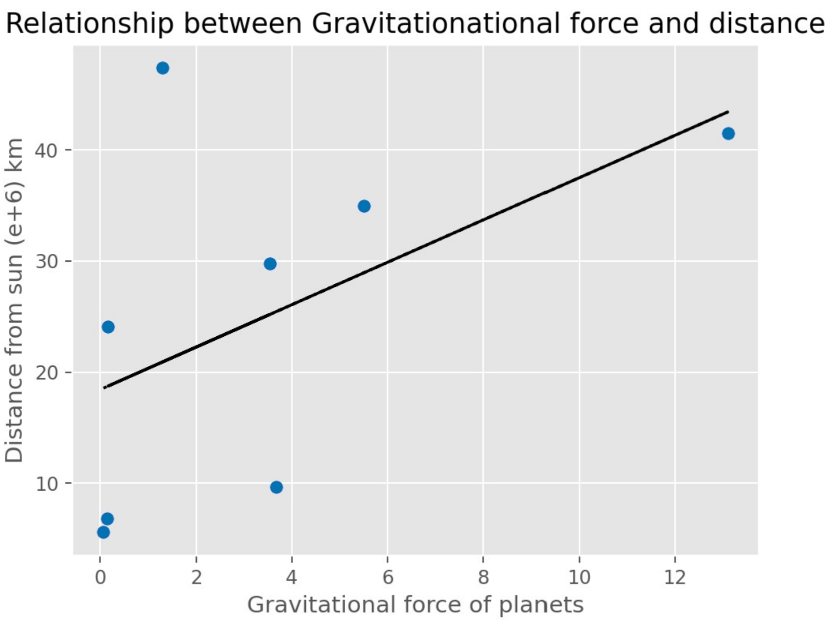


Side view

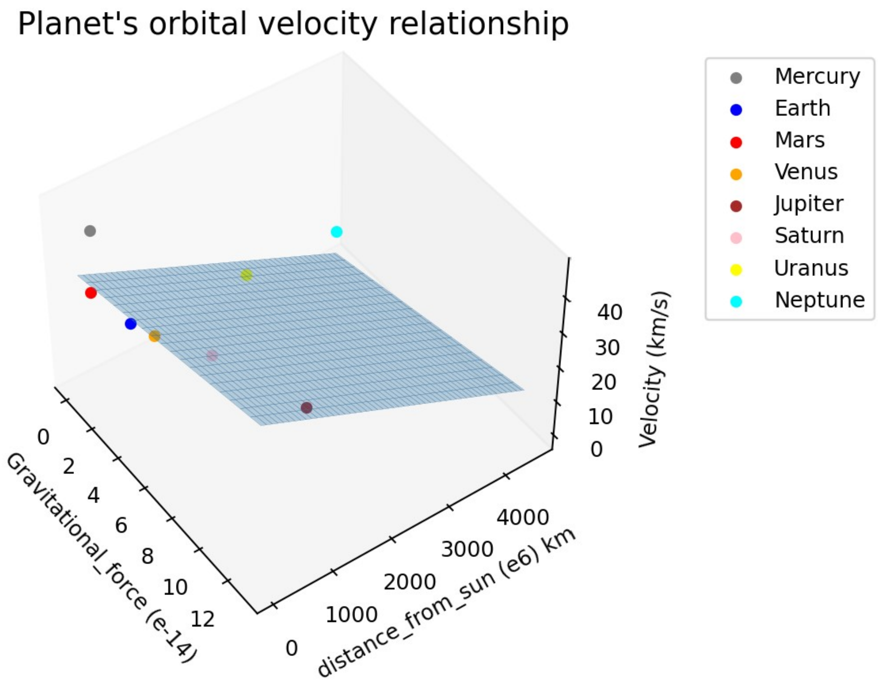


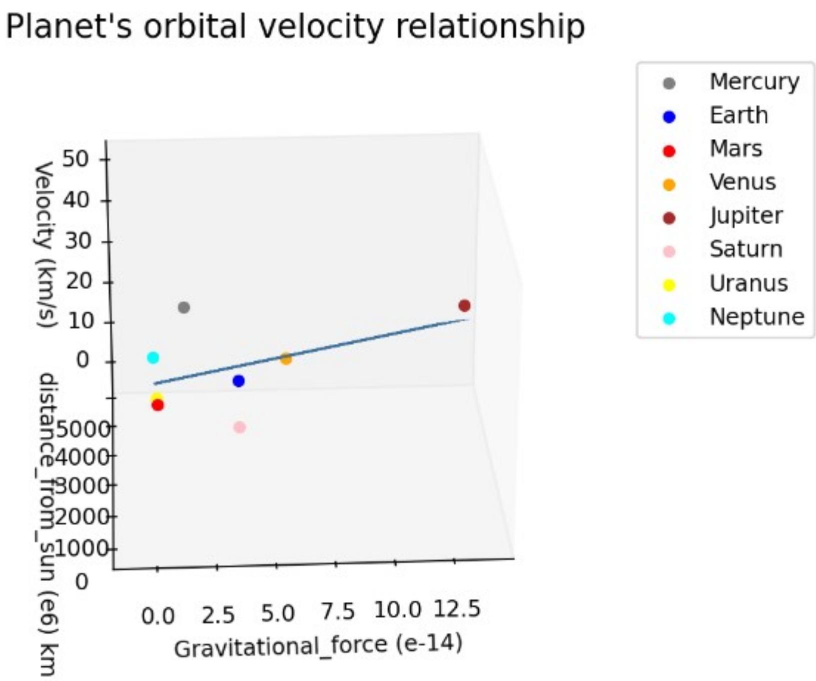
8.2 Simple linear regression



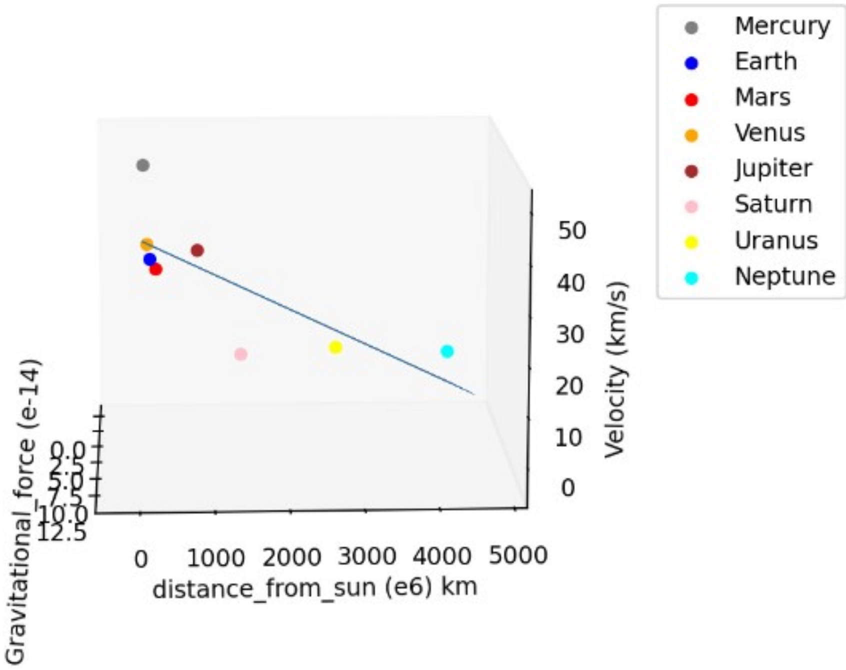


8.3 Multiple linear regression

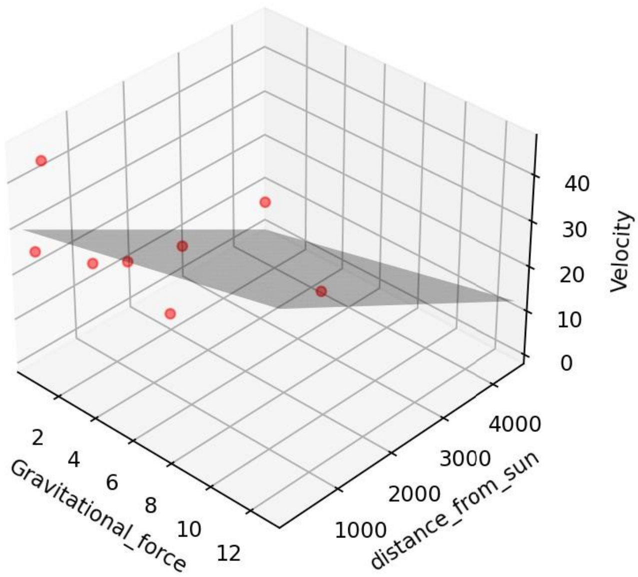




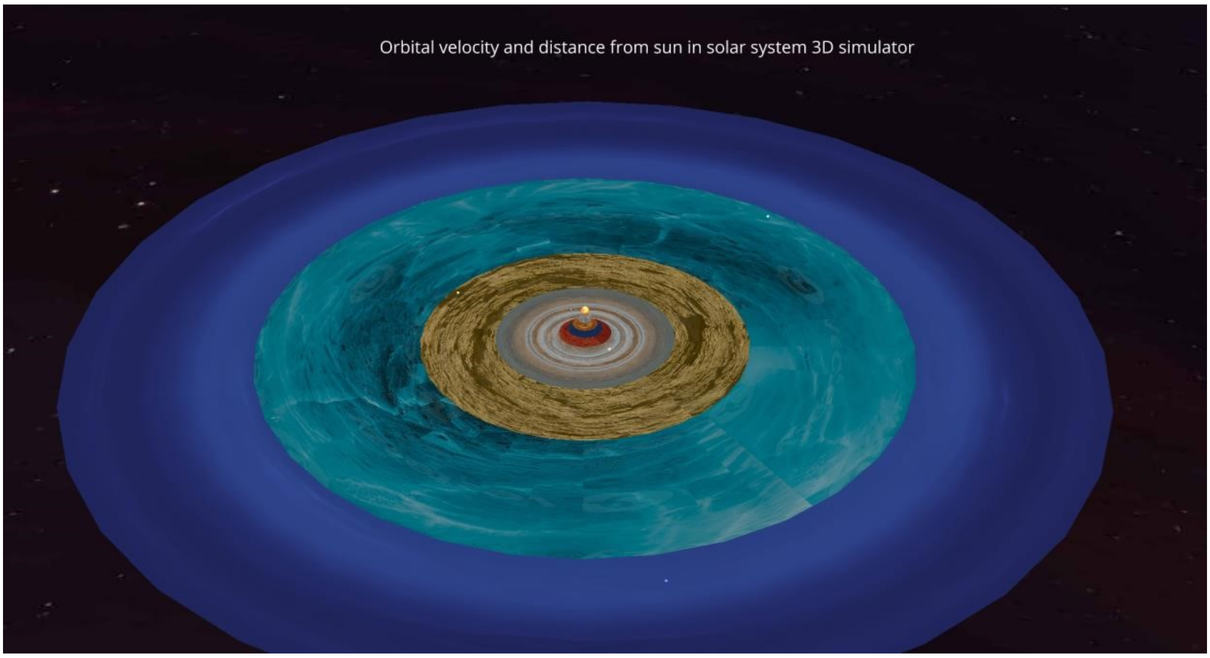
Planet's orbital velocity relationship

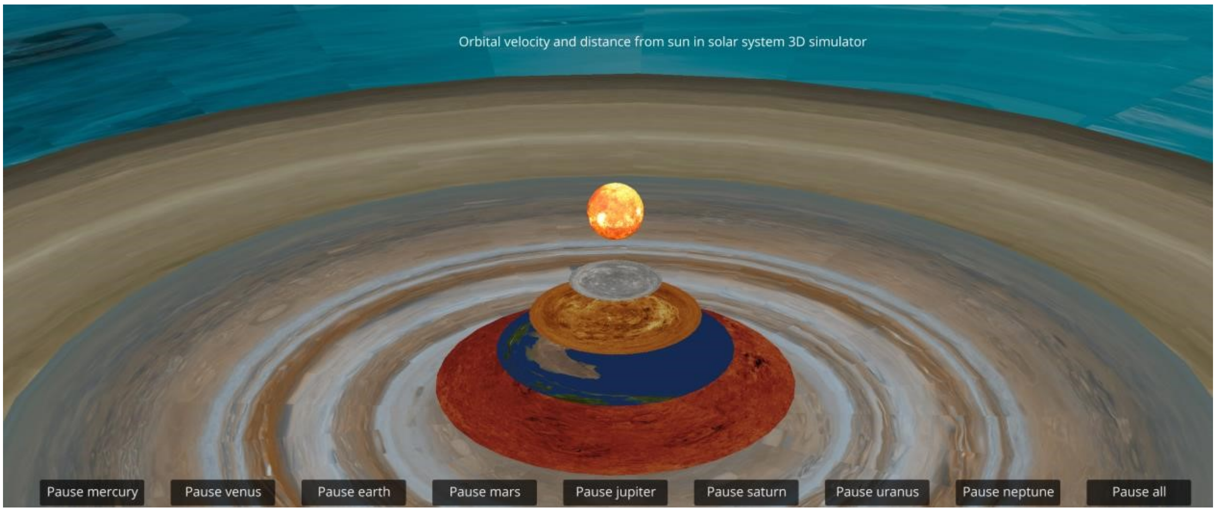


Relationship between planet's orbital velocity, gravitational force and distance from sun

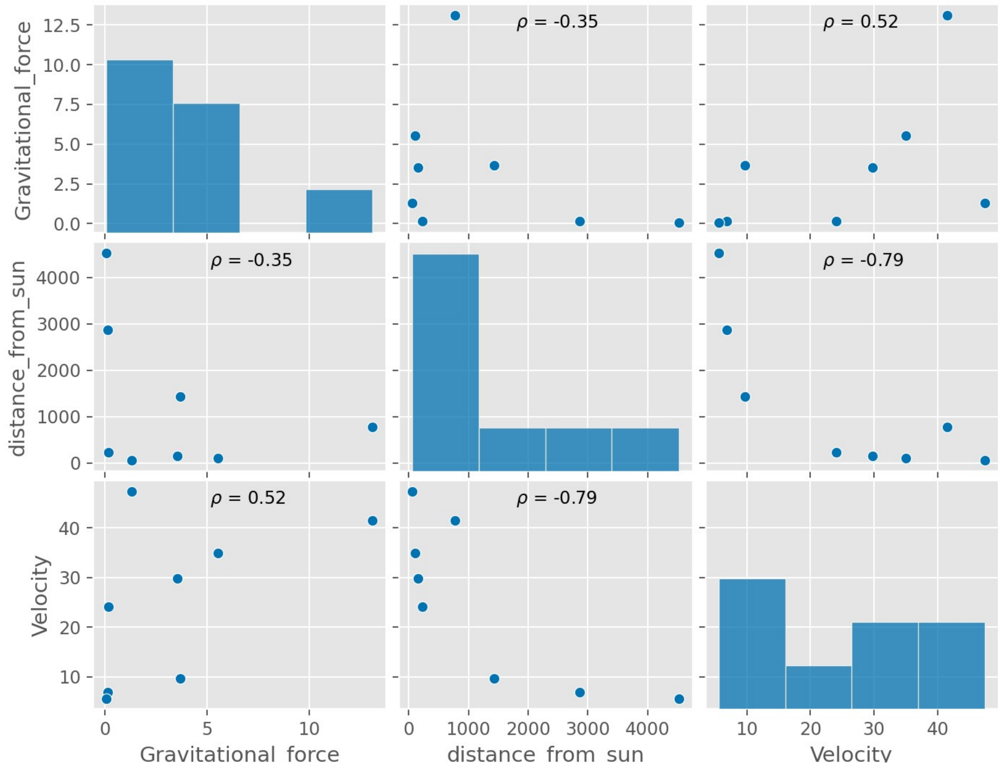


8.4 full 3d visual of all planets





8.5 pair plot



ρ = -0.35
ρ = 0.52
ρ = -0.35
ρ = -0.79
ρ = 0.52
ρ = -0.79
Gravitational_force
distance_from_sun
Velocity
Gravitational_force
distance_from_sun
Velocity