Calculating Hubble Constant using Type-1a Supernovae

What is dark energy?

Dark energy is an unknown force which causes the accelerated expansion of the universe

Its presence is detected by analyzing the change in distances between galaxies over periods of time

It accounts for over 60% of our universe

The rate at which dark energy accelerates the expansions of the universe is referred to as the **Hubble’s Constant**

The Hubbles Constant

Hubble’s Constant is the rate at which the universe is accelerating !

By looking at the change in distance of galaxies within our universe scientist have solved for a value which lies between 60 - 80 km/s/MPC

We can use the hubble’s constant to solve for age of the universe by dividing distance over the constant

$$\begin{tikzcd}
\text{Calculating the Age of the Universe} \arrow[r] & \text{Age = 1/Hc} \arrow[r] & \text{Hubbles Constant ?}
\end{tikzcd}$$

How do we find this?

Type 1a Supernovae

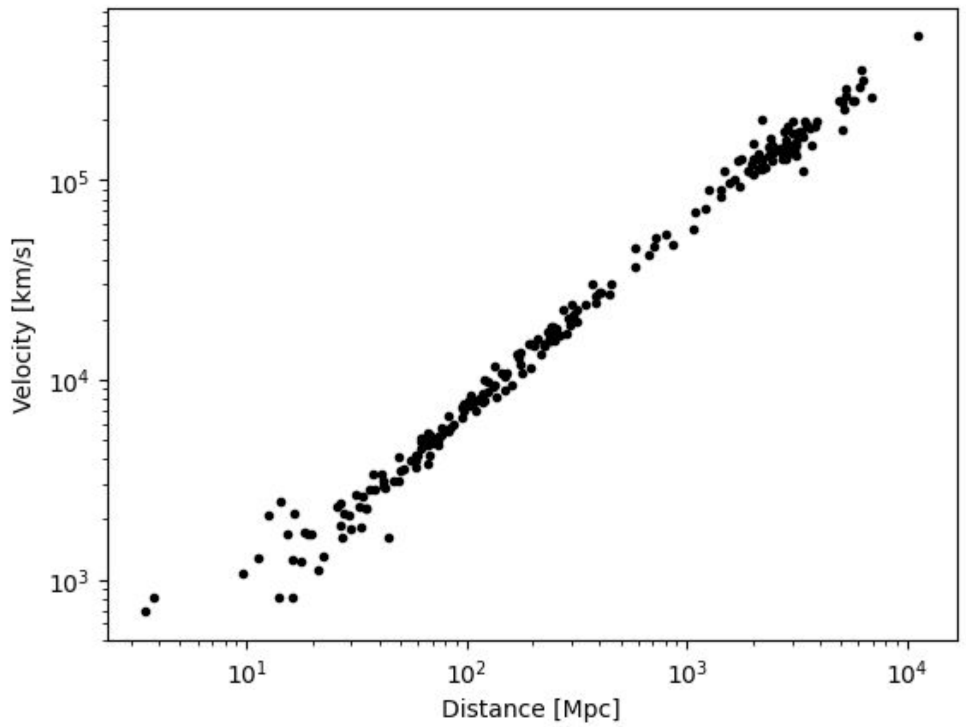
Type 1a Supernovae have a constant luminosity which can light up entire galaxies

With this constant luminosity the brightness of the supernovae can help determine its distance

The distance and velocity of the different supernovae also help us look back in time and calculate the Hubble’s Constant

The article “ Cosmological Results from High z Supernova” provided data of multiple Type 1a Supernovae

We were able to plot the distances of the Supernovae by there velocity to produce this graph



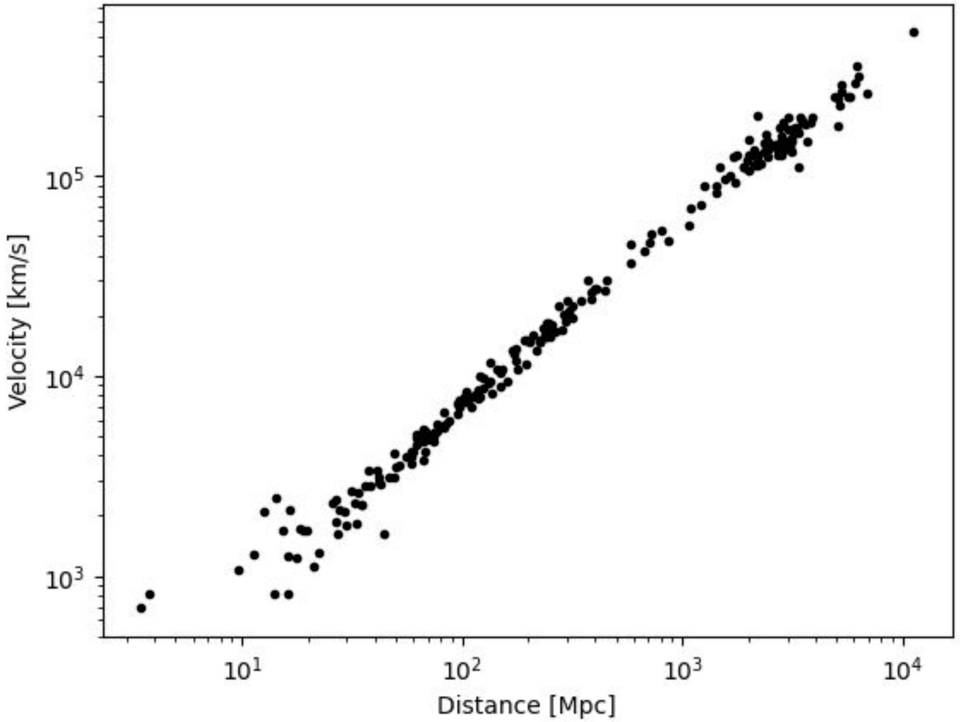
In particular after importing our data into Python, we were able to use several common Python functions to create our plot:

“plt.plot()” which is used to plot the specific arrays of our calculated data

“plt.xlabel/plt.ylabel()” which gives us the labels on the x and y axis of the graph

“plt.legend()” which gives us a legend for our graph

“plt.show()” which displays our graph

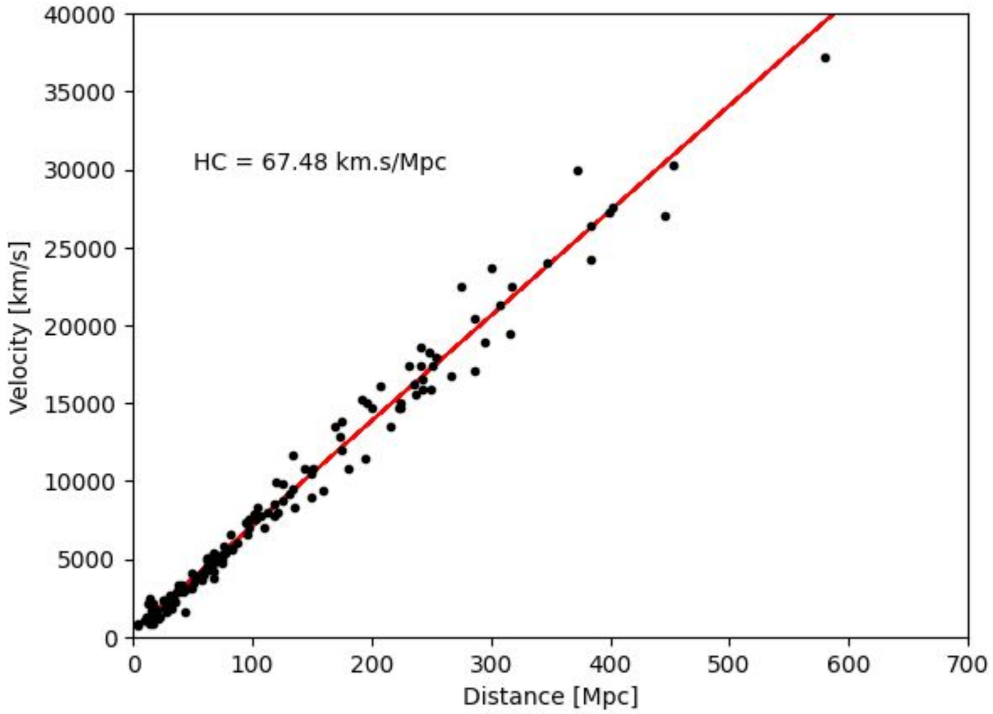


Now Add the Fit Line

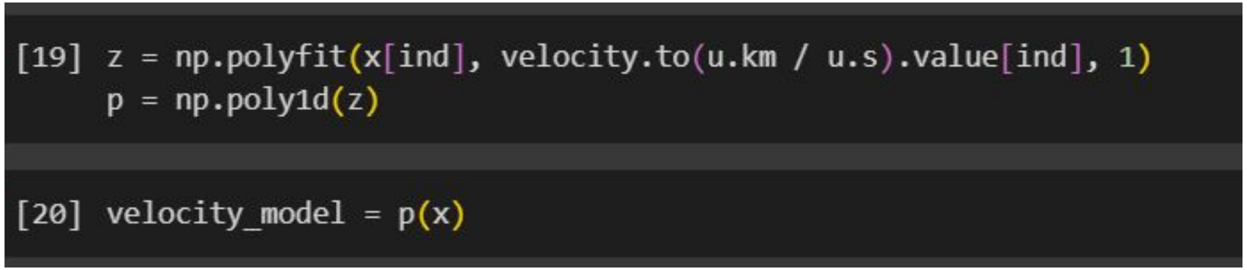
We were able to do so by using the Python command **“np.polyfit()”**

This command works by fitting the data provided within the code to a polynomial function, of your chosen degree.

In this case our data best matched a linear model.



More Specifically



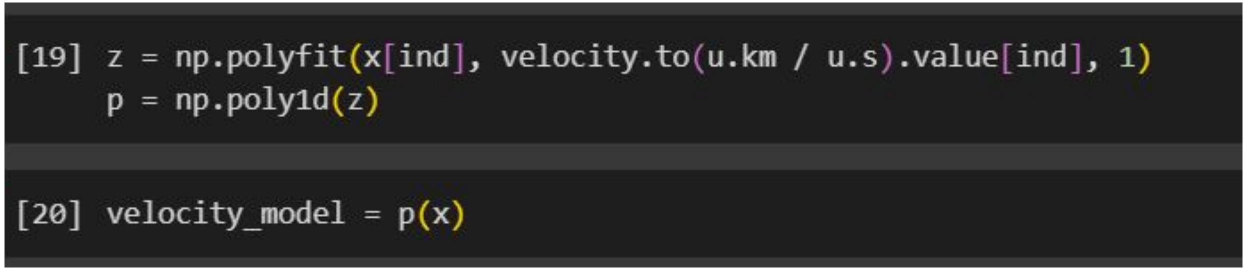
```
[19] z = np.polyfit(x[ind], velocity.to(u.km / u.s).value[ind], 1)
     p = np.poly1d(z)

[20] velocity_model = p(x)
```

In line 19 we use the **np.polyfit()** function to fit our data to our chosen degree polynomial (1), where x represents an array containing distance values converted to megaparsecs, with **x[ind]** giving us a subset of the x array that falls within our specified range.

Still within the polyfit function we see the velocity array originally in the units of km/s, and also **velocity[ind]** which similarly gives us the velocity values that correspond to the selected distance values. Where lastly, the “1” represents the degree of the polynomial, which in this case means a linear fit line.

Continued

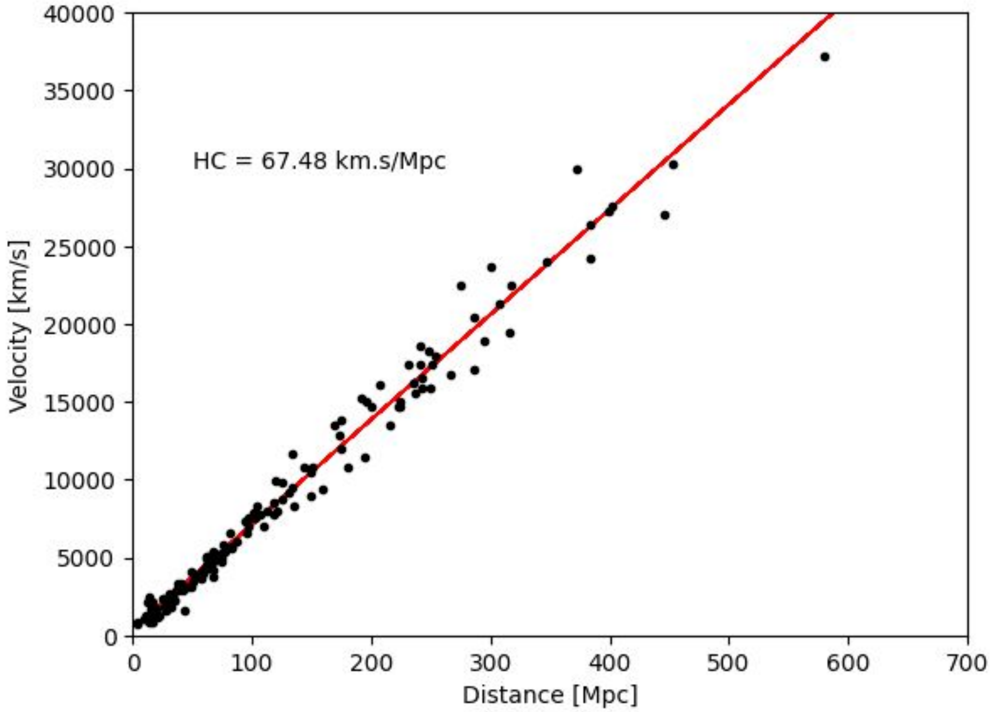


```
[19] z = np.polyfit(x[ind], velocity.to(u.km / u.s).value[ind], 1)
     p = np.poly1d(z)

[20] velocity_model = p(x)
```

Our next function **np.poly1d()** allows us to create a polynomial object, using the coefficients from z. By then setting the output equal to p, you can use a function p(x) to compute any set of x-values (which we see in line 20).

Using the data points we were able to create a linear model (red line) which had a slope of **67.48 Km/s/Mpc** —--› Calculated Hubble's Constant



Conclusions

From calculating the hubble's constant to be 67.48 km.s/Mpc we can input the value into the equation for the Age of the universe

Age = 1/Hc

= 1/67.48 km/s/Mpc

= 14490104055.731909 years

AI Statement

In this project our team utilized google Gemini (an advanced AI model), which is offered as a feature in colab, in hopes of more efficiently debugging our code and further supplementing it.