Calculating Hubble constant using Type-1a Supernovae

Dark energy is an unknown force which causes the accelerated expansion of the universe, whose presence is detected by analyzing the change in distances between galaxies over periods of time. Considering it accounts for over 60% of our universe, dark energy is pretty important.

By utilizing what we know about both the visible universe and dark energy we can calculate an expansion rate of the universe. The rate at which the universe expands is referred to as the Hubble's Constant. By looking at the change in distance of galaxies within our universe scientists have solved for a value which lies between 60 - 80 km/s/MPC, we can then use the Hubble Constant to solve for age of the universe by dividing distance over the constant.

Where the Type 1a Supernovae come in handy in this calculation is in their luminosity. Considering they have a constant luminosity which can light up entire galaxies, the brightness of the supernovae can help determine its distance. The distance and velocity of the different supernovae overall help us look back in time and calculate the Hubble Constant.

Motivation

Considering the Hubble constant provides us with an expansion rate of the universe, by determining it we can better understand the dynamics of our ever changing universe. By calculating the hubble constant we can also determine an estimate for the age of our universe. Calculating the age of the universe yields several important implications itself, such as it helping us better understand the timelines of important cosmic events.

Methods

The article “ Cosmological Results from High z Supernova” provided data of multiple Type 1a Supernovae, which we used to plot the distances of the Supernovae by their velocity to produce a graph (which was a scatter plot).

In particular after importing our data into Python, we were able to use several common Python functions to create our plot. Following the creation of our plot we were able to use basic Python functions to obtain a linear fit line. We were able to do so specifically 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 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. 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).

Results and Conclusions

By using the data points provided in the article we were able to create a linear model which had a slope of 67.48 Km/s/Mpc, which is our calculated Hubble's Constant. So, then from the calculated hubble's constant of 67.48 km.s/Mpc we were then able to input this value into the equation for the Age of the universe. The equation being Age = 1/Hc, which we find gives us 1/67.48 km/s/Mpc, or 14490104055.731909 years.

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.

Link to Google Slides Presentation:
https://docs.google.com/presentation/d/11-R-4JzaxpqJTeANE6DjTpuaY_NT3Rf1F2Mu53V8RBk/edit#slide=id.g2f9f92c8d6e_0_0