Calculating Hubble Constant

Jeremy Timog and Elena Penner

Motivations

- Understanding the origin and eventual fate of the entire universe
- Understanding the age of the universe and seeing how the expansion model we’ve calculated fits the data

Methods

- Utilizing python packages to plot and fit models to data
- Using the equation for the Hubble Constant and data from supernovae with known brightnesses

```python
import numpy as np
import matplotlib.pyplot as plt
import astropy.io.ascii
import astropy.units as u
import astropy.constants as ac

from google.colab import files
uploaded = files.upload()
#importing necessary packages and uploading data file
```

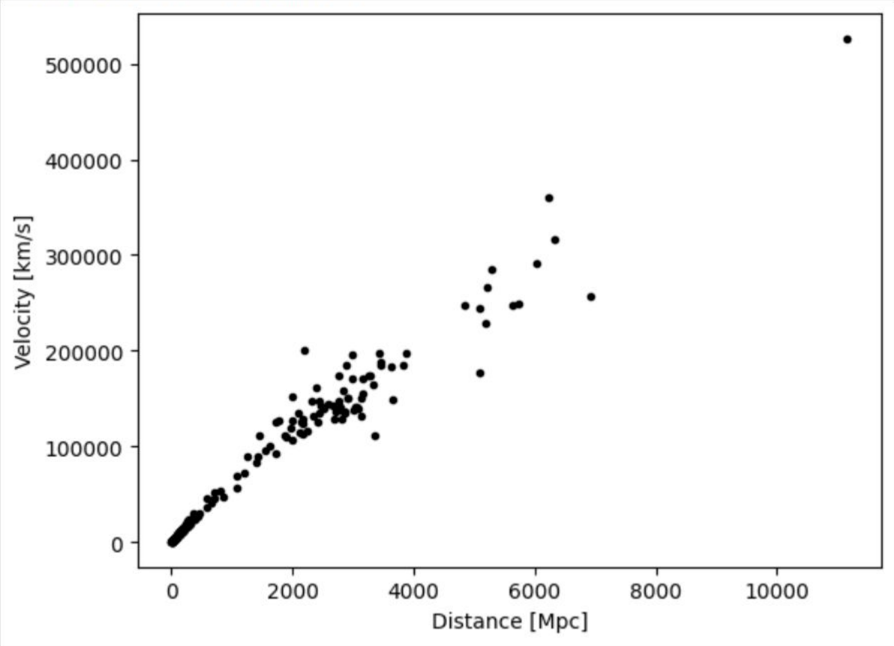
```python
data = astropy.io.ascii.read("Tonry_2003.vot") # formatting imported data into python
print(data) # printing data
```

```
/usr/local/lib/python3.10/dist-packages/astropy/io/ascii/html.py:91: XMLParsedAsHTMLWarn
  soup = BeautifulSoup("\n".join(lines))
  col1     col2    col3    col4     col5    col6  col7  col8  col9   col10    col11
-------- ------- ------- ------ -------- ----- ----- ----- ----- -------- --------
 SN1972E  314.84   30.08 0.0023    N5253    16 2.839 2.399 0.033 204.9697 -31.6692
 SN1980N 240.161 -56.689 0.0056    N1316     9 3.225  3.14 0.043  50.6753 -37.2074
 SN1981B  292.97  64.743 0.0072    N4536     2 3.334 3.077 0.041 188.6233   2.1995
 SN1981D 240.161 -56.689 0.0056    N1316     9 3.225 3.044 0.055  50.6753 -37.2074
 SN1986G 309.543  19.401 0.0027    N5128    26 2.908  2.44 0.035 201.4028 -43.0316
 SN1988U   8.737 -81.227   0.31     Anon    24 4.968 5.096 0.072   3.5756 -30.4164
 SN1989B 241.991  64.403 0.0036    N3627    37 3.033 2.844  0.03 170.0578  12.9718
 SN1990N 294.369  75.987 0.0044    N4639    21  3.12 3.204 0.035 190.7367  13.2566
 SN1990O  37.654   28.36 0.0307 M+034403    10 3.964 3.977 0.025 258.8999  16.3241
 SN1990T 341.503 -31.526   0.04   P63925    10 4.079 4.101 0.042 299.7601 -56.2583
     ...     ...     ...    ...      ...   ...   ...   ...   ...      ...      ...
SN2000cx 136.506 -52.482  0.007    N0524 11:19 3.322  3.32 0.023  21.1926    9.509
SN2000dk 126.834 -30.344 0.0164    N0382    11 3.692 3.677 0.023  16.8483  32.4068
SN2000dz  84.367  -56.39    0.5     Anon    13 5.176 5.352 0.053 352.6728   0.3119
SN2000ea 167.211 -61.402   0.42     Anon    13   5.1 5.091 0.053  32.4751  -5.4719
SN2000ec 166.295 -60.176   0.47     Anon    13 5.149  5.34 0.043  32.8836  -4.2321
SN2000ee 166.045 -53.429   0.47     Anon    13 5.149 5.343 0.041  36.8945   1.1971
SN2000eg 167.097 -53.108   0.54     Anon    13 5.209 5.236 0.049  37.5883   1.0639
SN2000eh 188.297 -31.651   0.49     Anon    13 5.167 5.224 0.042  63.7604   4.3888
SN2000fa 194.167  15.479 0.0218   U03770    11 3.815 3.837 0.026 108.8752  23.4287
 SN2001V 218.929  77.733 0.0162    N3987    22 3.686 3.662 0.023 179.3542  25.2024
Length = 230 rows
```

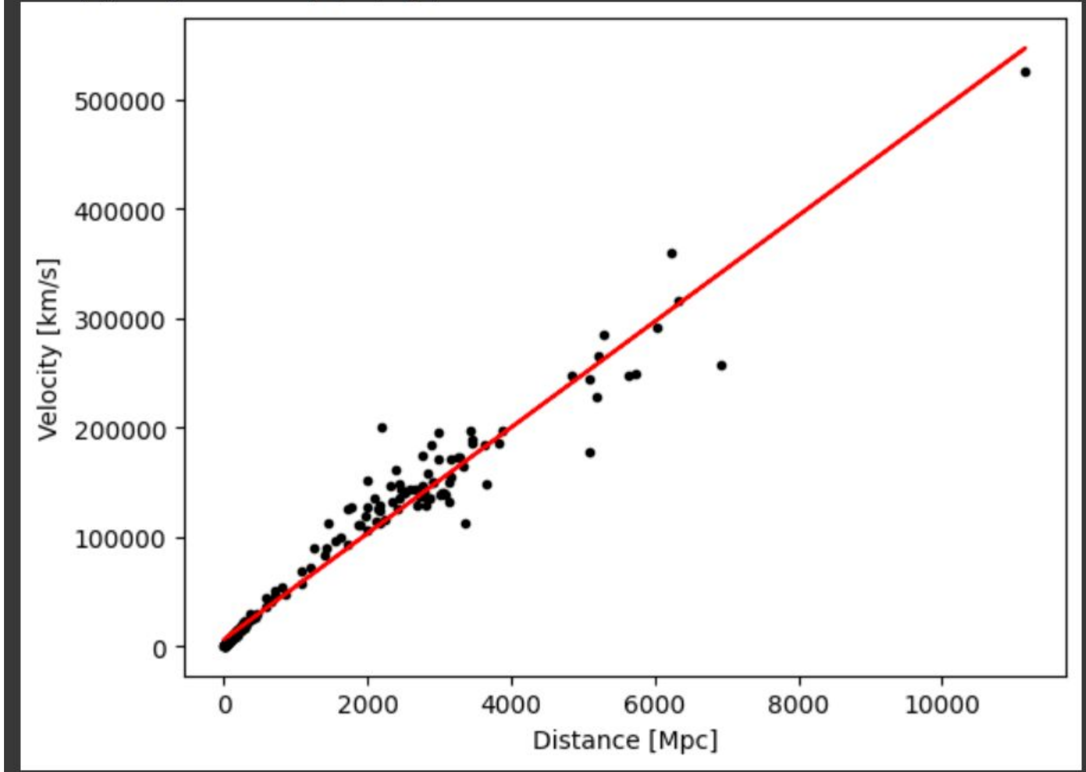
```
dist = ((10**data["col8"] * u.km / u.s) / (72 * ((u.km * u.s**-1)/ u.mpc)))
"""
"col8" has the values of Final median distance times H_0_ in log(km/s).
Raising 10 to the power of the value in column 8 gives the receding speed of the supernova in km/s.
Then dividing that by the assumed hubble constant gives the distance of the supernova in mpc
"""
vel = 10**data["col7"] * u.km / u.s # "col7" has the values of the receeding velocity in units of log(km/s). Raise 10 to the value for the velocity
```

```
plt.plot(dist, vel, marker=".", color="black", linestyle="none") # plotting and formatting graph
plt.xlabel("Distance [Mpc]")
plt.ylabel("Velocity [km/s]")
```

```
Text(0, 0.5, 'Velocity [km/s]')
```



```python
lin_fit = np.polyfit(dist.to(u.mpc).value, vel.to(u.km / u.s).value, 1) # calculating linear fit coefficients
vel_lin = lin_fit[0] * dist.to(u.mpc).value + lin_fit[1] # creating calculated velocity curve
plt.plot(dist, vel, marker=".", color="black", linestyle="none")
plt.plot(dist, vel_lin, color="red")
plt.xlabel("Distance [Mpc]")
plt.ylabel("Velocity [km/s]")
```



```
h = lin_fit[0] * u.km / u.s / u.Mpc # slope of graph = hubble constant
print(h)
```

```
age = 1/h # age of universe = inverse of hubble constant
print(age.to(u.Gyr))
```

```
20.174816604618965 Gyr
```

Conclusions

- Our model used a linear fit and included ALL data points
- This led to multiple inaccuracies in our model
  - The linear fit did not cross the origin
  - The large amounts of measurement error on farther away data points