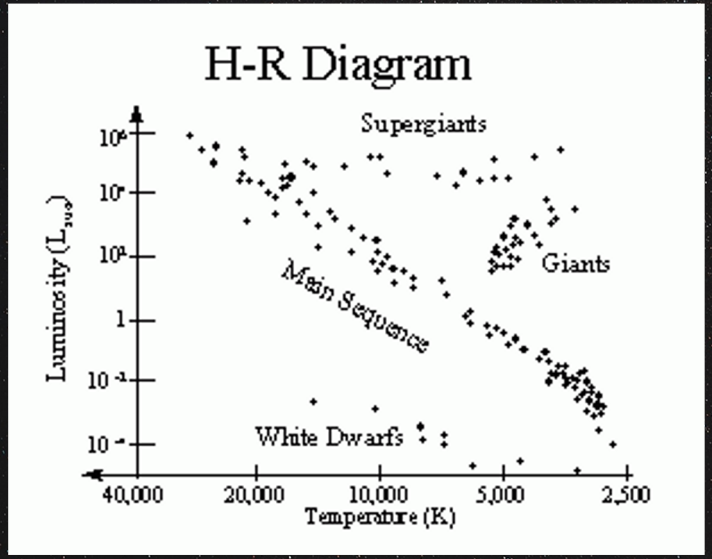
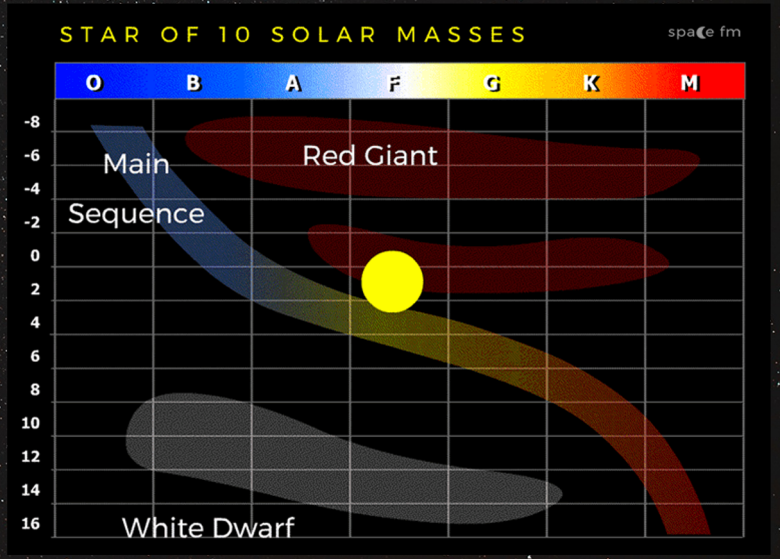
ANIMATED HR DIAGRAM

Victor Cruz Ramos and Nadia Laswi

3 December 2020
Astro 98 DeCal

HERTZSPRUNG-RUSSELL DIAGRAM





GAIA DATA

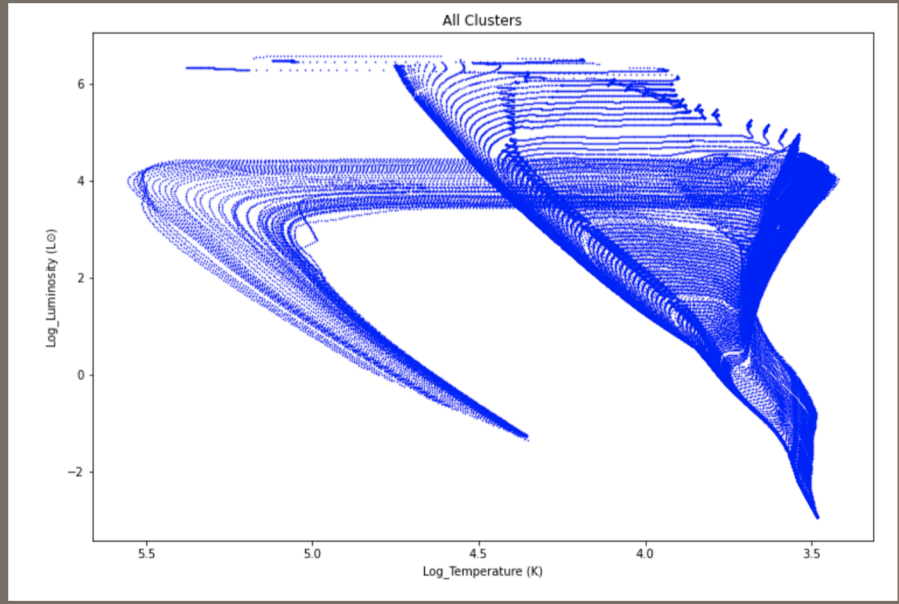
Data we were looking for:

- ☆ Age of star
- ☆ Temperature
- ☆ Luminosity

EEP	log10_isochrone_age_yr	initial_mass	star_mass	log_Teff	
int64	float64	float64	float64	float64	
14	5.0	0.1	0.09999998746585048	3.4862207608164812	3.131
15	5.0	0.10264483521052409	0.10264482205216272	3.487361840014425	3.12
16	5.0	0.10703921552827966	0.10703920129064491	3.4892434780670603	3.119
17	5.0	0.11141918414580901	0.1114191687755168	3.491101990955403	3.112
18	5.0	0.11578922006201725	0.11578920350356509	3.492937182847681	3.105
19	5.0	0.12015251878285713	0.12015250097890492	3.494751515584233	3.098
20	5.0	0.12450712023719923	0.1245071011296722	3.496542324145716	3.091
21	5.0	0.12885093010385934	0.12885090963400228	3.498308436059524	3.08
22	5.0	0.13318409778677381	0.13318407589474415	3.5000490469843273	3.07
23	5.0	0.13750667894275914	0.1375066555678251	3.5017638232875714	3.072
...	...	...	...	...	

PLOT! *(sort of)*

To see what we were working with, we plotted the Luminosity vs Temperature of all the ages.



TIME

To make an HR Diagram animated over time, we need to separate the plots by their age.

☆ Extract the star ages from the table:

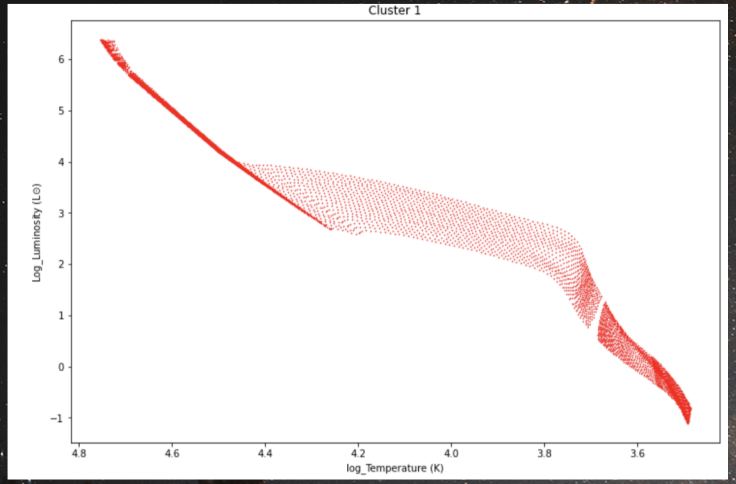
```
all_ages = np.array(np.unique(data['log10_isochrone_age_yr']))
print(all_ages)
```

```
[ 5.    5.05  5.1   5.15  5.2   5.25  5.3   5.35  5.4   5.45  5.5
  5.6   5.65  5.7   5.75  5.8   5.85  5.9   5.95  6.    6.05  6.1
  6.2   6.25  6.3   6.35  6.4   6.45  6.5   6.55  6.6   6.65  6.7
  6.8   6.85  6.9   6.95  7.    7.05  7.1   7.15  7.2   7.25  7.3
  7.4   7.45  7.5   7.55  7.6   7.65  7.7   7.75  7.8   7.85  7.9
  8.    8.05  8.1   8.15  8.2   8.25  8.3   8.35  8.4   8.45  8.5
  8.6   8.65  8.7   8.75  8.8   8.85  8.9   8.95  9.    9.05  9.1
  9.2   9.25  9.3   9.35  9.4   9.45  9.5   9.55  9.6   9.65  9.7
  9.8   9.85  9.9   9.95  10.   10.05 10.1  10.15 10.2  10.25 10.3 ]
```

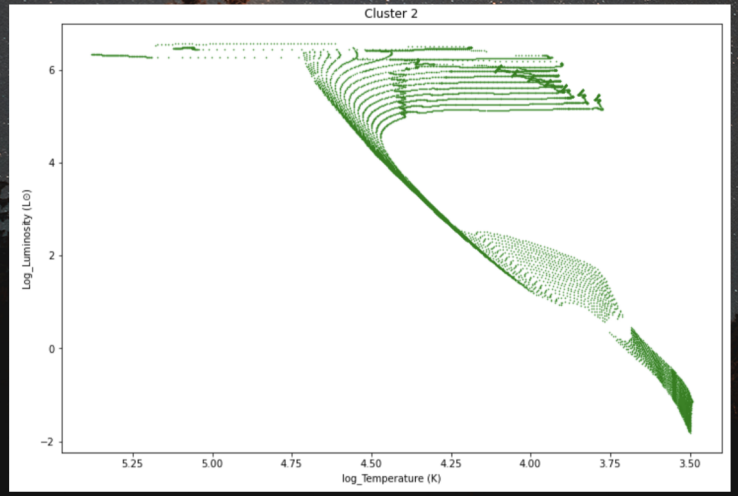
☆ Define a function that isolates data according to age:

```
def isochrone(age_1, age_2):

    values_1_temp = []
    values_1_lum = []

    for i in range(len(data['log10_isochrone_age_yr'])):
        if data['log10_isochrone_age_yr'][i] >= age_1 and data['log10_isoch
            values_1_temp.append(data['log_Teff'][i])
            values_1_lum.append(data['log_L'][i])
    return values_1_temp, values_1_lum

T1, L1 = isochrone(5.0, 6.0)
```

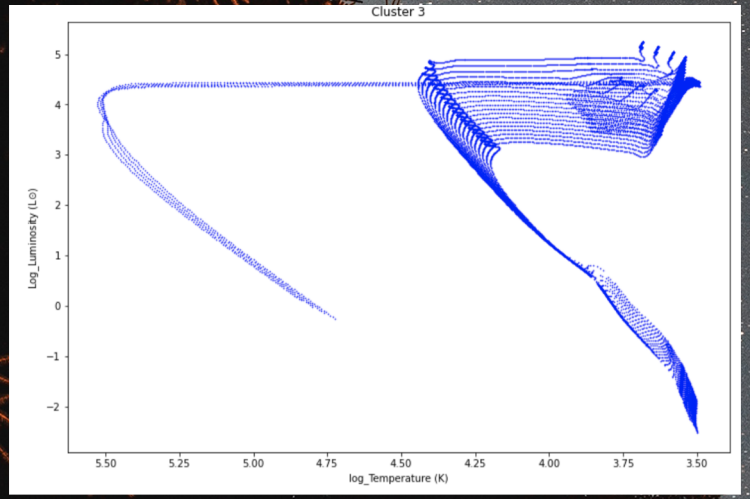
PLOT

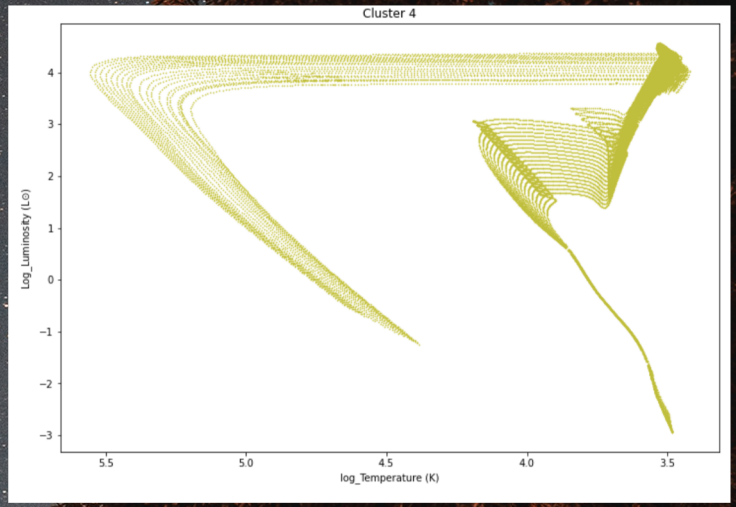


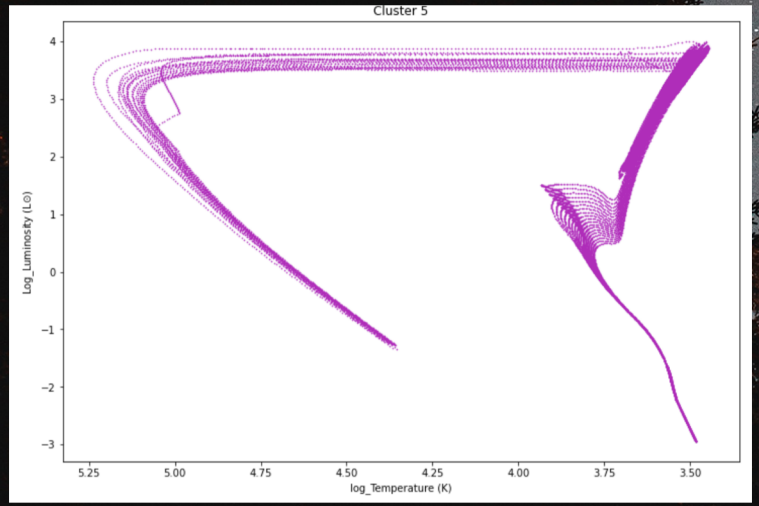
Note: the earliest age adheres most to the main sequence

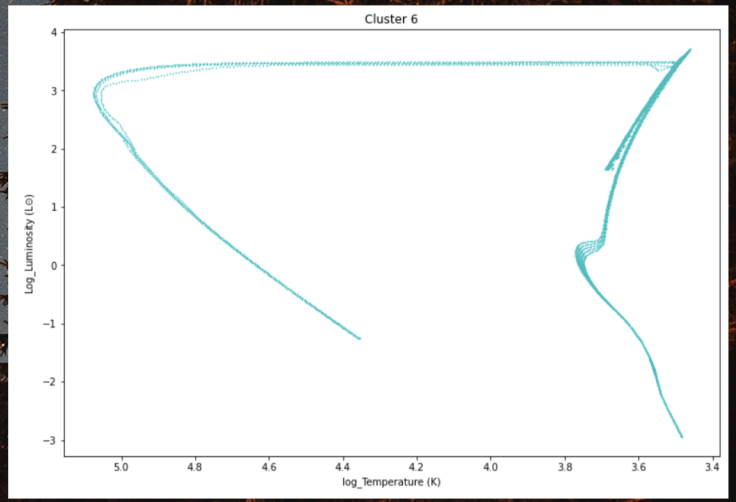


With the new function, we were able to plot 6 different graphs, according to age range.


Cluster 3
Log_Luminosity (L⊙)
log_Temperature (K)


Cluster 4
Log_Luminosity (L⊙)
log_Temperature (K)


Cluster 5
Log_Luminosity (L⊙)
log_Temperature (K)


Cluster 6
Log_Luminosity (L⊙)
log_Temperature (K)

ANIMATION

```
fig = plt.figure(figsize = (12,8))
ax = plt.axes(xlim=(3.0, 6.0), ylim=(-3, 6))
ax.set_xlabel('Log_Temperature (K)')
ax.set_ylabel('Log_Luminosity (L${\odot}$)')
plt.gca().invert_xaxis()
line, = ax.plot([], [],'m.', markersize=3)

def init():
    line.set_data([], [])
    return line,

def animate(i):
    age = 5+.05*i

    index = np.where(np.round(data['log10_isochrone_age_yr'], decimals = 2) == np.round(age, decimals=2))

    x = data['log_Teff'][index]
    y = data['log_L'][index]

    line.set_data(x, y)
    ax.set_title('HR Diagram over time (logarithmic)')
    return line,

anim = animation.FuncAnimation(fig, animate, init_func=init, frames=20, interval=20, blit=True)
print(type(anim))
#anim.save('basic_animation.mp4', fps=20, extra_args=['-vcodec', 'libx264'])
Writer = animation.writers['ffmpeg']
writer= Writer(fps=20, metadata=dict(artist='Me'), bitrate=1800)
anim.save('basic_animation.mp4', writer=writer)
plt.show()
```

Finally, we used matplotlib.animation to animate the plot.

THANK YOU

Shoutout to Yilun, James and Wendy for helping us all with our projects

CREDITS

Data for our animation is from **MIST.Harvard.edu**

We learned matplotlib.animation from **matplotlib.org**

FINAL PRODUCT