exam1Corrections

October 13, 2023

[ ]:
```
#SETUP CELL (modify at your own peril)
from astropy import units #access units by doing units.<unit> (i.e. units.au)
from astropy import constants
import numpy as np #common math functions (i.e. np.sin(x)) and better arrays (i.
 ↪e. np.array([1,2,3])
import matplotlib.pyplot as plt #plotting functions (i.e. plt.plot(x,y))
G = constants.G # gravitational constant
M_sun = constants.M_sun # mass of the sun
R_sun = constants.R_sun # radius of the sun
L_sun = constants.L_sun # luminosity of the sun
M_earth = constants.M_earth # mass of the earth
R_earth = constants.R_earth # radius of the earth
M_jup = constants.M_jup # mass of jupiter
R_jup = constants.R_jup # radius of jupiter
sigma_sb = constants.sigma_sb # Stefan-Boltzmann constant
c = constants.c # speed of light
h = constants.h # Planck constant
k_B = constants.k_B # Boltzmann constant
m_e = constants.m_e # mass of electron
m_p = constants.m_p # mass of proton
m_n = constants.m_n # mass of neutron (basically just the mass of a proton but␣
 ↪whatever)
g0 = constants.g0 # standard gravity, 9.8 m/s^2
e = constants.e # absolute value of electron/proton charge
```

0.1 Exam instructions:

For the multiple choice and short answer questions you have two options: 1. Answer these on the actual test (i.e. circle the multiple choice answer you select and write out by hand the answers to short answer questions) 2. Clearly label and type them into markdown cells (i.e. for MC type 1: a if you picked a for q1)

For the quantitative questions, if you opt to use the notebook you have two options: 1. Just use the notebook to do calculations, writing your final answers down by hand on the physical test. If you do this make sure you note this on the test so I know to check your work here! 2. Clearly label and fully answer the questions here, with a combination of markdown and code cells as you see fit.

After you have finished the test, create a PDF from your work (just like you have done with your homework) and upload it to the corresponding Midterm 1 Canvas assignment.

2. Multiple Choice

   #1. Original Answer: d, New Answer: b

   - The two forces that balance each other in stars that are in hydrostatic equilibrium are not the strong force and gravity, which was option d. The strong force is about as influential as the weak and electromagnetic forces that particles experience compared to gravity when talking about objects on the scales of stars. Gravity increases and becomes much stronger at these astronomical scales, while the strong force does not. The other balancing force in a star, is the pressure gradient which opposes the contracting force of gravity as it pushes outwards on the star, due to being very hot from the Star's core.

   #9. Original Answer: d, New Answer: c

   - I originally said that Alpha Centauri A was the star furthest away from Earth, but it is actually the closest of the stars listed (I think I read the question wrong). The furthest star from Earth is actually Rigel, which we can tell by observing its apparent and absolute magnitudes. The distance equation from these two values is $d = 10^{\frac{(m-M+5)}{5}}$, where m is apparent magnitude and M is absolute magnitude. Without even doing the calculation, we can tell by comparing the differences of $m - M$ between all the stars listed that this value is greatest for Rigel, and thus Rigel must be the furthest away from Earth.

   #19. Original Answer: d, New Answer: b

   - I originally said that the reason nuclear fusion requires such high temperatures and pressures is because "temperature is a reflection of the kinetic energy of particles." This is not the best explanation as to *why* high temperatures are necessary for fusion, even though it's not necessarily wrong. The correct answer is b, "to overcome their mutual electric repulsion," because this more accurately explains what the high temperatures and pressures are being used for. Protons must overcome their mutual electric repulsion in order to cause a fusion reaction, and this process requires these conditions.

3. Short Answer Questions

   1. Original Answer (-1: "How does it do the things you described?"): The "solar thermostat" is a concept meant to describe how the Sun is able to sustain self-regulation through a tight, negative feedback loop for fusion that prevents a runaway reaction from occuring. Additionally, the Sun maintains Hydrostatic Equilibrium, which prevents its gravitational collapse, as well as an Energy Balance between fusion and surface radiation. The Sun is getting gradually brighter because within its core, each nuclear fusion reaction releases Energy, causing the core to shrink and its temperature to tise, which increases the rate of fusion and thus causes an increase in brightness over time.

      - New Answer: The Sun's solar thermostat is able to maintain hydrostatic equilibrium because the force of gravity, which is pulling inward on the star, is balanced by the the pressure coming from nuclear fusion in its core. These is how the solar thermostat actually works and prevents a star from continuously expanding from the force of fusion or continuously contracting due to gravity.

   2. Original Answer (-2: only 1 reason given): There are more stars with masses less than / close to the mass of our Sun because there fundamentally must be more low-mass stars

in the Universe than high-mass stars. This is because it takes less energy to form more massive stars, and because the cloud from which a star forms must have more mass.

- New Answer: The reason that there are more low mass stars is because molecular clouds is because A), molecular clouds require higher densities to create more massive stars, and this is less common, and thus the formation of less massive stars is favored (this is shown via the initial mass function). B), low mass stars live longer than high mass stars, since high mass stars burn through their fuel at a faster rate, and this leads to their being more low mass stars present in the universe over time.

3. Original Answer (-4): The lifespan of the first generation of stars was …

   - New Answer: The lifespan of the first generation of stars was likely shorter than modern stars, with lifetimes in the millions of years. This is due to the fact that population III stars would have had to be extremely massive, which shortens a star's lifespan because they will run through their fuel much faster.

4. b. (WD reason good, other bad. -3) Original Answer: One difference between the diagrams is that the Gaia diagram does not display white dwarfs well. This may be because they are simply harder to observe as they are dimmer. Another difference is that there is a slight difference in the actual shape of the data, which may be from the fact that the Gaia diagram is only a sample of data, which may cause the skewing from the HR model.

   - New Answer: My second reason is incorrect since it is stated in the problem description that the data is not all from a single cluster, so we shouldn't consider inaccurate data as a significant reason for the graph's differences. A valid difference between these graphs is that the Gaia data is more dense for stars of bluer colors (lower wavelengths) than the standard HR diagram shows. This could mean that the Gaia data is observing a relatively younger population of stars, whereas the HR diagram illustrates a more uniform distribution.

4. Quantitative

1. 1 (-12 total)

- a. (-6) see code block.
  - Explanation: My original answer is incorrect

- b. (-6) Original Answer (was cut off on original PDF file): My answers imply that the spectral designation of this star is around a G
  - New Answer: My answers to part (a) suggest that the spectral designation of this mystery star with a luminosity of about 37 solar units, a radius about 1.5 larger than the Sun's, and a temperature of ~12000 K would have an A spectral designation. The corresponding life expectancy of this kind of star is about 10^9 years.

[ ]:
```
''' a. Original Answer

_max = 250 * units.nm
```

```
T = (constants.b_wien / _max).si # T was good

alpha = (0.130 * units.arcsec).to(units.rad)
d = 1*units.rad / alpha # bad

I = 2e-8 * units.W / units.m**2
L = (4*np.pi*(d**2) * I).si
L = L / L_sun

A = L / I

R = (np.sqrt(A / (4*np.pi))).si
R = R / R_sun # also bad

print(T, L, R)
'''

'''New Answer
d = 1 AU /  [rad]
I = P/A =  T**4 --> A = L/(P/A) --> R = sqrt(A/4 )
--> A = P /  T**4, or, L /  T**4
--> R = sqrt(P/4 T**4)
'''
 _max = 250 * units.nm
T = (constants.b_wien /  _max).si

  = (0.130 * units.arcsec).to(units.radian)
x = (1 * units.AU)
d = (x /  ) * units.radian

I = 2e-8 * units.W / units.m**2
L = (4 * np.pi * (d**2) * I).si

# A = (L / I).si
# A = L / (sigma_sb * T**4)
R = np.sqrt(L/(4 * np.pi * sigma_sb * T**4)).si
# R = (np.sqrt(A / (4 * np.pi))).si
L_s = (L / L_sun).to(units.dimensionless_unscaled)
R_s = (R / R_sun).to(units.dimensionless_unscaled)

print(f"T: {T}\nL: {L_s} L_suns = {L.to(units.W)}\nR: {R_s} R_suns = {R.si} ")
```

```
T: 11591.087820740688 K
L: 36.98982838578108 L_suns = 1.4159706306076997e+28 W
R: 1.5081548201017698 R_suns = 1049223308.3448013 m
```

2. (-18, did not attempt)

- see code blocks below

[ ]:
```
''' Part (a)
v = 2 R / P
(M_1 + M_2)*P**2 = R**3
M_1/M_2 = v_2/v_1
a = P/2pi*sqrt(v1^2+v2^2)
'''
P = 1 * units.yr
v_1 = 0.22 * units.km / units.s
v_2 = 110 * units.km / units.s
  = (0.1 * units.arcsec).to(units.radian)


# distance
# x = (1 * units.AU)
# d = (x / )  * units.radian
a = ((P / (2 * np.pi)) * np.sqrt(v_1**2 + v_2**2)).to(units.AU)
v_ratio_m1_m2 = (v_2 / v_1).to(units.dimensionless_unscaled)

G.to(units.AU**3 / (units.yr**2)) # solar masses
m_sum = (4 * np.pi**2 * a**3) / (G * P**2)

# M_2 = ((a**3) /(P**2 * (v_ratio_m1_m2+1)))

# R_1 = ((v_1 * P) / (2*np.pi)).si
# R_2 = ((v_2 * P) / (2*np.pi)).si
# M_2.to(units.dimensionless_unscaled)
M_2 = m_sum / (v_ratio_m1_m2 + 1)
M_1 = m_sum - M_2
print(f"Mass 1: {M_1} solar masses\nMass 2: {M_2} solar masses")
```

```
Mass 1: 50.27001325595759 solar masses
Mass 2: 0.10054002651191518 solar masses
```

[ ]:
```
''' (b)
L is proportional to M^3.5
lifetime is inversely proportional to L
'''
L_1 = 8e5 * L_sun
L_2 = 3e-3 * L_sun
t_born = 1e6 * units.yr

L_ratio = (L_1 / L_2).to(units.dimensionless_unscaled)
lifetime_2 = L_ratio * t_born
lifetime_2
```

[ ]:
2.6666667×10^{14} yr