PEU 327 – Spring 2025
Observatinal Astrophysics Laboratory

Lab (6) Manual

Physics of Universe (PU) Program
University of Science and Technology at Zewail City

Contents

Spectroscopic Stellar Classification and Distribution

Grading Sheet

	Grade
Experiment (XI)	20
Experiment (XII)	20
LaTeX Report	5
Total	45

Experiment (XI)

Balmer Series of Hydrogen Atom & Classifications of Stars

Objectives

In this Experiment, you will be repeating the process that was developed at Harvard around the turn of the 20th century by investigating and classifying six stars. The resulting classification was a key step in elucidating the underlying physics that produces stellar spectra. The relatively mundane step of classification eventually yields critical insights that allow us to understand the world, so we say.

You will learn the basic techniques and criteria of the spectral classification sequence by

I. examining and classifying spectra according to the strength of the hydrogen Balmer absorption lines;

II. examining and classifying spectra according to temperature using Wien's displacement law;

III. comparing the two schemes by identifying the temperature where the Balmer lines are strongest and recognizing the corresponding spectral class;

IV. summarizing the physical reason why both very cool stars and very hot stars have weak Balmer lines.

Introduction

It is no secret; classification lies at the foundation of nearly every science. Scientists develop classification systems based on perceived patterns and relationships. Biologists classify plants and animals into subgroups called genus and species. Geologists have an elaborate system of classification for rocks and minerals. Astrophysicists are no different. We classify planets according to their composition (terrestrial or Jovian), galaxies according to their shape (spiral, elliptical, or irregular), and stars according to their spectra.

The spectrum of a star is composed primarily of blackbody radiation–radiation that produces a continuous spectrum (the continuum). The star emits light over the entire electromagnetic spectrum, from the x-ray to the radio. However, stars do not emit the same amount of energy at all wavelengths. The peak emission of their blackbody radiation comes at a wavelength determined by their surface temperature, the relationship known as Wien's displacement law in equation 2. Most stars put out the maximum amount of radiation in and around the optical part of the electromagnetic spectrum. Figure 1 shows three blackbody curves for stars of different temperatures. As the temperature drops, the relative flux decreases, and the peak moves from the blue (hot) to the red (cool) wavelength regions of the spectrum.

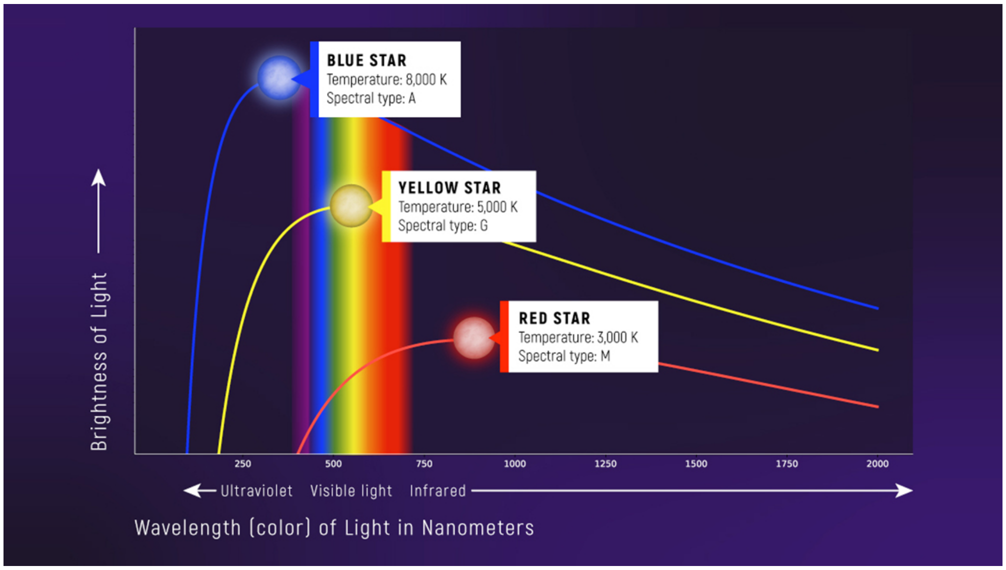


Figure 1: Continuous Spectra (Blackbody Curves) of Stars

In addition to the continuous spectrum, a star's spectrum will feature a number of either emission or absorption lines. Emission lines are produced by atoms when electrons drop from high energy levels to lower ones, emitting photons at specific frequencies in the process. This process adds radiation to the star's spectrum; emission lines are brighter than the region of the spectrum around them. Absorption lines are produced by atoms when their electrons absorb radiation at a specific frequency, thereby causing the electrons to move from a lower energy level to a higher one. This process removes some of the continuum being produced by the star and results in dark features in the spectrum. These lines are dimmer than the wavelength region around them.

Experimental Procedures

Note: In this lab, all the wavelengths are given in units of Ångstroms. 1 Ångstrom = 10^{-10} meters; 1 nm = 10 Ångstroms.

Task 1

Firstly, examine figure 2, where the electron transitions that correspond to the absorption/emission lines of the Lyman series, Balmer series, and Paschen series for the Hydrogen atom are provided.

(a) Calculate the identifying wavelengths (i.e., the wavelength of the photons the atom must absorb to allow the electron to make each transition) for both Balmer and

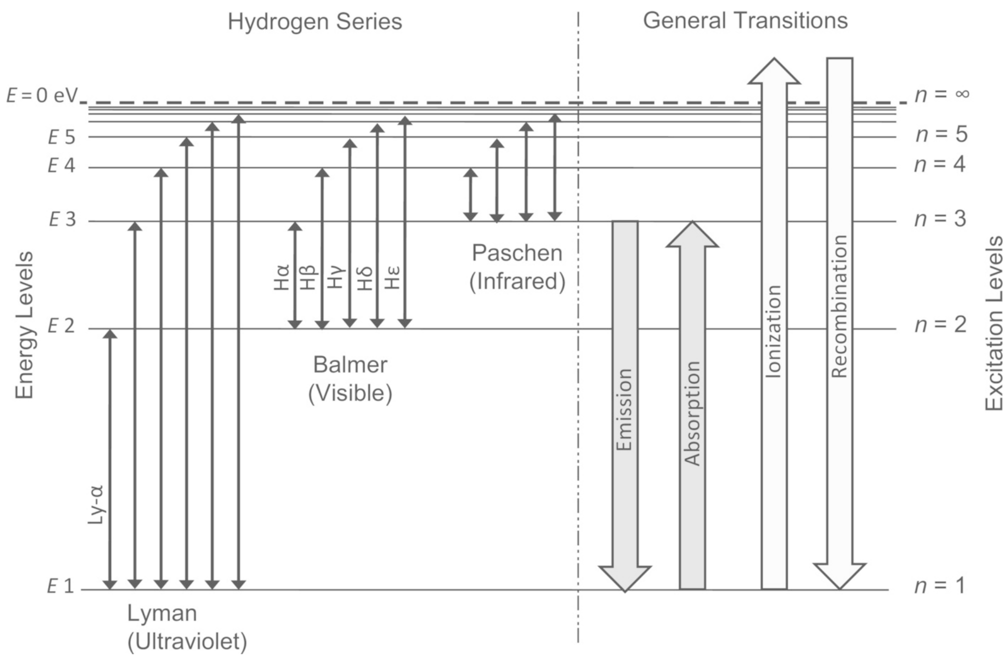


Figure 2: Orbital Energy Level Diagram for the Hydrogen Series

Lyman serieses using the Rydberg formula:

$$\frac{1}{\lambda_{if}} = R_{\mathrm{H}} \left(\frac{1}{n_f^2} - \frac{1}{n_i^2} \right) \tag{1}$$

where $R_{\mathrm{H}} = 1.0973731 \cdot 10^7 \, \mathrm{m}^{-1}$

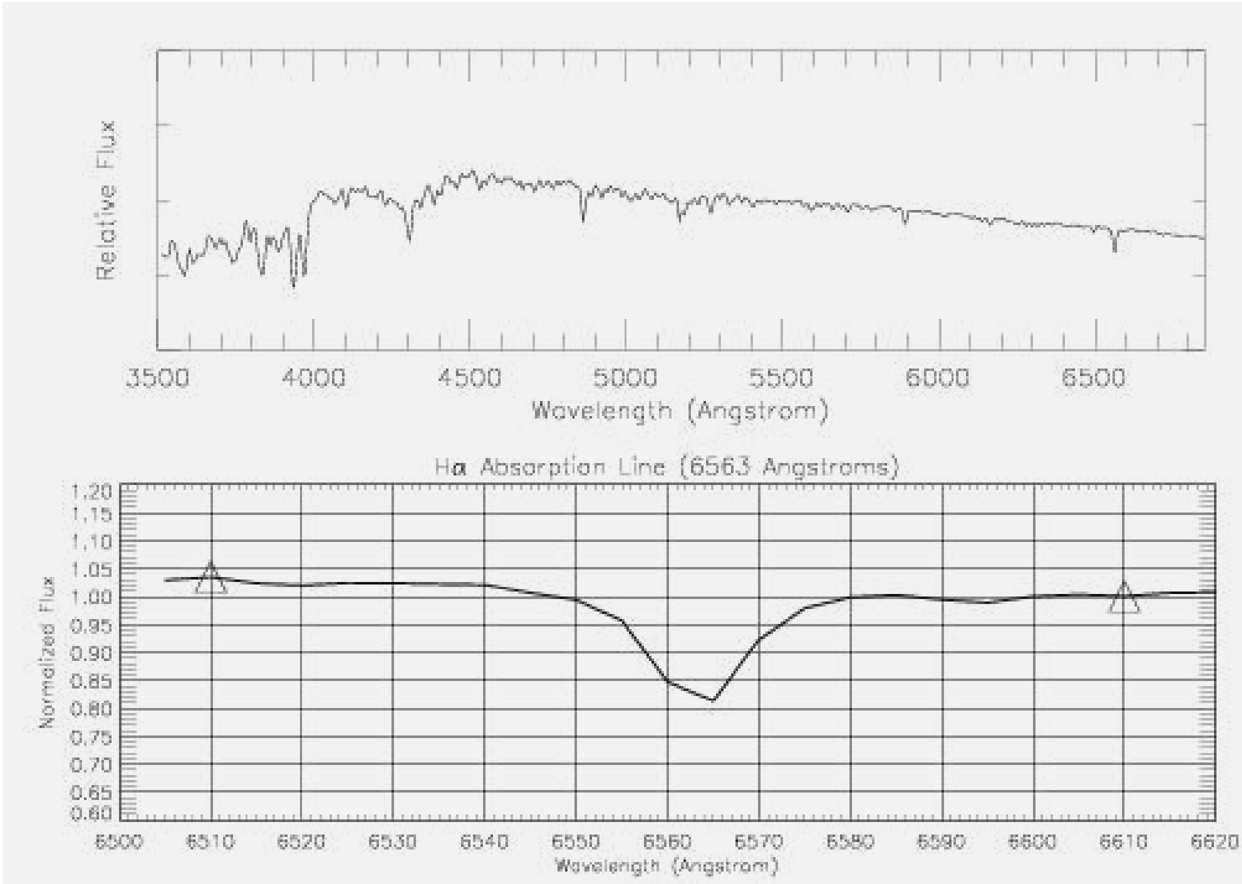
(b) An atom must absorb an ultraviolet photon to produce a Lyman series, an optical photon for the Balmer series, and an infrared photon for the Paschen series. Explain why the Lyman series requires ultraviolet photons as opposed to optical or infrared.

Task 2

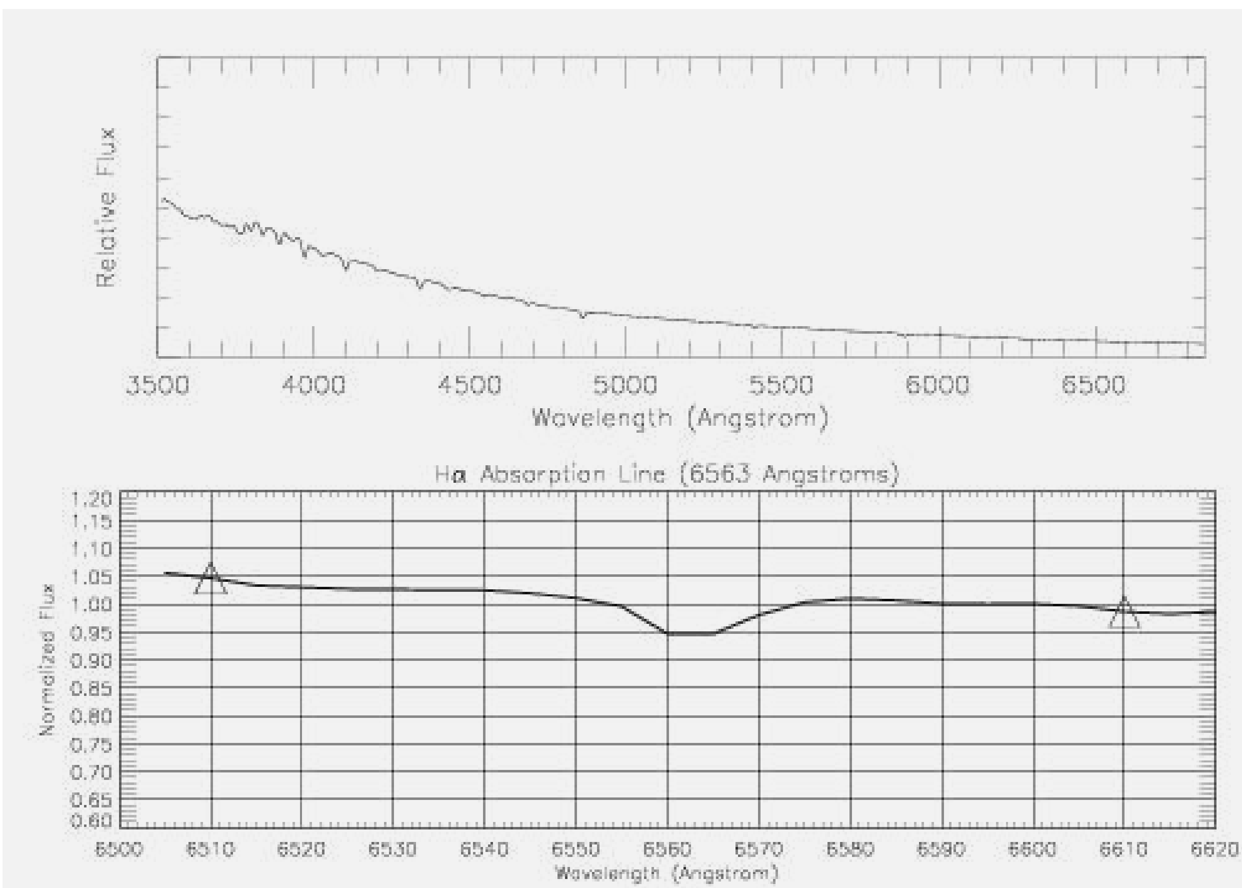
Examine the six panels of spectra on the following pages. Each panel shows the optical spectrum of six different stars. The top graph in each panel is the full spectrum; the bottom graph in each panel is a blowup of the Balmer series absorption line located at 6563 Ångstroms. In Astrophysics, we call this particular absorption line: Hydrogen-alpha (Hα) – because it is the first line in the Balmer series, and α is the first letter of the Greek alphabet – and it is created when an electron moves from the 2$^{\text{nd}}$ to the 3$^{\text{rd}}$ energy level. Note the difference between the scale on the wavelength axis for the top and bottom graphs.

(a) In one well written sentence, explain why Lyman and Paschen series absorption lines are not seen in each panels spectrum.

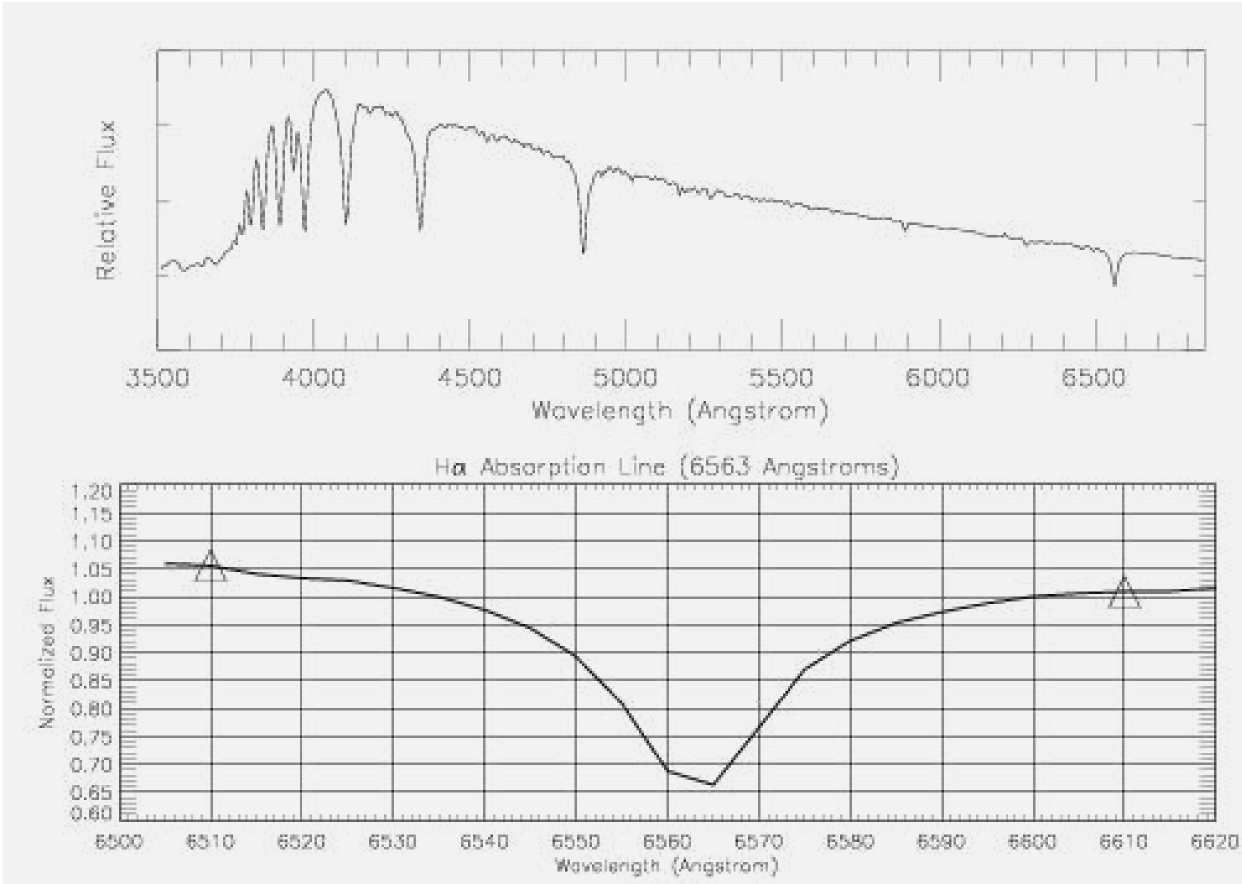
(b) In the top graph of each panel of spectra, circle each of the 3 Balmer series absorption lines you calculated in the first question.



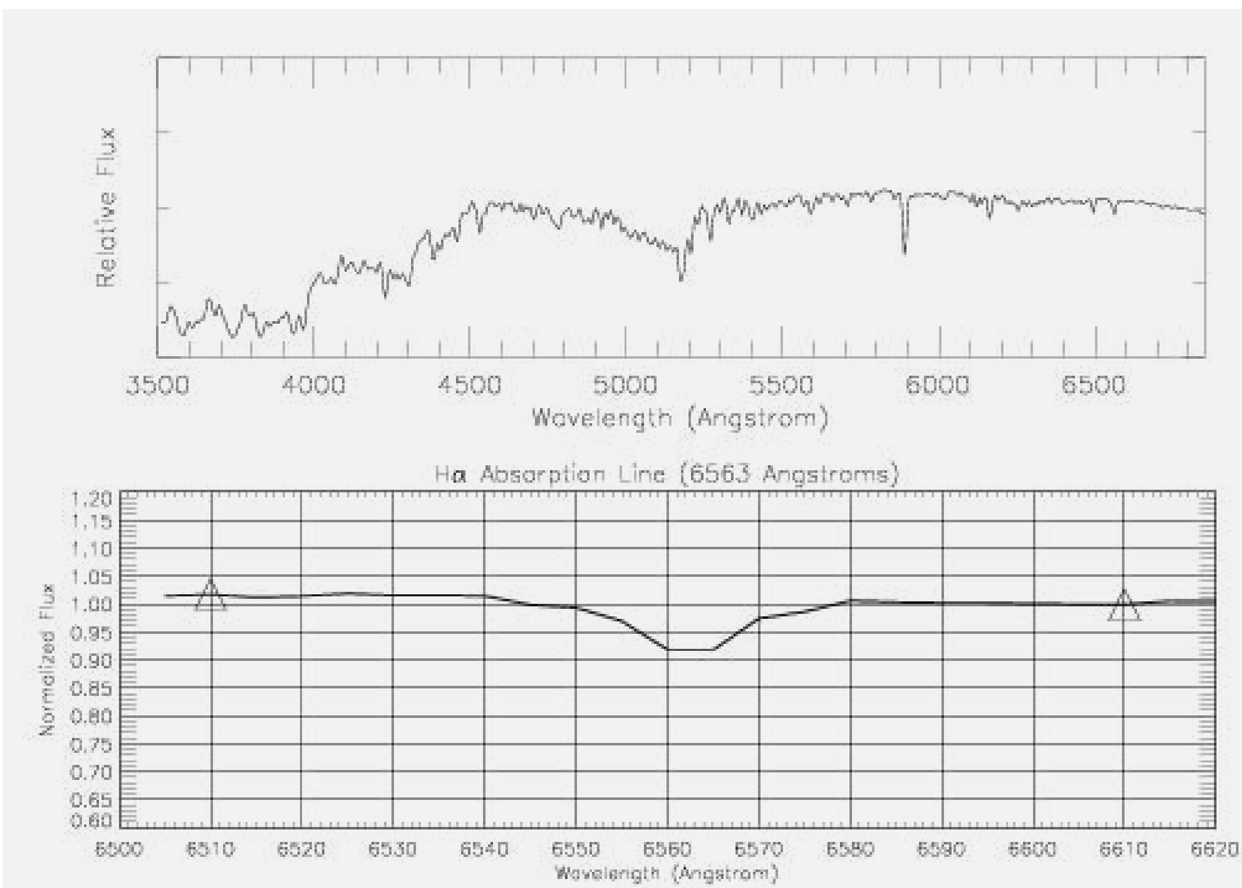
Panel 1



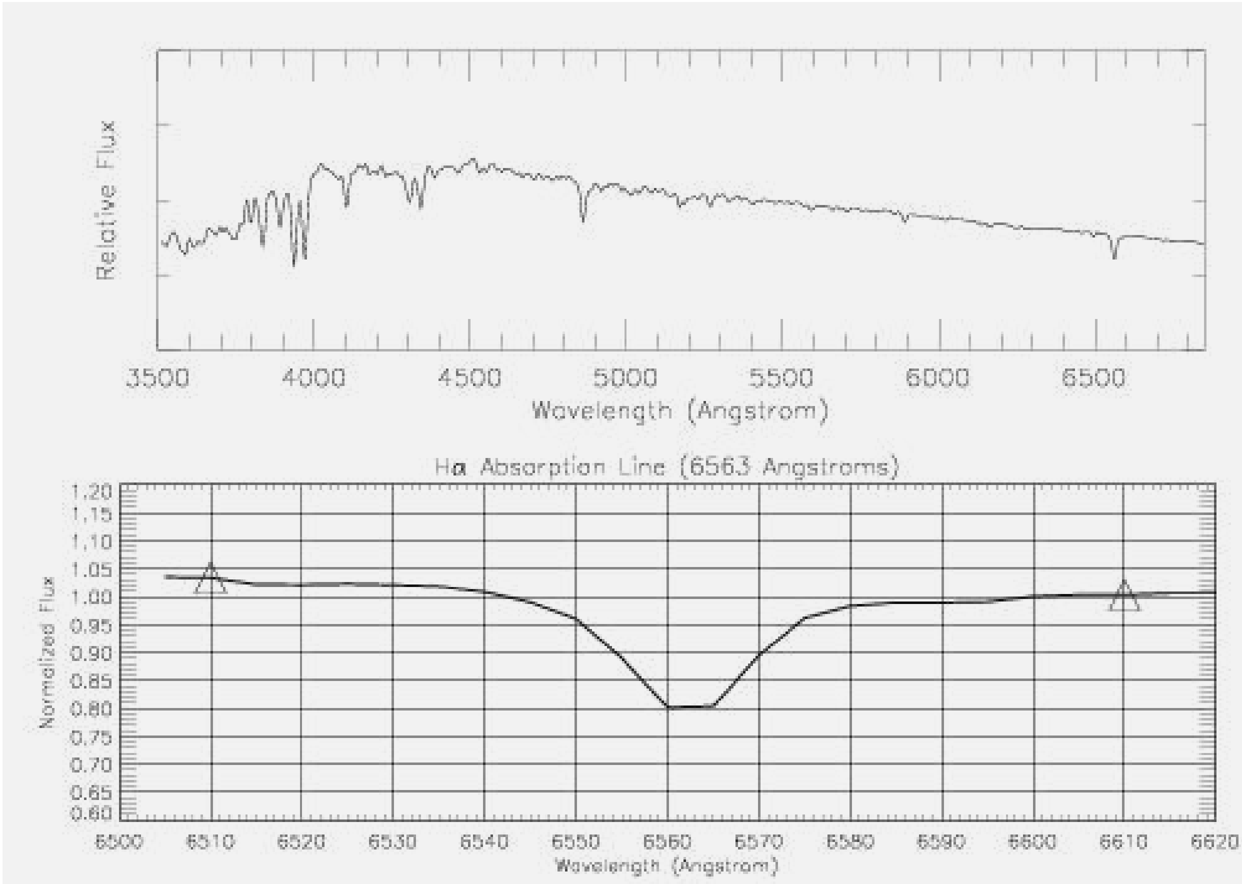
Panel 2



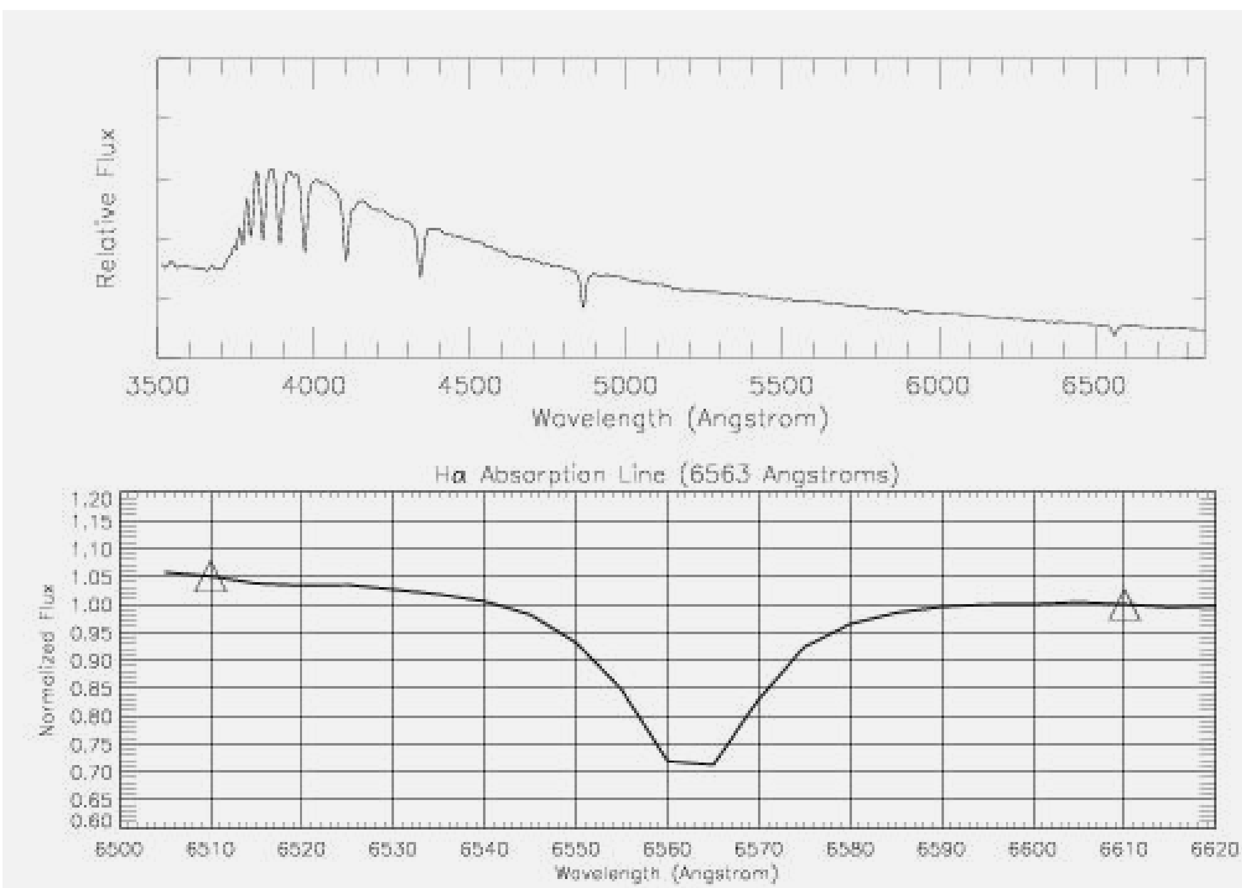
Panel 3



Panel 4



Panel 5



Panel 6

Task 3

To start, we are going to classify stars based on the 'strength' of their hydrogen absorption lines, specifically the Hα. Look at the blowup of the Hα in each panel. You will notice that each line dips down vertically and has a horizontal width associated with it. Absorption line 'strength' depends both on how much the line dips down and how wide it is. Measure the strength of each Hα in the bottom graph of each panel by following the procedures below:

(a) Draw a straight line in the bottom graph of each panel by connecting the triangles located at 6510 and 6610 Ångstroms. In doing so you are tracing the continuum near the absorption line.

(b) Lightly shade in the absorption line now enclosed by the continuum.

(c) The strength of an absorption line is the area enclosed by the absorption line. Measure the area by estimating the number of boxes contained inside of the absorption line. (This may be difficult to estimate as there will be "partial" boxes to consider; just do your best.) Record the strength (in number of boxes) of the absorption line in each panel in table 1.

Panel	Strength of Hα Line (No. of Boxes)
1	
2	
3	
4	
5	
6	

Table 1

Task 4

Originally, Astrophysicists classified those stars with the strongest hydrogen lines as 'A' stars; stars with the next strongest lines as 'B' stars; the next strongest 'C' and so on. Eventually some letters were deemed unnecessary and dropped from the classification sequence. The letter assigned to a star is termed its spectral class. Assign a spectral class to each panel by following the procedure below:

(a) In the second column of table 2, list the panel numbers from strongest Hα at the top to weakest Hα at the bottom.

(b) Each panel corresponds to one of the following classes of stars: A, B, F, G, K, and O. (We're skipping some letters here.) In the third column, mark the letter that corresponds to each panel using the described scheme.

Panel Number *Strongest* Hα to *Weakest*		Spectral Class Letter Designation
Strongest		
↓		
Weakest		

Table 2

Task 5

One can determine the surface temperature of each star (i.e. the temperature of each star's photosphere) using the full spectrum in the top graph of each panel and Wien's displacement law:

$$T = \frac{2.9 \cdot 10^7}{\lambda_{peak}} \tag{2}$$

where T is temperature in Kelvin and λ_{peak} is the wavelength in Ångstroms where the blackbody curve peaks. Determine the surface temperature of each star using the top graph of each panel by following the procedure below:

(a) Trace the underlying blackbody continuum in the top graph of each panel. This is what the spectrum would look like with no absorption lines (i.e. a perfect blackbody).

(b) In the second column of table 3 write down the wavelength (in Ångstroms) where the blackbody continuum in each panel peaks. If the peak is not shown on the graph, then write down a rough estimate of where you think the curve might peak.

Panel	Peak Wavelength (Å)	Surface Temperature (K)
1		
2		
3		
4		
5		
6		

Table 3

Task 6

It was later realized that the strength of a star's absorption lines can be predicted if the star's surface temperature is known. This is because the heat of a star can excite electrons up to higher energy levels. For example, most hydrogen atoms in very hot stars are ionized (the electron leaves the atom completely), and thus, show very weak Balmer Series absorption lines. Cool stars keep most of their electrons in the 1st energy level. The medium temperature stars show the strongest Balmer lines because most of their electrons start in the 2nd energy level.

Therefore, it is more intuitive to classify stars based on their temperature rather than on their Balmer lines alone. Astronomers reordered the classification sequence such that the hottest stars came first, but they retained the letters originally assigned to each star based on their Balmer line strengths.

(a) Reorder the classification sequence by listing the panel numbers from hottest star to coolest star in the first column of table 4.

(b) In the second column of table 4, write down the letter you assigned each panel in table 2.

	Panel Number *Hottest* Star to *Coolest*	Spectral Class Table 2 Letter Designation
Hottest		
↓		
Coolest		

Table 4

Discussion

7. What energy level must most hydrogen electrons start on to show Balmer absorption lines? What spectral class has the strongest Balmer lines? What Temperature does this correspond to?

8. Why do the hottest stars show weak Balmer lines? Why do the coolest stars show weak Balmer lines?

9. In addition to the 6 spectral types studied in this lab (A, B, F, G, K, and O), stars classified as L, M, and T also exist. L, M, and T stars are all cooler than K stars, with M stars being the hottest of the three, and T stars the coolest. Rewrite the classification sequence you found in table 4 to include L, M, and T stars.

10. In your own words, using good writing form, summarize why the Stellar Classification Sequence is not in alphabetical order.

Experiment (XII)

Milky Way Galaxy's Spectral Types & Stellar Distribution

Objectives

In this experiment, you will use equatorial and galactic coordinate systems to explore how stars of different spectral types (specifically classes O and G) are distributed throughout the Milky Way galaxy. You will be given a list of O-type and G-type stars. You will investigate the distribution of these two types of stars in the Galaxy and characterize and explain their distribution.

Experimental Procedures

For the purpose(s) of this experiment, the *SIMBAD Astronomical Database* will be queried via the Table Access Protocol (TAP) to gather some important stellar properties. TAP is an IVOA service protocol which defines a way to access general table data using queries, and return the results in VOTable.

Installation

First, use pip (or conda if you are using Anaconda) to install `astropy`, a community maintained common core package for astronomy in Python, and `astroquery`, a coordinated package in the Astropy Project (and `pandas`/`numpy`; however, they are probably installed if you have used Python before). It is highly recommended to check the `astroquery` documentation regarding the `Simbad` module and the `astropy` documentation.

```
pip install astroquery
pip install astropy
pip install --upgrade matplotlib
```

Secondly, import the `Simbad` object from the `simbad` module in `astroquery` to query the Simbad service and the modules: `units`, `coordinates`, `table` from astropy. We will revisit the latter ones after the first task.

```
from astroquery.simbad import Simbad
from astropy import units as u
import astropy.coordinates as coord
from astropy.table import QTable, Column
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

Task 1 Querying SIMBAD

Astronomical Data Query Language

ADQL, in short, is a query language based on the Structured Query Language (SQL) for astronomers. It allows us to retrieve information from astronomical databases according to our written query/program. Its syntax is straight forward.

Relevant ADQL Keywords:

- `SELECT` — Initializes every query, indicating data selection; hence, the name. It specifies the columns – separated by commas – to be fetched from their designated database tables.
- `FROM` — Specifies the table(s) from which said column(s) is/are to be extracted.
- `WHERE` — Specifies the rows/data that meet certain condition(s) to be included in the output; however, it is not necessary to include `WHERE` or conditions for a query program to "work". The number of rows is most likely limited by the service being queried though.

Note: ADQL is not case-sensitive but it is a best practice to capitalize its keywords and type column and table names in lowercase.

To write a query within Python code, we use String literals. Triple-quotes, `"""` or `'''`, are used at the beginning and end of any text strings of type `str` (e.g., a query prompt) to manipulate it while including line breaks. This is especially helpful for readability purposes (see the following basic use example).

```
query = """SELECT
                column_1,
                column_2,
                column_3 AS "New Column Heading",
                column_4,
                ...
                column_n

FROM table_1

WHERE condition_1, condition_2, ..., condition_n
"""
```

You can refer to SIMBAD's ADQL Cheat Sheet if needed.

SIMBAD Tables and Columns

SIMBAD is a relational database (i.e., a collection of linked tables, 30 of which to be exact). Hence, one must figure out which tables to include after the `FROM` keyword in the query. One way to show all of SIMBAD tables is the `list_tables` method.

```
Simbad.list_tables()
```

Out [3]:
```
Table length=30
table_name     description
object         object
basic          General data about an astronomical object
ids          all names concatenated with pipe
alltypes     all object types concatenated with pipe
ident        Identifiers of an astronomical object
cat          Catalogues name
flux         Magnitude/Flux information about an astronomical object
allfluxes    all flux/magnitudes U,B,V,I,J,H,K,u_,g_,r_,i_,z_
filter       Description of a flux filter
has_ref      Associations between astronomical objects and their
   bibliographic references
ref          Bibliographic reference
...            ...
mesPLX       Collection of trigonometric parallaxes.
otypedef     all names and definitions for the object types
mesIUE       International Ultraviolet Explorer observing log.
mesISO       Infrared Space Observatory (ISO) observing log.
mesFe_h      Collection of metallicity, as well as Teff, logg for stars.
mesDiameter Collection of stellar diameters.
mesDistance Collection of distances (pc, kpc or Mpc) by several means.
otypes       List of all object types associated with an object
mesSpT       Collection of spectral types.
journals     Description of all used journals in the database
```

One can also refer to SIMBAD's graphic representation of its tables which is depicted in figure 3.

Each one of these tables is a composite of columns. Similarly, the `list_columns` method can show the column names which should follow the `SELECT` keyword. However, calling it with no argument returns all 293-columns (as of March 2024) of SIMBAD. To restrict its output to columns within certain tables, add those tables' names in-between quotations, `"` or `'`, separated by commas.

```
Simbad.list_columns('table_1', 'table_2')
```

Example: Showing the metadata of the columns in the `basic` table

```
Simbad.list_columns('basic')
```

Out [4]:
```
Table length=67
table_name   column_name datatype   description                    unit
object       object       object     object                         object
basic        dec          DOUBLE     Declination                    deg
basic        main_id      VARCHAR    Main identifier for an object
basic        otype_txt    VARCHAR    Objet type
basic        ra           DOUBLE     Right ascension                deg
...          ...          ...        ...                            ...
```

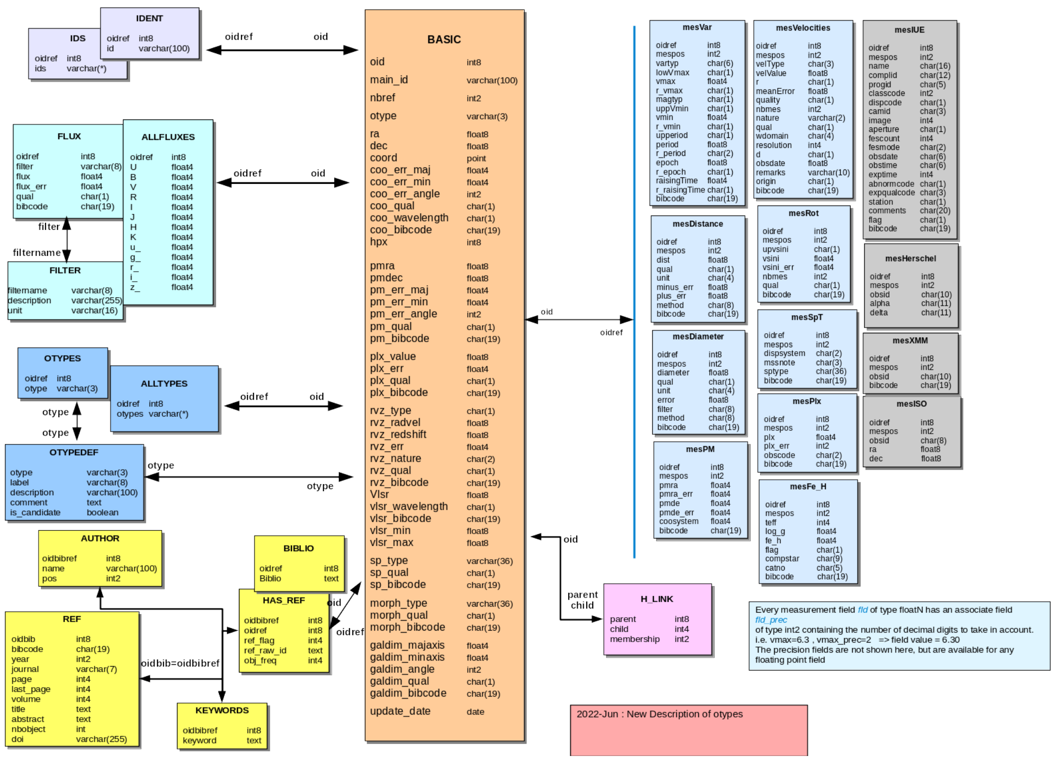


Figure 3: SIMBAD Tables

Your First Task: You are required to query SIMBAD (data collection) for the following data corresponding to the 43 stars provided in a string of text in Appendix A:

1. The equatorial coordinates, the right ascension (α) and declination (δ) — 2 columns.

2. The spectral type which includes both the spectral class (O# or G#) and luminosity class (I-V) — 1 column.

3. The apparent V-band magnitude (m_V) — 1 column.

The values of the absolute V-band magnitude, M_V, for each star, are also provided in Appendix A.

Note: It is a best practice to also record the id, identifier, and/or name of the objects in your data collection. In other words, query the column containing star identifiers and names as well.

Formatting Queries

Now, you might ask "Am I gonna write a condition for each star after the `WHERE` keyword?" The answer is NO. This will be slow, repetitive, and error-prone. Also, if you do so, you are subject to reach the character limit for the ADQL query. One can use the Python String `format()` Method to create formatted queries by embedding variables and values into a string placeholders.

Template: Writing a Python code that assembles a formatted query based on a string

```
string_text = """ 'star_1', 'star_2', ... """
query_base = """SELECT
                column_1,
                ...,
                column_n

FROM table_1
WHERE column_x IN ({string_placeholder})
"""
query = query_base.format(string_placeholder=string_text)
```

You should have noted by now that the ADQL criteria `IN` were used as a belong to, $\in$, operation. (It is worth noting that the $\in$ symbol is defined in LaTeX as `\in` as well.)

Also, the query was divided into two parts: a list strings and a "base "for the query. This base query contains one format specifier, `string_placeholder`, which is a placeholder for the list of strings assigned to the variable `string_text`.

To assemble the query, invoke `format()` on the base query to assign the values in `string_text` to `string_placeholder`. It is common to give variable that contains the list of strings and the variable in the format specifier the same name

Joining Tables

As previously stated, SIMBAD is a relational database. Therefore, the `JOIN` operation is used to join (get it) SIMBAD tables together using a common column. Then, it returns the rows for which the columns in these tables match. There is no guarantee that the corresponding rows of these tables are in the same order, so the `JOIN` operation involves some searching. Read up on Database Indexing for more information. Google it!

The `JOIN` operation specifies two arguments:

1. The name of the table to join with the table indicated by the `FROM` clause.

2. The common column(s) via the `ON` condition or `USING()`. The latter can be used only if the given column exists in both tables with exactly the same name.

```
SELECT main_id
FROM basic JOIN ident ON oidref =  oid
```

Tip: Join your tables with `ident` to query using all possible names of the objects in our list.

To find other possible joins, use the `list_linked_tables` method to show the link between various tables and the table `basic`.

Example: Showing common columns between the `ident` and `basic` table.

```
Simbad.list_linked_tables("ident")
```

Out [5]:

```
Table length=1
from_table  from_column target_table  target_column
object      object      object        object
ident       oidref      basic         oid
```

This means that the `oidref` column in table `ident` has a common column with `basic` by the name `oid`.

Note: You can debug your query prompt directly in the SIMBAD TAP Service website. However, you will still be using Python to conduct this experiment, utilizing the toolset `astropy` provided to your benefit.

Task 2 Plotting the Aitoff Projection

To investigate the stellar distribution across the Milky Way galaxy, we are going to create an Aitoff projection of 1) the stars' equatorial coordinate (α and δ) along with the path of the galactic plane 2) their Galactic longitude, l, (the azimuthal direction along the galactic plane, + to the left, - to the right) and Galactic latitude (b).

Matplotlib Plotting & NumPy Units

The most direct way to do this would usually be making those coordinates (columns) into a `SkyCoord` object and plotting them.

```
equatorial = coord.SkyCoord(ra = results['ra'], dec = results['dec'],
    frame='icrs', unit='deg')
```

Note: The square bracket operation, `[]`, was used in the command above to select the α and δ columns of the query results. You can also pinpoint a single value (e.g., `Quantity` object) corresponding to row 1 of column 'a' using table['a'][1].

Then, using the `transform_to()` method to transform the `SkyCoord` object to any coordinate system. It takes the frame parameter as a `str` class (e.g., `'galactic'`) and returns a `SkyCoord` object again which is optimum for our plotting purposes.

```
galactic = equatorial.transform_to('galactic')
```

However, if you plot it as is, you will find all the stars skewed to the right because the Aitoff projection in `matplotlib` needs the coordinates, in radians, between $-\pi$ and π, not 0 and 2π (their current state). Thus, we are gonna use the `astropy.coordinates.Angle` object to wrap the coordinates at 180 degrees using the `wrap_at()` method.

```
ra_rad = equatorial.ra.wrap_at(180 * u.deg).radian
dec_rad = equatorial.dec.radian

l_rad = galactic.l.wrap_at(180 * u.deg).radian
b_rad = galactic.b.radian
```

Template: Creating an Aitoff projection using `matplotlib` with hours for the "x-axis" labels

```
fig = plt.figure()
ax = fig.add_subplot(projection="aitoff")
xlab = ['14h','16h','18h','20h','22h','0h','2h','4h','6h','8h','10h']
ax.set_xticklabels(xlab)
ax.grid(True)
plt.title('Aitoff Projection')
fig.show()
```

Out [7]:

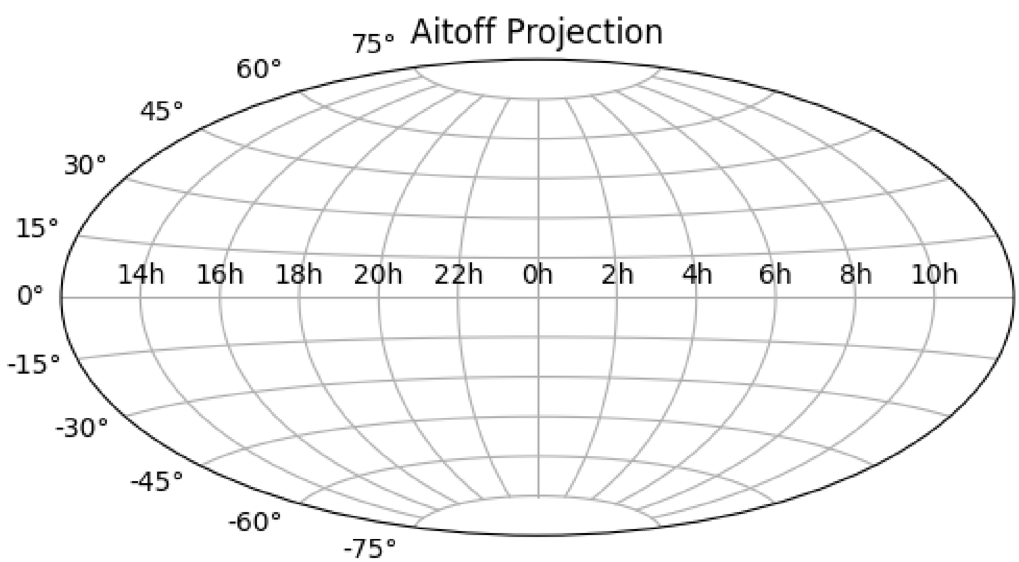


Figure 4: The Aitoff Projection

Tip: Use an `'o'`, circle, marker for the O-stars and `'x'`, cross, marker for the G-stars in the `pyplot.plot` of your Aitoff projection.

Coordinate Transformation

However, it is useful to understand the mathematical relationship between equatorial and galactic coordinate systems. Hence, in this task, you are required to define a conversion function using equations 3 to 5 below. Note carefully that, if you do use these equations in calculations of l and b in the future, be sure to carefully consider the quadrant of the output, because (co-)sinusoids are quadrant degenerate.

It is possible to derive conversions between different coordinate systems on the celestial sphere. You learned to convert between altitude-azimuth and equatorial coordinate systems. A similar analysis can be used to derive conversion equations between the equatorial and galactic coordinates systems. Since the equatorial coordinate system is geocentric (remember it was effectively an extension of the Earth's coordinate system onto the sky), it is often not especially handy when referring to sources outside of the solar system.

Figure 5 shows the relative orientations of these two coordinate systems. From this geometry it is possible to determine the equations of transformation using spherical geometry, as we discussed before.

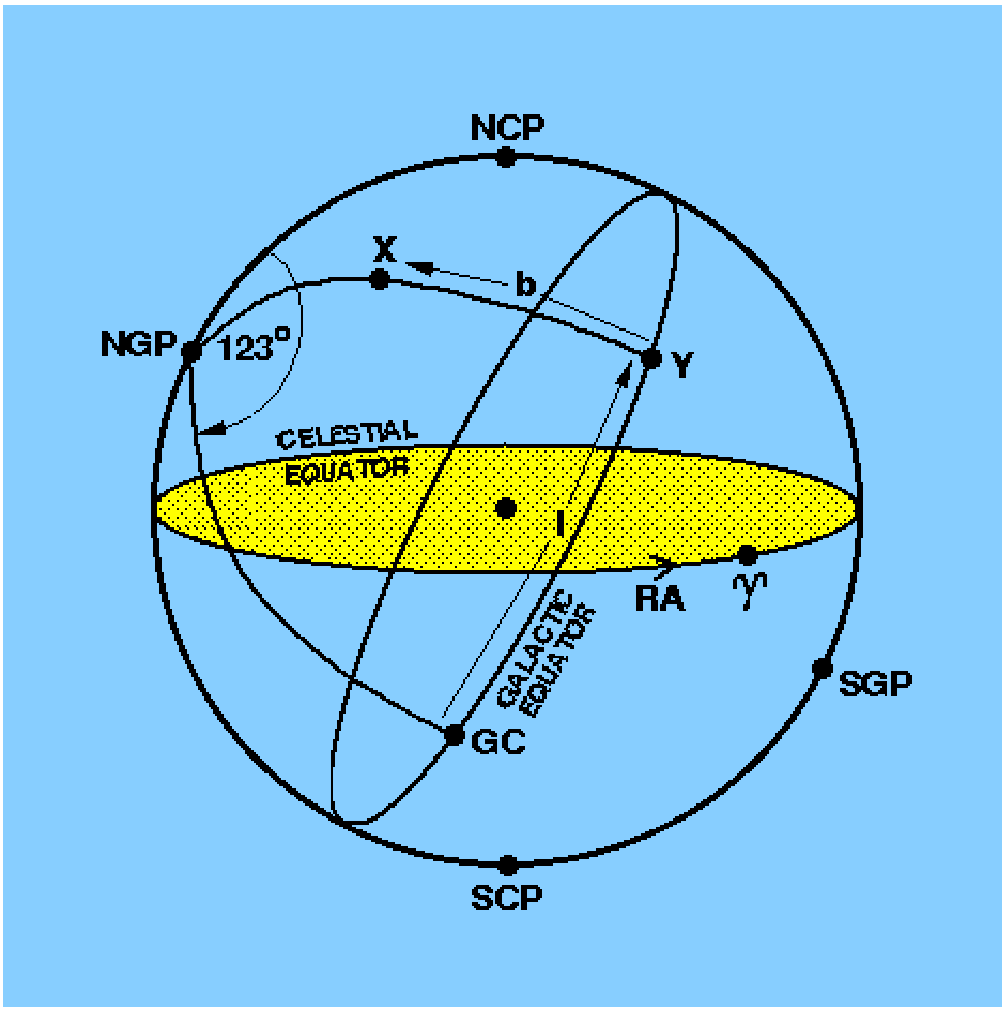


Figure 5: The Galactic Coordinate System

Tip: Set up a spherical triangle that has the North Galactic Pole (NGP; the direction the Galaxy's north rotation axis points toward), the North Celestial Pole (NCP), and the target as the three vertices.

Galactic-Equatorial Coordinates Conversion

$$\sin b = \sin\delta_P \sin\delta + \cos\delta_P \cos\delta \cos(\alpha - \alpha_P) \tag{3}$$

$$\tan(l - l_P) = \frac{\cos\delta \sin(\alpha - \alpha_P)}{\sin\delta_P \sin\delta + \cos\delta_P \cos\delta \cos(\alpha - \alpha_P)} \tag{4}$$

$$\begin{aligned} \alpha_P &= 192.9^\circ \\ \delta_P &= 27.1^\circ \\ l_P &= 122.9^\circ \end{aligned} \tag{5}$$

Your Second Task: Define a coordinate transformation function to calculate the stars' galactic coordinate. Add two columns for l and b to your query results. On an Aitoff projection map, plot

1. the O- and G-stars in equatorial coordinates (same figure as the galactic plane).
2. the galactic plane in equatorial coordinates (same figure as the stars).
3. the O- and G-stars in galactic coordinate (different figure).

Task 3 Histogram Representation

Create a histogram (number of objects within a given interval, or "bin", of a certain variable) by plotting the number of stars in a given b interval vs. different bin sizes of b ranges using the `matplotlib.pyplot.hist()` method. (You are advised to check its documentation.)

This should give you a sense of the angular offset for these stars in the direction perpendicular to the galactic plane.

Stellar distances may be obtained by comparing the apparent magnitude of a star to its absolute magnitude. Remember that the M_V, is the V-band magnitude the star would have if it were 10 pc away.

$$\begin{aligned} m_V - M_V &= -2.5\log\left(\frac{F_*}{F_* \cdot 10\,\text{pc}}\right) \\ &= -2.5\log\left(\frac{L_*/4\pi d_{\text{pc}}^2}{L_*/4\pi 10^2}\right) \\ &= -2.5\log\left(\frac{100}{d_{\text{pc}}^2}\right) \\ &= -5\log\left(\frac{10}{d_{\text{pc}}}\right) \\ \therefore d_{\text{pc}} &= 10 \cdot 10^{0.20(m_V - M_V)} \end{aligned} \tag{6}$$

Task 4 Distance Modulus

Add the absolute magnitudes provided in Appendix A with the rest of your data. From the absolute magnitude for each star, calculate the physical distance to each star and apend it to the final tabular data as well. Once you know the distance to the star, you can deduce from trigonometry and b, how far above or below the galactic plane the star resides. Recreate the histogram from Task 3, this time as a function of physical distance above or below the galactic plane.

Your Fourth Task:

1. Calculate the distance of each star and add its column to the tabular data, ending up with 8 columns, not counting the identifier column(s).

2. Plot a histogram of the physical distance between a star and the galactic plane.

Discussion

5. Investigating the stellar distribution of both the O- and G-stars in the galaxy, why is the distribution of O-stars different from that of the G-stars? Discuss your results agreement with your hypotheses on why certain types of stars, based on their masses, luminosities, makeups, etc... might exist only in certain parts of our galaxy (if that is what you observe), considering the relation between the main-sequence lifetimes and the mass or luminosity of the O-stars, given the approximate formula:

$$\tau_{\text{ms}} \approx 10^{10}\,\text{years}\frac{M}{M_\odot}\frac{L_\odot}{L}$$

and its effect on their stellar distribution although they were born with random velocities, σ, and travel through the galaxy throughout their life.

6. Deduce and report the correlation between the stars' absolute magnitudes and the luminosity classes of the stars (for both the O- and G-types).

Appendix A

The O- and G-Stars

```
'Alpha1 Centaurus', 'Alpha Auriga', 'Beta Cetus', 'Beta Corvus', 'Eta Bootes
', 'Eta Draco', 'Beta Hercules', 'Beta Draco', 'Zeta Hercules', 'Epsilon
Virgo', 'Beta Lepus', 'Beta Aquarius', 'Gamma Perseus', 'Eta Pegasus', '
Alpha Aquarius', 'Epsilon Leo', 'Gamma Hydra', 'Epsilon Gemini', 'Delta
Draco', 'Zeta Hydra', 'Zeta Cygnus', 'Epsilon Ophiuchus', 'Zeta Orion', '
Delta Orion', 'Zeta Puppis', 'Zeta Ophiuchus', 'Iota Orion', 'Lambda Orion',
 'Xi Perseus', 'Sigma Orion', 'Alpha Camelopard', 'Tau Canis Major', '10
Lacerta', '29 Canis Major', '68 Cygnus', 'Delta Circinus', 'Lamda Cepheus',
'19 Cepheus', '14 Cepheus', '9 Sagittarius', '9 Sagitta', '15 Monoceros', '
Theta2 Orion'
```

Their M_Vs

```
-6.99, -5.05, -2.71, -5.22, 0.32, -6.06, -5.11, -6.7, -3.88, -0.51, -7.39,
4.45, -3.47, -0.34, -0.51, -2.47, -0.53, -0.64, -6.46, 0.61, -4.84, -1.48,
-4.16, 0.61, 0.37, 2.38, 0.5, -1.19, -0.09, -1.58, -5.3, -4.73, -3.47,
-4.05, -3.75, -5.11, -0.57, -0.13, 2.68, -0.22, -3.24, -5.15, -5.96
```

Note: These magnitudes are ordered in correspondance to the `main_id` column resulted from SIMBAD.