Plot_HRD_with_student_stars_v1

June 10, 2026

[1]:
```
#!pip install jupyterlab plotly anywidget
```

[2]:
```
import requests
import webbrowser
from bs4 import BeautifulSoup

def wikisearchurl(tname):
    # 1. Define the Wikipedia URL
    #url = "https://en.wikipedia.org/wiki/Python_(programming_language)"
    url = f"https://en.wikipedia.org/w/index.php?
 ↪search={tname}&title=Special%3ASearch&fulltext=1&ns0=1"

    # 2. Fetch the page with a User-Agent to avoid 403 Forbidden errors
    headers = {'User-Agent': 'Mozilla/5.0'}
    response = requests.get(url, headers=headers)

    # 3. Check for successful request and parse HTML
    if response.status_code == 200:
        soup = BeautifulSoup(response.text, 'html.parser')

        # 4. Find all <a> tags with an 'href' attribute
        # Filters: starts with '/wiki/' and does not contain a ':'
        wiki_links = []
        for link in soup.find_all('a', href=True):
            href = link['href']
            if href.startswith('/wiki/') and ':' not in href:
                full_url = f"https://en.wikipedia.org{href}"
                wiki_links.append(full_url)

        # 5. Display the first 10 extracted URLs
        print(*wiki_links[:10], sep="\n")
        print("\nExamine these results above and click any related links ...\n")
        print("\nWhen all else has failed Open Google Search by executing next␣
 ↪cell!")

def wikihdsearch(tname):
    # 1. Define the Wikipedia URL
```

```
    #url = "https://en.wikipedia.org/wiki/Python_(programming_language)"
    url = f"https://en.wikipedia.org/wiki/{tname}"

    # 2. Fetch the page with a User-Agent to avoid 403 Forbidden errors
    headers = {'User-Agent': 'Mozilla/5.0'}
    response = requests.get(url, headers=headers)

    # 3. Check for successful request and parse HTML
    if response.status_code == 200:
        soup = BeautifulSoup(response.text, 'html.parser')

        # 4. Find all <a> tags with an 'href' attribute
        # Filters: starts with '/wiki/' and does not contain a ':'
        wiki_links = []
        for link in soup.find_all('a', href=True):
            href = link['href']
            if href.startswith('/wiki/Main_Page') and ':' not in href:
                full_url = f"https://en.wikipedia.org{href}"
                wiki_links.append(full_url)

        # 5. Display the first 10 extracted URLs
        print(*wiki_links[:10], sep="\n")
        print("\nExamine these results above and click any related links ...\n")
        print("\nIf star not found try a broader wiki search by executing next␣
 ↪cell!")

#wikisearchurl(user_input)
```

[3]:
```
import plotly.graph_objects as go
from plotly.subplots import make_subplots
import pandas as pd
import numpy as np

# --- 1. DATA PREP & SAFETY FILTERING ---
df_gaia = pd.read_csv('data_folder/gaia-hrd-dr3-200pc_100000_stars.csv')

# FIX #2: Eliminate non-positive parallaxes to prevent division by zero or INF
df_gaia = df_gaia[df_gaia['parallax'] > 0].copy()

# Ensure distance and MG are calculated for the background
df_gaia['dist'] = 1000 / df_gaia['parallax']
#df_gaia['abs_mag'] = df_gaia['phot_g_mean_mag'] - 5 * np.
 ↪log10(df_gaia['dist']) + 5
df_gaia['abs_mag'] = df_gaia['mg'] # already converted

# --- 2. SAMPLE SIZE & INITIAL SETUP ---
# Default starting sample size
```

```
initial_sample_size = 100000
df_sampled = df_gaia.sample(n=min(initial_sample_size, len(df_gaia)))

fig = make_subplots(specs=[[{"secondary_y": True}]])

# 3. BACKGROUND TRACE
fig.add_trace(go.Scattergl(
    x=df_sampled['bp_rp'], y=df_sampled['abs_mag'],
    mode='markers', name='Gaia Field Stars',
    marker=dict(color='rgba(150, 150, 150, 0.2)', size=1.5),
    hoverinfo='skip'
), secondary_y=False)

# (Add Student Stars and Sun traces here as in previous versions...)

# --- 4. INTEGRATED SLIDERS (Distance + Sample Size) ---
# FIX #1: Added a slider to control the density of background stars
# Note: Sample Size slider updates the "visible" data points
sample_options = [10000, 50000, 100000, 250000]

steps = []
for size in sample_options:
    temp_sample = df_gaia.sample(n=min(size, len(df_gaia)))
    step = dict(
        method="update",
        label=f"Sample: {size//1000}k",
        args=[{"x": [temp_sample['bp_rp']], "y": [temp_sample['abs_mag']]}]
    )
    steps.append(step)

fig.update_layout(
    sliders=[dict(active=2, currentvalue={"prefix": "Stellar Density: "},␣
 ↪steps=steps)],
    template="plotly_dark",
    # ... (Rest of layout with Age Axis and Spectral Class labels)
)

# --- 5. ASTROQUERY TIMEOUT (For the Search Tool) ---
# FIX #3: Setting a 60-second timeout for Simbad/Gaia remote queries
from astroquery.utils.tap.core import TapPlus
# Gaia.set_config(timeout=60)
```

1 Lab Tool: Targeted Gaia DR3 Search

1.0.1 Resolving proper names and HD catalog identifiers

This tool allows you to input a star's common name or catalog ID. It uses the **Simbad** resolver to find coordinates and then performs a **Cone Search** on the Gaia DR3 archive.

Example inputs: * `Alioth` (Proper name) * `HD 10307` (Henry Draper catalog) * `35 Sex` (Flamsteed designation for 35 Sextantis)

[4]:
```
from astroquery.gaia import Gaia
from astropy.coordinates import SkyCoord
import astropy.units as u
import pandas as pd

def get_gaia_data_by_name(target_name, radius_arcsec=5):
    """
    Resolves a name via Simbad and queries Gaia DR3 within a small radius.
    """
    try:
        # 1. Resolve coordinates from name
        print(f"Resolving coordinates for: {target_name}...")
        coords = SkyCoord.from_name(target_name)

        # 2. Perform Gaia Cone Search
        print(f"Searching Gaia DR3 within {radius_arcsec} arcsec of RA:{coords.
 ↪ra.deg:.4f}, Dec:{coords.dec.deg:.4f}...")
        job = Gaia.cone_search_async(coords, radius=radius_arcsec * u.arcsec)
        result = job.get_results()

        if len(result) == 0:
            return f"No Gaia sources found for '{target_name}'."

        # 3. Convert to Pandas for easy use
        df = result.to_pandas()
        return df

    except Exception as e:
        return f"Error resolving target: {e}"

def get_student_stars():
    mode = input("Enter 'csv' to load a file or 'single' for manual entry: ").
 ↪lower()

    if mode == 'csv':
        path = input("Enter the CSV filename (e.g., student_stars.csv): ")
        df = pd.read_csv(path)
    else:
```

```
        user_input = input("Enter Star Name or ID (e.g., Alioth, HD 10307, 35␣
↪Sex): ")
        target_df = get_gaia_data_by_name(user_input)
        wikisearchurl(tname)
        if isinstance(target_df, pd.DataFrame):
            #print(f"\nSuccess! Found {len(target_df)} source(s).")
            target_df['tgt_name'] = user_input

            # get more properties from wikipedia
            wikisearchurl(user_input)

            # enter properties from your analysis of spectrum or wikipedia

            user_spec_type = input("Spectral Type (e.g., K2III): ")
            target_df['spec_type'] = user_spec_type

            user_b_v = float(input("Enter B-V Color Index: "))
            target_df['b_v'] = user_b_v

    # Process the data using our existing physics engine
    return target_df
```

```
In preparation for Gaia DR4, the Gaia archive is in evolution. Unfortunately, it
may be unstable at times and particular types of queries may time out. Please
consider registering for a user account (https://www.cosmos.esa.int/web/gaia-
users/register). For questions or advice, please contact the Gaia helpdesk
(https://www.cosmos.esa.int/web/gaia/gaia-helpdesk).
```

1.1 Workflow for Students:

1. **Input Name:** Enter the target exactly as it appears in Wikipedia or catalog lists.
2. **Cross-Reference:** Check the `parallax` and `bp_rp` columns in the output table.
3. **Integrate:** Copy these values into your `student_observations.csv` to see them appear on your interactive HR Diagram!

[ ]:

[ ]:

2 Lab: The Interactive Stellar Explorer

2.0.1 Integrating BARO Telescope Spectra with Gaia Space Mission Data

---

2.1 1. The Scientific Workflow

In this lab, you will combine professional-grade datasets to build a dynamic Hertzsprung-Russell Diagram (HRD).

Your Goal: Overlay your personal spectroscopic observations onto a background of 500,000 local stars to determine the age, temperature, and size of your targets.

2.1.1 The Physics Pipeline:

1. **Distance (d):** Calculated as 1000/parallax (mas).
2. **Absolute Magnitude (M_G):** Calculated using the Distance Modulus: $M_G = G_{\text{mag}} - 5\log_{10}(d) + 5$.
3. **Color Conversion:** Converting your Johnson $B-V$ index to the Gaia $BP-RP$ system using a 3rd-order polynomial.
4. **Physical Properties:** Estimating Temperature (T_{eff}) and Radius ($R_\odot$) for interactive tooltips.

[5]:
```
import plotly.graph_objects as go
from plotly.subplots import make_subplots
import pandas as pd
import numpy as np
```

2.2 3. Creating the Interactive Dashboard

This cell builds the dual-axis plot with the distance slider. Note the use of `Scattergl` for the 500k background stars to ensure smooth performance.

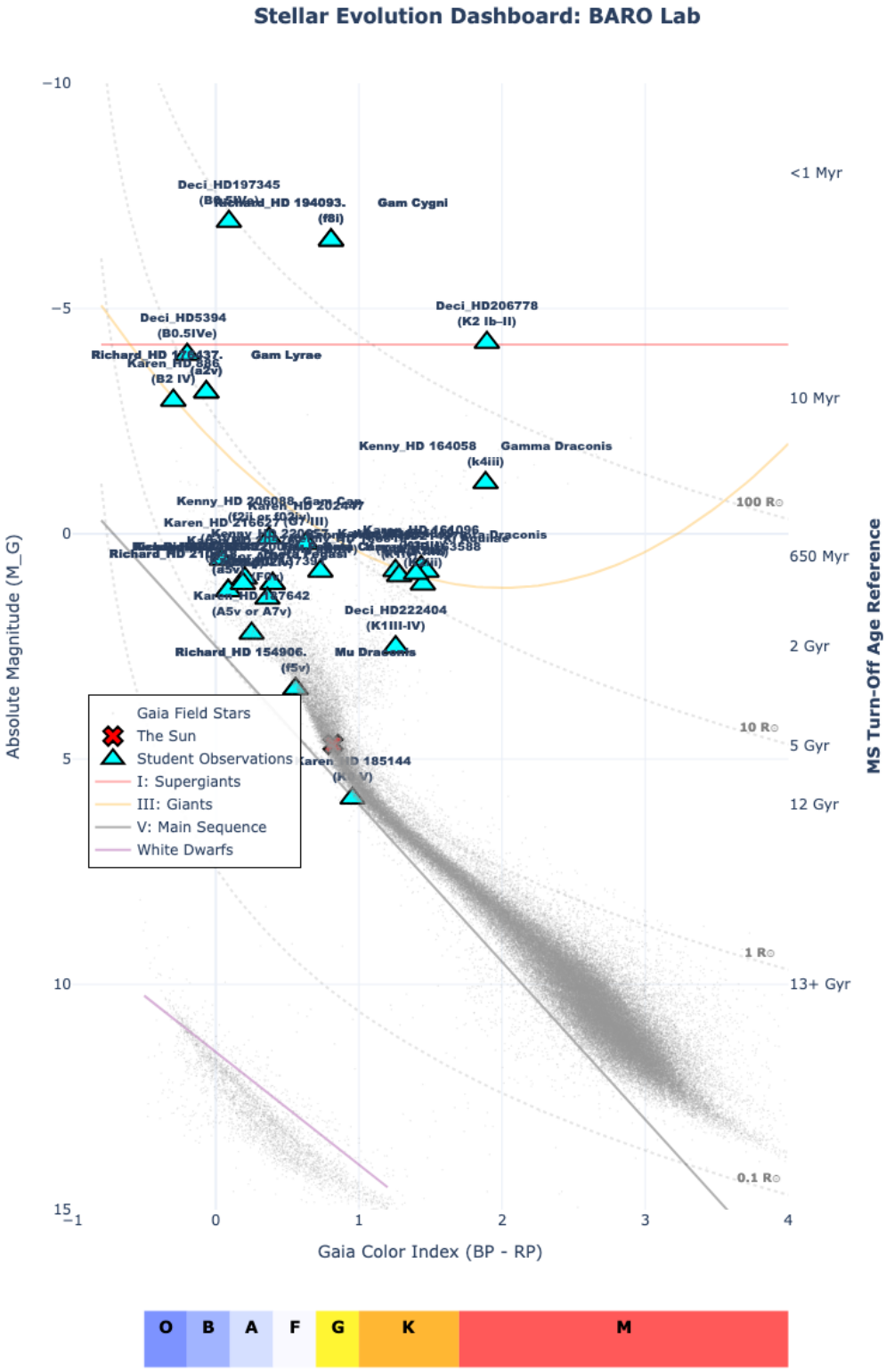
[6]:
```
# --- 1. DATA INPUT ---
print("--- BARO Spectrography Data Entry ---")
mode_in = input("Enter 'csv' or 'manual' [Default: csv]: ").lower().strip()
mode = mode_in if mode_in else "csv"

if mode == 'csv':
    path_in = input("Enter filename [Default: student_stars.csv]: ").strip()
    path = path_in if path_in else "student_stars.csv"
    df_students = pd.read_csv(path)
else:
    star_list = []
    while True:
        name = input("\nStar Name (or 'n' to plot): ")
        if name.lower() == 'n': break
        bv_raw = input(f"B-V for {name}: ").replace('−', '-').strip()
        bv = float(bv_raw) if bv_raw else 0.0
        mag_raw = input(f"Apparent Mag (G) for {name}: ").replace('−', '-').
 ↪strip()
        app_mag = float(mag_raw) if mag_raw else 5.0
        plx_raw = input(f"Parallax (mas) for {name}: ").replace('−', '-').
 ↪strip()
        parallax = float(plx_raw) if plx_raw else 10.0
        spec = input(f"Spectral Type for {name}: ")
        star_list.append({'star_name': name, 'b_v': bv, 'app_mag': app_mag,␣
 ↪'parallax': parallax, 'spec_type': spec})
```

```
    df_students = pd.DataFrame(star_list)

# --- 2. PHYSICS ENGINE ---
df_students['bp_rp'] = -0.0176 + 1.2156*df_students['b_v'] + 0.
 ↪0149*(df_students['b_v']**2) + 0.0058*(df_students['b_v']**3)
df_students['dist'] = 1000 / df_students['parallax']
df_students['abs_mag'] = df_students['app_mag'] - 5 * np.
 ↪log10(df_students['dist']) + 5
df_students['temp_k'] = 9000 / (df_students['bp_rp'] + 0.93)
lum_ratio = 10**((4.67 - df_students['abs_mag']) / 2.5)
df_students['radius_solar'] = np.sqrt(lum_ratio) * (5778 /␣
 ↪df_students['temp_k'])**2
df_students['hover_text'] = [f"<b>{r['star_name']}</b><br>Type:␣
 ↪{r['spec_type']}<br>Temp: {int(r['temp_k'])}K<br>Radius: {r['radius_solar']:.
 ↪2f}R " for _, r in df_students.iterrows()]

# --- 3. BACKGROUND & PLOT SETUP ---
mask = (df_gaia['bp_rp'] >= -1) & (df_gaia['bp_rp'] <= 4) & (df_gaia['abs_mag']␣
 ↪<= 15) & (df_gaia['abs_mag'] >= -10)
df_bg = df_gaia[mask].sample(n=min(100000, len(df_gaia[mask])))

fig = make_subplots(specs=[[{"secondary_y": True}]])

# IMPORTANT: Define these lists BEFORE the loops
all_shapes = []
all_annotations = []

# Traces (Background, Sun, Students)
fig.add_trace(go.Scattergl(x=df_bg['bp_rp'], y=df_bg['abs_mag'],␣
 ↪mode='markers', name='Gaia Field Stars', marker=dict(color='rgba(150, 150,␣
 ↪150, 0.2)', size=1.5)), secondary_y=False)
fig.add_trace(go.Scatter(x=[0.82], y=[4.67], mode='markers', name='The Sun',␣
 ↪marker=dict(symbol='x', size=14, color='red', line=dict(width=2,␣
 ↪color='black')), hovertext="Sun (G2V)", hoverinfo='text'), secondary_y=False)
labels = [f"{n}<br>({s})" for n, s in zip(df_students['star_name'],␣
 ↪df_students['spec_type'])]
fig.add_trace(go.Scatter(x=df_students['bp_rp'], y=df_students['abs_mag'],␣
 ↪mode='markers+text', name='Student Observations', text=labels,␣
 ↪hovertext=df_students['hover_text'], textposition="top center",␣
 ↪textfont=dict(family="Arial Black", size=10),␣
 ↪marker=dict(symbol='triangle-up', size=18, color='cyan', line=dict(width=2,␣
 ↪color='black')), hoverinfo='text'), secondary_y=True)

# Evolutionary Curves
color_range = np.linspace(-0.8, 4.0, 100)
```

```
lum_curves = [{'n': 'I: Supergiants', 'y': -4.2 * np.ones_like(color_range),␣
 ↪'c': 'rgba(255, 0, 0, 0.3)'}, {'n': 'III: Giants', 'y': -0.8 * (color_range␣
 ↪- 2.0)**2 + 1.2, 'c': 'rgba(255, 165, 0, 0.3)'}, {'n': 'V: Main Sequence',␣
 ↪'y': 3.5 * color_range + 2.5, 'c': 'rgba(0, 0, 0, 0.3)'}, {'n': 'White␣
 ↪Dwarfs', 'x': np.linspace(-0.5, 1.2, 50), 'y': 2.5 * np.linspace(-0.5, 1.2,␣
 ↪50) + 11.5, 'c': 'rgba(128, 0, 128, 0.3)'}]
for c in lum_curves:
    fig.add_trace(go.Scatter(x=c.get('x', color_range), y=c['y'], mode='lines',␣
 ↪line=dict(color=c['c'], width=2), name=c['n']), secondary_y=False)

# Isoradii
for r in [0.1, 1, 10, 100]:
    iso_mg = 4.67 - 2.5 * np.log10((r**2) * ((9000/(color_range+0.93))/5778)**4)
    fig.add_trace(go.Scatter(x=color_range, y=iso_mg, mode='lines',␣
 ↪line=dict(color='rgba(0,0,0,0.1)', dash='dot'), showlegend=False),␣
 ↪secondary_y=False)
    all_annotations.append(dict(x=3.8, y=iso_mg[-10], text=f"<b>{r} R </b>",␣
 ↪showarrow=False, font=dict(size=10, color="gray")))

# Spectral Strip
spectral_data = [{'class':'O','s':-0.5,'e':-0.2,'c':'#5d78ff'},{'class':'B','s':
 ↪-0.2,'e':0.1,'c':'#89a1ff'},{'class':'A','s':0.1,'e':0.4,'c':
 ↪'#cad7ff'},{'class':'F','s':0.4,'e':0.7,'c':'#f8f7ff'},{'class':'G','s':0.
 ↪7,'e':1.0,'c':'#fff200'},{'class':'K','s':1.0,'e':1.7,'c':
 ↪'#ffa500'},{'class':'M','s':1.7,'e':4.0,'c':'#ff3030'}]
for s in spectral_data:
    all_shapes.append(dict(type="rect", xref="x", yref="paper", x0=s['s'],␣
 ↪x1=s['e'], y0=-0.14, y1=-0.09, fillcolor=s['c'], opacity=0.8, line_width=0))
    all_annotations.append(dict(x=(s['s']+s['e'])/2, y=-0.115, xref="x",␣
 ↪yref="paper", text=f"<b>{s['class']}</b>", showarrow=False,␣
 ↪font=dict(color="black", size=14)))

# --- 4. FINAL LAYOUT ---
fig.update_layout(
    template="plotly_white", title=dict(text="<b>Stellar Evolution Dashboard:␣
 ↪BARO Lab</b>", x=0.5),
    height=1200, width=800, margin=dict(b=160, r=120, t=100),
    xaxis=dict(title="Gaia Color Index (BP - RP)", range=[-1, 4]),
    yaxis=dict(title="Absolute Magnitude (M_G)", range=[15, -10],␣
 ↪zeroline=False),
    yaxis2=dict(title="<b>MS Turn-Off Age Reference</b>", side="right",␣
 ↪overlaying="y", tickvals=[-8, -3, 0.5, 2.5, 4.7, 6, 10], ticktext=['<1 Myr',␣
 ↪'10 Myr', '650 Myr', '2 Gyr', '5 Gyr', '12 Gyr', '13+ Gyr'], range=[15,␣
 ↪-10], showgrid=False, autorange=False),
    legend=dict(
        x=0.02,
```

```
        y=0.38, # Shifted down slightly to clear the White Dwarf gap
        bgcolor="rgba(255,255,255,0.9)",
        bordercolor="black",
        borderwidth=1
    ),
    shapes=all_shapes, annotations=all_annotations, hovermode='closest'
)
fig.show()
```

```
--- BARO Spectrography Data Entry ---

Enter 'csv' or 'manual' [Default: csv]:  csv
Enter filename [Default: student_stars.csv]:  combined_final_student_6.csv
```

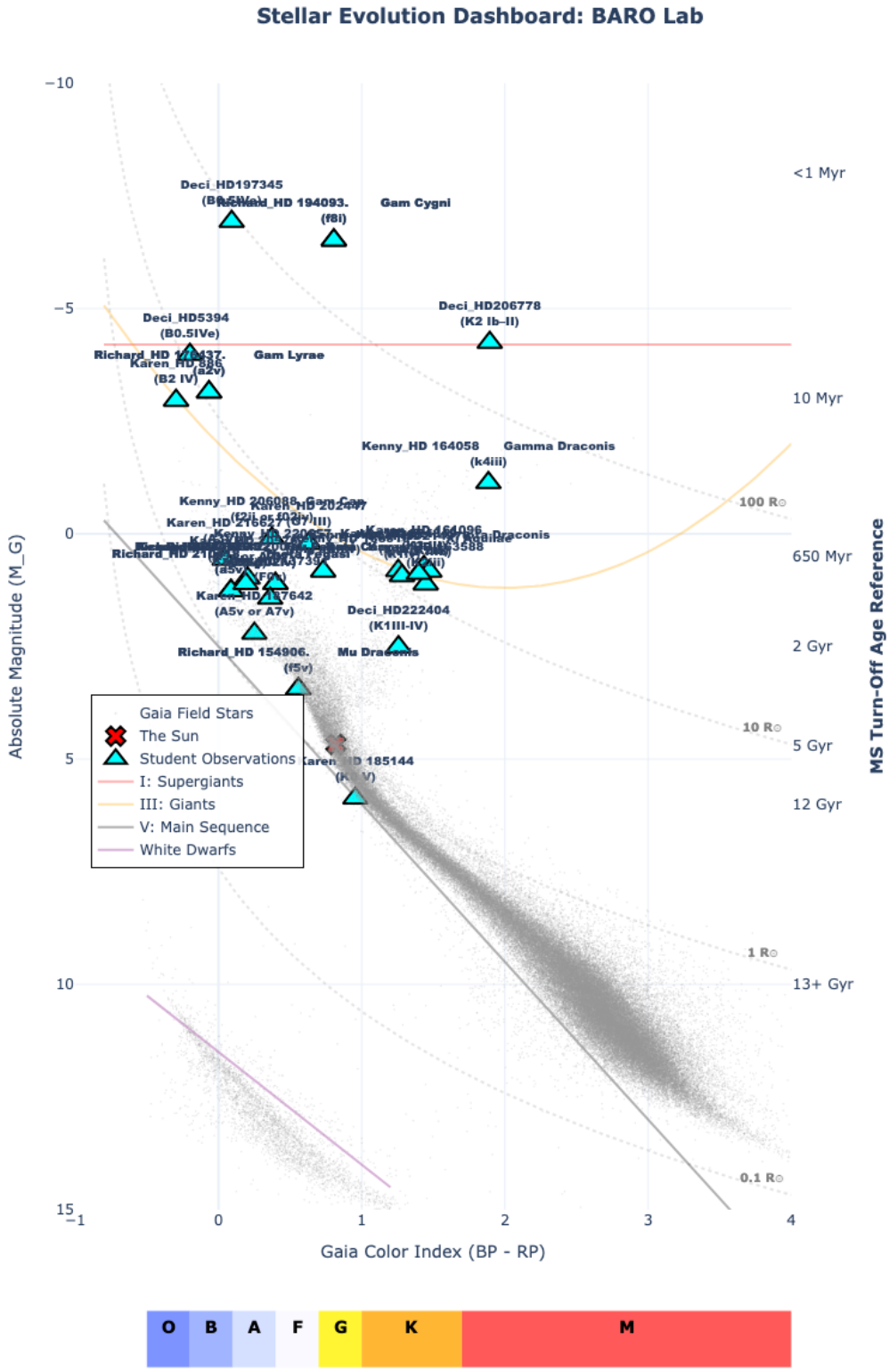

Stellar Evolution Dashboard: BARO Lab
Absolute Magnitude (M_G)
Gaia Color Index (BP - RP)
MS Turn-Off Age Reference
<1 Myr
10 Myr
650 Myr
2 Gyr
5 Gyr
12 Gyr
13+ Gyr
100 R☉
10 R☉
1 R☉
0.1 R☉
Gaia Field Stars
The Sun
Student Observations
I: Supergiants
III: Giants
V: Main Sequence
White Dwarfs
Deci_HD197345
Gam Cygni
Deci_HD206778
(K2 Ib–II)
Deci_HD5394
(B0.5IVe)
Gam Lyrae
(B2 IV)
Kenny_HD 164058
Gamma Draconis
(k4iii)
Karen_HD 217675
(A5v or A7v)
Deci_HD222404
(K1III-IV)
Mu Draconis
(f5v)

[7]:
```
# --- 1. DATA INPUT ---
print("--- BARO Spectrography Data Entry ---")
mode_in = input("Enter 'csv' or 'manual' [Default: csv]: ").lower().strip()
mode = mode_in if mode_in else "csv"

if mode == 'csv':
    path_in = input("Enter filename [Default: student_stars.csv]: ").strip()
    path = path_in if path_in else "student_stars.csv"
    df_students = pd.read_csv(path)
else:
    star_list = []
    while True:
        name = input("\nStar Name (or 'n' to plot): ")
        if name.lower() == 'n': break
        bv_raw = input(f"B-V for {name}: ").replace('−', '-').strip()
        bv = float(bv_raw) if bv_raw else 0.0
        mag_raw = input(f"Apparent Mag (G) for {name}: ").replace('−', '-').
 ↪strip()
        app_mag = float(mag_raw) if mag_raw else 5.0
        plx_raw = input(f"Parallax (mas) for {name}: ").replace('−', '-').
 ↪strip()
        parallax = float(plx_raw) if plx_raw else 10.0
        spec = input(f"Spectral Type for {name}: ")
        star_list.append({'star_name': name, 'b_v': bv, 'app_mag': app_mag,␣
 ↪'parallax': parallax, 'spec_type': spec})
    df_students = pd.DataFrame(star_list)

# --- 2. PHYSICS ENGINE ---
df_students['bp_rp'] = -0.0176 + 1.2156*df_students['b_v'] + 0.
 ↪0149*(df_students['b_v']**2) + 0.0058*(df_students['b_v']**3)
df_students['dist'] = 1000 / df_students['parallax']
df_students['abs_mag'] = df_students['app_mag'] - 5 * np.
 ↪log10(df_students['dist']) + 5
df_students['temp_k'] = 9000 / (df_students['bp_rp'] + 0.93)
lum_ratio = 10**((4.67 - df_students['abs_mag']) / 2.5)
df_students['radius_solar'] = np.sqrt(lum_ratio) * (5778 /␣
 ↪df_students['temp_k'])**2
df_students['hover_text'] = [f"<b>{r['star_name']}</b><br>Type:␣
 ↪{r['spec_type']}<br>Temp: {int(r['temp_k'])}K<br>Radius: {r['radius_solar']:.
 ↪2f}R⊙" for _, r in df_students.iterrows()]

# --- 3. BACKGROUND & PLOT SETUP ---
mask = (df_gaia['bp_rp'] >= -1) & (df_gaia['bp_rp'] <= 4) & (df_gaia['abs_mag']␣
 ↪<= 15) & (df_gaia['abs_mag'] >= -10)
df_bg = df_gaia[mask].sample(n=min(100000, len(df_gaia[mask])))
```

```
fig = make_subplots(specs=[[{"secondary_y": True}]])

# IMPORTANT: Define these lists BEFORE the loops
all_shapes = []
all_annotations = []

# Traces (Background, Sun, Students)
fig.add_trace(go.Scattergl(x=df_bg['bp_rp'], y=df_bg['abs_mag'],␣
 ↪mode='markers', name='Gaia Field Stars', marker=dict(color='rgba(150, 150,␣
 ↪150, 0.2)', size=1.5)), secondary_y=False)
fig.add_trace(go.Scatter(x=[0.82], y=[4.67], mode='markers', name='The Sun',␣
 ↪marker=dict(symbol='x', size=14, color='red', line=dict(width=2,␣
 ↪color='black')), hovertext="Sun (G2V)", hoverinfo='text'), secondary_y=False)
labels = [f"{n}<br>({s})" for n, s in zip(df_students['star_name'],␣
 ↪df_students['spec_type'])]
fig.add_trace(go.Scatter(x=df_students['bp_rp'], y=df_students['abs_mag'],␣
 ↪mode='markers+text', name='Student Observations', text=labels,␣
 ↪hovertext=df_students['hover_text'], textposition="top center",␣
 ↪textfont=dict(family="Arial Black", size=10),␣
 ↪marker=dict(symbol='triangle-up', size=18, color='cyan', line=dict(width=2,␣
 ↪color='black')), hoverinfo='text'), secondary_y=True)

# Evolutionary Curves
color_range = np.linspace(-0.8, 4.0, 100)
lum_curves = [{'n': 'I: Supergiants', 'y': -4.2 * np.ones_like(color_range),␣
 ↪'c': 'rgba(255, 0, 0, 0.3)'}, {'n': 'III: Giants', 'y': -0.8 * (color_range␣
 ↪- 2.0)**2 + 1.2, 'c': 'rgba(255, 165, 0, 0.3)'}, {'n': 'V: Main Sequence',␣
 ↪'y': 3.5 * color_range + 2.5, 'c': 'rgba(0, 0, 0, 0.3)'}, {'n': 'White␣
 ↪Dwarfs', 'x': np.linspace(-0.5, 1.2, 50), 'y': 2.5 * np.linspace(-0.5, 1.2,␣
 ↪50) + 11.5, 'c': 'rgba(128, 0, 128, 0.3)'}]
for c in lum_curves:
    fig.add_trace(go.Scatter(x=c.get('x', color_range), y=c['y'], mode='lines',␣
 ↪line=dict(color=c['c'], width=2), name=c['n']), secondary_y=False)

# Isoradii
for r in [0.1, 1, 10, 100]:
    iso_mg = 4.67 - 2.5 * np.log10((r**2) * ((9000/(color_range+0.93))/5778)**4)
    fig.add_trace(go.Scatter(x=color_range, y=iso_mg, mode='lines',␣
 ↪line=dict(color='rgba(0,0,0,0.1)', dash='dot'), showlegend=False),␣
 ↪secondary_y=False)
    all_annotations.append(dict(x=3.8, y=iso_mg[-10], text=f"<b>{r} R </b>",␣
 ↪showarrow=False, font=dict(size=10, color="gray")))

# Spectral Strip
```

```
spectral_data = [{'class':'O','s':-0.5,'e':-0.2,'c':'#5d78ff'},{'class':'B','s':
 ↪-0.2,'e':0.1,'c':'#89a1ff'},{'class':'A','s':0.1,'e':0.4,'c':
 ↪'#cad7ff'},{'class':'F','s':0.4,'e':0.7,'c':'#f8f7ff'},{'class':'G','s':0.
 ↪7,'e':1.0,'c':'#fff200'},{'class':'K','s':1.0,'e':1.7,'c':
 ↪'#ffa500'},{'class':'M','s':1.7,'e':4.0,'c':'#ff3030'}]
for s in spectral_data:
    all_shapes.append(dict(type="rect", xref="x", yref="paper", x0=s['s'],␣
 ↪x1=s['e'], y0=-0.14, y1=-0.09, fillcolor=s['c'], opacity=0.8, line_width=0))
    all_annotations.append(dict(x=(s['s']+s['e'])/2, y=-0.115, xref="x",␣
 ↪yref="paper", text=f"<b>{s['class']}</b>", showarrow=False,␣
 ↪font=dict(color="black", size=14)))

# --- 4. FINAL LAYOUT ---
fig.update_layout(
    template="plotly_white", title=dict(text="<b>Stellar Evolution Dashboard:␣
 ↪BARO Lab</b>", x=0.5),
    height=1200, width=800, margin=dict(b=160, r=120, t=100),
    xaxis=dict(title="Gaia Color Index (BP - RP)", range=[-1, 4]),
    yaxis=dict(title="Absolute Magnitude (M_G)", range=[15, -10],␣
 ↪zeroline=False),
    yaxis2=dict(title="<b>MS Turn-Off Age Reference</b>", side="right",␣
 ↪overlaying="y", tickvals=[-8, -3, 0.5, 2.5, 4.7, 6, 10], ticktext=['<1 Myr',␣
 ↪'10 Myr', '650 Myr', '2 Gyr', '5 Gyr', '12 Gyr', '13+ Gyr'], range=[15,␣
 ↪-10], showgrid=False, autorange=False),
    legend=dict(
        x=0.02,
        y=0.38, # Shifted down slightly to clear the White Dwarf gap
        bgcolor="rgba(255,255,255,0.9)",
        bordercolor="black",
        borderwidth=1
    ),
    shapes=all_shapes, annotations=all_annotations, hovermode='closest'
)
fig.show()
```

```
--- BARO Spectrography Data Entry ---
Enter 'csv' or 'manual' [Default: csv]:  csv
Enter filename [Default: student_stars.csv]:  combined_final_student_6.csv
```


Stellar Evolution Dashboard: BARO Lab
Absolute Magnitude (M_G)
Gaia Color Index (BP - RP)
MS Turn-Off Age Reference
<1 Myr
10 Myr
650 Myr
2 Gyr
5 Gyr
12 Gyr
13+ Gyr
100 R☉
10 R☉
1 R☉
0.1 R☉
Deci_HD197345
Gam Cygni
Deci_HD206778
(K2 Ib–II)
Deci_HD5394
(B0.5IVe)
Gam Lyrae
(B2 IV)
Kenny_HD 164058
Gamma Draconis
(k4iii)
Karen_HD 217675
(A5v or A7v)
Deci_HD222404
(K1III-IV)
Richard_HD 154906.
(f5v)
Mu Draconis
Gaia Field Stars
The Sun
Student Observations
I: Supergiants
III: Giants
V: Main Sequence
White Dwarfs
O
B
A
F
G
K
M

[ ]:

[ ]:

[ ]:

[ ]:

[ ]: