Storytelling with matplotlib

A David Vizgan Production

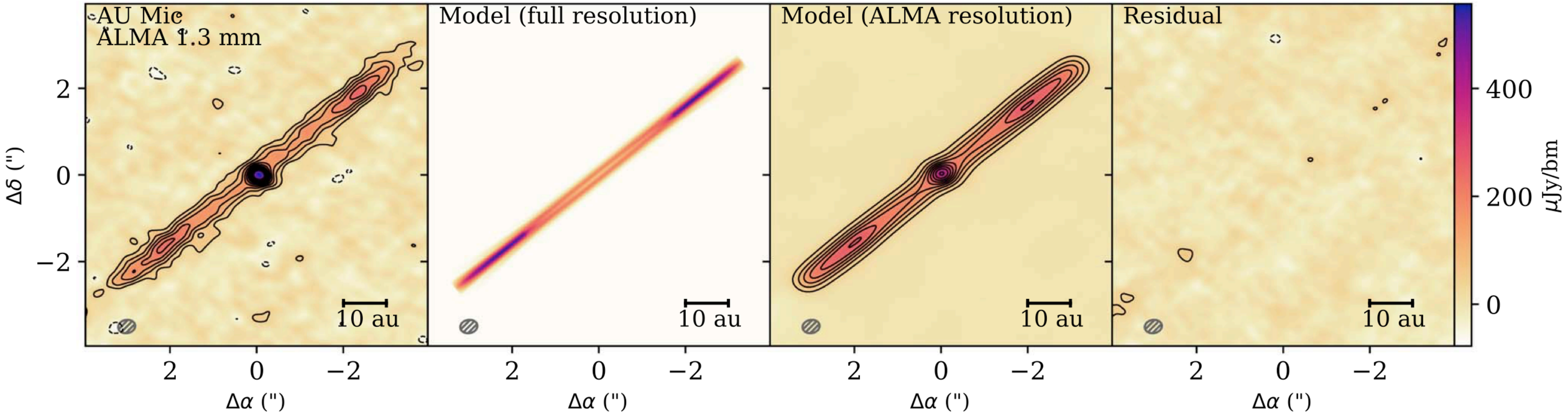
22 March 2024

Storytelling with `matplotlib`

aka: David rants about graphic design for 30+ mins

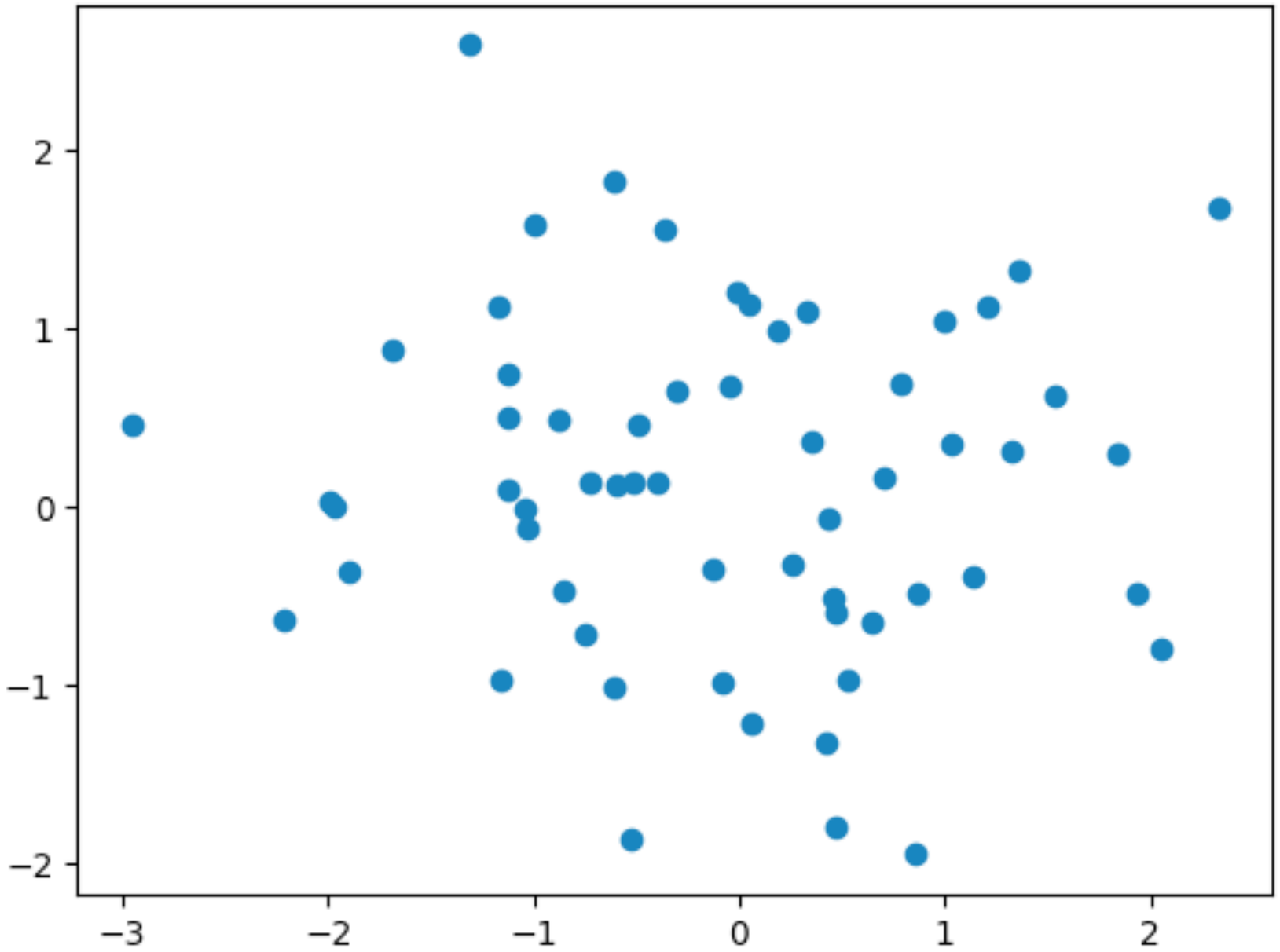
1. Why do we need nice plots?
2. Spatial data visualization
3. Spectral data visualization
4. Scatter plots and misc. data visualization

Why do we need nice plots?

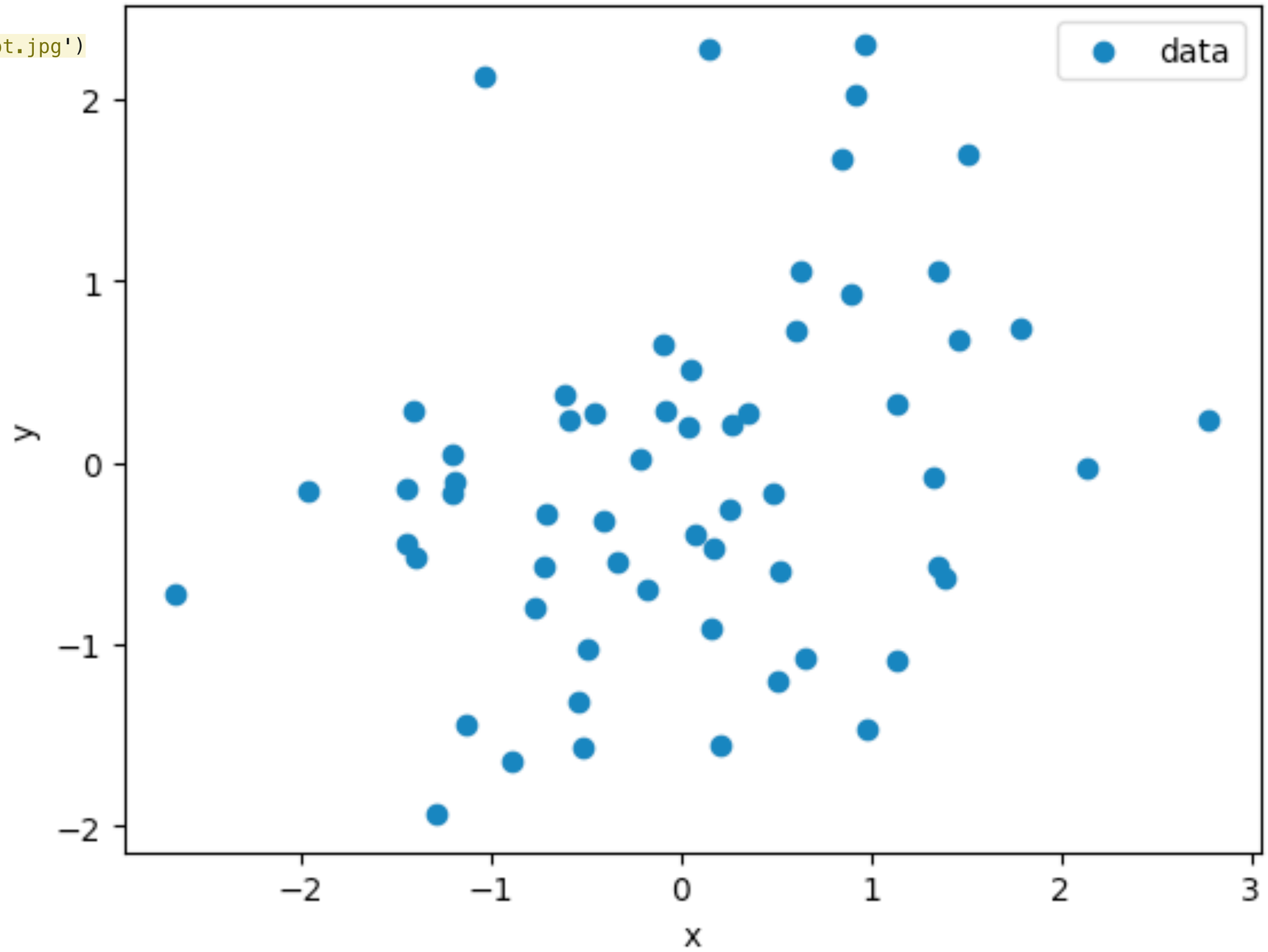


Vizgan *et al*. (2022b)

```
plt.scatter(np.random.randn(60), np.random.randn(60))
plt.savefig('plot.jpg')
plt.show()
```



```
plt.scatter(np.random.randn(60), np.random.randn(60), label='data')
plt.xlabel('x')
plt.ylabel('y')
plt.legend()
plt.savefig('plot.jpg')
plt.show()
```



Style sheets!!!

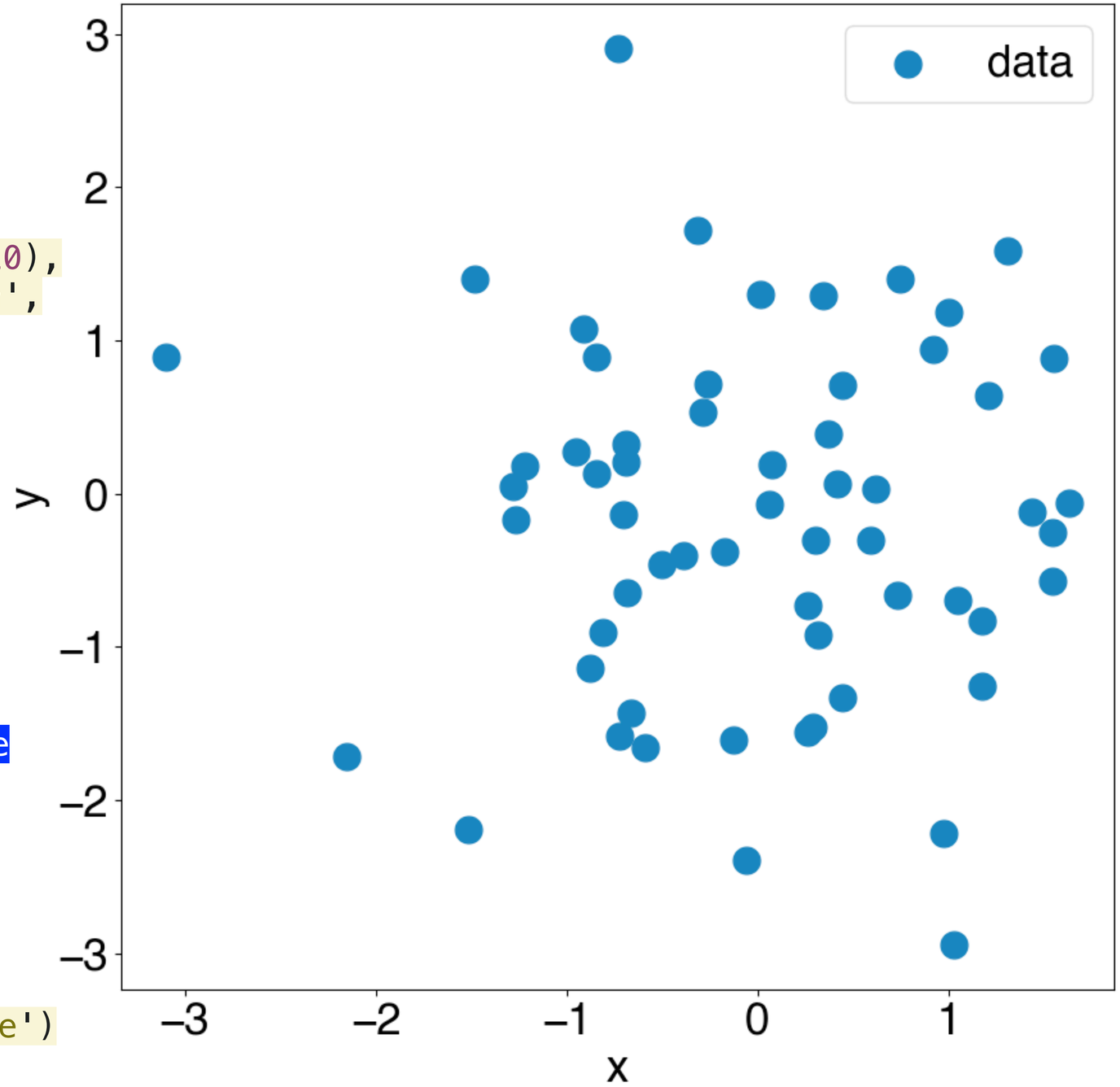
```
plt.rcParams.update({'figure.figsize': (10, 10),
                     'font.family': 'Helvetica',
                     'font.size': 25,
                     'lines.markersize': 15
})
```

OR

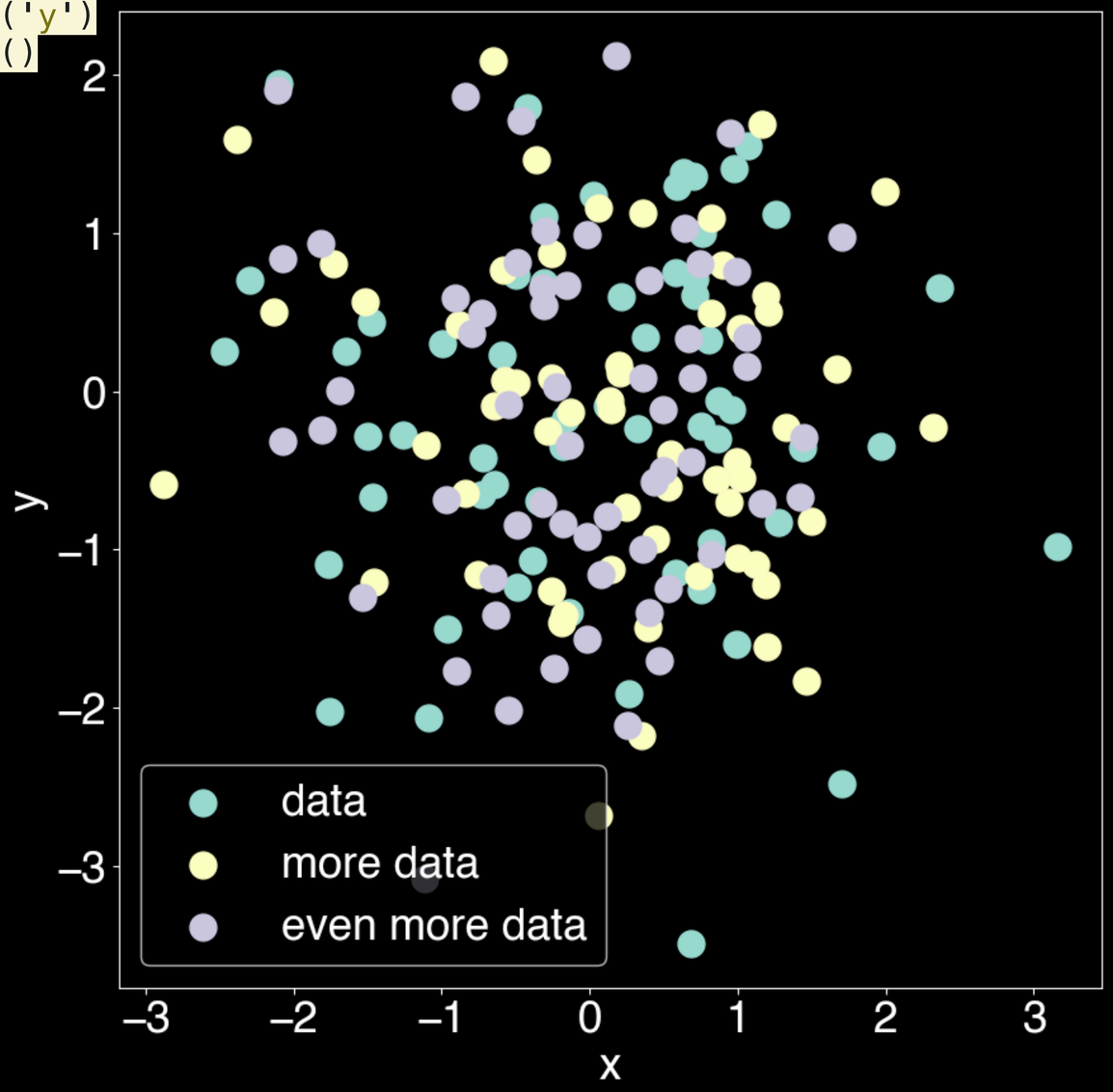
```
(base) ☁ ~  vim .images/presentation.mplstyle
```

```
figure.figsize : (10,10)
font.family : 'Helvetica'
font.size : 25
lines.markersize : 15
```

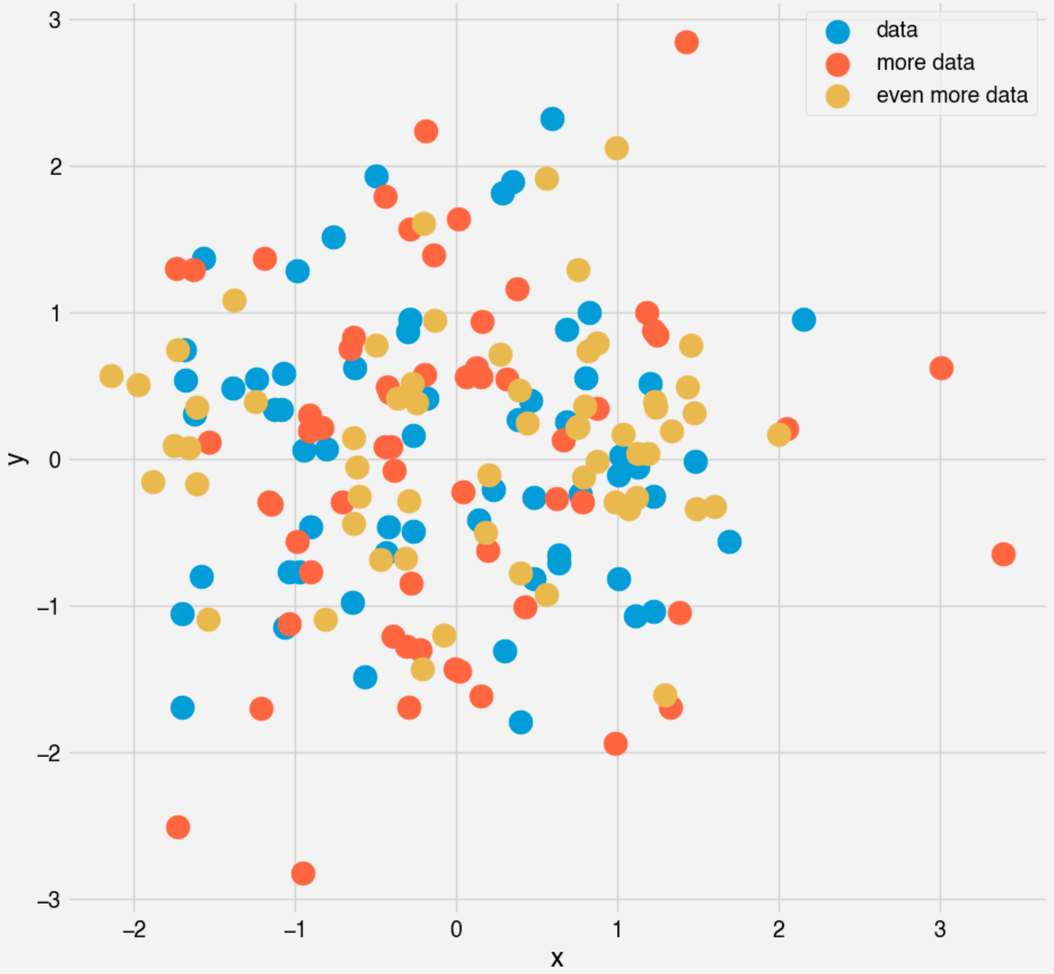
```
plt.style.use('./images/presentation.mplstyle')
```



```
plt.style.use(['dark_background'])
plt.scatter(np.random.randn(60), np.random.randn(60), label='data')
plt.scatter(np.random.randn(60), np.random.randn(60), label='more data')
plt.scatter(np.random.randn(60), np.random.randn(60), label='even more data')
plt.xlabel('x')
plt.ylabel('y')
plt.legend()
plt.show()
```



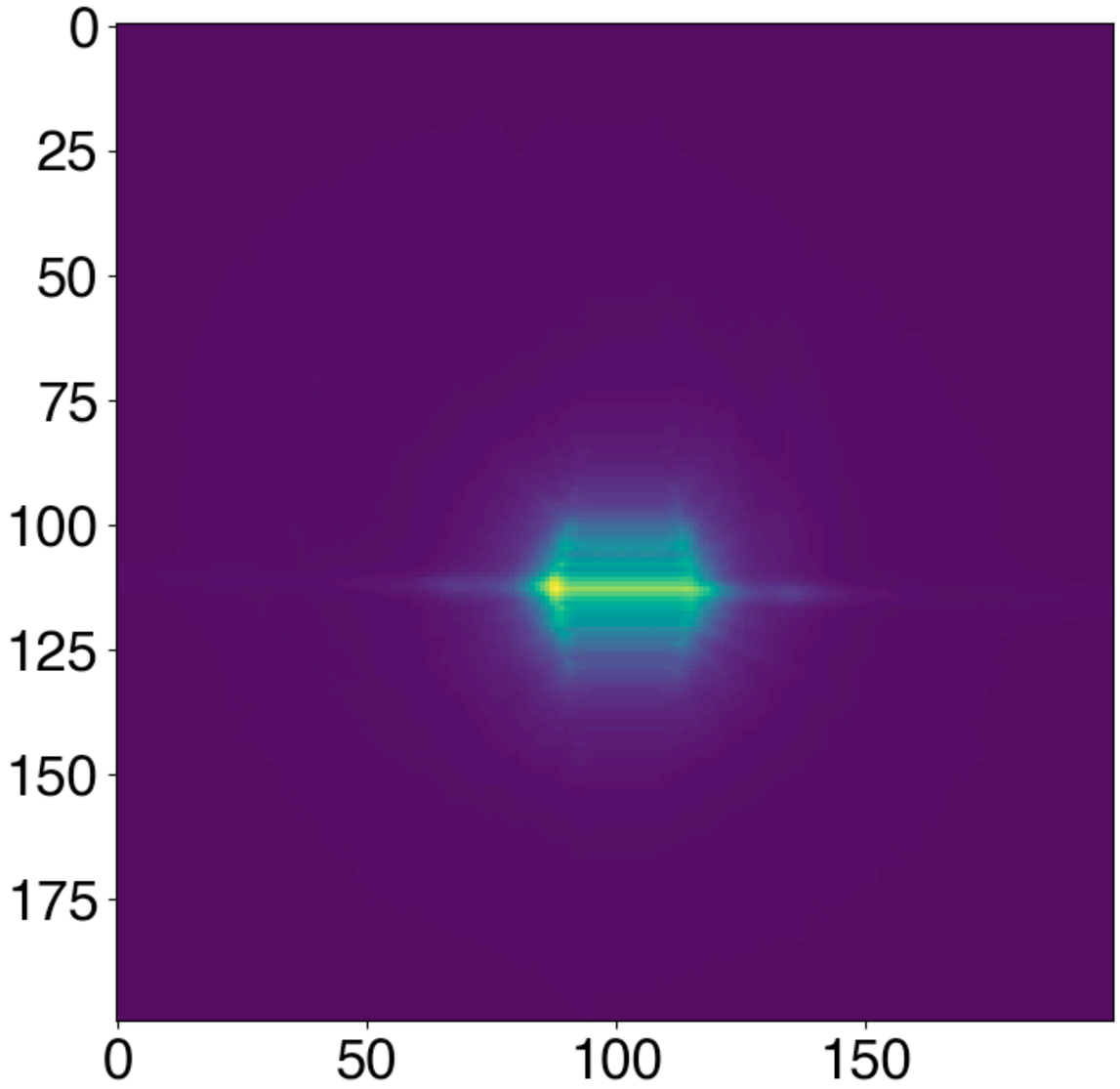
```
plt.style.use(['fivethirtyeight'])
plt.scatter(np.random.randn(60), np.random.randn(60), label='data')
plt.scatter(np.random.randn(60), np.random.randn(60), label='more data')
plt.scatter(np.random.randn(60), np.random.randn(60), label='even more data')
plt.xlabel('x')
plt.ylabel('y')
plt.legend()
plt.show()
```



Spatial data visualization

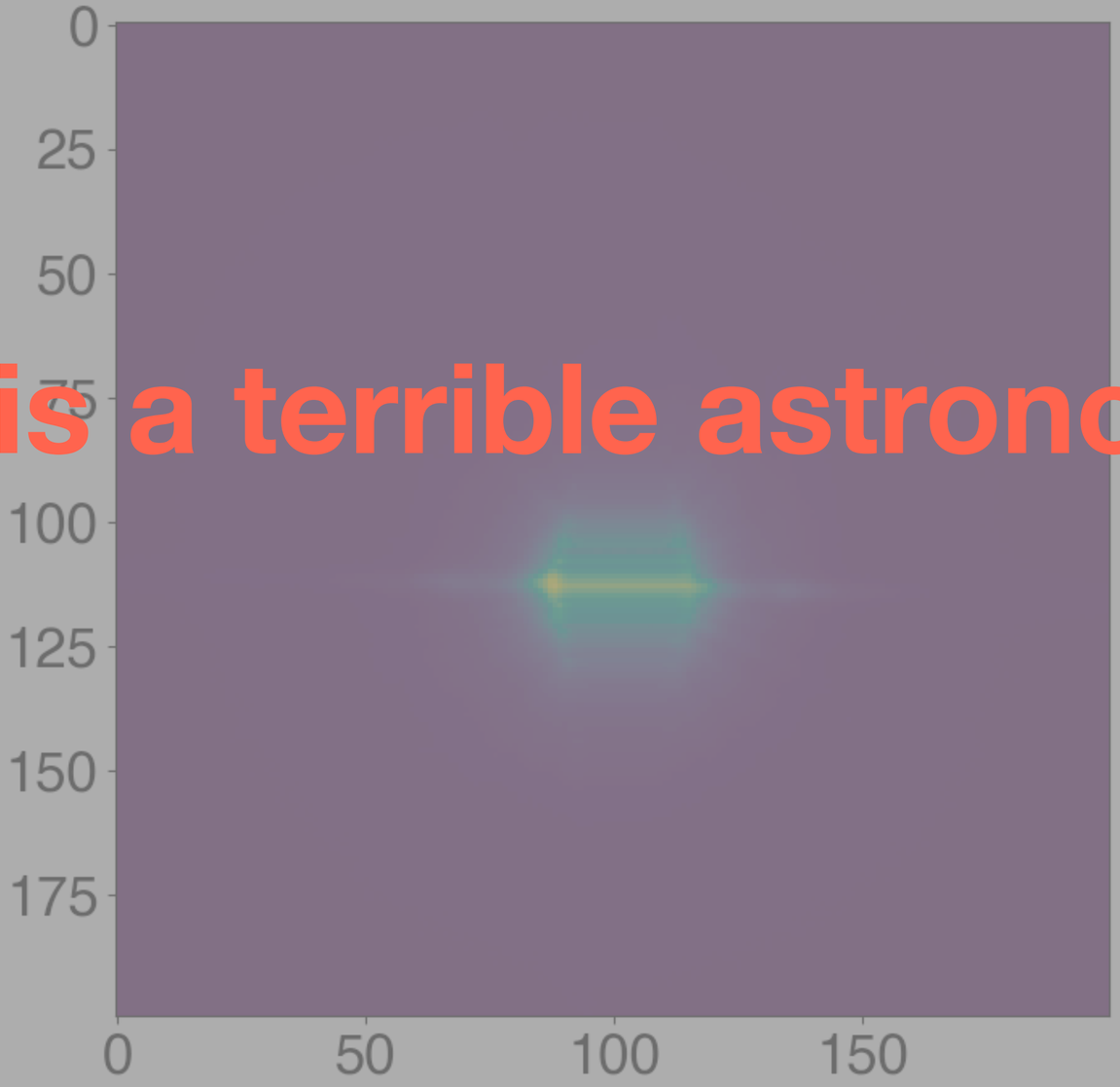
```
u = fits.getdata('51 peg data/frame-u-008108-3-0144.fits')[304:504, 14:214]
g = fits.getdata('51 peg data/frame-g-008108-3-0144.fits')[304:504, 14:214]
r = fits.getdata('51 peg data/frame-r-008108-3-0144.fits')[304:504, 14:214]
i = fits.getdata('51 peg data/frame-i-008108-3-0144.fits')[304:504, 14:214]
z = fits.getdata('51 peg data/frame-z-008108-3-0144.fits')[304:504, 14:214]
```

```
plt.imshow(g)
```



```
u = fits.getdata('51 peg data/frame-u-008108-3-0144.fits')[304:504, 14:214]
g = fits.getdata('51 peg data/frame-g-008108-3-0144.fits')[304:504, 14:214]
r = fits.getdata('51 peg data/frame-r-008108-3-0144.fits')[304:504, 14:214]
i = fits.getdata('51 peg data/frame-i-008108-3-0144.fits')[304:504, 14:214]
z = fits.getdata('51 peg data/frame-z-008108-3-0144.fits')[304:504, 14:214]
```

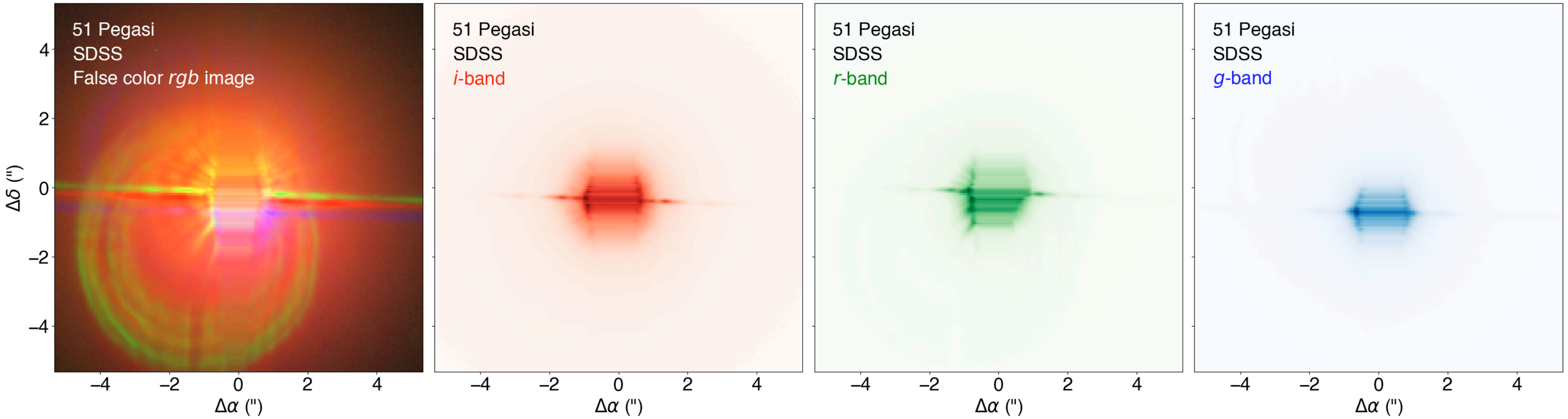
```
plt.imshow(g)
```



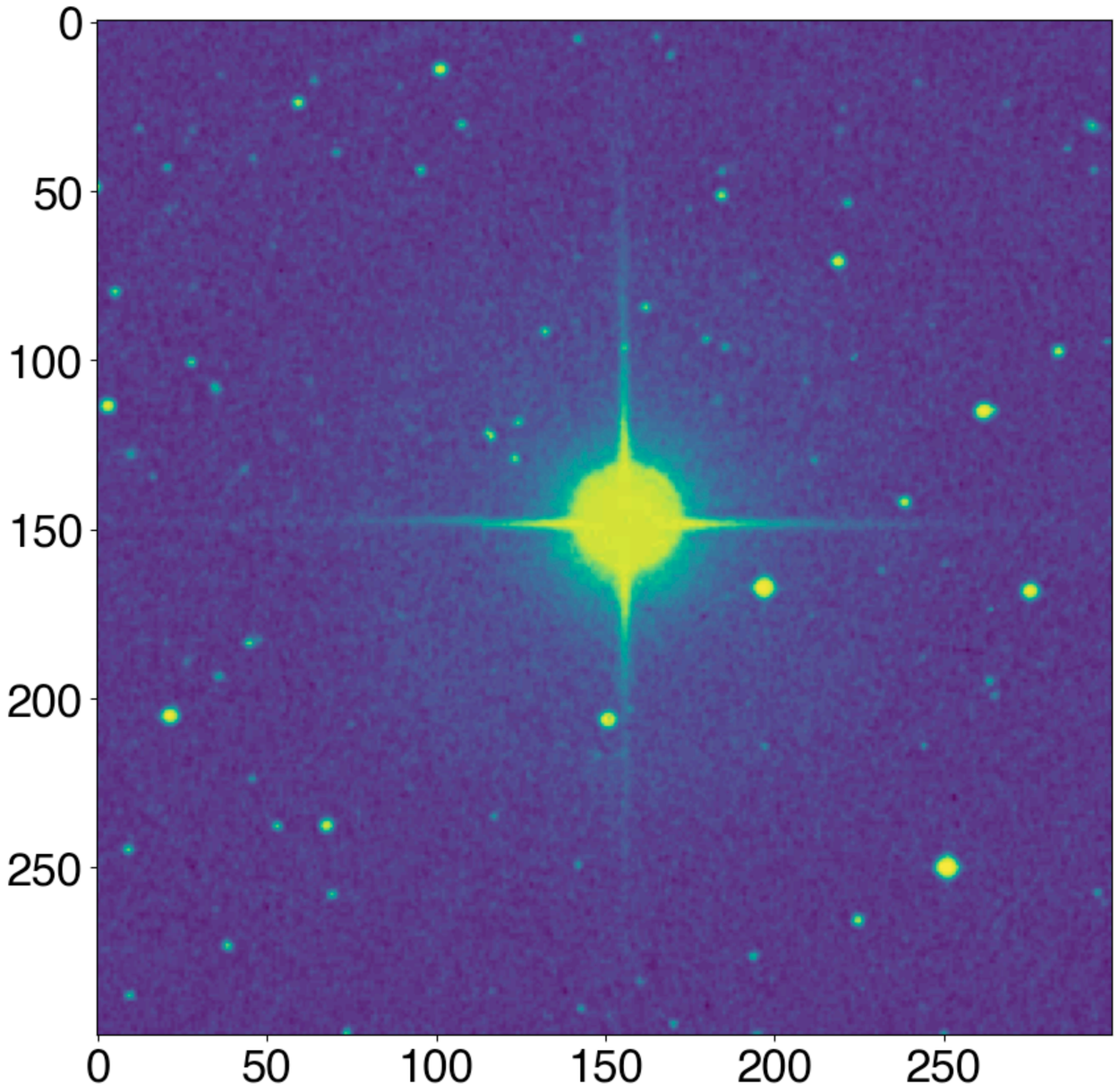
Why is this a terrible astronomy plot?

```
fig, axs = plt.subplots(1, 4, figsize=[30, 8], sharey=True)

pixel_scale = 1.48e-5 * 200 * 3600
extent = [-pixel_scale/2, pixel_scale/2, -pixel_scale/2, pixel_scale/2]

axs[0].imshow(rgb, extent=extent)
axs[0].set_xlabel(r'$\Delta \alpha$ (")')
axs[0].set_ylabel(r'$\Delta \delta$ (")')
axs[0].text(-4.8, 4.4, '51 Pegasi', color='white', size=25)
axs[0].text(-4.8, 3.7, 'SDSS', color='white', size=25)
axs[0].text(-4.8, 3.0, r'False color $rgb$ image', color='white', size=25)

axs[1].imshow(i, cmap='Reds', extent=extent)
axs[1].set_xlabel(r'$\Delta \alpha$ (")')
axs[1].text(-4.8, 4.4, '51 Pegasi', color='black', size=25)
axs[1].text(-4.8, 3.7, 'SDSS', color='black', size=25)
axs[1].text(-4.8, 3.0, r'$i$-band', color='red', size=25)

axs[2].imshow(r, cmap='Greens', extent=extent)
axs[2].set_xlabel(r'$\Delta \alpha$ (")')
axs[2].text(-4.8, 4.4, '51 Pegasi', color='black', size=25)
axs[2].text(-4.8, 3.7, 'SDSS', color='black', size=25)
axs[2].text(-4.8, 3.0, r'$r$-band', color='green', size=25)

axs[3].imshow(g, cmap='Blues', extent=extent)
axs[3].set_xlabel(r'$\Delta \alpha$ (")')
axs[3].text(-4.8, 4.4, '51 Pegasi', color='black', size=25)
axs[3].text(-4.8, 3.7, 'SDSS', color='black', size=25)
axs[3].text(-4.8, 3.0, r'$g$-band', color='blue', size=25)

plt.tight_layout(w_pad=0.5)
plt.show()
```

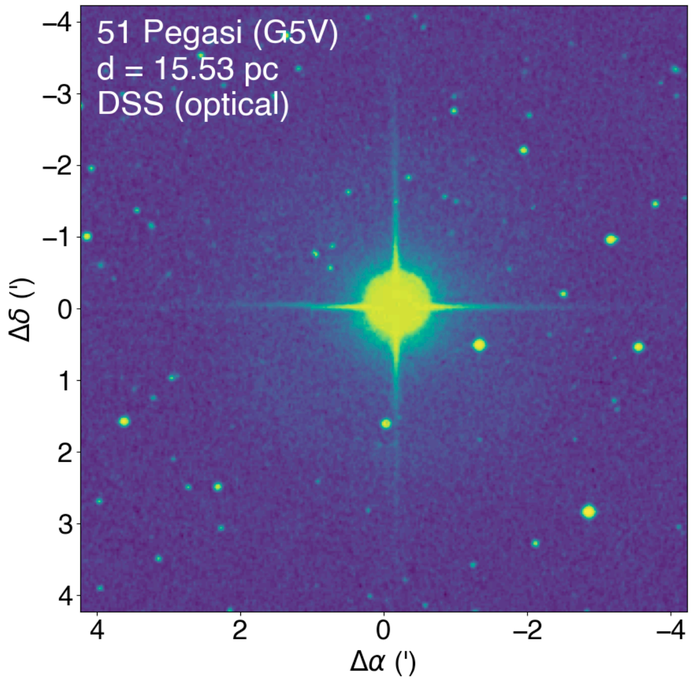
```
from astropy.visualization import make_lupton_rgb
rgb = make_lupton_rgb(i, r, g, filename="51 peg data/51peg_SDSS.jpeg")
```



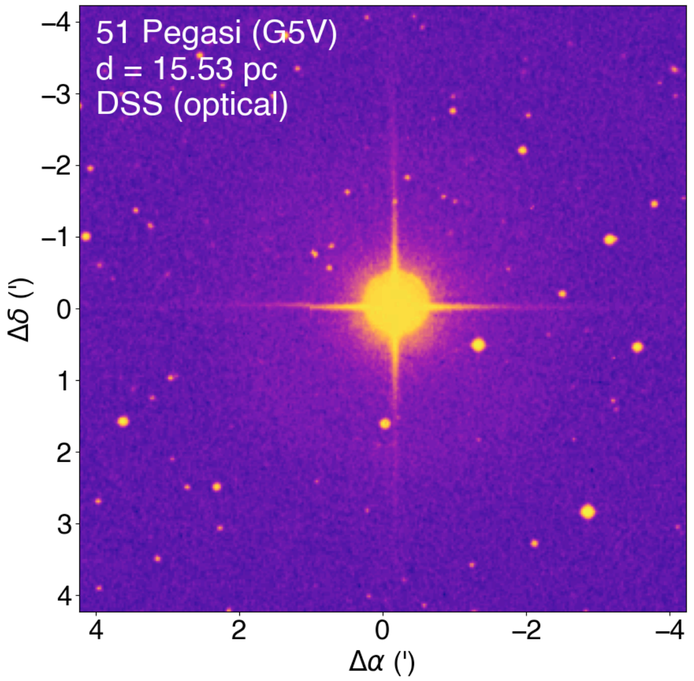
```
data = fits.getdata('51 peg data/51peg_DSS.fits')
plt.imshow(data)
```



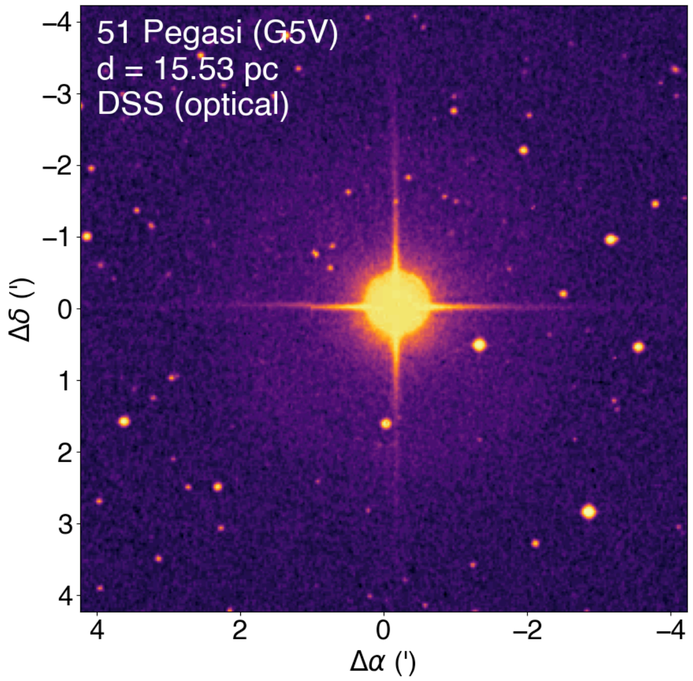
```
from matplotlib import colormaps
list(colormaps)
```



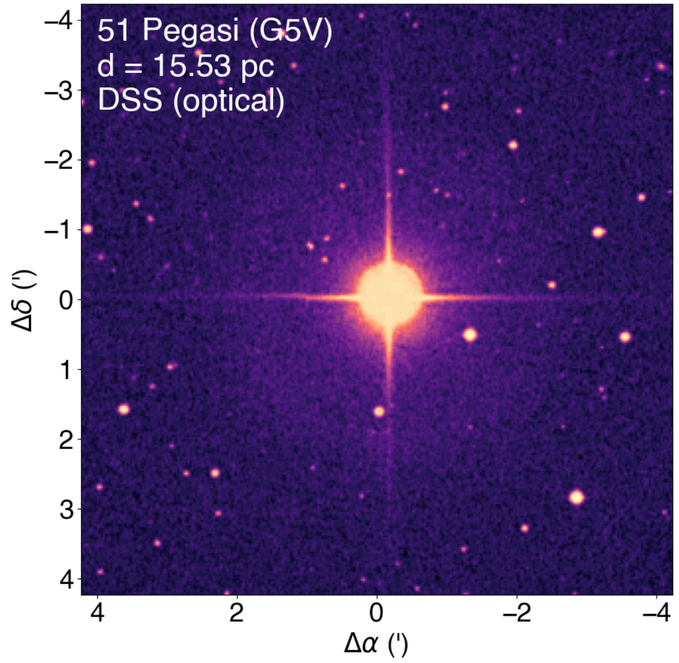
“viridis”



“plasma”

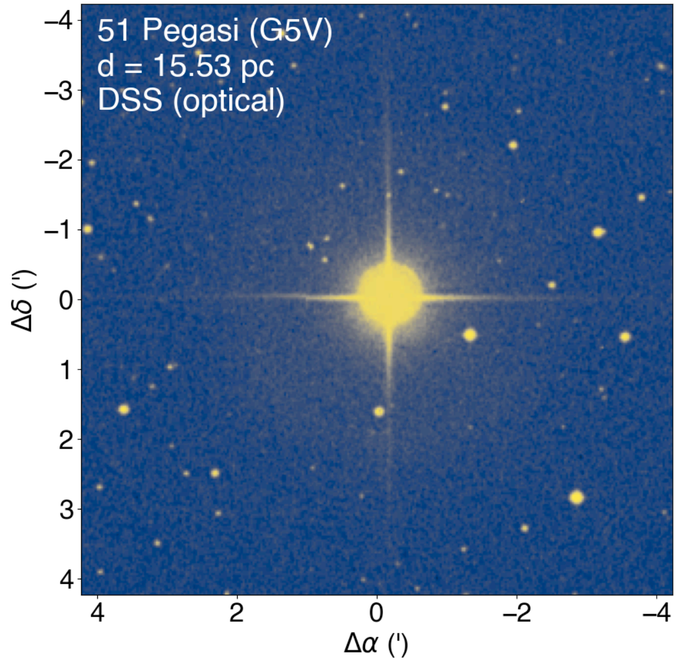


“inferno”



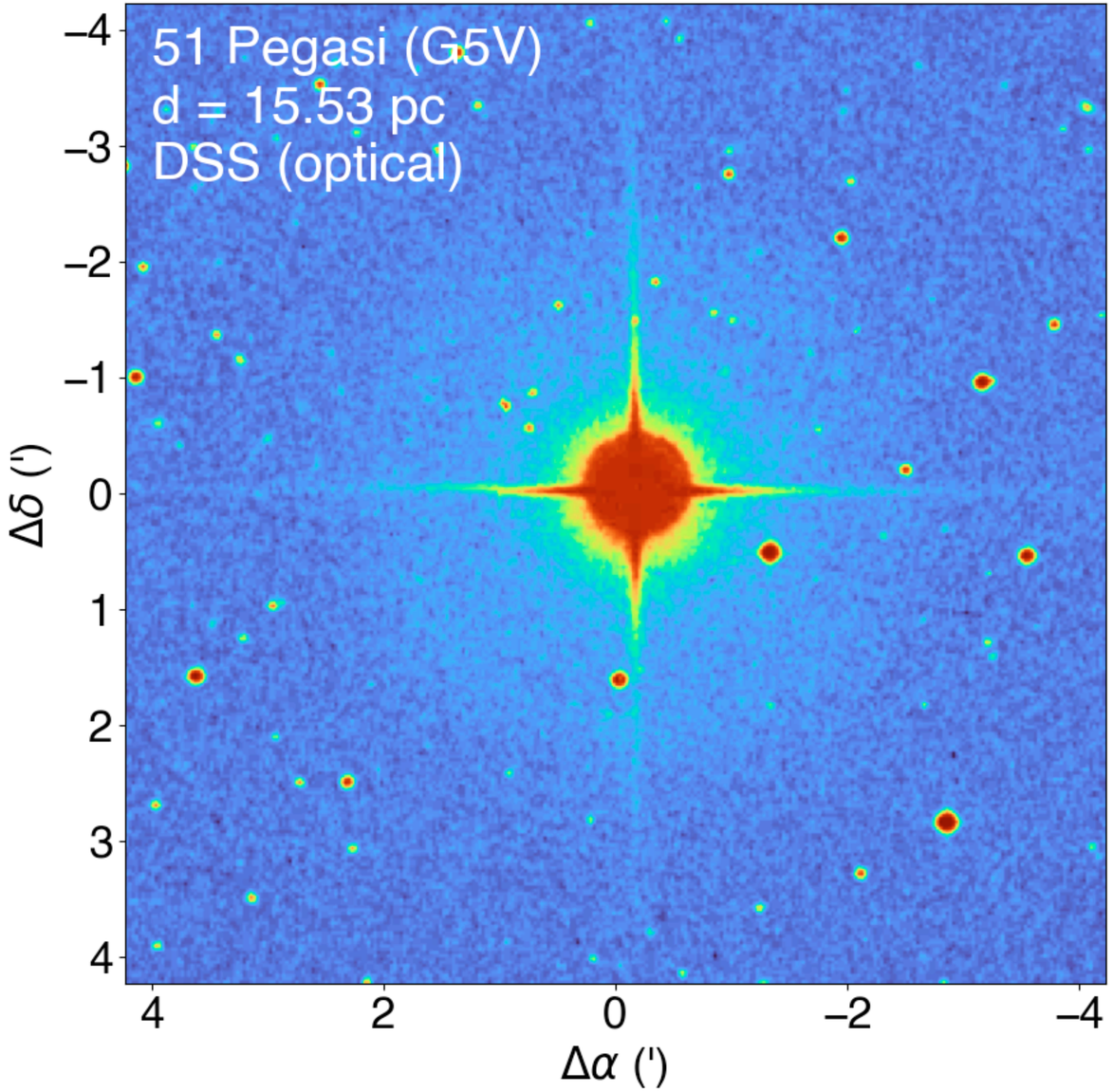
“magma”

```
plt.figure()
pixel_scale = -0.00047 * 300 * 60
extent = [-pixel_scale/2, pixel_scale/2, -pixel_scale/2, pixel_scale/2]

plt.imshow(data, extent=extent, cmap='cividis')
plt.text(4, -3.7, '51 Pegasi (G5V)', color='white', size=30)
plt.text(4, -3.2, 'd = 15.53 pc', color='white', size=30)
plt.text(4, -2.7, 'DSS (optical)', color='white', size=30)
plt.xlabel(r"$\Delta \alpha$ (') ")
plt.ylabel(r"$\Delta \delta$ (') ")
plt.show()
```

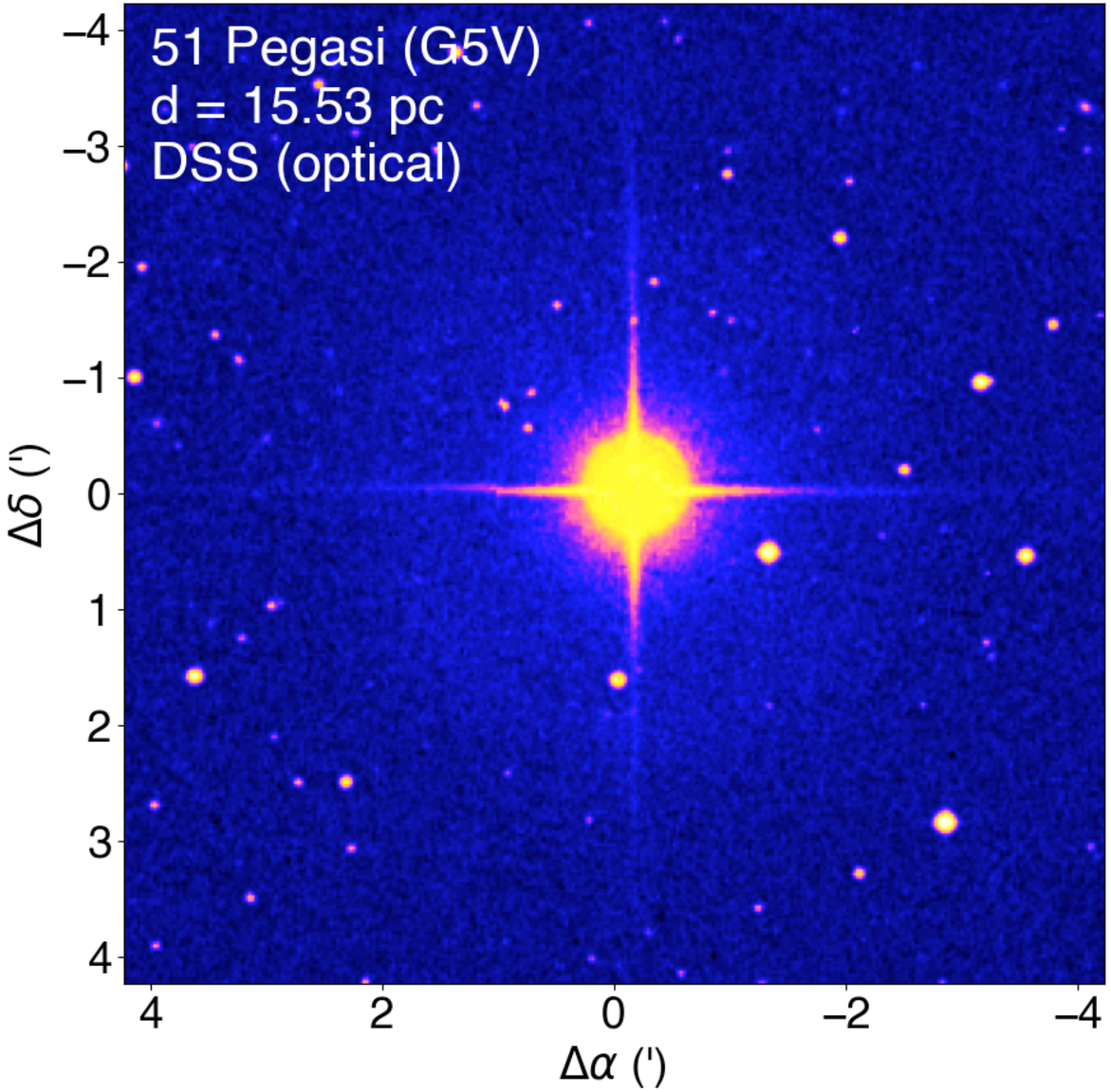


“cividis”

Many other colormaps though!



“turbo”



“gnuplot2”

Colorcet

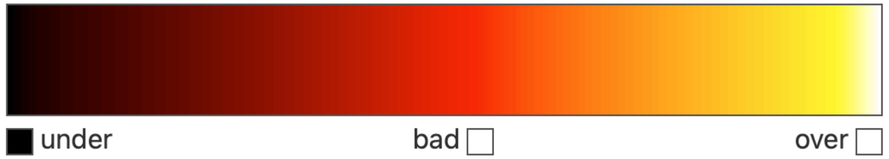
Using `colorcet` via `matplotlib.cm.get_cmap`

The `colorcet` colormaps are all available through `matplotlip.cm.get_cmap` by prepending `cet_` to the colormap name.

```
from matplotlib.cm import get_cmap

get_cmap("cet_fire")
```

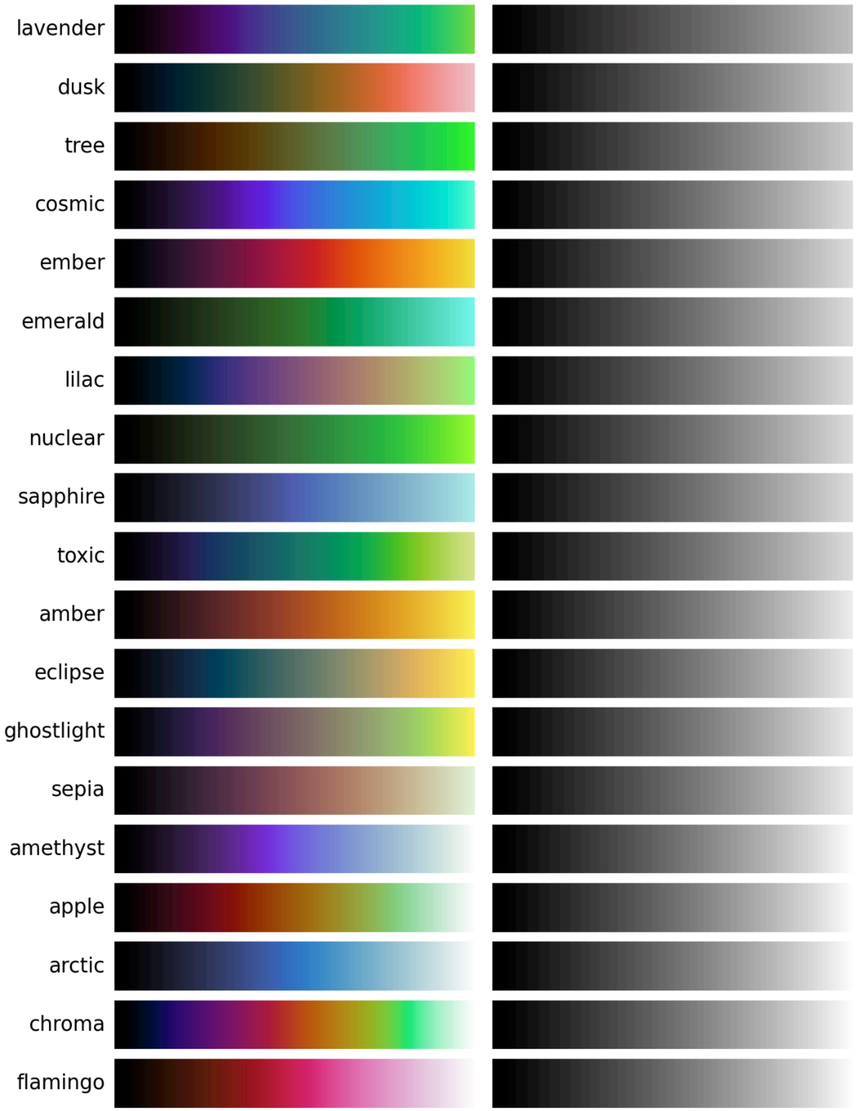
```
/tmp/ipykernel_2445/1693774644.py:3: MatplotlibDeprecationWarning: The get_cmap function was deprecated in Matplotlib
  get_cmap("cet_fire")
```

cet_fire



Using `colorcet` to visualize custom colormaps using `swatch`

```
color_list = ['#000000', '#380000', '#560000', '#760100', '#980300', '#bb0600', '#df0d00', '#f93500', '#fe6800', '#ff9
swatch(name='custom', cmap=color_list)
```



CMasher: Scientific colormaps for making accessible, informative and *cmashing* plots

This is the documentation for the *CMasher* package, a collection of scientific colormaps and utility functions for making accessible, informative and *cmashing* plots in Python. It is written in pure Python 3 and publicly available on GitHub. *CMasher* has also been published in the Journal of Open Source Software.

The documentation of *CMasher* is spread out over several sections:

User Documentation


```
> pip install cmasher
```

Palettable

Color palettes for Python

Palettable (formerly brewer2mpl) is a library of color palettes for Python. It's written in pure Python with no dependencies, but it can supply color maps for matplotlib. You can use Palettable to customize matplotlib plots or supply colors for a web application.

Palettable has color palettes from:

- CartoColors
- cmocean
- Colorbrewer2
- Cubehelix
- Light & Bartlein
- matplotlib
- MyCarta
- Scientific
- Tableau
- The Wes Anderson Palettes blog

Documentation

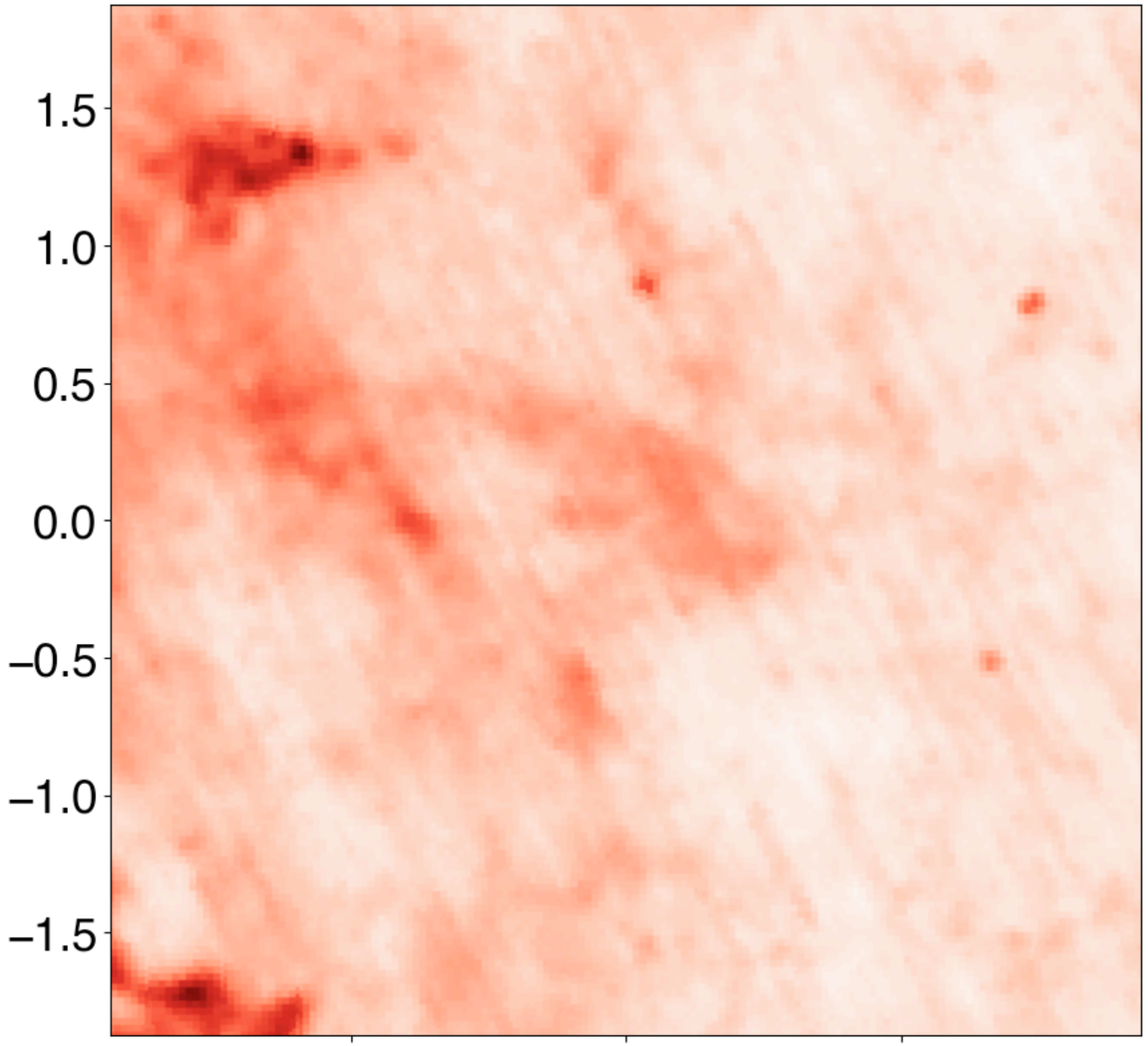
Installation

Palettable is available on PyPI for installation via pip: `pip install palettable`. Palettable is compatible with Python 2.6, 2.7, and Python 3.

Finding Palettes

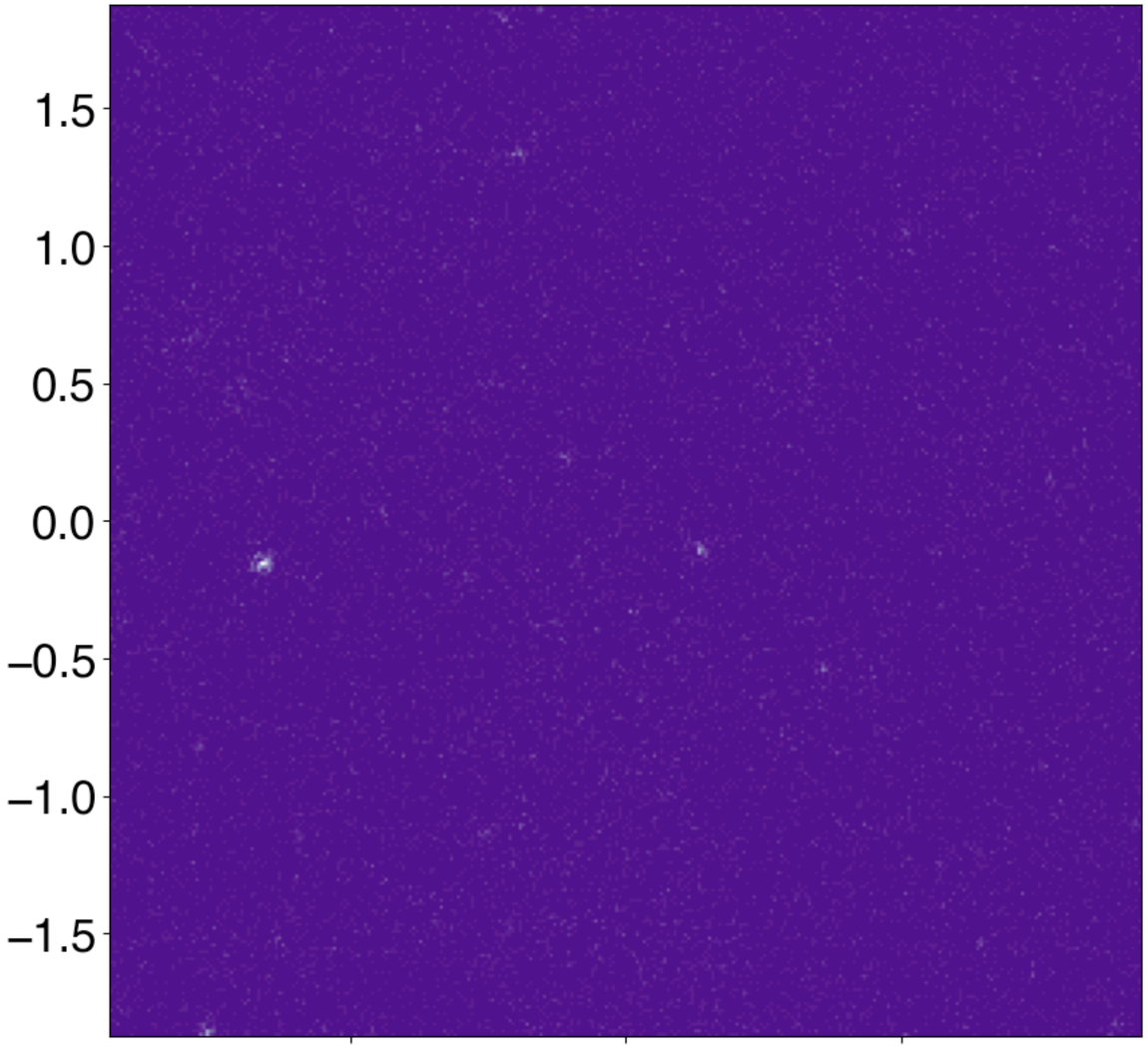
Palettes are pre-built and loaded at import time. They have a naming convention of `<Name>_<number of colors>`. For example, the Colorbrewer2 palette Dark2 with seven colors is named `Dark2_7`. Every palette also has a reversed version with the same name plus the suffix `_r` (e.g. `Dark2_7_r`).

```
iras = fits.getdata('51 peg data/51peg_iras.fits')
extent = [-3.75/2, 3.75/2, -3.75/2, 3.75/2]
plt.imshow(iras, cmap='Reds', alpha=1, extent=extent)
plt.show()
```



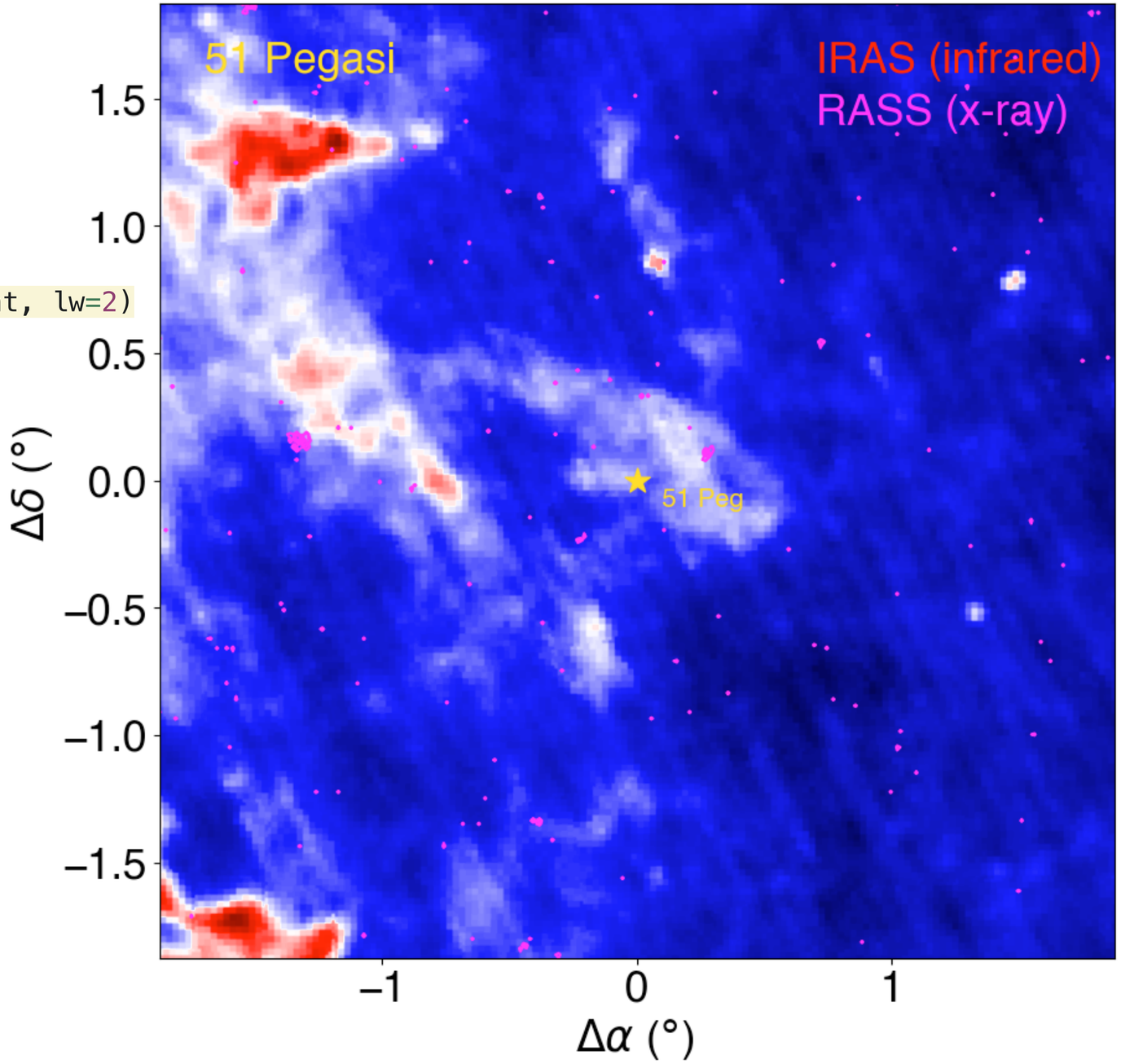
infrared

```
rass = fits.getdata('51 peg data/51peg_rass.fits')
extent = [-3.75/2, 3.75/2, -3.75/2, 3.75/2]
plt.imshow(rass, cmap='Purples_r', alpha=1, extent=extent)
plt.show()
```



x-ray

```
plt.contour(rass, colors='magenta', mincnt=1,
            levels=[2,3,5,8,11,14], alpha=1, extent=extent, lw=2)
plt.imshow(iras, cmap='seismic', alpha=1, extent=extent)

plt.text(0.1, -0.1, '51 Peg', color='gold', size=15)
plt.scatter(0,0, marker='*', color='gold')

plt.text(-1.7, 1.6, r'51 Pegasi', color='gold')
plt.text(0.7, 1.6, 'IRAS (infrared)', color='red')
plt.text(0.7, 1.4, 'RASS (x-ray)', color='magenta')

plt.xlabel(r'$\Delta \alpha$ ($\degree$) ')
plt.ylabel(r'$\Delta \delta$ ($\degree$) ')

plt.show()
```




black
dimgray
dimgrey
gray
grey
darkgray
darkgrey
silver
lightgray
lightgrey
gainsboro
whitesmoke
white
snow
rosybrown
lightcoral
indianred
brown
firebrick
maroon
darkred
red
mistyrose
salmon
tomato
darksalmon
coral
orangered
lightsalmon
sienna
seashell
chocolate
saddlebrown
sandybrown
peachpuff
peru
linen
bisque
darkorange
burlywood
antiquewhite
tan
navajowhite
blanchedalmond
papayawhip
moccasin
orange
wheat
oldlace
floralwhite
darkgoldenrod
goldenrod
cornsilk
gold
lemonchiffon
khaki
palegoldenrod
darkkhaki
ivory
beige
lightyellow
lightgoldenrodyellow
olive
yellow
olivedrab
yellowgreen
darkolivegreen
greenyellow
chartreuse
lawngreen
honeydew
darkseagreen
palegreen
lightgreen
forestgreen
limegreen
darkgreen
green
lime
seagreen
mediumseagreen
springgreen
mintcream
mediumspringgreen
mediumaquamarine
aquamarine
turquoise
lightseagreen
mediumturquoise
azure
lightcyan
paleturquoise
darkslategray
darkslategrey
teal
darkcyan
aqua
cyan
darkturquoise
cadetblue
powderblue
lightblue
deepskyblue
skyblue
lightskyblue
steelblue
aliceblue
dodgerblue
lightslategray
lightslategrey
slategray
slategrey
lightsteelblue
cornflowerblue
royalblue
ghostwhite
lavender
midnightblue
navy
darkblue
mediumblue
blue
slateblue
darkslateblue
mediumslateblue
mediumpurple
rebeccapurple
blueviolet
indigo
darkorchid
darkviolet
mediumorchid
thistle
plum
violet
purple
darkmagenta
fuchsia
magenta
orchid
mediumvioletred
deeppink
hotpink
lavenderblush
palevioletred
crimson
pink
lightpink

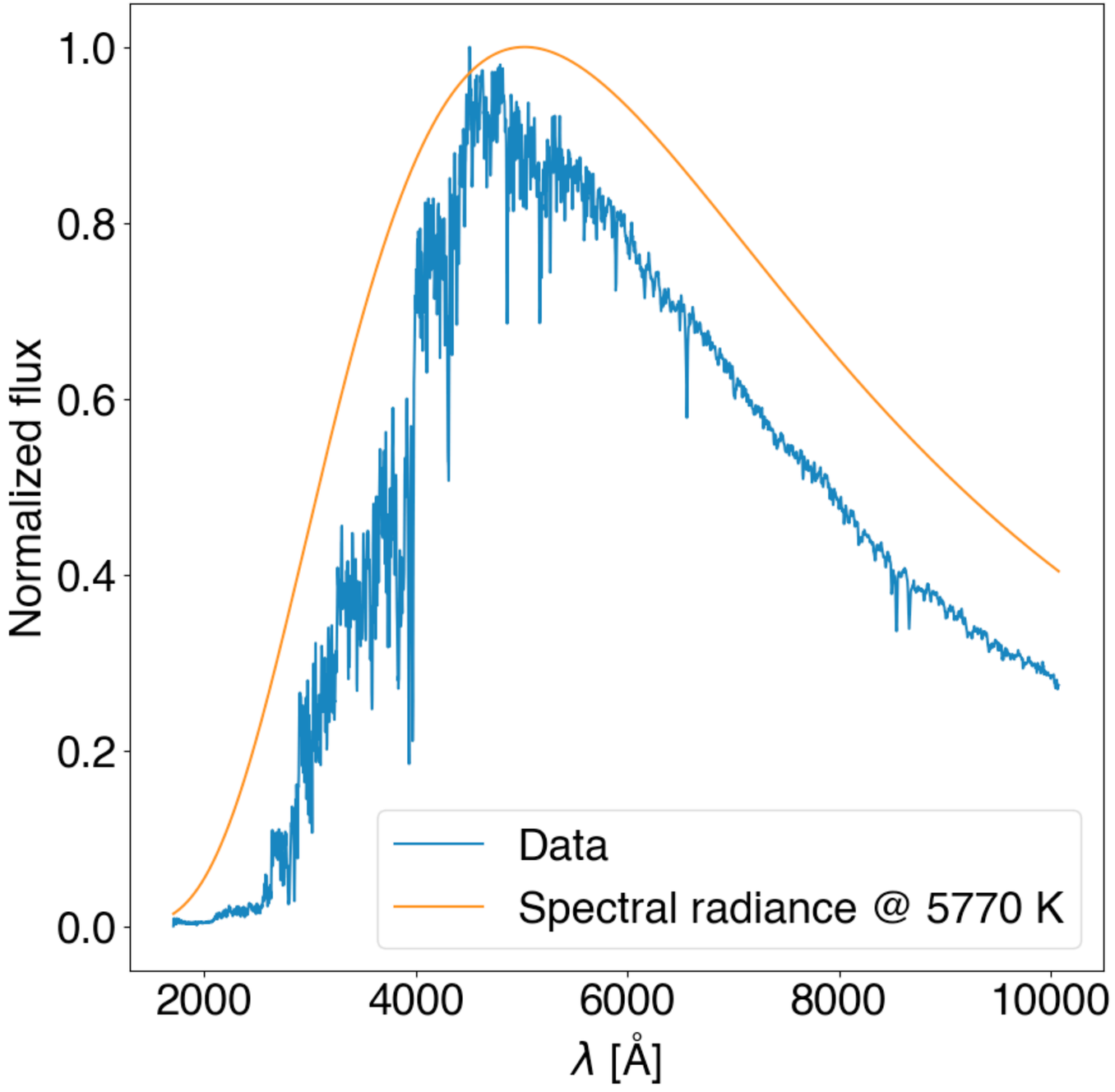
XKCD Colors

Matplotlib supports colors from the xkcd color survey, e.g. `"xkcd:sky blue"`. Since this contains almost 1000 colors, a figure of this would be very large and is thus omitted here. You can use the following code to generate the overview yourself

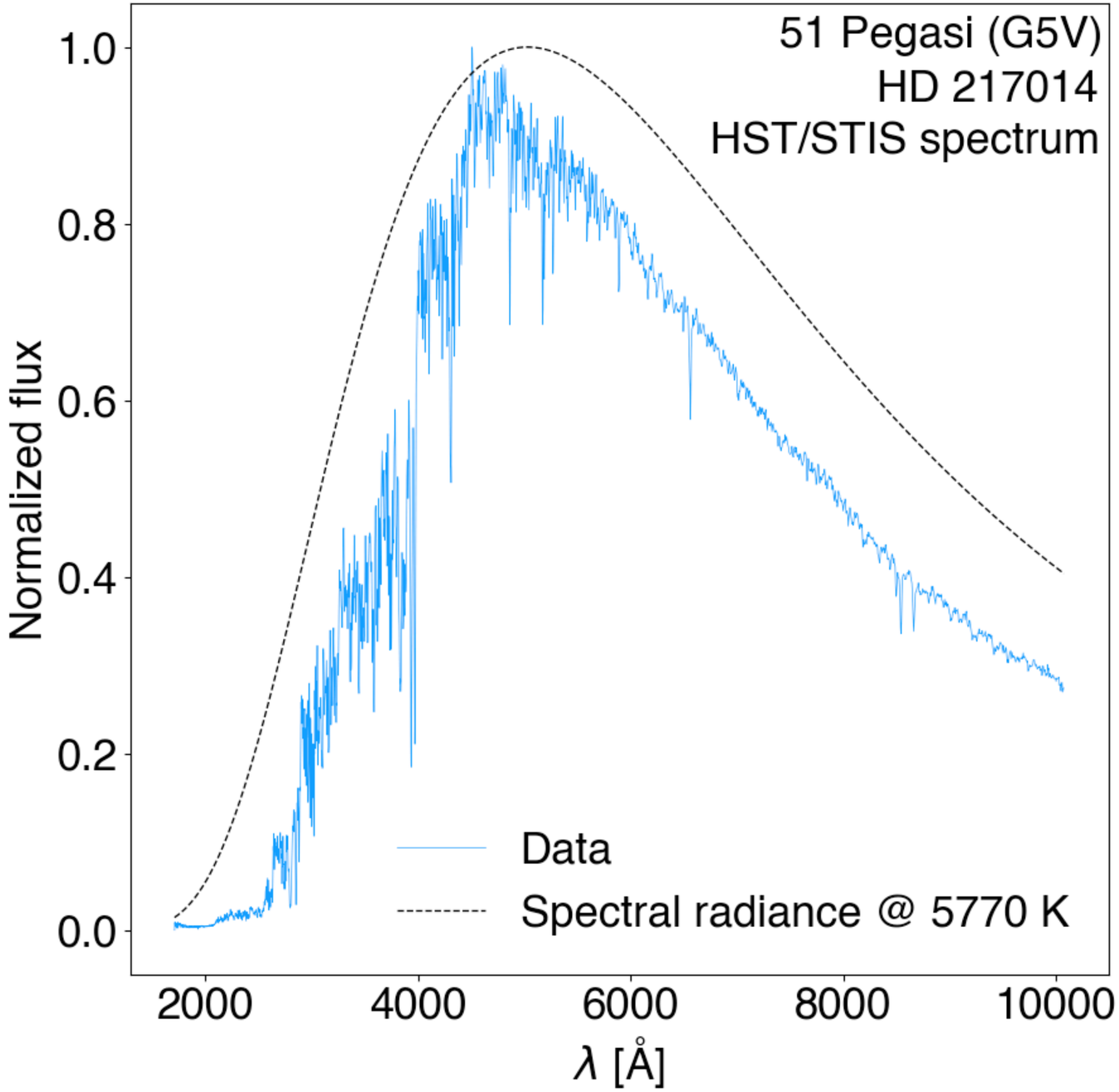
```
xkcd_fig = plot_colortable(mcolors.XKCD_COLORS)
xkcd_fig.savefig("XKCD_Colors.png")
```

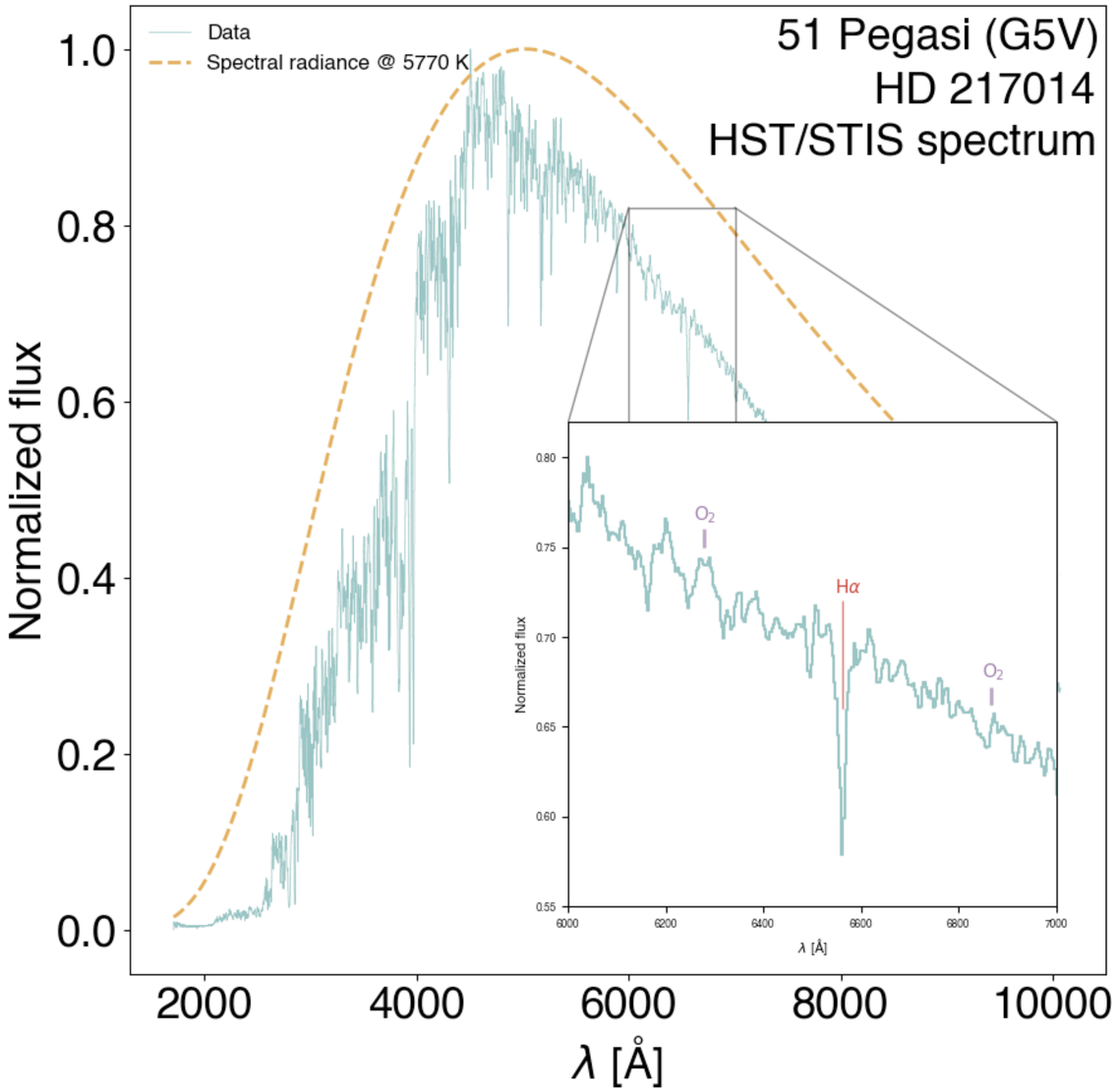
Spectral data visualization

```
column_names = ['wavelength', 'flux', 'u_flux']
spectrum = pd.read_csv('HD217014.txt', sep='\s+', names=column_names)

def blackbody(wavelength, temperature):
    h = 6.626e-34  # Planck constant
    c = 3.0e8      # Speed of light
    k = 1.38e-23   # Boltzmann constant

    wavelength_m = wavelength * 1e-10
    intensity = (2 * h * c**2 / wavelength_m**5) / (np.exp(h * c / (wavelength_m * k * temperature)) - 1)

    return intensity

plt.plot(spectrum['wavelength'], spectrum['flux'] / np.max(spectrum['flux']), label='Data')
plt.xlabel(r'$\lambda$ [Å]')
plt.ylabel(r'Normalized flux')

xx = np.linspace(np.min(spectrum['wavelength']), np.max(spectrum['wavelength']), len(spectrum['wavelength']))
plt.plot(xx, blackbody(xx, 5770)/np.max(blackbody(xx, 5770)), label='Spectral radiance @ 5770 K')

plt.legend(loc='lower right')
plt.show()
```

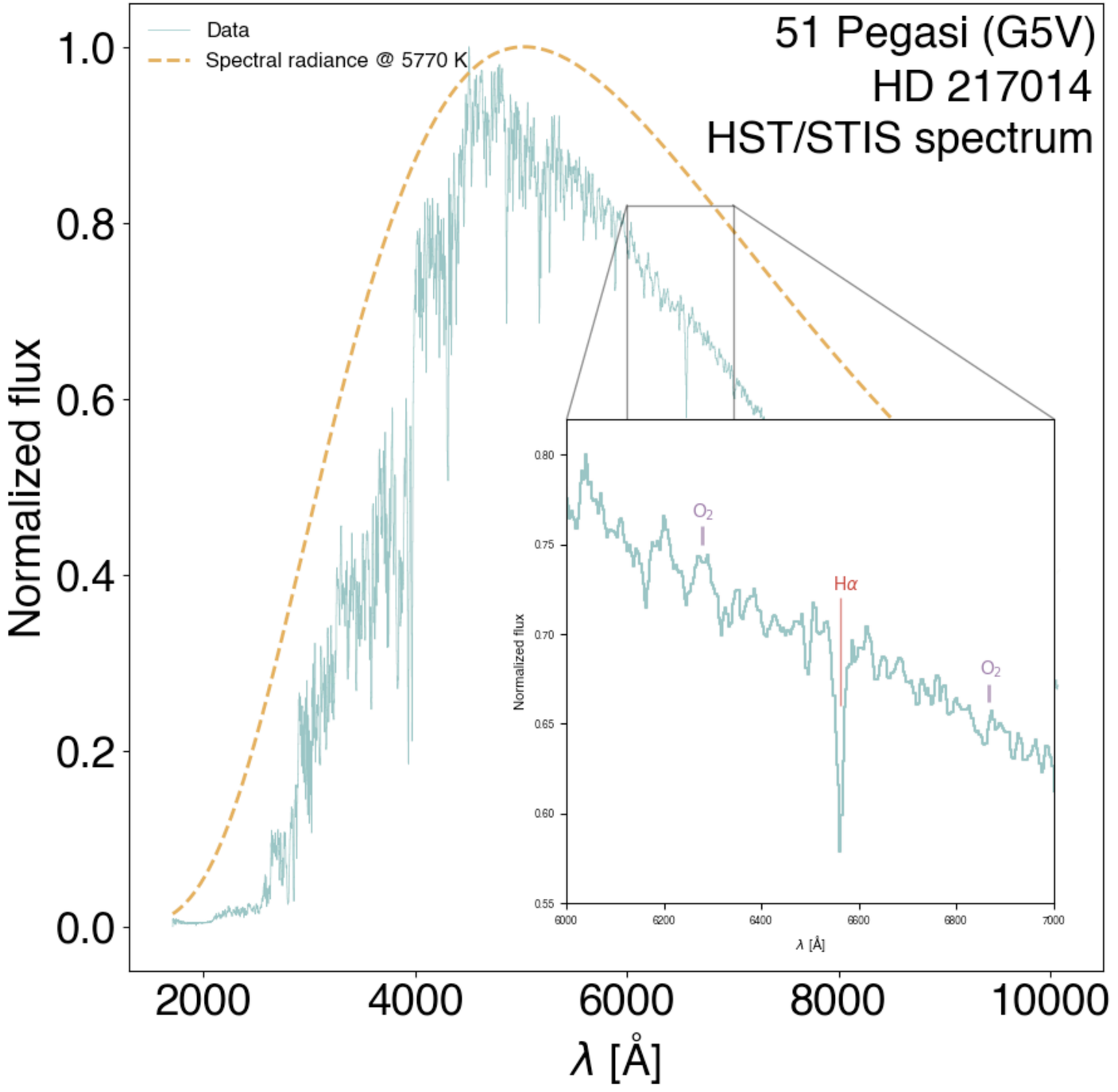

1.0
0.8
0.6
0.4
0.2
0.0
Normalized flux
2000
4000
6000
8000
10000
λ [Å]
Data
Spectral radiance @ 5770 K

```
plt.plot(spectrum['wavelength'],
         spectrum['flux'] / np.max(spectrum['flux']),
         color='dodgerblue', label='Data', lw=0.5)
plt.xlabel(r'$\lambda$ [Å]')
plt.ylabel(r'Normalized flux')

xx = np.linspace(np.min(spectrum['wavelength']),
                 np.max(spectrum['wavelength']),
                 len(spectrum['wavelength']))
plt.plot(xx, blackbody(xx, 5770)/np.max(blackbody(xx, 5770)),
         color='black', linestyle='--', lw=1,
         label='Spectral radiance @ 5770 K')

plt.text(7400, 1.0, '51 Pegasi (G5V)')
plt.text(8310, 0.94, 'HD 217014')
plt.text(6740, 0.88, 'HST/STIS spectrum')

plt.legend(loc='lower right', frameon=False)
plt.show()
```



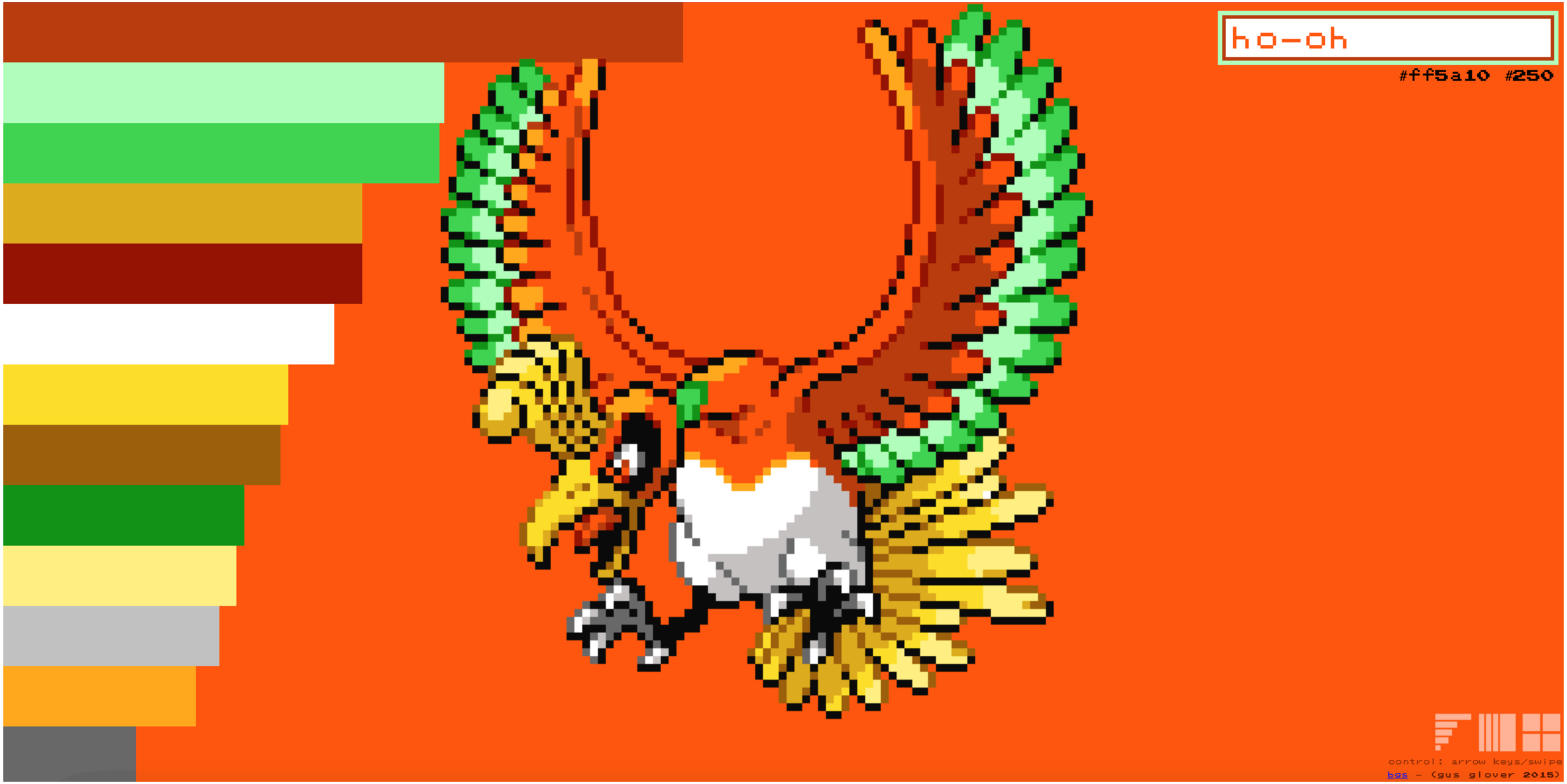



```
x1, x2, y1, y2 = 6000, 7000, 0.55, 0.82
axins  = ax.inset_axes([0.45, 0.07, 0.5, 0.5],
    xlim=(x1, x2), ylim=(y1, y2))

axins.step(spectrum['wavelength'],
           spectrum['flux'] / np.max(spectrum['flux']),
           lw=1.5, color='#8CBCB9', where='mid')
ax.indicate_inset_zoom(axins, edgecolor="black")
axins.tick_params(axis='both', which='major', labelsize=6)
axins.tick_params(axis='both', which='minor', labelsize=6)
axins.set_xlabel(r'$\lambda$ [Å]', size=8)
axins.set_ylabel(r'Normalized flux', size=8)

axins.vlines(6562.81, 0.66, 0.72, color='#BB342F', lw=1, alpha=0.7)
axins.text(6545, 0.725, r'H$\alpha$', size=10, color='#BB342F')

axins.vlines(6276.66, 0.75, 0.76,
             color='#8D6A9F', lw=2, alpha=0.7)
axins.text(6258, 0.765, r'O$_2$', size=10, color='#8D6A9F')

axins.vlines(6867.19, 0.662, 0.672,
             color='#8D6A9F', lw=2, alpha=0.7)
axins.text(6848, 0.677, r'O$_2$', size=10, color='#8D6A9F')
```


Data
Spectral radiance @ 5770 K
51 Pegasi (G5V)
HD 217014
HST/STIS spectrum
Normalized flux
λ [Å]
O2
Hα
O2

Press the spacebar to generate color palettes!

View Export Save



pokepalettes.com

3382 RESULTS:

ATOM-8 POTD

Created by polyphrog

very retro ;)

71 likes 5 comments

#atom8 | 8 colors | 654 downloads

Tags: retro, vibrant, polyphrog, atom

AETHER QUARTET RELAUNCHED NEW

Created by Birds Probably

A version of Aether Quartet using the originally intended colours.

7 likes 0 comments

#aetherquartetrelaunched | 4 colors | 17 downloads

GAWR8 NEW

Created by gawrone

2 likes 0 comments

#gawr8 | 8 colors | 2 downloads

Tags: turquoise, gold, coral

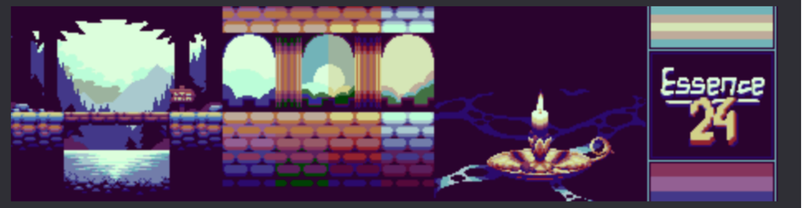
ESSENCE24 REFRESHED NEW

Created by Birds Probably

A version of Essence24 using the originally intended colours.

3 likes 0 comments

#essence24refreshed | 24 colors | 15 downloads



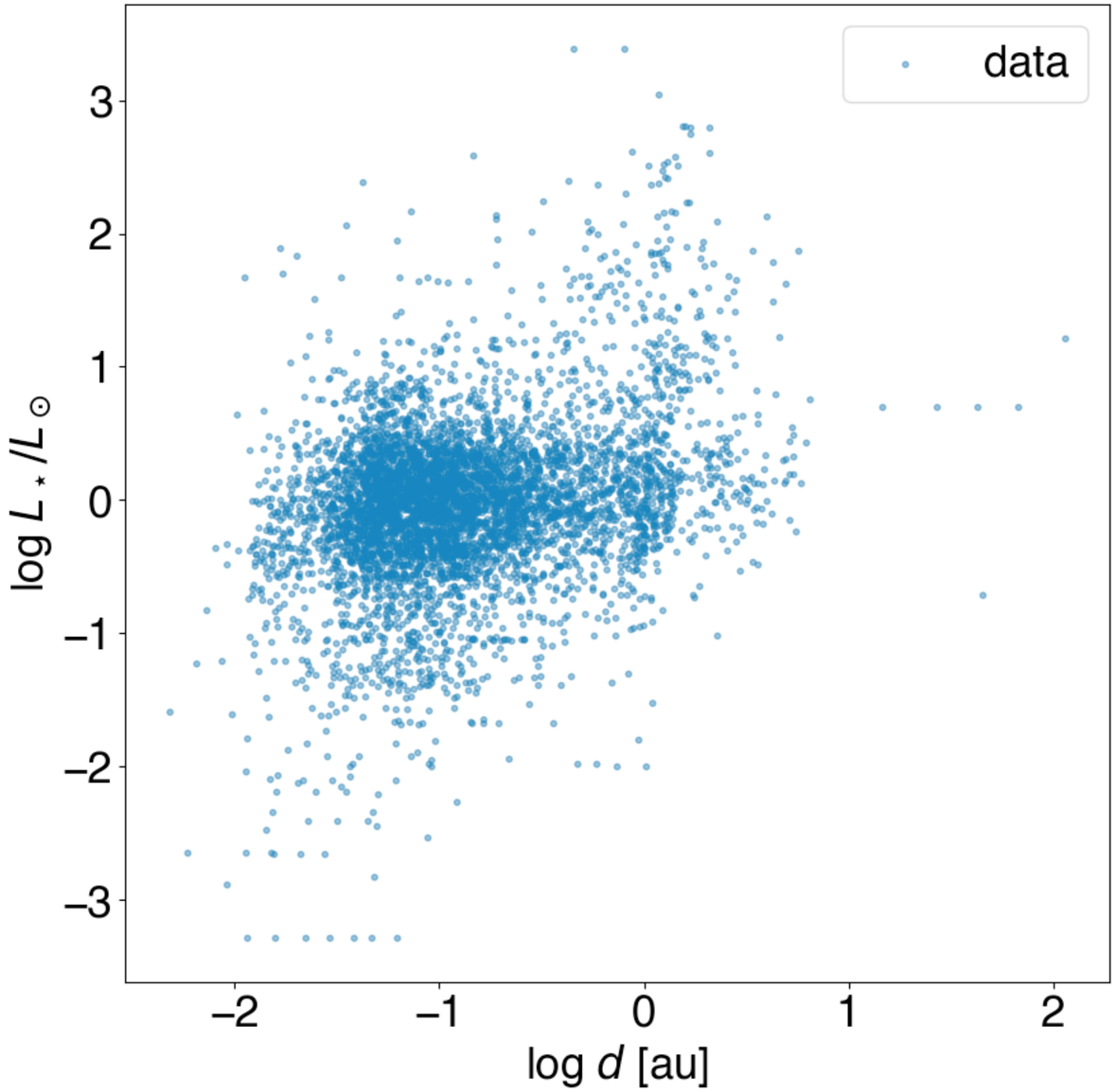
Tags: versatile, new24palettecontest, essence24

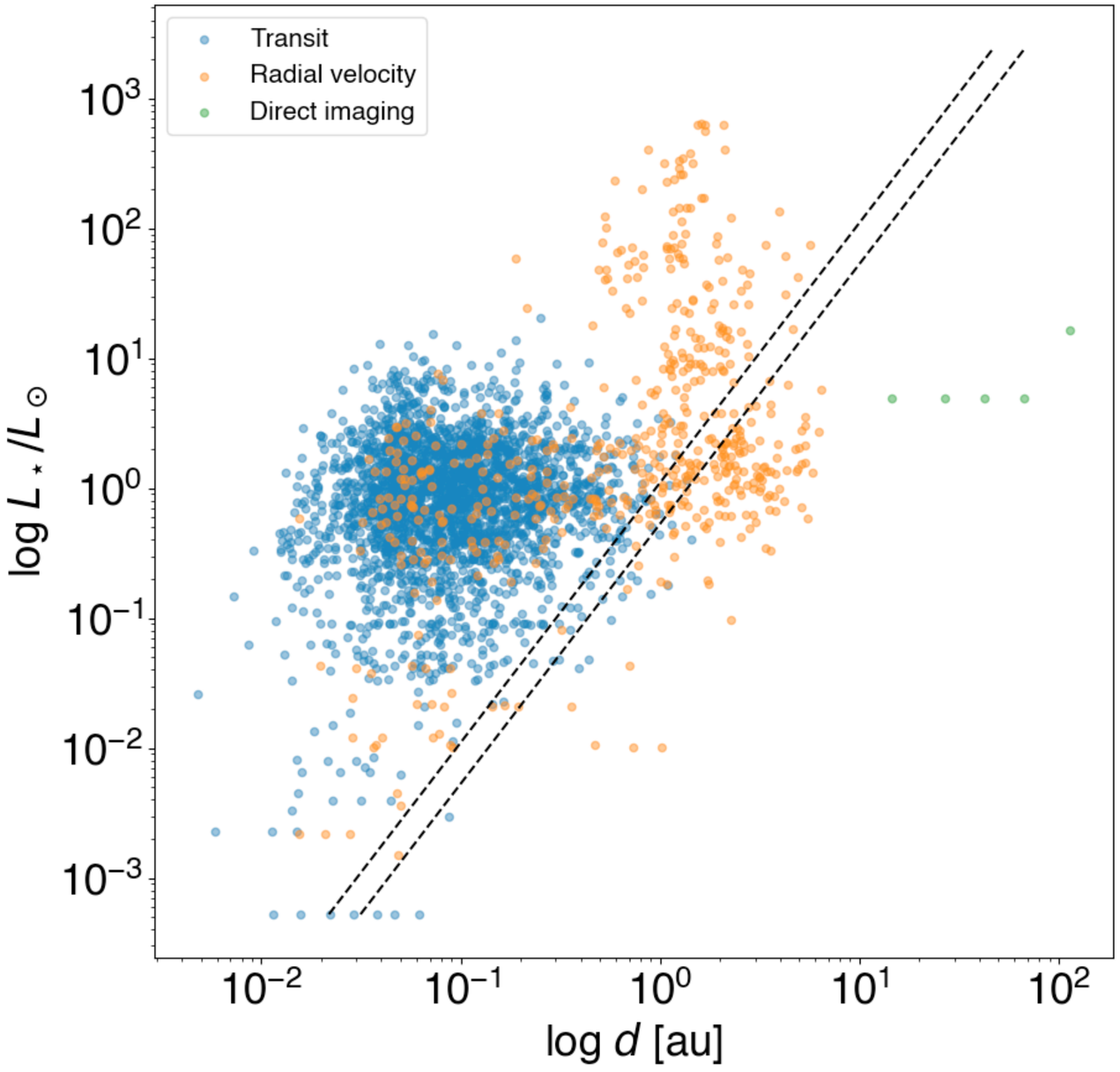
https://lospec.com/palette-list

Scatter plots

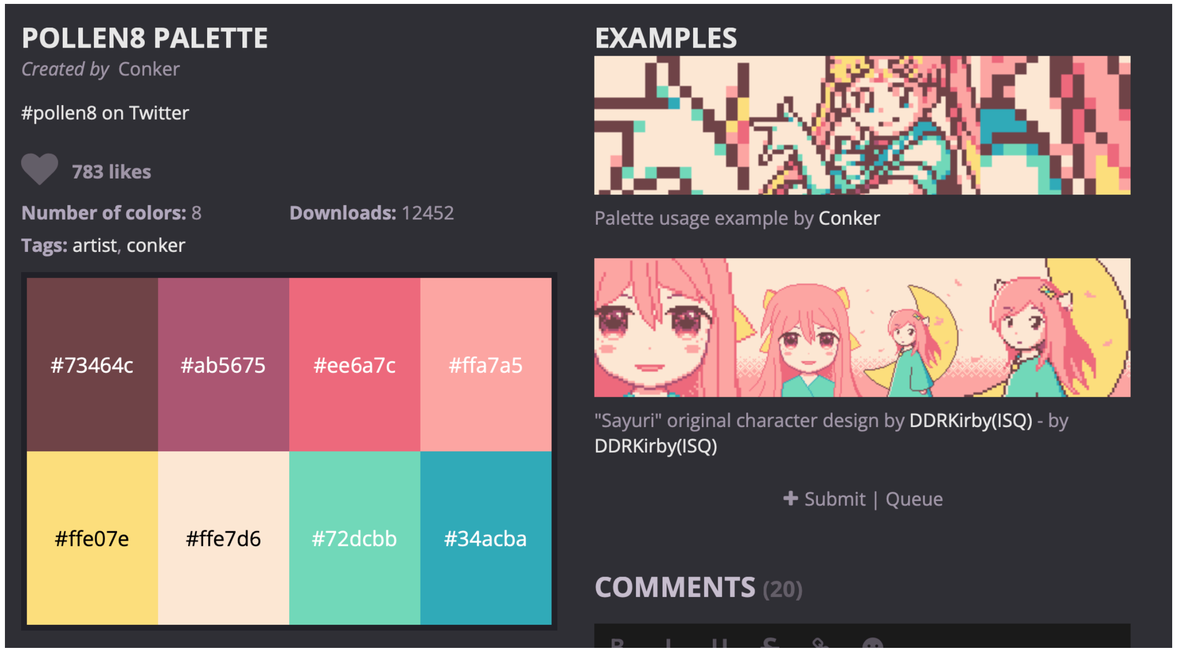
& misc data visualization

```
from astropy.constants import sigma_sb

df = pd.read_csv('exoplanets.csv')
L_star = sigma_sb * df['TEFF']**4 * 4 * np.pi * df['RSTAR']**2
L_sun = sigma_sb * 5780**4 * 4 * np.pi * 1**2
L_norm = L_star / L_sun

plt.scatter(np.log10(df['A']),
            np.log10(L_norm), s=10,
            alpha=0.5, label='data')
plt.xlabel(r'log $d$ [au]')
plt.ylabel(r'log $L_{\rm \star} / L_{\rm \odot}$')
plt.legend()
plt.show()
```

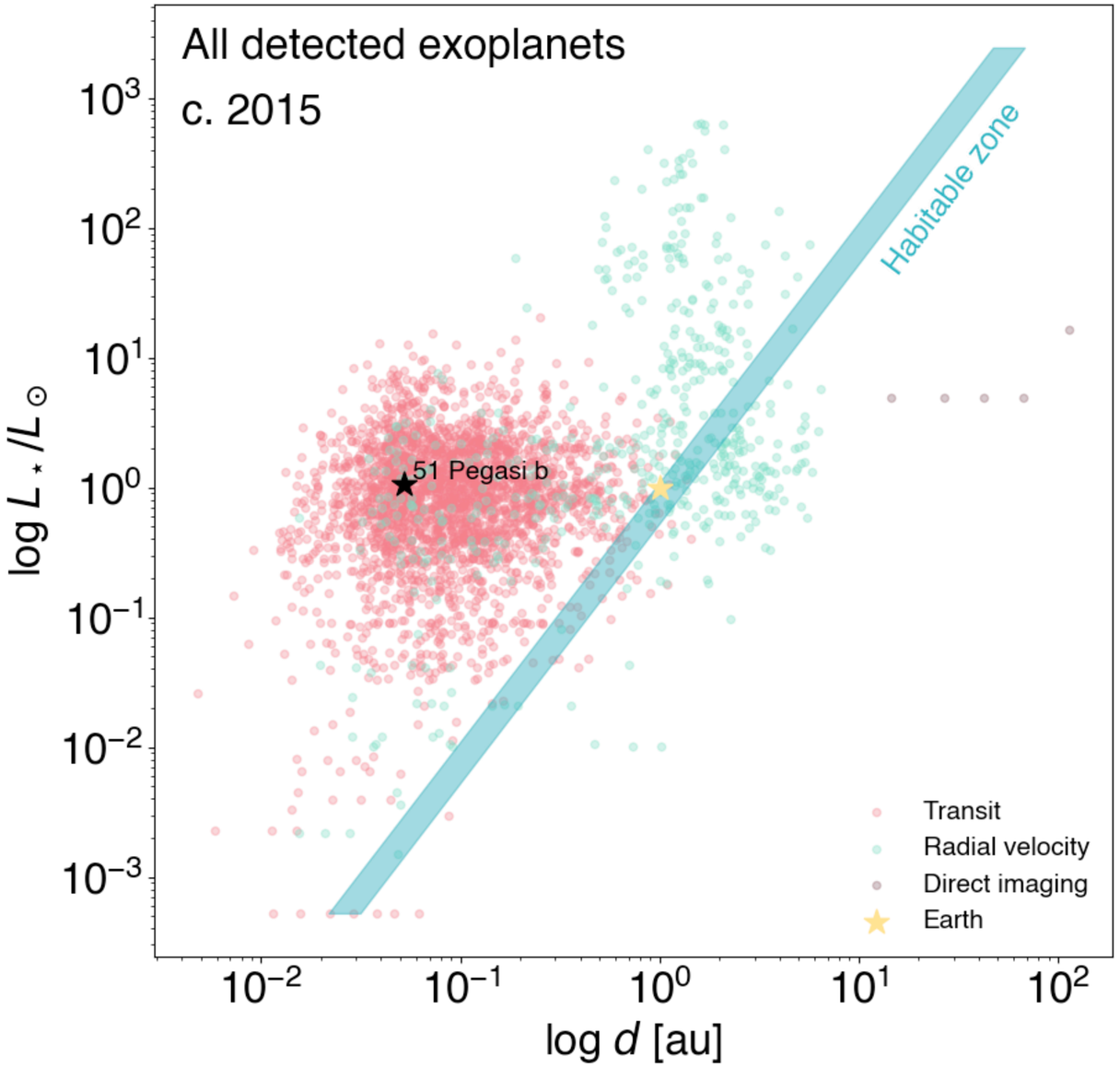




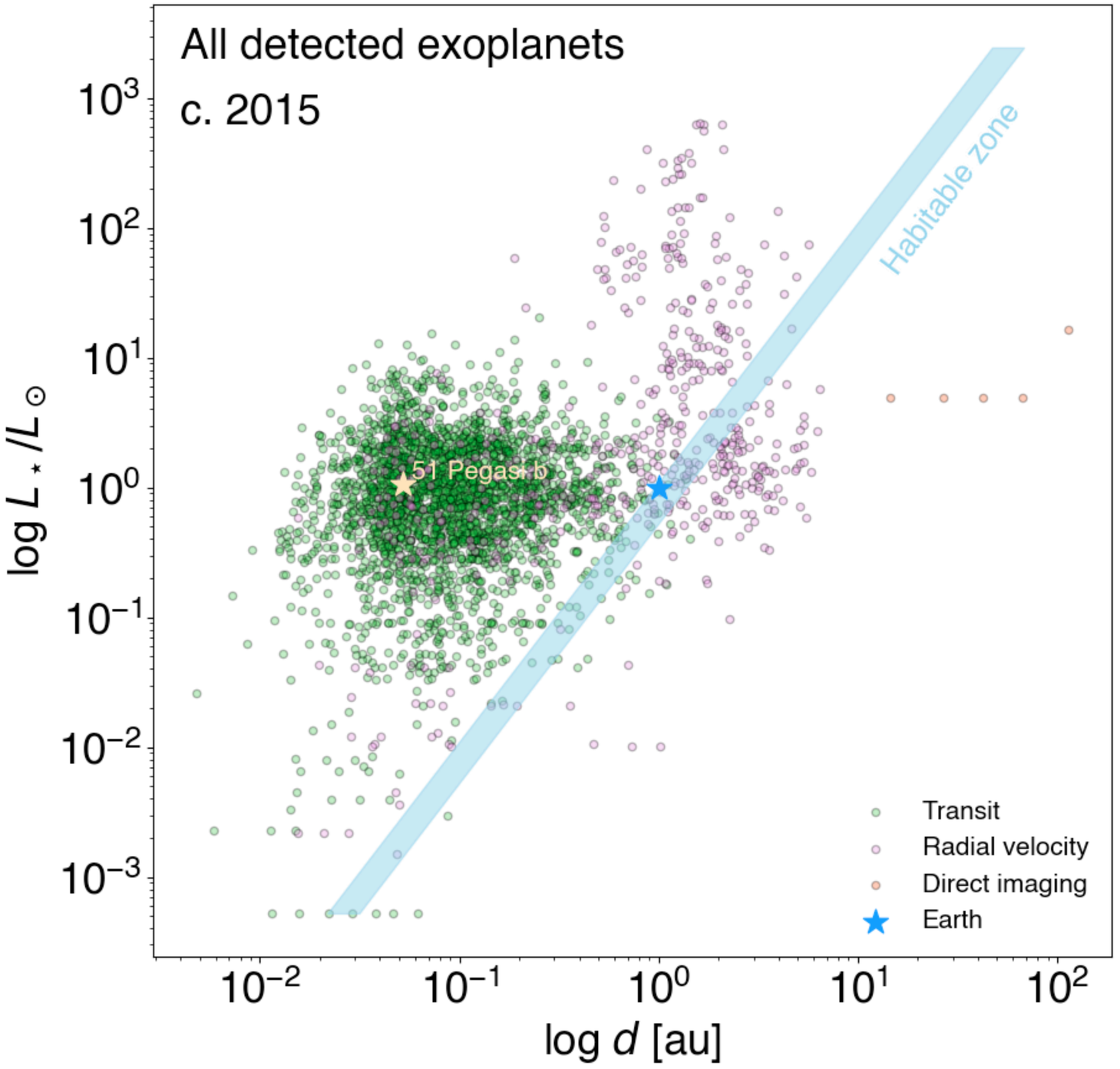

```
plt.figure()

yy = np.linspace(np.min(L_norm), np.max(L_norm), 100)

plt.scatter(df['A'][df['PLANETDISCMETH'] == 'Transit'],
            L_norm[df['PLANETDISCMETH'] == 'Transit'],
            s=20, alpha=0.5, label='Transit')
plt.scatter(df['A'][df['PLANETDISCMETH'] == 'RV'],
            L_norm[df['PLANETDISCMETH'] == 'RV'],
            s=20, alpha=0.5, label='Radial velocity')
plt.scatter(df['A'][df['PLANETDISCMETH'] == 'Imaging'],
            L_norm[df['PLANETDISCMETH'] == 'Imaging'],
            s=20, alpha=0.5, label='Direct imaging')

plt.plot((yy / 0.53)**(1/2), yy, color='black',
          linestyle = '--')
plt.plot((yy / 1.1)**(1/2), yy, color='black',
          linestyle = '--')

plt.xlabel(r'log $d$ [au]')
plt.ylabel(r'log $L_{\rm \star} / L_{\rm \odot}$')
plt.loglog()
plt.legend(fontsize=15)
plt.show()
```

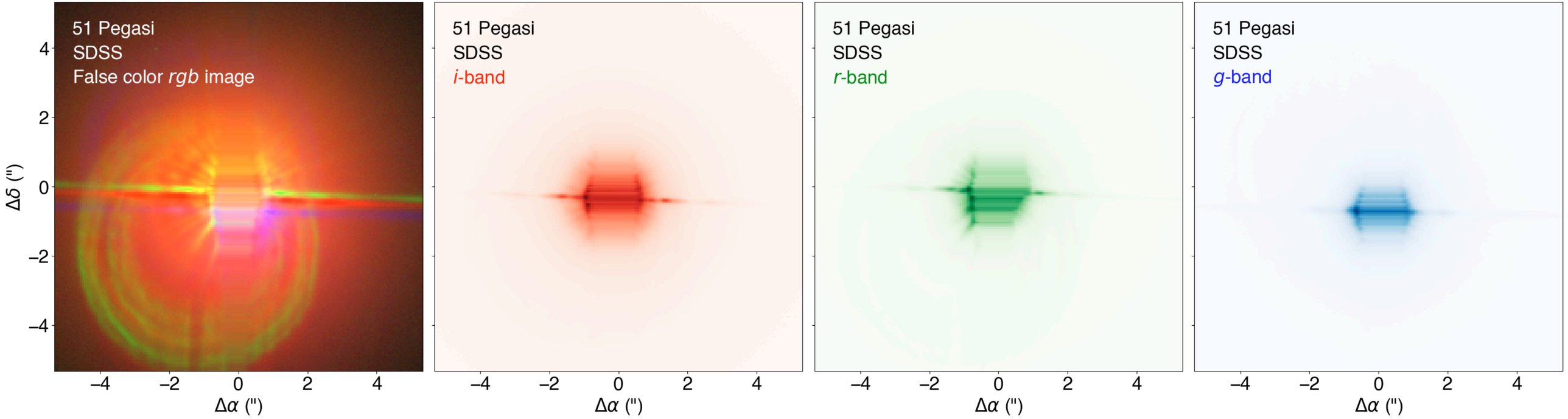


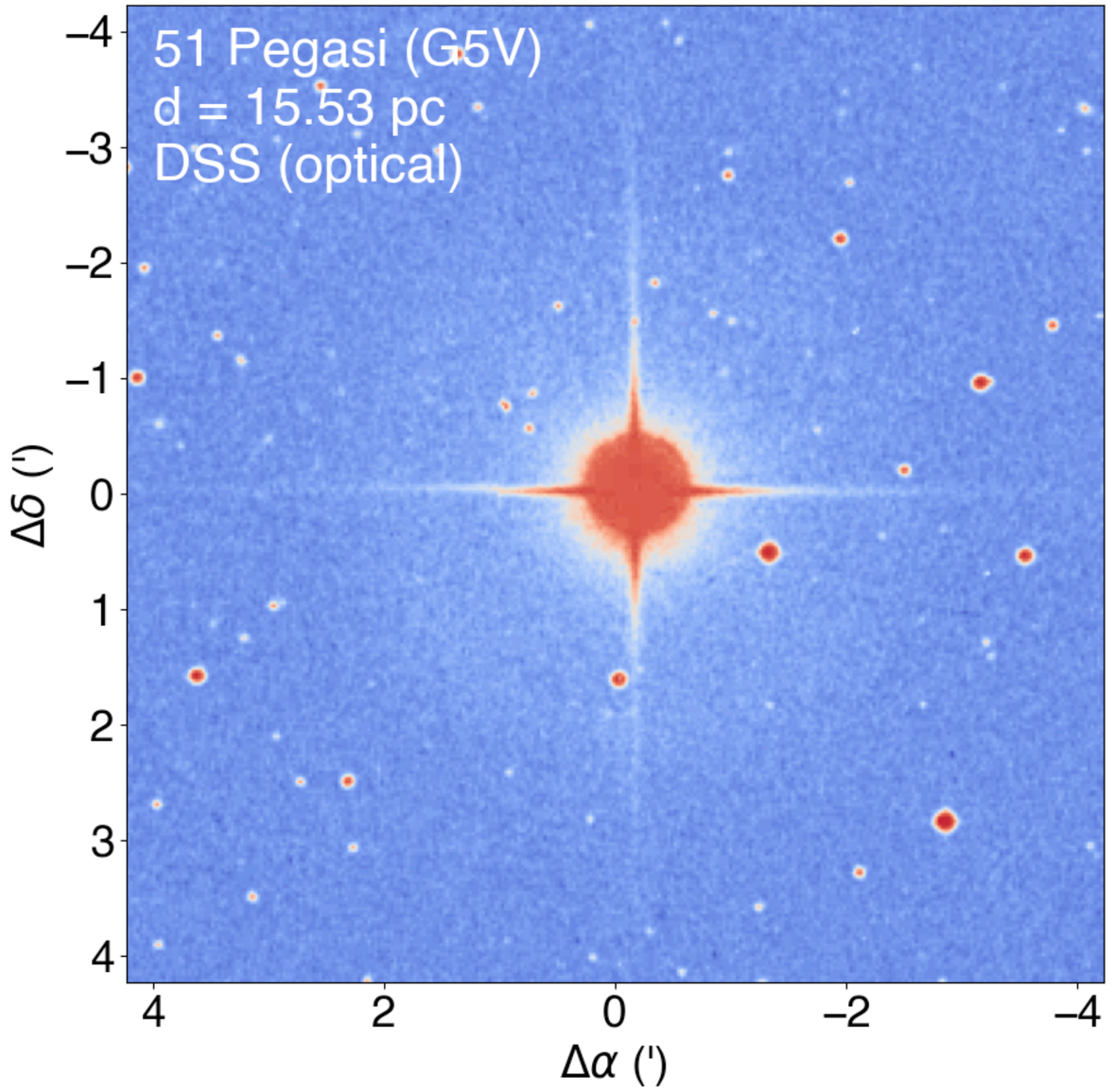
https://lospec.com/palette-list/pollen8

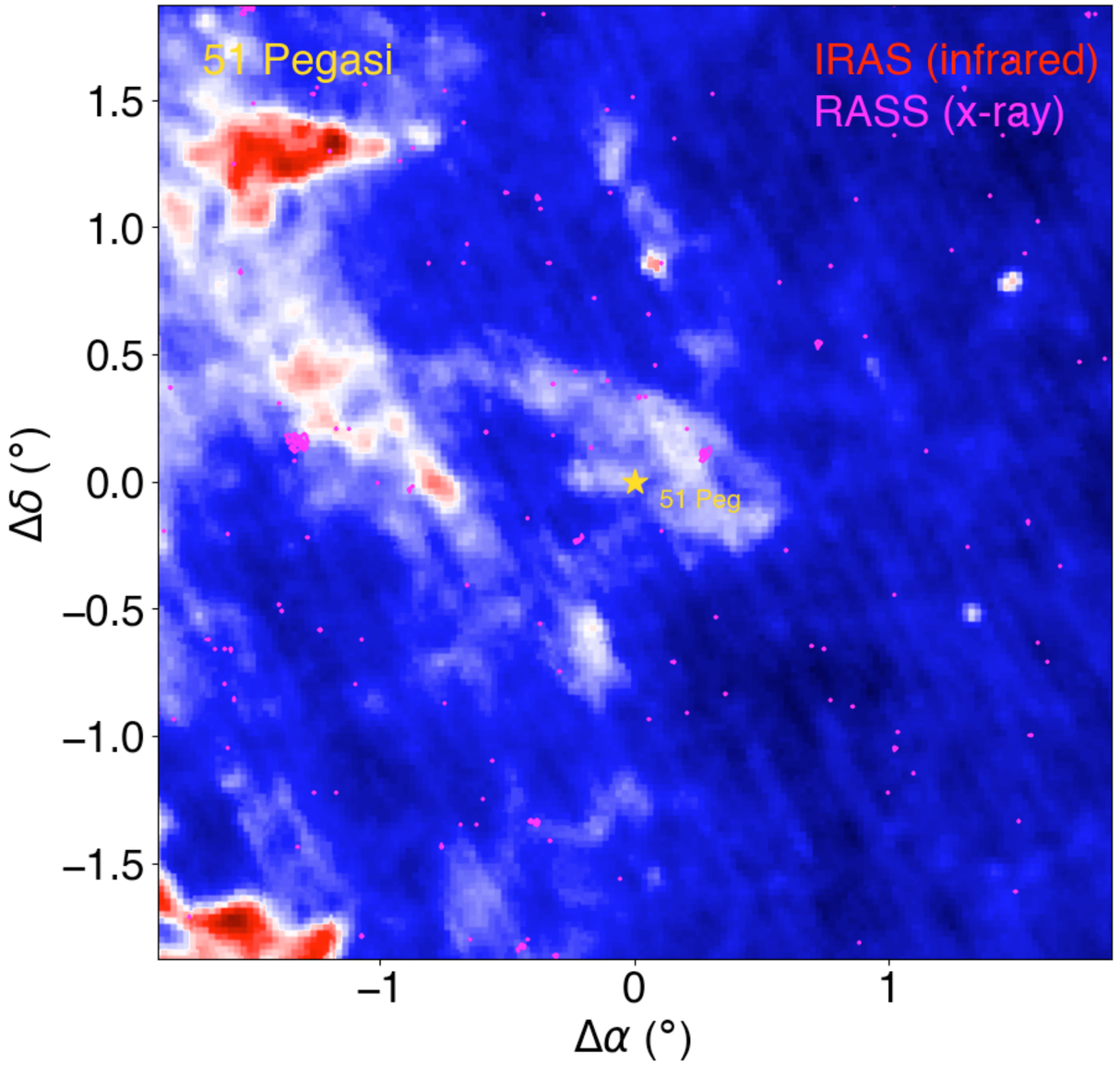


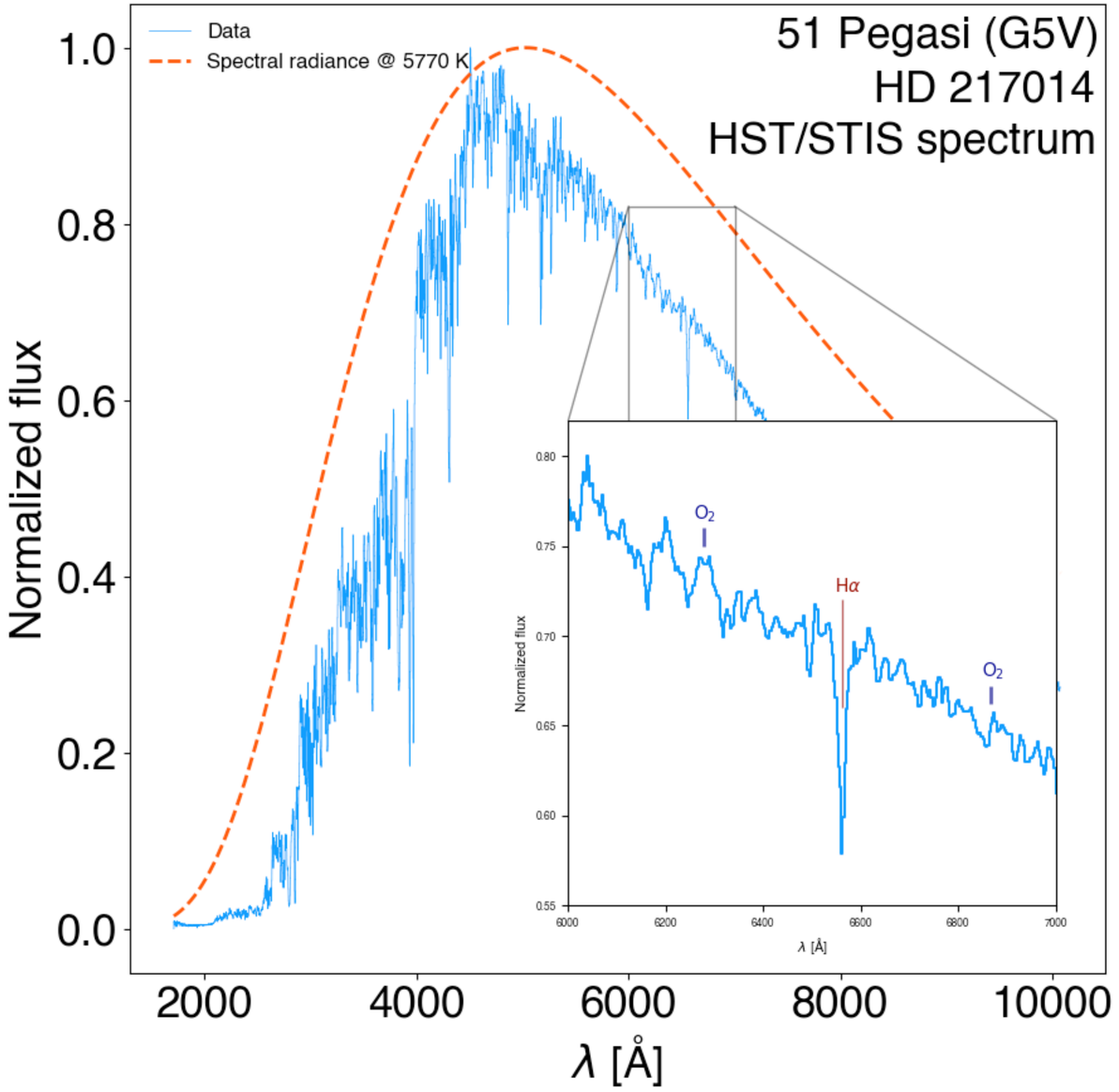
```
plt.scatter(df['A'][df['PLANETDISCMETH'] == 'Transit'],
            L_norm[df['PLANETDISCMETH'] == 'Transit'],
            s=20, edgecolors='black',
            alpha=0.33, color='limegreen', label='Transit')
plt.scatter(df['A'][df['PLANETDISCMETH'] == 'RV'],
            L_norm[df['PLANETDISCMETH'] == 'RV'],
            s=20, edgecolors='black',
            alpha=0.33, color='violet', label='Radial velocity')
plt.scatter(df['A'][df['PLANETDISCMETH'] == 'Imaging'],
            L_norm[df['PLANETDISCMETH'] == 'Imaging'],
            s=20, edgecolors='black',
            alpha=0.33, color='orangered', label='Direct imaging')

plt.fill_betweenx(yy,(yy / 0.53)**(1/2), (yy / 1.1)**(1/2),
                  color='skyblue', alpha=0.5)

plt.text(10**-2.4, 10**3.3, 'All detected exoplanets')
plt.text(10**-2.4, 10**2.8, 'c. 2015')
plt.text(10**1.1, 10**1.65, r'Habitable zone',
         fontsize=18, rotation=53, color='skyblue')

plt.scatter(1,1, marker='*', color='dodgerblue', label='Earth')

plt.scatter(df['A'][df['EANAME'] == '51 Peg b'],L_norm[df['EANAME'] == '51 Peg b'],
            marker='*', color='navajowhite')

plt.text(df['A'][df['EANAME'] == '51 Peg b']*1.1, L_norm[df['EANAME'] == '51 Peg b']* 1.1,
         '51 Pegasi b', color='navajowhite', size=15)

plt.xlabel(r'log $d$ [au]')
plt.ylabel(r'log $L_{\rm \star} / L_{\rm \odot}$')
plt.loglog()
plt.legend(loc='lower right', frameon=False, fontsize=15)
plt.show()
```

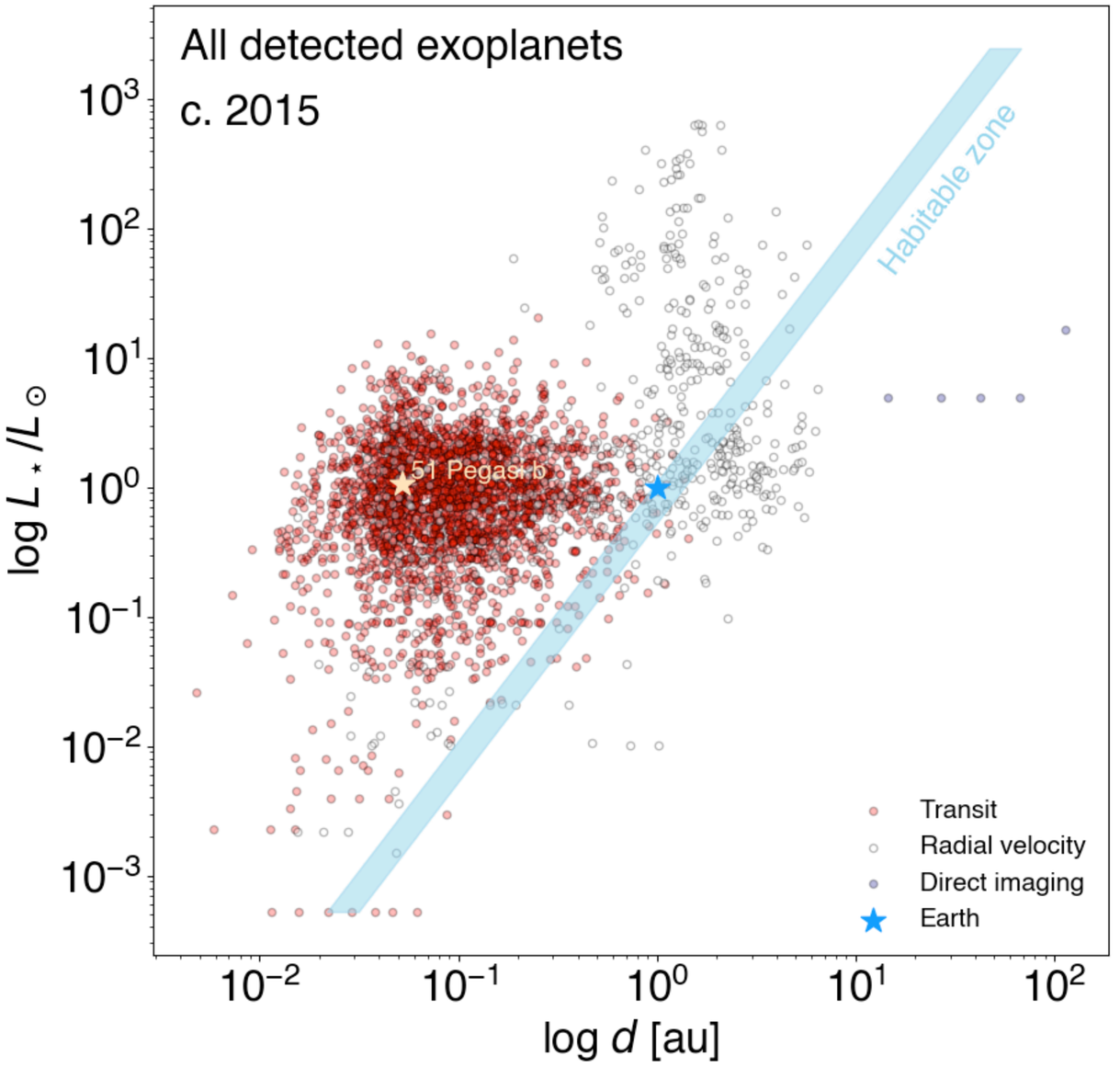

All detected exoplanets
c. 2015
Habitable zone
51 Pegasi b
Transit
Radial velocity
Direct imaging
Earth
log $L_\star / L_\odot$
log d [au]
10^{3}
10^{2}
10^{1}
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-2}
10^{-1}
10^{0}
10^{1}
10^{2}


51 Pegasi
SDSS
False color rgb image
51 Pegasi
SDSS
i-band
51 Pegasi
SDSS
r-band
51 Pegasi
SDSS
g-band
Δδ (")
Δα (")
4
2
0
−2
−4


51 Pegasi (G5V)
d = 15.53 pc
DSS (optical)
−4
−3
−2
−1
0
1
2
3
4
Δδ (')
4
2
0
−2
−4
Δα (')


51 Pegasi
IRAS (infrared)
RASS (x-ray)
51 Peg
1.5
1.0
0.5
0.0
–0.5
–1.0
–1.5
Δδ (°)
–1
0
1
Δα (°)


Data
Spectral radiance @ 5770 K
51 Pegasi (G5V)
HD 217014
HST/STIS spectrum
Normalized flux
λ [Å]
O2
Hα
O2


All detected exoplanets
c. 2015
Habitable zone
51 Pegasi b
Transit
Radial velocity
Direct imaging
Earth
log $L_\star / L_\odot$
log d [au]
10^{3}
10^{2}
10^{1}
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-2}
10^{-1}
10^{0}
10^{1}
10^{2}

I will make a notebook available this weekend

Thank you!!