Getting computing into the classroom: playing with exoplanets

Erik Spence

SciNet HPC Consortium

15 January 2015

Planetary data

Social media data are not the only data available online. Some of the latest scientific data can be accessed using the correct tools.

The plan for today:

- A review of exoplanets, for those that aren't familiar with them.
- Introduction to the Kepler satellite and its mission to discover exoplanets.
- An re-introduction to Python's dictionary datatype.
- An introduction to Python DataFrames.
- An introduction to the Python kplr package, which can be used to access the latest data on exoplanets.
- Test to see which exoplanets might be inhabitable.

SciNet compute • calcul CANADA

Exoplanets

What are exoplanets?

- "Exoplanets" are the unfortunate name of "extrasolar planets", meaning planets which orbit a star (or stellar remnant, or brown dwarf) other than the Sun.
- These do not usually include "rogue planets", meaning planets which do not orbit anything, other than the galactic centre.
- The naming convention is all-weird, with the naming of planets starting with the star's name, followed by 'b', 'c', *etc*.
- ('a' is reserved for the brightest object in the system, namely the star.)
- As of 5 January 2015, there were 1523 confirmed exoplanets, according to exoplanets.org.

Are you sure?

How are exoplanets found? There are several techniques that have been used.

- Transit method: A planet crosses in front of the parent star, resulting in a drop in brightness, which is used to determine the radius of the planet. Often results in false positives.
- Radial velocity method: As the planet moves around the star, the star wobbles. The Doppler shift of the star's light, which corresponds to the 'radial velocity' of the star with respect to the Earth, will be changed by the motion of the planet.
- Direct imaging: sometimes a planet is large enough and far enough away from the host star to be directly observable.
- A bunch of others.

The "Kepler" mission is a NASA satellite whose purpose is to detect exoplanets using the transit method.

The Kepler telescope

The Kepler space observatory is a NASA mission to discover Earth-like planets orbiting other stars.

- Launched on 7 March 2009. Started taking data the next day.
- Uses the transit method to detect planets.
- Continuously monitors the brightness of 145,000 Milky Way stars.
- The initial lifetime was 3.5 years, but was extended in 2012 until 2016.
- However, hardware failures in 2013 resulted in a temporary cessation of planet searching.
- Planet searching resumed in 2014 using a different technique, and lower accuracy.

We will use data from Kepler to study what is known thus far about exoplanets.

SciNet compute • calcul CANADA

Dictionaries

Dictionaries are a python data type which associates keys to values.

Create a dictionary:

```
In [0]: a = dict(one = 1, two = 2, three = 3)

In [1]:

In [1]: b = {'one': 1, 'two': 2, 'three': 3}

In [2]:

In [2]: c = {}

In [3]: c['one'] = 1; c['two'] = 2; c['three'] = 3

In [4]:

In [4]: c
{'one': 1, 'three': 3, 'two': 2}

In [5]:
```

All of the above commands are equivalent.

Dictionaries, continued

Dictionaries have some handy functions built in.

```
In [5]: a.keys()
['three', 'two', 'one']
```

```
In [6]: 'one' in b
True
```

```
In [7]: 'four' in b
False
```

```
In [8]: a.setdefault('four',0)
```

```
In [9]: a
{'four': 0, 'one': 1, 'three': 3, 'two': 2}
```

```
In [10]: c['one']
1
```

Dictionaries are most useful when you need a dynamic datatype that can grow as needed.

Getting exoplanet data

To get the exoplanet data, we use the "kplr" package, which provides interfaces to several types of data:

- KOIs ("Kepler Objects of Interest") from the Exoplanet Archive.
- Confirmed exoplanets from the Mikulski Archive for Space Telescopes (MAST).
- The Kepler Input Catalogue (also from MAST).
- NASA Exoplanet Archive data.

```
In [11]:
In [11]: import kplr
In [12]:
In [12]: client = kplr.API()
In [13]:
In [13]: data =
  client.ea_request('exoplanets',
  select = '*')
In [14]:
In [14]: len(data)
1781
In [15]:
In [15]: len(data[0].keys())
389
In [16]:
```

What did we get?

Entry	Meaning
pl_hostname	Host Star Name
pl_letter	Planet Letter
pl_discmethod	Discovery Method
pl_pnum	Number of Planets in System
pl_orbper	Orbital Period (days)
pl_orbsmax	Orbital Semi-Major Axis (AU)
pl_orbeccen	Eccentricity
pl_massj	Planet Mass (Jupiter mass)
pl_radj	Planet Radius (Jupiter radii)
st_mass	Stellar Mass (solar mass)
st_rad	Stellar Radius (solar radii)
st_teff	Stellar Temperature (K)
st_lum	Stellar Luminosity (log Solar Luminosity)
st_dist	Distance (pc)
ra	Right Ascension (degrees)
dec	Declination (degrees)

Example data

```
In [16]: data[0]['pl_hostname']
'Kepler-126'
In [17]: data[0]['pl_discmethod']
'Transit'
In [18]:
In [18]: data[0]['pl_pnum']
3
In [19]:
In [19]: data[0]['pl_letter']
'b'
In [20]:
In [20]: data[0]['pl_massj']

In [21]:
In [21]: data[0]['pl_radj']
0.136
In [22]:
```

```
In [22]: data[0]['pl_orbper']
10.495711
In [23]: data[0]['pl_orbsmax']
0.099
In [24]:
In [24]: data[0]['pl_orbeccen']

In [25]:
In [25]: data[0]['st_rad']
1.36
In [26]:
In [26]: data[0]['st_teff']
6239.0
In [27]:
```

The Sun's surface temperature is about 5778K.

Questions for students

The exoplanet data immediately lend themselves to tonnes of questions:

- What is the most massive planet discovered thus far? How heavy is it? By what method was it discovered?
- What about the smallest planet?
- What is the distribution of planets, as a function of discovery method? Which method has been most successful?
- How many star systems contain more than one planet?
- What is the typical mass of a star where an exoplanet is discovered?
- Create a histogram plot of planet radius.
- Are there any planets that are similar to Earth, in terms of size, orbital radius?
- Are any of the planets inhabitable?

SciNet compute • calcul CANADA

Using dataframes

Dictionaries are useful, but they can sometimes be clunky when analysing batches of similar objects, such as in this case.

DataFrames are somewhat like Excel spreadsheets, but more versatile, and are available in Python using the pandas package.

```
In [27]: import pandas as pd
```

```
In [28]: df = pd.DataFrame(data)
```

```
In [29]: df.shape
(1781, 389)
```

```
In [30]: df[0:3]
           dec         dec_str   hd_name   hip_name   pl_dens   pl_denserr1
 0   44.208553   +44d12m30.8s      None       None       NaN           NaN
 1   44.208553   +44d12m30.8s      None       None       NaN           NaN
 2   44.208553   +44d12m30.8s      None       None       NaN           NaN

⋮
[3 rows x 389 columns]
```

```
In [31]:
```

Using dataframes, continued

Suppose we wish to answer the question: What is the distribution of exoplanet masses? Using a DataFrame we can easily filter and sort the data, without the need to do any loops.

```
In [31]: df['pl_massj'][0]
nan
```

```
In [32]:
```

```
In [32]: weighty_planets = df[ df['pl_massj'].notnull() ]
```

```
In [33]: len(weighty_planets)
523
```

```
In [34]: type(weighty_planets)
pandas.core.frame.DataFrame
```

```
In [35]:
```

```
In [35]: import numpy as np
```

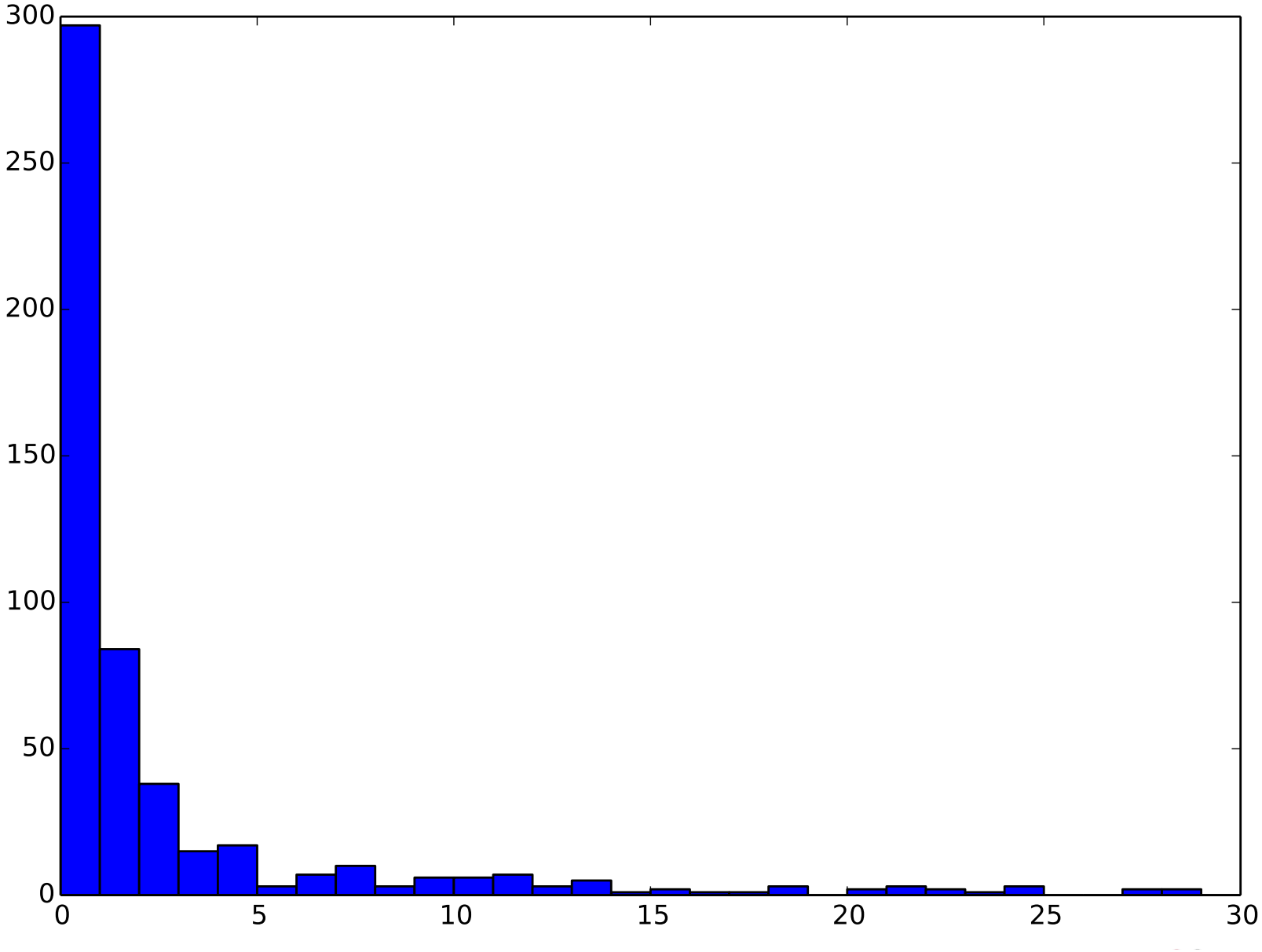
```
In [36]: weights = np.array(weighty_planets['pl_massj'])
```

```
In [37]:
```

```
In [37]: import matplotlib.pyplot as plt
```

```
In [38]: h = plt.hist(weights, np.linspace(0.0, 30.0, 31))
```

Exoplanet weights



How did that work?

The command

```
weighty_planets = df[ df['pl_massj'].notnull() ]
```

picked out the non-zero planet masses. How did that work?

```
In [39]: df["pl_massj"].notnull()
  0       True
  1      False
  2      False
  3      False
  ...
  1777   False
  1778    True
  1779   False
  1780   False
Name: pl_massj, Length:  1781, dtype:  bool
---------------------------------------------------
In [40]:
```

When we index our variable df with this array of booleans, we pick out the entries (or entry) we're interested in.

The biggest planet

I was curious about that huge planet. Let's poke it and see what it says.

```
In [40]: weighty_planets['pl_massj'].max()
28.5
```

```
In [41]: bigboy =
  df[ df['pl_massj'] == 28.5 ]
```

```
In [42]: len(bigboy)
1
```

```
In [43]: bigboy['pl_radj']
534    NaN
```

```
In [44]: bigboy['pl_discmethod']
534    Astrometry
```

```
In [45]: bigboy['pl_orbper']
534    246.36
```

```
In [46]: bigboy['pl_orbsmax']
534    0.36
```

```
In [47]: bigboy['st_mass']
534    0.07
```

SciNet compute • calcul CANADA

The biggest planet

I was curious about that huge planet. Let's poke it and see what it says.

```
In [51]: weighty_planets['pl_massj'].max()
28.5
```

```
In [52]: bigboy =
  df[ df['pl_massj'] == 28.5 ]
```

```
In [53]: len(bigboy)
1
```

```
In [54]: bigboy['pl_radj']
534    NaN
```

```
In [55]: bigboy['pl_discmethod']
534    Astrometry
```

```
In [56]: bigboy['pl_orbper']
534    246.36
```

```
In [57]: bigboy['pl_orbsmax']
534    0.36
```

```
In [58]: bigboy['st_mass']
534    0.07
```

```
In [59]: s_mass = 1.989e30
```

```
In [60]: j_mass = 1.898e27
```

```
In [61]: (j_mass * 28.5) /
  (s_mass * 0.07)
0.388515406162465
```

```
In [62]:
```

This planet is 1/3 the size of it's host star!

Astrometry is the measurement of the wobble of the star caused by the planet.

The fastest planet

What about planets with really short orbital periods?

```
In [62]: mintime = df['pl_orbper'].min()
In [63]: print(mintime)
0.0907062899999995
In [64]: mintime * 24
2.1769509600000001
In [65]:
In [65]: fastboy =
  df[ df['pl_orbper'] == mintime ]
In [66]:
In [66]: fastboy['pl_orbsmax']
1625   0.0044
In [67]: fastboy['pl_massj']
1625   1.4
In [68]: fastboy['st_mass']
1625   1.2
In [69]: fastboy['st_rad']
1625   0.04
```

```
In [70]: fastboy['pl_discmethod']
1625   Pulsar Timing
In [71]: fastboy['pl_hostname']
1625   PSR J1719-1438
In [72]:
```

Looking this star up in Wikipedia, we discover that this is a pulsar, and the planet is likely an ex-star itself.

Pulsar timing involves measuring variations in a pulsar's pulses, to determine the existence of a planet.

SciNet compute • calcul CANADA

Inhabitability

Whether or not a planet is capable of supporting life as we know it is one of the first things that is checked whenever a new planet is discovered.

How do we determine if a planet is inhabitable? A planet needs two things to support life as we know it:

- An oxygen atmosphere.
- Liquid water.

The first criteria requires a planet to have a radius at least half that of Earth's. The second requires a radius of no more than twice that of Earth's, lest all water be crushed into ice by gravity.

But obviously the orbital radius is the most important data point with respect to the second criteria.

SciNet compute • calcul CANADA

Inhabitability, continued

How do we determine the acceptable inhabitable radii for a given star? The Planetary Habitability Laboratory* has given us some equations for the inner and outer inhabitable orbital radii, in units of AU:

$$r_i = \left[r_{is} - a_i(T_{\text{eff}} - T_s) - b_i(T_{\text{eff}} - T_s)^2\right]\sqrt{L}$$
$$r_o = \left[r_{os} - a_o(T_{\text{eff}} - T_s) - b_o(T_{\text{eff}} - T_s)^2\right]\sqrt{L}$$

where L is the stellar luminosity, in solar units, and T_{eff} is the effective stellar temperature, in Kelvin. Code which calculates these radii can be found in inhabitability.py, which is on your Desktop.

*`http://phl.upr.edu`

symbol	Value
T_s	5700 (K)
a_i	2.7619e-5
b_i	3.8095e-9
a_o	1.3786e-4
b_o	1.4286e-9
r_{is}	0.72
r_{os}	1.77

Checking the data for inhabitability

Let's check if any of the exoplanets are potentially inhabitable. First check to see which planets have the data that we need: $T_{\mathbf{eff}}$, stellar luminosity and planet radius. Note that 1 Jupiter radius is equal to 10.97 Earth radii.

```
In [72]: good_stars = df['st_lum'].notnull() & df['st_teff'].notnull()
In [73]:
In [73]: good_planets = (df['pl_radj'] * 10.97 >= 0.5) &
  (df['pl_radj'] * 10.97 <= 2.0)
In [74]:
In [74]: data2 = df[good_stars & good_planets]
In [75]: len(data2)
38
```

Now that we've reduced the dataset to those data which have the required fields, and meet the radius criteria, we're ready to test against the inhabitability radii.

Checking the data for inhabitability, cont

Now that we've filtered the dataset, check each row in the resulting DataFrame to see if the planet's inhabitable.

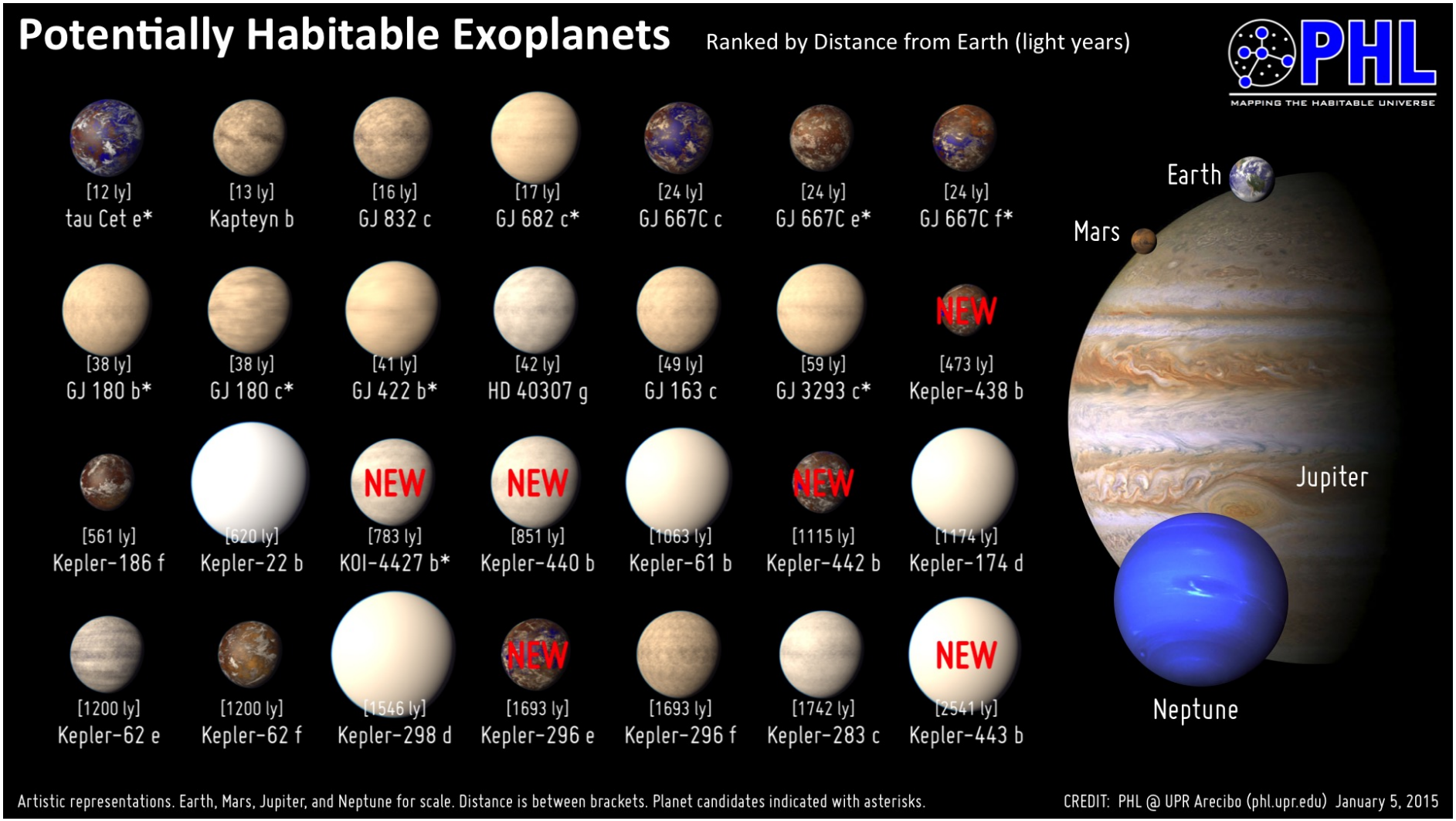
```
In [76]: import inhabitability as en
```

```
In [77]: for index, row in data2.iterrows():
  ...  r_inner = en.ri(row['st_teff'], 10**row['st_lum'])
  ...  r_outer = en.ro(row['st_teff'], 10**row['st_lum'])
  ...  if ((row['pl_orbsmax'] >= r_inner) and (row['pl_orbsmax'] <= r_outer)):
    ...  print(index, row['pl_hostname'], row['pl_letter'])
  ...
103 Kepler-442 b
1077 Kepler-438 b
1079 Kepler-440 b
1264 Kepler-186 f
1481 Kepler-62 e
1482 Kepler-62 f
```

```
In [78]:
```

How many winners are there? Six by these simple criteria.

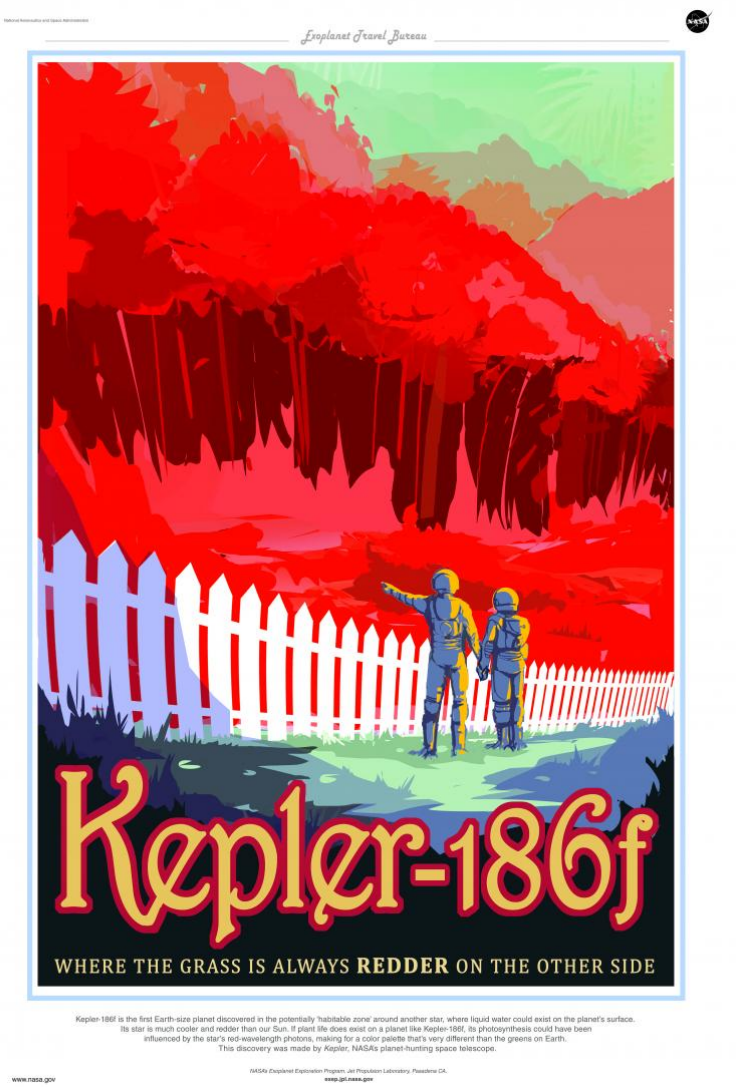
Potentially inhabitable exoplanets



http://phl.upr.edu/hec

SciNet compute • calcul CANADA

NASA exoplanet travel posters



http://planetquest.jpl.nasa.gov/media_categories?category=6

Other exoplanet resources

There are many other resources available for exoplanet lessons.

- `http://gk12.ciera.northwestern.edu/classroom/2013/royster_lesson.pdf`
- `http://space.jpl.nasa.gov`
- `http://exoplanets.org`
- `http://apod.nasa.gov/apod/ap150112.html`

The last one, in particular, is alot of fun.

SciNet compute • calcul CANADA