Exoplanets

PHYS 246 class 3

https://jnoronhahostler.github.io/IntroductionToComputationalPhysics/intro.html

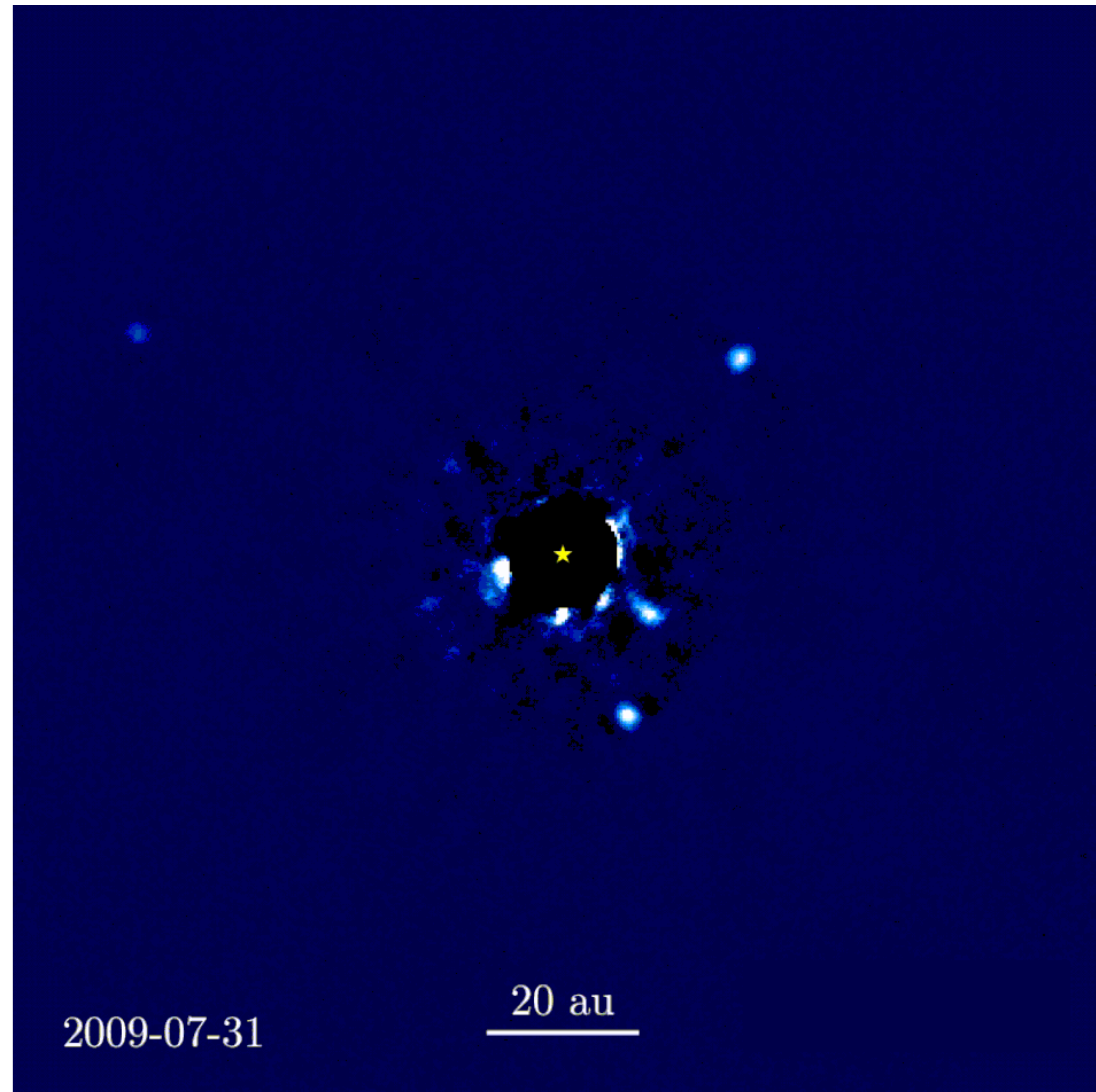
Announcements/notes

- Try to use the provided nbconvert command to generate the PDF, double check to make sure that it generates correctly!
- Please only label the section in your assignment where the answer is
- Always include plot axes (even if they are somewhat trivial)
- Let us know if you have PDF questions in class, but also please check them before submission

Exoplanets

Planets outside of our solar system

- First exoplanet wasn't confirmed until 1992 (although a possible detection occurred back in 1917)
- As of August 2025, there are 5,983 confirmed exoplanets in 4,470 planetary systems with 1,001 systems that have more than one identified exoplanet
- How do we find exoplanets?



Doppler effect

Static Source

Velocity

Car (source) and person (receiver) are both at rest

Frequency

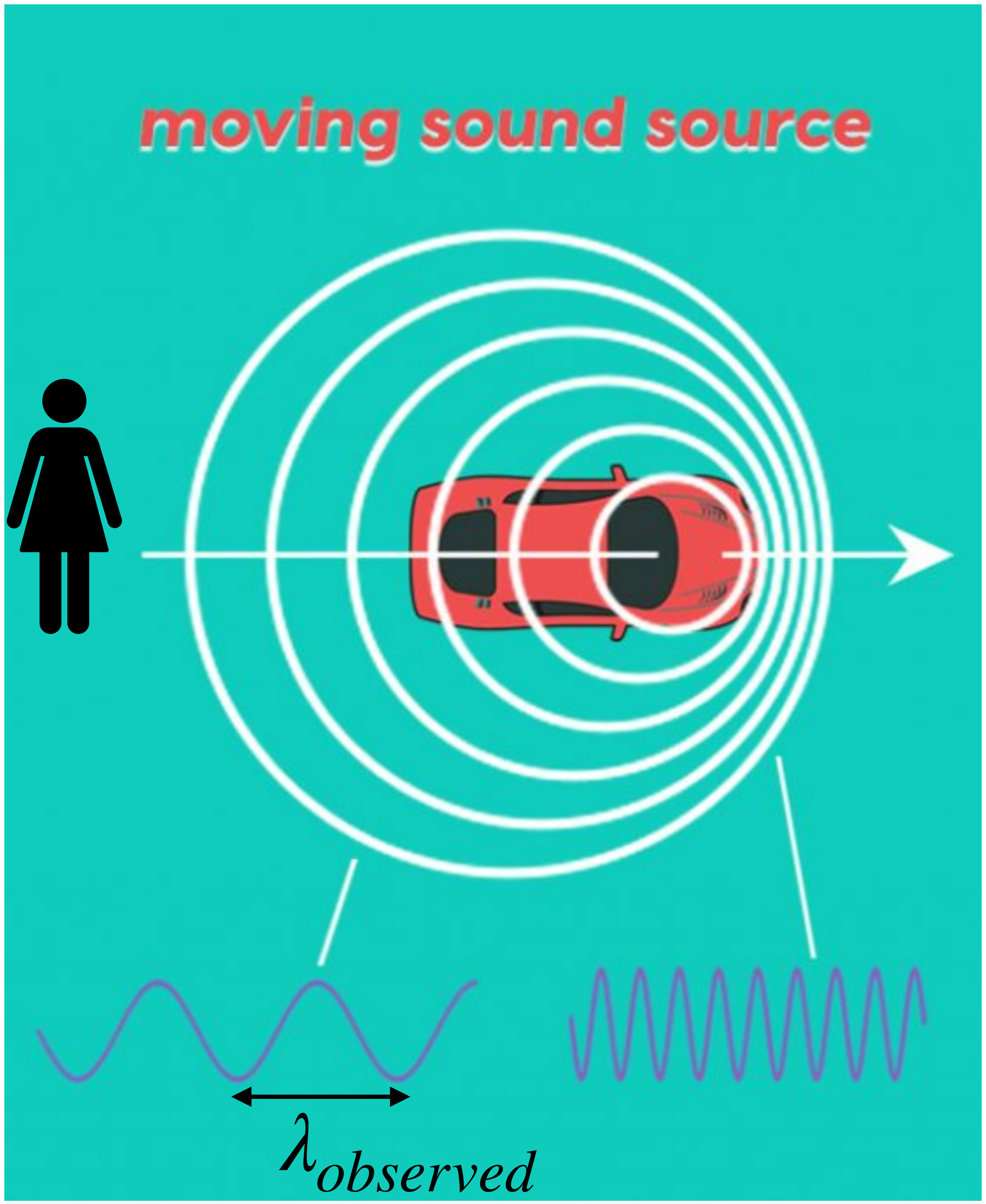
$$f_{source} = \frac{v}{\lambda_{source}}$$

Wave's velocity

Wavelength

Doppler effect

Redshift: $\downarrow f$ **or in other words** $\uparrow \lambda$



$$\lambda_{\text{observed}} > \lambda_{\text{source}}$$

$$\frac{\lambda_{\text{observed}} - \lambda_{\text{source}}}{\lambda_{\text{source}}} = \frac{v_r}{c}$$

$$\frac{f_{\text{source}} - f_{\text{observed}}}{f_{\text{observed}}} = \frac{v_r}{c}$$

When $v_r > 0$, motion away from observer

Doppler effect

Blueshift: $\uparrow f$ **or in other words** $\downarrow \lambda$

$$\lambda_{\text{observed}} < \lambda_{\text{source}}$$

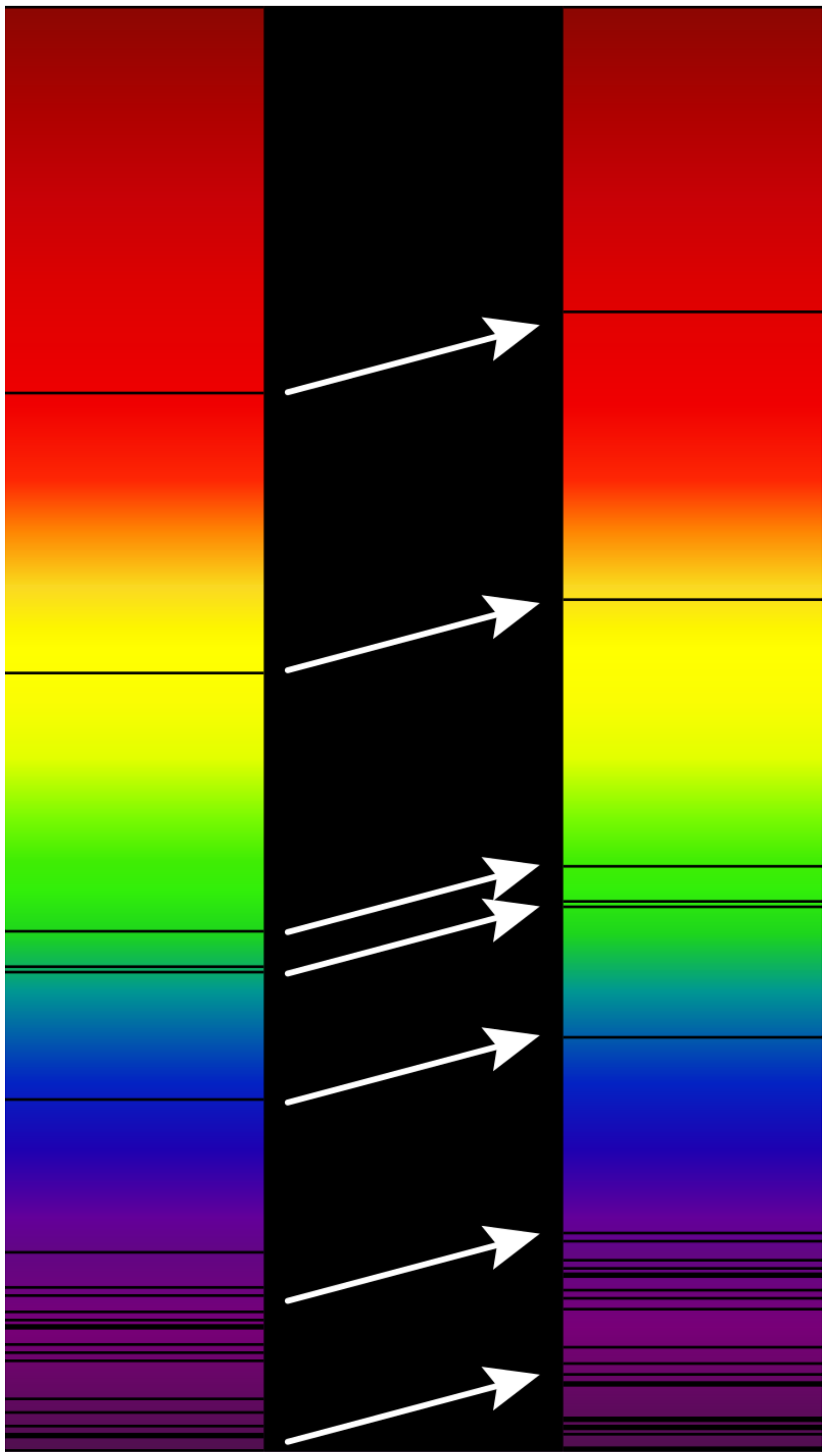
$$\frac{\lambda_{\text{observed}} - \lambda_{\text{source}}}{\lambda_{\text{source}}} = \frac{v_r}{c}$$

$$\frac{f_{\text{source}} - f_{\text{observed}}}{f_{\text{observed}}} = \frac{v_r}{c}$$

When $v_r < 0$, motion towards observer

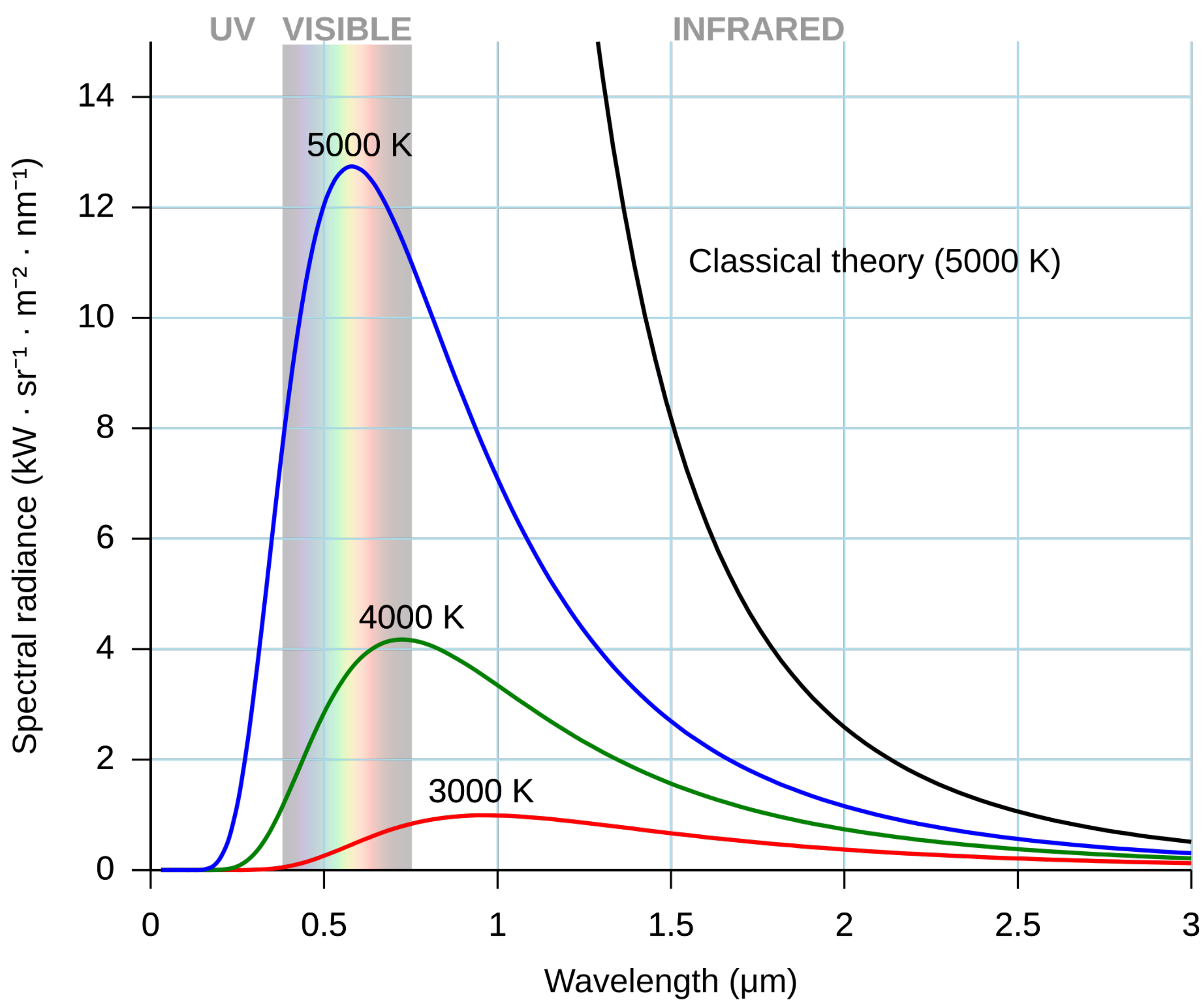
Redshift vs blueshift

Absorption lines of visible spectra



Redshift

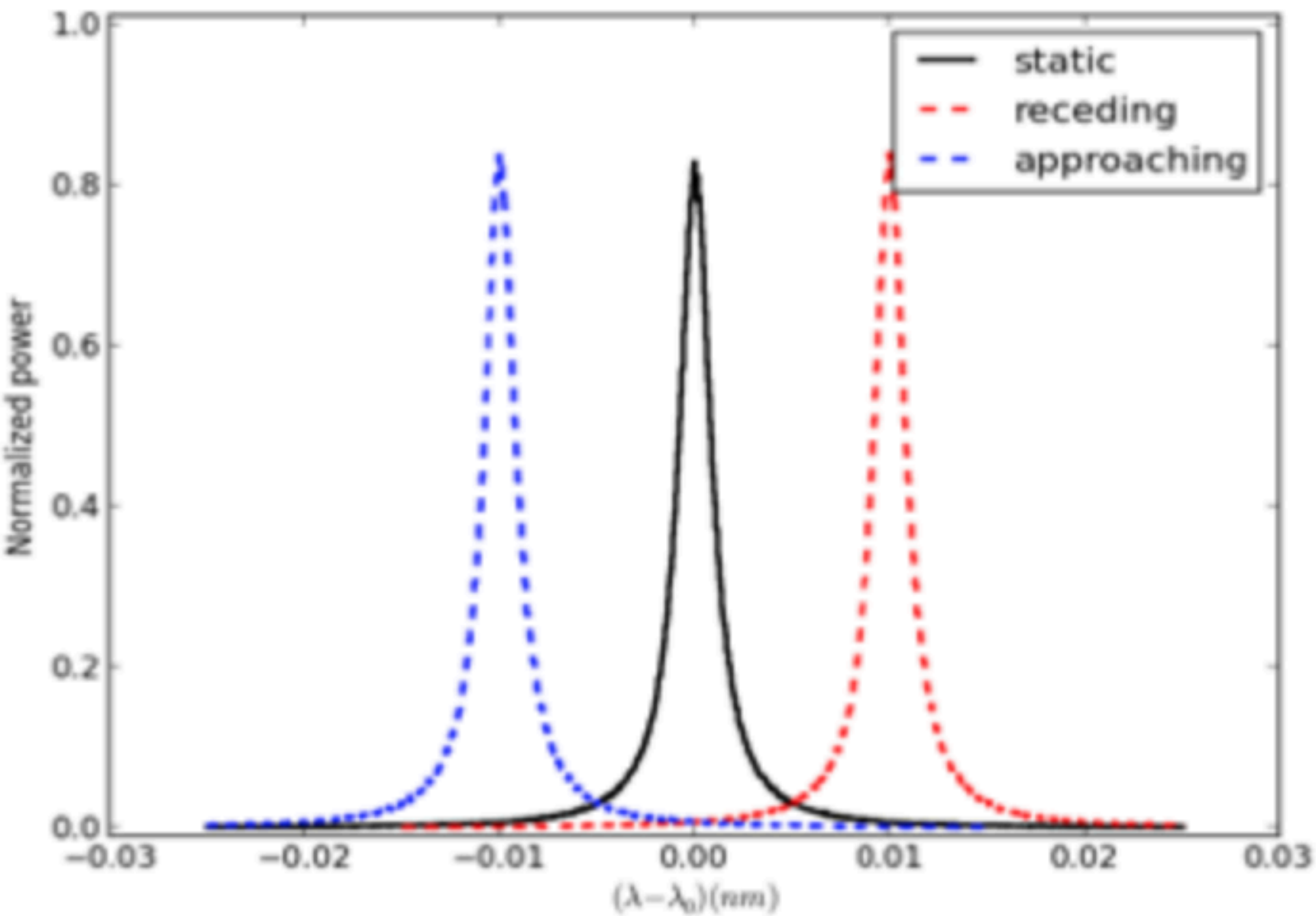
Blackbody radiation



Continuous radiation, depends on T

Spectra lines, indicate elements

Doppler effect: measuring relative velocities



$$\lambda_{observed} = \lambda_{source}\sqrt{\frac{c+v_r}{c-v_r}}$$

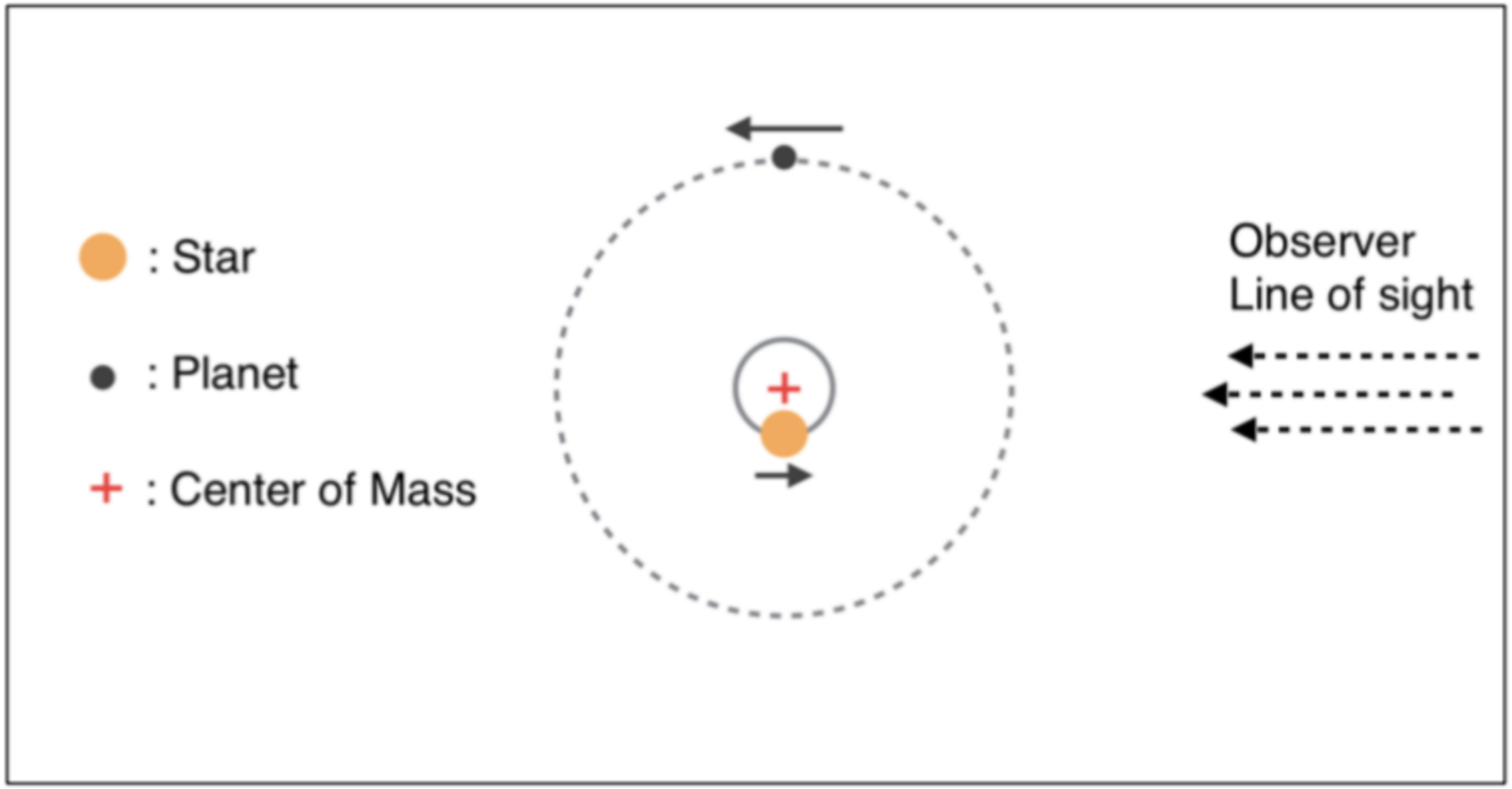
$$\lambda_{observed} \sim \lambda_{source}\left(1+\frac{v_r}{c}\right)$$

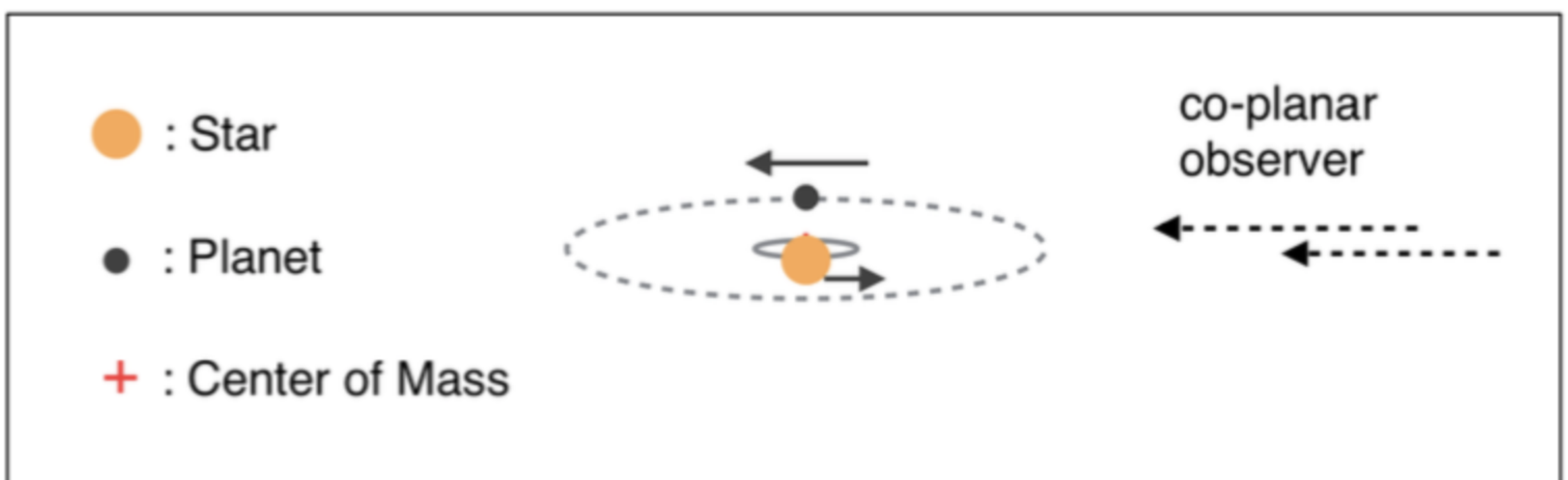
$$\frac{\lambda_{\text{observed}} - \lambda_{\text{source}}}{\lambda_{\text{source}}} = \frac{v_r}{c}$$

This is how we know that the universe is expanding!

Variation in host star's radial velocity

Doppler only sensitive to v_r (line of sight)





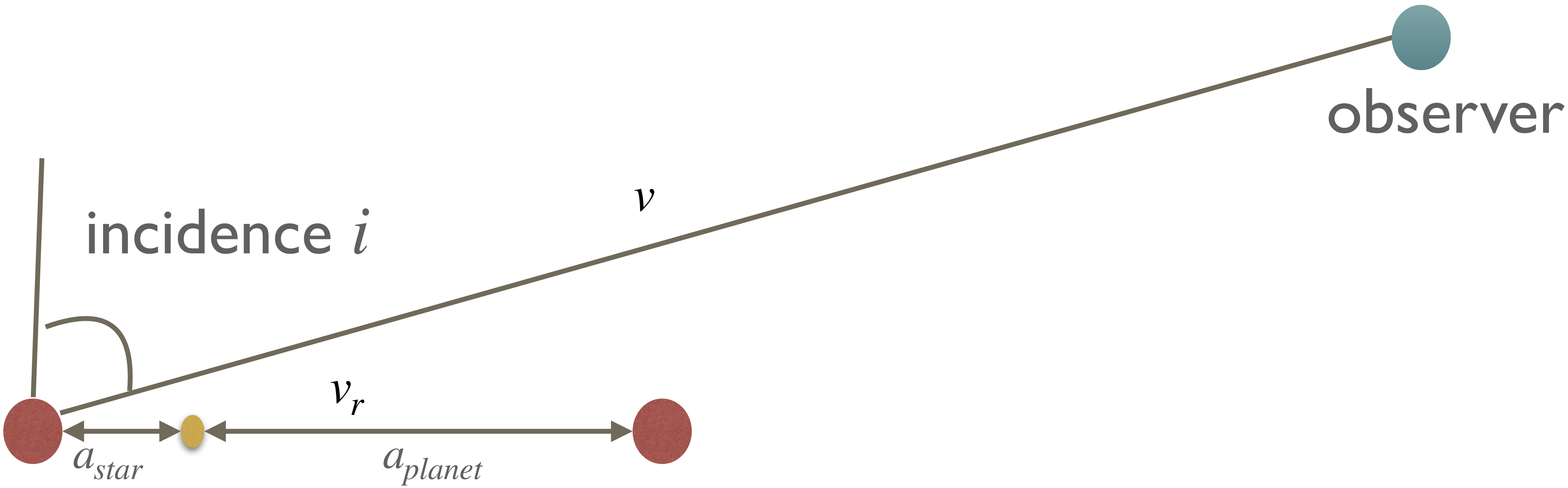
v_r is inner product:

(1) unit vector along observer's line-of-sight

(2) $\vec{v}_{star}$

Radial velocity

Kepler's third law



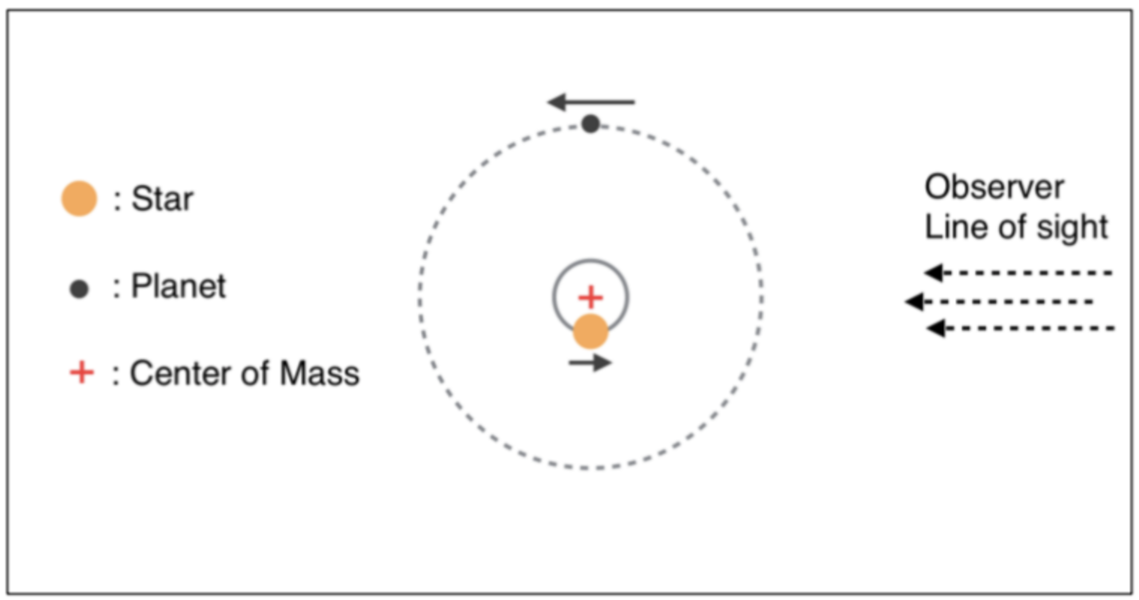
$$Gm_{star} \sim \left(\frac{2pi}{P}\right)^2 a_{planet}^3$$

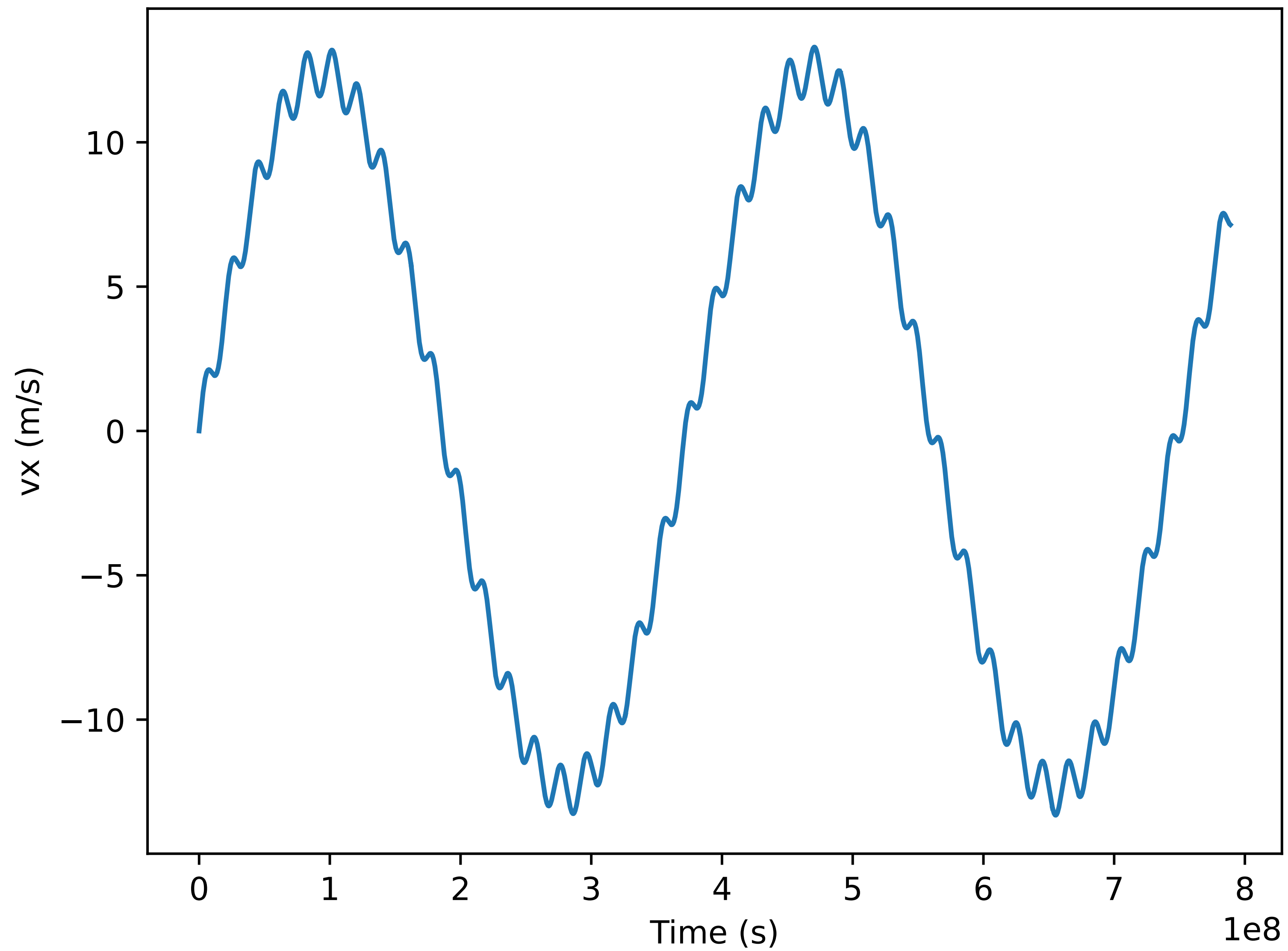
$$v_r = v\sin(i)$$

$$m_{\text{projected}} = \left(\frac{m_{\text{star}}^2 P}{2\pi G}\right)^{1/3} v_r$$

Velocity oscillations in the star

Due to gravity of planet





Measures "stellar wobble" through shifts of the velocity

Fourier Transform

Interested in specific frequencies?

Fourier transform: $f(x)$ input, outputs $\hat{f}(\xi)$ that describes presence of various frequencies in $f(x)$

Fourier Transform

Decomposing signal

Write a general function as

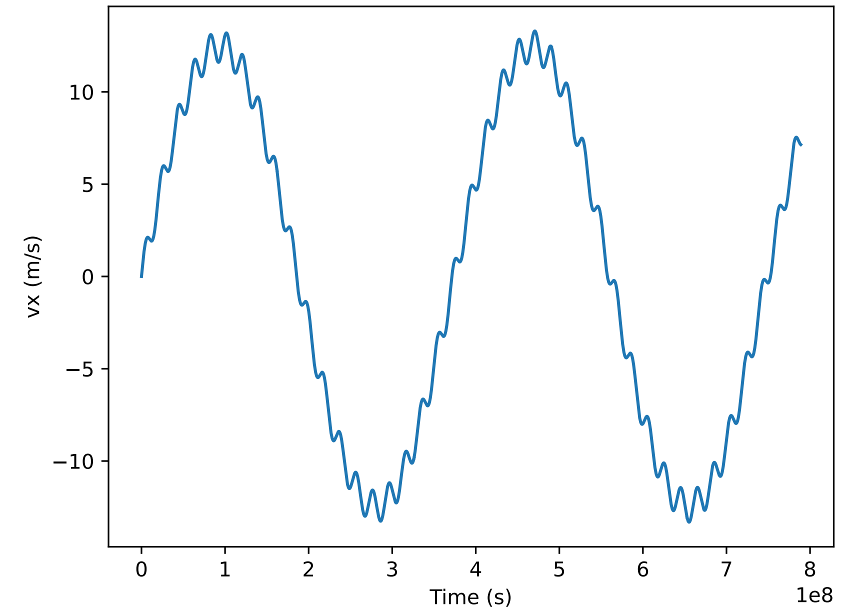
$$f(x) = \sum_k c_k e^{ikx}$$

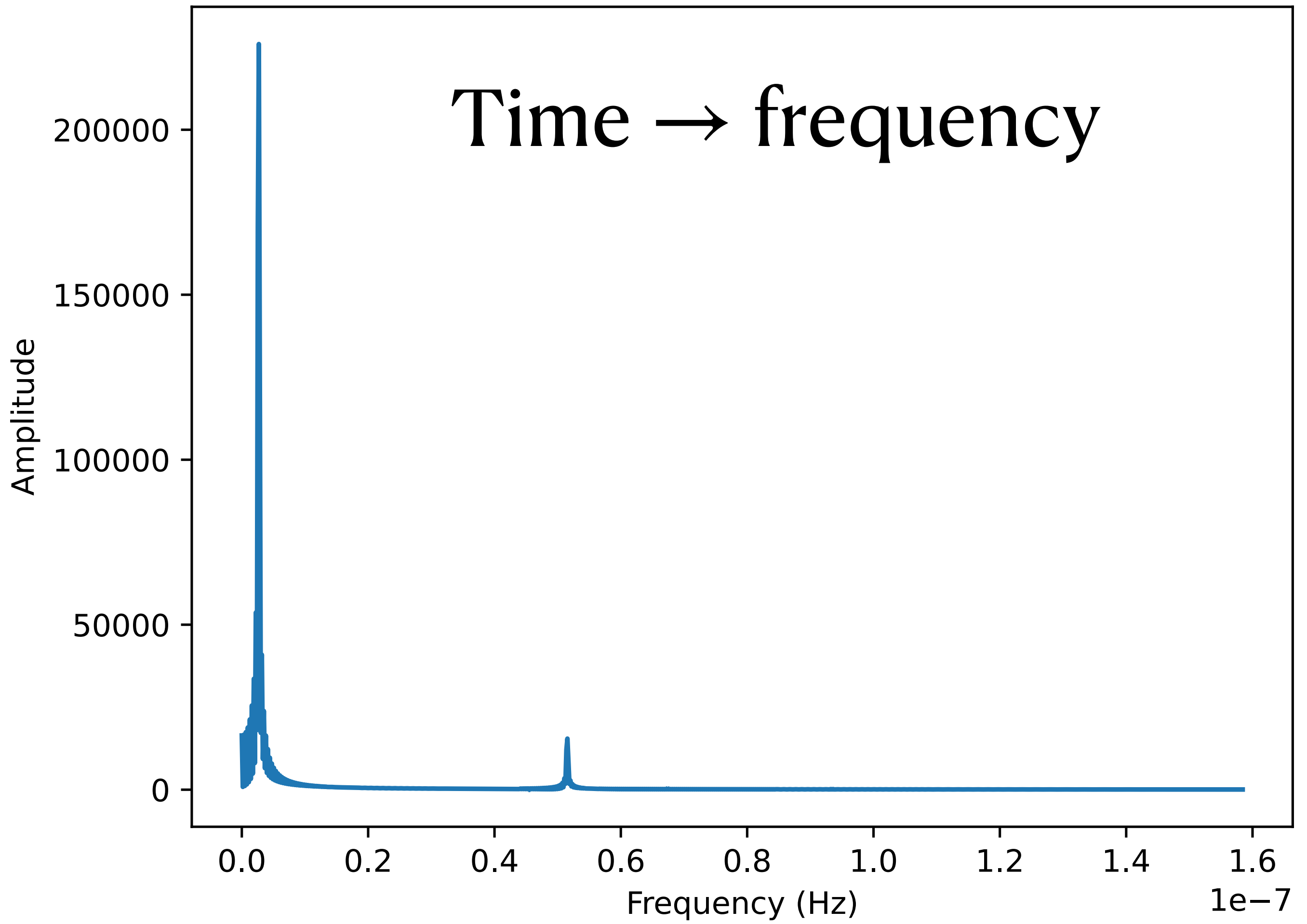
Note that

$$\int e^{-ikx} e^{ik'x} dx = \delta(k - k')$$

So

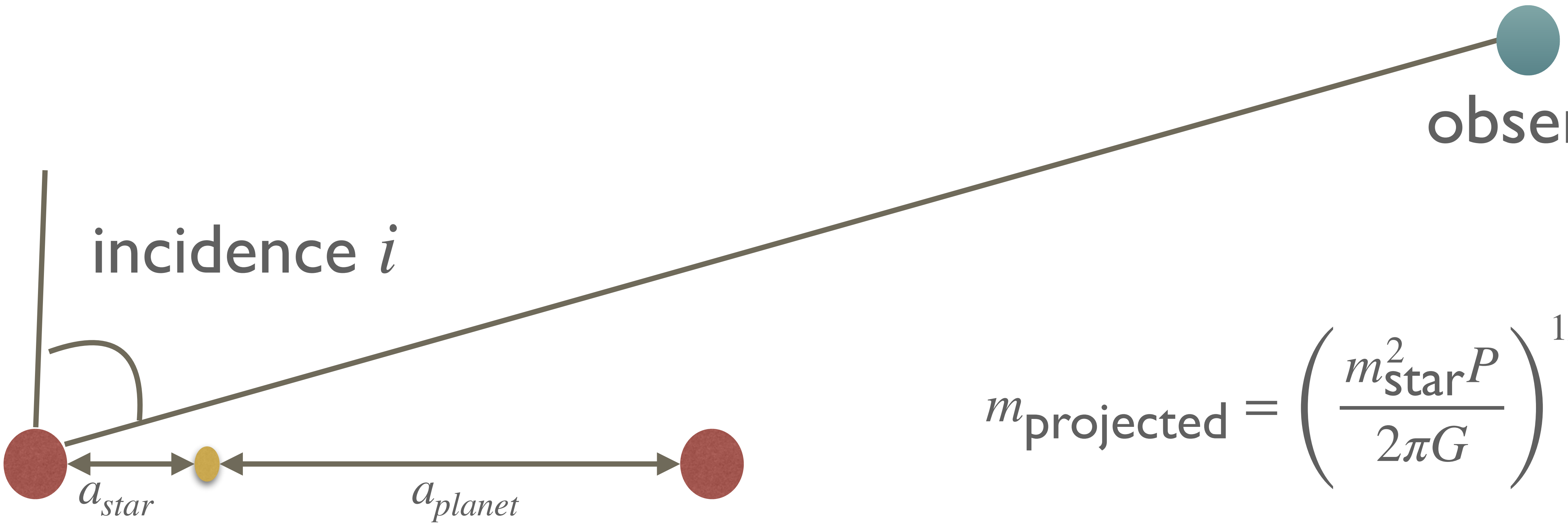
$$c_k = \int e^{-ikx} f(x) dx$$





Radial velocity

Orbital planes of exoplanetary systems not aligned with line of sight



$$m_{\text{projected}} = \left(\frac{m_{\text{star}}^2 P}{2\pi G}\right)^{1/3} v_r$$

1. Estimate the mass of the star from the spectrum. (given here)
2. Use Doppler effect to measure star's velocity.
3. Take the Fourier transform to find the period of the star's velocity P
4. Use the max velocity, mass, and period to estimate the planet's mass.

Coding tricks in Python

Numpy lists continued

- np.newaxis the intent of adding a new axis (new dimension of length 1)
  - Going from a 1D (vector) into 2D (matrix)
- np.linspace generate evenly spaced numbers over specified interval
  - Note: NOT “linespace”, easy typo!
  - xaxislist = np.linspace(0,100,0.5)

```
>>> x= np.arange(0,3)
>>> x
array([0, 1, 2])
>>> y = np.arange(5,9)
>>> y
array([5, 6, 7, 8])
>>> x[:,np.newaxis] + y[np.newaxis,:]
array([[ 5,  6,  7,  8],
       [ 6,  7,  8,  9],
       [ 7,  8,  9, 10]])
>>> 
```

Questions?

- How is a Fourier Transform useful?
-