Hot Spot Phase Shift on Tidally-locked Exoplanets

Jesus Javier Serrano

1-day after pi-day

Exoplanet HD 189733b has a hotspot that is somewhat shifted off it's substella point by about 30 degrees. This offset is a possible indication of Eastward jet stream winds which may transfer radiative energy in the form of thermal kinetic energy away from the substellar point. Breaking down this problem into a one-dimensional distribution function we are able to define the temperature profile of this planet as a function of longitude λ as:

$$\frac{\partial T}{\partial t} + \frac{1}{\tau_{adv}}\frac{\partial T}{\partial \lambda} = \frac{T_{eq}(\lambda) - T}{\tau_{rad}}$$

where T is the temperature at the longitude λ, T_eq is the radiative equilibrium temperature at λ, τ_{rad} is the radiative relaxation timescale, and τ_{adv} is the advection timescale. T_eq may be defined using a linear approximation such that $T_{eq}(\lambda) = T_o + \Delta T\cos(\lambda)$ where T_o is the global mean temperature and ΔT is the temperature difference between dayside and nightside. The advection timescale is defined as $\tau_{adv} = \frac{a}{\bar{U}}$ and the radiative relaxation timescale defined as $\tau_{rad} = \frac{Pc_p}{g\sigma T_o^3}$. The following parameters are defined:

- a = planetary radius
- $\bar{U}$ = mean zonal jet speed
- P = atmospheric pressure of layer with hot-spot shift
- c_p = heat capacity
- g = planetary gravity
- σ = Stefan Boltzmann constant Note that at $\lambda = 0$ the longitudinal position is the substellar point and at $\lambda = \pi$ the longitudinal position is at the anti-stellar point.

For the sake of simplicity, let us build our model by assuming a steady-state system. This defines the temperature profile as being time-independent. If we define a new parameter $\epsilon = \frac{\tau_{rad}}{\tau_{adv}}$ then the steady-state distribution is defined as:

$$\frac{\partial T}{\partial \lambda} = \frac{T_o + \Delta T\cos(\lambda) - T}{\epsilon}$$

This is simply a first-order linear differential equation which can be solved using several different methods. I personally enjoy using Laplace transformations to solve this equation but I'll work in more simpler terms and just apply the inverse product rule. If we recall, the product rule states that:

$$\frac{\partial(u * v)}{\partial \lambda} = \frac{\partial u}{\partial \lambda}v + u\frac{\partial v}{\partial \lambda}$$

Here $u = T$ and v is not well defined. Rearranging some terms we obtain:

$$\frac{\partial u}{\partial \lambda} + u\frac{1}{v}\frac{\partial v}{\partial \lambda} = \frac{1}{v}\frac{\partial(u * v)}{\partial \lambda}$$

Generalizing this equation we may define $\frac{1}{v}\frac{\partial v}{\partial \lambda} = \frac{1}{\epsilon}$ and $\frac{1}{v}\frac{\partial(u*v)}{\partial \lambda} = \frac{T_o + \Delta T\cos(\lambda)}{\epsilon}$. Then, $v = e^{\int \frac{1}{\epsilon}d\lambda} = e^{\frac{\lambda}{\epsilon}}$ and $\int d(T * e^{\frac{\lambda}{\epsilon}}) = \int e^{\frac{\lambda}{\epsilon}}\frac{T_o + \Delta T\cos(\lambda)}{\epsilon}d\lambda$. This leaves us with the solution:

$$T = e^{-\frac{\lambda}{\epsilon}}\int e^{\frac{\lambda}{\epsilon}}\frac{T_o + \Delta T\cos(\lambda)}{\epsilon}d\lambda + e^{-\frac{\lambda}{\epsilon}}C_o$$

where C_o is a constant that is yet to be determined. Solving the integral and simplying the equation with algebra we find that:

$$T(\lambda) = T_o + \Delta T \frac{1}{\epsilon^2+1}(\epsilon\sin(\lambda) + \cos(\lambda)) + e^{-\frac{\lambda}{\epsilon}} C_o$$

Please look into the scratch paper for the full derivation of this equation. If we state that at $\lambda = 0$ the temperature at the substellar point is T_i then we find that $C_o = T_i - T_o - \frac{\Delta T}{\epsilon^2+1}$. Before we go any further, we would like to simplify the trigonemetric functions within the temperature profile as much as possible. Thus, let us define $(\epsilon\sin(\lambda) + \cos(\lambda))$ as the inner product of two $\mathbb{R}^2$ cartesian-space vectors. Then:

$$(\epsilon\sin(\lambda) + \cos(\lambda)) = (1, \epsilon) \circ (\cos(\lambda), sin(\lambda))$$

Since we are treating the scalar components of the trig. function as a vector then we can break down the vector into a scalar and unit-vector. This lets us define:

$$(1, \epsilon) = \sqrt{\epsilon^2+1} * \left(\frac{1}{\sqrt{\epsilon^2+1}}, \frac{\epsilon}{\sqrt{\epsilon^2+1}}\right)$$

where $\frac{1}{\sqrt{\epsilon^2+1}} = \cos(\phi)$ and $\frac{\epsilon}{\sqrt{\epsilon^2+1}} = \sin(\phi)$ with $\phi = \arctan(\frac{\epsilon}{1})$. So:

$$\epsilon\sin(\lambda) + \cos(\lambda) = \sqrt{\epsilon^2+1}\,(\sin(\phi)\sin(\lambda) + \cos(\phi)\cos(\lambda)) = \sqrt{\epsilon^2+1}\cos(\lambda - \phi)$$

This leaves us with the equation:

$$T(\lambda) = T_o + \Delta T \frac{1}{\sqrt{\epsilon^2+1}} \cos(\lambda - \phi) + e^{-\frac{\lambda}{\epsilon}} C_o$$

The temperature rate in respect to the longitude position is defined as:

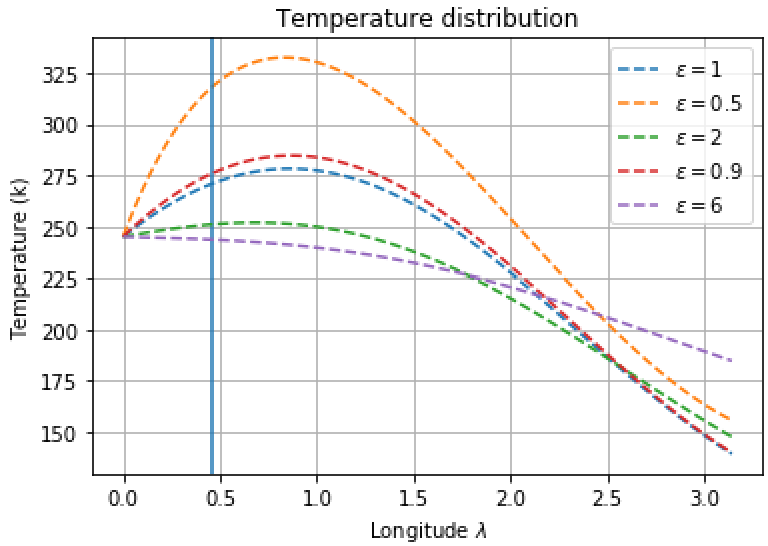
$$\frac{\partial T(\lambda)}{\partial \lambda} = -\Delta T \frac{1}{\sqrt{\epsilon^2+1}} \sin(\lambda - \phi) - \frac{1}{\epsilon} e^{-\frac{\lambda}{\epsilon}} C_o$$

The hot spot must be defined as the maximum of the temperature distribution profile, which is where the temperature rate is zero. At the hot spot we see that:

$$\sin(\lambda - \phi) e^{\frac{\lambda}{\epsilon}} = -\frac{\sqrt{\epsilon^2+1}}{\Delta T} \frac{1}{\epsilon} C_o$$

where in this case λ is the phase shift. Notice that if we observe the first order approximation of small phase shifts λ then we may determine that as ϵ increases then the closer λ approaches a quantity proportional to $\arccos(\frac{1}{\Delta T})$. This also describes the relation between ΔT and λ. Since $\epsilon \propto \bar{U}$ then as $\bar{U}$ increases, the phase shifts converges closer to $\arccos(\frac{1}{\Delta T})$ as well. Note that for very small $\arccos(\frac{1}{\Delta T})$, the phase shift is apparent for larger and larger ϵ values on the temperature profiles shown below.

In [17]:
```
import numpy as np
import matplotlib.pyplot as plt

#parameters needed
init_ep = 1
delt_temp = 120 #kelvin
prim_T_o = 200 #kelvin
subst_T = 245 #kelvin

def T_o(eps):
    return ((((prim_T_o)**3)*init_ep)/eps)**0.3333333333333333

def const(eps):
    return subst_T - T_o(eps) - (delt_temp/(eps**2 + 1))

def inv_sqrt(eps):
    return (1/((eps**2) + 1))**0.5

def phi(eps):
    return np.arctan(eps)

def T(eps, lam):
    return T_o(eps) + (delt_temp*inv_sqrt(eps)*np.cos(lam - phi(eps))) + (cons
t(eps)*np.exp(-lam/eps))

lam_dom = np.linspace(0, np.pi, 100)
frst_temp_prof = T(np.repeat(1, len(lam_dom)), lam_dom)
sec_temp_prof = T(np.repeat(0.5, len(lam_dom)), lam_dom)
thr_temp_prof = T(np.repeat(2, len(lam_dom)), lam_dom)
fr_temp_prof = T(np.repeat(0.9, len(lam_dom)), lam_dom)
ft_temp_prof = T(np.repeat(6, len(lam_dom)), lam_dom)

plt.plot(lam_dom, frst_temp_prof, '--', label = r'$\epsilon = 1$')
#plt.axvline(x = np.arctan(1))
plt.plot(lam_dom, sec_temp_prof, '--', label = r'$\epsilon = 0.5$')
plt.axvline(x = np.arctan(0.5))
plt.plot(lam_dom, thr_temp_prof, '--', label = r'$\epsilon = 2$')
#plt.axvline(x = np.arctan(2))
plt.plot(lam_dom, fr_temp_prof, '--', label = r'$\epsilon = 0.9$')
#plt.axvline(x = np.arctan(0.9))
plt.plot(lam_dom, ft_temp_prof, '--', label = r'$\epsilon = 6$')
#plt.axvline(x = np.arctan(6))
plt.grid(True)
plt.title('Temperature distribution')
plt.xlabel(r'Longitude $\lambda$')
plt.ylabel('Temperature (k)')
plt.legend()
plt.show()
```



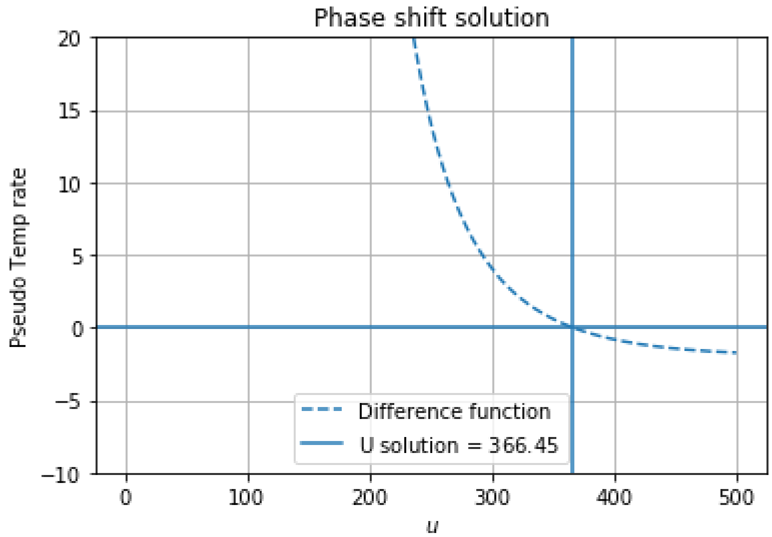
Suppose we know the parameters for this planet and $T_i = T_o$ and thus $C_o = -\frac{\Delta T}{\epsilon^2+1}$. So the phase shift equation is:

$$\sin(\lambda - \phi)e^{\frac{\lambda}{\epsilon}} = \frac{1}{\sqrt{\epsilon^2+1}}\frac{1}{\epsilon}$$

If we use Newton's method of finding roots then we may solve this equation given λ as the phase shift of the hot-spot as 30° and $\epsilon = \frac{Pc_p\bar{U}}{ag\sigma T_o^3}$ where all parameters are known except for mean zonal jet speed $\bar{U}$, then we can determine $\bar{U}$. Newton's algorithm would be fun to do, but for the sake of time a graphing calculator was used to determine the zonal jet speed $\bar{U} = 366.45\frac{m}{s} = 1319.22\frac{km}{hr}$. This is about the same speed of the Jupiter zonal jet speed which is about $1200\frac{km}{hr}$.

In [49]:
```
import numpy as np
import matplotlib.pyplot as plt

#parameters needed
a = 79000e3 #m
P = 50e2 #Pa
init_T = 1100 #K
c_p = 1.3e4 #J kg^-1 K^-1
g = 21 #m s^-2
sig_bol = 5.670367e-8 #W m^-2 K^-4
phase_shft = 30*np.pi/180 #radians
cnst = (a*g*sig_bol*(init_T**3))/(P*c_p)

def epsl(u):
    return u/cnst

def left_side(u):
    return ((1/((epsl(u)**2) + 1))**0.5)*(1/epsl(u))

def phi_st(u):
    return np.arctan(epsl(u))

def right_side(u, lam):
    trig = np.sin(lam - phi_st(u))
    expn = np.exp(lam/epsl(u))
    return trig*expn


pos_u = np.linspace(0.1, 500, 1000)
left_int = left_side(pos_u)
right_int = right_side(pos_u, np.repeat(phase_shft, len(pos_u)))

#plt.plot(pos_u, left_int, '--', label = r'left side')
#plt.plot(pos_u, right_int , '--', label = r'right side')
plt.plot(pos_u, (right_int - left_int), '--', label = 'Difference function')
plt.axvline(x = 366.45, label = r'U solution = $366.45$')
plt.axhline(y = 0)
plt.grid(True)
plt.title('Phase shift solution')
plt.xlabel(r'$u$')
plt.ylabel('Pseudo Temp rate')
plt.ylim(-10, 20)
plt.legend()
plt.show()
```

```
C:\Users\Jesus\Anaconda3\lib\site-packages\ipykernel_launcher.py:25: RuntimeW
arning: overflow encountered in exp
```



In [ ]: