

```

import numpy as np
import matplotlib.pyplot as plt

### this function solves the integral (sum) for the length of the parker spiral #####
def integrate_parker(dr,u,r0=0,r1=1.5e8):
    # dr: steplength in km
    # u: solar wind speed in km/s
    # r0, r1: start and end of integration interval in km
    omega=1.7e-4/360*2*np.pi # rad/s
    counter=0
    l=0
    while r0+(counter+1)*dr<r1:
        current_r=r0+(counter+0.5)*dr
        l+=np.sqrt(1+(omega*current_r/u)**2)*dr
        counter+=1
    rest_dr=r1-r0-counter*dr
    current_r=r1-rest_dr/2.
    l+=np.sqrt(1+(omega*current_r/u)**2)*rest_dr
    return l
#####

### this function derives the proton/electron velocity (m/s) as function of its kinetic energy (MeV)
def energy_to_velocity(ekin,particle="proton"):
    c=3e8 #m/s
    if particle=="proton": e0=938. #in MeV
    elif particle=="electron": e0=0.511 #in MeV
    v=np.sqrt(1-(1/(ekin/e0+1)**2))
    return v*c
#####

### plot length of parker spiral as function of solar wind speed #####
plt.figure()
radial_distances=[0.7,1,1.52]
names=["Venus","Earth","Mars"]
colors=["g","b","r"]
# loop over three different planets (radial distances)
for w in range(len(radial_distances)):
    trmax=radial_distances[w]
    tname=names[w]
    tcolor=colors[w]
    sw=np.linspace(300,800,50)
    ls=np.zeros(len(sw))
    # loop over different solar wind speeds
    for q in range(len(sw)):
        print "Calculating for ",tname," vsw=",sw[q],"km/s..."
        ls[q]=integrate_parker(1e5,sw[q],r1=trmax*1.5e8)/1.5e8

    plt.plot(sw,ls,label=tname,color=tcolor)
    plt.plot(sw,np.ones(len(sw))*trmax,'--',color=tcolor)
plt.grid()
plt.xlabel("solar wind speed / km/s")
plt.ylabel("length of Parker spiral / AU")
plt.legend()
plt.savefig("plot_parker_spiral_vsw.pdf")
#####

### plot travel time as function of c/v #####
plt.figure()
c=3e8 #m/s
v=np.linspace(1e8,3e8,100)
colors=["g","b","r"]
vsws=[300,500,700]
# loop over different solar wind speeds

```

```

for q in range(len(vsws)):
    tvsw=vsws[q]
    tcolor=colors[q]
    ls=integrate_parker(1e5,tvsw,r1=1.0*1.5e8)*1000. # now in units of m/s
    time=ls/v # in units of s
    time/=60 # now in unites of minutes
    plt.plot(c/v,time,color=tcolor,label="vsw=%i km/s"%tvsw)
plt.xlabel("c/v")
plt.ylabel("travel time / minutes")
plt.legend()
plt.grid()
plt.savefig("plot_travel_time_cv.pdf")
#####

### plot travel time as function of kinetic energy #####
plt.figure()
c=3e8 #m/s
ekin=np.logspace(-1,4,100) # from 1 MeV to 1 GeV
v=np.zeros(len(ekin))
v_e=np.zeros(len(ekin))
# loop over different kinetic energies
for q in range(len(ekin)):
    v[q]=energy_to_velocity(ekin[q])
    v_e[q]=energy_to_velocity(ekin[q],particle="electron")
    colors=["g","b","r"]
    vsws=[300,500,700]
# loop over different solar wind speeds
for q in range(len(vsws)):
    tvsw=vsws[q]
    tcolor=colors[q]
    ls=integrate_parker(1e5,tvsw,r1=1.0*1.5e8)*1000. # now in units of m/s
    time=ls/v # in units of s
    time/=60 # now in unites of minutes
    time_e=ls/v_e # in units of s
    time_e/=60 # now in unites of minutes
    plt.plot(ekin,time,"-",color=tcolor,label="protons - vsw=%i km/s"%tvsw)
    plt.plot(ekin,time_e,"--",color=tcolor,label="electrons - vsw=%i km/s"%tvsw)
plt.xscale("log")
plt.yscale("log")
plt.ylim(7,1e3)
plt.xlabel("kinetic energy / MeV")
plt.ylabel("travel time / minutes")
plt.legend()
plt.grid()
plt.savefig("plot_travel_time_energy.pdf")
#####

```