PlanetsHW3

November 3, 2020

1 ASTR 5490 Homework 3

[6]:
```
# Import relevant modules/packages
import numpy as np
import matplotlib.pyplot as plt
from astropy import units as u
from astropy import constants as const
from astropy.timeseries import LombScargle
from PeriodicityTools import Periodicity
from GrazingTransit import GrazingTransit
from LimbDarkening import LimbDarkening
from Spectra import SpectralFeatures, Accuracy
from MathTools import Gaussian, NonRelDoppler
from TransitPlotter import LightCurveCompare
from Batman import BatmanModel

# Reload PeriodicityTools to acknowledge changes made in the script
%load_ext autoreload
%autoreload 2
```

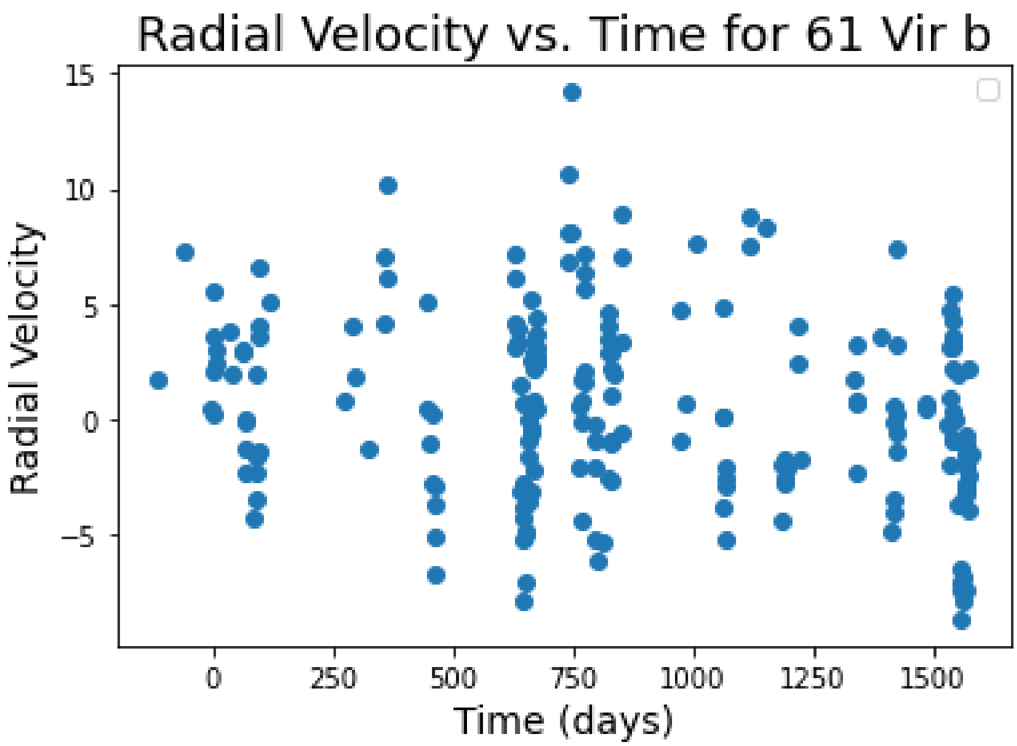
```
The autoreload extension is already loaded. To reload it, use:
  %reload_ext autoreload
```

2 1) Exploring Radial Velocity Curve of 61 Vir b from Vogt et al. (2009)

2.0.1 https://iopscience.iop.org/article/10.1088/0004-637X/708/2/1366#apj330338s2

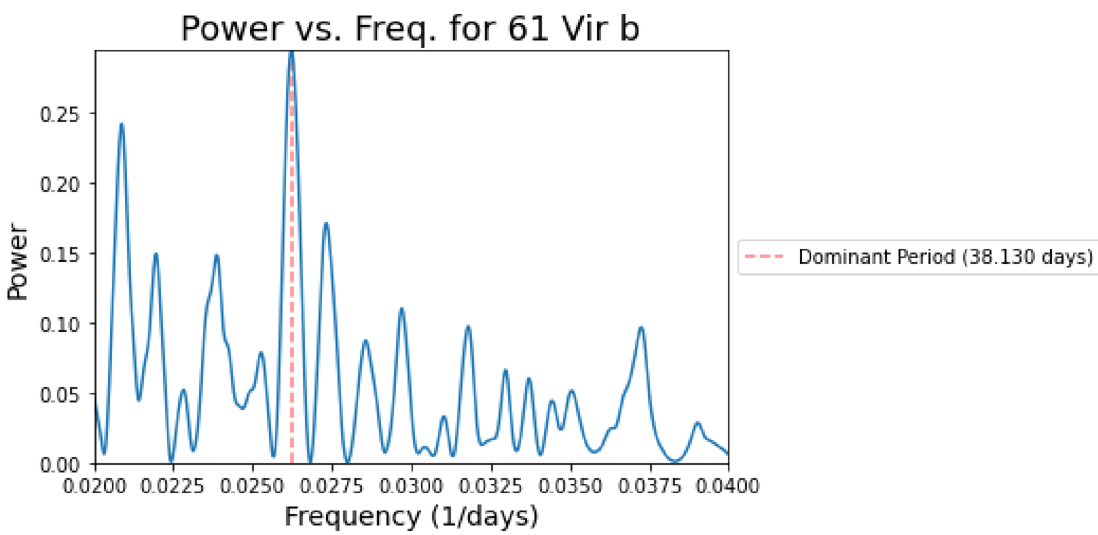
[9]:
```
# Create instance of class
Virb = Periodicity('Vogt2009_61Virb_vels.dat','61 Vir b',None)

# Generate light curve for data
times,RVs,errors = Virb.LightCurve(curve='Radial Velocity')
```

```
No handles with labels found to put in legend.
```



2.1 1a) Make a periodogram of the whole dataset

[45]:
```
# Make a Lomb Scargle Periodogram of the data
original_maxPower = Virb.LS(25,50,1000,1,flux=[],plot=True)
```



2.1.1 Looks quite similar to upper panel of Vogt et al. Figure 3, but my x-axis is the inverse of their's so mine's not exactly the same. I find the same as they do (38.13 d)

2.2 1b) Compute the false alarm probability considering the 38 day planet

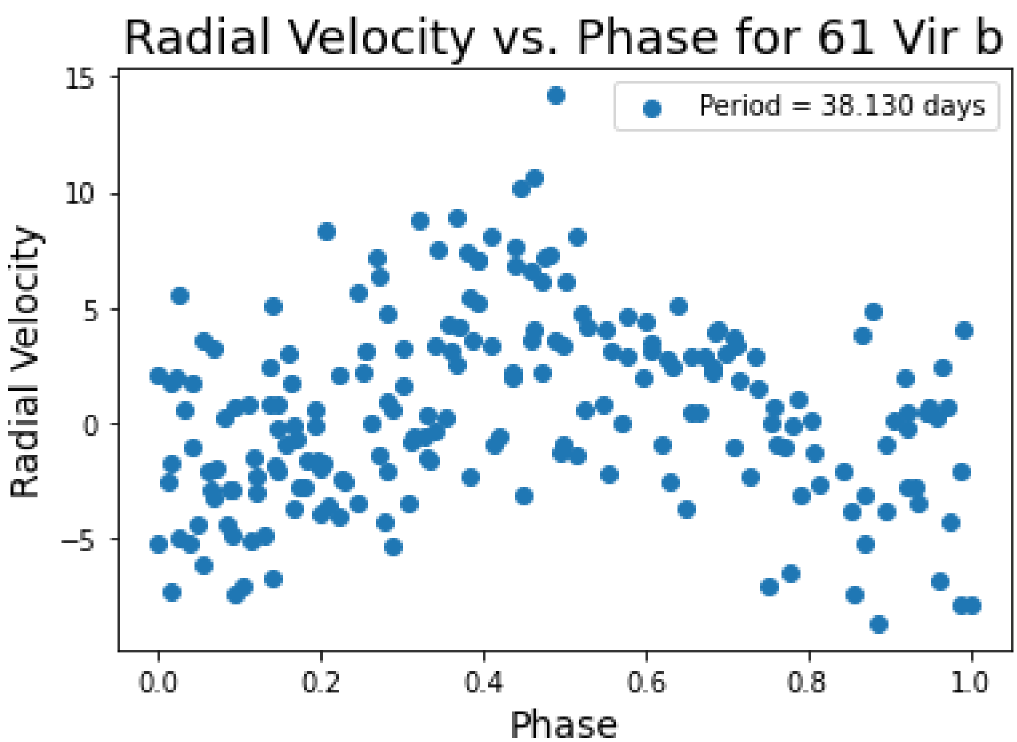
2.2.1 The F.A.P. is defined as "the fraction of trials for which the periodogram power exceeds the observed value" (Page 3 of Cumming 2004 - https://academic.oup.com/mnras/article/354/4/1165/1052087)

[15]:
```
fap = Virb.FAP(10000)
```

```
FAP = 0.000e+00
```

2.3 1c) Fold the RV data at the period determined from the periodogram

[47]:
```
Virb_folded = Periodicity('Vogt2009_61Virb_vels.dat','61 Vir b',None,period=38.
 ↪13)
new_times,new_RVs,new_errors = Virb_folded.
 ↪LightCurve(xaxis='Phase',curve='Radial Velocity')
```



2.4 1d) Pick a ________ interval and find the FAP

2.4.1 i) contiguous 100d

[13]:
```
# Find FAP over first 100 days of data
Virb_contig = Periodicity('Vogt2009_61Virb_vels.dat','61 Vir b',100,period=38.
 ↪13)
contiguous_fap = Virb_contig.FAP(10000)
```

```
FAP = 0.000e+00
```

2.4.2 ii) sparsed 100d

[14]:
```
# Find FAP using first 50 and last 50 days of data
Virb_sparsed = Periodicity('Vogt2009_61Virb_vels.dat','61 Vir␣
 ↪b',50,contiguous=False,period=38.13)
sparsed_FAP = Virb_sparsed.FAP(10000)
```

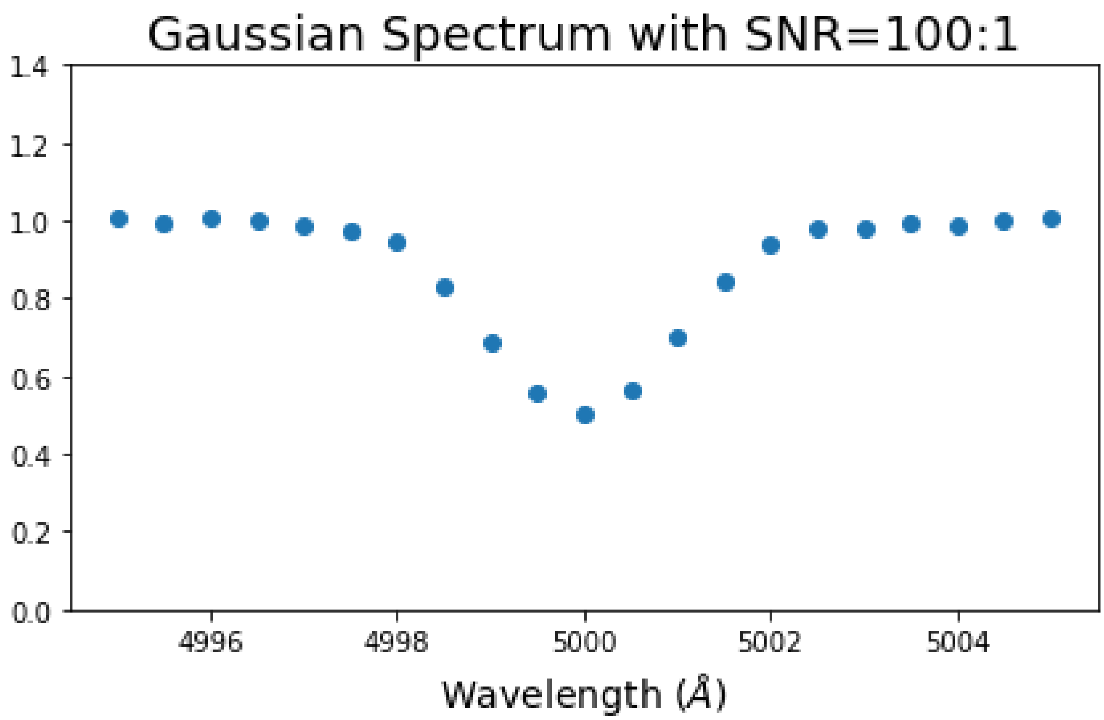
```
FAP = 0.000e+00
```

2.4.3 My calculated FAP for each method of splicing the data was 0 and I ran them all for 10000 Monte Carlo iterations so the FAP must be $\leq 10^{-4}$. This indicates a high probability ($\geq 99.99\%$) of this being a true planet detection

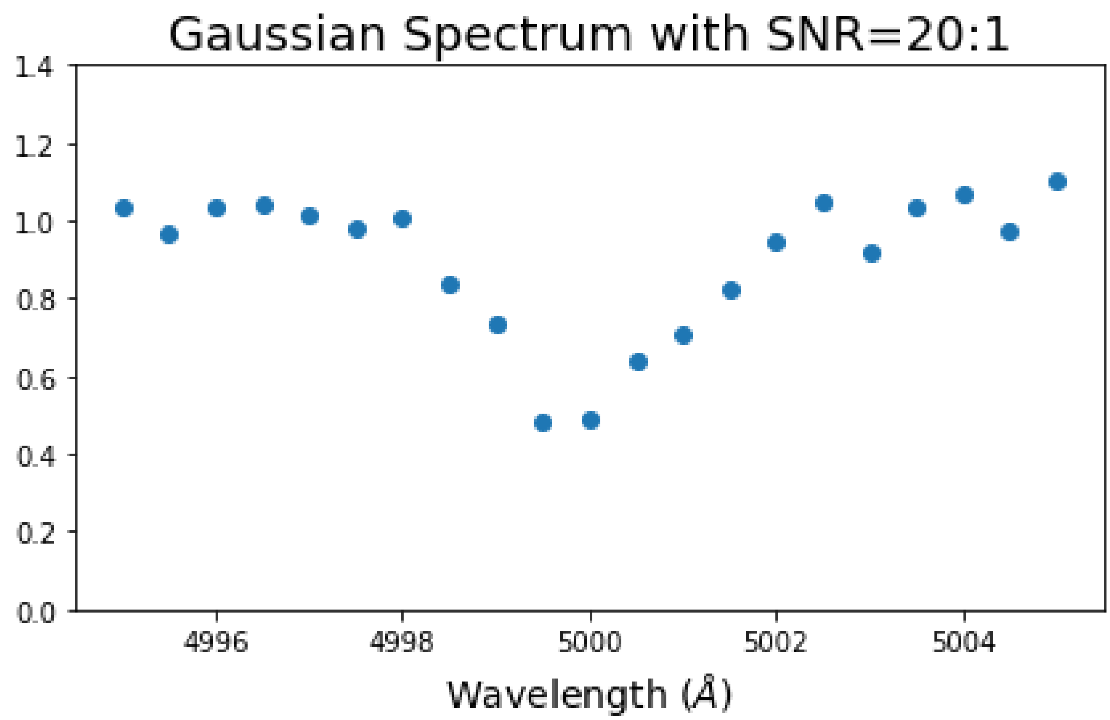
3 2) Investigating Gaussian absorption lines

3.1 2a) Generate Gaussian absorption line for normalized spectrum with: $A = 0.5$, $\sigma = 1$ angs. centered at 5000 angs. with pixels every 0.5 angs.

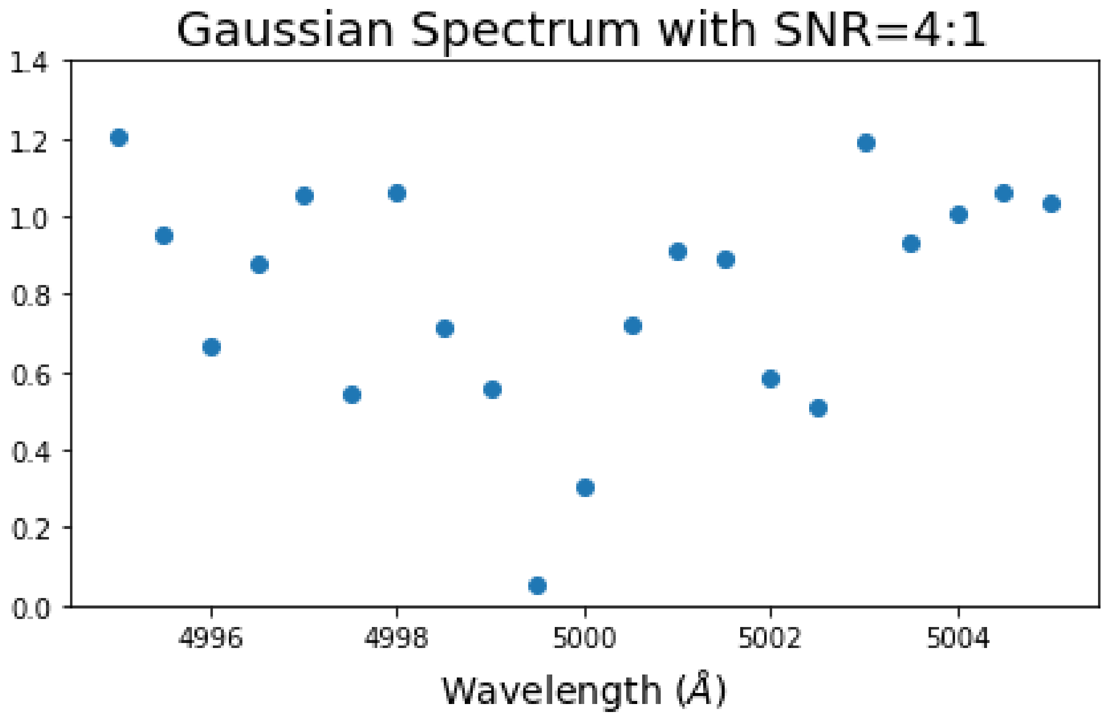
[3]:
```
# Gaussian spectrium with SNR=100:1
SNR_100 = SpectralFeatures(5000.0,1.0,0.5,0.0,1.0,0.5,100)
x,y_100 = SNR_100.GaussianNoise()
```



[4]:
```
# Gaussian spectrium with SNR=20:1
SNR_20 = SpectralFeatures(5000.0,1.0,0.5,0.0,1.0,0.5,20)
x,y_20 = SNR_20.GaussianNoise()
```

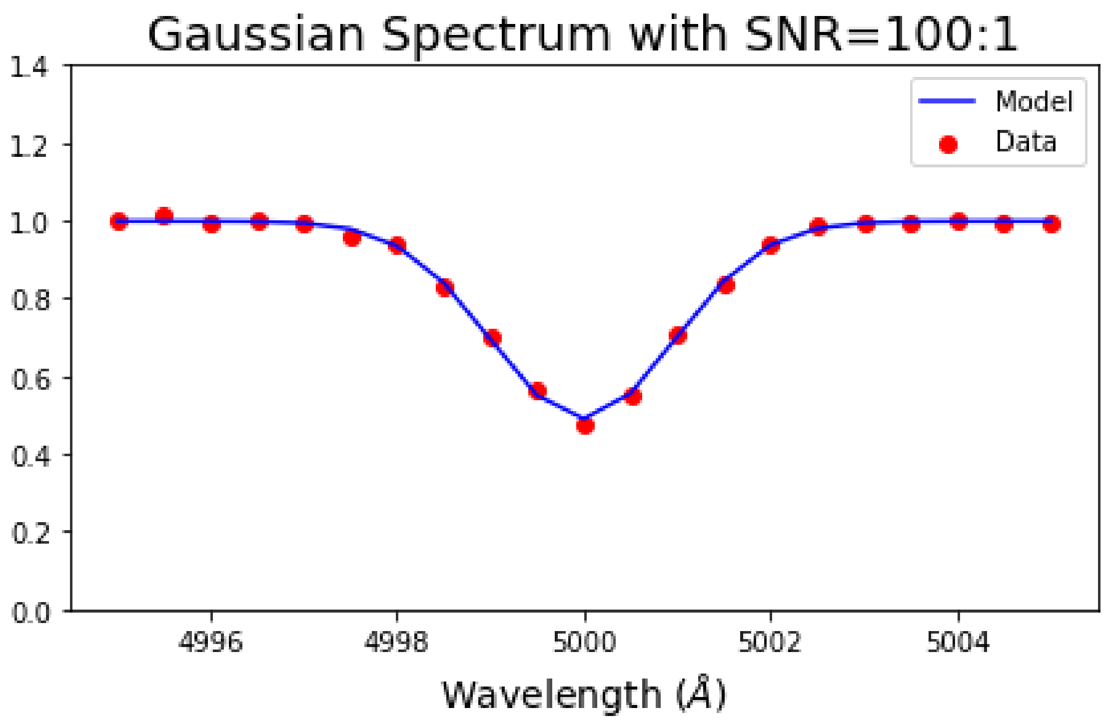


[5]:
```
# Gaussian spectrium with SNR=4:1
SNR_4 = SpectralFeatures(5000.0,1.0,0.5,0.0,1.0,0.5,4)
x,y_4 = SNR_4.GaussianNoise()
```

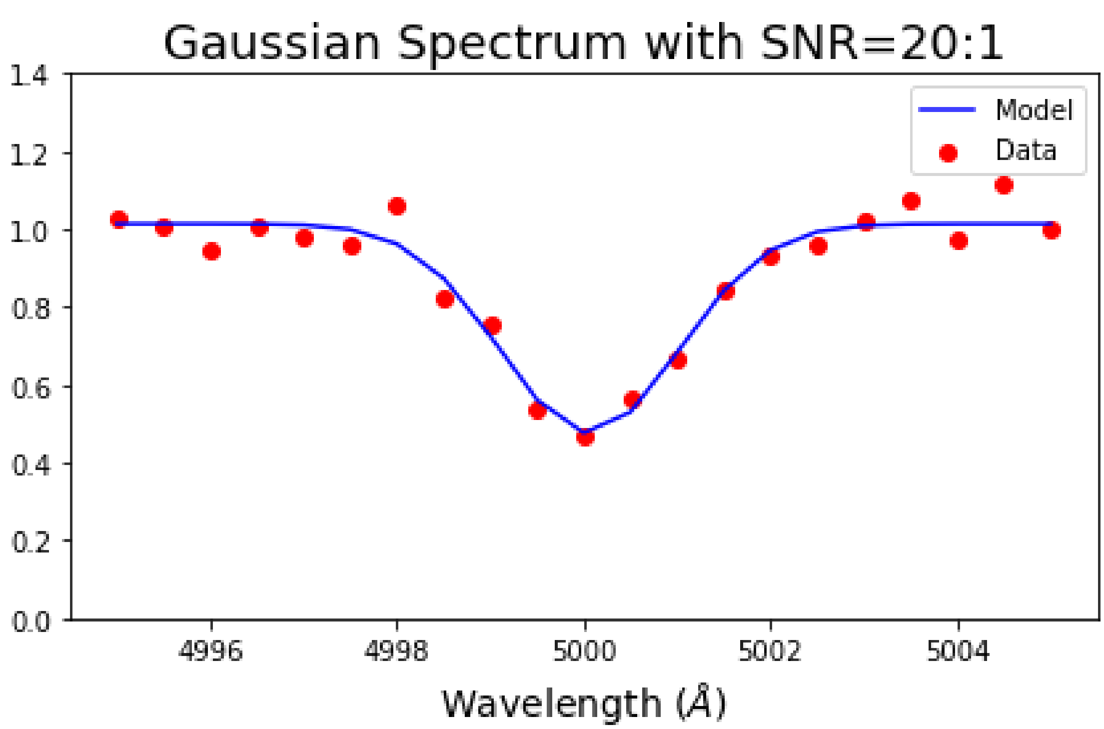


3.2 2a) Use curve_fit to fit Gaussian to noisy data from 2a. Plot accuracy of the center of the profile (mean) vs. SNR

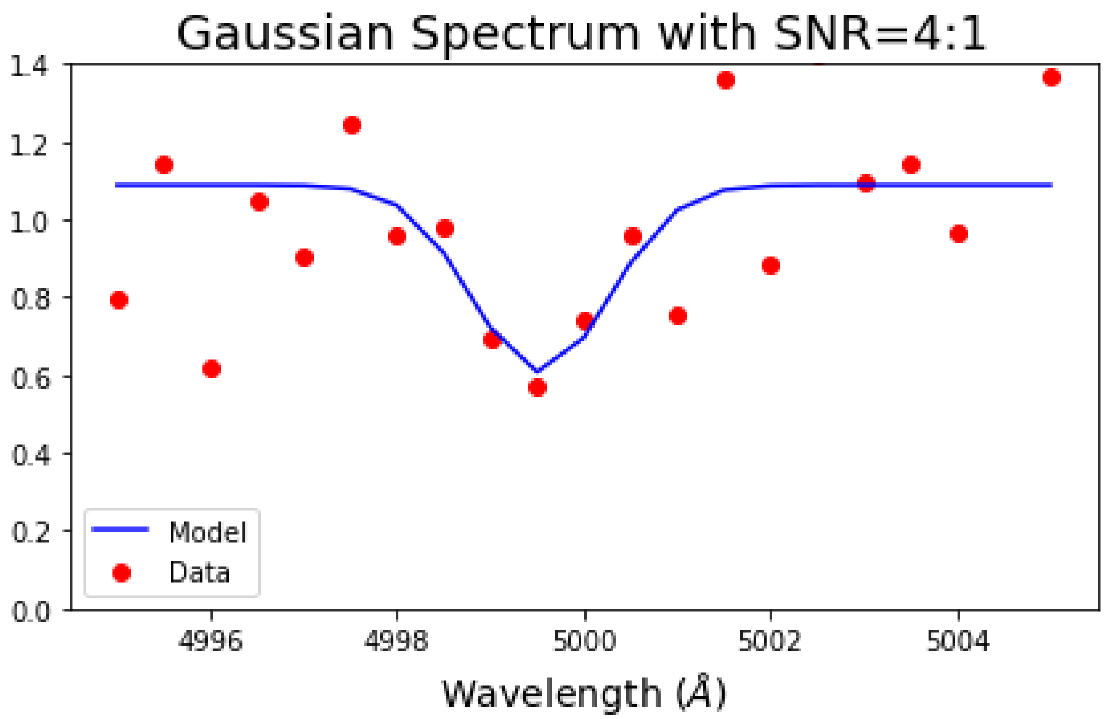
[6]:
```
# Plot data vs. model for SNR = 100:1
accuracy_100 = SNR_100.GaussianModel(plot=True)
```



[8]:
```
# Plot data vs. model for SNR = 20:1
accuracy_20 = SNR_20.GaussianModel(plot=True)
```

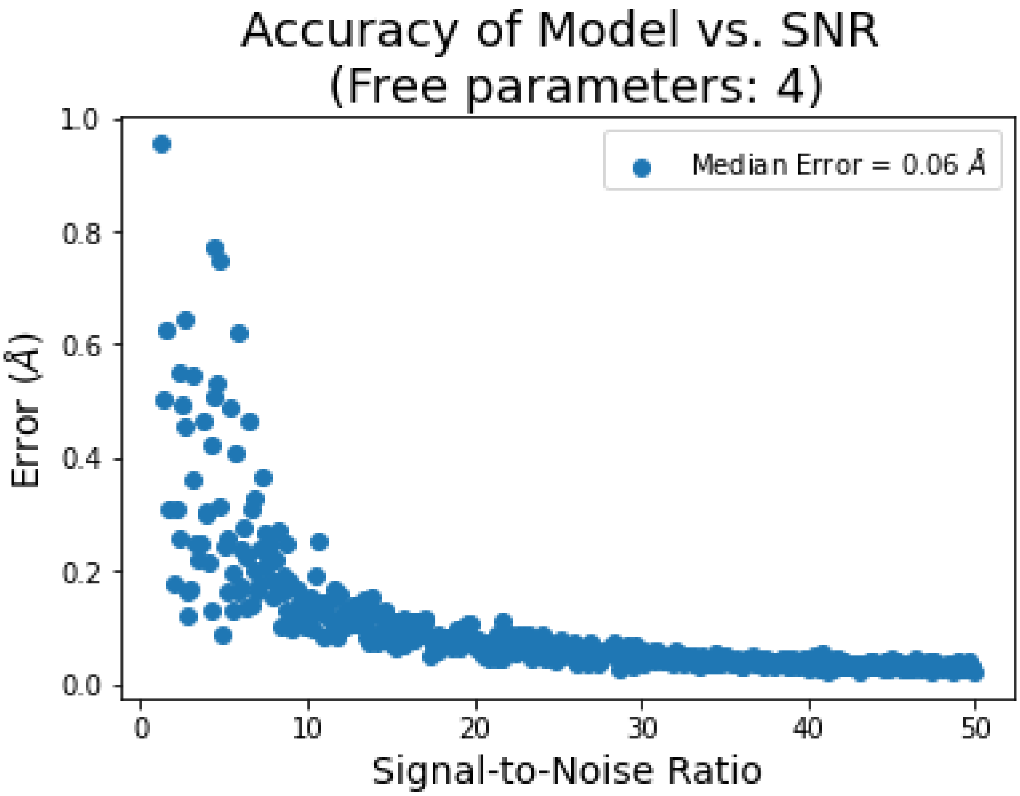


[10]:
```
# Plot data vs. model for SNR = 4:1
accuracy_4 = SNR_4.GaussianModel(plot=True)
```



[48]:
```
# Make list of SNRs and empty list of accuracies
SNRs = np.arange(1,50,.1)
Accuracy(4,SNRs)
```

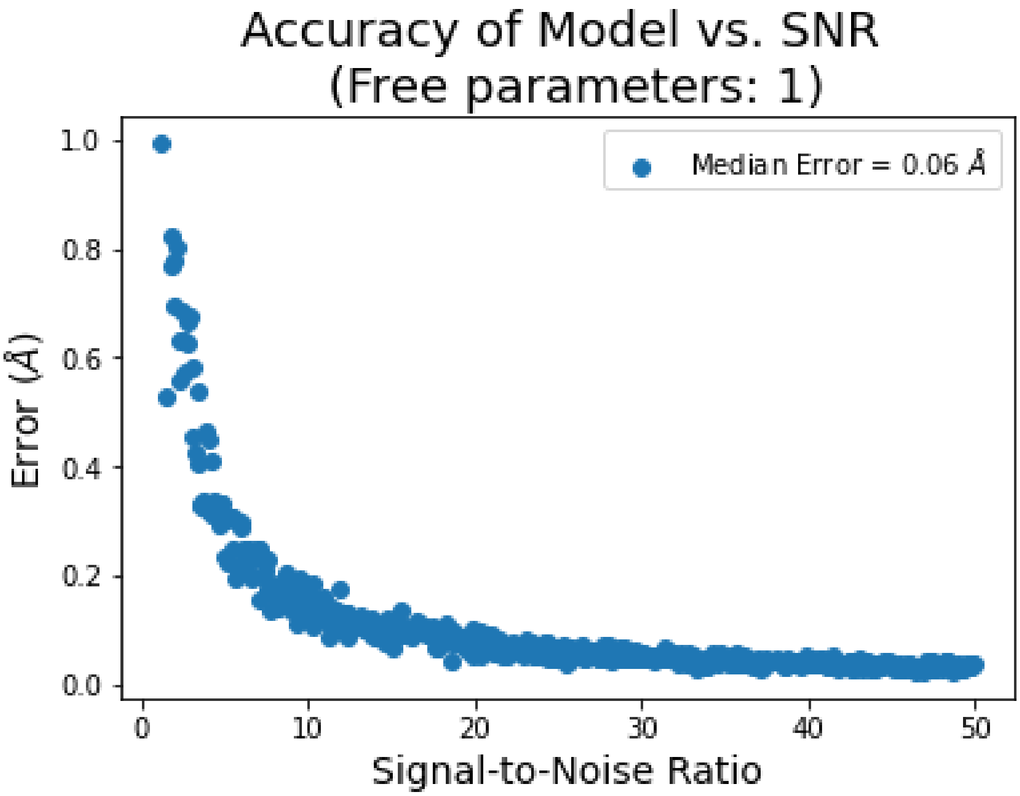
```
10 out of 490 trials had accuracy > 1
```



3.3 2c) Repeat 2b for the mean being the only free parameter

[49]:
```
Accuracy(1,SNRs)
```

```
8 out of 490 trials had accuracy > 1
```

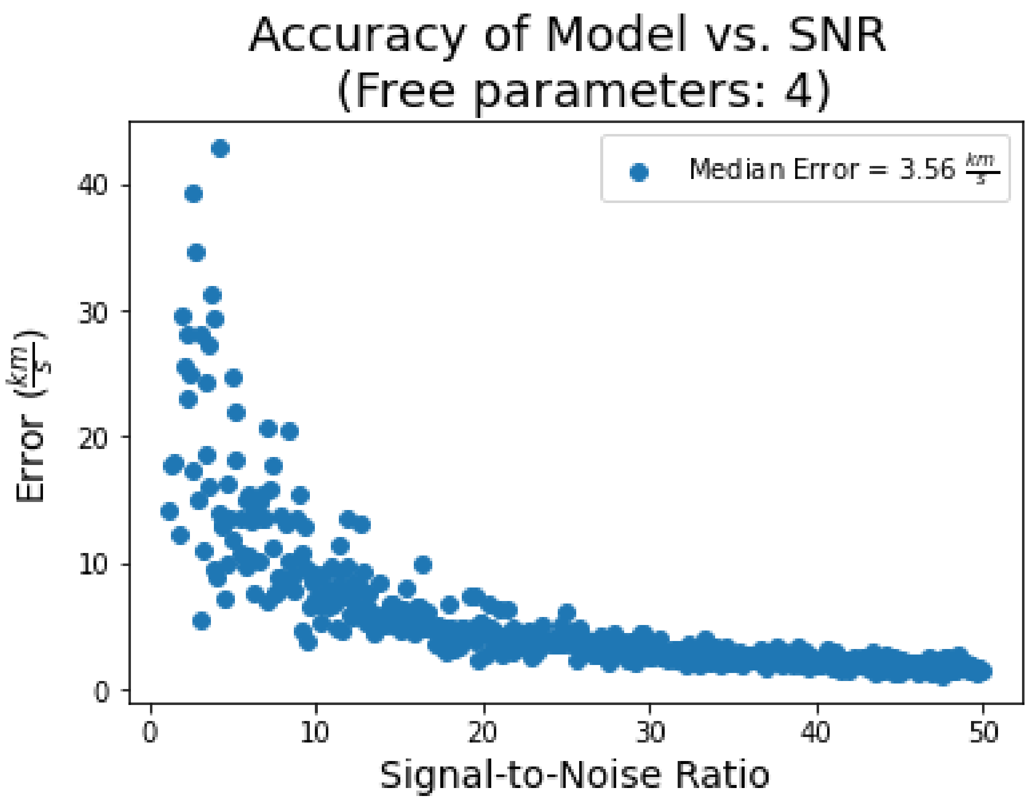


3.4 2d) Convert accuracy in angs to km/s for 2b and 2c (using non-relativistic Doppler Eqn.)

3.4.1 $\lambda' = \lambda_0 \left(1 + \frac{v}{c_0}\right)$ −> $v = c_0 \left(\frac{\lambda'}{\lambda_0} - 1\right)$

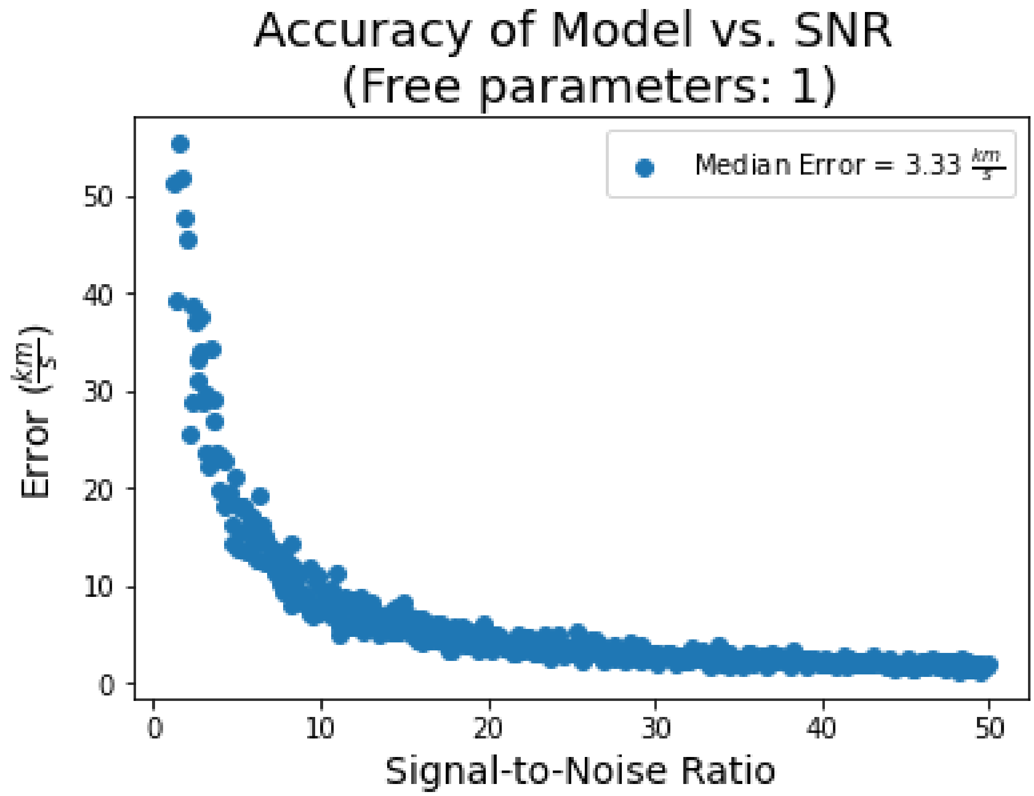
[51]:
```
Accuracy(4,SNRs,'km/s')
```

```
11 out of 490 trials had accuracy > 1
```



[52]:
```
Accuracy(1,SNRs,'km/s')
```

```
7 out of 490 trials had accuracy > 1
```


Accuracy of Model vs. SNR
(Free parameters: 1)
Median Error = 3.33 $\frac{km}{s}$
Error ($\frac{km}{s}$)
Signal-to-Noise Ratio
0
10
20
30
40
50

3.4.2 Models agree much better for data sets with higher SNR. Once SNR > 30, model doesn't fit significantly better if SNR keeps getting better (increasing). Models also agree better when you fit less free parameters (less deviation this way)

3.5 2e) What mass planet could you detect if you have $N = 10^1, 10^2, 10^3$ absorption lines? Remember that accuracy improves as $\sqrt{N}$ (so $\frac{accuracy}{\sqrt{N}}$)

3.5.1 For $N = 10$: $\sqrt{10} \approx 3.16$ −> $acc. \approx 1.10\frac{km}{s} = 1110\frac{m}{s}$ −> detect $M_p > 10^{28.5}kg \approx 16.6M_{Jup}$

3.5.2 For $N = 10^2$: $\sqrt{10^2} = 10$ −> $acc. \approx 0.34\frac{km}{s} = 340\frac{m}{s}$ −> detect $M_p > 10^{28}kg \approx 5.26M_{Jup}$

3.5.3 For $N = 10^3$: $\sqrt{10^3} \approx 31.6$ −> $acc. \approx 0.10\frac{km}{s} = 100\frac{m}{s}$ −> detect $M_p > 10^{27.5}kg \approx 1.66M_{Jup}$

4 3) Plot of orbital inclination vs. semi-major axis with lines corresponding to grazing transit for planets from Earth to Jupiter-size. Also shading approx. habitable zone.

4.1 Grazing transit

4.1.1 i) Impact parameter (b): $b = acos(i)$

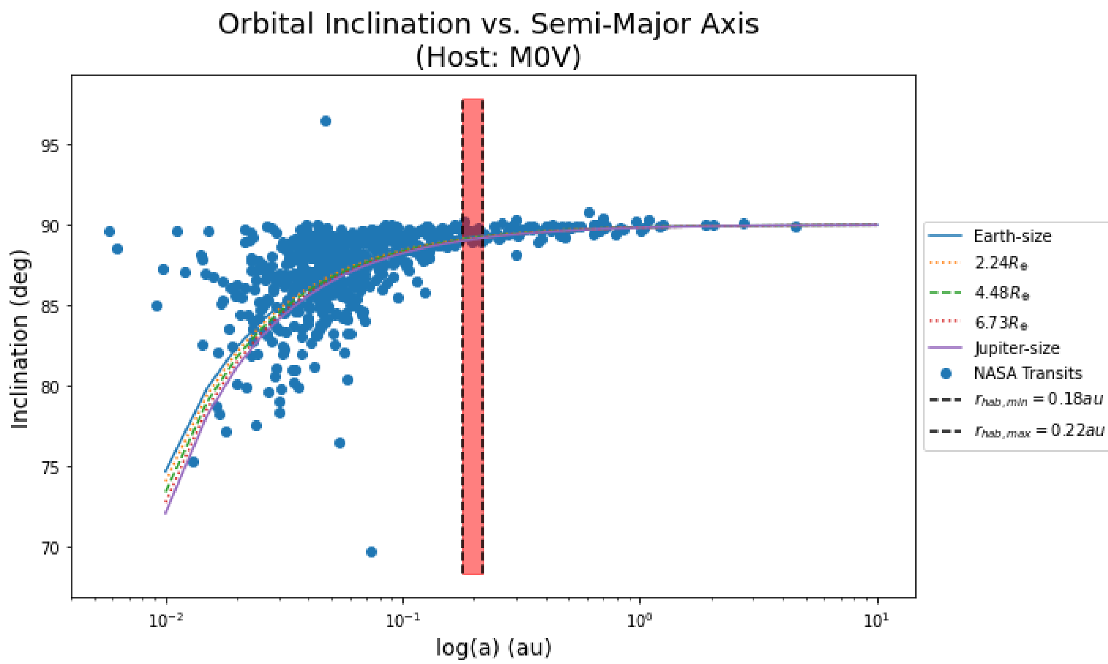
4.1.2 ii) Grazing condition: $b = acos(i) \leq R_p + R_*$

4.2 3a) M0V (dwarf) Host Star: $T_{eff} = 3870K; R_* = .559R_\odot$

4.3 3b) K0III (giant) Host Star: $T_{eff} = 4810K, R_* = 10R_\odot$ (couldn't find exact radius)

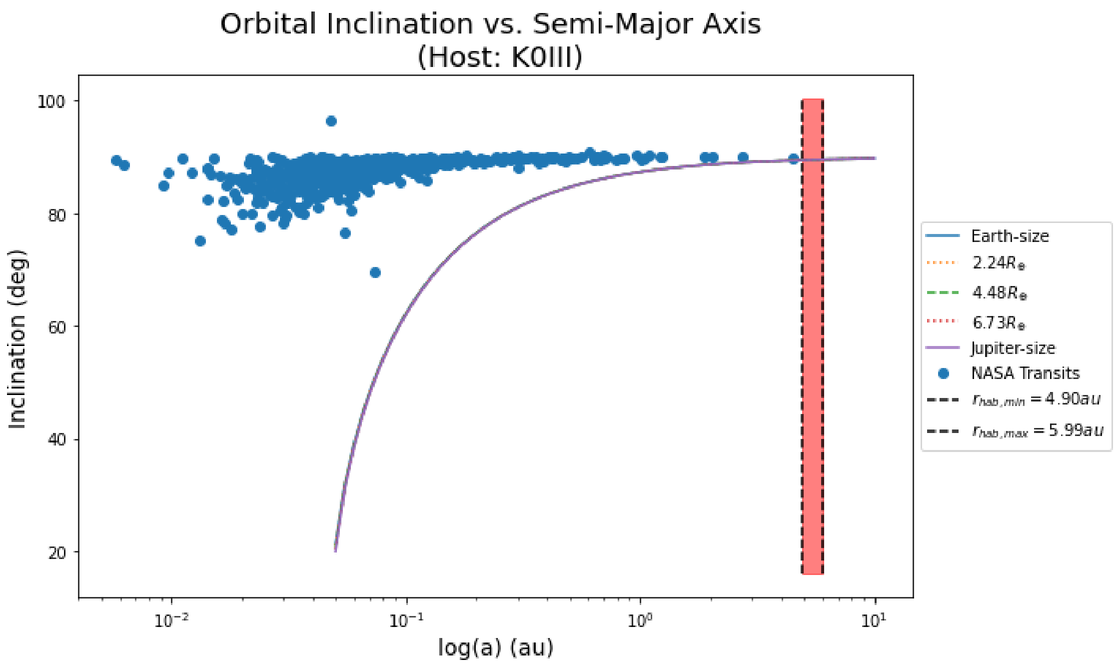
4.3.1 Property source: http://www.pas.rochester.edu/~emamajek/EEM_dwarf_UBVIJHK_colo

[4]:
```python
# Plot grazing transits and habitable zone for MOV star
MOV = GrazingTransit('M')
MOV.InclinationSemiMajor()
```



[5]:
```python
# Plot grazing transits and habitable zone for M0V star
K0III = GrazingTransit('K')
K0III.InclinationSemiMajor()
```

```
/d/users/jimmy/Documents/ASTR5490/HW3/GrazingTransit.py:56: RuntimeWarning:
invalid value encountered in arccos
  def InclinationSemiMajor(self):
```



5 4) Simulate a star as a solid face-on disk by breaking the stellar surface into a grid of 1000x1000 square pixels

5.1 4a) Assign each pixel a surface brightness with 1.0 at the center. Use quadratic limb darkening tables-Van Hamme (1993)-to assign a relative brightness to other pixels. Assume G2V star with $T_{eff} = 5500K$ and $\lambda_{obs} = 5000$ angs

5.1.1 Van Hamme (1993): https://ui.adsabs.harvard.edu/abs/1993AJ....106.2096V/abstract

5.1.2 Source for log(g) of dwarf (V) stars: http://www.astro.sunysb.edu/metchev/PHY521/lecture

5.2 Q.L.D.: $I(\mu) = I(0)\left[1 - a(1-\mu) - b(1-\mu)^2\right]$

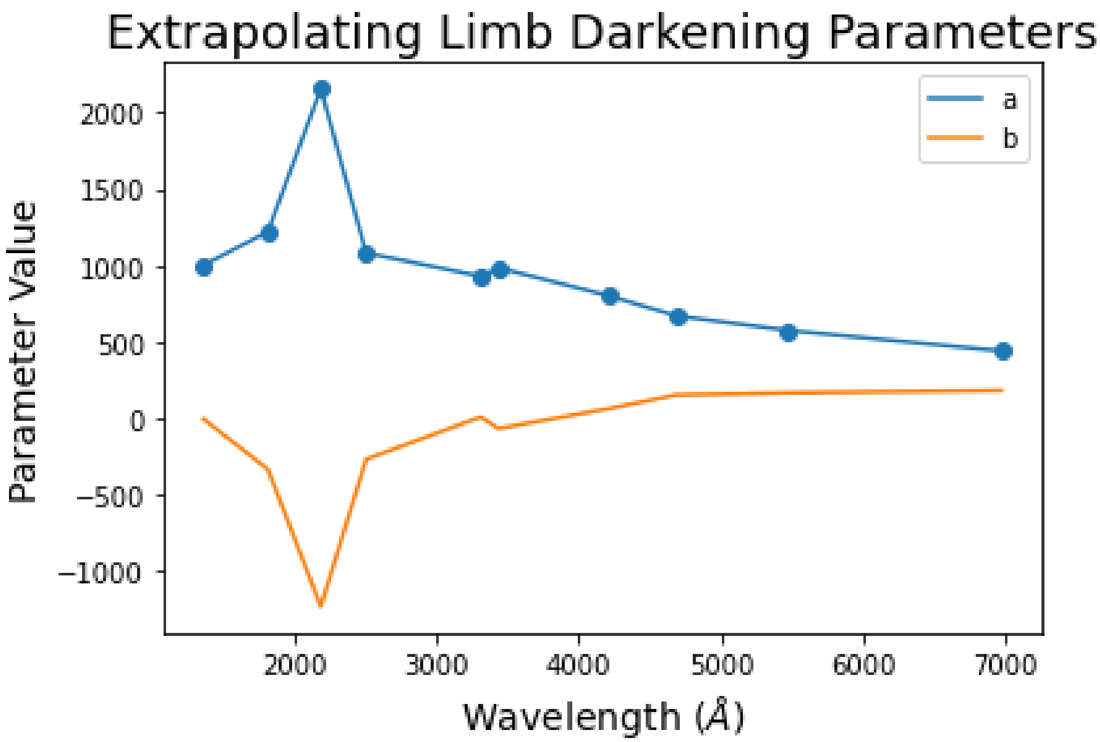
5.3 Substituting for μ: $I(\mu) = I(0)\left[1 - a\left(1 - \sqrt{\frac{R_*^2 - r^2}{R_*^2}}\right) - b\left(1 - \sqrt{\frac{R_*^2 - r^2}{R_*^2}}\right)^2\right]$ where a is the radius of the disc and r is the distance from the center of the disc (guidance from http://orca.phys.uvic.ca/~tatum/stellatm/atm6.pdf)

5.4 I'm sourcing a and b from Table 5 on the 10th page of Wade & Rucinski (1985): http://articles.adsabs.harvard.edu/pdf/1985A%26AS...60..471W

[57]:
```
# We're using 5000 angstroms so this is my attempt to extrapolate proper a and
↪b values
lambdas = [1362,1815,2187,2506,3312,3437,4212,4687,5475,6975]
a_list = [1002,1222,2157,1082,930,985,803,671,576,442]
b_list = [-3,-332,-1230,-266,11,-65,65,154,168,183]
```

```
plt.plot(lambdas,a_list,label='a')
plt.plot(lambdas,b_list,label='b')
plt.scatter(lambdas,a_list)
plt.xlabel(r'Wavelength ($\AA$)',fontsize=14)
plt.ylabel('Parameter Value',fontsize=14)
plt.title('Extrapolating Limb Darkening Parameters',fontsize=18)
plt.legend()
```

[57]: <matplotlib.legend.Legend at 0x7fc85429eeb8>

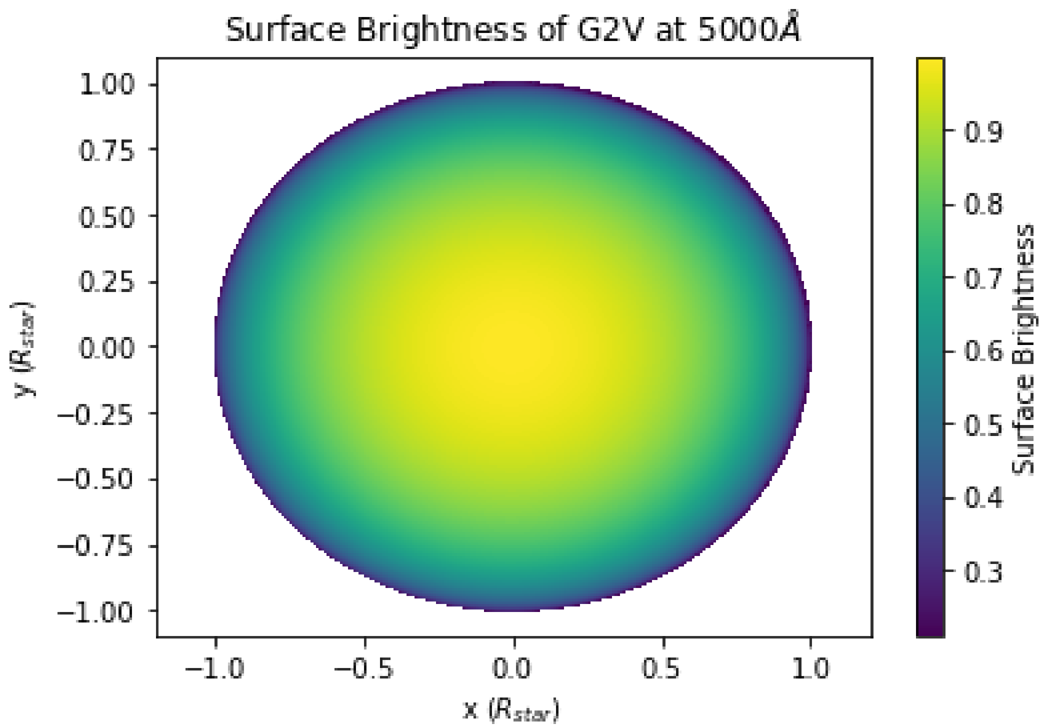


5.4.1 In range of 5000 angs., the relationship for both parameters seems pretty linear so I'll use a linear extrapolation to find a and b at 5000 angs

[58]:
```
# List indexes of lambdas that lie above and below 5000 angs.
min_index = 7
max_index = 8

# Define sublists of points below and above point of interest
a_sub = [a_list[min_index],a_list[max_index]]
b_sub = [b_list[min_index],b_list[max_index]]
lambdas_sub = [lambdas[min_index],lambdas[max_index]]
```

```python
# Interpolate to find proper values at 5000 angs
a_new = np.interp(5000,lambdas_sub,a_sub)
b_new = np.interp(5000,lambdas_sub,b_sub)
print(a_new,b_new)
```

[59]:
```python
# Use LimbDarkening function to plot surface of G2V star
star = LimbDarkening(633.27/1000,159.56/1000)
x,y,intensities = star.Star(1000)
```

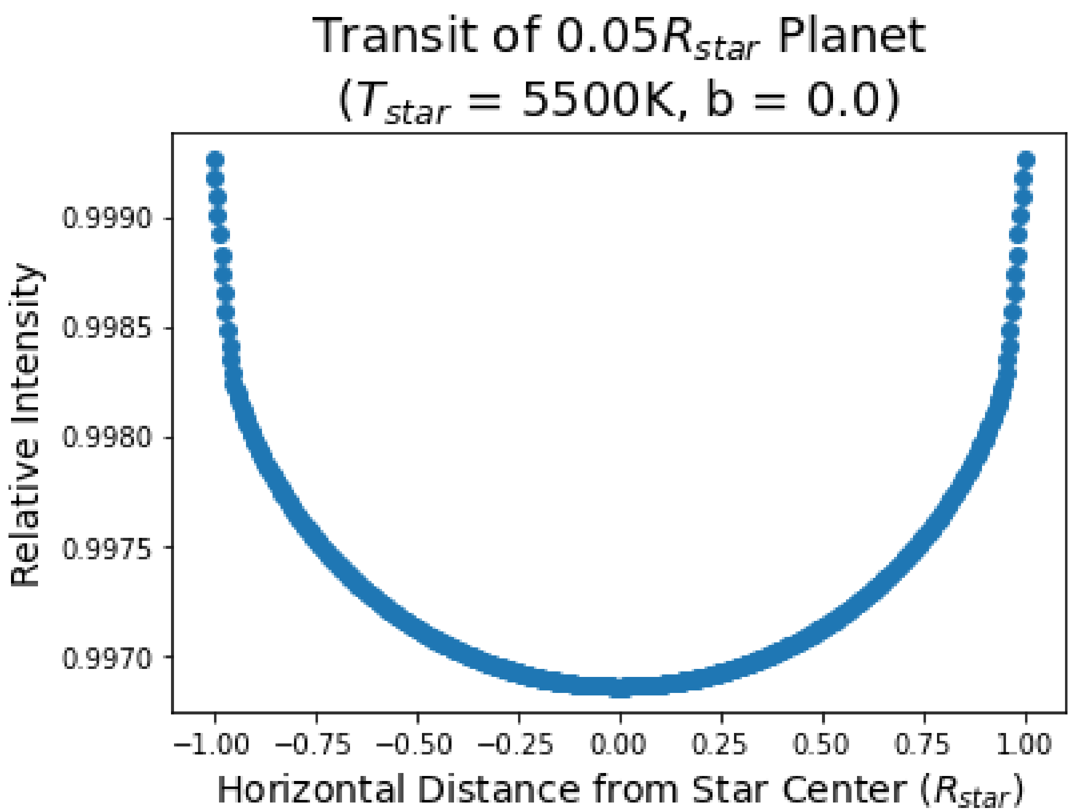


5.5 4b) Plot light curve of transit of $.05R_{star}$ at impact parameter 0

[10]:
```python
star_b = LimbDarkening(5500,500)
```

[27]:
```python
star_b.Transit(0.05,0.0,False)
```

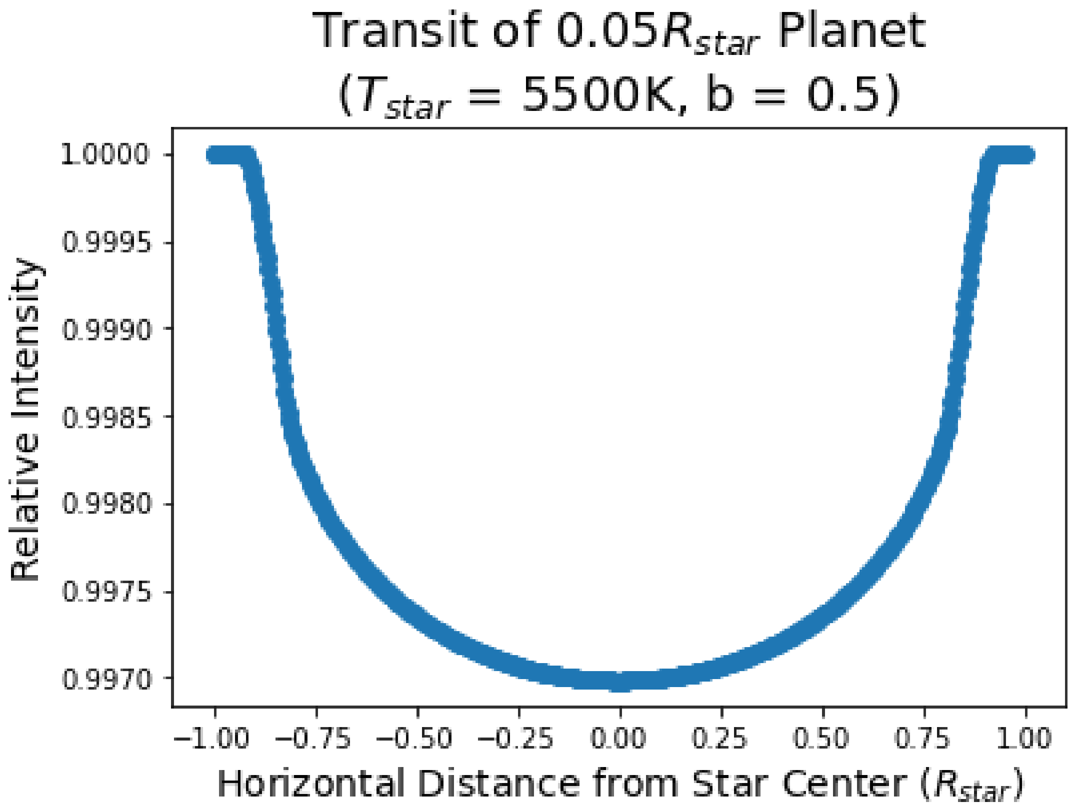
```
My program took 240.88 seconds to run
```



5.6 4c) Plot light curve of transit of $.05R_{star}$ at impact parameter 0.5

[11]:
```
star_b.Transit(0.05,0.5,False)
```

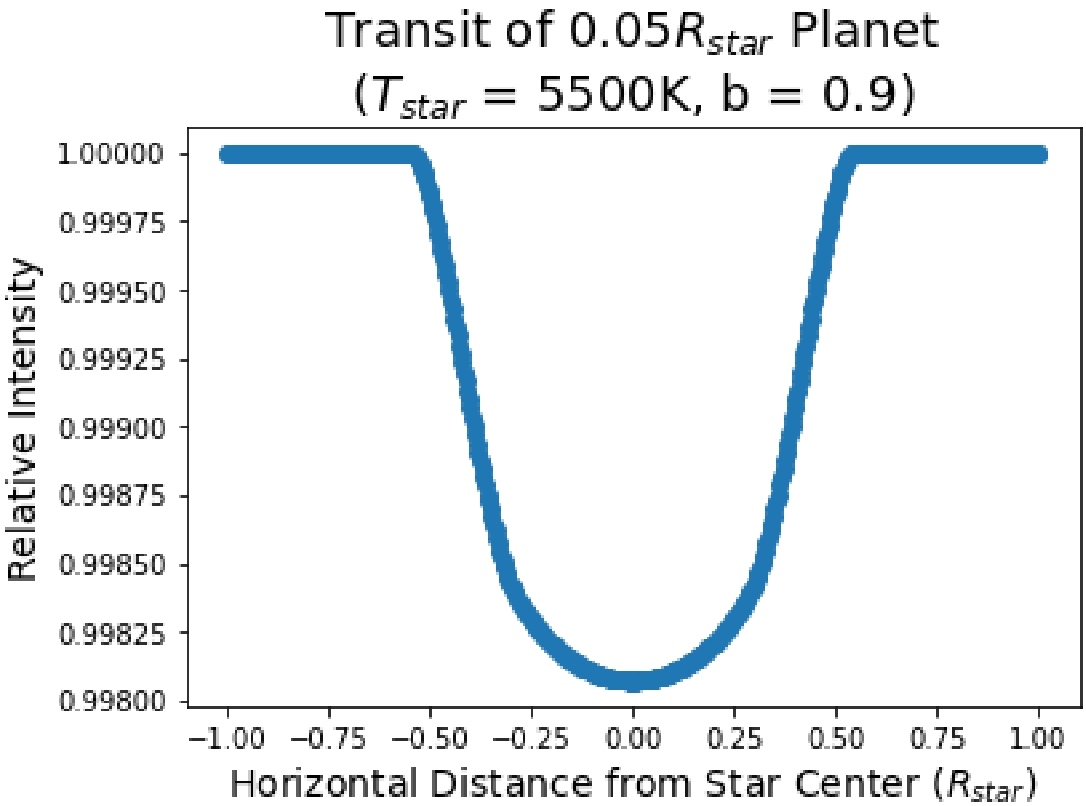
```
My program took 3.75 minutes to run
```



5.7 4d) Plot light curve of transit of $.05R_{star}$ at impact parameter 0.9 (grazing transit)

[12]:
```
star_b.Transit(0.05,0.9,False)
```

```
My program took 3.84 minutes to run
```



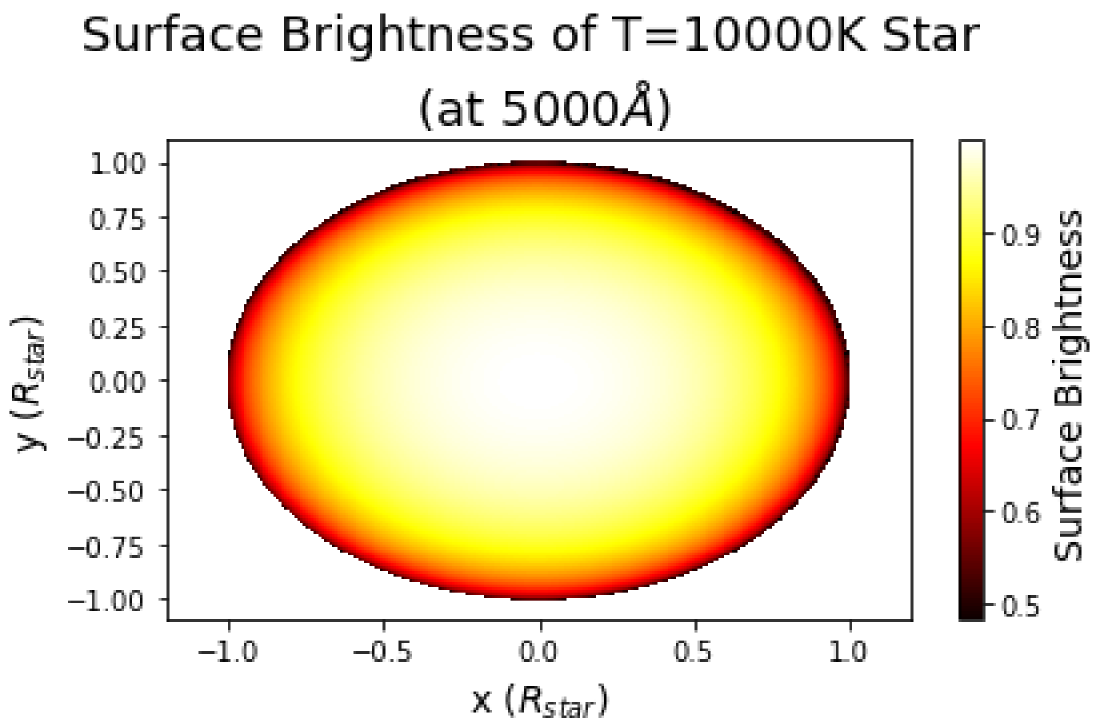
5.8 4e) Overplot on the plots for B/C/D the light curve for a T=10,000 K host star and a 3600 K host star and summarize the differences

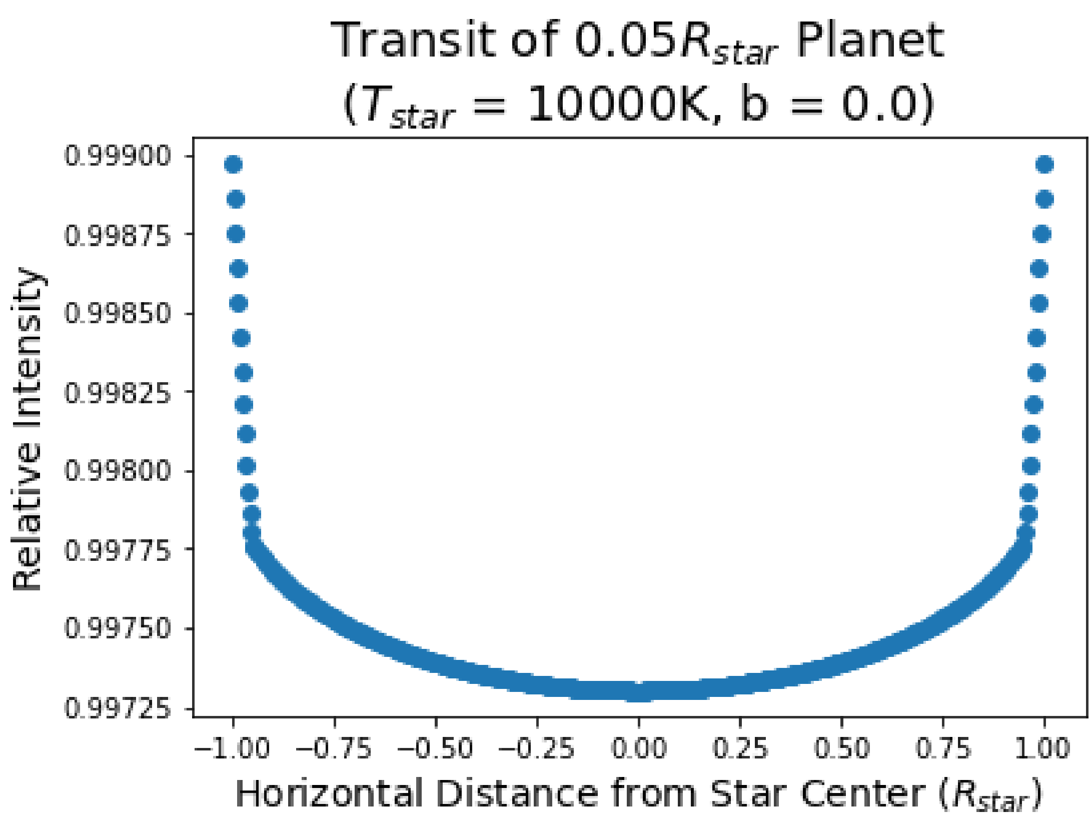
5.8.1 For 10000K, got parameters from: First two rows at relevant temp. in Table 3 of Claret (2013) http://cdsarc.u-strasbg.fr/ftp/cats/J/A+A/552/A16/ori/Table3

[13]:
```
star_10000K = LimbDarkening(10000,500)
```

[30]:
```
# impact parameter = 0
star_10000K.Star(plot=True)
star_10000K.Transit(0.05,0.0,False)
```

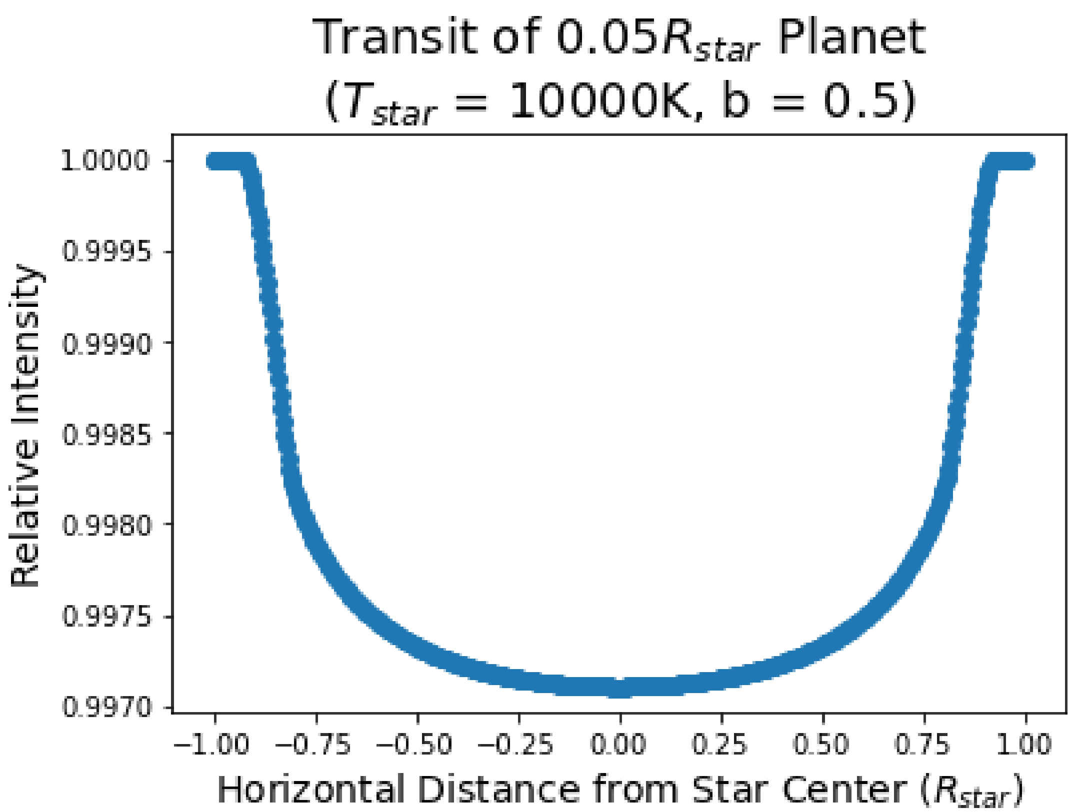
```
My program took 4.87 minutes to run
```


Surface Brightness of T=10000K Star
(at 5000Å)
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
y (R_{star})
−1.0
−0.5
0.0
0.5
1.0
x (R_{star})
0.9
0.8
0.7
0.6
0.5
Surface Brightness


Transit of 0.05R_{star} Planet
(T_{star} = 10000K, b = 0.0)
0.99900
0.99875
0.99850
0.99825
0.99800
0.99775
0.99750
0.99725
Relative Intensity
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
Horizontal Distance from Star Center (R_{star})

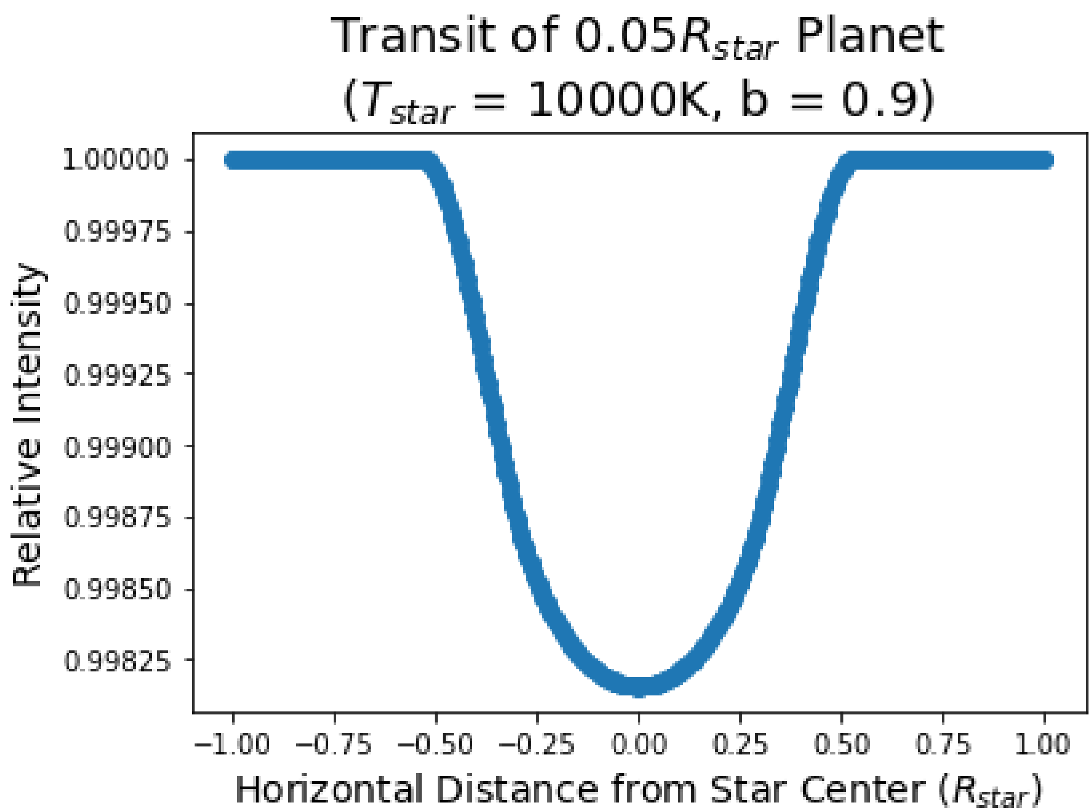
[31]:
```
# impact parameter = 0.5
star_10000K.Transit(0.05,0.5,False)
```

My program took 4.15 minutes to run



[32]:
```
# impact parameter = 0.9
star_10000K.Transit(0.05,0.9,False)
```

My program took 4.15 minutes to run

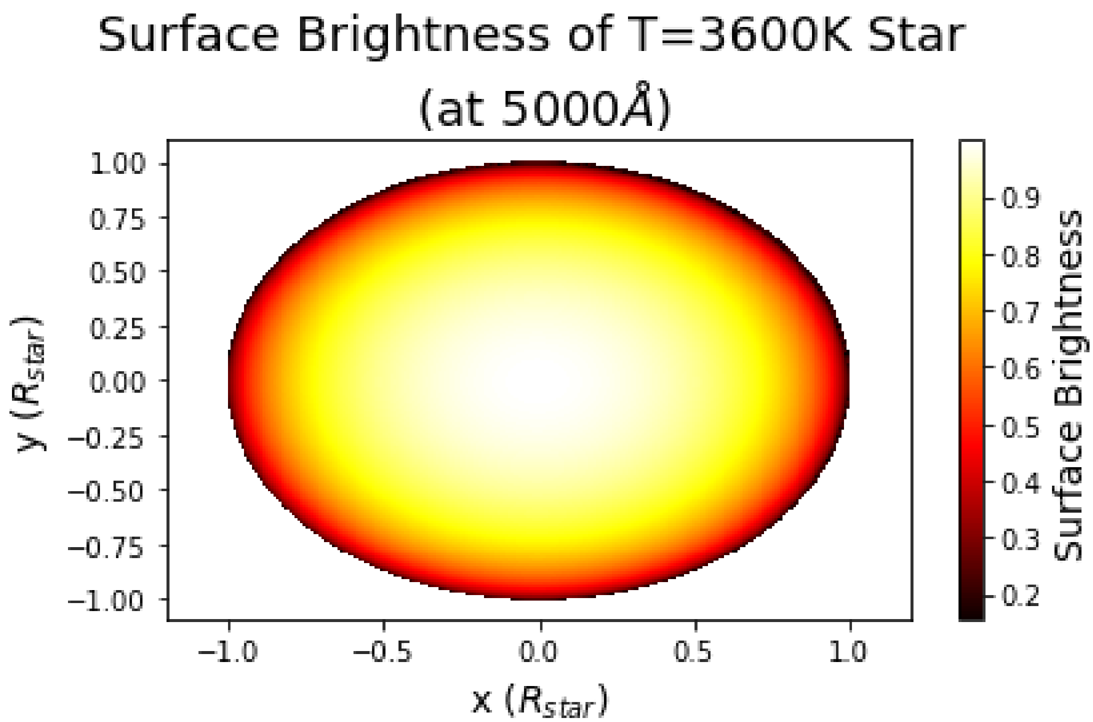


5.8.2 For 3600K, got parameters from: Column 10 (V filter) of Table 2 from Claret (1998) https://cdsarc.unistra.fr/viz-bin/ReadMe/J/A+A/335/647?format=html&tex=true

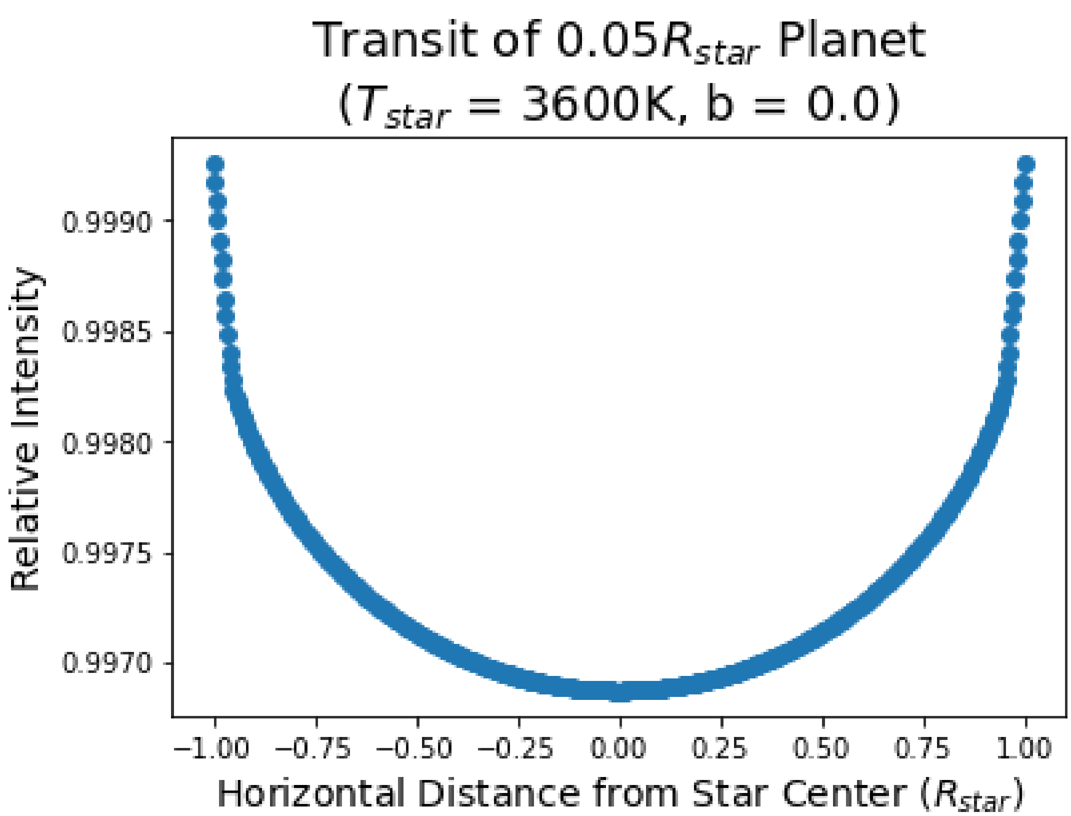
[14]:
```
star_3600K = LimbDarkening(3600,500)
```

[35]:
```
# impact parameter = 0
star_3600K.Star(plot=True)
star_3600K.Transit(0.05,0.0,False)
```

```
My program took 4.13 minutes to run
```



Surface Brightness of T=3600K Star
(at 5000Å)
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
y (R_{star})
−1.0
−0.5
0.0
0.5
1.0
x (R_{star})
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
Surface Brightness

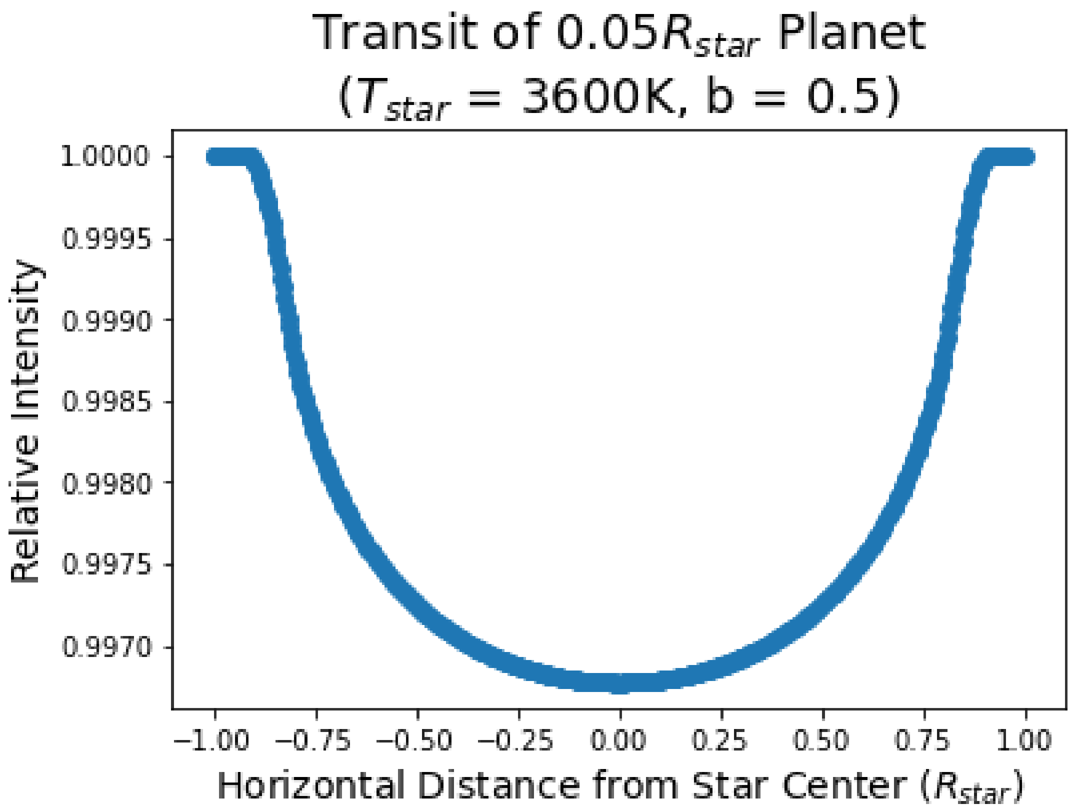


Transit of $0.05R_{star}$ Planet
(T_{star} = 3600K, b = 0.0)
0.9990
0.9985
0.9980
0.9975
0.9970
Relative Intensity
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
Horizontal Distance from Star Center (R_{star})

[36]:

```
# impact parameter = 0.5
star_3600K.Transit(0.05,0.5,False)
```

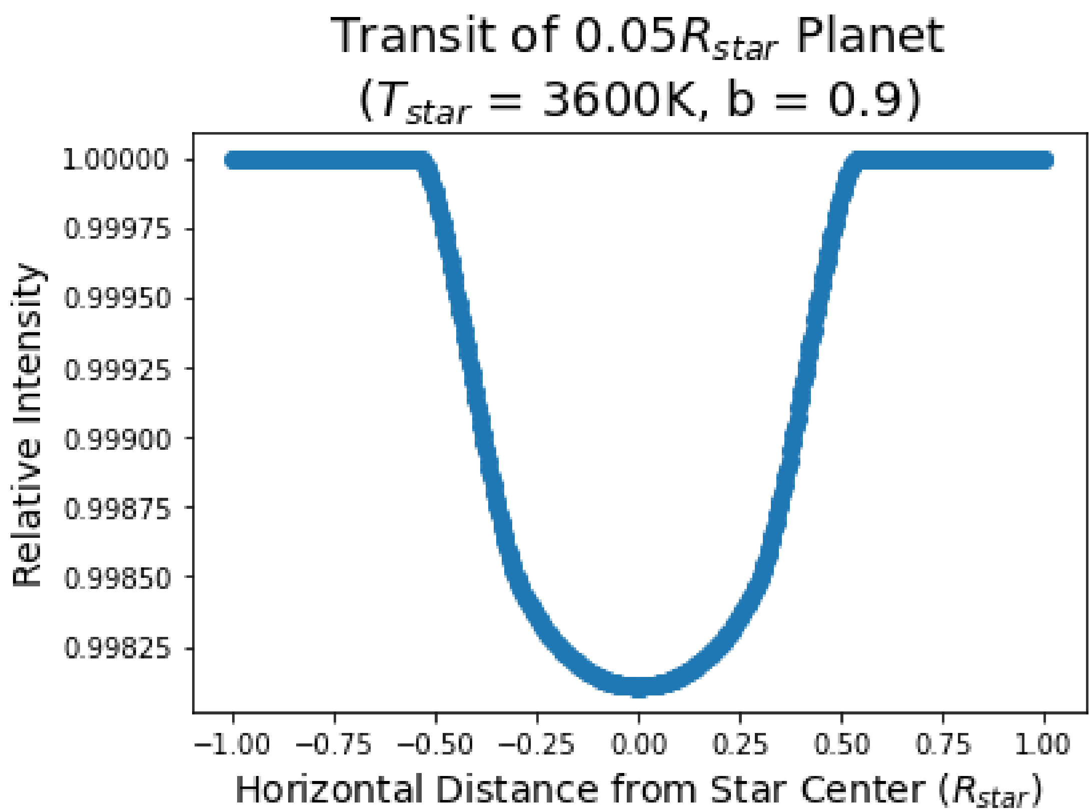
```
My program took 4.20 minutes to run
```



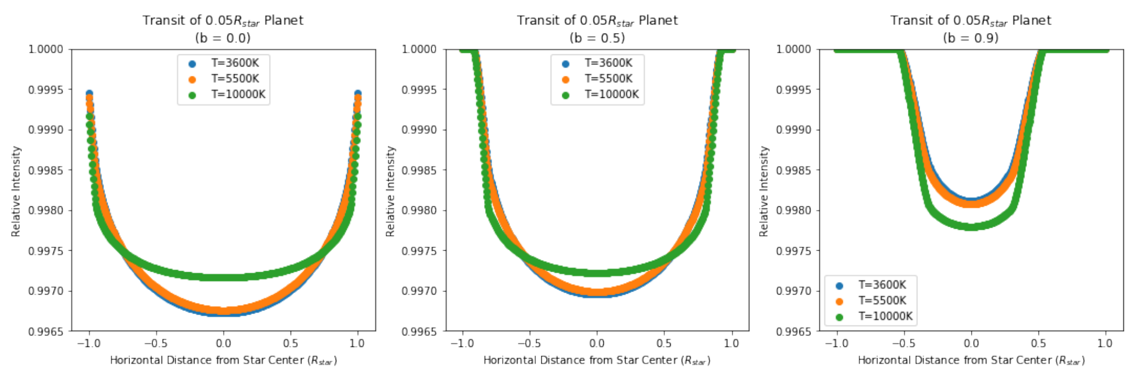
[15]:

```
# impact parameter = 0.9
star_3600K.Transit(0.05,0.9,False)
```

```
My program took 3.74 minutes to run
```



[30]:
```
LightCurveCompare()
```



5.9 4f) Comparing result from 4c (b=0.5) to results from Kreidberg 2015 who used 'batman'

5.9.1 Figure 2 of Kreidberg 2015 https://iopscience.iop.org/article/10.1086/683602/pdf

5.9.2 My plots are very similar qualititatively with the same wide U-shaped curve with flat edges at the beginning and end of the transit. Quantitatively, my model dooes not dip as low as Kreidberg (0.997 for mine, 0.990 for theirs). Overall, I'd say my model replicates theirs quite well

6 5) Now let your star rotate with an equatorial velocity of $v_R = 10\frac{km}{s}$. The projected radial velocity of each surface element is a function of stellar latitude ϕ and longitude θ

##

$v_{rad} = v_R sin\theta cos\phi$

6.1 5a) Using your pixelated numerical star surface code, produce an intensity-weighted velocity profile plot for the star with no transit. This is also known as the rotational velocity profile (in the following, I use conventions define in Table 1 here https://en.wikipedia.org/wiki/Del_in_cylindrical_and_spherical_coordinates)

##

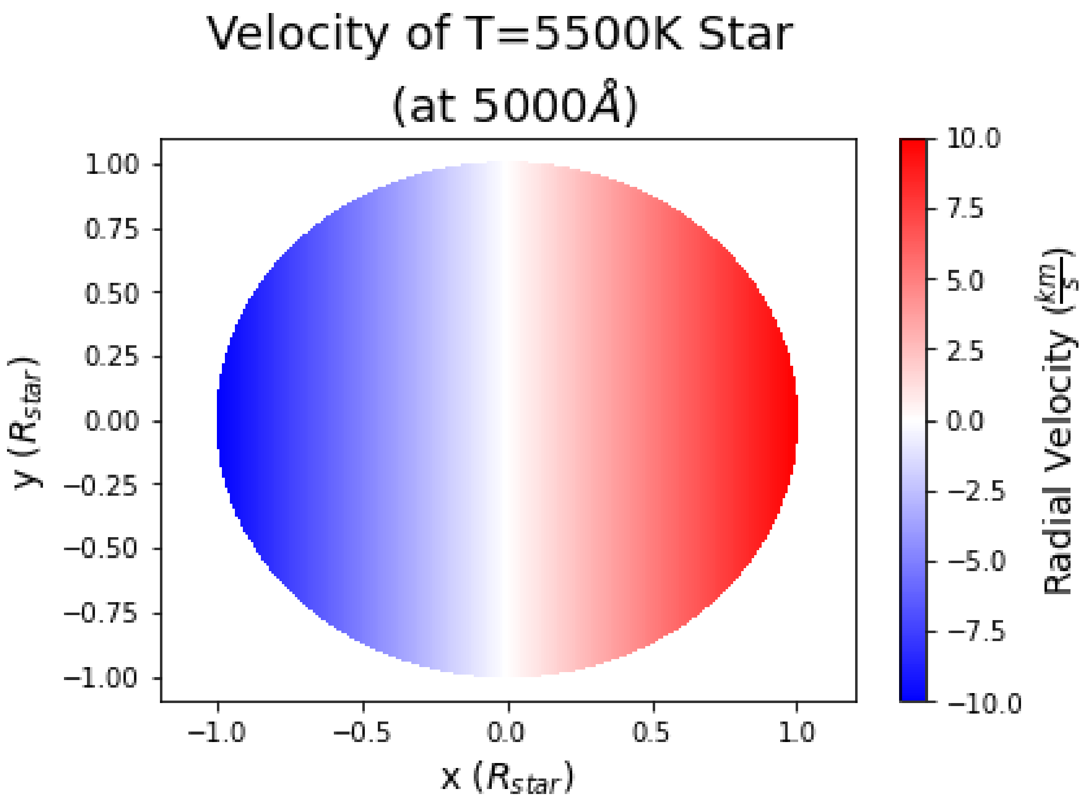
$\theta = arctan\left(\frac{\sqrt{x^2+y^2}}{z}\right)$, $\phi = arctan\left(\frac{y}{x}\right)$

##

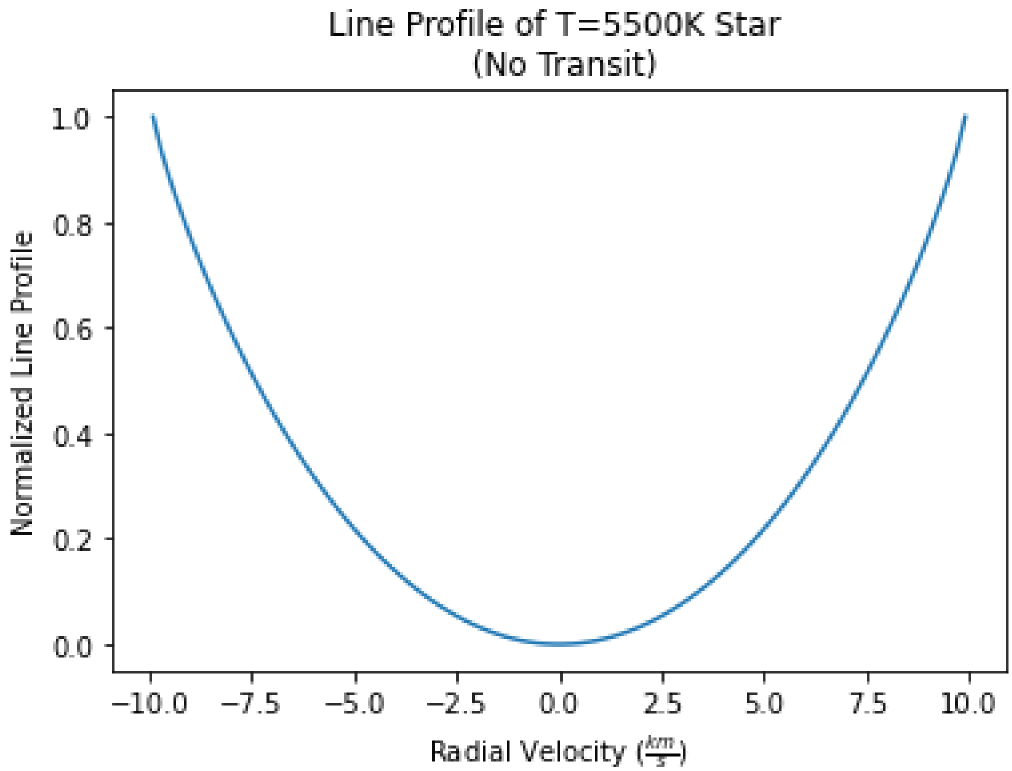
$v_{rad} = v_R sin\left(arctan\left(\frac{\sqrt{x^2+y^2}}{\sqrt{R_{star}^2-x^2-y^2}}\right)\right) cos\left(arctan\left(\frac{y}{x}\right)\right)$

[5]:
```
# Tests of class and functions within the class
RotatingStar = LimbDarkening(5500,1000)
RotatingStar.RVProfile(plot='star')
```

```
/d/users/jimmy/Documents/ASTR5490/HW3/LimbDarkening.py:53: RuntimeWarning:
invalid value encountered in sqrt
  mu = np.sqrt(1-abs(r**2/R_star**2))
/d/users/jimmy/Documents/ASTR5490/HW3/LimbDarkening.py:213: RuntimeWarning:
invalid value encountered in sqrt
  theta = np.sqrt((x**2+y**2)/(1.0-x**2-y**2))
```



[6]:
```
RotatingStar.RVProfile(plot='profile')
```



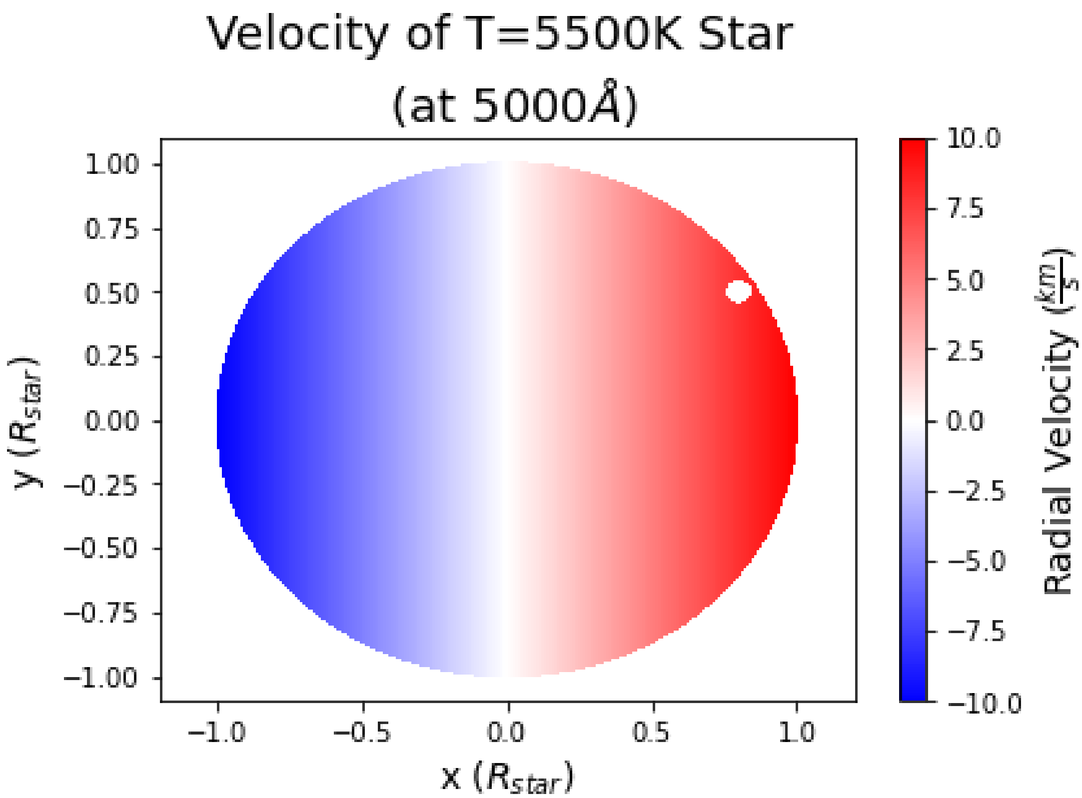
6.2 5b) Produce the same proifle for the star for when the planet is at four different positions across the disk of the star at an impact parameter of b=0.5

6.3 5c) Plot the difference between the 5a and 5b plots. Compare quantitatively to those in Figure 1 of Gaudia & Winn (2007, ApJ).

6.4 https://ui.adsabs.harvard.edu/abs/2007ApJ...655..550G/abstract

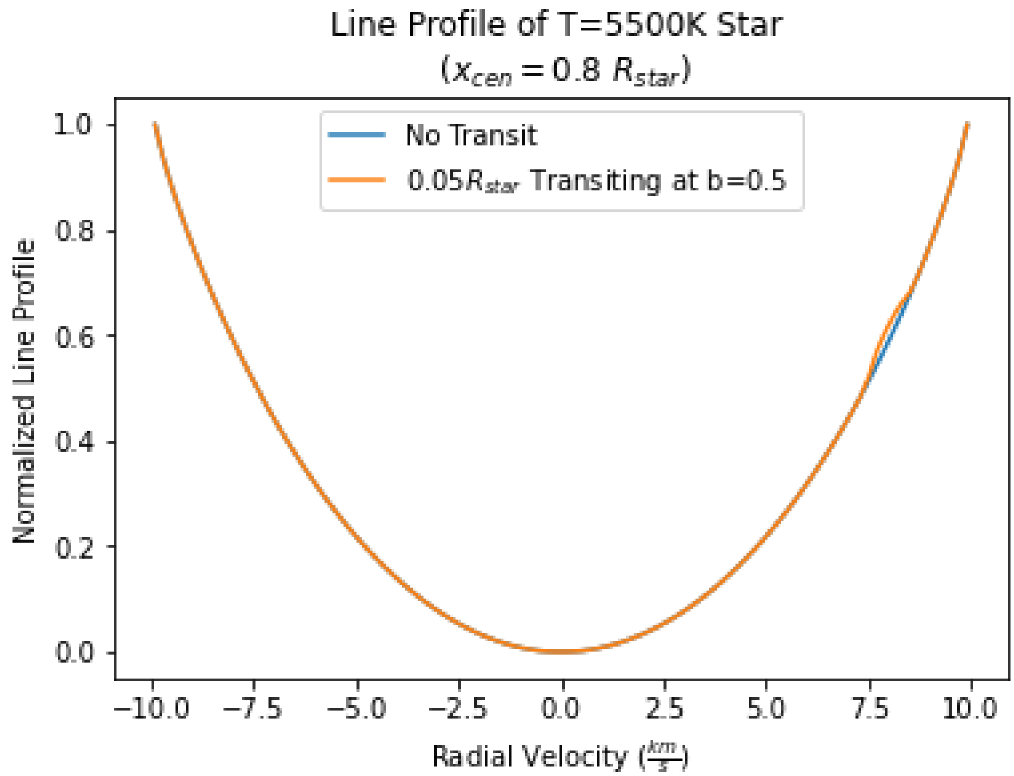
[10]:
```
RotatingStar.RVProfileTransit(x_center=0.8,plot='star')
```

```
/d/users/jimmy/Documents/ASTR5490/HW3/LimbDarkening.py:53: RuntimeWarning:
invalid value encountered in sqrt
  mu = np.sqrt(1-abs(r**2/R_star**2))
/d/users/jimmy/Documents/ASTR5490/HW3/LimbDarkening.py:213: RuntimeWarning:
invalid value encountered in sqrt
  theta = np.sqrt((x**2+y**2)/(1.0-x**2-y**2))
```



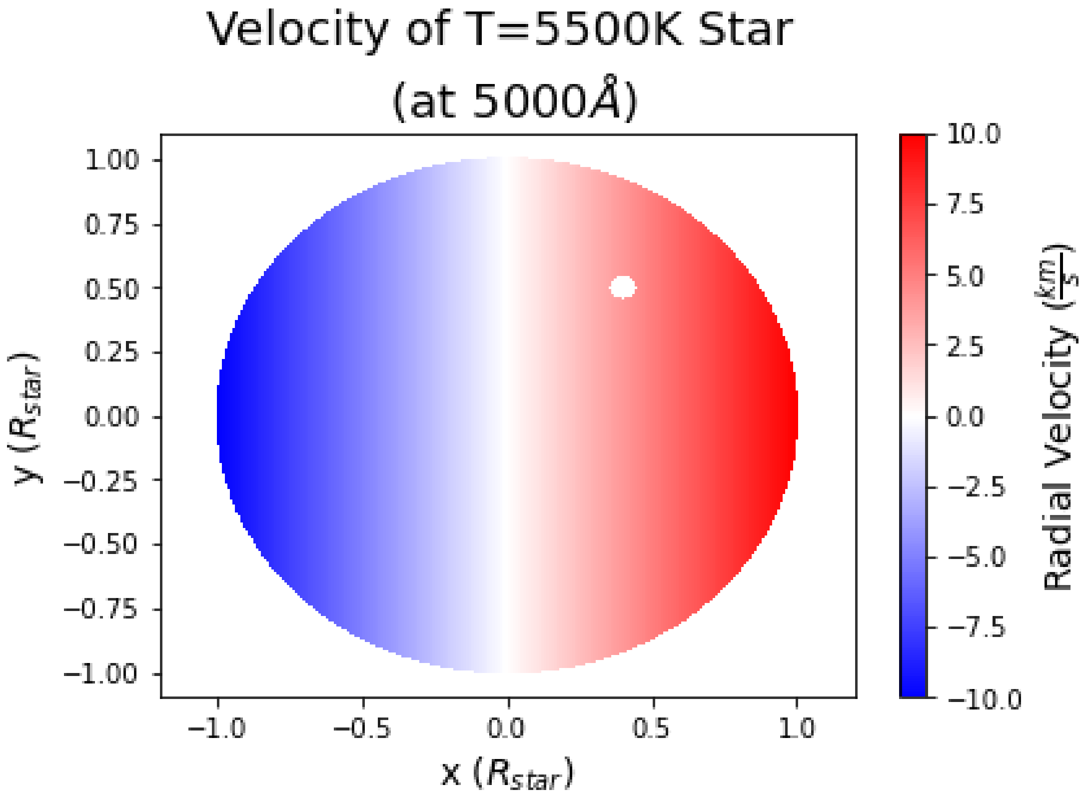
[14]:
```
RotatingStar.RVProfileTransit(x_center=0.8,plot='profile')
```

```
/d/users/jimmy/Documents/ASTR5490/HW3/LimbDarkening.py:53: RuntimeWarning:
invalid value encountered in sqrt
  mu = np.sqrt(1-abs(r**2/R_star**2))
/d/users/jimmy/Documents/ASTR5490/HW3/LimbDarkening.py:213: RuntimeWarning:
invalid value encountered in sqrt
  theta = np.sqrt((x**2+y**2)/(1.0-x**2-y**2))
```

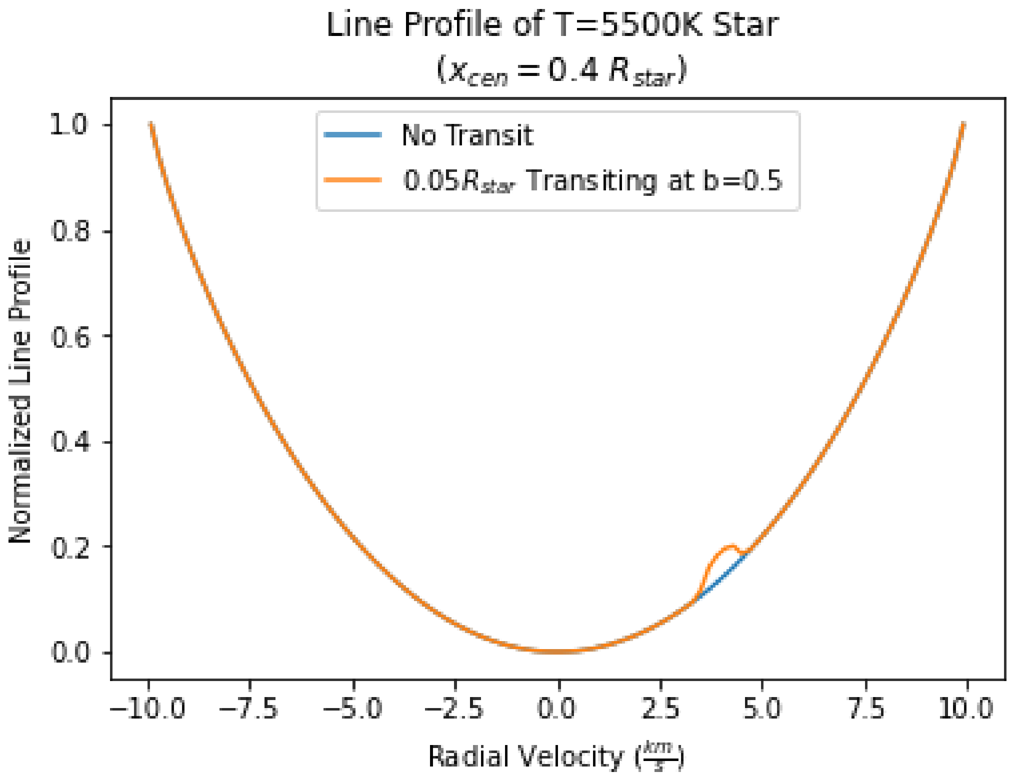


[15]:
```
RotatingStar.RVProfileTransit(x_center=0.4,plot='star')
```

```
/d/users/jimmy/Documents/ASTR5490/HW3/LimbDarkening.py:53: RuntimeWarning:
invalid value encountered in sqrt
  mu = np.sqrt(1-abs(r**2/R_star**2))
/d/users/jimmy/Documents/ASTR5490/HW3/LimbDarkening.py:213: RuntimeWarning:
invalid value encountered in sqrt
  theta = np.sqrt((x**2+y**2)/(1.0-x**2-y**2))
```

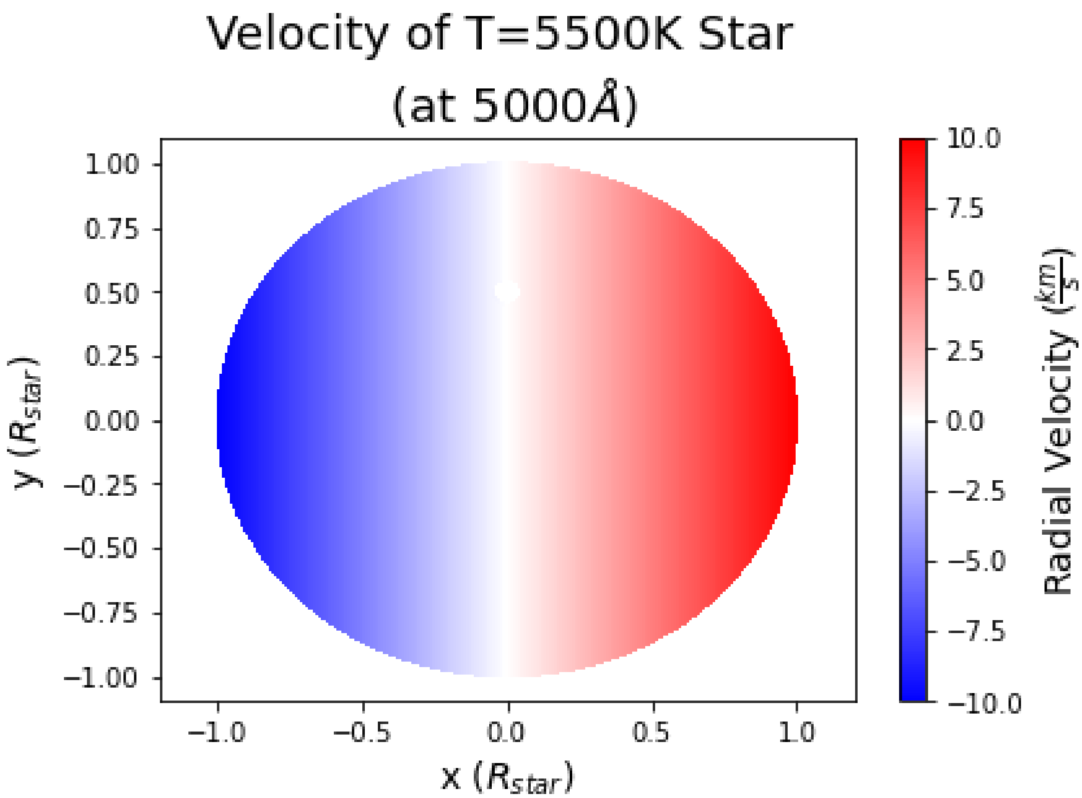


[16]:
```
RotatingStar.RVProfileTransit(x_center=0.4,plot='profile')
```

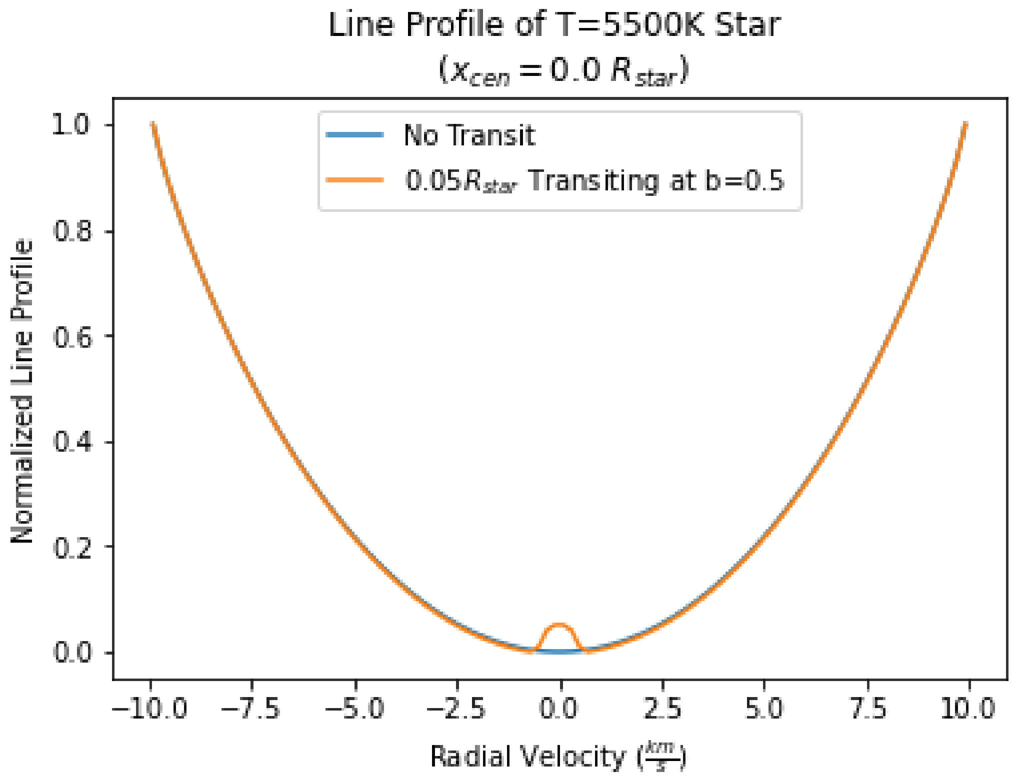


[17]:
```
RotatingStar.RVProfileTransit(x_center=0.0,plot='star')
```

```
/d/users/jimmy/Documents/ASTR5490/HW3/LimbDarkening.py:53: RuntimeWarning:
invalid value encountered in sqrt
  mu = np.sqrt(1-abs(r**2/R_star**2))
/d/users/jimmy/Documents/ASTR5490/HW3/LimbDarkening.py:213: RuntimeWarning:
invalid value encountered in sqrt
  theta = np.sqrt((x**2+y**2)/(1.0-x**2-y**2))
```

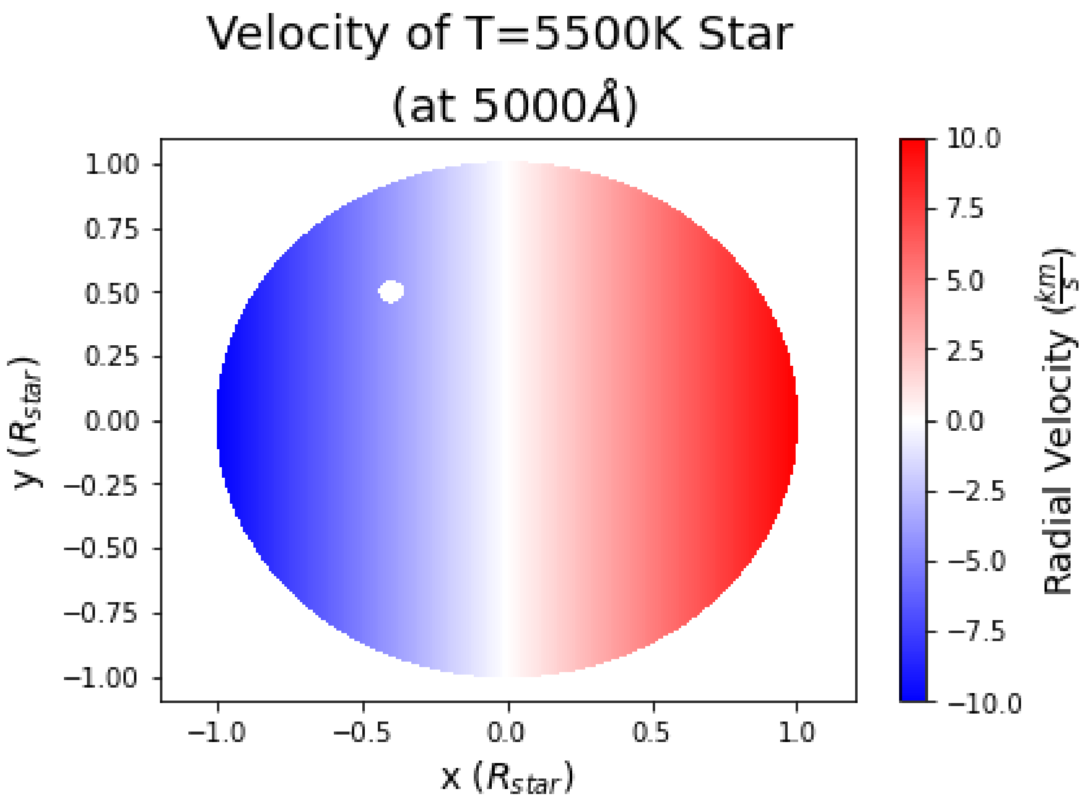


[18]:
```
RotatingStar.RVProfileTransit(x_center=0.0,plot='profile')
```

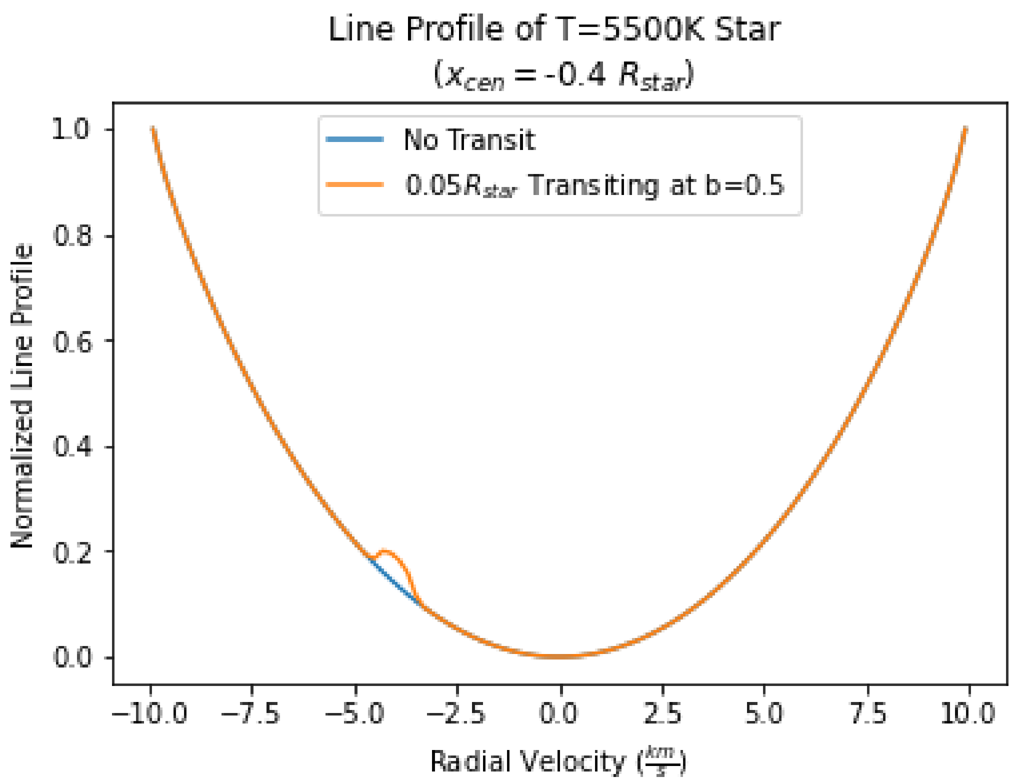


[19]:
```
RotatingStar.RVProfileTransit(x_center=-.4,plot='star')
```

```
/d/users/jimmy/Documents/ASTR5490/HW3/LimbDarkening.py:53: RuntimeWarning:
invalid value encountered in sqrt
  mu = np.sqrt(1-abs(r**2/R_star**2))
/d/users/jimmy/Documents/ASTR5490/HW3/LimbDarkening.py:213: RuntimeWarning:
invalid value encountered in sqrt
  theta = np.sqrt((x**2+y**2)/(1.0-x**2-y**2))
```



[20]:
```
RotatingStar.RVProfileTransit(x_center=-.4,plot='profile')
```



6.5 My plots are qualitatively very similar to those in Gaudi & Winn (2007, ApJ). My plots only include when the planet is just out-of-transit and fully in-transit so I don't have the zero-slope edges at the top-right and top-left of my plots like they do.

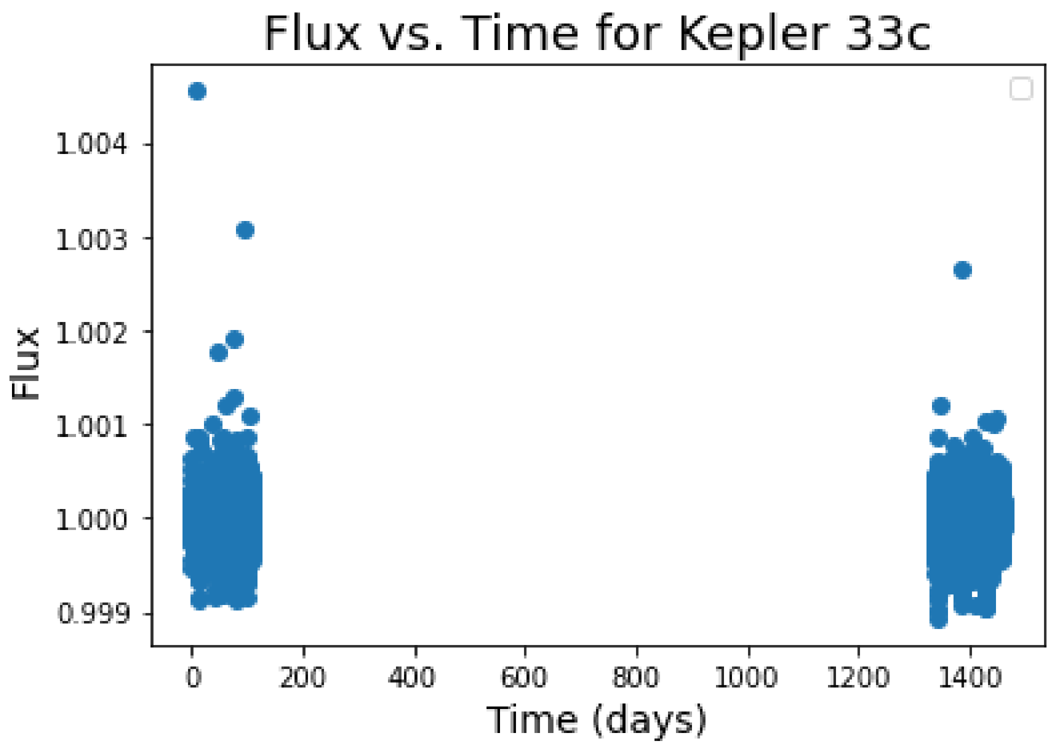
7 6) Fold Kepler 33 data at period of Planet C. Use 'batman' light curve tool (https://www.cfa.harvard.edu/~lkreidberg/batman/tutorial.html) to fit a light transit curve to the data and recover planet parameters from Lissauer et al. (2012; https://ui.adsabs.harvard.edu/abs/2012ApJ...750..112L/abstract)

7.1 Raw Light Curve (chose subset of phase interval to zoom in on dip)

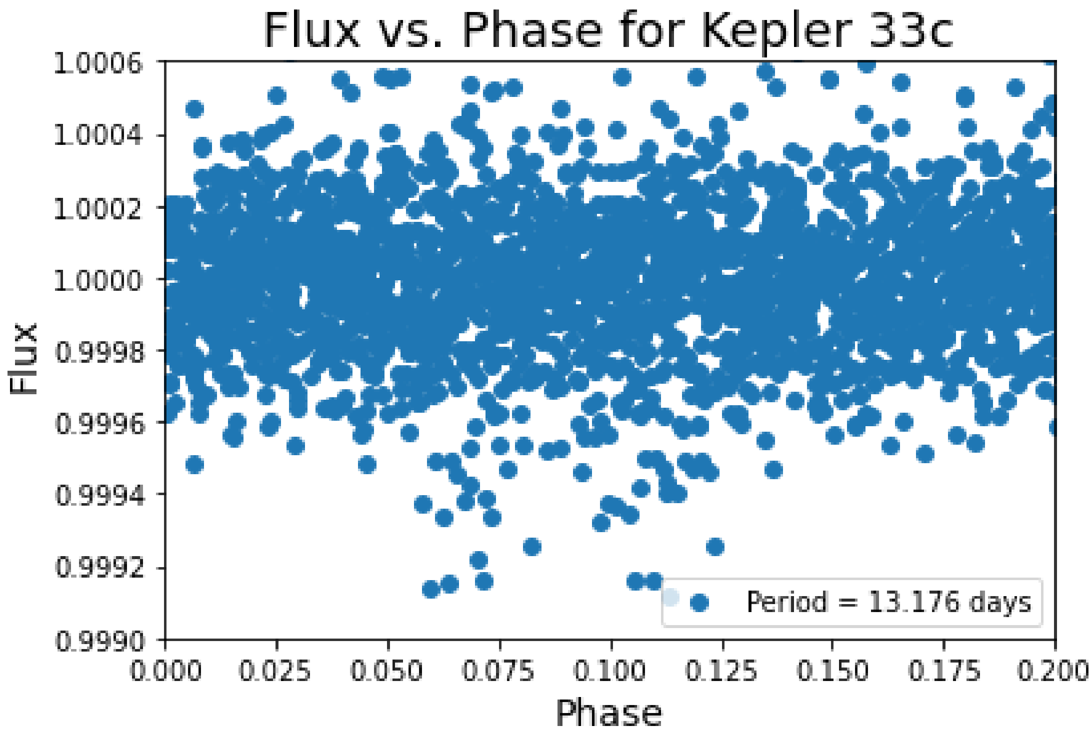
[35]:
```
# Initialize my Periodicity class and plot the light curve
Kepler33c = Periodicity('Kepler33.dat','Kepler␣
↪33c',['col0','col1','col2'],5000,contiguous='Bookend',period=13.17562)
```

[36]:
```
time,rel_flux,err = Kepler33c.LightCurve(plot=True,xaxis='Time')
```

```
No handles with labels found to put in legend.
```




```
[37]: phase,rel_flux,err = Kepler33c.LightCurve(plot=True,xaxis='Phase')
```



[110]:
```
batman_time,batman_flux = BatmanModel(P=13.17562,rad_pl=0.023,a=13.8,i=88.
↪19,e=0.2,w=0.0,coeff=[0.4899, 0.1809],plot=False)
```

[111]:
```
# Cut real data to zoom in on transit
phase_cut = [(i,x) for (i,x) in enumerate(phase) if x <= 0.2]
cuts = [x[0] for x in phase_cut]
phase_cuts = [x[1] for x in phase_cut]
rel_flux_cuts = []
for index in cuts:
    flux = rel_flux[index]
    rel_flux_cuts.append(flux)

# Plot real phased data and Batman model
fig,ax = plt.subplots()
ax.scatter(phase_cuts,rel_flux_cuts,label='Phased Data')
ax.scatter(batman_time,batman_flux,label='Batman Model')
ax.get_yaxis().get_major_formatter().set_useOffset(False)
ax.set_ylim(0.999,1.0005)
ax.set_xlabel('Phase',fontsize=18)
ax.set_ylabel('Relative Flux',fontsize=18)
ax.set_title('Modeling Kepler 33c Transit',fontsize=18)
ax.legend()
```

[111]: <matplotlib.legend.Legend at 0x7fdb8d17fc88>