Nikhil Patten
30 September 2022
Dr. Moe
ASTR5420

99%

Please show all work. If you collaborate with other students, write their names at the top of your homework. Please also write the total number of hours you spent on this. If you use any code or plotting routines, print those out and attach them to your hand-written solutions.

Collaborated with Alex

This took me approximately 16 hours to complete. thanks for putting in the hard work

1. **(15%) Assuming a grey atmosphere, calculate the average effective temperature of the Sun assuming the effective temperature near the center of the Sun's disk is $T_{eff}\,(u=1) = 6380$ K while the effective temperature near the solar limb is $T_{eff}\,(u=0) = 5080$ K. Your answer should be within 30 K of the true value of 5778 K (the Sun does not have perfect grey atmosphere).**

15/15

$$\frac{I(0,u)}{I(0,1)} = \frac{3}{5}\left[u + \frac{2}{3}\right]$$

$$I(0,u) \propto T(u)^4$$

$$\frac{T(u)^4}{T_{u=1}^4} = \frac{3}{5}\left[u + \frac{2}{3}\right]$$

$$\frac{T(u)}{T_{u=1}} = \left(\frac{3}{5}\left[u + \frac{2}{3}\right]\right)^{0.25}$$

$$T(u) = T_{u=1}\left(\frac{3}{5}\left[u + \frac{2}{3}\right]\right)^{0.25}$$

$$< T_{eff} > = \frac{\int_{u_1}^{u_2} T(u)}{u_2 - u_1}$$

$$< T_{eff} > = T_{u=1}\int_0^1 \left(\frac{3}{5}\left[u + \frac{2}{3}\right]\right)^{0.25} du \left(\frac{1}{1-0}\right)$$

$$< T_{eff} > = T_{u=1}\int_0^1 \left(\frac{3}{5}\left[u + \frac{2}{3}\right]\right)^{0.25} du$$

$$v = \frac{3}{5}\left[u + \frac{2}{3}\right], dv = \frac{3}{5}du$$

$$< T_{eff} > = T_{u=1}\int_0^1 v^{1/4}\frac{5}{3}dv$$

$$< T_{eff} > = \frac{5}{3}T_{u=1}\int_0^1 v^{1/4}dv$$

$$< T_{eff} > = \frac{5}{3}T_{u=1}\left[\frac{4}{5}v^{5/4}\Big|_0^1\right]$$

$$< T_{eff} > = \frac{4}{3}T_{u=1}\left[\left(\frac{3}{5}\left[u + \frac{2}{3}\right]\right)^{5/4}\Big|_0^1\right]$$

$$< T_{eff} > = \frac{4}{3}T_{u=1}\left[1 - \left(\frac{2}{5}\right)^{5/4}\right]$$

$$< T_{eff} > = \frac{4}{3}\left[1 - \left(\frac{2}{5}\right)^{5/4}\right](6380)\ \text{K}$$

Code:

In [6]:
```
T_1 = 6380
T_effavg = (4 /3) *(1 -(2 /5) **(5 /4)) *T_1
print('Average effective temperature across entire Sun: '\
    +format(T_effavg, '.2E') +' K')
```

```
Average effective temperature across entire Sun: 5.80E+03 K
```

$$\boxed{< T_{eff} >= 5800 \text{ K}}$$

+20/20

2. **(20%) Given a 3D Maxwell velocity distribution (Equation 4.51 in book):**
$f\left(V\right)dV = 4\pi\left(\frac{m}{2\pi kT}\right)^{3/2}V^2e^{-\frac{mV^2}{2kT}}dV$

(a) **Calculate the average speed $< V >$ as defined in Equation 4.52.**

$$\overline{V} = \int_0^\infty V f\left(V\right)dV$$
$$\overline{V} = \int_0^\infty 4\pi\left(\frac{m}{2\pi kT}\right)^{3/2}V^3\exp\left(-\frac{mV^2}{2kT}\right)dV$$
$$\overline{V} = 4\pi\left(\frac{m}{2\pi kT}\right)^{3/2}\int_0^\infty V^3\exp\left(-\frac{mV^2}{2kT}\right)dV$$

From Wikipedia (Lists of integrals of exponential functions), for $n = 2k + 1$:

$$\int_0^\infty x^n\exp\left(-ax^2\right)dx = \frac{k!}{2\left(a^{k+1}\right)}$$

Apply to above equation.

$$n = 3, k = 1, a = \frac{m}{2kT}$$
$$\overline{V} = 4\pi\left(\frac{m}{2\pi kT}\right)^{3/2}\frac{1!}{2}\left(\frac{m}{2kT}\right)^{-2}$$
$$\overline{V} = \frac{4\pi}{2}\left(\frac{1}{\pi}\right)^{3/2}\left(\frac{m}{2kT}\right)^{-1/2}$$
$$\overline{V} = 2\left(\frac{1}{\pi}\right)^{1/2}\left(\frac{2kT}{m}\right)^{1/2}$$

$$\boxed{\overline{V} = \sqrt{\frac{8kT}{m\pi}}}$$

45 (b) **Calculate the most probable speed V_0 as defined in Equation 4.53.**

46 $$\left.\frac{\partial f}{\partial V}\right|_{V=V_0} = 0$$

47 $$\frac{\partial f}{\partial V} = 4\pi \left(\frac{m}{2\pi kT}\right)^{3/2} \left[-V^2 \left(\frac{mV}{kT}\right) \exp\left(\frac{-mV^2}{2kT}\right) + 2V \exp\left(-\frac{mV^2}{2kT}\right)\right]$$

48 $$0 = 4\pi \left(\frac{m}{2\pi kT}\right)^{3/2} \left[-V_0^2 \left(\frac{mV_0}{kT}\right) \exp\left(\frac{-mV_0^2}{2kT}\right) + 2V_0 \exp\left(-\frac{mV_0^2}{2kT}\right)\right]$$

49 $$0 = -V_0^2 \left(\frac{mV_0}{kT}\right) \exp\left(\frac{-mV_0^2}{2kT}\right) + 2V_0 \exp\left(-\frac{mV_0^2}{2kT}\right)$$

50 $$0 = V_0 \exp\left(-\frac{mV_0^2}{2kT}\right) \left[2 - V_0 \left(\frac{mV_0}{kT}\right)\right]$$

51 $$0 = 2 - V_0^2 \frac{m}{kT}$$

52 $$V_0^2 \frac{m}{kT} = 2$$

53 $$V_0^2 = \frac{2kT}{m}$$

54 $$\boxed{V_0 = \sqrt{\frac{2kT}{m}}}$$

55 Code used for rest of problem:

```
In [117]: def f_V(m, T, V):
              '''Function to return the velocity distribution
              at a given temperature for a particle.
              Inputs:
              ------------------------------------------------
              m: Mass of the particle in kg. -float.
              T: Temperaure in Kelvin. -float
              V: Velcoities to run distribution over. -np.array
              Returns:
              ------------------------------------------------
              f_V: Velocity distribution for inputted parameters
              -np.array'''
              k = 1.38e-23
              # NP Boltzmann constant in kgs units
              f_V = V **2 *np.exp(-1 *(m *V **2) /(2 *k *T))\
                  *4 *np.pi *((m) /(2 *np.pi *k *T)) **1.5
              return f_V

          def v0(T, m):
              '''Function to return the the most probable speed
              at a given temperature for a particle.
              Inputs:
              ------------------------------------------------
              m: Mass of the particle in kg. -float.
              T: Temperaure in Kelvin. -float
              Returns:
              ------------------------------------------------
              v0: Most probable speed in m/s for inputted parameters.
              -float'''
              k = 1.38e-23
              # NP Boltzmann constant in kgs units
              v0 = np.sqrt((2 *k *T) /(m))
              return v0

          def vbar(T, m):
              '''Function to return the average speed
              at a given temperature for a particle.
              Inputs:
              ------------------------------------------------
              m: Mass of the particle in kg. -float.
              T: Temperaure in Kelvin. -float
              Returns:
              ------------------------------------------------
              v_bar: Average speed in m/s for inputted parameters.
              -float'''
              k = 1.38e-23
              # NP Boltzmann constant in kgs units
              v_bar = np.sqrt((8 *k *T) /(np.pi * m))
              return v_bar

          def vesc(M, R):
              '''Function to return the escape velocity
              at a given temperature for a particle.
              Inputs:
              ------------------------------------------------
              M: Mass of the planet in kg. -float.
              T: Radius of the planet. -float
              Returns:
              ------------------------------------------------
              v_esc: Escape velocity in m/s for inputted parameters.
              -float'''
              G = 6.67e-11
              # NP Gravatational constant in kgs units
              v_esc = np.sqrt(2 *G *M /R)
              return v_esc
```

(c) **Given Earth's atmosphere has mean $T = 285$ K, what fraction of molecular hydrogen H_2 exceeds Earth's escape velocity $v_{esc} = \sqrt{2GM_{\oplus}/R_{\oplus}}$?**

$$f_{esc} = \frac{\int_{v_{esc}}^{\infty} f dV}{\int_0^{\infty} f dV}$$

$$\boxed{f_{esc} = 6.74 \times 10^{-23}}$$

Code:

In [119]:
```python
fescHEarth = np.trapz(f_VHEarth[iiE],\
    V[iiE])/np.trapz(f_VHEarth, V)
# NP Calculating fraction of escaping H_2
print('Fraction of escaping H_2: '\
    +format(fescHEarth, '.2E'))
# NP Printing result
fescOEarth = np.trapz(f_VOEarth[iiE],\
    V[iiE])/np.trapz(f_VOEarth, V)
# NP Calculating fraction of escaping O_2
print('Fraction of escaping O_2: '\
    +format(fescOEarth, '.2E'))
# NP Printing result
```

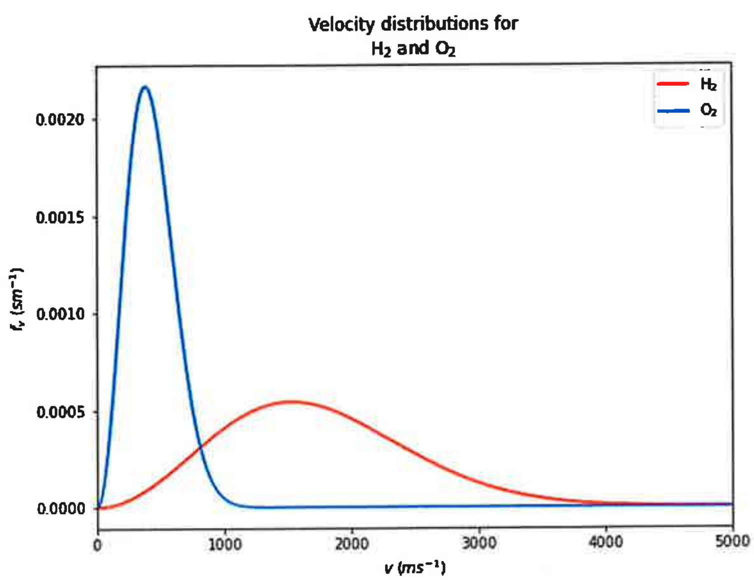
```
Fraction of escaping H_2: 6.74E-23
Fraction of escaping O_2: 0.00E+00
```

(d) **What fraction of molecular oxygen O_2 exceeds Earth's escape velocity?**

$$f_{esc} = \frac{\int_{v_{esc}}^{\infty} f dV}{\int_0^{\infty} f dV}$$

$$\boxed{f_{esc} = 0}$$

Plot:



(e) **Given Jupiter's atmosphere has mean T = 120 K, what fraction of molecular hydrogen H_2 exceeds Jupiter's escape velocity $v_{esc} = \sqrt{2GM_J/R_J}$.**

$$f_{esc} = \frac{\int_{v_{esc}}^{\infty} f dV}{\int_0^{\infty} f dV}$$

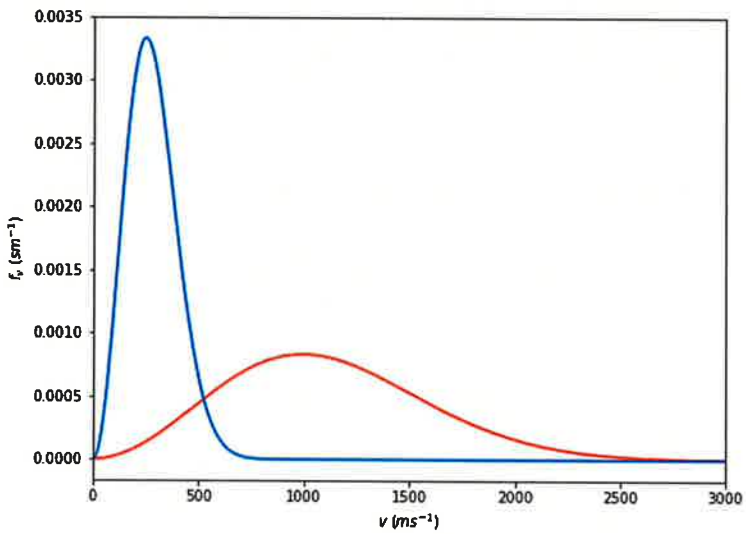
$$\boxed{f_{esc} = 0}$$

Code:

```
In [674]: print('Hydrogen escape fraction: ' +format(np.trapz\
              (f_VHJ[iiJ], V[iiJ])/np.trapz(f_VHJ, V), '.2E'))
          print('Oxygen escape fraction: ' +format(np.trapz\
              (f_VOJ[iiJ], V[iiJ])/np.trapz(f_VOJ, V), '.2E'))

          Hydrogen escape fraction: 0.00E+00
          Oxygen escape fraction: 0.00E+00
```

Plot:



Plot Code:

```
In [675]: M_J = 1.9e27
          # NP Earth's mass
          R_J = 7.15e7
          # NP Calculating Earth's Radius
          T_J = 120
          # NP Earth's temperature
          v_escJ = vesc(M_J, R_J)
          f_VHJ = f_V(mH2, T_J, V)
          f_VOJ = f_V(mO2, T_J, V)
          iiJ = V > v_escJ
          f = plt.figure(figsize = [8,6])
          plt.plot(V, f_VHJ, 'r')
          plt.plot(V, f_VOJ, 'b')
          plt.xlim(0, 3000)
          plt.xlabel(r'$v$ $(m s^{-1})$')
          plt.ylabel(r'$f_v$ $(s m^{-1})$')
          plt.savefig('/d/www/nikhil/public_html/'
              'ASTR5420/images/Jupitervdistrib.png')
```

This question should demonstrate why warm, small, terrestrial planets near the Sun have lost their hydrogen atmospheres but retain heavier molecules, and why cold, massive Jovian planets have retained most of their hydrogen.

10/10

3. **(10%) The lifetime of an electron in the $n = 2$ excited state of hydrogen is 10^{-8} s before it spontaneously decays to the $n = 1$ ground state. Compute the natural line width of Lyman α in units of $\Delta\lambda$ (Å), $\Delta\lambda/\lambda = v/c$, and v (cm s^{-1}). What resolution $R = \lambda/\Delta\lambda$ spectrograph would you need to resolve natural line broadening?**

Calculate Full-width at half-maximum.

$$\phi_\nu = \frac{\frac{\Gamma}{4\pi}}{(\nu-\nu_0)^2 + \left(\frac{\Gamma}{4\pi}\right)^2}$$

$$\phi_{\nu,max} = \frac{4\pi}{\Gamma}$$

$$\frac{1}{2}\left(\frac{4\pi}{\Gamma}\right) = \frac{\frac{\Gamma}{4\pi}}{(\nu-\nu_0)^2 + \left(\frac{\Gamma}{4\pi}\right)^2}$$

$$\frac{2\pi}{\Gamma} = \frac{\frac{\Gamma}{4\pi}}{(\nu-\nu_0)^2 + \left(\frac{\Gamma}{4\pi}\right)^2}$$

$$1 = \frac{\frac{\Gamma^2}{8\pi^2}}{(\nu-\nu_0)^2 + \left(\frac{\Gamma}{4\pi}\right)^2}$$

$$(\nu-\nu_0)^2 + \left(\frac{\Gamma}{4\pi}\right)^2 = 2\left(\frac{\Gamma}{4\pi}\right)^2$$

$$(\nu-\nu_0)^2 = \left(\frac{\Gamma}{4\pi}\right)^2$$

$$\nu_2 - \nu_0 = \frac{\Gamma}{4\pi}$$

$$\nu_2 = \frac{\Gamma}{4\pi} + \nu_0$$

$$\nu_1 - \nu_0 = -\frac{\Gamma}{4\pi}$$

$$\nu_1 = -\frac{\Gamma}{4\pi} + \nu_0$$

$$\Delta\nu = \nu_2 - \nu_1$$

$$\Delta\nu = \left(\frac{\Gamma}{4\pi} + \nu_0\right) - \left(-\frac{\Gamma}{4\pi} + \nu_0\right)$$

$$\Delta\nu = \frac{\Gamma}{2\pi}$$

Convert this width into wavelengths.

$$\nu = \frac{c}{\lambda}$$

$$\partial\nu = -\frac{c}{\lambda^2}\partial\lambda$$

$$|\Delta\nu| = |-\frac{c}{\lambda^2}\Delta\lambda|$$

$$\Delta\nu = \frac{c}{\lambda^2}\Delta\lambda$$

$$\Delta\lambda = \frac{\lambda^2}{c}\Delta\nu$$

$$\Delta\lambda = \frac{\lambda^2}{c}\frac{\Gamma}{2\pi}$$

$$\Delta\lambda = \frac{(1216\times10^{-10})^2}{3\times10^8}\frac{\frac{1}{10^{-8}}}{2\pi}\times10^{10}\ \text{Å}$$

$$\boxed{\Delta\lambda = 7.84\times10^{-6}\ \text{Å}}$$

Code:

```
In [16]: lambda1 = 1216e-10
         # NP Lyman alpha wavelength in meters
         c = 3e8
         # NP Speed of light in m/s
         Gamma = 1 /(10**-8)
         # NP Radiative decay constant for Lyman alpha
         dlambda = (lambda1**2) /(c) *(Gamma) /(2 *np.pi)\
             *1e10
         # NP Calculating wavelength of FWHM of this transition
         print('Width in angstroms: ' +format(dlambda, '.2E'))
         # NP Printing result

         Width in angstroms: 7.84E-06
```

Find $\frac{\Delta\lambda}{\lambda}$.

$$\frac{\Delta\lambda}{\lambda} = \frac{7.84 \times 10^{-6}}{1216}$$

$$\boxed{\frac{\Delta\lambda}{\lambda} = 6.45 \times 10^{-9}}$$

Code:

```
In [23]: dldivl = dlambda /lambda1
         # NP Calculating dl/l
         print('dlmabda/lambda: ' +format(dldivl, '.2E'))
         # NP Printing result

         dlmabda/lambda: 6.45E-09
```

Find v in cm s^{-1}.

$$\frac{v}{c} = \frac{\Delta\lambda}{\lambda}$$

$$v = c\frac{\Delta\lambda}{\lambda}$$

$$v = \left(3 \times 10^{10}\right)\left(6.45 \times 10^{-9}\right) \text{ cm s}^{-1}$$

$$\boxed{v = 194 \text{ cm s}^{-1}}$$

Code:

```
In [31]: v_1 = c *dldivl
         # NP Calculating speed for this transition
         print('v: ' +format(v_1 *1e2, '.2E') +' cm/s')
         # NP Printing result in cm/s

         v: 1.94E+02 cm/s
```

Find required resolution.

$$R = \frac{\lambda}{\Delta\lambda}$$

$$R = \frac{1216}{7.84 \times 10^{-6}}$$

$$\boxed{R = 1.55 \times 10^8}$$

Code:

```
In [32]: R = lambda1 /dlambda
         # NP Calculating required resoltion to see broadening
         print('R: ' +format(R, '.2E'))
         # NP Printing result

R: 1.55E+08
```

+20/20

4. **(20%) Curve of Growth.**

   (a) **For small $\tau(\nu_0) < 1$ at line center, show that the equivalent width $W \propto N$. Hint: Doppler core dominates W; Taylor expand $e^{-\tau(\nu)}$ where $\tau(\nu)$ is a Gaussian profile.**

$$e^{-\tau(\nu)} \approx 1 - \tau(\nu)$$

$$1 - \tau(\nu) = 1 - N\sigma \exp\left(\frac{-(\nu-\nu_0)^2}{2\sigma^2}\right)$$

$$W = \int_0^\infty 1 - \left(1 - \exp\left(-\frac{(\nu-\nu_0)^2}{2\sigma^2}\right)\right) d\nu$$

$$W = N\sigma \int_0^\infty \exp\left(-\frac{(\nu-\nu_0)^2}{2\sigma}\right) d\nu$$

$$\boxed{W \propto N}$$

   (b) **For large $\tau(\nu_0) > 10$ at line center, show that the equivalent width $W \propto N^{1/2}$. Hint: Lorentzian wings dominate W; integrate $\left[1 - e^{-\tau(\nu)}\right] d\nu$ where $\tau(\nu)$ is a Lorentzian profile in the limit $|\nu - \nu_0| >> \Gamma$.**

$$W = \int_0^\infty 1 - \exp(-\tau(\nu))\, d\nu$$

$$W = \int_0^\infty 1 - \exp\left(-N\sigma \frac{\gamma^2}{(\nu-\nu_0)^2 + \gamma^2}\right) d\nu$$

$$W \approx \int_0^\infty 1 - \exp\left(-N\sigma \frac{\gamma^2}{(\nu-\nu_0)^2}\right) d\nu$$

?

$$W = \int_0^\infty 1 - \exp\left(N\sigma.\nu^2\right) d\nu$$

$$W \propto \sqrt{N\sigma}$$

+34/35

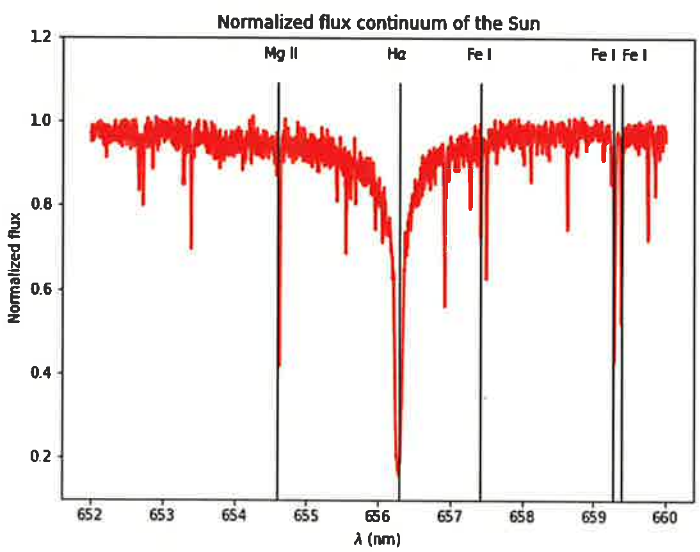
$$\boxed{W \propto \sqrt{N}}$$

5. **(35%) Download the normalized spectra of the Sun ($G2V$), epsilon Virginis ($G8III$), and Vega ($A0V$) across wavelengths $\lambda = 652 - 660$ nm from WyoCourses.**

   (a) **Plot the normalized spectrum of the Sun. Label at least 4 absorption lines in addition to $H\alpha$. To do this, go to the NIST Atomic Spectral Line Database (https://physics.nist.gov/PhysRefData/ASD/lines_form.html) and search for atomic transitions across the covered wavelength range. Consider only strong (allowed) transitions from neutral and singly ionized atoms with better than a B accuracy.**
   Read in spectra.

```
In [31]: names = ['Spectrum_Vega.txt',\
                  'Spectrum_Sun.txt',\
                  'Spectrum_EpsVir.txt']
         # NP Files of models to read in
         model = [np.loadtxt('/d/users/nikhil/Downloads/'
             +i, usecols = (0, 1)) for i in names]
         ws = [model[i].T[0] for i in range(len(model))]
         # NP Reading in wavelengths of models
         fs = [model[i].T[1] for i in range(len(model))]
         # NP Reading in SED value of model
```

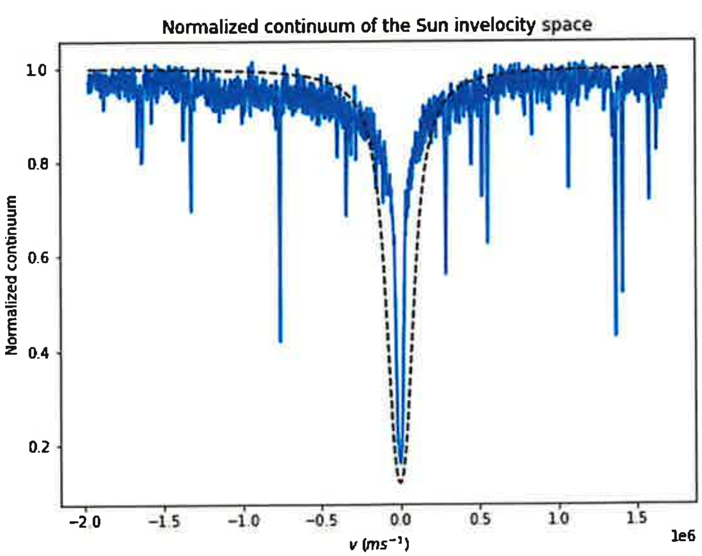
Plot:



Code:

In [74]:
```python
f = plt.figure(figsize = [8, 6])
# NP Making figure large
plt.plot(ws[1], fs[1], 'r')
# NP Plotting Sun values
plt.axvline(x = 656.3, ymin = 0.00, ymax = 0.9,\
    color = 'k', lw = 1)
plt.axvline(x = 659.2609, ymin = 0.00, ymax = 0.9,
    color = 'k', lw = 1)
plt.axvline(x = 659.38701, ymin = 0.00, ymax = 0.9,
    color = 'k', lw = 1)
plt.axvline(x = 657.4228, ymin = 0.00, ymax = 0.9,\
    color = 'k', lw = 1)
plt.axvline(x = 654.5973, ymin = 0.00, ymax = 0.9,\
    color = 'k', lw = 1)
# NP Plotting lines for different absorption features
plt.text(656.3 - 0.2, 1.15, r'H$\alpha$')
plt.text(659.2609 - 0.3, 1.15, r'Fe I')
plt.text(659.38701 - 0.0, 1.15, r'Fe I')
plt.text(657.4228 - 0.2, 1.15, r'Fe I')
plt.text(654.5973 - 0.2, 1.15, r'Mg II')
# NP Labeling features
plt.xlabel(r'$\lambda$ (nm)')
# NP Labeling x-axis
plt.ylabel('Normalized flux')
# NP Labelin y-axis
plt.title('Normalized flux continuum of the Sun')
# NP Labeling figure
plt.ylim(0.1, 1.2)
# NP Changing y-bounds to see labels
plt.savefig('/d/www/nikhil/public_html/ASTR5420/images'
    '/Sunspec.png')
# NP Saving figure
```

(b) **Plot the spectrum of the sun in velocity space $v = c\Delta\lambda/\lambda$ relative to $H\alpha$ (656.28 nm). By eye, fit a velocity profile $e^{-\tau(v)}$ to the normalized flux where $\tau(v; v_0, \sigma, \gamma)$ is a Voigt function with line center v_0, Gaussian standard deviation σ, and Lorentzian damping factor γ. Overplot your best-fit Voigt function and report the three best-fit parameters.** Plot:



Code:

```
In [436]: v_space = [[c *((l - 656.28) /(l)) for l in w]\
              for w in ws]
          # NP Converting wavelengths to velocity space
          f = plt.figure(figsize = [8, 6])
          # NP Making figure larger
          plt.plot(v_space[1], fs[1], 'b')
          # NP Plotting Sun continuum in v space
          plt.plot(v_space[1], np.exp(-400000 \
              *(vp(v_space[1], 10000, 50000))), '--k')
          # NP Plotting best-fit Voigt
          plt.xlabel(r'$v$ ($ms^{-1}$)')
          # NP Labeling x-axis
          plt.ylabel('Normalized continuum')
          # NP Labeling y-axis
          plt.title('Normalized continuum of the Sun in'
              'velocity space')
          # NP Labeling Plot
          plt.savefig('/d/www/nikhil/public_html/ASTR5420'
              '/images/Sunspecv.png')
          # NP Saving figure
```

(c) **Explain why $H\alpha$ is much broader than the other absorption lines. Estimate the effective temperature of the Sun based on the $H\alpha$ Gaussian dispersion σ.**
$H\alpha$ is so broad is because Doppler-broadening is inversely proportional to the square root of the mass. Since Hydrogen much less massive than other metals in the Solar spectrum, $H\alpha$ appears so broad.

$$\sigma = 10000$$
$$\sigma = \sqrt{\frac{kT}{m}}$$
$$\sigma^2 = \frac{kT}{m}$$
$$T = \frac{\sigma^2 m}{k}$$
$$\boxed{T = 12100}$$

Code:

```
In [441]: T = (10000) **2 *(mH2 /2) /(1.38e-23)
          # NP Calculating temperature
          print('Temperature: ' +format(T, '.2E') +' K')
          # NP Printing result

          Temperature: 1.21E+04 K
```

(d) **Calculate the equivalent width W of $H\alpha$ (integrating the actual data is sufficient).**
Code:

```
In [442]: W = np.trapz(1-fs[1], ws[1])
          # NP Calculating equivalent width
          print('Equivalent width: ' +format(W, '.2E') +\
              ' nm')
          # NP Printing result

          Equivalent width: 6.58E-01 nm
```

$$\boxed{W = 0.658 \text{ nm}}$$

great!

(e) **Suppose your spectrograph has a poor resolution with a Gaussian line-split function of $\sigma_{res} =$ 100 km/s. What is the resolution R of the spectrograph? Convolve the spectrum of the Sun with this line-split function. Plot the convolved spectrum on top of the original spectrum (both in velocity space but in different colors). Describe the differences in the $H\alpha$ profile, i.e., does σ of the Voigt profile change?**

Calculate convolution in pixel space:

$$\sigma_{pix} = \sigma_v p \tag{1}$$

Where p is pixel scale.

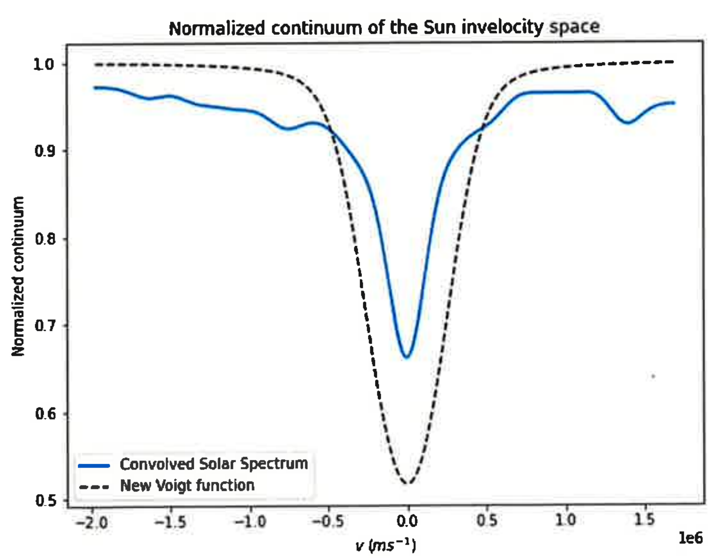
$$\boxed{\sigma_{pix} = 219 \text{ pix}}$$

Code:

```
In [493]: print('Pixel scale: ' +format(len(v_space[1])\
              /(max(v_space[1]) -min(v_space[1])), '.2E')\
              +' pixels / m/s')
          # NP Calculating pixel scale
          print('Convolution for models: ' +format(100000/\
              ((max(v_space[1])-min(v_space[1]))/\
              (len(v_space[1]))), '.2E') +' pix')
          # NP Calculating convolution for models in pixel space

Pixel scale: 2.19E-03 pixels / m/s
Convolution for models: 2.19E+02 pix
```

Convolved spectrum:



Plotting code:

```
In [584]: f = plt.figure(figsize = [8, 6])
          # NP Making figure larger
          sun_convol = gaussian_filter(fs[1], sigma=219)
          # NP Convolving Solar spectrum
          plt.plot(v_space[1], sun_convol, '-b',\
              label = 'Convolved Solar Spectrum')
          # NP Plotting convolved Solar spectrum
          plt.plot(v_space[1], np.exp(-400000 \
              *(vp(v_space[1], 200000, 50000))), '--k'\
              ,label = 'New Voigt function')
          # NP Plotting best-fit Voigt
          plt.xlabel(r'$v$ ($ms^{-1}$)')
          # NP Labeling x-axis
          plt.ylabel('Normalized continuum')
          # NP Labeling y-axis
          plt.title('Normalized continuum of the Sun in'
              'velocity space')
          # NP Labeling Plot
          plt.legend()
          plt.savefig('/d/www/nikhil/public_html/ASTR5420'
              '/images/convolvedsunspec.png')
          # NP Saving figure
```

The convolved spectrum looks smoothed over the Hα profile. The convolved spectrum has broad valleys corresponding to sharp absorption features in the original spectrum. A new Voigt function is overplotted the convolved spectrum. It is worth noting, the new Voigt function has a different σ than the Voigt function fitted to the high resolution Solar spectrum.

(f) **Calculate the equivalent width W of $H\alpha$ in your convolved spectrum. Is it different from part d? Why or why not?**
Code:

```
In [561]: W2 = np.trapz(1-sun_convol, v_space[1])
          # NP Calculating equivalent width
          print('Equivalent width: ' +format(W2, '.2E') +\
              ' m/s')
          # NP Printing result in m/s
          print('Pixel scale: ' +format(len(ws[1])\
              /(max(ws[1]) -min(ws[1])), '.2E')\
              +' pixels / nm')
          # NP Calculating pixel scale in nm
          print('Equivalent width in nm: ' \
              +format(W2 *2.19e-6, '.2E'))
          # NP Converting equivalent width to nm

Equivalent width: 3.01E+05 m/s
Pixel scale: 1.00E+03 pixels / nm
Equivalent width in nm: 6.59E-01
```

"area under the curve"

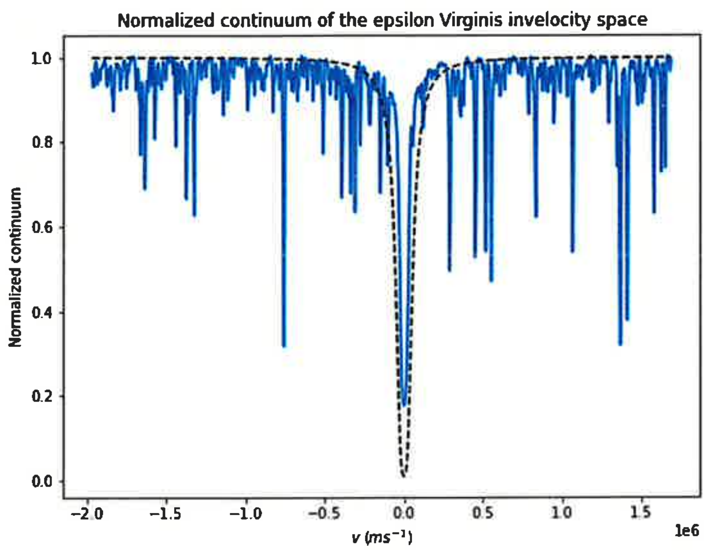
EW is conserved quantity (nothing to do w/ energy)

The equivalent width is the same! We would expect the same equivalent width in the convolved spectrum ~~because the equivalent width is related to the energy absorbed by the atom. Since this is the same atomic absorption line, we would therefore expect the same equivalenth width.~~

(g) **Repeat part (b) for epsilon Virginis. Report and discuss the differences in the parameters v_0, σ, and γ compared to the solar values.**
Code:

```
In [665]: v_space = [[c *((l - 656.28) /(l)) for l in w]\
              for w in ws]
          # NP Converting wavelengths to velocity space
          f = plt.figure(figsize = [8, 6])
          # NP Making figure larger
          plt.plot(v_space[2], fs[2], 'r')
          # NP Plotting epsilon virginis continuum in v space
          plt.plot(v_space[2], np.exp(-350000 \
              *(vp(v_space[2], 10000, 20000))), '--k')
          # NP Plotting best-fit Voigt
          plt.xlabel(r'$v$ ($ms^{-1}$)')
          # NP Labeling x-axis
          plt.ylabel('Normalized continuum')
          # NP Labeling y-axis
          plt.title('Normalized continuum of the epsilon Virginis in'
              'velocity space')
          # NP Labeling Plot
          plt.savefig('/d/www/nikhil/public_html/ASTR5420'
              '/images/epsvir.png')
          # NP Saving figure
```

Plot:



$$v_0 \approx v_0$$
$$\sigma_{vir} \approx \sigma_\odot$$
$$\gamma_{vir} \approx 0.4\gamma_\odot$$

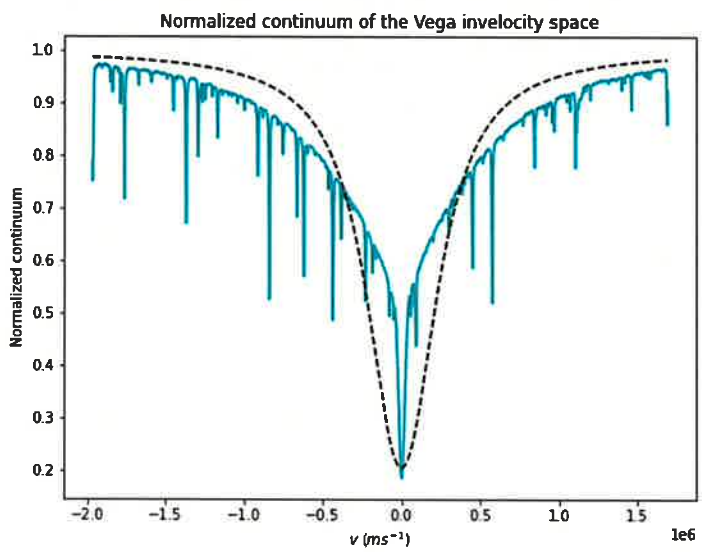
Not surprisingly, v_0 is the same is it is for the sun. This is because v_0 corresponds to the wavelength of $H\alpha$, which is the same for every star. We can also see that σ is roughly the same as it is for the Sun. This would make sense if epsilon Virginis is roughly the same temperature as the sun. The only difference between the Sun's spectrum and epsilon Virginis' spectrum is the γ parameter. I found that this value was roughly half that of the Sun. This corresponds to pressure broadening. Since this value was lower than that of the Sun, I would expect epsilon Virginis to have a surface gravity less than the Sun's surface gravity.

(h) **Repeat part (b) for Vega. Report and discuss the differences in the parameters v_0, σ, and γ compared to the solar values. Can you derive the effective temperature of Vega from σ? If**

not, explain what other property of Vega you can measure from σ.
Code:

```
In [670]: f = plt.figure(figsize = [8, 6])
          # NP Making figure larger
          plt.plot(v_space[0], fs[0], 'c')
          # NP Plotting Sun continuum in v space
          plt.plot(v_space[0], np.exp(-900000 \
              *(vp(v_space[0], 15000, 180000))), '--k')
          # NP Plotting best-fit Voigt
          plt.xlabel(r'$v$ ($ms^{-1}$)')
          # NP Labeling x-axis
          plt.ylabel('Normalized continuum')
          # NP Labeling y-axis
          plt.title('Normalized continuum of the Vega in'
              'velocity space')
          # NP Labeling Plot
          plt.savefig('/d/www/nikhil/public_html/ASTR5420'
              '/images/vegspec.png')
          # NP Saving figure
```

Plot:



$$v_0 \approx v_0$$

$$\sigma_{vir} \approx 1.5\sigma_\odot$$

$$\gamma_{vir} \approx 3.6\gamma_\odot$$

Again, v_0 is the same is it is for the sun. For Vega, it seems that σ is roughly 1.5 σ_{sun}. This means that Vega must be hotter than the Sun, which is expected. We can also see the γ for Vega is approximately 3.6 times γ for the Sun. From this, I would expect Vega to have a higher surface gravity compared to the Sun.
From before:

$$T = \frac{\sigma^2 m}{k}$$

Yes, Vega is hotter, but also rotating (above Kraft break), which dominates σ

$$\boxed{T \approx 27200 \text{ K}}$$

Code:

```
In [671]: Tvega = (15000) **2 *(mH2 /2) /(1.38e-23)
          # NP Calculating temperature
          print('Temperature: ' +format(Tvega, '.2E') +' K')
          # NP Printing result

          Temperature: 2.72E+04 K
```