In [1]:
```
%matplotlib inline
import numpy as np

from astropy import units as u
# from astropy.constants import c
from astropy.constants import k_B
```

In [2]:
```
%%javascript
MathJax.Hub.Config({
    TeX: { equationNumbers: { autoNumber: "AMS" } }
});
```

As discussed in our previous telecon, we can use equation 10.19 from Stahler & Palla's book

$$\delta u = \delta u_i \left(1 - \frac{i\omega}{n_i \langle \sigma_{in} u_i' \rangle}\right)^{-1} , \tag{1}$$

where $n_i \langle \sigma_{in} u_i' \rangle$ is "the frequency with which a given natural atom or molecule is struck by ions", δu is the perturbation on the neutral's velocity, and δu_i is the perturbation on the ion's velocity.

The other relation to use is equation 10.21 from Stahler & Palla's book

$$\frac{\omega^2}{k^2} = \frac{B_o^2}{4\pi\rho_o}\left(1 - \frac{i\omega}{n_i \langle \sigma_{in} u_i' \rangle}\right) , \tag{2}$$

which relates the ω and k.

Finally, equation 10.17 from Stahler & Palla's book relates the perturbation in the magnetic field as

$$\delta u_i = -\frac{\omega}{k}\frac{\delta B}{B_o} . \tag{3}$$

We estimate the velocity perturvations as the velocity dispersions derived from the spectral lines and therefore rewrite equation 1 as

$$\left|\frac{\delta u_i}{\delta u}\right|^2 = 1 + \left(\frac{\omega}{n_i \langle \sigma_{in} u_i' \rangle}\right)^2 \approx \left(\frac{\sigma_v(\mathrm{N_2H^+})}{\sigma_v(\mathrm{NH_3})}\right)^2 . \tag{4}$$

Now, equation 2 can also be rewritten as

$$\left|\frac{\omega^2}{k^2}\right| = \left(\frac{B_o^2}{4\pi\rho_o}\right)\left[1 + \left(\frac{\omega}{n_i \langle \sigma_{in} u_i' \rangle}\right)^2\right]^{1/2} \approx \frac{B_o^2}{4\pi\rho_o}\frac{\sigma_v(\mathrm{N_2H^+})}{\sigma_v(\mathrm{NH_3})} . \tag{5}$$

Combining equations 3, 4 and 5 we obtain

$$\frac{\delta B}{B_o} = \sqrt{\sigma_v(\mathrm{N_2H^+})\sigma_v(\mathrm{NH_3})}\frac{\sqrt{4\pi\rho_o}}{B_o}$$

or

$$\delta B = \sqrt{4\pi\rho_o}\sqrt{\sigma_v(\mathrm{N_2H^+})\sigma_v(\mathrm{NH_3})}$$

The term n_i is estimated using the ionization degree from Caselli et al. (2002)

$$x(e) = \frac{n_i}{n(\mathrm{H_2})} = 5.2\times10^{-6}\left(\frac{n(\mathrm{H_2})}{\mathrm{cm^{-3}}}\right)^{-0.56}$$

or

$$x(e) = 1.3\times10^{-5}\left(\frac{n(\mathrm{H_2})}{\mathrm{cm^{-3}}}\right)^{-0.5}$$

and the term

$$\langle \sigma_{in} u_i' \rangle \approx 10^{-9}\mathrm{cm^3 s^{-1}}$$

is approximated by the Langevin term.

Using the relation

$$\lambda = \frac{2\pi}{k} = \frac{2\pi}{\omega}\frac{\omega}{k}$$

and that

$$\frac{1}{\omega} = \frac{1}{n_i \langle \sigma_{in} u_i' \rangle}\sqrt{\frac{\sigma_v(\mathrm{NH_3})^2}{\sigma_v(\mathrm{N_2H^+})^2 - \sigma_v(\mathrm{NH_3})^2}}$$

then we can write

$$\lambda = \sqrt{\frac{\pi}{\rho_o}}\frac{B_o}{n_i \langle \sigma_{in} u_i' \rangle}\sqrt{\frac{\sigma_v(\mathrm{NH_3})\sigma_v(\mathrm{N_2H^+})}{\sigma_v(\mathrm{N_2H^+})^2 - \sigma_v(\mathrm{NH_3})^2}} \tag{1}$$

In [3]:
```
ef sigma_thermal(mu_mol, tk=10*u.K):
    """
    Returns the sound speed for temperature Tk and molecular weight mu.
    This is also used to determine the thermal velocity dispersion of
    a molecular transition.

    """
    return np.sqrt(k_B * tk/(mu_mol * u.u)).to(u.km/u.s)


ef density_ion(dens_all, do_Caselli=True):
    """
    1.3e-5 x n(H2)^{0.5}  (from McKee 1989) or 5.2e-6 x n*H2)^{0.44}
    """
    if do_Caselli:
        xe = 5.2e-6 * (dens_all/(u.cm**-3))**-0.56
    else:
        xe = 1.3e-5 * (dens_all/(u.cm**-3))**-0.5
    return xe*dens_all


ef get_omega(sigma_ion=0.1*u.km/u.s, sigma_neutral=0.08*u.km/u.s, density=1e6/u.cm**3, do_Caselli=True):
    """
    """
    n_i = density_ion(density, do_Caselli=do_Caselli)
    return (sig_in_v_i * n_i * np.sqrt((sigma_ion/sigma_neutral)**2 - 1)).decompose()


ef get_wavelength(Bfield=100*u.uG, sigma_ion=0.1*u.km/u.s, sigma_neutral=0.08*u.km/u.s,
                  n_H2=1e6/u.cm**3, do_Caselli=True):
    """
    """
    rho0 = (n_H2 * u.u * 2.8).cgs
    n_i = density_ion(n_H2, do_Caselli=do_Caselli).cgs
    return np.sqrt(np.pi/rho0) * (Bfield/(n_i*sig_in_v_i)) * np.sqrt(sigma_ion.cgs*sigma_neutral.cgs/(sigma_ion.cgs**2 - sigma_neutr

g_in_v_i = 1e-9*u.cm**3/u.s
auss_B = (u.g/u.cm)**(0.5)/u.s
quiv_B = [(u.G, gauss_B, lambda x: x, lambda x: x)]

_o = np.array([100, 150]) * u.uG
_sound = sigma_thermal(2.38, tk=10*u.K).cgs
gma_ion = 0.62 * c_sound
gma_neutral = 0.47 * c_sound
_H2 = 7e4 / u.cm**3
ho_o = (7e4 / u.cm**3 * u.u * 2.8).cgs
```

In [4]:
```
delta_B_Bo = (np.sqrt(sigma_ion*sigma_neutral * 4*np.pi * rho_o) / B_o.to((u.g/u.cm)**(1/2)/u.s, equivalencies=equiv_B))
print(delta_B_Bo)
```

```
[0.20404732 0.13603155]
```

In [5]:
```
delta_B = ((np.sqrt(sigma_ion*sigma_neutral * 4*np.pi * rho_o) / B_o) * B_o).to(u.uG, equivalencies=equiv_B)
print(delta_B)
```

```
[20.40473243 20.40473243] uG
```

This corresponds to a variation of 20 μG, which corresponds between 13 to 20% of the field.

If we want to estimate the wavelength then we will need to estimate $n_i\langle\sigma_{in}u_i'\rangle$ to derive ω and then obtain k.

In [6]:
```
(get_wavelength(Bfield=B_o, sigma_ion=sigma_ion, sigma_neutral=sigma_neutral, n_H2=n_H2)).to(u.pc*u.G*u.s*(u.cm/u.g)**0.5)
# .to(u.pc, equivalencies=equiv_B)
```

Out[6]: $[0.19081443,\ 0.28622165]\ \frac{\mathrm{G\,pc\,s\,cm^{1/2}}}{\mathrm{g^{1/2}}}$

In [7]:
```
(get_wavelength(Bfield=B_o, sigma_ion=sigma_ion, sigma_neutral=sigma_neutral, n_H2=n_H2, do_Caselli=False)).to(u.pc*u.G*u.s*(u.cm/
```

Out[7]: $[0.03908097,\ 0.058621454]\ \frac{\mathrm{G\,pc\,s\,cm^{1/2}}}{\mathrm{g^{1/2}}}$

In [ ]: