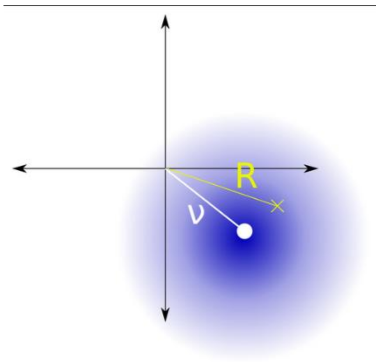
Assignment 1



1. Given a random distributed x and y at mean radius ν with standard deviation σ, the probability distribution of $r = \sqrt{x^2 + y^2}$ is described by Rician distribution:

   $$f(r, \nu, \sigma) = \frac{r}{\sigma^2} \exp\left(\frac{-(r^2+\nu^2)}{2\sigma^2}\right) I_0\left(\frac{r\nu}{\sigma^2}\right), \text{ where}$$

   $I_0(x)$ is modified Bessel function of first kind.
   This distribution has great importance in noise processing in astronomy polarization image processing. It shows that the combined signal r is higher than expected, ν .
   Hence,
   a) Given $\sigma = 1$, use bisection method to find the most probable r for $\nu = [0,10]$.
   b) Plot the result in (a) against ν
   Hint: you will need to use modules numpy, scipy.special.iv, matplotlib.pyplot

2. One of the near field diffraction approximation is Fresnel diffraction. The intensity is described by

   $$I = |\int p(x', y') e^{-i\pi[(x-x')^2+(y-y')^2]} dx'dy'|^2 = |g(x)|^2$$

   where p is the aperture shape, x' y' is the aperture coordinate, and x y is screen coordinate. Your should take the coordinate system where their origin lie in one line.
   a) For the case of slit aperture of width $2\sqrt{10}$ in x', and infinitely long in y' direction, **plot** the diffraction pattern. Then repeat for width $2\sqrt{0.1}$, $2\sqrt{0.5}$, $2\sqrt{1}$, $2\sqrt{5}$
   b) For the case of square aperture with side $2\sqrt{10}$, plot 2D intensity map.
   Hint: you will need modules scipy.special.fresnel, matplotlib.pyplot.imshow. Make sure you read the documentation of scipy.

3. Assume both m are equal, and all spring has the same k. Write down the potential energy of the system, then use the partial derivative of potential energy to find equation of motion of x and y. Assume x and y have the solution $x = x_0 e^{i\omega t}$, $y = y_0 e^{i\omega t}$. Use system of linear equation to solve for eigenvalues, eigenvector, then find the characteristic frequencies. You should use python to solve for eigenvalues and eigenvector.
   Hint: use numpy.linalg