Assignment # 5

Abraham Mathews

SC22M077

MS Astronomy and Astrophysics

Assignment 05

Integration (part II)

Submission open till 02nd November

Below e, m_e, and c *are fundamental constants; charge of electron, mass of electron, and speed of light respectively.*

✭ The single electron synchrotron power spectrum is given by

$$P(\nu, \gamma) = \frac{\sqrt{3}e^3 B}{2\pi m_e c^2} F(x), \qquad \ldots\ldots\ldots\ldots\ldots\ldots(1)$$

where ν is the frequency of radiation, γ is the Lorentz factor of the radiating electron, and B is the magnetic field.

✭ The dimensionless $F(x) = x \int_x^{\infty} d\zeta\, K_{5/3}(\zeta)$, $\qquad \ldots\ldots\ldots\ldots(2)$

where $x = \nu/\nu_{\rm syn}$ and $K_{5/3}(\zeta)$ is the Modified Bessel function of order $5/3$.

✭ The characteristic synchrotron frequency $\nu_{\rm syn} = \dfrac{3}{4\pi}\dfrac{\gamma^2 eB}{m_e c}$, $\qquad \ldots\ldots\ldots\ldots\ldots(3)$

with the same parameter definitions as in eqn(1).

✭ Consider a bunch of electrons having the following distribution of Lorentz factors $n(\gamma) \propto \left(\dfrac{\gamma}{10}\right)^{-2.5}$. Find the normalisation of this expression by assuming that the total number of electrons with Lorentz factors $10 \le \gamma \le 100$ is 2500 *(some random strange number, you are free to change this)*.

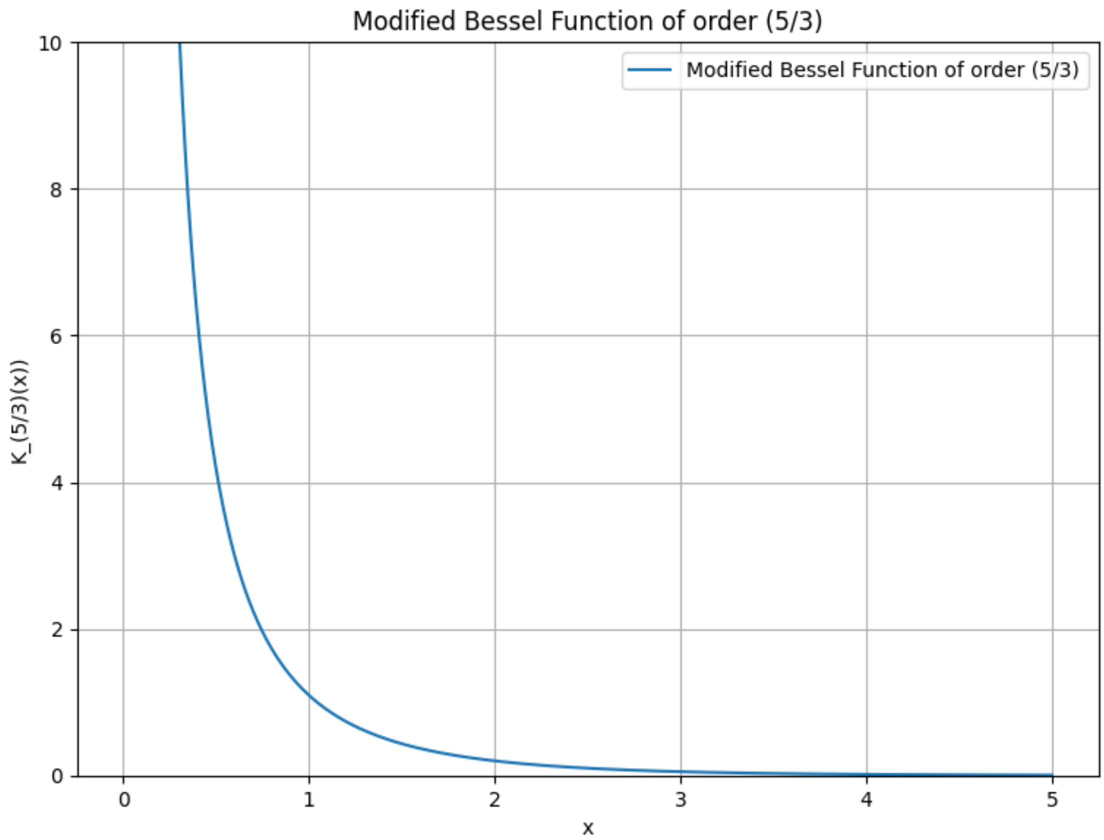
✭ Obtain the synchrotron spectral energy distribution $P(\nu)$ emitted by these electrons. That is, one has to basically do $P(\nu) = \int_{10}^{100} d\gamma\, n(\gamma) P(\nu, \gamma)$. $\qquad \ldots\ldots\ldots\ldots\ldots(4)$

✭ Assume B=1 Gauss.

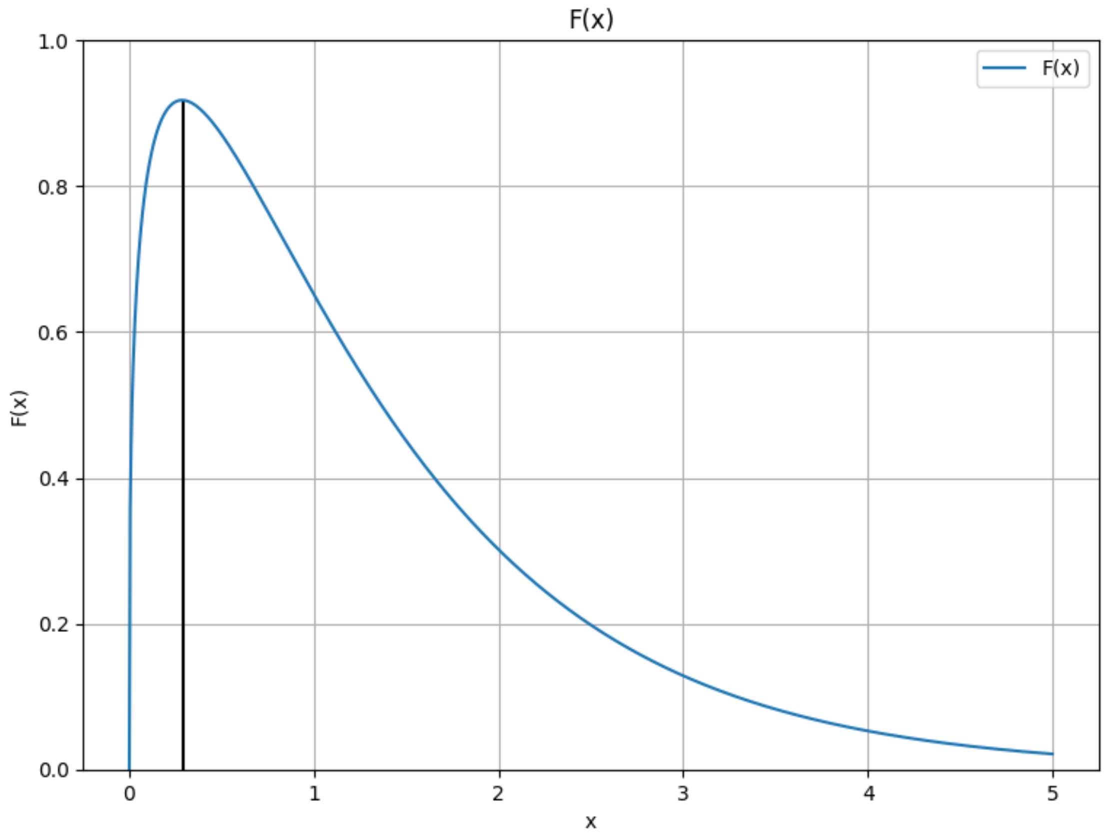
In [4]:

```
import numpy as np
from scipy import constants
from scipy import special
from scipy import integrate
import math
import matplotlib.pyplot as plt

# Constants
B = 1 * pow(10,-4) # Tesla = 1 Gauss
e = constants.e
me = constants.m_e
c = constants.c
pi = math.pi


###
# Plotting the modified Bessel Function of order (5/3)
plt.style.use('default')
fig = plt.figure(figsize = (9,6.5))
y = np.linspace(0,5,1000)
plt.plot(y,special.kv(5/3,y),label = "Modified Bessel Function of
order (5/3)")
plt.ylim(0, 10)
plt.legend()
plt.title("Modified Bessel Function of order (5/3)")
plt.xlabel("x")
plt.ylabel("K_(5/3)(x))")
plt.grid()
plt.show()
###

###
# Plotting F(x)
def besselFn(x):
    return(special.kv(5/3,x))
def integrandFn(x):
    result = integrate.quad(besselFn,x,np.inf)
    return(result)
def F(x):
    result = integrandFn(x)
    return(x*result[0])
x = np.linspace(0,5,1000)
F_array = np.array([]) #Create empty array, appending will start f
illing from index 0
for t in x:
    # print("t = ",t)
    # print("F(t) = ",F(t))
    F_t = F(t)
    #plt.scatter(t,F_t,color="black",marker='.')
    F_array = np.append(F_array,F_t)
#F_array = np.delete(F_array,[0])
#print("F_array = ",F_array)
fig = plt.figure(figsize = (9,6.5))
plt.vlines(0.29,0,np.amax(F_array))
plt.plot(x,F_array,label="F(x)")
plt.ylim(0, 1)
plt.title("F(x)")
plt.xlabel("x")
plt.ylabel("F(x)")
```

```
plt.grid()
plt.legend()
plt.show()
###


# Function P_single(v,gamma) spectrum for one electron
def P_single(v,gamma):
    v_syn = (3*pow(gamma,2)*e*B)/(4*pi*me*c)
    x=v/v_syn
    Nr = math.sqrt(3)*pow(e,3)*B*F(x)
    Dr = 2*pi*me*pow(c,2)
    result = (Nr/Dr)
    return(result)
#print(P_single(100,10))

###
# To find normalization factor N_o
def n(gamma):
    result = math.pow((gamma/10),-2.5)
    return(result)
integ_n = integrate.quad(n,10.0,100.0)
N_o = 2500/integ_n[0]  # Output of integration is the first elemen
t
print("N_o = ",N_o)
###

###
# function dP(v)
def dP(gamma,v):
    result = N_o*n(gamma)*P_single(v,gamma)
    return(result)

# function P(v) spectrum for the entire distribution of electrons
def P(v):
    v = v
    result = integrate.quad(dP,10,100,args=v)
    return(result[0])

# Plotting spectrum for frequency 10^2 to 10^8 Hz
v = np.logspace(2,8,5000)
P_dist_array = np.array([])
for t in v:
    P_t = P(t)
    #plt.scatter(t,P_t)
    P_dist_array = np.append(P_dist_array,P_t)

plt.style.use('default')
fig = plt.figure(figsize = (9,6.5))
plt.plot(np.log10(v),np.log10(P_dist_array),label ="SED")
plt.title("Synchrotron Spectral Energy Distribution")
plt.xlabel("log10(v)")
plt.ylabel("log10(P)")
plt.grid()
plt.legend()
plt.show()
###
```




```
<ipython-input-4-fe8578912393>:36: IntegrationWarning: The integra
l is probably divergent, or slowly convergent.
  result = integrate.quad(besselFn,x,np.inf)
```




```
N_o =  387.2457870126441
```

```
<ipython-input-4-fe8578912393>:106: RuntimeWarning: divide by zero
encountered in log10
  plt.plot(np.log10(v),np.log10(P_dist_array),label ="SED")
```