MCRAD1

February 27, 2019

In [150]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from scipy import reshape
def MC_rad_transfer(minz,maxz,stepz,w0,theta):
    #w0 = 0.8; # single-scattering albedo
    N = 1; # number of photons to trace
    ns = 1; # max number of scatters per photon #zeta control
    E = np.zeros(N*maxz); # initialize detector
    Eabs=np.zeros(N*maxz)
    Et=np.zeros(N*maxz)
#minz=0.1
#maxz=3
#stepz=1
#%how far does it go?
#l=-(1/c)*log(1-rand); %total path length Tow C
    for i in range(0,N):
        #z = np.arange(0.1,3,1) #initial position of the photon
        z = np.arange(minz,maxz,stepz)
        for k in range (0, maxz):
    #z = 0.1; # initial position depth
          w = 1; # initial photon weight
          #theta=(np.pi)/4 #default pi/2
          muz = np.cos(theta); # incident light direction (collimated)
          for j in range(ns):
            zeta=np.random.rand()
#    s = 1*(np.log(np.random.rand))/c; # geometric path length
            s= -1*np.log(1-zeta)
#     s=0.8147
# print(s)
            z = z + muz*s; # move photon
# print(z)
            if np.any(z<0):
              E+=1;
              break #count photons leaving out top
            w=w*w0; # absorb fraction of photon packet
            muz= 2*zeta-1; # isotopic scattering
            if np.any(z>s):
```

```
                    Et+=1; #photons transmitted
                    break
                  if (np.any(z>0) and np.any(z<s)):
#                   w=w*w0
                    if (zeta>w0):
                      Eabs+=1
                      break
                    if np.any(zeta<=w0):
                      muz= 2*zeta-1; #scattering angle

 #print(muz)
#print(Eabs)
#print(Et)
#print(E)
        Ref=E/N;
        Abs=Eabs/N;
        Transm=Et/N;
      #print((Ref))
      #print((Abs))
        Ref=Ref
 #       Abs=Abs
 #       Transm=Transm

 #       return Ref,Abs,Transm
        return Ref
```

In [109]:
```
MC = MC_rad_transfer(0,25,1,1,np.pi/4)# [minz,maxz,stepz,N,ns,w,theta]
print(MC)
```

```
(array([0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0.]), array([0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.]), array([25., 25., 25., 25., 25., 25., 25., 25., 25., 25.
       25., 25., 25., 25., 25., 25., 25., 25., 25., 25., 25., 25.]))
```

In [141]:
```
def gamma(wl,c0=0.04,k=0.2):
    alpha = [c0*tau*np.exp(-k*tau)for l in wl]
    return alpha
```

In [142]:
```
import numpy as np
import pandas as pd
import scipy
from math import pi
tau =np.arange(minz,maxz,stepz)
print(tau)
```

```
[ 0.1  1.1  2.1  3.1  4.1  5.1  6.1  7.1  8.1  9.1 10.1 11.1 12.1 13.1
 14.1 15.1 16.1 17.1 18.1 19.1 20.1 21.1 22.1 23.1 24.1 25.1 26.1 27.1
 28.1 29.1 30.1 31.1 32.1 33.1 34.1 35.1 36.1 37.1 38.1 39.1 40.1 41.1
```

```
 42.1 43.1 44.1 45.1 46.1 47.1 48.1 49.1]
```

```
In [149]: import numpy as np
          import pandas as pd
          import scipy
          from math import pi
          minz=0.1
          maxz=50
          stepz=1

          #tau =np.arange(minz,maxz,stepz)
          tau =np.arange(minz,maxz,stepz)
          #print(tau)
          RT=np.asarray(gamma(tau,c0=0.04,k=0.2))
          print((RT))
          VT=np.asarray(MC_rad_transfer(minz,maxz,stepz,1,np.pi/4))
          #print(VT)
          #OT=RT*VT
          OT=RT.dot(VT)
          #print(OT)
          #numerical value
          dy1 = np.trapz(OT,tau)
          #print(dy1)
```

```
[[0.00392079 0.03531083 0.05519193 ... 0.00015277 0.00012773 0.00010675]
 [0.00392079 0.03531083 0.05519193 ... 0.00015277 0.00012773 0.00010675]
 [0.00392079 0.03531083 0.05519193 ... 0.00015277 0.00012773 0.00010675]
 ...
 [0.00392079 0.03531083 0.05519193 ... 0.00015277 0.00012773 0.00010675]
 [0.00392079 0.03531083 0.05519193 ... 0.00015277 0.00012773 0.00010675]
 [0.00392079 0.03531083 0.05519193 ... 0.00015277 0.00012773 0.00010675]]
```