Problem 1.

^{147}Pm is a pure beta emitter with a 2.6234 yr half life. By using radioactive seeds, a radioisotope can be evenly distributed in a tumor site. Assume a tumor site with a density of $1\,\mathrm{g/cm^3}$ and a mass of 11 g. If the plan is to deliver 25 Gy of dose to the entire prostate, calculate the activity of the ^{147}Pm at the time of implantation into the prostate. Assume it has a biological half life similar to ^{131}I.

Solution

Decay information was obtained from `http://nucleardata.nuclear.lu.se/toi/nuclide.asp?iZA=610147` See calculations in attached code. Assuming all of the activity is captured in the prostate, the activity required is:

$$
\begin{aligned}
A &= Dm\frac{\lambda_b + \lambda_p}{E} \\
&= 5.10 \times 10^5\,\mathrm{Gy}
\end{aligned}
$$

Problem 2.

1×10^{-6} g of ^{59}Co is placed into the high flux reactor at ORNL. After 24 days of irradiation, what is the activity of ^{60}Co in the sample? How many atoms of ^{59}Co have been lost in that time period? Use a flux of 1×10^{15} thermal neutrons /cm^2/s.

Solution

Cross section obtained from the BNL Sigma page at `http://www.nndc.bnl.gov/sigma/index.jsp?as=59&lib=endfb7.1&nsub=10`. Calculations were performed in the attached code. From Turner, equation 4.40, the concentration of ^{60}Co is:

$$N_{60} = \frac{\lambda_{59} N_{59}}{\lambda_{60} - \lambda_{59}} \left(\mathrm{e}^{-} - \lambda_{59} t - \mathrm{e}^{-} - \lambda_{60} t \right)$$

Substituting the removal term $\phi\sigma_{59}$ for λ_{59}

$$\begin{aligned} &= \frac{\phi\sigma_{59} N_{59}}{\lambda_{60} - \phi\sigma_{59}} \left(\mathrm{e}^{-} - \phi\sigma_{59} t - \mathrm{e}^{-} - \lambda_{60} t \right) \\ &= 2.87 \times 10^{6}\,\mathrm{Bq} \end{aligned}$$

Problem 3.

What fluence of neutrons from a DT generator ($\mathrm{d} + t \rightarrow \mathrm{n} + {}^{4}\mathrm{He}$) is required to deliver a KERMA of 1 Gy?

Solution

Anderson Appendix 10 lists values for Kerma per fluence of neutrons in water. The energy of neutrons from a DT generator is 14.1 MeV. Interpolating between 10 MeV and 15 MeV (see attached code) and solving, we get:

$$
\begin{aligned}
\phi &= \frac{K}{K/\phi} \\
&= 1.45 \times 10^{6}\ /\mathrm{m}^2
\end{aligned}
$$

Problem 4. Anderson 10.11

The linear attenuation coefficient for ^{60}Co radiation in water is $6.5\,\mathrm{m}^{-1}$.

(a) Calculate the dose at points at depths 0.01 m, 0.05 m, 0.1 m, 0.2 m along the central axis for F of 0.8 m. Assume the maximum dose is 100 rad. Ignore scatter.

(b) Compare your calculations with the measured values in Appendix 11 for a 10×10 cm field. Calculate the dose attributable to scatter and the buildup factor at each depth.

Solution

Part (a)

Calculations performed in attached code. From Anderson equation 10.22:

$$D(d, F, A/P) = D(d_m, F, A/P)\frac{BF(d, A/P)}{BF(d_m, A/P)}\left[\frac{F + d_m}{F + d}\right]^2 \exp\{-\mu_{eff}(d - d_m)\} \qquad (1)$$

From Appendix 11, we determine that, since at $d = 0.5\,\mathrm{cm}$, $BF = 1$, $d_m = 0.5\,\mathrm{cm}$. We ignore the BF terms and calculate:

Depth (m)	Calculated	Measured	BF	BS Dose Contribution
0.01	95.610844	98.2	1.027080	2.589156
0.05	66.945713	78.5	1.172592	11.554287
0.10	43.144943	55.6	1.288679	12.455057
0.20	18.244145	27.2	1.490889	8.955855

Part (b)

Comparison with the values in Appendix 11 was performed in the table above and graphically in figure 1. The difference between the two was attributed to the backscatter terms that were neglected in equation 1.

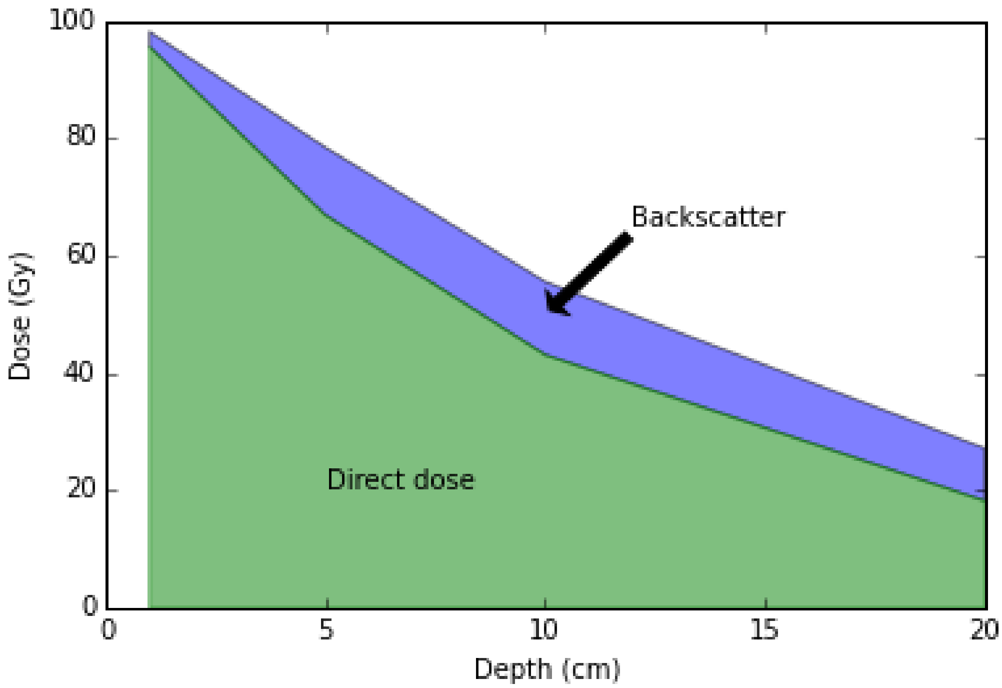


Figure 1: Dose contribution from direct and BS

NE551_homework_10

November 15, 2016

```
In [90]: %matplotlib inline
         import matplotlib.pyplot as plt
         import numpy as np
         import pandas as pd
         from scipy.interpolate import interp1d, UnivariateSpline
         from __future__ import division
```

1 Problem 1

```
In [91]: t_decay = 2.6234 * 365.241 * 24 * 60 * 60
         t_bio = 138 * 24 * 60 * 60
         t_effective = t_decay * t_bio / (t_decay + t_bio) * 24 * 60 * 60
         lambda_p = np.log(2) / t_decay
         lambda_b = np.log(2) / t_bio
         t_effective # seconds

Out[91]: 900474488372.5688
```

Energy information obtained from Lund/LBNL Cinderella site

```
In [92]: E = 224.1 * 1000. * 1.6e-19 # keV to J
         w = 1.

In [93]: 25. * (lambda_b + lambda_p) / E * .011

Out[93]: 510080.41059187689

In [94]: 25. * (lambda_b + lambda_p) / E

Out[94]: 46370946.417443357
```

2 Problem 2

Cross sections downloaded from the BNL Sigma web page

```
In [95]: cross_section = pd.read_csv('cobalt-60-n-gamma.txt')

In [96]: sigma_interp = interp1d(cross_section['27-Co-59(n'], cross_section[u'&gamm
         sigma_interp(0.0253)
```

```
Out[96]: array(37.2756)
```

The absorption cross section for Co-60 is assumed to be negligible.

```
In [97]: N_0_59 = 1e-6 * 6.022e23 / 60.
         N_0_59

Out[97]: 1.0036666666666666e+16

In [98]: def N_59(t):
             flux = 1e15
             cross_section = sigma_interp(0.0253) * 1e-24 # convert barns to cm^2
             return N_0_59 * np.exp(-flux * cross_section * t)

In [99]: def N_60(t):
             flux = 1e15
             cross_section = sigma_interp(0.0253) * 1e-24 # convert barns to cm^2

             decay_const = np.log(2.) / (5.2713 * 365.241 * 24. * 60. * 60.)

             numerator = flux * cross_section * N_59(t)
             denominator = decay_const - flux * cross_section

             time_factor = np.exp(-flux * cross_section * t) - np.exp(-decay_const

             return numerator * time_factor / denominator

In [100]: def activity(t):
              decay_const = np.log(2.) / (5.2713 * 365.241 * 24. * 60. * 60.)

              return decay_const * N_60(t)

In [101]: activity(24 * 24 * 60 * 60)

Out[101]: 2866880.7768756356
```

The activity is 2.87 * 10^6 Bq

```
In [102]: N_0_59 - N_59(24 * 24 * 60 * 60)

Out[102]: 746556887576438.0
```

7.47 * 10^14 atoms of Co-59 have been removed in that time

3 Problem 3

```
In [103]: E = [1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 1., 2., 3., 4., 5., 6., 7., 8., 9., 10
          kerma_per_fluence = [1.36e-11, 1.23e-10, 1.21e-9, 1.30e-8, 7.23e-8, 2.71e
          interp = interp1d(E, kerma_per_fluence)
          # interp = UnivariateSpline(E, kerma_per_fluence, k=5)
```

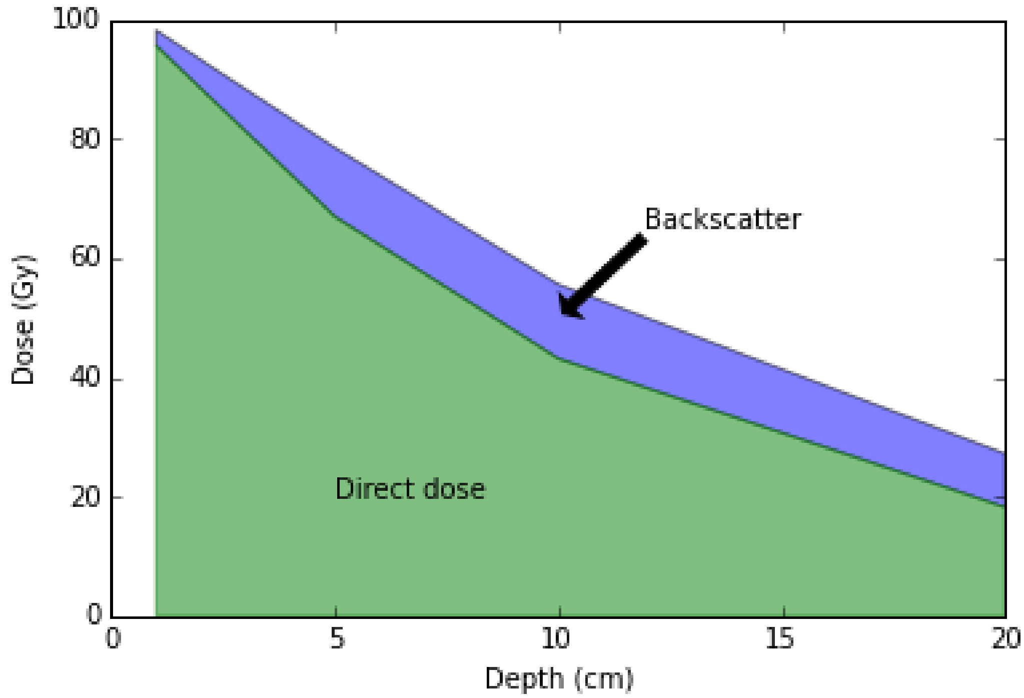
```
In [104]: phi = 1 / interp(14.1)

In [105]: phi

Out[105]: 1453319.3814672711
```

4 Problem 4

```
In [106]: data = pd.DataFrame(
              {
                  "Depth": [0.01, 0.05, 0.1, 0.2],
              }
          )

In [107]: def dose(depth):
              # Neglecting buildup factor (ignoring scatter)
              # Assuming, based on appendix 11, that d_m = 0.5cm = 0.005m
              mu = 6.5 # inverse meters
              F = 0.8 # meters
              D_max = 100 # rad
              d_m = 0.005 # meters

              ret = D_max * ((F + d_m) / (F + depth))**2 * np.exp(-mu * (depth - d_

              return ret

In [108]: data["Calculated"] = dose(depth)

In [109]: data["Measured"] = [98.2, 78.5, 55.6, 27.2]

In [110]: data["BF"] = data["Measured"] / data["Calculated"]

In [111]: data["BS Dose Contribution"] = data["Measured"] - data["Calculated"]

In [141]: data

Out[141]:
```

	Depth	Calculated	Measured	BF	BS Dose Contribution
0	0.01	95.610844	98.2	1.027080	2.589156
1	0.05	66.945713	78.5	1.172592	11.554287
2	0.10	43.144943	55.6	1.288679	12.455057
3	0.20	18.244145	27.2	1.490889	8.955855

```
In [139]: plt.fill_between(data["Depth"]*100, data["Calculated"], data["Measured"],
          plt.fill_between(data["Depth"]*100, 0, data["Calculated"], color="green",
          plt.text(0.05*100, 20, "Direct dose")
          plt.annotate("Backscatter", xy=(0.1*100, 50), xytext=(0.12*100, 65), arro
          plt.ylabel("Dose (Gy)")
          plt.xlabel("Depth (cm)")
          plt.legend(loc="upper right")
          plt.savefig('images/problem4.png')
          plt.show()
```


Backscatter
Direct dose
Dose (Gy)
Depth (cm)
0
20
40
60
80
100
0
5
10
15
20

```
In [ ]:
```