exercises_1_solutions

February 1, 2021

[6]:
```
import math as m
```

0.1 EM showers - Shower development in lead and iron

First we define some useful constants and extract relevant numbers from the table

[28]:
```
photon = 0
electron = 1
MeV = 1.
GeV = 1e3*MeV
TeV = 1e6*MeV
# Material properties
lead_Ec = 7.2*MeV
iron_Ec = 20.5*MeV
lead_Z = 82
iron_Z = 26
lead_X0 = 0.56 # cm
iron_X0 = 1.76 # cm
```

0.1.1 Shower maximum

The shower maximum in units of X_0 is given by the following formula:

$$t_{max}[X_0] = \ln(E_0/E_c) - \begin{pmatrix} 1 \text{ for electrons} \\ 0.5 \text{ for photons} \end{pmatrix}$$

where E_0 is the initial energy of the particle and E_c the critical energy of the material

[26]:
```
def shower_max(E0, Ec, particle=electron):
    tmax = m.log(E0/Ec)
    if particle==photon:
        tmax -= 0.5
    elif particle==electron:
        tmax -= 1
    else:
        raise RuntimeError('Unknown particle type')
    return tmax
```

Now we use the `shower_max` function to compute t_{max} for each case

[27]:
```
max_e_100GeV_Pb = shower_max(100*GeV, lead_Ec, electron)
max_g_1TeV_Pb = shower_max(1*TeV, lead_Ec, photon)
max_e_100GeV_Fe = shower_max(100*GeV, iron_Ec, electron)
max_g_1TeV_Fe = shower_max(1*TeV, iron_Ec, photon)
print('Shower maximums:')
print('  ele 100GeV, Pb:', max_e_100GeV_Pb, 'X0')
print('  pho 1TeV, Pb:', max_g_1TeV_Pb, 'X0')
print('  ele 100GeV, Fe:', max_e_100GeV_Fe, 'X0')
print('  pho 1TeV, Fe:', max_g_1TeV_Fe, 'X0')
```

```
Shower maximums:
  ele 100GeV, Pb: 8.53884443894822 X0
  pho 1TeV, Pb: 11.341429531942264 X0
  ele 100GeV, Fe: 7.492500578825865 X0
  pho 1TeV, Fe: 10.295085671819912 X0
```

The results are:

- 100 GeV electron in lead: $t_{max} = 8.5\ X_0$
- 1 TeV photon in lead: $t_{max} = 11.3\ X_0$
- 100 GeV electron in iron: $t_{max} = 7.5\ X_0$
- 1 TeV photon in iron: $t_{max} = 10.3\ X_0$

0.1.2 Shower length

Similarly we define a function to compute the 95% containement shower length :

$$t_{95\%}\ [X_0] = t_{max} + 0.08\ Z + 9.6$$

[29]:
```
def shower_length(E0, Ec, Z, particle=electron):
    tmax = shower_max(E0, Ec, particle)
    t95 = tmax + 0.08*Z + 9.6
    return t95
```

[20]:
```
length_e_100GeV_Pb = shower_length(100*GeV, lead_Ec, lead_Z, electron)
length_g_1TeV_Pb = shower_length(1*TeV, lead_Ec, lead_Z, photon)
length_e_100GeV_Fe = shower_length(100*GeV, iron_Ec, iron_Z, electron)
length_g_1TeV_Fe = shower_length(1*TeV, iron_Ec, iron_Z, photon)
print('Shower containement:')
print('  ele 100GeV, Pb:', length_e_100GeV_Pb, 'X0')
print('  pho 1TeV, Pb:', length_g_1TeV_Pb, 'X0')
print('  ele 100GeV, Fe:', length_e_100GeV_Fe, 'X0')
print('  pho 1TeV, Fe:', length_g_1TeV_Fe, 'X0')
```

```
Shower containement:
  ele 100GeV, Pb: 24.69884443894822 X0
  pho 1TeV, Pb: 27.501429531942264 X0
  ele 100GeV, Fe: 19.172500578825865 X0
  pho 1TeV, Fe: 21.97508567181991 X0
```

The results are:

- 100 GeV electron in lead: $t_{95\%} = 24.7\ X_0$
- 1 TeV photon in lead: $t_{95\%} = 27.5\ X_0$
- 100 GeV electron in iron: $t_{95\%} = 19.2\ X_0$
- 1 TeV photon in iron: $t_{95\%} = 22.0\ X_0$

Finally these $t_{95\%}$ values are converted to `cm`, using the X_0 value in `cm`

[22]:
```
print('Shower containement [cm]:')
print('  ele 100GeV, Pb:', length_e_100GeV_Pb*lead_X0, 'cm')
print('  pho 1TeV, Pb:', length_g_1TeV_Pb*lead_X0, 'cm')
print('  ele 100GeV, Fe:', length_e_100GeV_Fe*iron_X0, 'cm')
print('  pho 1TeV, Fe:', length_g_1TeV_Fe*iron_X0, 'cm')
```

```
Shower containement [cm]:
  ele 100GeV, Pb: 13.831352885811006 cm
  pho 1TeV, Pb: 15.400800537887669 cm
  ele 100GeV, Fe: 33.74360101873352 cm
  pho 1TeV, Fe: 38.67615078240304 cm
```

- 100 GeV electron in lead: $t_{95\%} = 13.8$ cm
- 1 TeV photon in lead: $t_{95\%} = 15.4$ cm
- 100 GeV electron in iron: $t_{95\%} = 33.7$ cm
- 1 TeV photon in iron: $t_{95\%} = 38.7$ cm

We can therefore conclude that electrons and photons can be stopped in lead much more efficiently than in iron. It means that an electromagnetic calorimeter based on lead will be much more compact.

0.2 EM showers - Properties of CsI crystal

We define the needed material quantities here

[42]:
```
csi_rho = 4.51 # g.cm-3
csi_Ec = 11.17*MeV
cesium_m = 132.9
iodine_m = 126.9
cesium_X0 = 8.31 # g.cm-2
iodine_X0 = 8.48 # g.cm-2
cesium_RM = 15.53 # g.cm-2
iodine_RM = 15.75 # g.cm-2
```

0.2.1 Radiation length

For compounds, we can compute the X_0 value using the relative fractions of each element, using their atomic mass:

$$\frac{1}{X_{0,CsI}} = \frac{f_{Cs}}{X_{0,Cs}} + \frac{f_I}{X_{0,I}}$$

where f_{Cs} and f_I are the relative fraction (by mass) of the two elements:

$$f_{Cs} = \frac{m_{Cs}}{m_{Cs} + m_I}$$

$$f_I = 1 - f_{Cs}$$

Finally:

$$X_{0,CsI} = \frac{1}{\frac{f_{Cs}}{X_{0,Cs}} + \frac{f_I}{X_{0,I}}}$$

[44]:
```
cesium_mass_fraction = cesium_m / (cesium_m + iodine_m)
iodine_mass_fraction = 1. - cesium_mass_fraction
print("Cesium fraction =", cesium_mass_fraction)
print("Iodine fraction =", iodine_mass_fraction)
```

```
Cesium fraction = 0.5115473441108545
Iodine fraction = 0.48845265588914555
```

- $f_{Cs} = 0.512$
- $ f_{I} = 0.488$

[46]:
```
csi_X0 = 1./(cesium_mass_fraction/cesium_X0 + iodine_mass_fraction/iodine_X0)
print("CsI X0 =", csi_X0, "g.cm-2")
print("CsI X0 =", csi_X0/csi_rho, "cm")
```

```
CsI X0 = 8.392176980295032 g.cm-2
CsI X0 = 1.860793122016637 cm
```

$X_{0,CsI} = 8.39\ g.cm^{-2} = 1.86\ cm$

0.2.2 Moliere radius

The Moliere radius is defined from the radiation length and the critical energy:

$$\rho_M = \frac{21\ MeV \times X_0}{E_c[MeV]}$$

[39]:
```
csi_RM = 21.*MeV * csi_X0 / csi_Ec
print("CsI RM", csi_RM, "g.cm-2")
print("CsI RM", csi_RM/csi_rho, "cm")
```

```
CsI RM 15.777593248540347 g.cm-2
CsI RM 3.4983577047761303 cm
```

$\rho_{M,CsI} = 15.78\ g.cm^{-2} = 3.50\ cm$

```
[ ]:
```