IPython – a Versatile Platform for Numerical Simulations in Ion Radiotherapy

L. Grzanka[1,2], M. Klodowska[1], J. Gajewski[1], P. Olko[1], M.P.R. Waligorski[1,3]

1. Institute of Nuclear Physics Polish Academy of Sciences, Krakow

2. AGH University of Science and Technology, Krakow

3. The Marie-Skłodowska-Curie Centre of Oncology, Kraków Division

an executable talk

Background

Several concepts:

- **proton radiotherapy** - precisely locates the radiation dose in patient body
- **Bragg peak** - profile of energy loss of slowing down proton
- **beam characteristics** - dose, energy, fluence, LET

Aim:

- to demontrate potential of **IPython** in numerical simulation in proton radiotherapy

Proton radiotherapy in Cracow

In [3]:

```python
from IPython.display import HTML
HTML('<iframe src=http://en.wikipedia.org/wiki/Proton_therapy#Outside_the_USA?use
```

Out[3]:

[illegible]	[illegible]	[illegible]	[illegible]	Germany
RPTC Rinecker *Proton Therapy Center*	250	2009	Munich	Germany
PTC Uniklinikum Dresden	230	2014	Dresden	Germany
Wanjie *Proton Therapy Center*	230	2004	Zibo	China
Proton Medical Research Center University of Tsukuba	250	2001	Tsukuba	Japan
Centre de protonthérapie de l'Institut Curie	235	1991	Orsay	France
Centre Antoine Lacassagne	63	1991	Nice	France
Paul Scherrer Institut	250	1984	Villigen	Switzerland
Instytut Fizyki Jądrowej PAN	60	2011	Krakow	Poland
Centrum Cyklotronowe Bronowice	230	2015	Krakow	Poland
Proton Therapy Center, Prague	230	2012	Prague	Czech Republic
Shanghai Proton and Heavy Ion Center	230	2014	Shanghai	China
Proton Therapy Center, Korea National Cancer Center	230	2007	Seoul	Korea

United Kingdom [edit]

Monte Carlo simulations

Simulation setup:

- protons of initial energy **60 MeV**
- narrow pencil beam of size **0.1 mm**
- water containter of dimensions: **20x20x50** cm^3
- scoring detector: cylinder with **5 mm** radius and variable thickness
- scoring quantities: fluence and energy spectra at 79 depth steps

Calculations

- simulations performed on Zeus @ Cyfronet
- SHIELD-HIT12A r624 used
- traced 10^8 histories

Scoring quantities:

- **Energy** of protons at depths $E\ [MeV]$
- **Fluence** i.e. the flux of protons at depth $\phi\ [\frac{1}{cm^2}]$
- **Dose** - energy deposited per unit mass $D\ [Gy]$

Show time

In [4]:

```
# IPython magic - load libraries
import spectrum_mc
import matplotlib.pyplot as pylab
%matplotlib inline
```

Load simulation dataset for 60 MeV protons

In [5]:
```
filename = "60MeV.spc"
input_dose_Gy = 1 # normalisation [Gy]
dataset = spectrum_mc.Dataset(filename, input_dose_Gy)
```

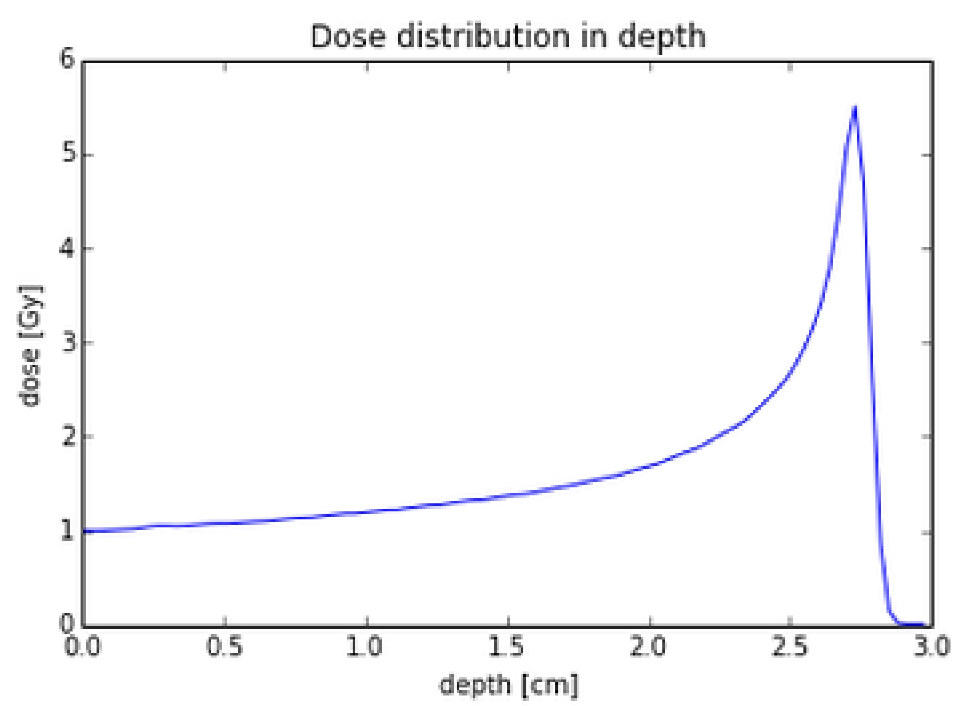
```
particle =  1001 , initial energy =  60.0  [MeV]
79  steps in depth, from  0.000 [cm] to  6.171  [cm]
10000  steps in energy, from  0.006 [MeV] to  99.995  [MeV]
Reading took:  4.98370409012 s
```

Dose distribution in depth

In [6]:
```
# Calculate dose distribution
depth_dose = dataset.depth_dose()
```

In [17]:
```
# Generate plot
pylab.title("Dose distribution in depth")
pylab.xlabel('depth [cm]')
pylab.ylabel('dose [Gy]')
pylab.plot(depth_dose.X(), depth_dose.Y())
```

Out[17]:
```
[<matplotlib.lines.Line2D at 0x7fd1561b5e90>]
```



Protons transmit energy to matter mostly due to **Culoumb interaction** with outer shell **atomic electrons**.

The energy loss is described by Bethe-Bloch equation:

$$\frac{dE}{dx} = \frac{4\pi}{m_e c^2} \cdot \frac{nz^2}{\beta^2} \cdot \left(\frac{e^2}{4\pi\varepsilon_0}\right)^2 \cdot \left[\ln\left(\frac{2m_e c^2\beta^2}{I \cdot (1-\beta^2)}\right) - \beta^2\right]$$

approximately

$$\frac{dE}{dx} \approx \frac{1}{v^2}$$

particle with relative speed β, charge z, and energy E, traveling a distance x into a medium of electron number density n and mean excitation potential I

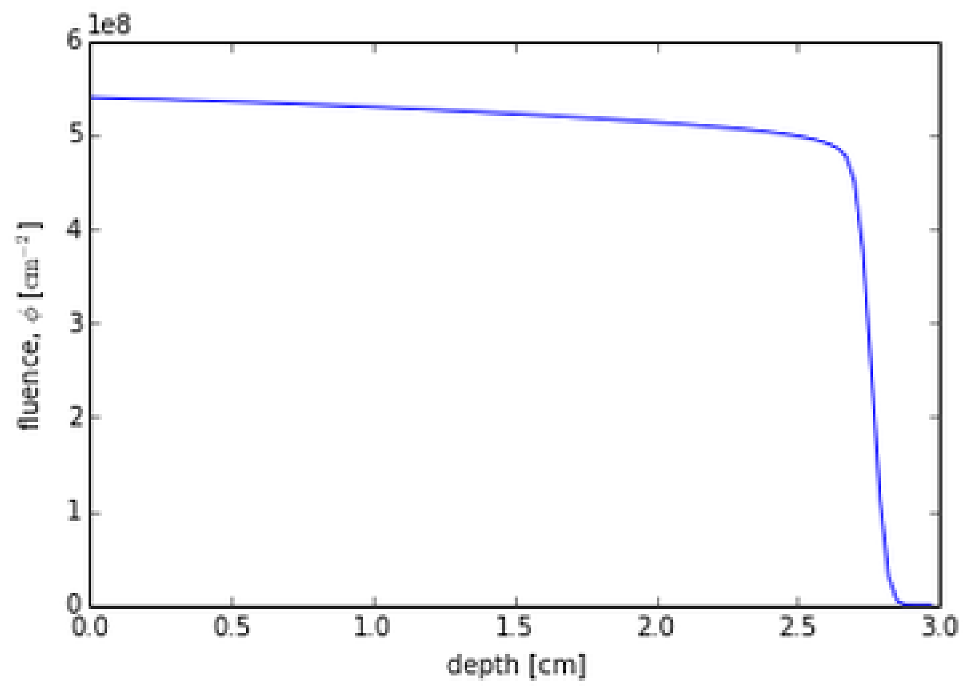
Fluence distribution in depth

In [8]:

```
# Get fluence distribution
depth_fluence = dataset.depth_fluence()
# Generate plot
pylab.xlabel('depth [cm]')
pylab.ylabel('fluence, $\phi$ [$\mathrm{cm^{-2}}$]')
pylab.plot( depth_fluence.X(), depth_fluence.Y())
```

Out[8]:

```
[<matplotlib.lines.Line2D at 0x7fd15676fa10>]
```



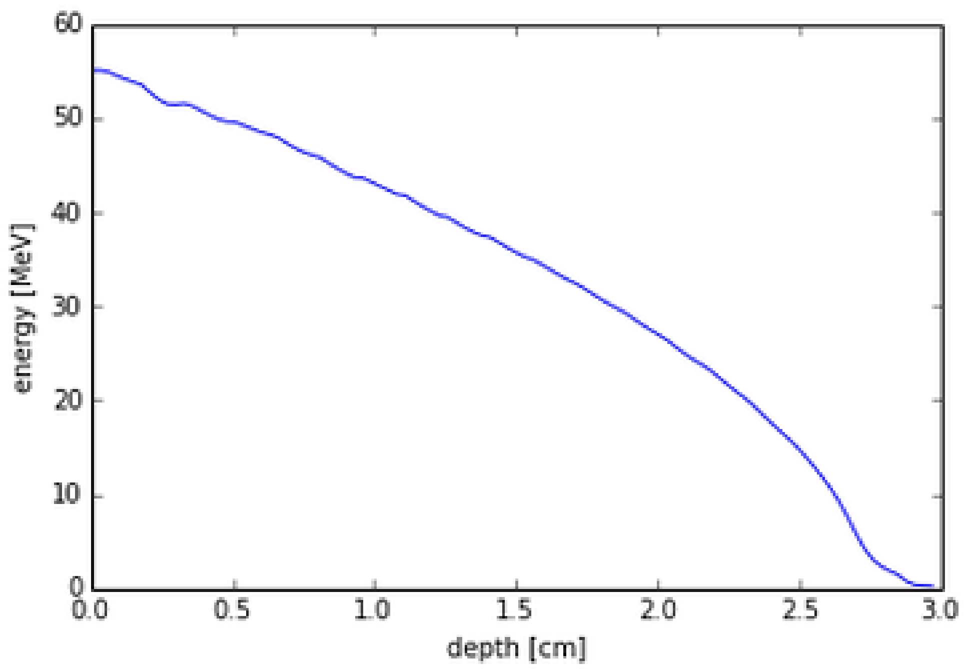
The number of protons, fluence ϕ, decreases with depth, mostly due to **non-elactic nuclear interactions**.

Energy distribution in depth

In [9]:
```
depth_energy = dataset.depth_energy() # Calculate mean LET distribution

pylab.xlabel('depth [cm]')
pylab.ylabel('energy [MeV]')
pylab.plot( depth_energy.X(), depth_energy.Y()) # Generate plot
```

Out[9]:
```
[<matplotlib.lines.Line2D at 0x7fd156719e50>]
```

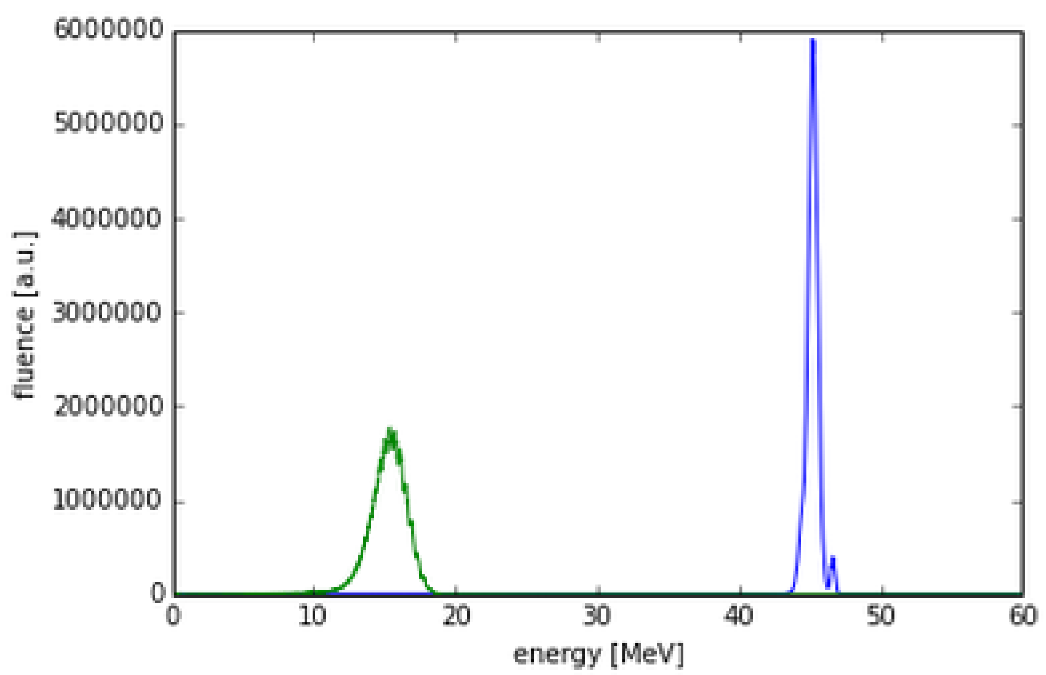


- As traveling protons transmit energy to medium they are less and less energetic.
- Most of the energy lost by proton is deposited in the medium.

Energy spectrum - more details

In [10]:
```
# Get energy spectrum at two depths
energy_spectrum_1 = dataset.energy_spectrum(0.9)
energy_spectrum_2 = dataset.energy_spectrum(2.5)
# Generate plot
pylab.xlabel('energy [MeV]')
pylab.ylabel('fluence [a.u.]')
pylab.xlim( [0,60])
pylab.plot( energy_spectrum_1.X(), energy_spectrum_1.Y())
pylab.plot( energy_spectrum_2.X(), energy_spectrum_2.Y())
```

Out[10]:
```
[<matplotlib.lines.Line2D at 0x7fd156661790>]
```



single proton => precisely determined energy
many protons => spectrum of proton energies

spread of energy spectrum is related to **elastic scattering** of protons on **atomic nuclei** real application

Experiment exapmle

Aim of the experiment:

Measure the dose response and the energy efficiency of detectors.

Detector type:

Kodak® EDR2 radiographic films (AgBr active material)

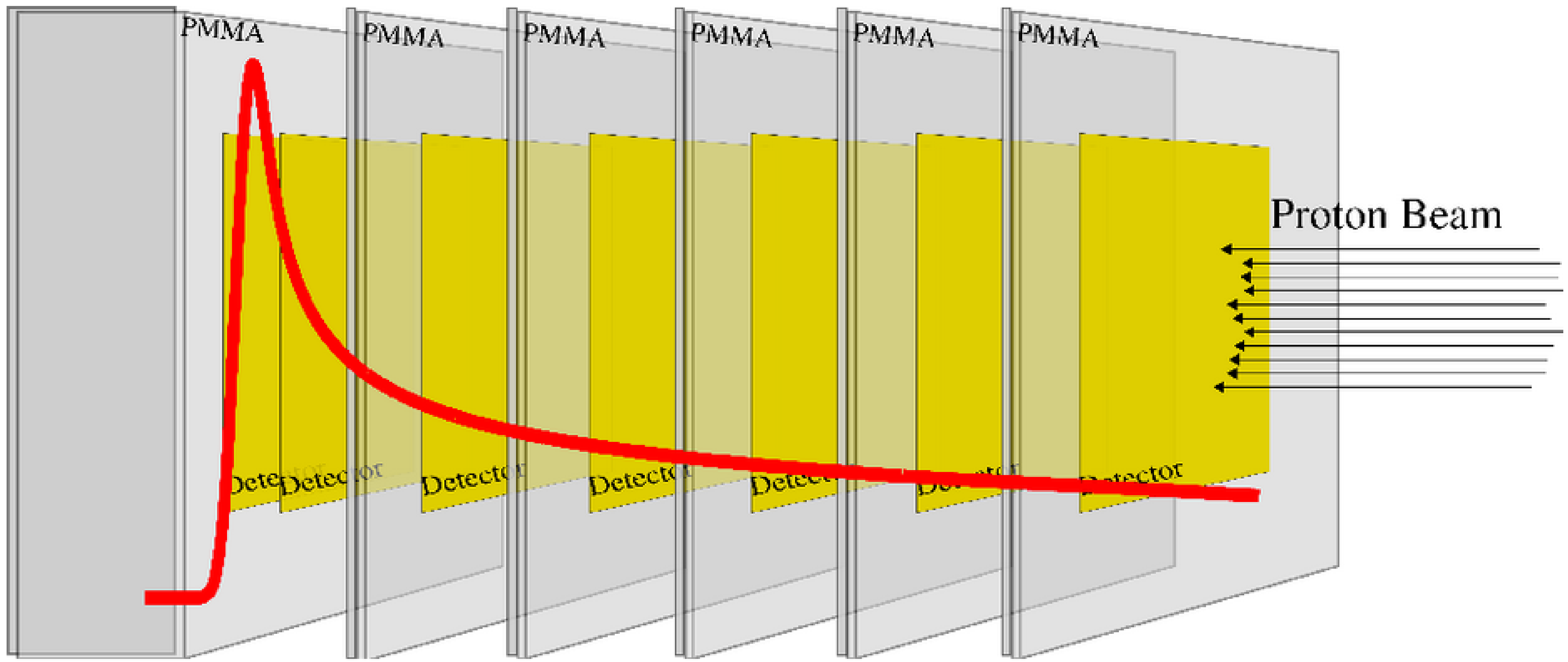
Experiment idea:

Irradiate the detectors at different depths in Bragg peak (protons 60 MeV)

In [11]:

```
from wand.image import Image as WImage
img = WImage(filename='exp.eps')
img
```

Out[11]:



Get parameters at certain depth

In [12]:

```
depth_cm = 2.5
print "at depth = ", depth_cm, " [cm] we have:"
print " - dose = ", dataset.dose_Gy( depth_cm ), " [Gy] "
print " - fluence = ", dataset.fluence_cm2( depth_cm ), " [1/cm2] "
print " - energy = ", dataset.energy_MeV( depth_cm )[0], " [MeV] "
print " - LET = ", dataset.let_MeV_cm( depth_cm )[0], " [MeV/cm] "
```

```
at depth =  2.5  [cm] we have:
 - dose =  2.67196446392  [Gy]
 - fluence =  498342488.51  [1/cm2]
 - energy =  14.8074092442  [MeV]
 - LET =  38.2399238699  [MeV/cm]
```

OR in the other way

Find depth with cetrain beam parameters...

...with given dose

In [13]:
```
dose_Gy = 3
print "dose ", dose_Gy , " [Gy] is at depth:", dataset.depth_cm_at_dose_Gy( dose_
```

```
dose  3  [Gy] is at depth: 2.55879478784  [cm]
```

...with given mean energy

In [14]:
```
energy_MeV = 30.0
print "energy ", energy_MeV , " [MeV] is at depth:", dataset.depth_cm_at_energy_M
```

```
energy  30.0  [MeV] is at depth: 1.84320410482  [cm]
```

...with given fluence

In [15]:
```
fluence_cm2 = dataset.fluence_cm2(depth_cm=0) / 2
print "fluence ", fluence_cm2 , " [1/cm^2] is at depth:", dataset.depth_cm_at_flu
```

```
fluence  269713144.257  [1/cm^2] is at depth: 2.75493784937  [cm]
```

Technology stack

- SHIELD-HIT12A Monte-Carlo code (@Zeus, Cyfronet)
- libamtrack numerical library (@Zeus, Cyfronet)
- Ubuntu VM (@Cracow Cloud One, IFJ PAN)
- Jupyter IPython notebook (@Cracow Cloud One, IFJ PAN)

IPython notebook solutions

- easy to use for beginners
- scientific writing: LaTeX support
- visualisation: matplotlib support, slideshow support
- multiple user support: JupyterHub
- calculation sharing
    - Google: "ipython notebook lectures" - interactive lectures on web
    - JSON format behind, easily versioned on GitHub
- stability issues
- still under heavy development

Thank you