FLUKA LAB

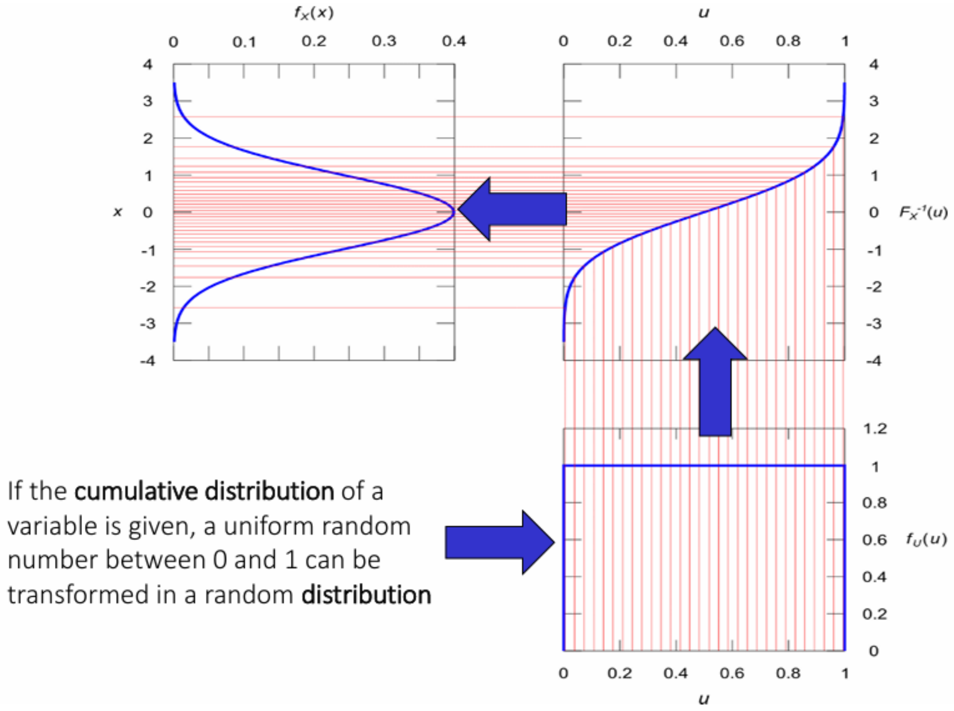
MICHELE ANASTASI

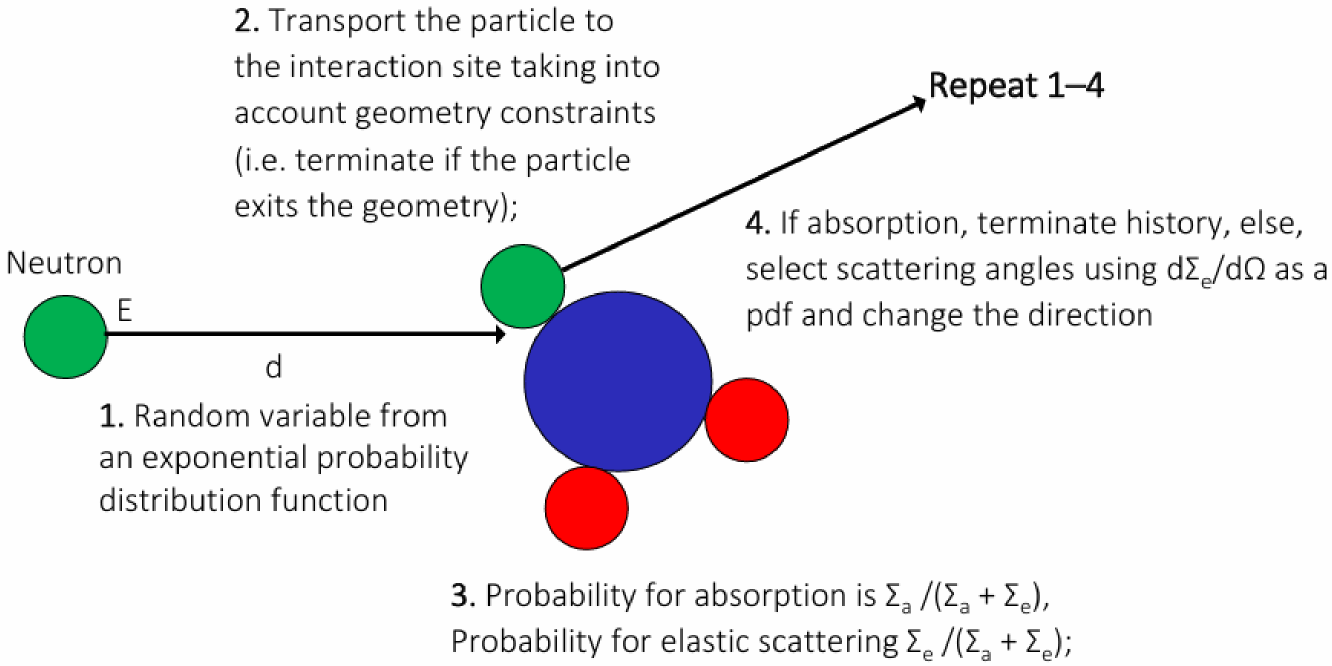
NOTE: Slides purposely prepared to hold the “Medical application of rad. fields" exam

Main goals of the lab experience

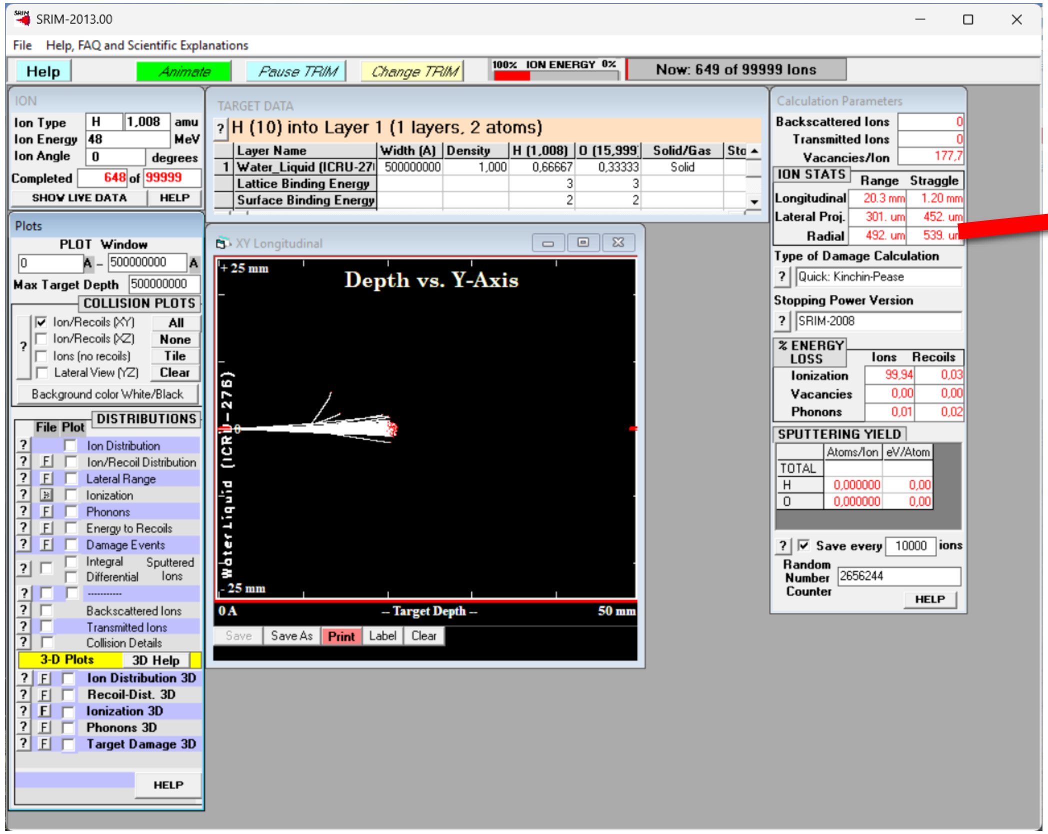
- Simulation of a SOBP covering a depth between 2 and 3 cm in water
- USRBIN cards for the dose and/or energy deposition inside the target
- Calculation of the minimum and maximum proton energies
- Simulation of the depth-dose distribution of different energies
- Calculation of the weight of each energy

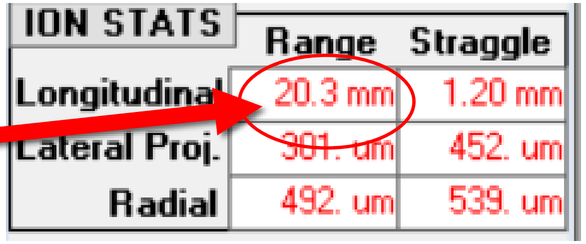
Notes about FLUKA/FLAIR





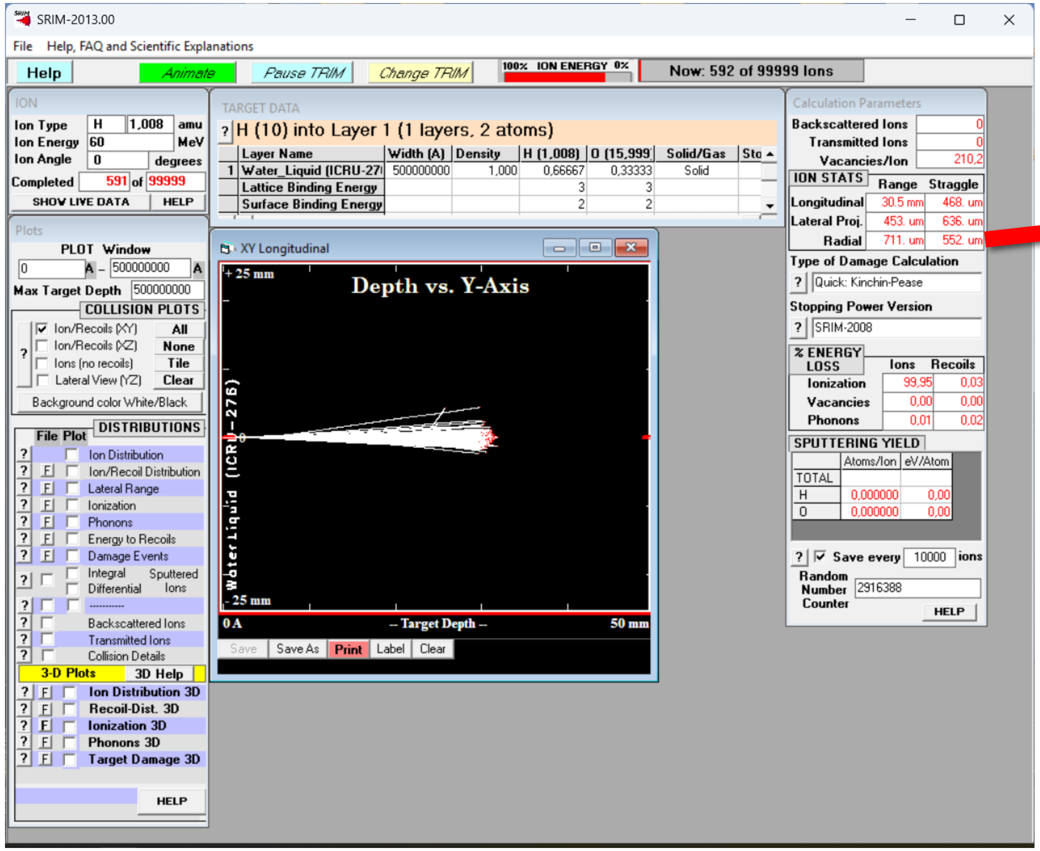
Getting energies by using SRIM

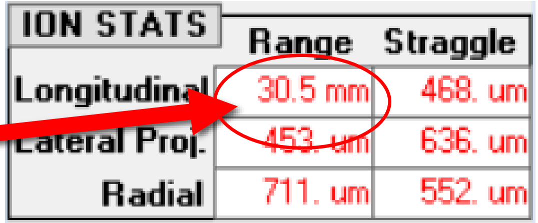




Example for **48 MeV** protons in liquid water target: getting about 2 cm as range

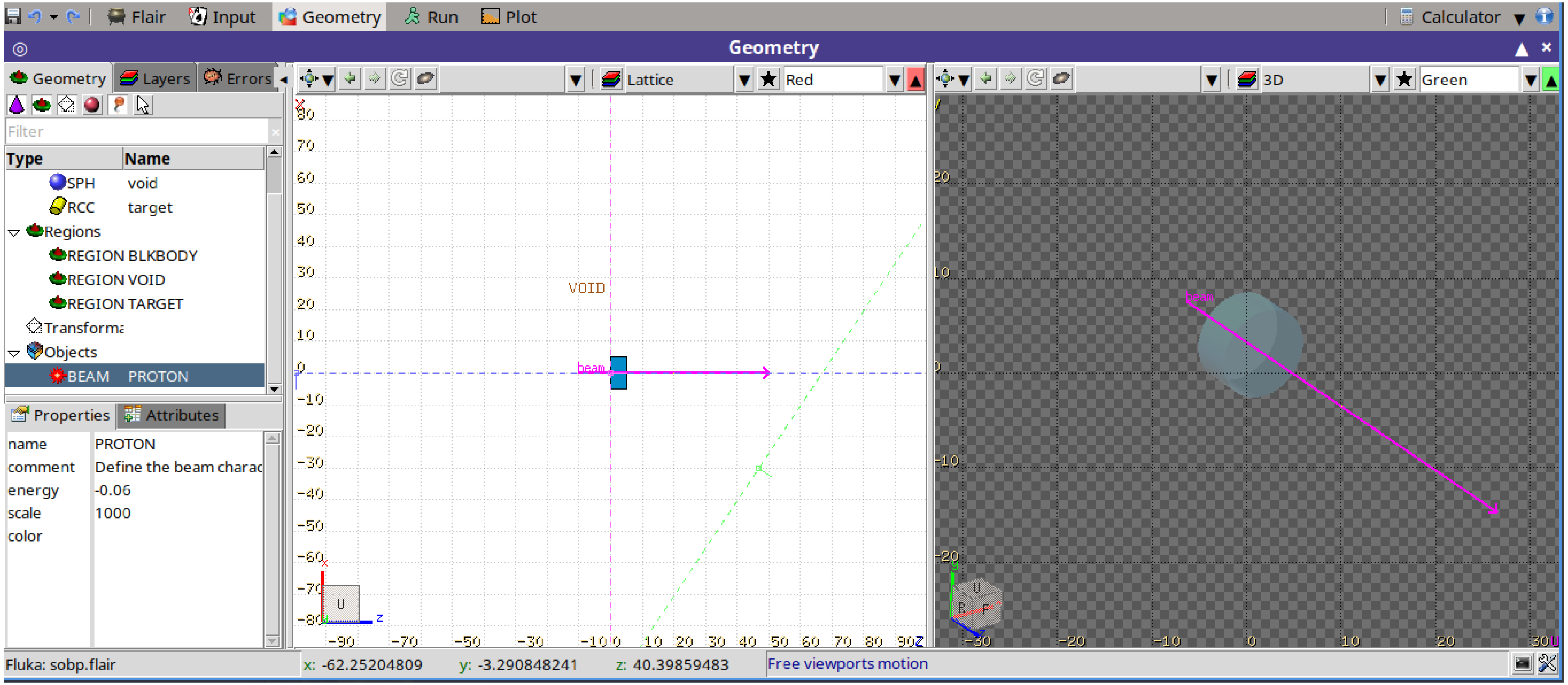
Getting energies by using SRIM



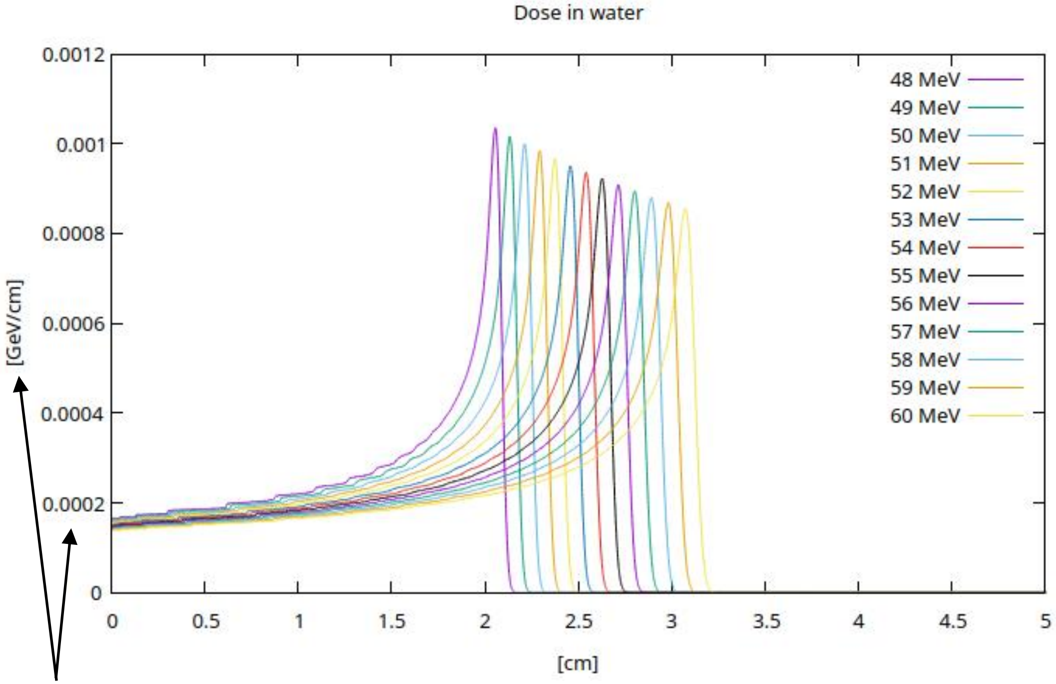


Example for **60 MeV** protons in liquid water target: getting about 3 cm as range

FLUKA: (very basic, actually) setup



FLUKA: results



Note: consider those numbers as arbitrary as further confirms would be required

Simulations run with energies between 48 and 60 MeV (pencil-beam of protons)

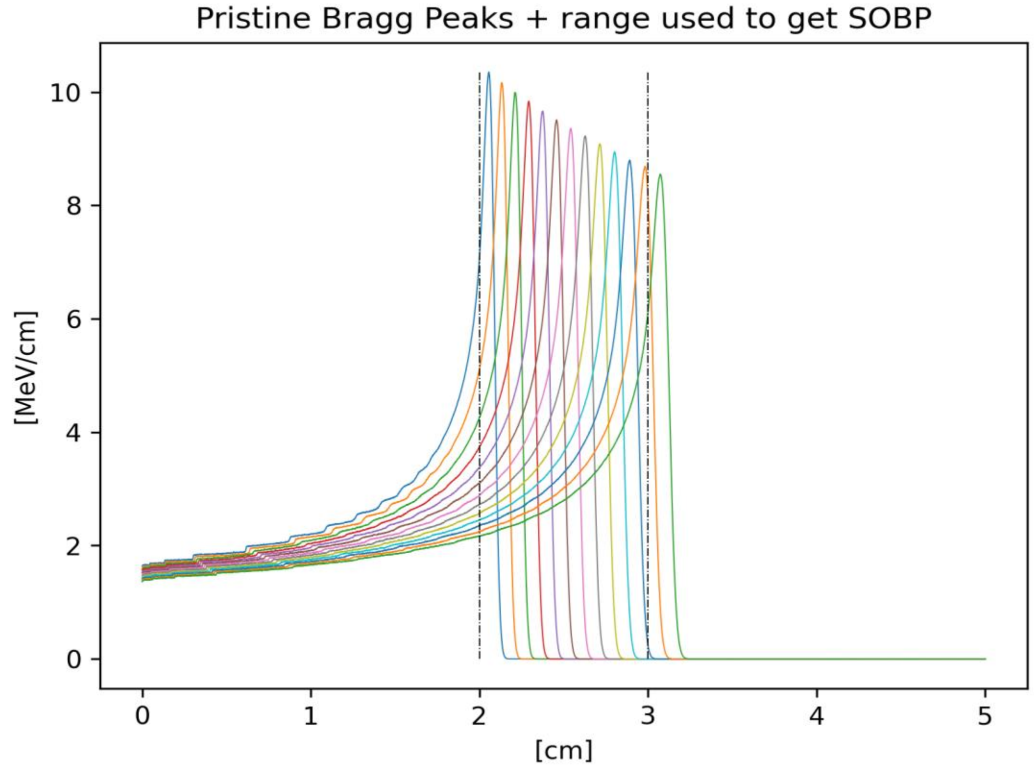
Python script to get weights

```python
# plot
yy_sum = np.sum(yy, axis=0)
for i in range(len(energies)):
    plt.plot(xx,yy[i,:],lw='0.5')
plt.vlines(a/1e3, 0,np.max(yy),linestyles='dashdot',linew
plt.vlines(b/1e3, 0,np.max(yy),linestyles='dashdot',linew
plt.title("Pristine Bragg Peaks + range used to get SOBP")
plt.xlabel("[cm]")
plt.ylabel("[MeV/cm]")
plt.savefig("saved_plots/pristinePeaks.png",dpi=300)
plt.show()

# define functions to minimize peaks differences
def resto(w):
    return ref - np.sum(yy_extr * w[:, np.newaxis], axis=0)
res = sp.optimize.least_squares( resto,weights_0 )
wf = res.x
print("Weights for the pristine peaks to get SOBP:")
print(wf)

# plotting
yy = yy/ref # normalize to reference
sobp = np.sum(yy * wf[:, np.newaxis],axis=0)

plt.plot(xx, sobp, lw='0.5')
plt.vlines(a/1e3, 0,np.max(yy),linestyles='dashdot',linew
plt.vlines(b/1e3, 0,np.max(yy),linestyles='dashdot',linew
plt.hlines(ref/ref, 0, 5,linestyles='dashdot',linewi

yy_w = np.zeros([len(energies), len(xx)])
for i in range(len(energies)):
    yy_w[i,:] = yy[i,:]*wf[i]
for i in range(len(energies)):
    plt.plot(xx, yy_w[i, :], lw='0.5')
plt.title("SOBP")
plt.xlabel("[cm]")
plt.ylabel("Relative dose")
plt.savefig("saved_plots/sobp_1.png",dpi=300)
plt.show()
```

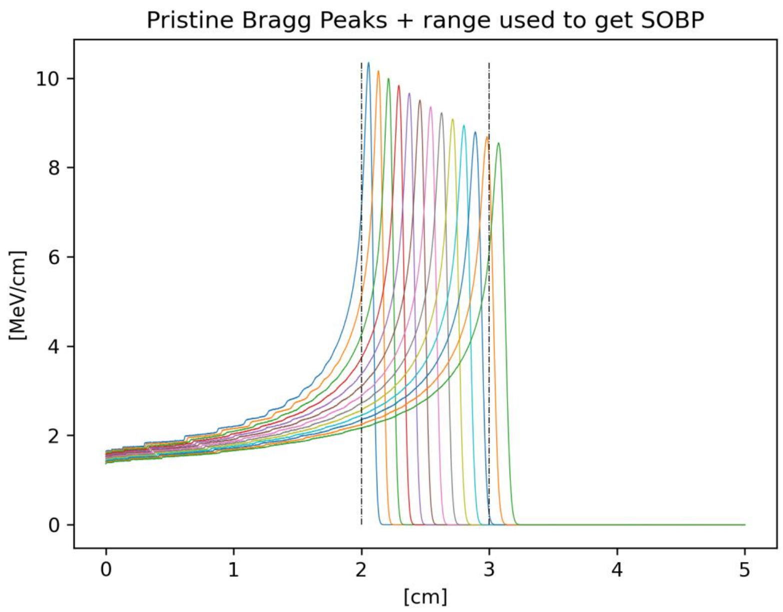
```python
# define functions to minimize peaks differences
def resto(w):
    return ref - np.sum(yy_extr * w[:, np.newaxis], axis=0)
res = sp.optimize.least_squares( resto,weights_0 )
wf = res.x
```

Minimizing the difference between the reference chosen and the sum of the functions multiplied by the weights

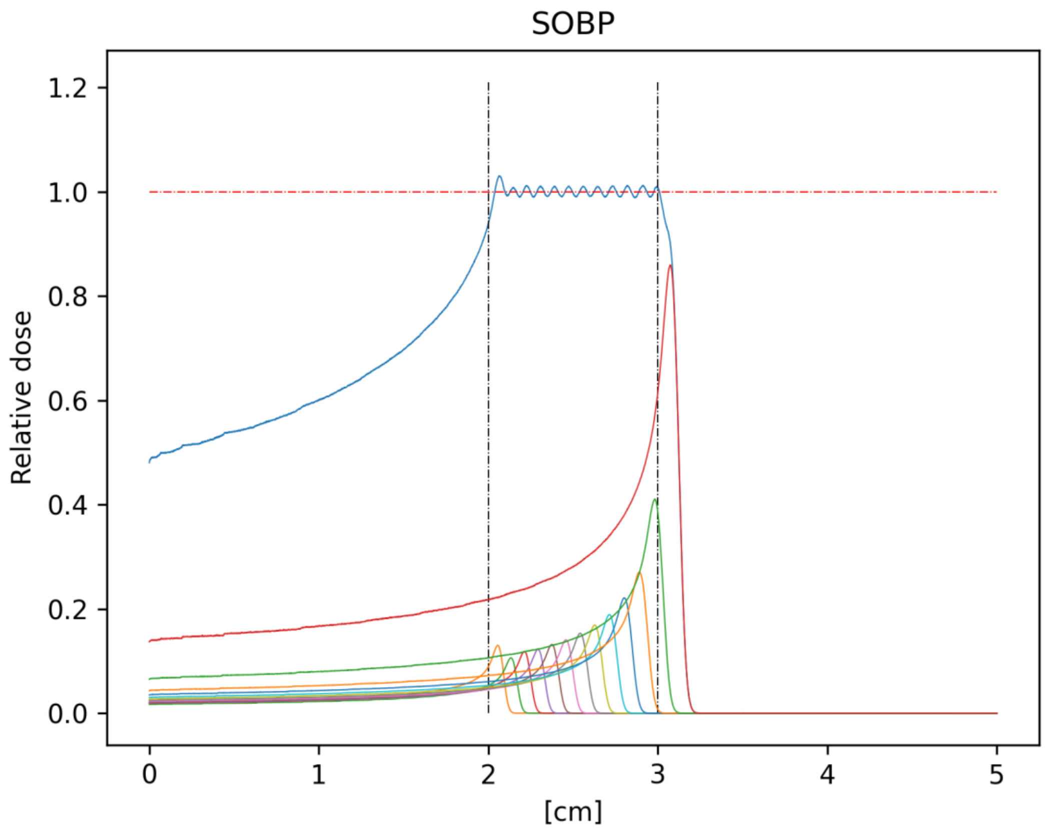
With reference (and then normalized) to the last peak original height



Python script: results

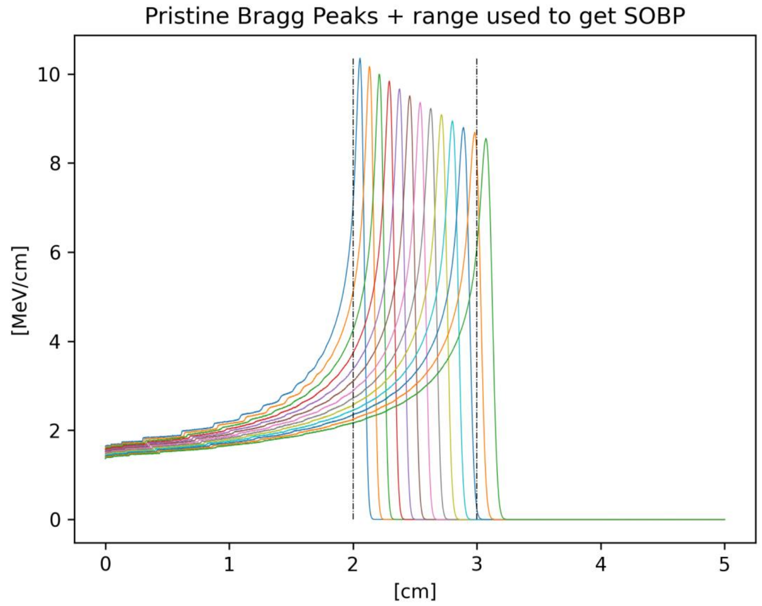


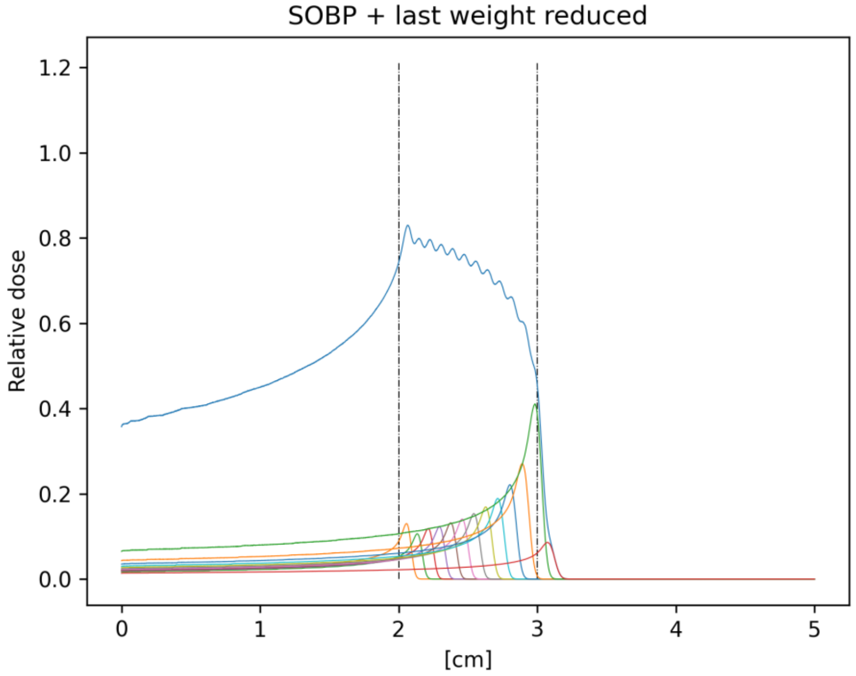
48 MeV	0.10753443	55 MeV	0.15680211
49 MeV	0.08933658	56 MeV	0.17794599
50 MeV	0.10134561	57 MeV	0.2114914
51 MeV	0.10663039	58 MeV	0.26320801
52 MeV	0.11665866	59 MeV	0.40412899
53 MeV	0.12666842	60 MeV	0.85962244
54 MeV	0.14027139		



With reference (and then normalized) to the last peak original height

FOCUS: what if critical tissue at 3 cm depth?





48 MeV	0.10753443	55 MeV	0.15680211
49 MeV	0.08933658	56 MeV	0.17794599
50 MeV	0.10134561	57 MeV	0.2114914
51 MeV	0.10663039	58 MeV	0.26320801
52 MeV	0.11665866	59 MeV	0.40412899
53 MeV	0.12666842	60 MeV	**0.08596224**
54 MeV	0.14027139		

Just for instance, decreased at 10% of the original value

It's wise to smooth the SOBP when treating in proximity of a critical tissue since the RBE tends to increase a bit for higher LETs: this is generally true for protons, whereas for other heavier ions it depends.