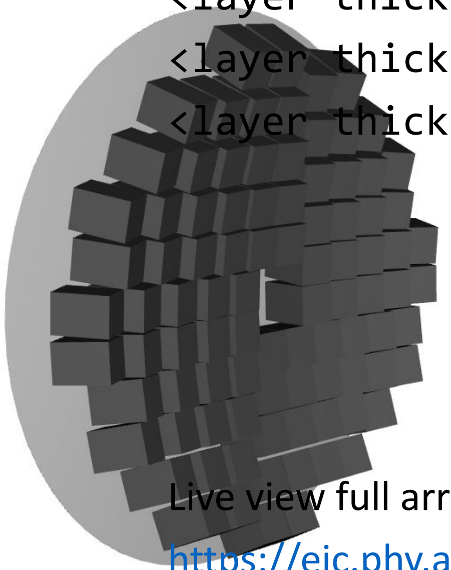
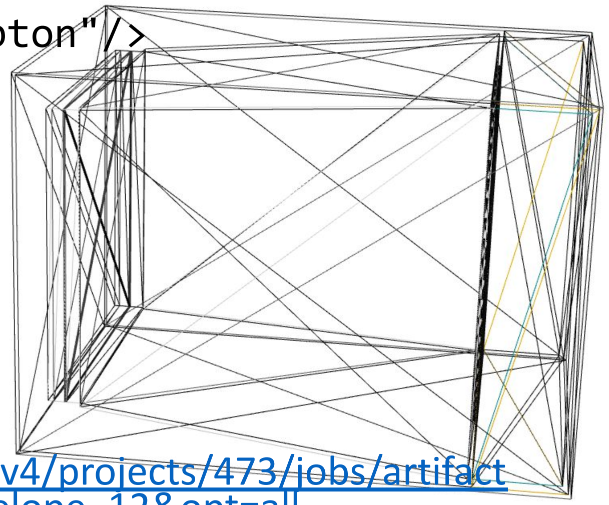
mRICH Readout Material Budgets

```
<layer thickness="MRICHPhotoDetMCPlate_thickness=0.3*mm" material="PyrexGlass"/>
<layer thickness="MRICHPhotoDetMCPlate_thickness=0.3*mm" material="PyrexGlass"/>
<layer thickness="MRICHPhotoDetAnode_thickness=3.8*mm"   material="AluminumOxide"/>
<layer thickness="MRICHPhotoDetPCB_thickness=2.0*mm"     material="Fr4"/>
<layer thickness="MRICHPhotoDetCopper_thickness=0.1*mm"  material="Copper"/>
<layer thickness="MRICHPhotoDetKapton_thickness=0.2*mm"  material="Kapton"/>
```





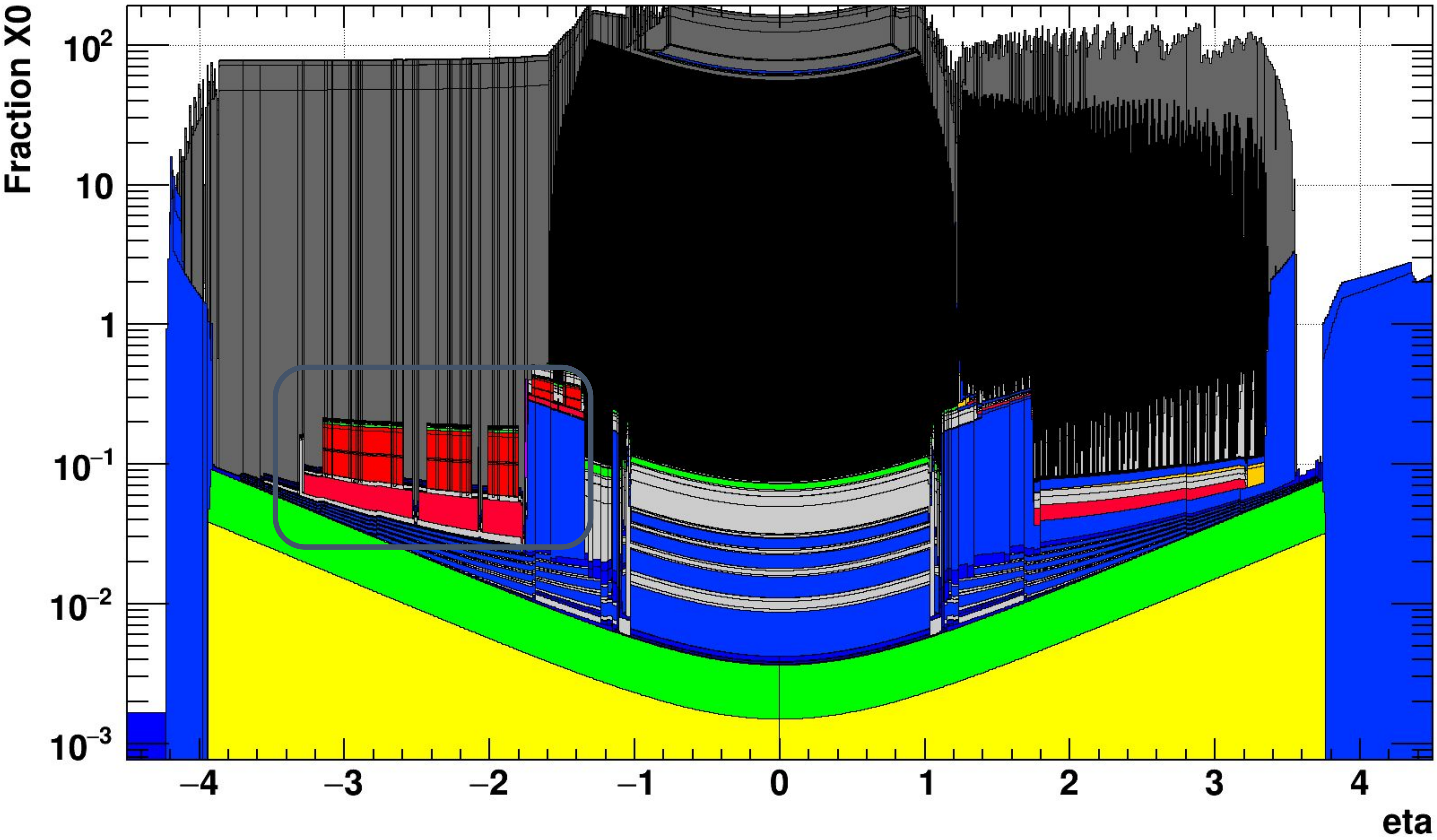
Live view full array :

https://eic.phy.anl.gov/geoviewer/index.htm?nobrowser&file=https://eicweb.phy.anl.gov/api/v4/projects/473/jobs/artifacts/master/raw/geo/detector_geo.root?job=report&item=default;1/world_volume/MRICH_Envelope_12&opt=all

Live view single module (wireframe):

https://eic.phy.anl.gov/geoviewer/index.htm?file=https://eicweb.phy.anl.gov/api/v4/projects/473/jobs/artifacts/master/raw/geo/detector_geo.root?job=report&item=default;1/world_volume/MRICH_Envelope_12/MRICH_module1_0&opt=wire

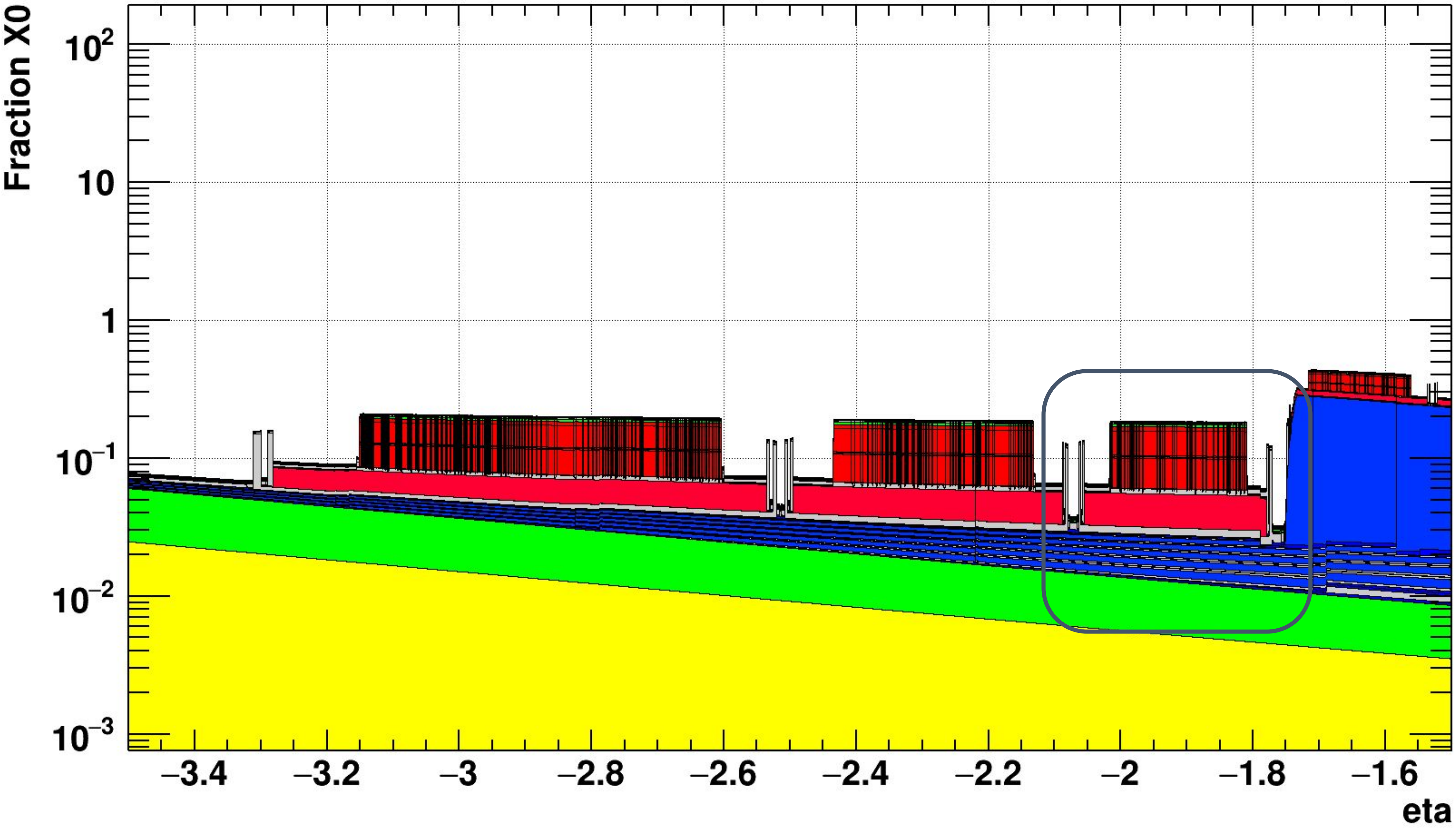
Material Scan (max distance 500 cm)



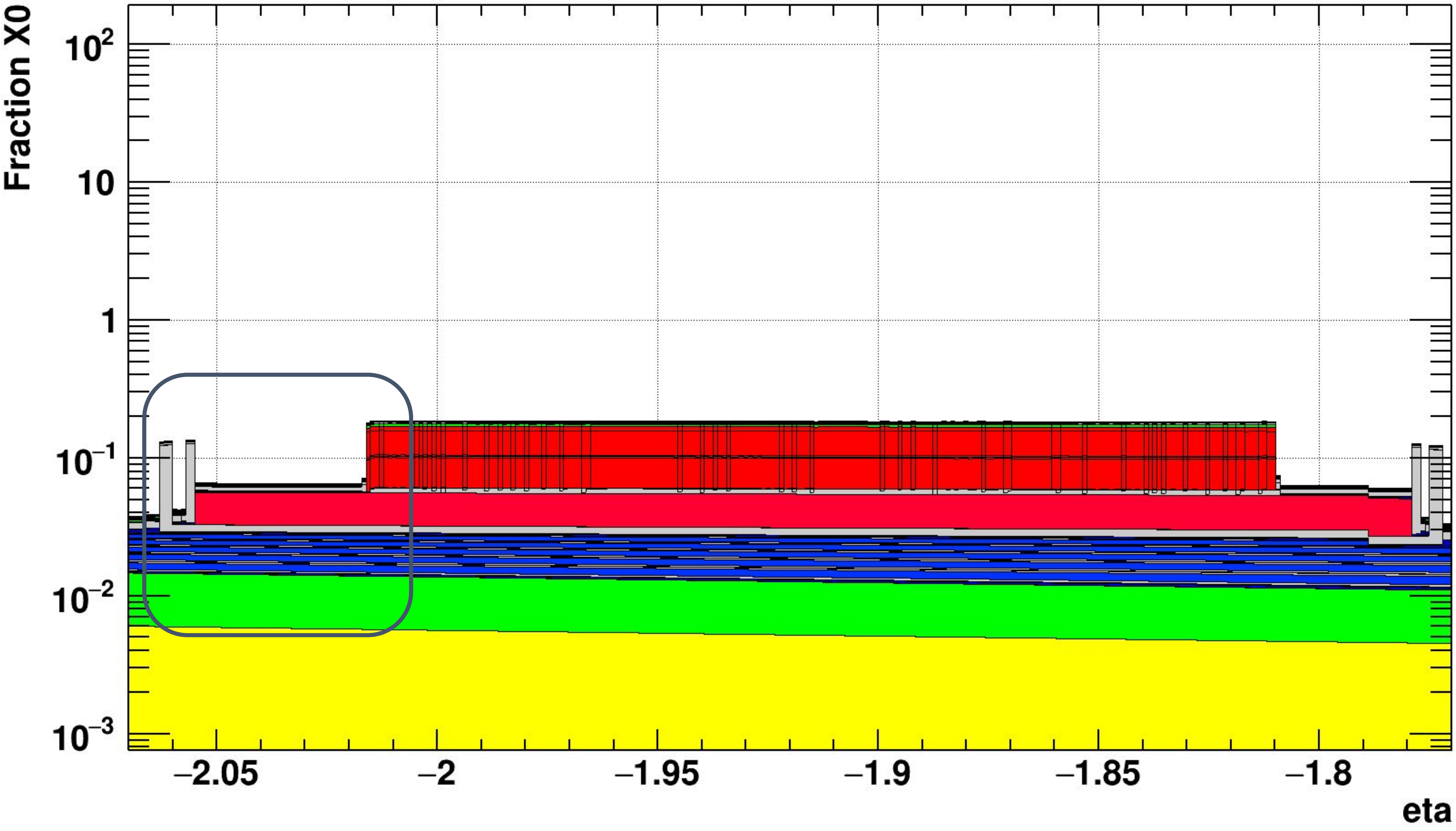
```
Color_t color(const Material& m) {
  if      (m.name() == std::string("Silicon"))          return kGray;
  else if (m.name() == std::string("Aluminum"))         return kAzure;
  else if (m.name() == std::string("CarbonFiber"))      return kGray;
  else if (m.name() == std::string("Beryllium"))        return kGreen;
  else if (m.name() == std::string("Gold"))             return kYellow;
  else if (m.name() == std::string("Mylar"))            return kGreen;
  else if (m.name() == std::string("Kapton"))           return kGreen;
  else if (m.name() == std::string("Copper"))           return kGreen;
  else if (m.name() == std::string("C2F6_DRICH"))       return kOrange;
  else if (m.name() == std::string("Ar10CO2"))          return kOrange;
  else if (m.name() == std::string("Aerogel"))          return kPink;
  else if (m.name() == std::string("AerogelOptical"))  return kPink;
  else if (m.name() == std::string("Aerogel_DRICH"))   return kPink;
  else if (m.name() == std::string("Lead"))             return kBlack;
  else if (m.name() == std::string("Steel235"))         return kGray+2;
  else if (m.name() == std::string("TungstenDens24"))  return kBlack;
  else if (m.name() == std::string("Polystyrene"))      return kGray;
  else if (m.name() == std::string("PolystyreneFoam")) return kGray;
  else if (m.name() == std::string("Epoxy"))            return kGray;
  else if (m.name() == std::string("PlasticScint"))     return kGray;
  else if (m.name() == std::string("AcrylicOptical"))  return kGray;
  else if (m.name() == std::string("Acrylic_DRICH"))   return kGray;
  else if (m.name() == std::string("Quartz"))           return kViolet;
  else if (m.name() == std::string("Air"))              return kBlue;
  else if (m.name() == std::string("AirOptical"))       return kBlue;
  else if (m.name() == std::string("Vacuum"))           return kWhite;
  else {
   std::cout << "Unknown material: " << m.name() << std::endl;
   return kRed;
  }
}
```

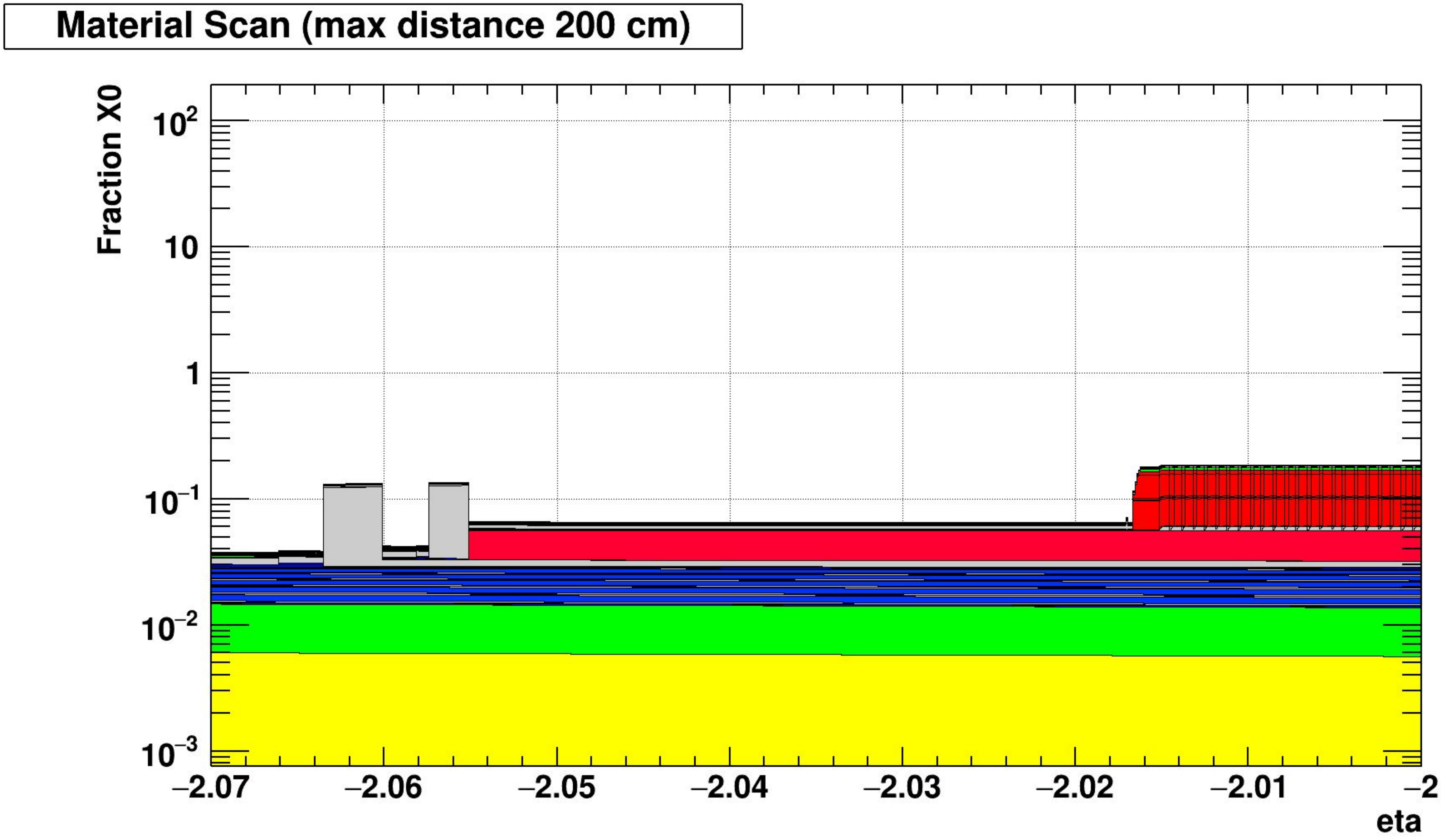
```
Unknown material: PyrexGlassOptical
Unknown material: PyrexGlass
Unknown material: PyrexGlass
Unknown material: AluminumOxide
Unknown material: Fr4
Unknown material: PyrexGlassOptical
Unknown material: PyrexGlass
Unknown material: PyrexGlass
Unknown material: AluminumOxide
Unknown material: Fr4
```

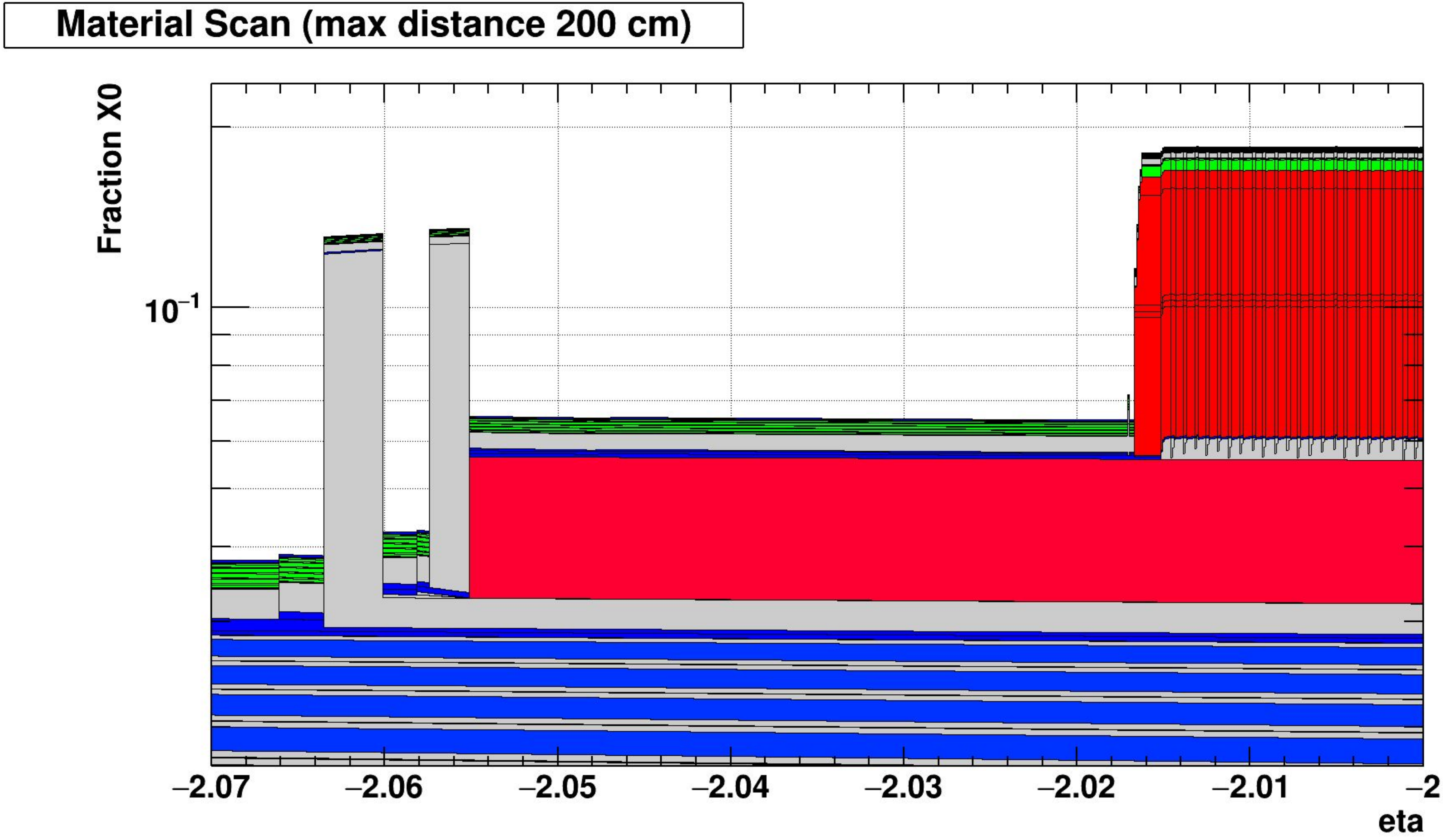
Material Scan (max distance 200 cm)



Material Scan (max distance 200 cm)




Material Scan (max distance 200 cm)
Fraction X0
10^2
10
1
10^{-1}
10^{-2}
10^{-3}
−2.07
−2.06
−2.05
−2.04
−2.03
−2.02
−2.01
−2
eta

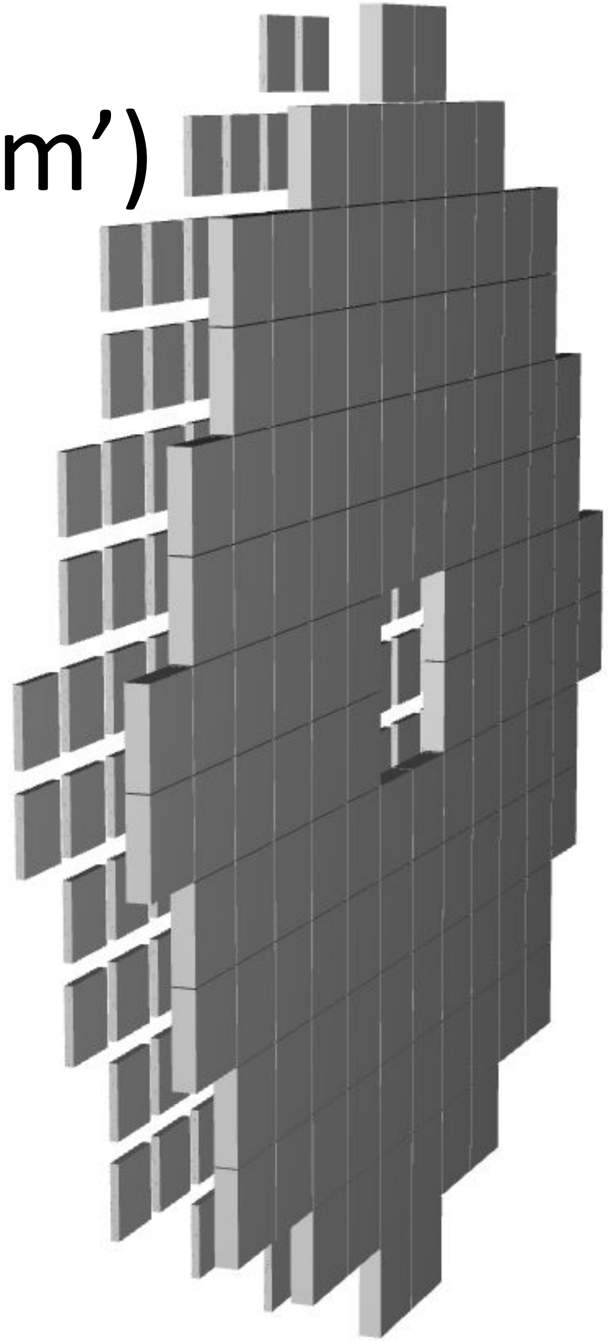

Material Scan (max distance 200 cm)
Fraction X0
10^{-1}
−2.07
−2.06
−2.05
−2.04
−2.03
−2.02
−2.01
−2
eta

Alternative mRICH (without the ‘m’)

Implemented as mRICH_alt.xml:

- non-projective layout
- same module material budget
- no mirrors (2 mm Al2O3), no frame (1 mm CF)

Not really tested yet.



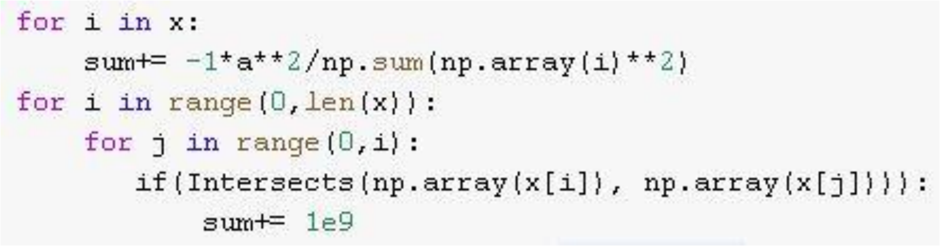
MRICH Modules Placement Optimization

Assume, number of modules to be constant = 110
Fix rotational degree of freedom about axis connecting to origin.
x/y dimension (mm), a = $\sqrt{\pi \cdot (950^2 - 100^2) / 110}$

z dimension (mm) = 200

Variables: 3 position coordinates for each module.

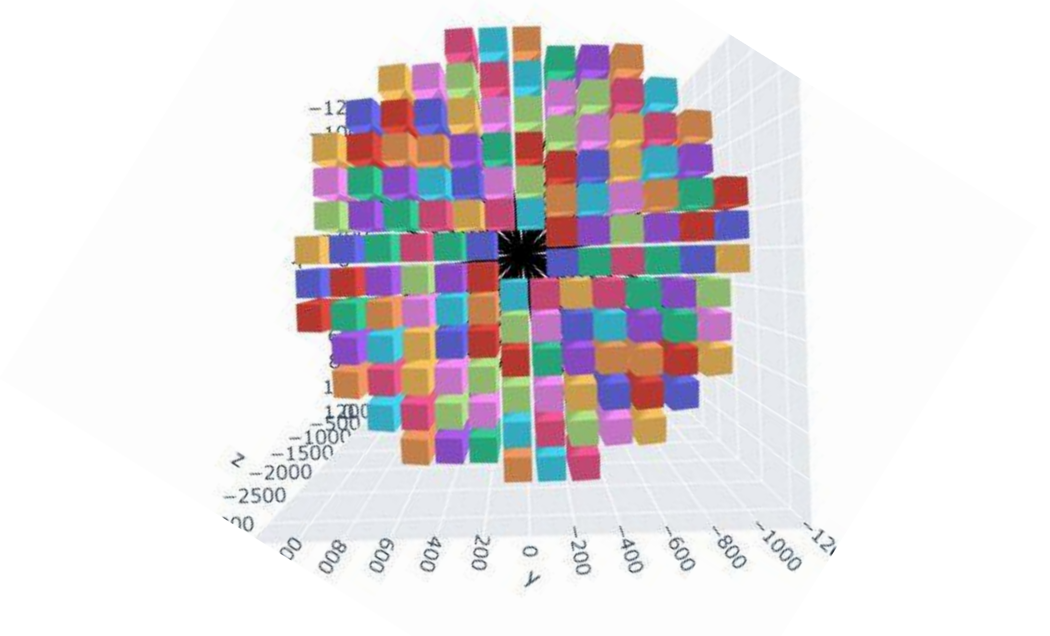
Algorithm: Bayesian Optimization using Scikit-Learn
Minimize the negative of the solid angles sum for all modules.
Domain: [-950, 950] for x and y and [-2945-200, -2945+200] in z



```
for i in x:
    sum+= -1*a**2/np.sum(np.array(i)**2)
for i in range(0,len(x)):
    for j in range(0,i):
       if(Intersects(np.array(x[i]), np.array(x[j]))):
           sum+= 1e9
```

If unique pairs of modules intersect with each other (Separation Axis Theorem) or boundary (any corner is within 100 mm inner radius or outside 950 mm outer radius), add large cost.
Google Collab Link

Developed simple visualization in python assuming rectangular boxes



Current Status

1) Formalism well defined
2) Too slow to run on google collab.
3) Trying to automate on compute canada/OSG now.