Curved Modular OB – update to outer silicon barrel geometry

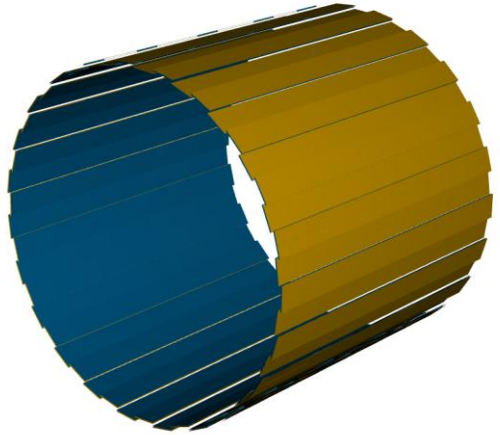
PR #1083

- New geometry for active silicon in OB
- Curved surface modelled as series of flat segments
- Modules on top and bottom of staves
- Dimensions set to match February 2026 CAD files
- ‘Floating’ at correct position
- Hit maps, material thickness scans, tracking benchmark plots done
- https://github.com/eic/epic/tree/curved_modular_OB

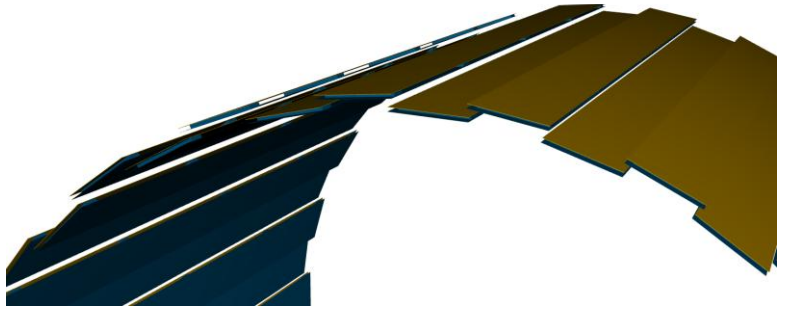
Next steps

- Update passive material geometry
- Build passive structure from GDML files

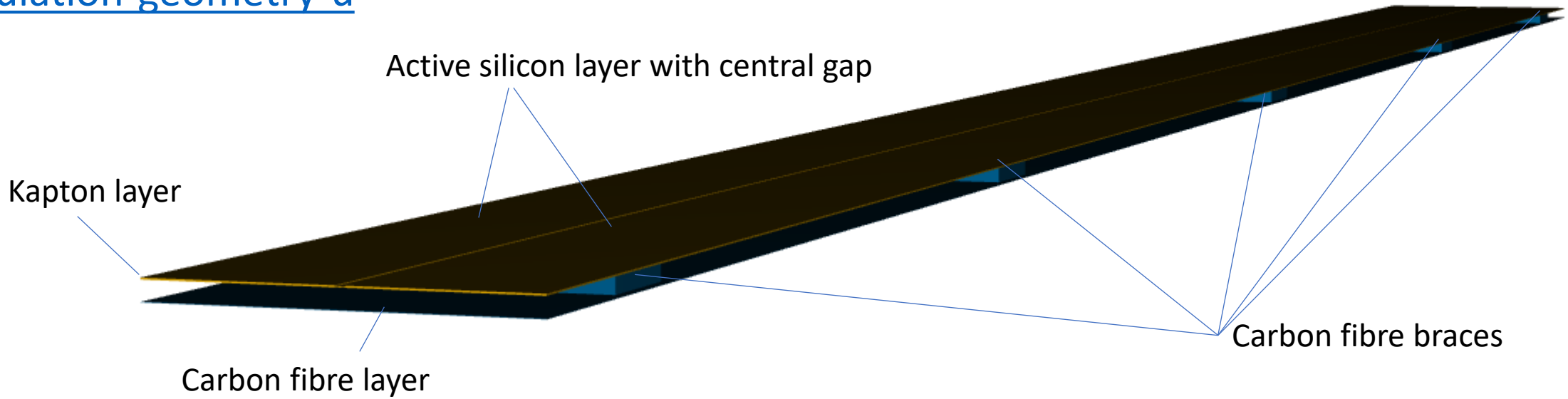
Current flat stave OB geometry epic 25.10



- Carbon fibre braces to reproduce peaks on material thickness scan
- Gap in active silicon to reproduce dead area
- Castellated stave arrangement (alternate staves at +6mm radius)

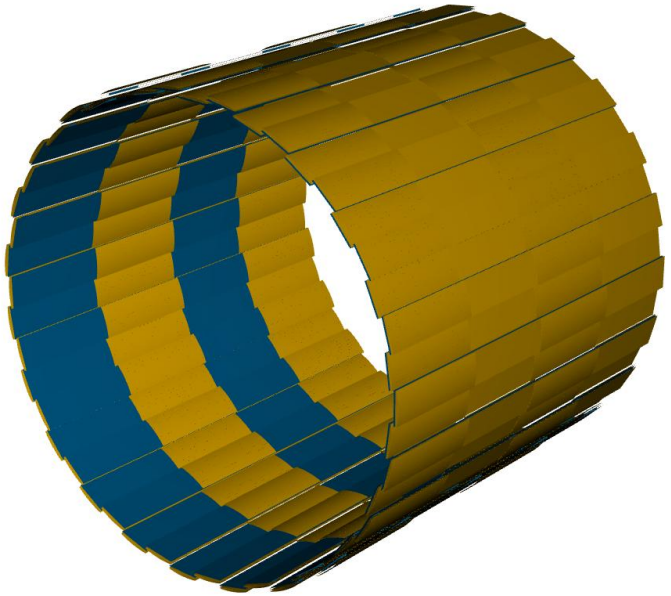


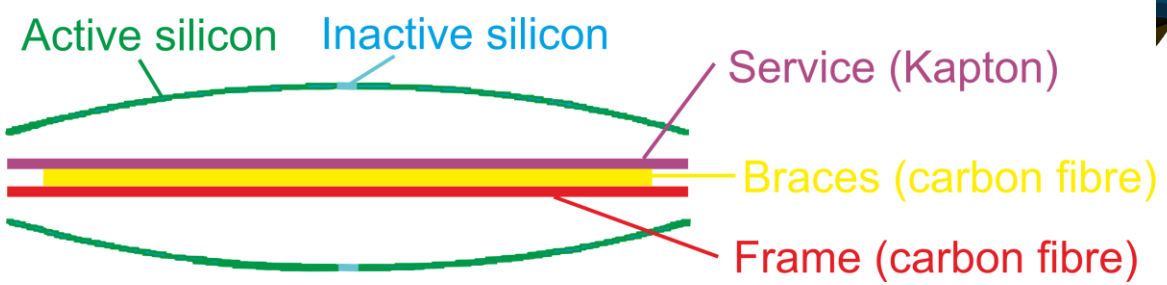
https://indico.bnl.gov/event/29542/#2-svt-ob-simulation-geometry-u

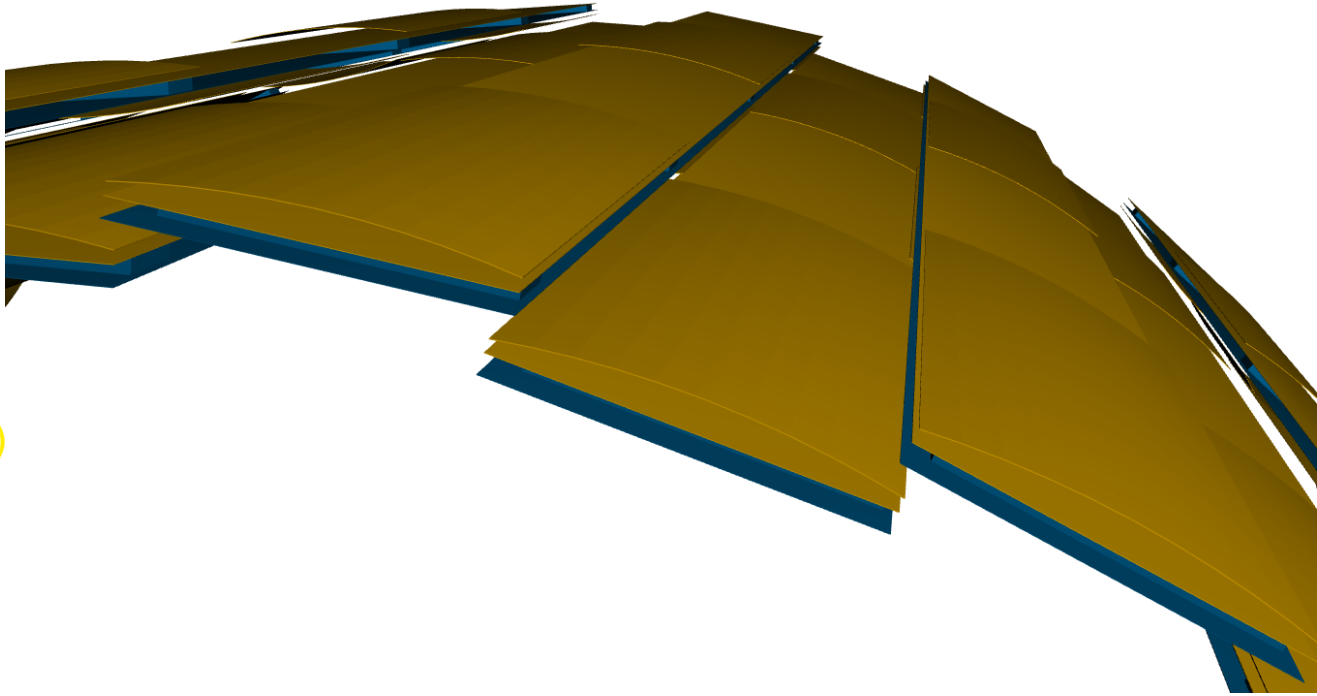


Curved Modular Geometry

- DD4HEP cylindrical surface readout using CylindricalGridPhiZ – only works for cylinder with axis along beamline
- This version: curved surfaces modelled as N flat strips
- Separate modules on top and bottom of staves







New XML code for curved component

```
<module_component name="CurvedSi3R"
     material="Silicon"
     sensitive="true"
     innerthickness="SiBarrel1_LongThickness"
     outerthickness="SiBarrel_ShortThickness"
     nsegments="SiBarrelSiliconCurveNoSegments"
     radius="SiBarrelSiliconCurveRadius"
     phi0="-SiBarrelSiliconCurveHalfAngle1"
     phi1="-SiBarrelSiliconGapHalfAngle"
     length="SiBarrelMod1_length/4"
     thickness="SiBarrelSensorMod_thickness"
     vis="TrackerLayerVis">
  <position x="0" y="3*SiBarrelMod1_length/8" z="SiBarrelUpperCurveAxis1" />
</module_component>
```

- Identified by “Curved” in name
- Cylinder segment defined by radius, phi0, phi1, length, thickness
- Modelled as nsegments flat strips
- Inner and outer thicknesses must be given for active component
- Position – now required for all components – centre of cylinder axis relative to centre of module

```
<constant name="SiBarrelSiliconCurveNoSegments" value="5" />
<constant name="SiBarrelSiliconCurveRadius" value="100*mm" />
<constant name="SiBarrelSiliconSpace" value="1.26*mm"/>
<constant name="SiBarrelSiliconCurveHeight1" value="SiBarrelSiliconCurveRadius - sqrt(SiBarrelSiliconCurveRadius^2 - (SiBarrelStave1_width/2)^2)" />
<constant name="SiBarrelSiliconCurveHeight2" value="SiBarrelSiliconCurveRadius - sqrt(SiBarrelSiliconCurveRadius^2 - (SiBarrelStave2_width/2)^2)" />
<constant name="SiBarrelLowerCurveAxis1" value="SiBarrelSiliconCurveRadius - SiBarrelSiliconCurveHeight1 - SiBarrelMod1Frame_thickness - 0.5*SiBarrelBrace_thickness - SiBarrelSiliconSpace" />
<constant name="SiBarrelLowerCurveAxis2" value="SiBarrelSiliconCurveRadius - SiBarrelSiliconCurveHeight2 - SiBarrelMod2Frame_thickness - 0.5*SiBarrelBrace_thickness - SiBarrelSiliconSpace" />
<constant name="SiBarrelUpperCurveAxis1" value=" - SiBarrelSiliconCurveRadius + SiBarrelSiliconCurveHeight1 + SiBarrelMod1Service_thickness + 0.5*SiBarrelBrace_thickness + SiBarrelSiliconSpace" />
<constant name="SiBarrelUpperCurveAxis2" value=" - SiBarrelSiliconCurveRadius + SiBarrelSiliconCurveHeight2 + SiBarrelMod2Service_thickness + 0.5*SiBarrelBrace_thickness + SiBarrelSiliconSpace" />
<constant name="SiBarrelSiliconCurveHalfAngle1" value="asin(0.5*SiBarrelStave1_width / SiBarrelSiliconCurveRadius)" />
<constant name="SiBarrelSiliconCurveHalfAngle2" value="asin(0.5*SiBarrelStave2_width / SiBarrelSiliconCurveRadius)" />
<constant name="SiBarrelSiliconGapHalfAngle" value="atan(0.5*SiBarrelSiliconGap_width / SiBarrelSiliconCurveRadius)" />
<constant name="SiBarrel_ShortThickness" value="0.5*SiBarrelSensorMod_thickness" />
<constant name="SiBarrel1_LongThickness" value="1.5*SiBarrelSensorMod_thickness + SiBarrelMod1Service_thickness + SiBarrelMod1Frame_thickness +  SiBarrelBrace_thickness + 2*SiBarrelSiliconSpace + SiBarrelSiliconCurveHeight1"  />
<constant name="SiBarrel2_LongThickness" value="1.5*SiBarrelSensorMod_thickness + SiBarrelMod2Service_thickness + SiBarrelMod2Frame_thickness +  SiBarrelBrace_thickness + 2*SiBarrelSiliconSpace + SiBarrelSiliconCurveHeight1"  />
```

XML <readout>

```
<readouts>
  <readout name="SiBarrelHits">
    <segmentation type="CartesianGridXY" grid_size_x="0.020*mm" grid_size_y="0.020*mm" />
    <id>system:8,layer:4,module:7,sensor:7,x:32:-12,y:-20</id>
  </readout>
</readouts>
```

Maximum number of sensors (L4)

Si modules x sides x nsegments
8 x 2 x 5 = 80
≥7 bits needed

76 DD4HEP modules (staves) for L4
≥7 bits needed

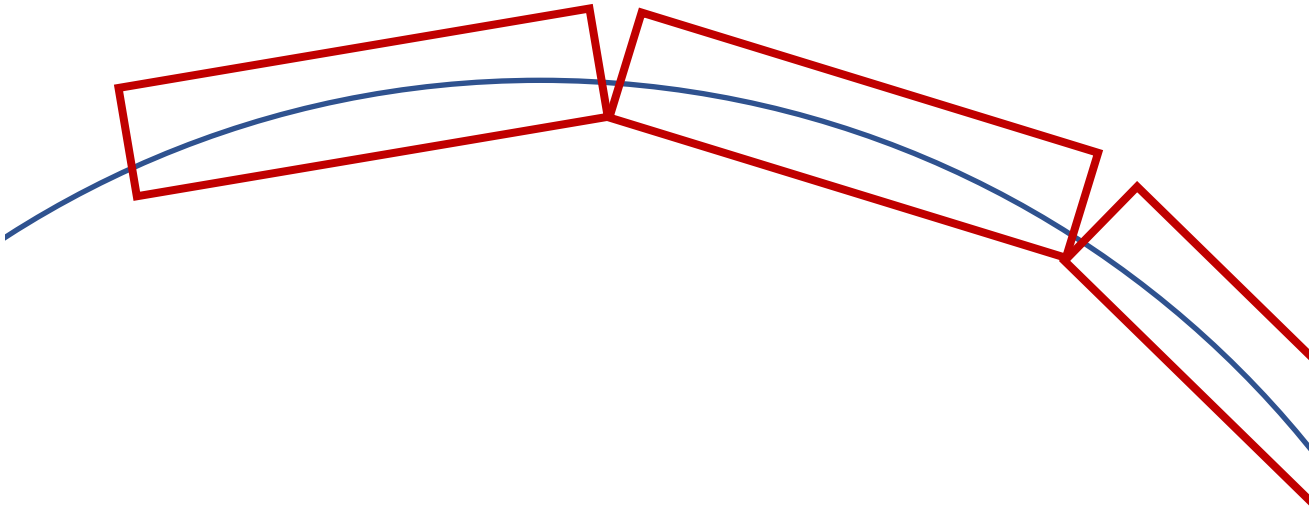
- CylinderGridPhiZ does not work for cylinders where axis is not the beamline
- Two layers – for L3 and L4 – overlapping top and bottom silicon not a problem as separate DD4HEP sensors

BarrelTrackerWithCurves_geo.cpp

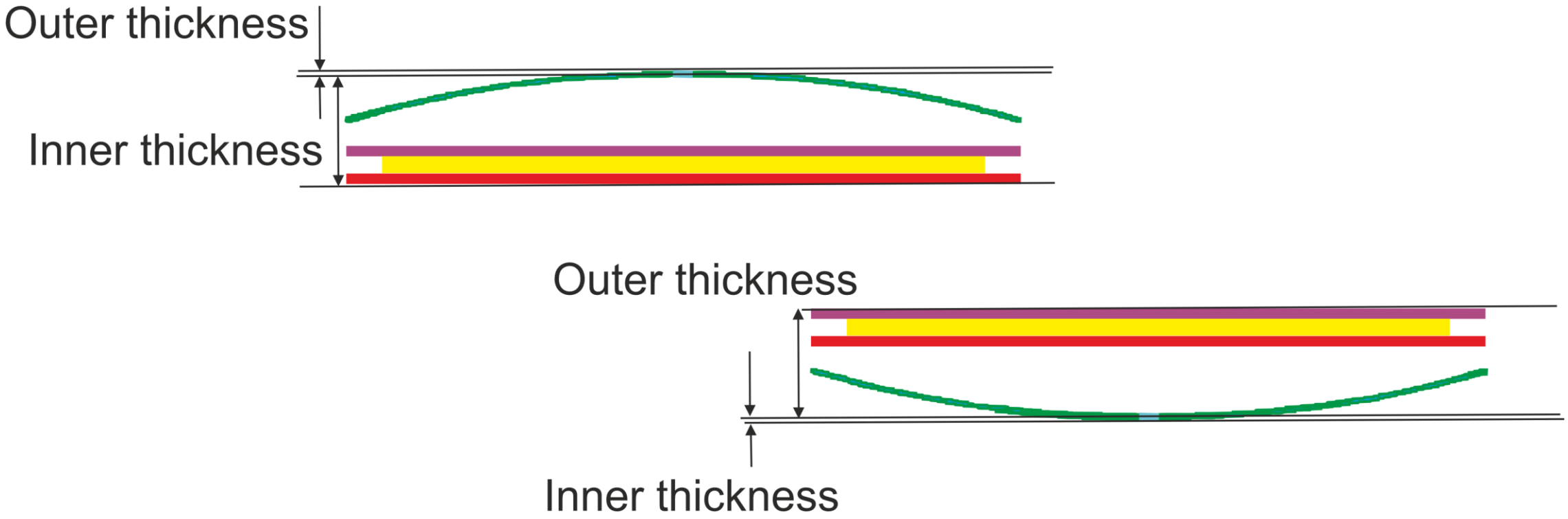
- Derived from BarrelTrackWithFrame_geo.cpp

```
Volume c_vol;
if (x_comp.nameStr().find("Curved") != std::string::npos) {
  double c_rmin = x_comp.radius(); // radius of curvature in cm
  double c_phi0 = x_comp.phi0(); // start and stop angle of segment in rad - zero is centre of stave
  double c_phi1 = x_comp.phi1();
  double c_Nseg = x_comp.nsegments(); //  number of flat segments used to approximate cylinder
  double dphi = (c_phi1 - c_phi0) / (double)c_Nseg;
  vector<double> Xp, Zp;
  double phiP = c_phi0;

  for(int i=0; i<c_Nseg+1; ++i){
      Xp.push_back(c_rmin * sin(phiP));
      Zp.push_back(c_rmin*cos(phiP));
      phiP += dphi;
  }
  phiP = c_phi0 + dphi/2;
       for(int i=0; i<c_Nseg; ++i){
      double segwidth = sqrt( (Xp[i+1] - Xp[i])*(Xp[i+1] - Xp[i]) + (Zp[i+1] - Zp[i])*(Zp[i+1] - Zp[i]));
      double midX = 0.5*(Xp[i] + Xp[i+1]);
      double midZ = 0.5*(Zp[i] + Zp[i+1]);
      Box seg_box ((segwidth/2) * ((c_rmin - x_comp.thickness()/2)/c_rmin), x_comp.length()/2, x_comp.thickness()/2);
        Trd1 seg_box( (segwidth/2) * ((c_rmin - x_comp.thickness()/2)/c_rmin), (segwidth/2) * ((c_rmin + x_comp.thickness()/2)/c_rmin), x_comp.length()/2, x_comp.thickness()/2);
      string seg_nam = c_nam + "." + _toString(i);
      Volume seg_vol(seg_nam, seg_box, description.material(x_comp.materialStr()));

       if (x_pos && x_rot) {
            Position c_pos(midX + x_pos.x(0), x_pos.y(0), midZ + x_pos.z(0));
            RotationZYX c_rot(x_rot.z(0), phiP+ x_rot.y(0), x_rot.x(0));
            pv = m_vol.placeVolume(seg_vol, Transform3D(c_rot, c_pos));
       } else if (x_rot) {
           Position c_pos(midX, 0, midZ);
           pv = m_vol.placeVolume(seg_vol, Transform3D(RotationZYX(x_rot.z(0), x_rot.y(0) + phiP, x_rot.x(0)), c_pos));
       } else if (x_pos) {
           RotationZYX c_rot(0, phiP, 0);
           Position c_pos(midX+x_pos.x(0), x_pos.y(0), x_pos.z(0)+midZ);
           pv = m_vol.placeVolume(seg_vol, Transform3D(c_rot, c_pos));
      } else {
           RotationZYX c_rot(0, phiP, 0);
           Position c_pos(midX, 0, midZ);
           pv = m_vol.placeVolume(seg_vol, Transform3D(c_rot, c_pos));
      }
      phiP += dphi;
      seg_vol.setRegion(description, x_comp.regionStr());
      seg_vol.setLimitSet(description, x_comp.limitsStr());
      seg_vol.setVisAttributes(description, x_comp.visStr());
if (x_comp.isSensitive()) {
  pv.addPhysVolID("sensor", sensor_number++);
  seg_vol.setSensitiveDetector(sens);
  sensitives[m_nam].push_back(pv);
    module_thicknesses[m_nam] = {x_comp.innerthickness(),x_comp.outerthickness()};

  // -------- create a measurement plane for the tracking surface attched to the sensitive volume -----
  Vector3D u(-1., 0., 0.);
  Vector3D v(0., -1., 0.);
  Vector3D n(0., 0., 1.);
  //    Vector3D o( 0. , 0. , 0. ) ;

  // compute the inner and outer thicknesses that need to be assigned to the tracking surface
  // depending on wether the support is above or below the sensor
```

- Gap ~1.5µm to avoid overlap
- Trapedoid (Trd1) volume not compatible with ACTS readout. TessellatedSolid may work



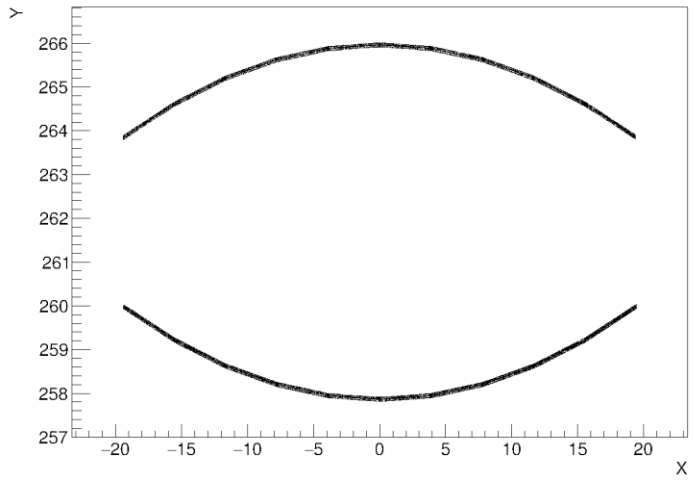
Outer and inner thicknesses – required for sensitive surface

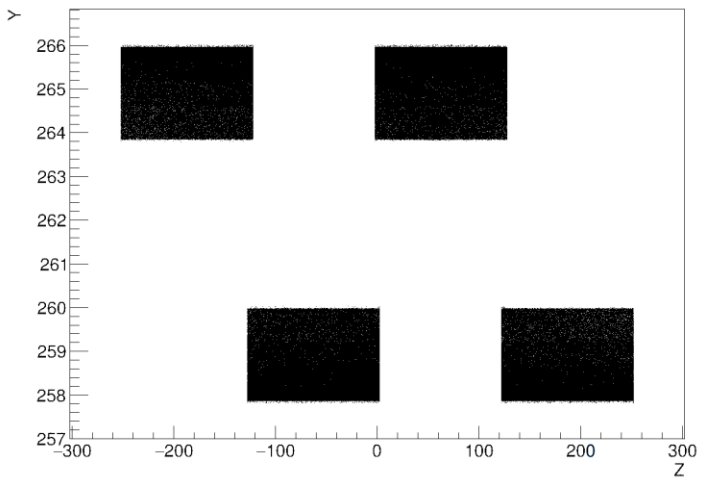


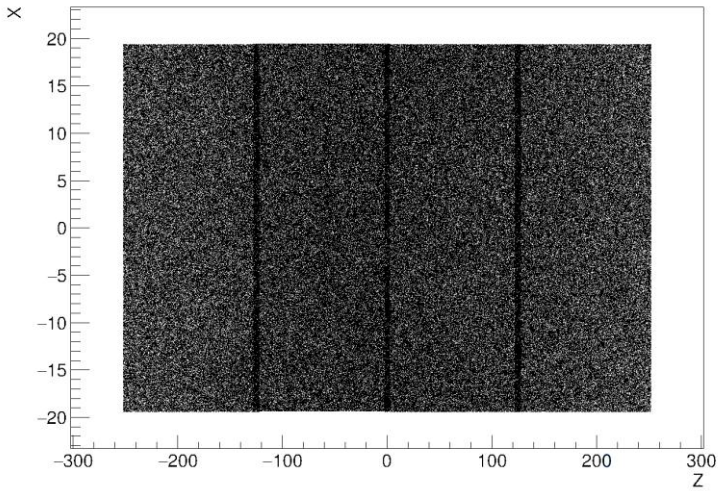
Dimensions of active surface from latest CAD (Feb 2026)

Single stave hit maps from SVTOB_UK, SiBarrelHits

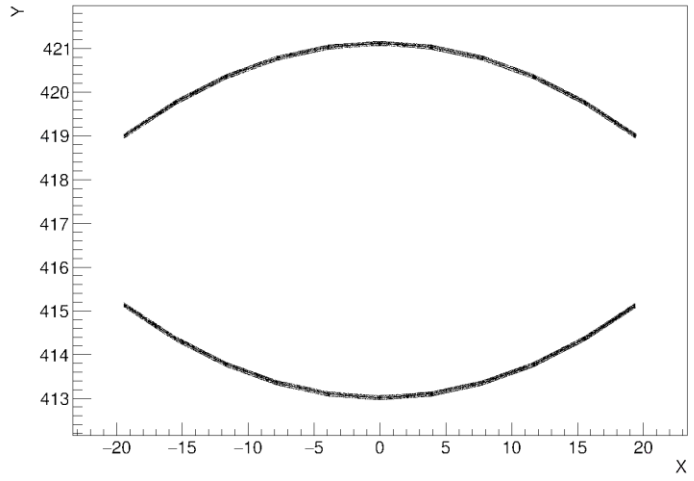
L3

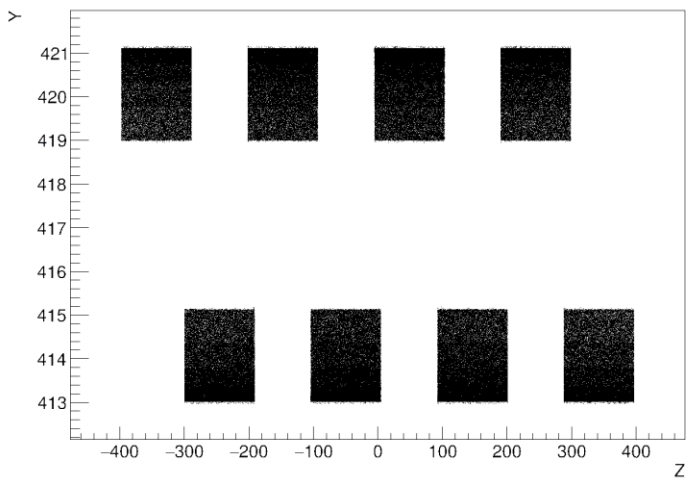


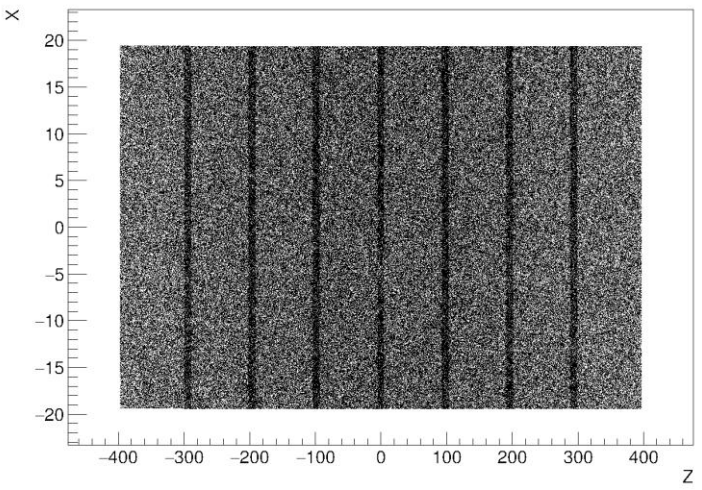




L4



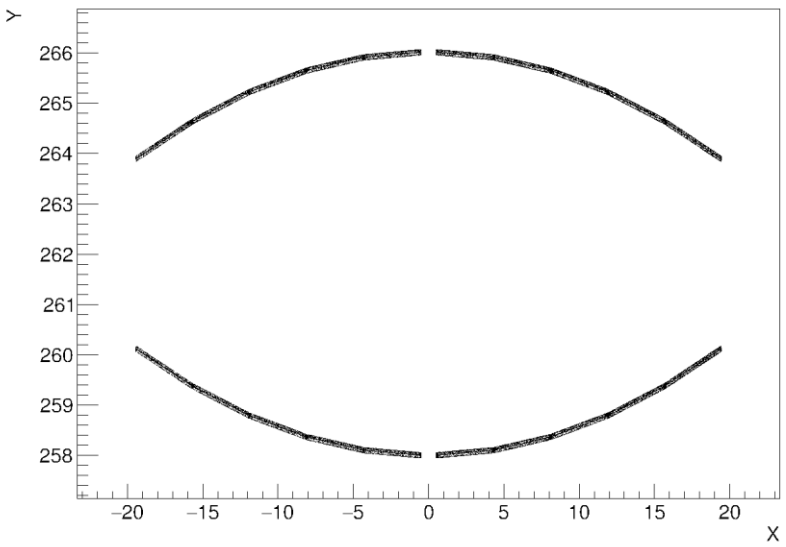


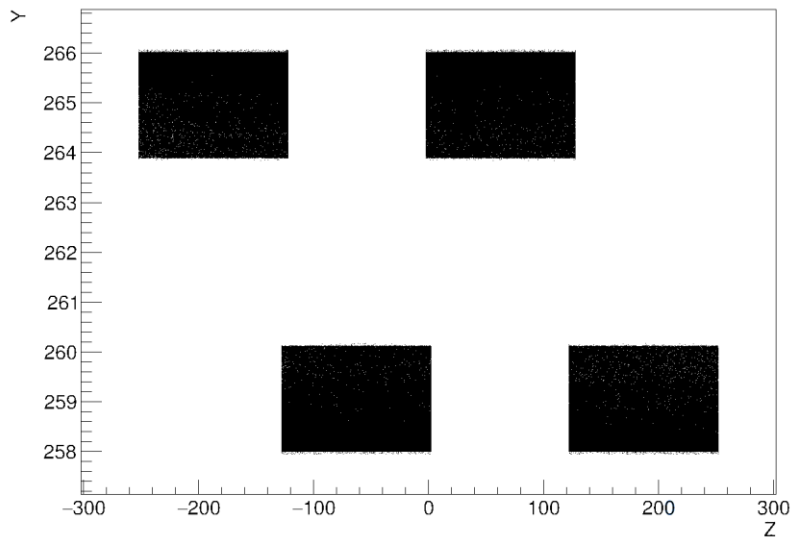


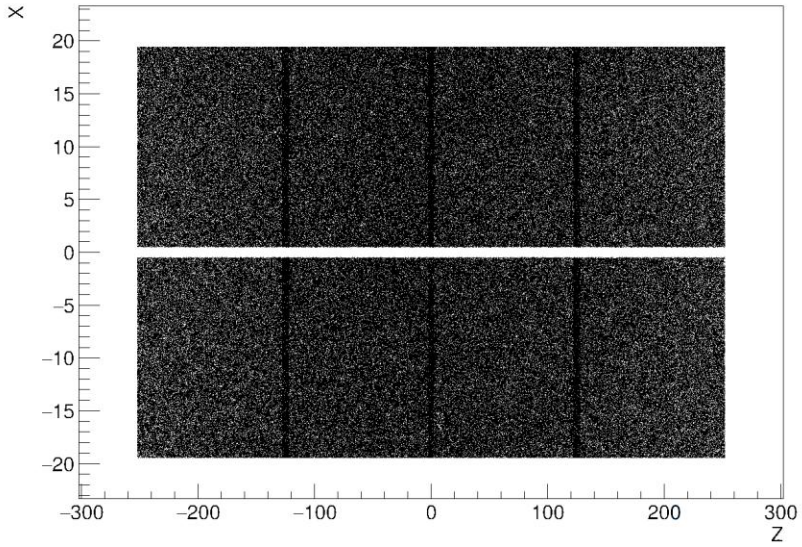
curved_modular_OB hit maps

Single stave, SiBarrelHits

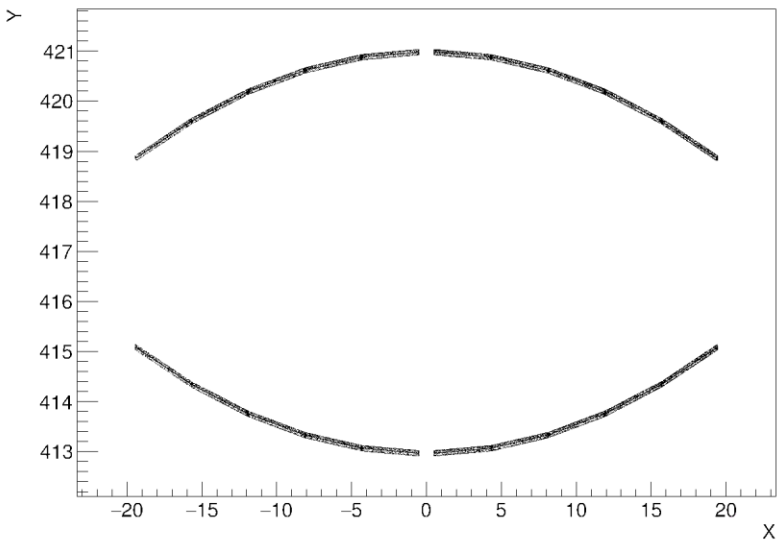
L3

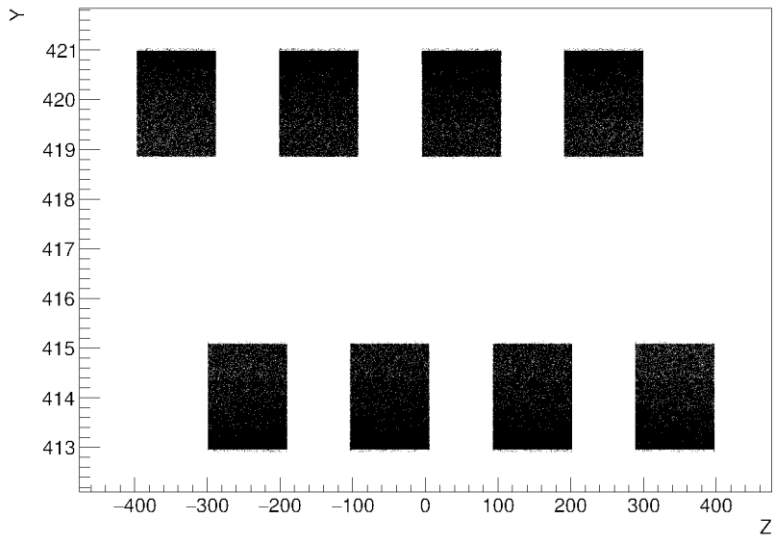


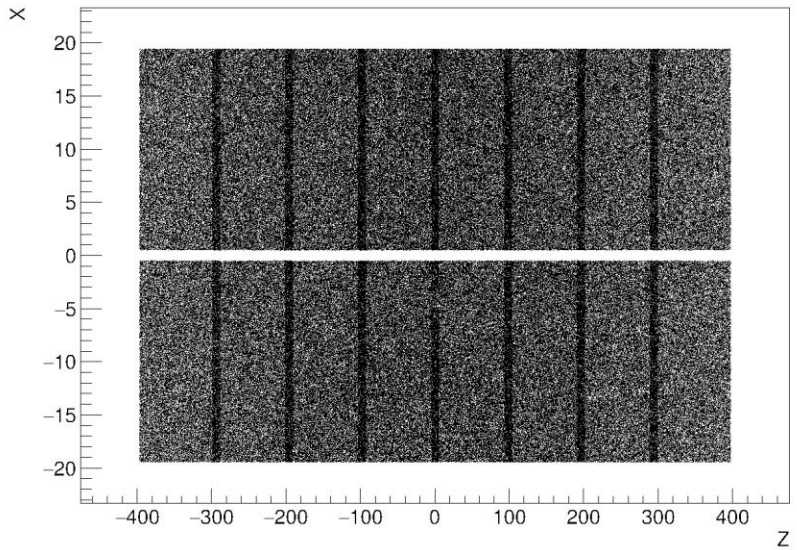




L4



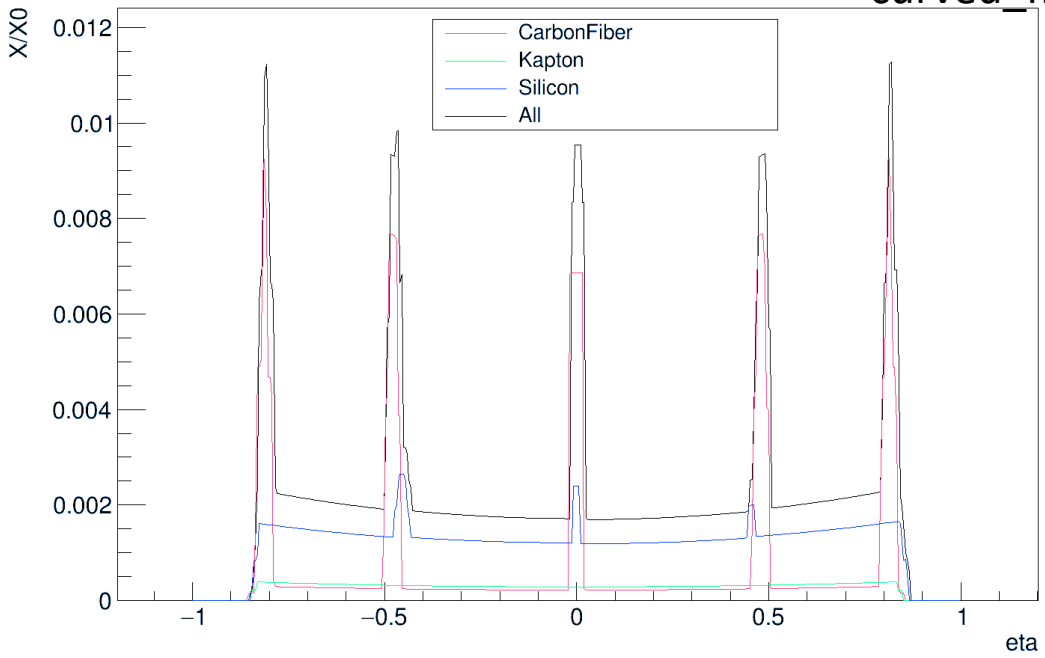


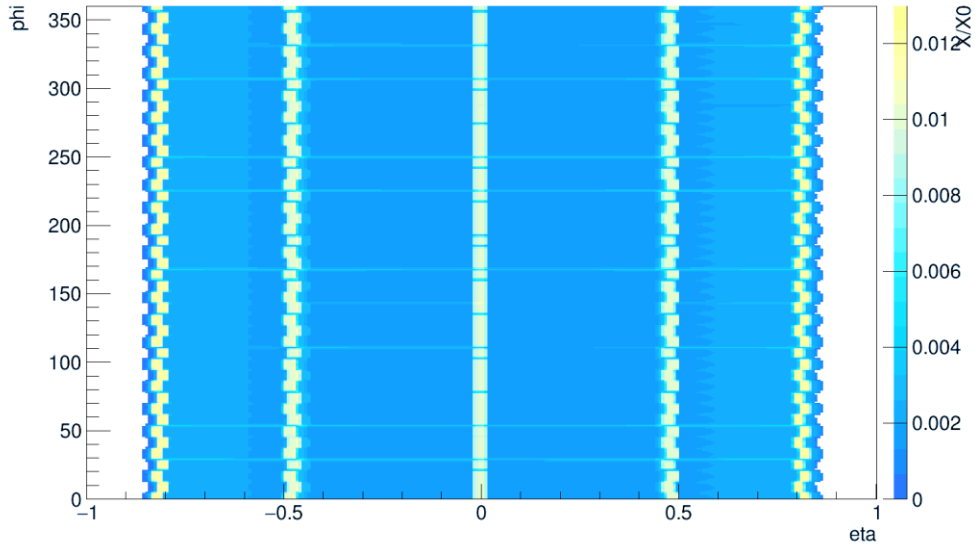


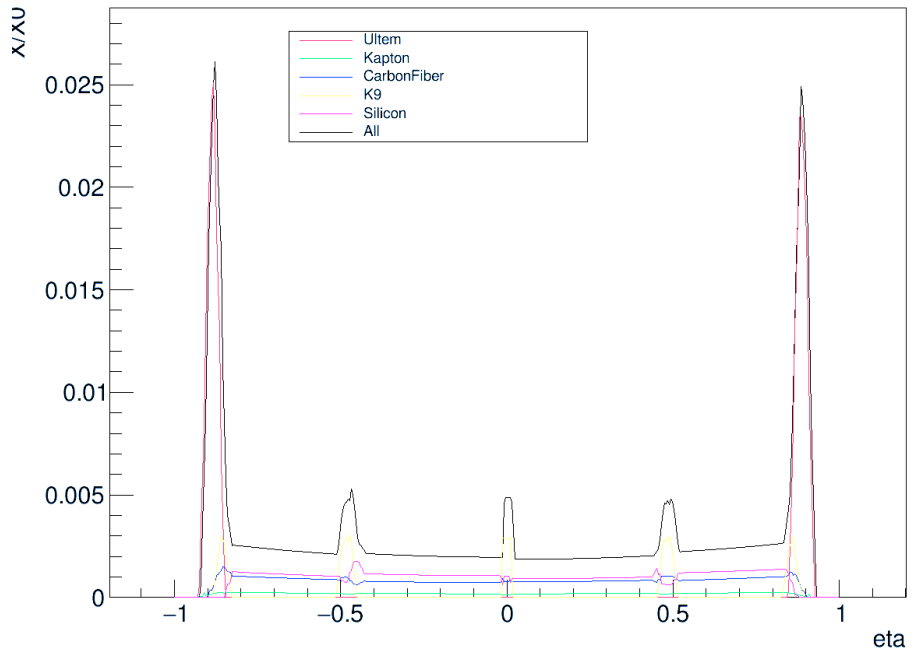
Material Thickness Scans: L3

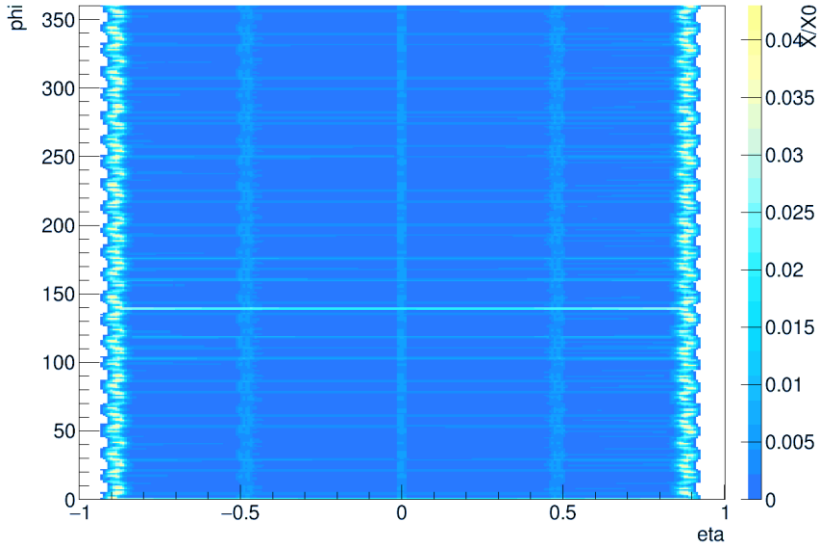
CAD February 2026

curved_modular_OB



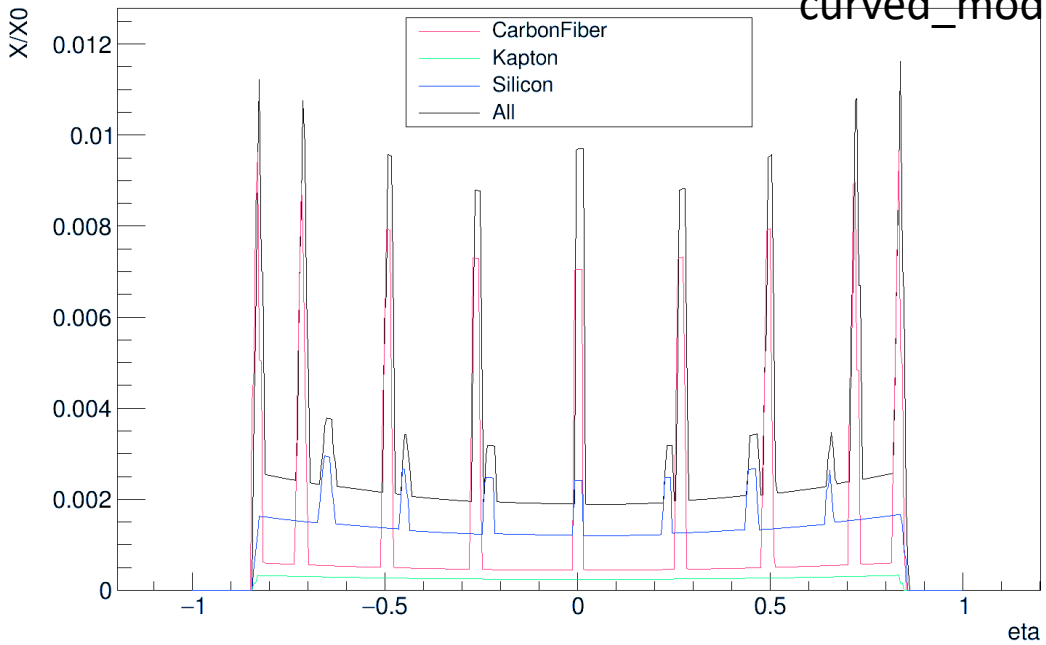


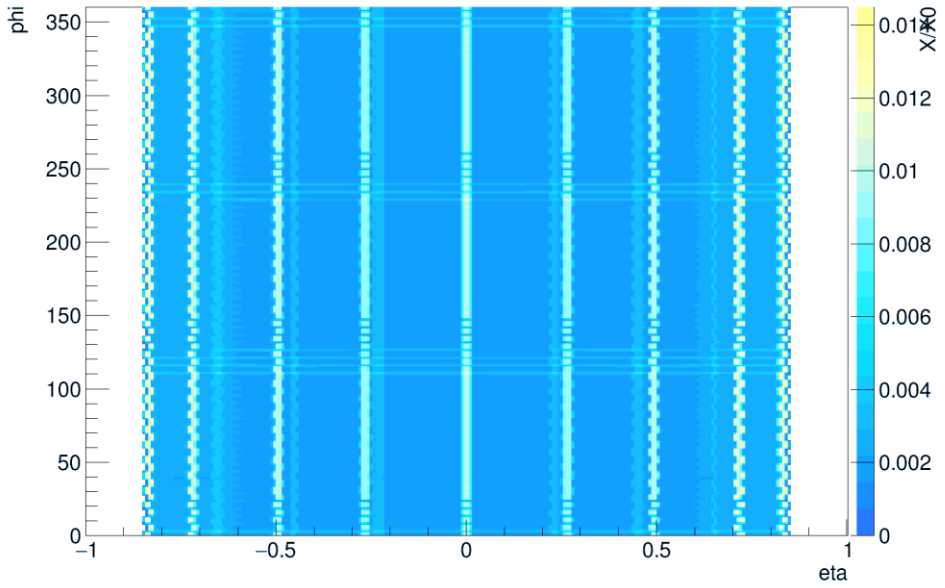




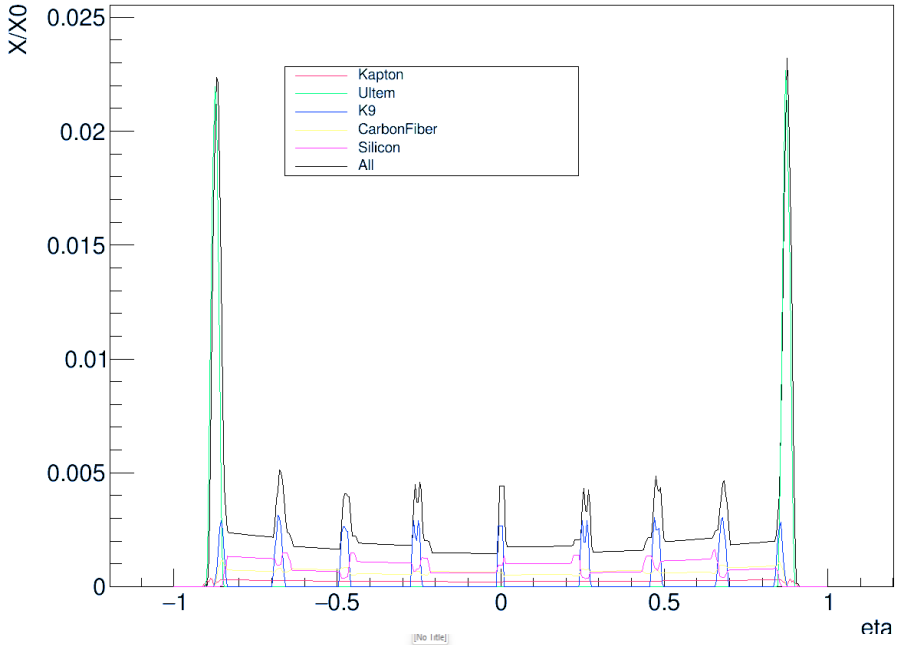
Material Thickness Scans: L4

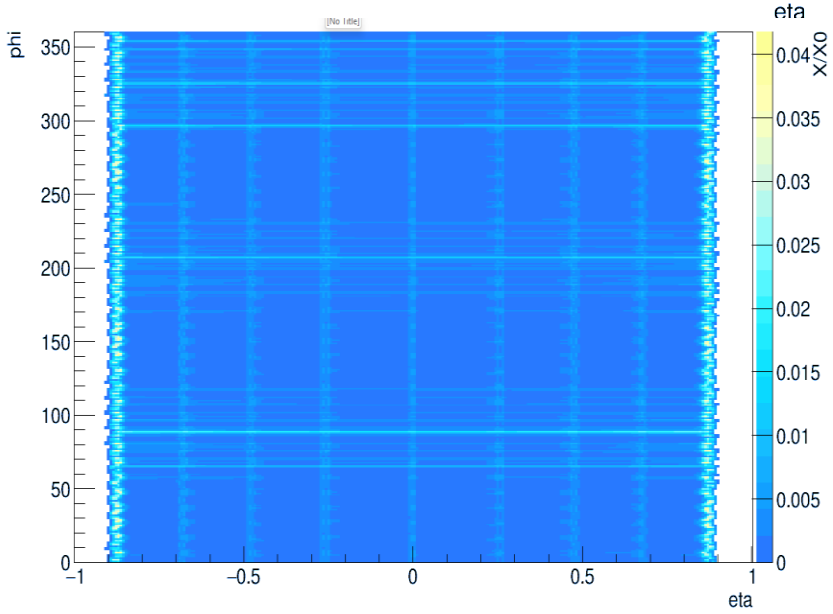
curved_modular_OB





CAD February 2026



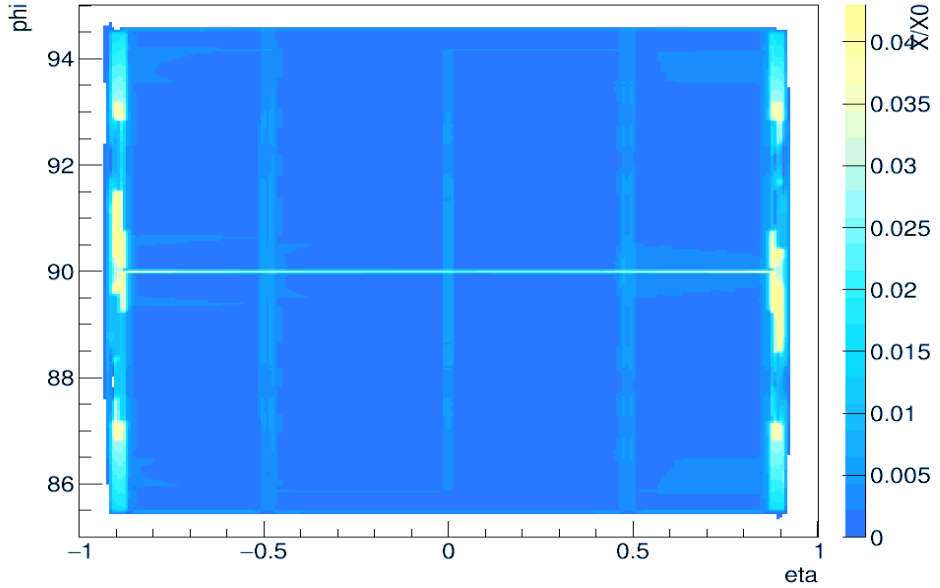


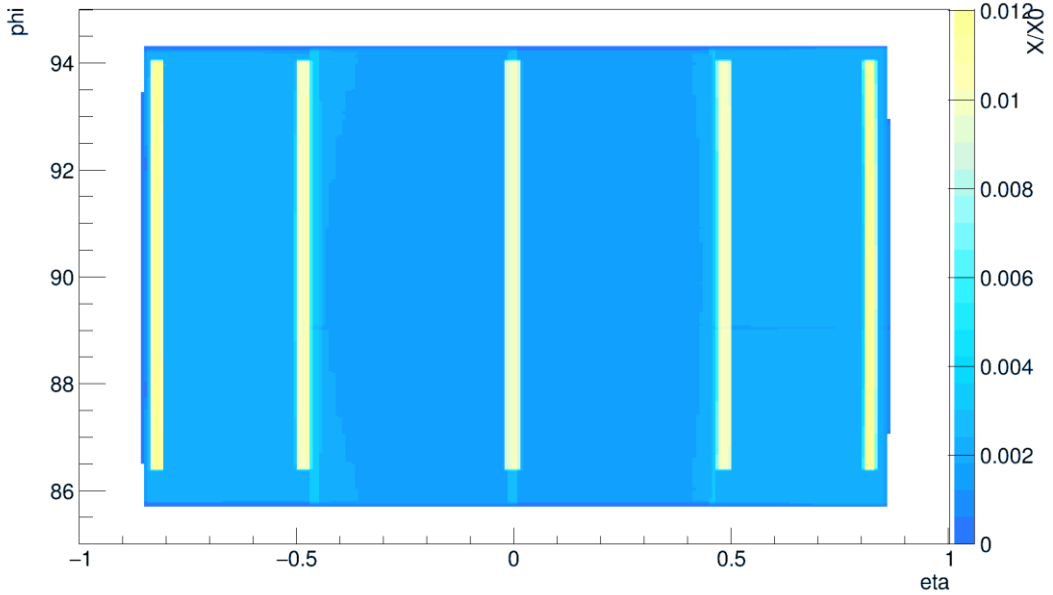
Single stave material thickness scans

curved_modular_OB

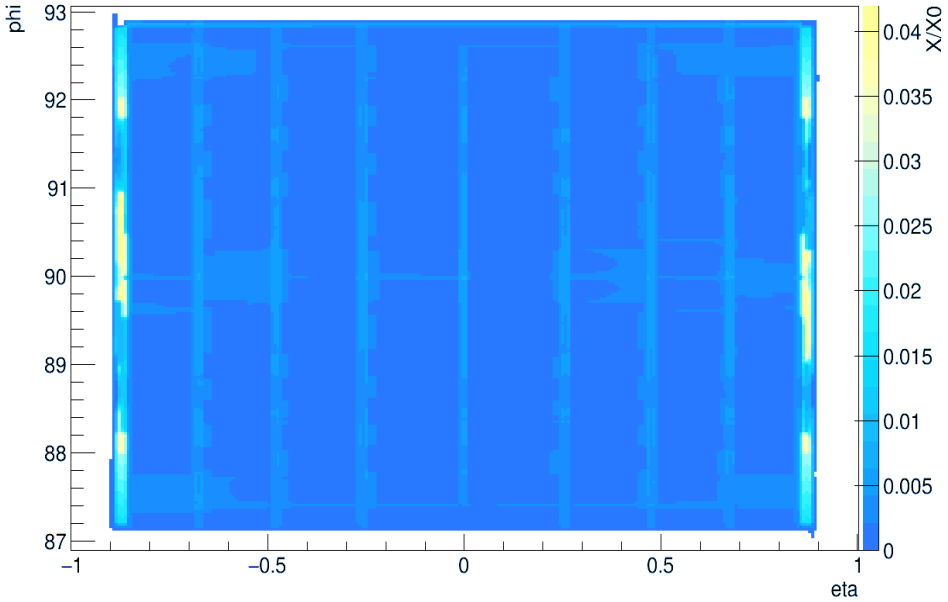
CAD 2026

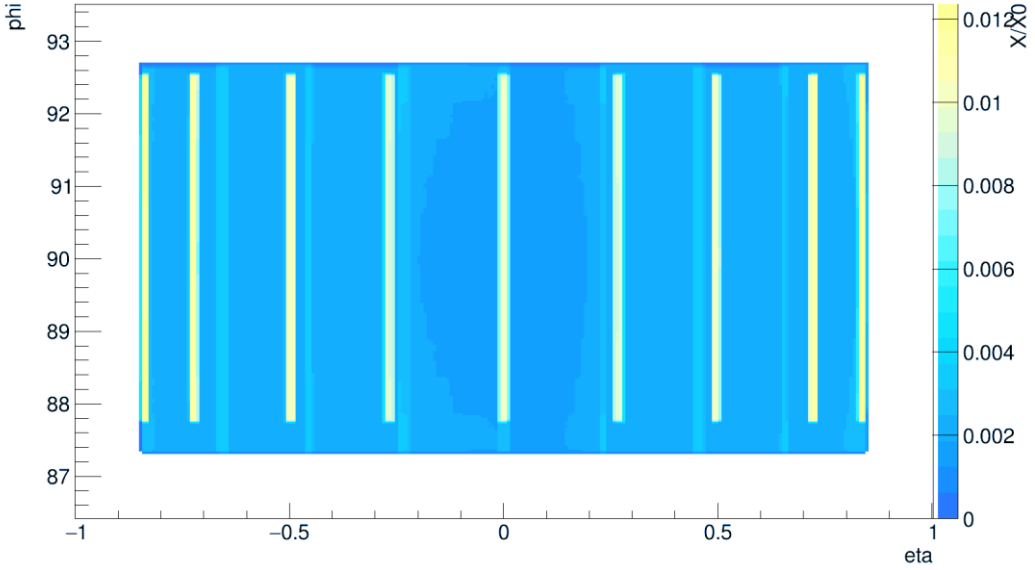
L3





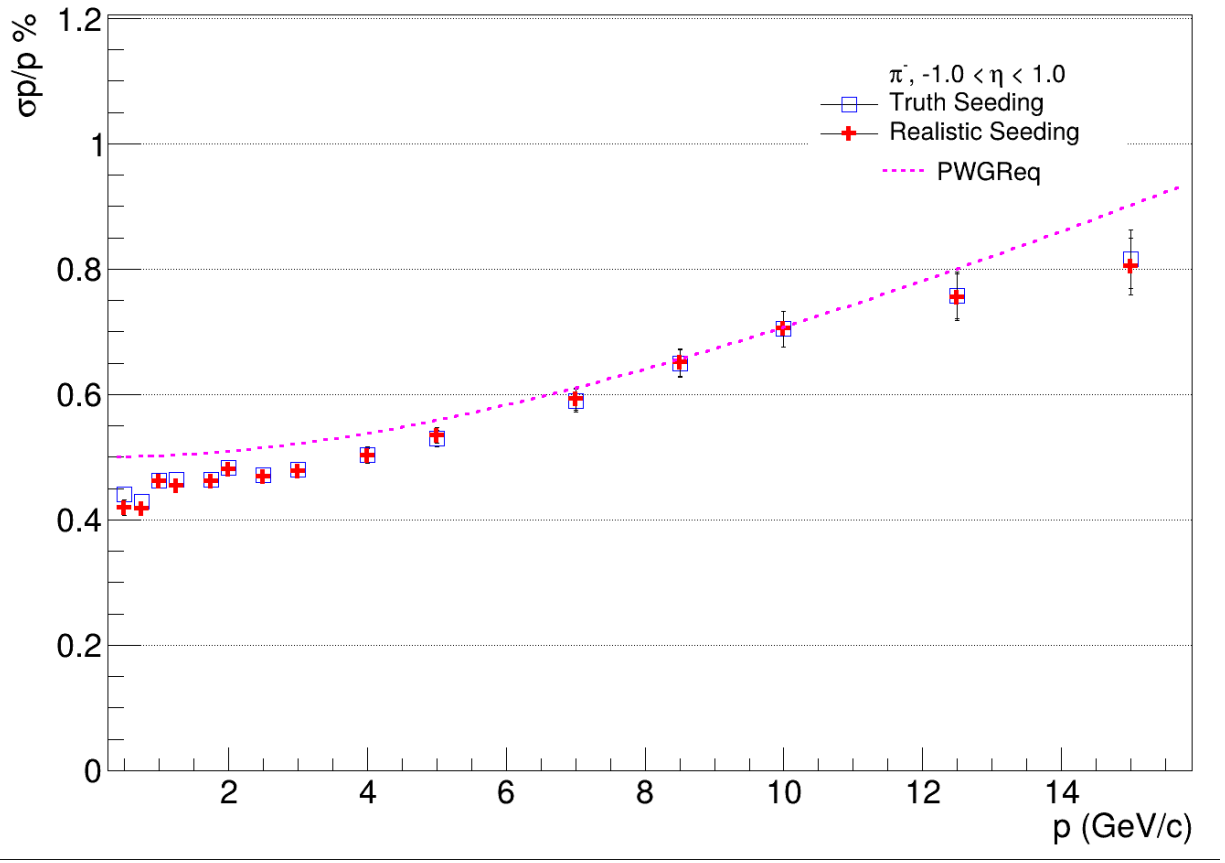
L4



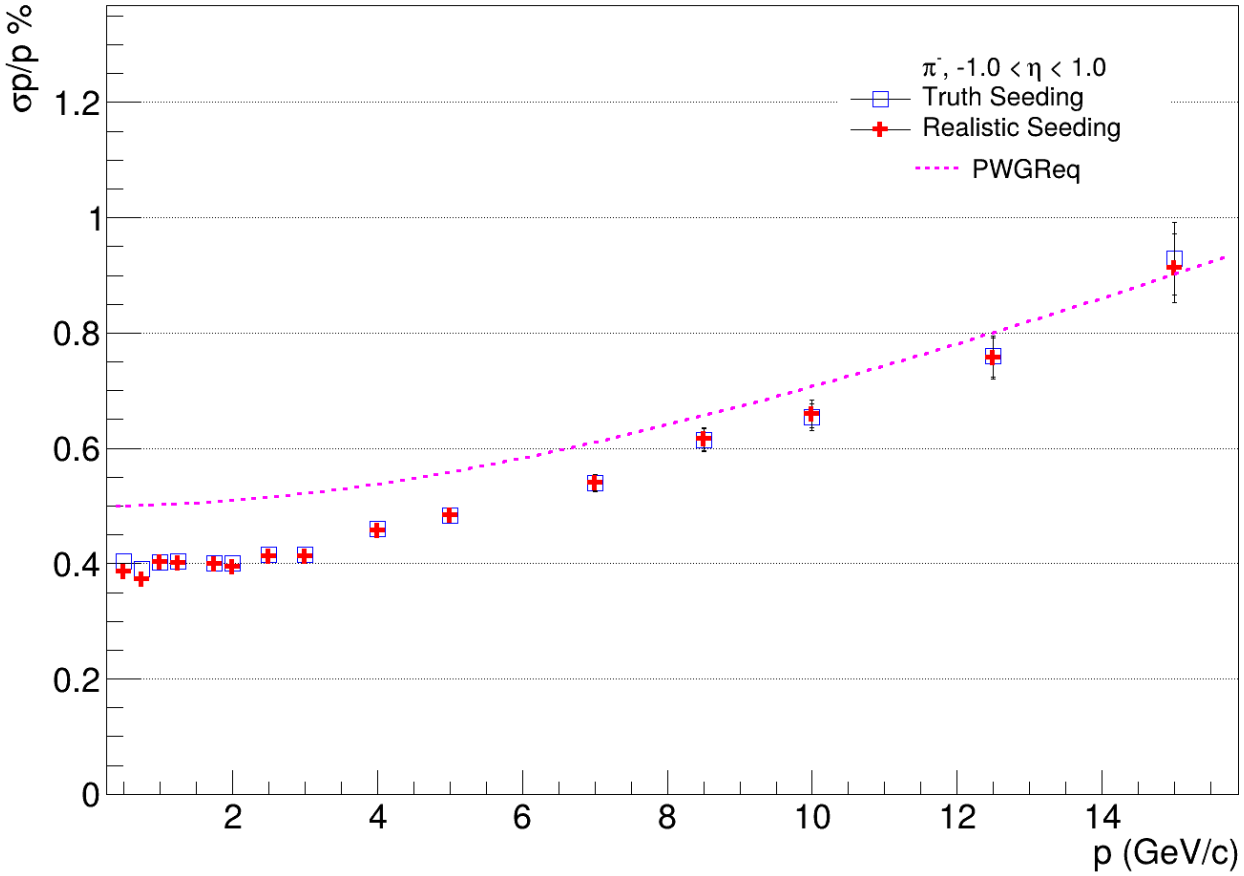


Tracking Benchmark Plots - momentum

curved_modular_OB
29 April 2026
Material map updated

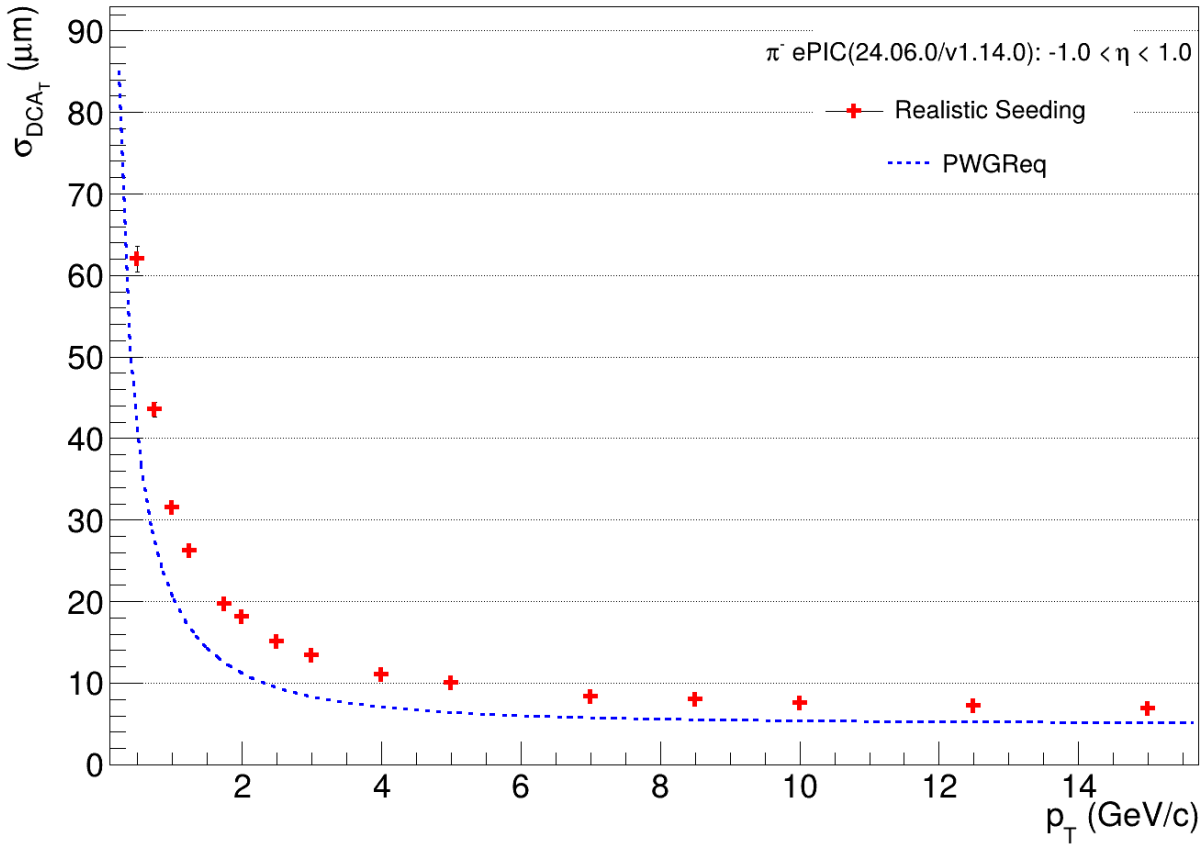


epic-main
29 April 2026

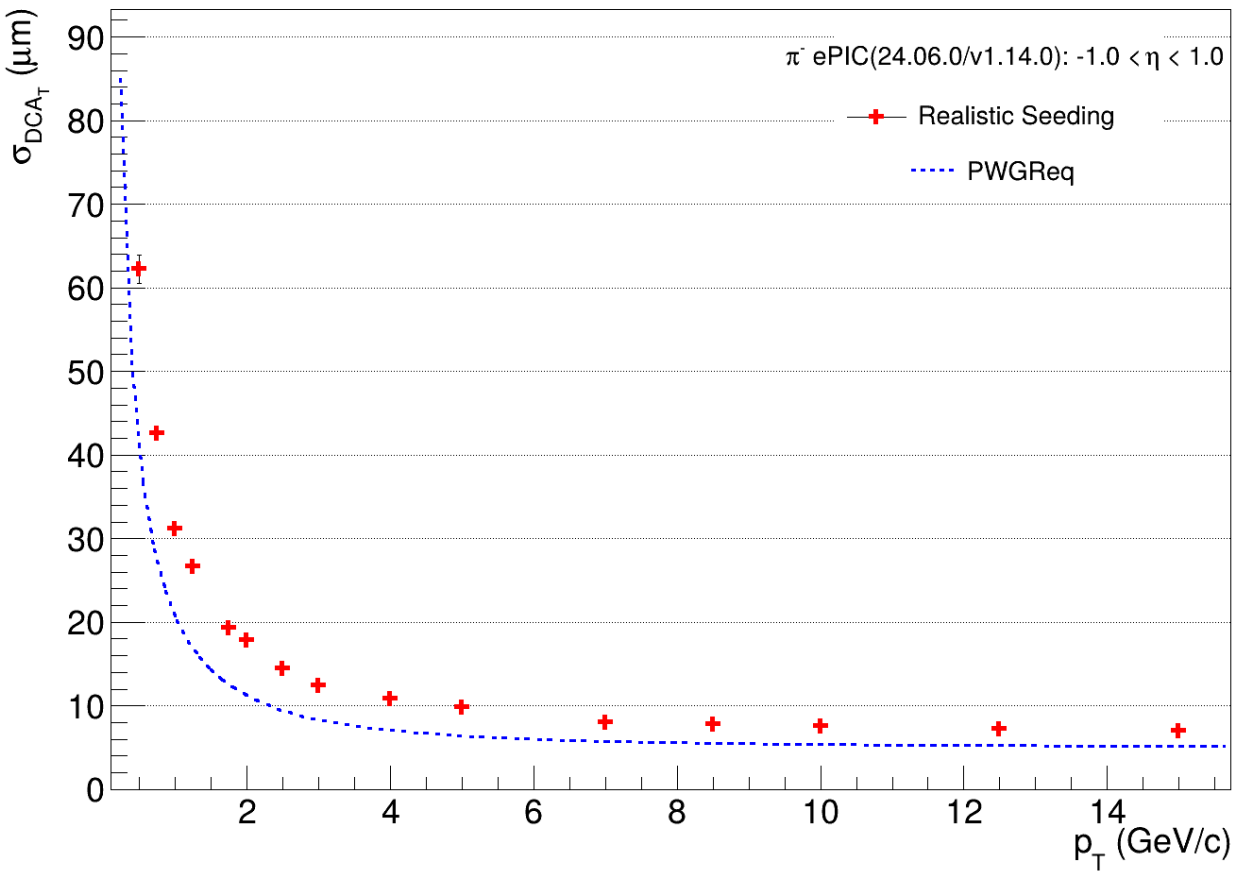


Tracking Benchmark Plots - dca_T

curved_modular_OB
29 April 2026
Material map updated

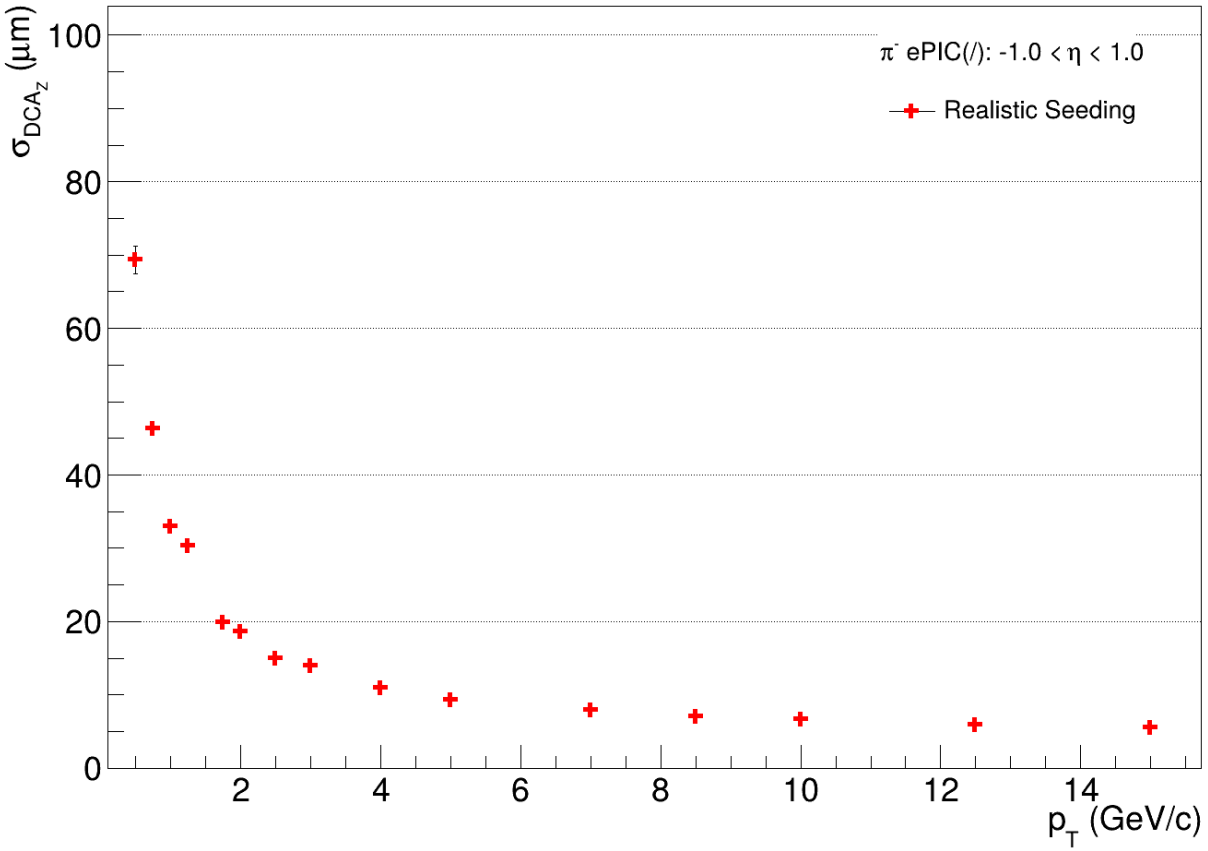


epic-main
29 April 2026

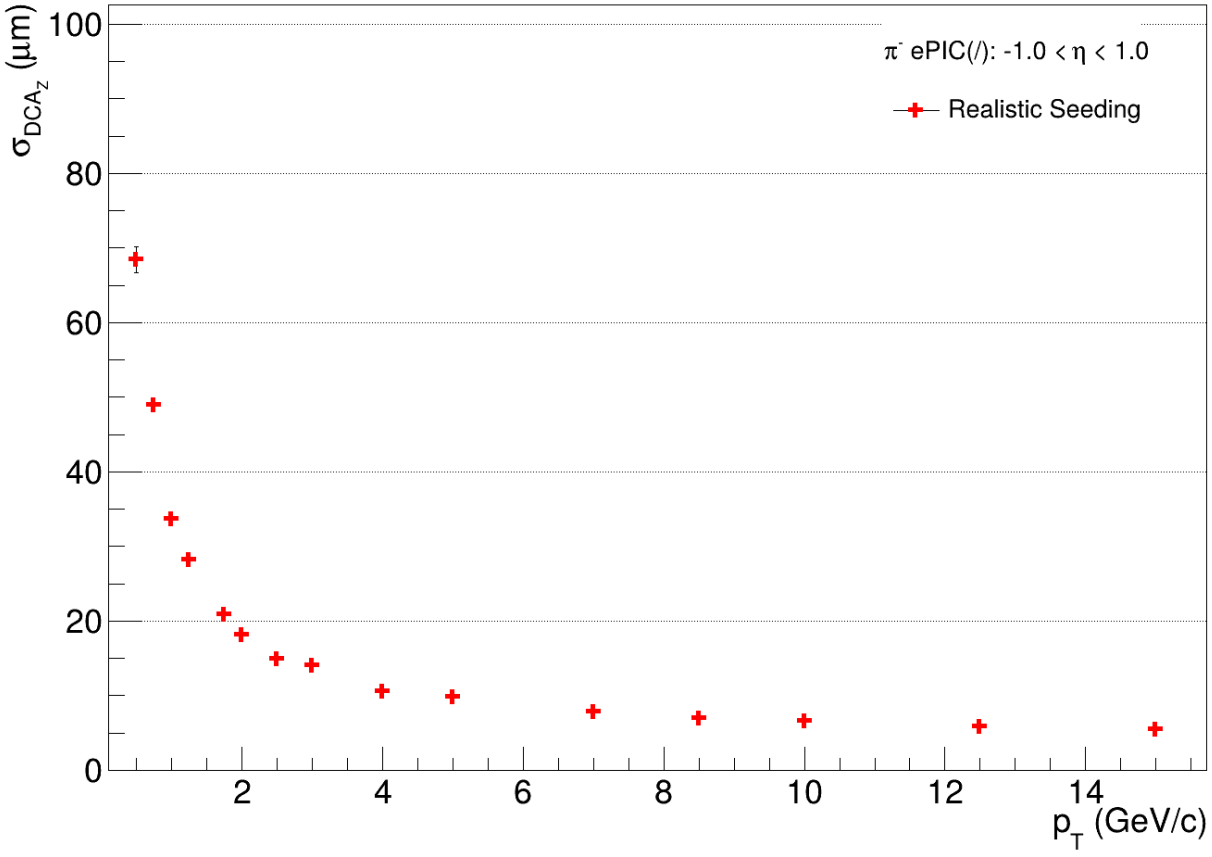


Tracking Benchmark Plots - dca_z

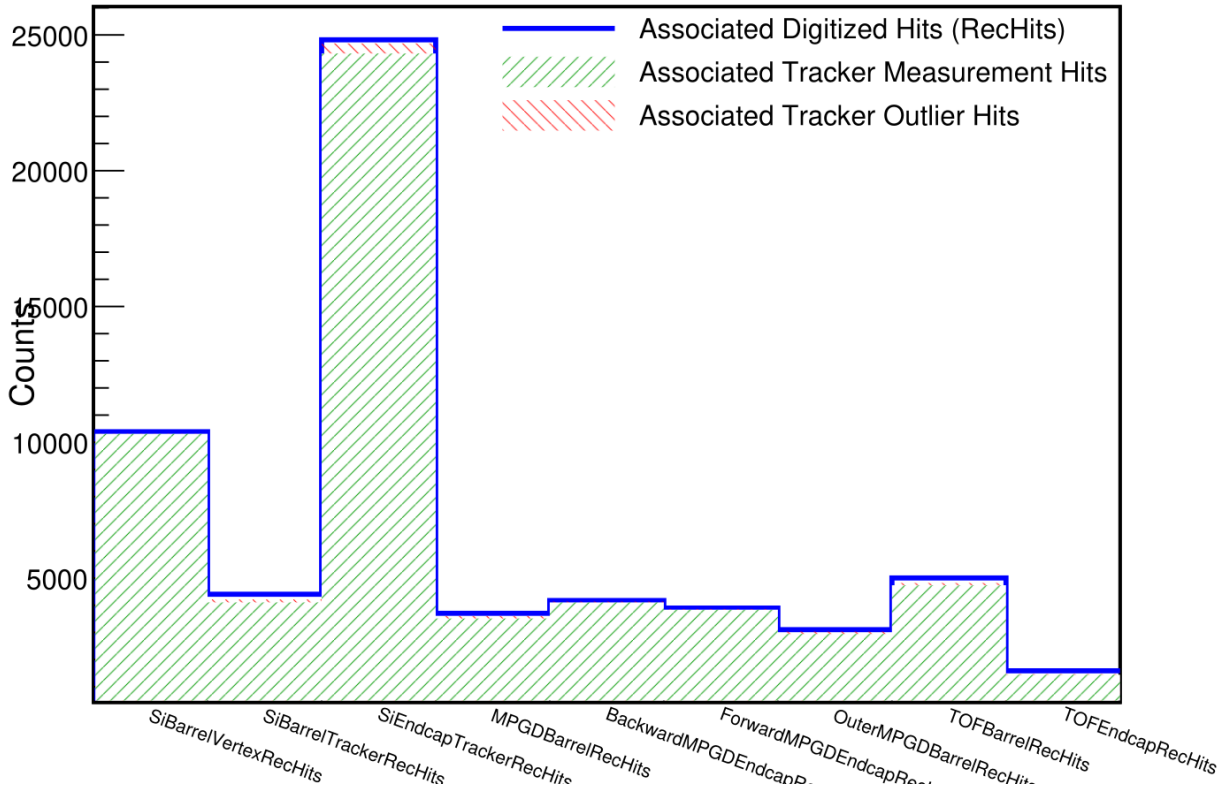
curved_modular_OB
29 April 2026
Material map updated

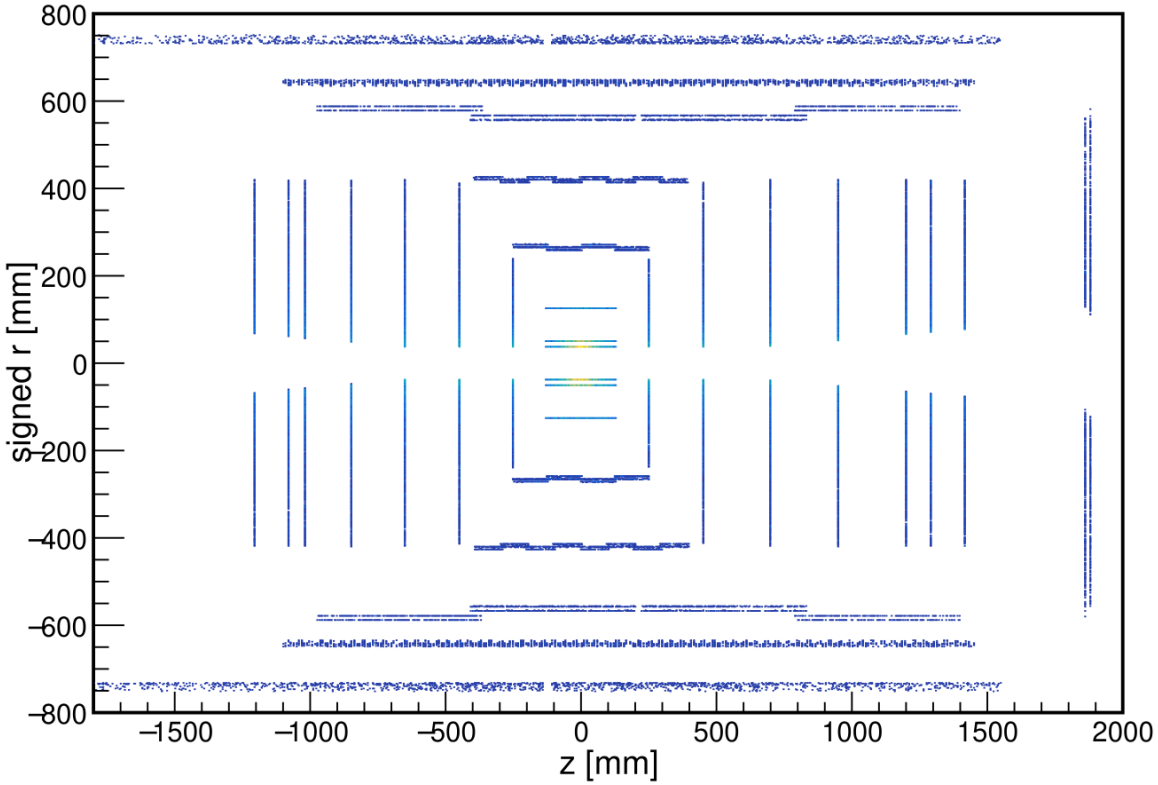


epic-main
29 April 2026



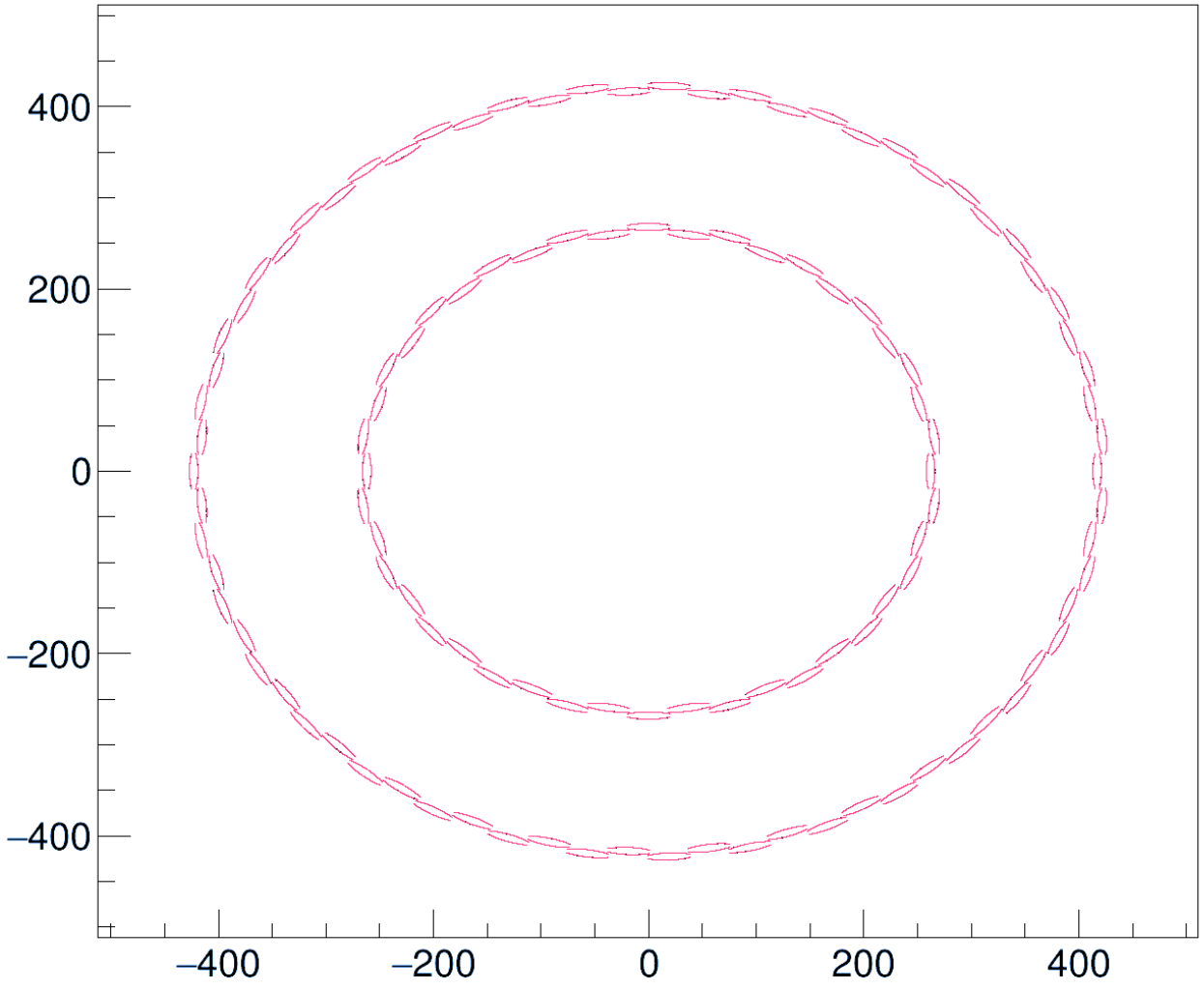
Hit_matching.C output – eicrecon is using SiBarrel hits

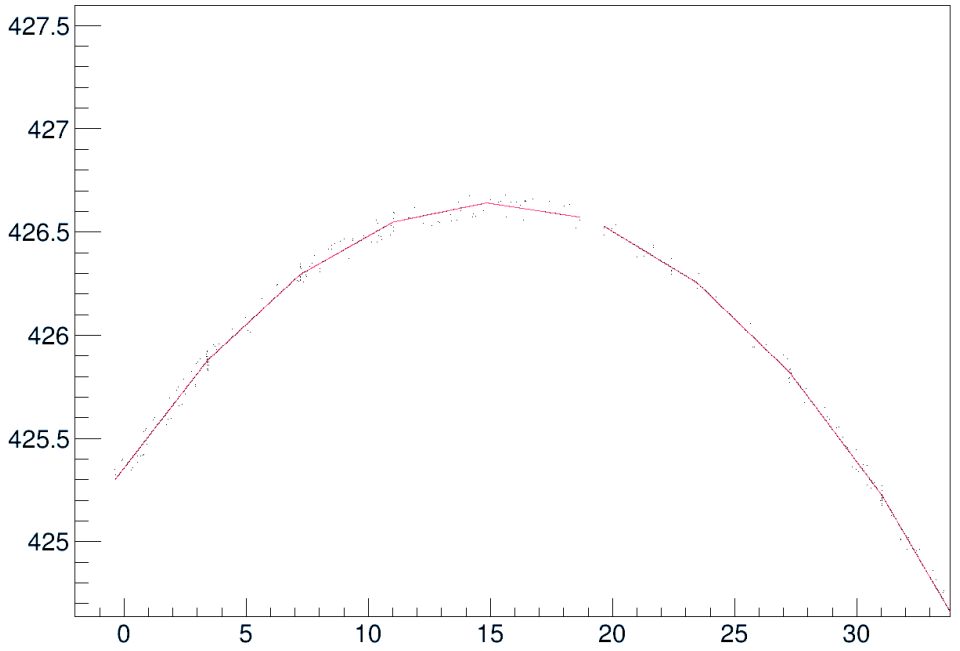




SiBarrelHits

SiBarrelTrackerRecHits





Summary and Next Steps

Curved Modular OB is now ready to go, pending approval

Next steps: Small Improvements

- Adjust spacing and thickness of braces to match latest CAD
- Add Ultem endcaps

Hybrid geometry

- Combine curved_modular_OB and SVTOB_UK branches
- Passive structure defined by GDML files
- Work in progress – first test successful
- All work to be completed by September 2026