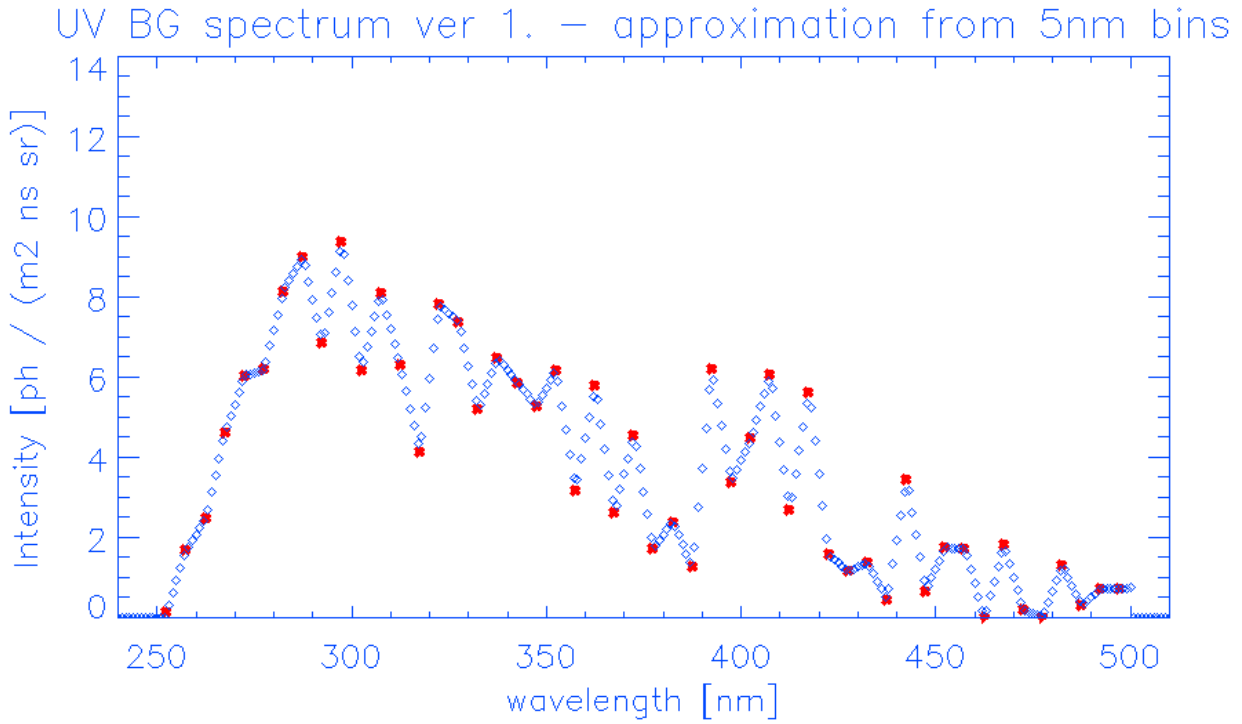
Update of ray trace background

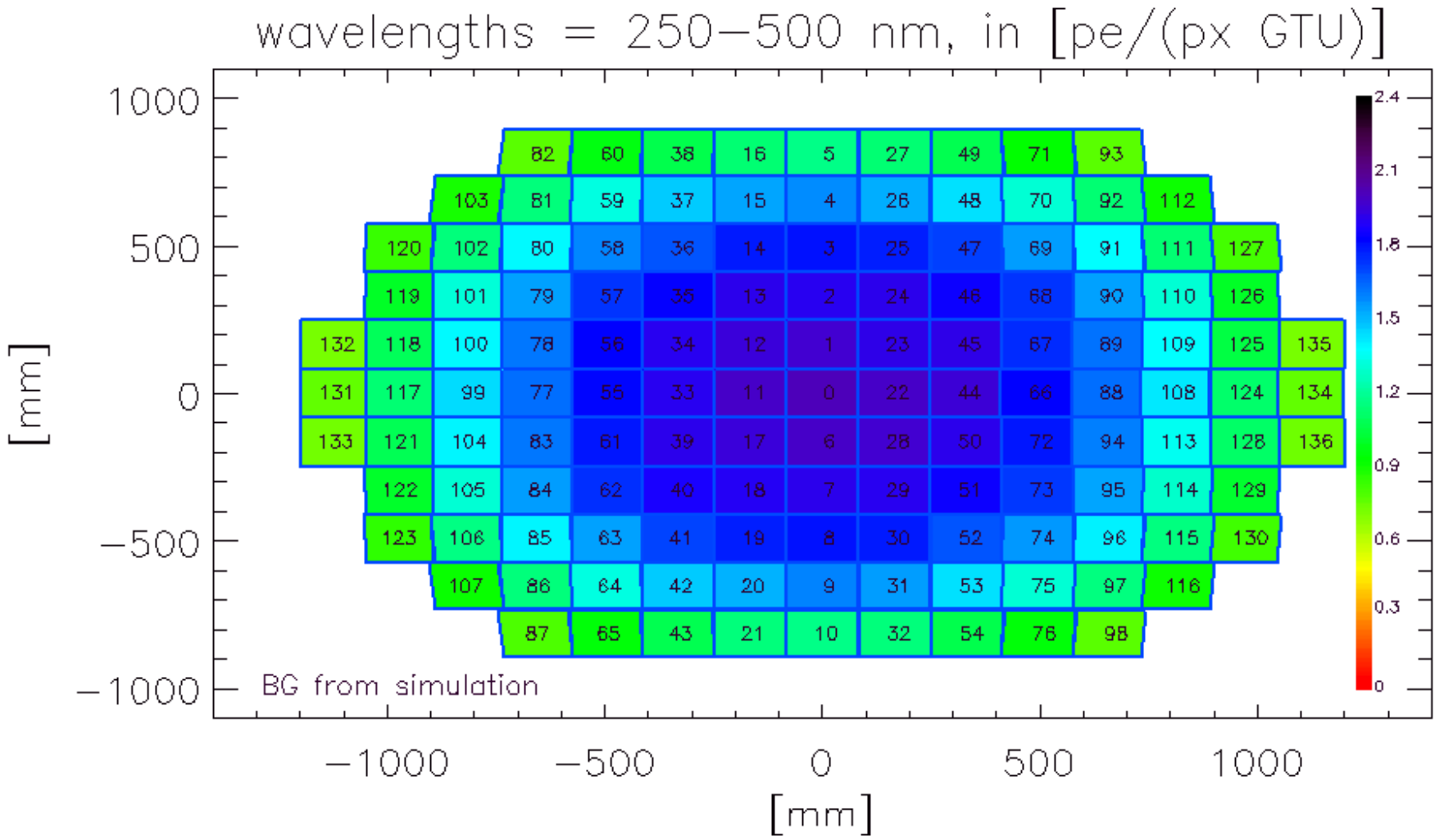
P. Bobik, K. Kudela, B. Pastircak,
M. Putis, K. Shinozaki

Simulation of raytrace BG at FS

- Simulation code for raytrace BG: Ccode-ns-100911-101019_release_for_ESA
- Input file for optics: telparm_PPP_2010_08_NOptics_v2.dat
- Input spectrum: JEM-EUSO UV BG spectrum ver.1
  - 953 ph/(m^2 ns sr) in 250-500 nm
  - 500 ph/(m^2 ns sr) in 300-400 nm



Output of raytrace BG simulation at FS



Average - 1.42 pe/(px GTU)
PDM with maximal value of BG - 2.02 pe/(px GTU)
PDM with minimal value of BG - 0.73 pe/(px GTU)

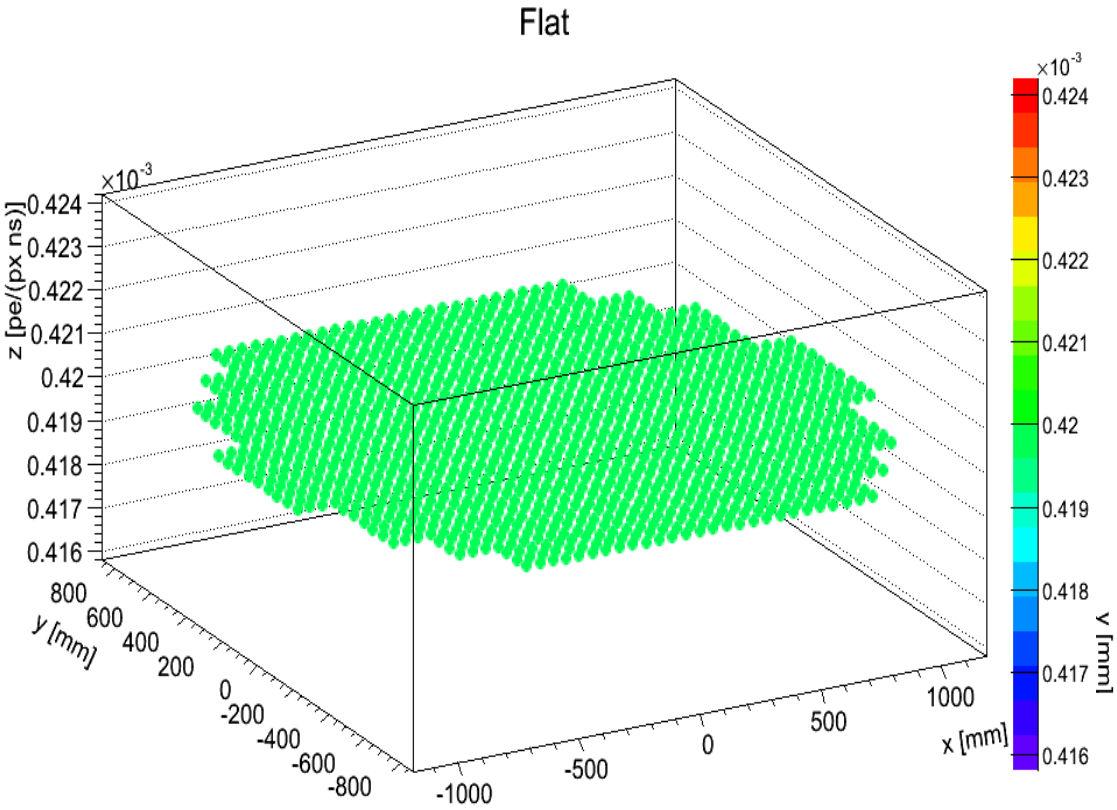
These results were used to produce background for each EC – background for simulation stored in file config/Electronics/EusoElectronics/NightGlow_RayTraceBG.dat

Current status of code for BG in shower simulations

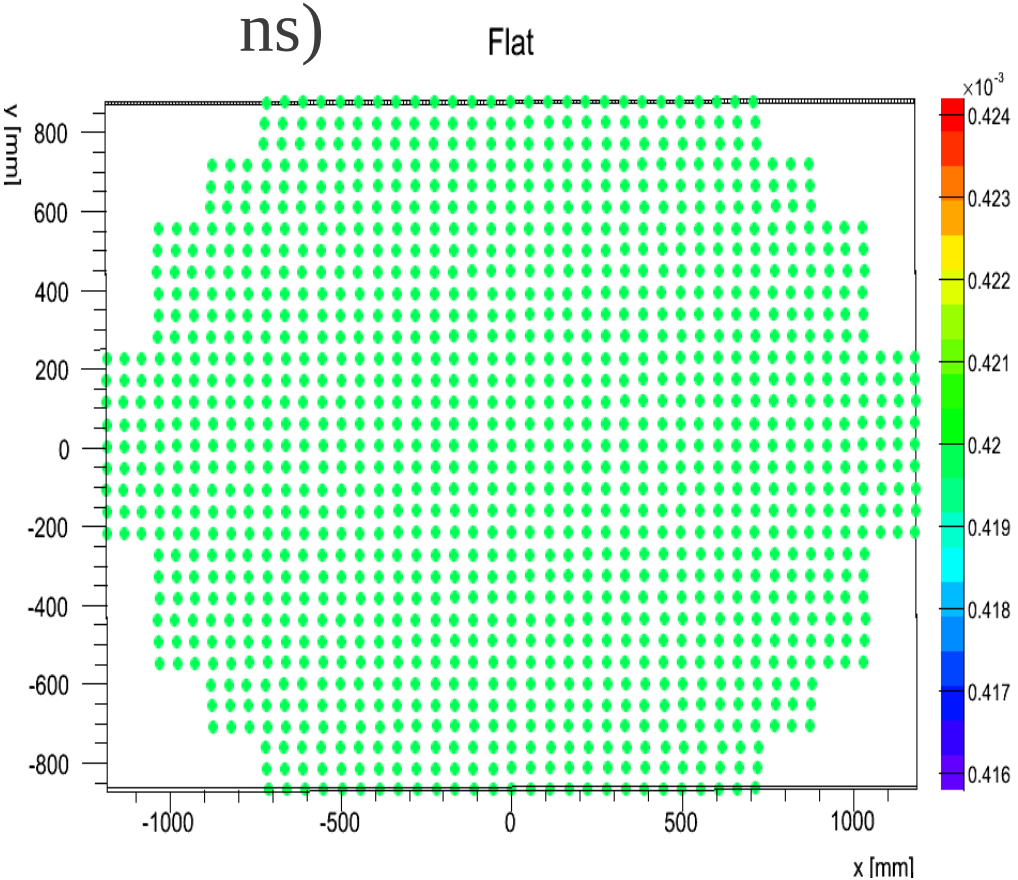
- Class for background calculations: EusoElectronics
- Two main options for Night Glow background:
  - byRate
  - byRadiance (not used in this study)
- byRate – options for shape:
  - Flat – 0.42 pe/(px μs) for each elementary cell
  - CosTheta – cos(θ) dependence of rate, (0.42 pe/(px μs) refers to theta 0)
  - New switch: RayTraceBG – load value from ray trace simulation for each EC from data file. (config/Electronics/EusoElectronics/NightGlow_RayTraceBG.dat)

- Configuration for this study:
  - temporal_complutensian_MAR2012_PPP2010_ammend_v1.cfg
  - Wavelength range in config file 250 – 485 nm in BG study (previous slide) 250 – 500 nm
  - Difference in background between these two case is only 0.4%

Flat

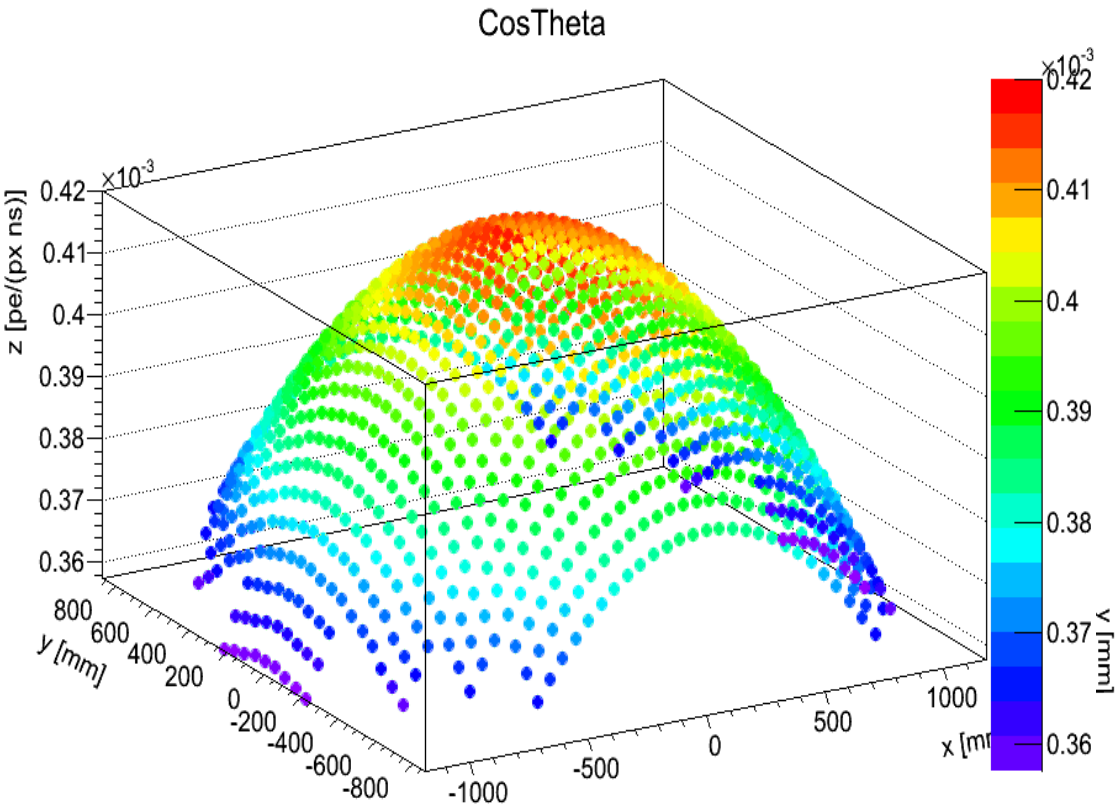


- x,y axis - position of center of EC in mm
- z axis (also color palette) show values of background in pe/(px ns)



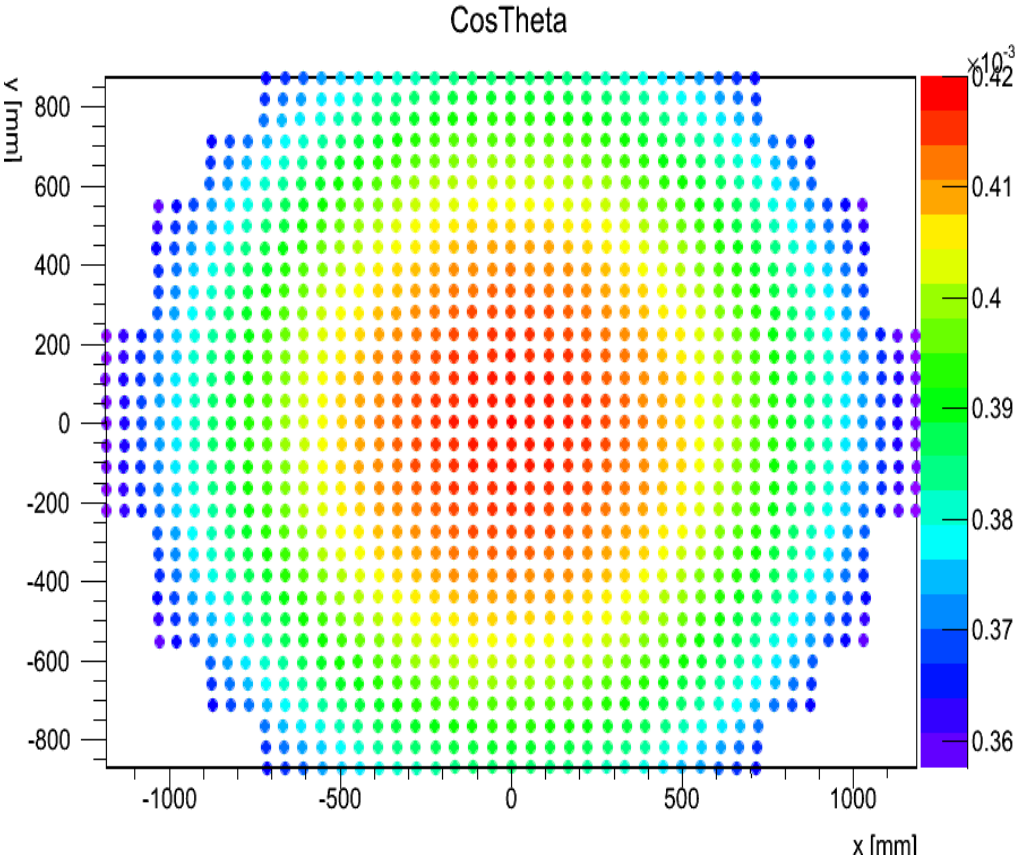
- 0.42 pe/(px μs) for each elementary cell

CosTheta

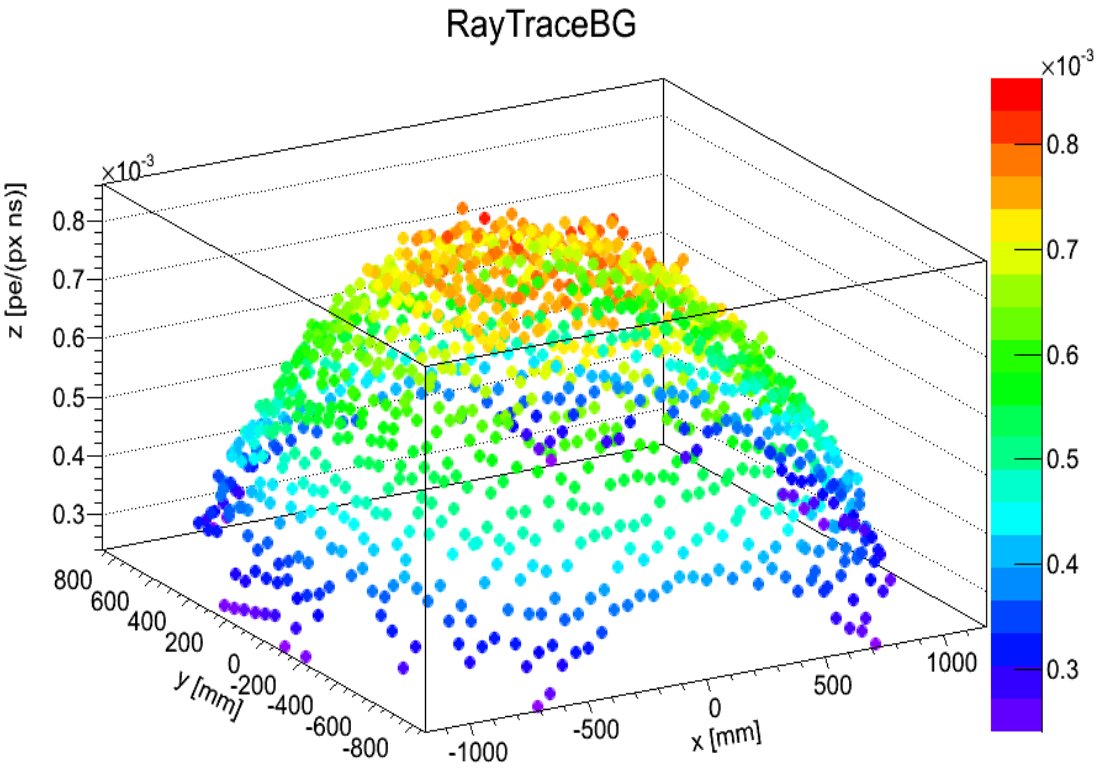


- Range of values ~ 0.36 - 0.42 pe/(px µs)

- CosTheta dependence of rate
- 0.42 pe/(px µs) corresponding to theta 0

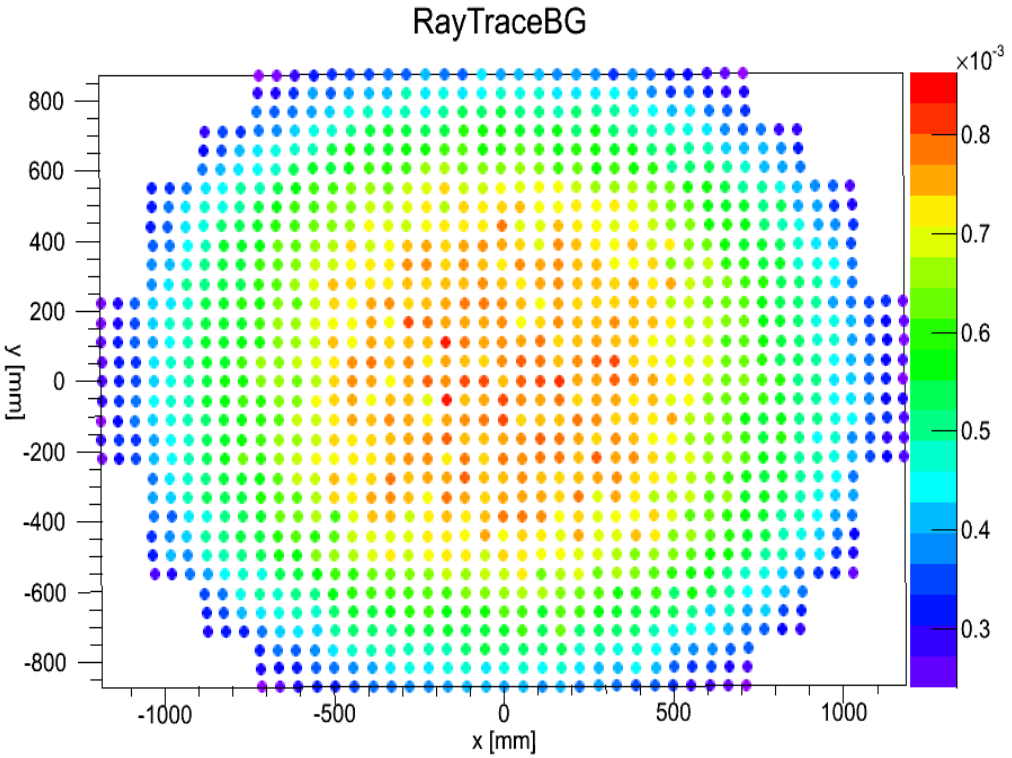


RayTraceBG

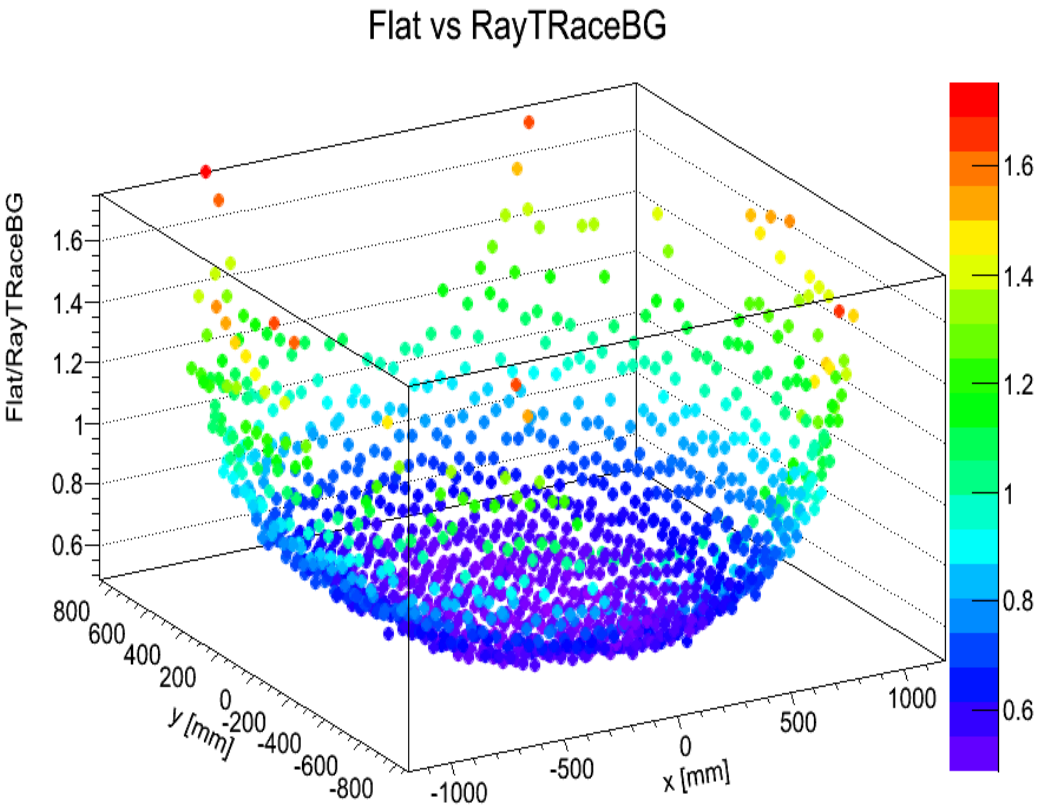


- For more details about ray trace simulations ask Pavol Bobik

- Range of values ~ 0.25 - 0.8 pe/(px µs)
- Average value: ~ 0.582201 pe/(px µs)

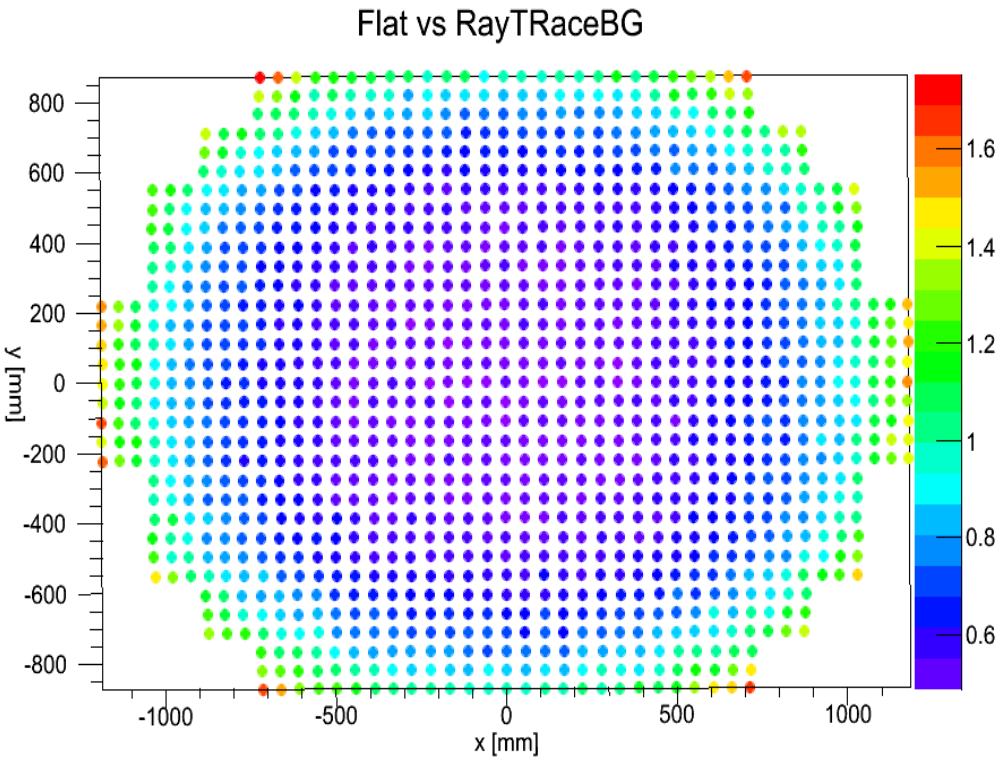


Flat vs RayTraceBG



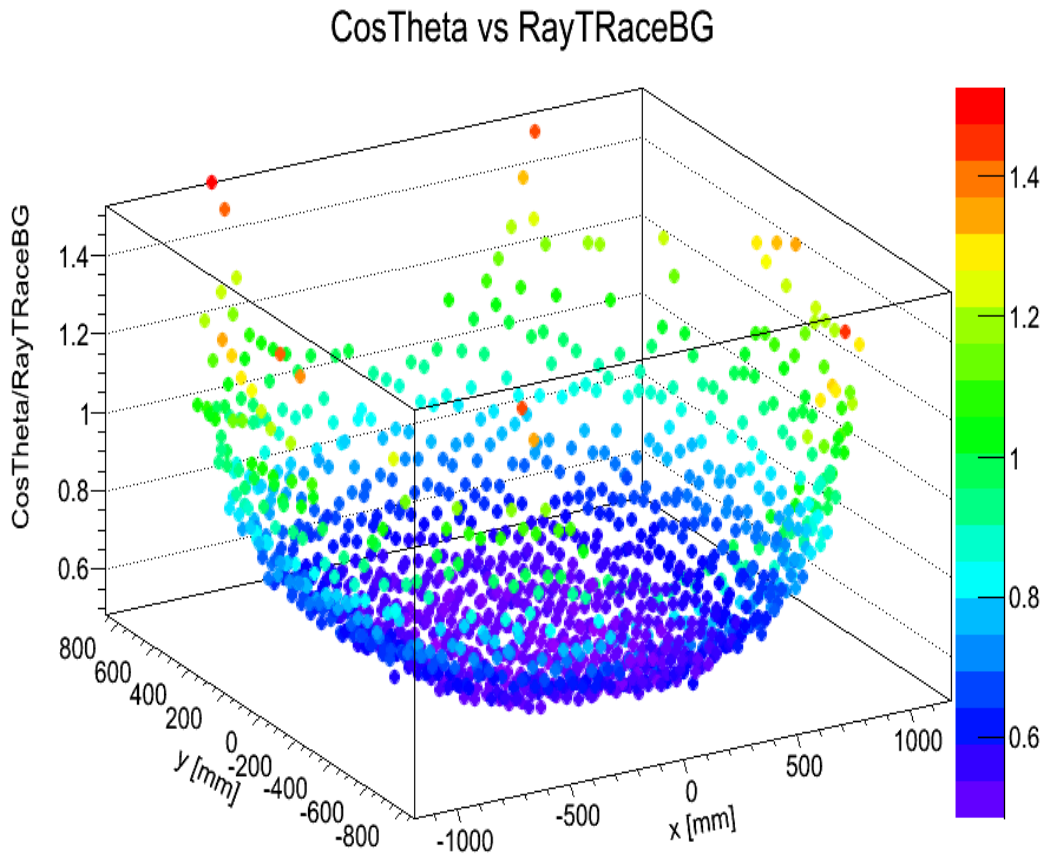
- z axis (also color palette)

  Flat/RayTraceBG

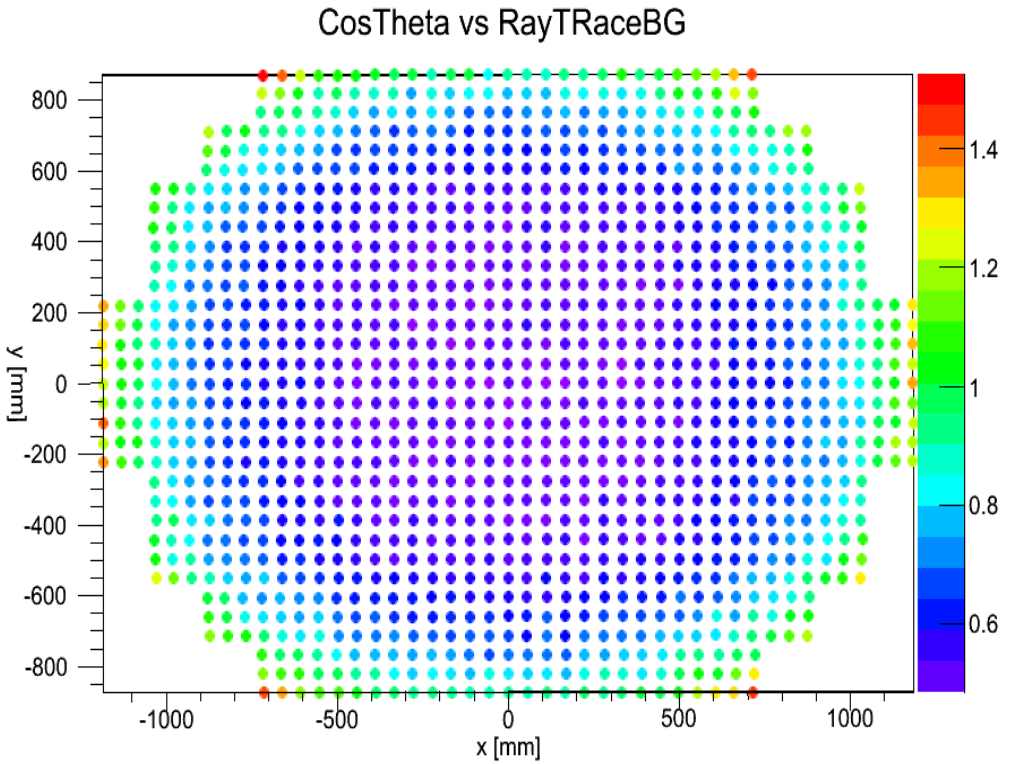


- Ratio:
  - In center ~ 0.55
  - On the edge ~ 1.6

CosTheta vs RayTraceBG



- z axis (also color palette) CosTheta/RayTraceBG



- Ratio:
  - In center ~ 0.5
  - On the edge ~ 1.4

ERayTraceReader.cc

```
#include <iostream>
#include "ERayTraceReader.hh"
#include <fstream>

using namespace std;
ClassImp(ERayTraceReader)

  ERayTraceReader::ERayTraceReader() //constructor
  {
                x_mm = new Double_t [1233];
                y_mm = new Double_t [1233];
                value_xy_mm = new Double_t [1233];

        for (Int_t j=0;j<1233;j++){
                x_mm[j]=0;
                y_mm[j]=0;
                value_xy_mm[j]=0;  }

  }

  ERayTraceReader::~ERayTraceReader() //destructor
  {
        delete [] x_mm;
        delete [] y_mm;
        delete []  value_xy_mm;
  }

  void ERayTraceReader::LoadFromFile_mm(const Char_t *lnfe)
  {
    Int_t i=0;
    Double_t x,y,value=0;

    ifstream myfile1;
    myfile1.open(lnfe); // open file to read data

          while(!myfile1.eof()) {

                myfile1 >>x;
                myfile1 >>y;
                myfile1 >>value;
                x_mm[i]=x;
                y_mm[i]=y;
                value_xy_mm[i]=value;
                i=i+1;
                if (i==1233) break;
                  }
  myfile1.close();
  }
```

ERayTraceReader.hh

```
#ifndef _ERAYTRACEREADER_H_
#define _ERAYTRACEREADER_H_

#include "TObject.h"

  class ERayTraceReader
  {

  public:

         ERayTraceReader();
         virtual  ~ERayTraceReader();
         void LoadFromFile_mm(const Char_t * lnfe);
         virtual Double_t GetX(Int_t j) { return fx=x_mm[j]; }
         virtual Double_t GetY(Int_t j) { return fy=y_mm[j]; }
         virtual Double_t GetValueXY(Int_t j) { return fv=value_xy_mm[j]; }

  private:
          Double_t *x_mm;
          Double_t *y_mm;
          Double_t *value_xy_mm;
          Double_t fx;
          Double_t fy;
          Double_t fv;

    ClassDef(ERayTraceReader,1)
  };


#endif
```

EusoElectronics::Bulid

```
    if (fNightGlowShape =="RayTraceBG"){

         fGtu=0;
         string path_RayTraceBG = Conf()->GetCfgDir()+'/'+
             ClassType()+'/'+
             ClassName()+'/'+"NightGlow_"+fNightGlowShape+".dat";
             const Char_t *path_RayTraceBG_Ch;
             path_RayTraceBG_Ch=path_RayTraceBG.c_str();

         fNightGlowrayTraceBG = new ERayTraceReader();
         fNightGlowrayTraceBG->LoadFromFile_mm(path_RayTraceBG_Ch);
         fGtu= Config::Get()->GetCF("Electronics","MacroCell")->GetNum("MacroCell.fGtuTimeLength");

                                    }
```

EusoElectronics::NightGlowRate

```
//_____________________________________________________________________________
Double_t EusoElectronics::NightGlowRate( const TVector3& pos, const TVector3& norm,
         Double_t pxsize, Double_t pde ) const {
    //
    // Nightglow rate as function of the location on the FS
    // (number of hits per microseconds)

    Double_t ngr(0),x(0),y(0),xx(0),yy(0);

    if ( fNightGlow == "byRate" ) {
        if ( fNightGlowShape == "Flat" )
            ngr = fNightGlowRateOnAxis;
        else if ( fNightGlowShape == "CosTheta" ) {
            Double_t fsPosZ = 2200*mm;

            Double_t tg2th = pos.Perp2();//(pos.x()*pos.x() + pos.y()*pos.y());
            tg2th /= ( (fsPosZ+pos.z()) * (fsPosZ+pos.z()));
            Double_t cth = 1. / TMath::Sqrt((1. + tg2th));
            ngr = fNightGlowRateOnAxis*cth;

        }else if (fNightGlowShape == "RayTraceBG")  {
                x=pos.x();
                y=pos.y();

        for (Int_t i=0; i<1233;i++){
                xx=fNightGlowrayTraceBG->GetX(i);
                yy=fNightGlowrayTraceBG->GetY(i);

            if ( (x<xx+0.1) && (x>xx-0.1) && (y < yy+0.1) && (y>yy-0.1) ) {
                ngr=(fNightGlowrayTraceBG->GetValueXY(i))*(pxsize*pxsize/fGtu); //conversion from pe/(mm2 GTU) on pe/(px ns), GTU is in [ns]
                                                                                }
                        }
        }
         else
            FatalError("Unknown night glow shape:"+fNightGlowShape);
    } else if ( fNightGlow == "byRadiance" ) {
        Double_t r = pos.Perp();

        if ( r > fNightGlowDist->GetXmax() ) {
            ngr = 0;
        } else {
            ngr = fNightGlowDist->GetValue( r )*fNightGlowRadiance;

            // apply quantum efficiency and pmt orientation
            ngr *= pxsize*pxsize*Abs(norm.CosTheta());
            ngr *= pde;
            ngr *= fDetectorScaleFactor*fDetectorScaleFactor;

        }

    }
    return ngr;
```

Conclusion

- Raytrace background have more sharp shape than CosTheta background.
- RayTrace average value ~ 0.582201 pe/(px μs), 1.455 pe/(px GTU)

- Flat/RayTraceBG: In center ~ 0.55, On the edge ~ 1.6
- CosTheta/RayTraceBG: In center ~ 0.5, On the edge ~ 1.4

Backup

Different input files for optics – difference is ~ 15 %

- telparm_PPP_2010_08_NOptics_v2.dat

- In unit of [pe/(px GTU)]
- average - 1.2578
- Max. PDM - 1.7088
- Min. PDM - 0.7005

- In unit of [pe/(px μs)]
- Average - 0.503
- Max. PDM - 0.684
- Min. PDM - 0.28

- telparm_PPP_2010_08a_NOptics.dat

- In unit of [pe/(px GTU)]
- average - 1.42
- Max. PDM - 2.02
- Min. PDM - 0.73

- In unit of [pe/(px μs)]
- Average - 0.586
- Max. PDM - 0.808
- Min. PDM - 0.292