SKLI Warwick Optics Library (v1.0, analyser)

This is a library of ROOT objects which correspond to test-stand results of optics developed and produced by the University of Warwick as part of a light injection system for optical calibration purposes in Super-Kamiokande.

You are receiving the file `skli_warwick_opticlib_analyser_vX.Y.root` to aid your studies of the performance of the SK light injection system. Hence, you are being provided with high-order polynomial fits (`TF1`s) to the angular distributions of each collimator and diffuser installed in SK (denoted `$injector_$optic_theta_air`).

In case of any issues or queries, your first port of call is **Billy Vinning** (B.Vinning@warwick.ac.uk), who assembled this library. Past mid-2021, he will hopefully be gone, so at that point you are better off contacting **Steve Boyd** (S.B.Boyd@warwick.ac.uk) instead.

Please do not distribute this data to collaboration non-members.

Getting Started

The scans are organised by `TDirectory` in the `TFile` like `$injector/$optic` (e.g. `b1/collimator` or `b3/diffuser`).

It is instructive to explore the library in the ROOT `TBrowser` by executing:

```
root -l $opticlib_fname -e 'TBrowser B'
```

in terminal.

For Python users, to recursively list the keys of all objects available in the library (with `uproot`), you can do:

```
import uproot
file = uproot.open('$opticlib_fname')
print(file.allkeys())
```

The reading out of `TF1`s requires a working `pyroot` installation. To get any object in the library, do:

```
import ROOT
file = ROOT.TFile($opticlib_fname)
obj = file.Get('$injector_position/$diffuser_type/$obj_name')
```

Details

Before you get started, please read the following sections about what these distributions are actually portraying to get an idea of how to use these objects for generating MC samples.

Diffusers

The diffuser test-stand captures the angular distribution of light output of a diffuser by rotating one in place from [$-\theta$, θ] at some distance away from a small PMT fixed in place. In spherical coordinates, these distributions give the polar angle distribution in degrees for some azimuthal angles ϕ (+ve (or -ve) θ values) and $\phi - \pi$ (-ve (or +ve) θ values), where $\hat{z}$ is aligned along the pointing direction of the diffuser. It is likely you will want to average over the distribution after taking absolute values of θ (imagine folding the plot on θ = 0), or you could just be lazy and throw away one half of the distribution. The y-axis scale is given by the average of the integrated area under all pulses recorded by the PMT and is left unnormalised.

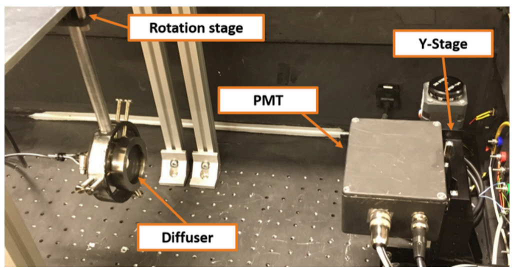

Figure 1: The diffuser test-stand.

Collimators

The collimator test-stand captures the beam cross-section of a collimator by moving a CMOS camera along the beam direction. The angular distributions provided give the polar angle distribution in degrees of light intensity, relative to the virtual origin of the light cone, as averaged over all orientations of ϕ. The distribution is area normalised to 1.

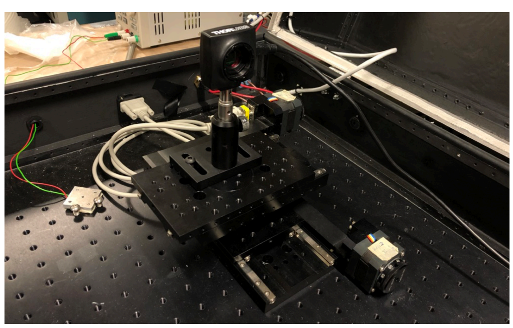

Figure 2: The collimator test-stand.

Generating MC samples

The distributions provided give the light intensity *measured* at a particular angle, you cannot use these unprocessed as a PDF for generating a sample of θ directions for photons in a fake data sample. If you were to do this, the θ positional distribution of your projected photons would not at all match the test-stand data (there would be far too many photons in the core of the beam). The reason for this is trigonometry. The solution is to multiply this `TF1` by $\sin\theta$ (or apply this as a weight to each bin if you have converted to a `TH1`. Remember also that the test-stand data is recorded in air, if you are simulating in water the transformation $\theta_{water} = \theta_{air}/n_{water}$ needs to be applied to account for refraction.

The orientation in ϕ of all optics in their test-stands and in their installation points in the tank are not recorded, therefore you can only assume a uniform distribution from $-\pi$ to π for photon directions in ϕ.

Random samples are easily generated with ROOT `TF1s`, in Python you can do:

```
sample = [func.GetRandom() for i in range(int(n_photons))]
```

and `TF1` will be used as a PDF to generate a random sample with the default ROOT seed.

Example MC hitmaps

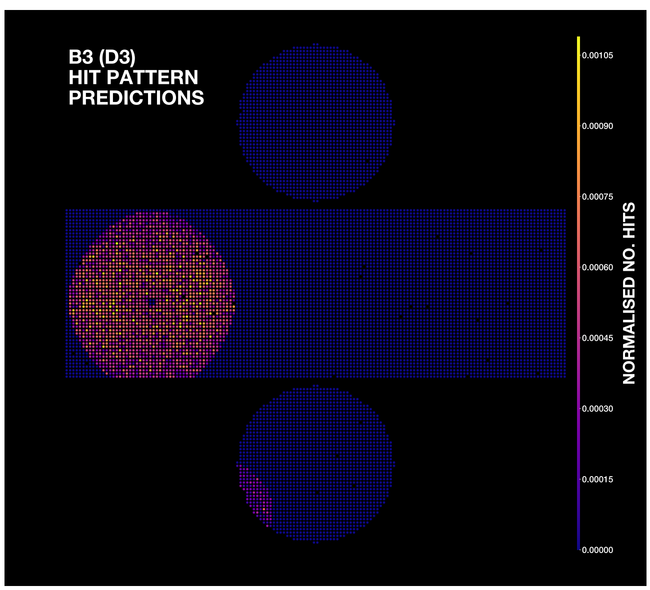

Figure 3: MC prediction of B3 diffuser.

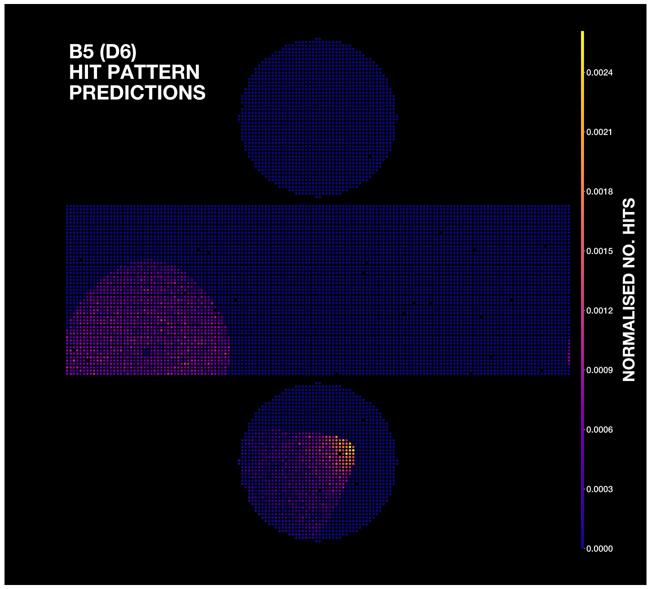

Figure 4: MC prediction of B5 diffuser.

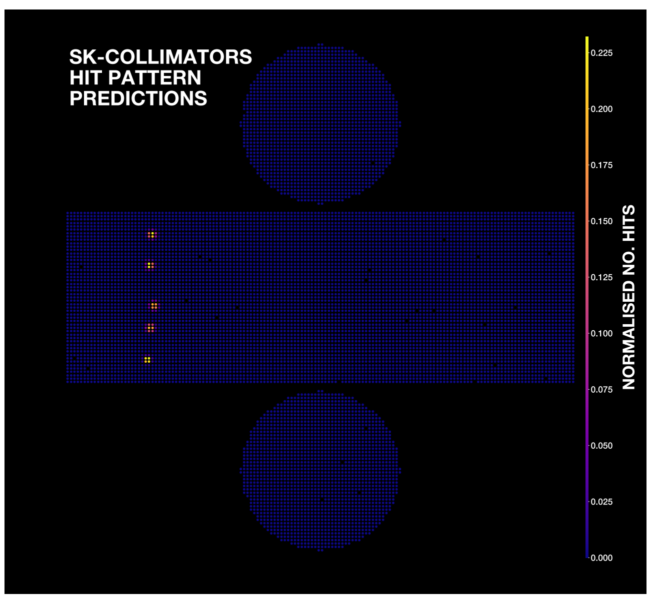

Figure 5: MC predictions of B1-B5 collimators.