Notes on the implementation of likelihood weighting in the ScienceTools.

E. Charles[1], J. Ballet[2],

ABSTRACT

This is a collection of notes and equations about the likelihood weighting implementation.

1. Introduction

This implementation and equations are based on Jean Ballet's work.[1]

2. The effective background: B

The first step in computing the weights is to derive the "effective background" for every pixel and energy bin. This is essentially the contribution of the background to the analysis of a point source at a particular energy bin.

We derive the effective background starting from some representation of the counts in the region of interest (ROI). This can be either binned data, or a model of the ROI. Following notation we used elsewhere[2] let's call this M_{ik}, where the i index runs over the pixels in the model, and the k index runs over the energy bins. The energy bin edges are at E_k^-, E_k^+ (typically $E_k^+ = E_{k+1}^-$). The geometric energy bin centers are $E_k = \sqrt{E_k^+ E_k^-}$. The energy bin widths are $\delta E_k = E_k^+ - E_k^-$, and the pixel sizes are $\delta\Omega_i$.

In words, we want to define the effective background B_{ik} by convolving M_{ik} with the point-spread function (PSF) at each energy P_k and then sum the result over all energies greater than or equal to a particular energy.

$$B_{ik} = \sum_{j \geq k} \frac{M_{ij} \bigotimes P_j}{P_j^{\max}}, \tag{1}$$

[1]W. W. Hansen Experimental Physics Laboratory, Kavli Institute for Particle Astrophysics and Cosmology, Department of Physics and SLAC National Accelerator Laboratory, Stanford University, Stanford, CA 94305, USA

[2]Laboratoire AIM, CEA-IRFU/CNRS/Université Paris Diderot, Service d'Astrophysique, CEA Saclay, F-91191 Gif sur Yvette, France

[1]Extensive notes can be found at: `https://confluence.slac.stanford.edu/x/ZphdCw`

[2]Specifically, in the write up of the binned likelihood implementation.

where $P_j^{\rm max}$ is the maximum value of the PSF at energy j. Another way of expressing this is that it is the PSF-weighted integral over the background map.

The convolution routines in the ScienceTools work on objects of type `ProjMap` (actually, the subclasses `HealpixProjMap` and `WcsMap2`), which are differential quantities (i.e., they are intensities, defined at specific energy / directions, rather than begin integrated across a range of energies and over a pixel. In practical terms, this just means that we have to convert the model counts from either `CountsMap` or `CountsMapHealpix` to a `WcsMap2` or `HealpixProjMap`, we do this just by dividing the bin contents by the energy bin widths and pixels sizes.

$$I_{ik} = \frac{M_{ik}}{\delta\Omega_i \delta E_k}. \tag{2}$$

What we actually get back from the PSF convolution routine is the normalized convolution:

$$\tilde{I}_{ik} = I_{ik} \bigotimes P_k. \tag{3}$$

To get the effective background, we have to convert that quantity back to counts and sum of all the energy bins greater than or equal to a particular energy.

$$B_{ik} = \sum_j^{j \geq k} \frac{\tilde{I}_{ij} \delta E_j}{P_j^{\rm max}}. \tag{4}$$

Although the factors of δE_k in Eq. 2 and δE_j in Eq. 4 cancel out, we have explicitly kept them in the notation, as they appear in the code because of the way it is structured.

The quantity B_{ik} has units of counts, and is essentially a counts map. We store it as such. It can be produced by the standalone application `gteffbkg` or using the `pyLikelihood` interface. The resulting file will look almost identical to a binned counts map file, including the `EBOUNDS` and `GTI` HDUs, and the DSS keywords are copied from the input file. The only difference will be the addition of keywords to the primary header of the output file:

1. `MAPTYPE` will be set to "BKG_EFF".
2. `INPUTMAP` which will give the name of the input binned counts map file.

3. The weighted sum over components: α

In order to properly deal with multiple analysis components we need to compute the weighted sum over the components. This quantity depends on the level of systematic error we are considering (ϵ), on the

individual B_{ikm} for each component (indexed by m), and on the minimum B_{ikm} for the various components for each pixel and energy bin: $\hat{B}_{ik}$.

We define this weighted sum as:

$$\alpha_{ik} = \frac{1 + \epsilon^2 \hat{B}_{ik}}{1 + \epsilon^2 \hat{B}_{ik} \sum_m (\frac{\hat{B}_{ik}}{B_{ikm}})^2}. \tag{5}$$

In the case that we are using a single component, then all of the $\alpha_{ik} \equiv 1$.

This quantity is dimensionless, and has the same binning as the input counts map. It can be produced by the standalone app `gtalphabkg` or using the `pyLikelihood` interface. This will store almost exactly the same information as a `CountsMap` or `CountsMapHealpix`. Since we are using several components, we remove the DSS keywords. However we do add some keywords specifying the provenance of the map:

1. `MAPTYPE` will be set to "ALPHA_BKG";
2. `EPSILON`: giving the value of ϵ used in the computation;
3. `BKGMAPXX`: will list the input effective background maps.

4. The likelihood weights: w

Given the effective background maps and the α_{ik}, the likelihood weights for a particular component are defined as:

$$w_{ikm} = \frac{\alpha_{ik}}{1 + \epsilon^2 B_{ikm}}. \tag{6}$$

This quantity is dimensionless, but has the same binning as the input counts map. It can be produced by the standalone app `gtwtsmap` or using the `pyLikelihood` interface. However, for "historical" reasons, the `BinnedLikelihood` object expects the weights to be given as an object of the class `ProjMap`. In the ScienceTools `ProjMap` objects usually represent intensity maps, but there is nothing enforcing this. In practical terms the only real difference for a weights file is that it is now defined at specific energies rather than over energy ranges. This means that we replace the `EBOUNDS` HDU with an `ENERGIES` HDU.

Aside from that, writing the w_{ik} will store almost exactly the same information as a `CountsMap` or `CountsMapHealpix`, including the DSS keywords copied from the file with the B_ik; the only differences being the additional keywords:

1. `MAPTYPE` will be set to "WEIGHT_MAP".

2. EPSILON: giving the value of ϵ used in the computation;

3. BKGMAP: will give the input effective background map.

4. ALPHAMAP: the file containing the input α map (if used).

5. Using the likelihood weights.

The idea is that using the likelihood weights is almost transparent. If a likelihood weights map is specified when the BinnedLikelihood object is being created, those weights will be used, otherwise no weights will be used.

Depending on the interface used, the likelihood weights can be specified in a number of ways.

1. By passing an object of type ProjMap (actually either a WcsMap2 or a HealpixProjMap) into the constructor of BinnedLikelihood. The map will be re-sampled, taking the values from the bin and pixel centers of the CountsMapBase and requesting the ProjMap value for those.

2. By passing a file name into the constructor of pyLikelihood.BinnedAnalysis. This points to any file that contains a valid ProjMap, and the file will be handled according to the rules above.

3. By specifying a file name for the hidden wmap parameter of gtlike or gtscrmaps. This points to any file that contains a valid ProjMap, and the file will be handled according to the rules above.