Mon. Not. R. Astron. Soc. **000**, 1–?? (2002) Printed 9 March 2015 (MN LaTeX style file v2.2)

HELP: XID+, The Probabilistic De-blender for Herschel SPIRE maps

P.D. Hurley,[1⋆] S. Oliver,[1] and other HELP team members
[1] *Astronomy Centre, Department of Physics and Astronomy, University of Sussex, Falmer, Brighton BN1 9QH, UK*

Released 2002 Xxxxx XX

ABSTRACT
The Herschel Extragalactic Legacy Project (HELP) will provide ancillary data from other wavelengths alongside the extra Software available at `https://github.com/pdh21/XID_plus/`.

Key words: galaxies: statistics – infrared: galaxies

1 INTRODUCTION

2 DATA

3 XID+ ALGORITHM

The basic goal of XID+ is to use the SPIRE maps to infer the likely SPIRE flux of sources we already know about. Bayesian inference is well suited to these requirements. It allows the use of prior information and provides a posterior distribution of the parameter(s) after taking into account the observed data.

As discussed previously, at a higher level, we also want to provide a framework to do science directly with the maps rather than with catalogues, which are basically a form of data compression.

We therefore adopt a Bayesian probabilistic modelling approach for our XID+ algorithm. It aims to:

- map out the posterior rather than the traditional point estimate, thereby providing a precise measure of uncertainty.
- Extend the use of prior information beyond just using positional information about sources.

In the following section, we describe our XID+ algorithm. As this algorithm is meant to build upon knowledge gained from the original XID (a.k.a DESPHOT) algorithm used by HerMES (**???**), we describe XID+ in the context of how it differs from DESPHOT.

3.1 Basic Model

Our data $\mathbf{d}$ are maps with $n_1 \times n_2 = M$ pixels. Our model assumes the map are formed from n known point sources, with an unknown flux density $\mathbf{f}$. The point response function (PRF) tells us the contribution each source makes to each pixel in the map. As the SPIRE maps are mean subtracted, our model requires a global background level (B) and some unknown noise term (δ). Our map can therefore be described as follows:

$$\mathbf{d} = \sum_{i=1}^{n} \mathbf{P_i} f_i + B + \delta \tag{1}$$

where $\mathbf{d}$ is the image, $\mathbf{P_i}$ is the PRF for source i, f_i is the flux density for source i, B is a global background estimate and δ is the noise term.

We can rewrite the above equation in the linear form:

$$\mathbf{d} = \mathbf{A}\mathbf{f} + \delta \tag{2}$$

Where A is a the pointing matrix, an $M \times n + 1$ matrix giving the contribution of sources and background to each pixel.

Our map $\mathbf{d}$ will have an associated, and measurable, variance and possibly covariance between the pixels, which we define here as $\mathbf{N_d} = \langle \delta\delta^T \rangle$. We define the likelihood as the Gaussian probability function for the data given the flux densities

$$L(\hat{\mathbf{f}}) = p(\mathbf{d}|\hat{\mathbf{f}}) \propto |\mathbf{N_d}|^{-1/2} \exp\big\{ -\frac{1}{2}(\mathbf{d}-\hat{\mathbf{d}})^T {\mathbf{N_d}}^{-1} (\mathbf{d}-\hat{\mathbf{d}})\big\} \tag{3}$$

where $\hat{\mathbf{d}} = \mathbf{A}\hat{\mathbf{f}}$. The maximum likelihood solution to this equation can be found by setting $\chi = (\mathbf{d}-\hat{\mathbf{d}})^T {\mathbf{N_d}}^{-1} (\mathbf{d}-\hat{\mathbf{d}})$, finding the minimum and rearranging such that:

$$\hat{\mathbf{f}} = (\mathbf{A^T N_d^{-1} A})^{-1} \mathbf{A^T N_d^{-1} A} \tag{4}$$

Equation 4 can be solved directly, either by brute-force matrix inversion or via other linear methods. As discussed in **???**), linear approaches ignore prior knowledge that fluxes cannot have negative flux density, which in very degenerative cases can result in any symmetric pairing of positive and negative flux providing a good fit. They are also incapable of discriminating between real and spurious sources, which can

⋆ Email: p.d.hurley@sussex.ac.uk

result in overfitting. To overcome these issues, **?**) used the non-negative weighted LASSO algorithm (Tibshirani 1996; Zou 2006; ter Braak et al 2010).

LASSO works by treating sources either 'inactive' and flux density set to zero, or 'active'. It switches sources on one at a time, with the order determined by reduction in chi-squared gained by turning them on. The process continues until some tolerance is reached.

In the first iteration, DESPHOT uses LASSO on each segment, to estimate the source fluxes. It then estimates a value for the background (B) via

$$B = \mathbf{d} - \sum_{i=1}^{n} \mathbf{P_i} f_i \tag{5}$$

The estimate from B is subtracted, and the LASSO fitting is rerun to get the final flux density estimates.

For XID+, we want to map out the entire posterior, $p(\mathbf{f}|\mathbf{d})$, rather than find the maximum likelihood solution. This has the benefit that it gives us more accurate information about how certain we are about the predicted fluxes. The posterior can be defined as:

$$p(\mathbf{f}|\mathbf{d}) \propto p(\mathbf{d}|\mathbf{f}) \times p(\mathbf{f}) \tag{6}$$

where $p(\mathbf{d}|\mathbf{f})$ is our likelihood, defined in equation 3 and $p(\mathbf{f})$ is our prior on the fluxes.

In our probabilistic framework, we can illustrate our model for the map, defined in equation 1 via a probabilistic graphical model (PGM). Figure 1 shows our PGM for our basic XID+ model, where boxes represent dimensions, open circles as random variables, dots as deterministic or (fixed) variables. For our simplest model, the sky co-ordinates of our sources are treated as fixed, as is the PRF. Both these fixed variables are used to make the pointing matrix A which gives the contribution each source makes to each pixel j in the map. Each source has its own flux f which is a random variable. By multiplying f, A and adding our global estimate for the background B, we can make our model for the map, M.

3.1.1 Stan

Now we have our probabilistic model, we need a Bayesian inference tool capable of sampling from it to obtain the posterior. We use the Bayesian inference tool, *Stan*, which is 'a probabilistic programming language implementing full Bayesian statistical inference with MCMC sampling'. *Stan* uses the adaptive Hamiltonian Monte Carlo (HMC) No-U-Turn Sampler (NUTS) of Hoffman Gelman (In press) to efficiently sample from the posterior. It does this by using the gradient, which allows fast traversing of high dimensional and highly correlated joint posterior distributions.

Stan has its own modelling language, in which one constructs probabilistic models. Our model for *Stan* can be found in Appendix A.

3.1.2 Estimating Convergence

As with all MCMC routines, one needs to be careful in running enough chains and they have been run long enough to find the global minimum.

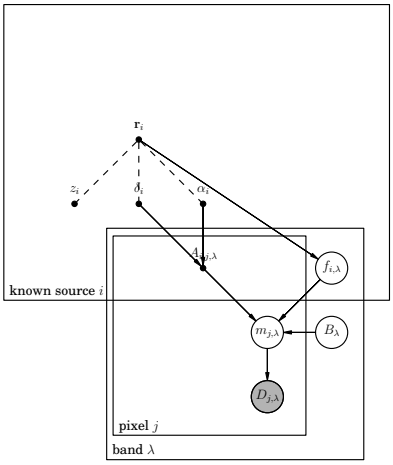

Figure 1. Our probabilistic model for XID+. Boxes represent dimensions, open circles as variables, dots as deterministic or (fixed) variables. Created with DAFT (`http://daft-pgm.org/`)

3.1.3 Effective sample size

3.2 Map segmentation

The survey fields in HELP vary in size from .... to ..... square degrees. Ideally, source photometry and background estimation would be done on full image, in practice it is often computationally infeasible. DESPHOT segmented the map by locating islands of high SNR pixels enclosed by low SNR pixels. The segmentation algorithm operates thus:

- Locates all pixels with a SNR above some threshold (default value of SNR= 1);
- Takes the first of these high SNR pixel starting in the bottom left corner of the image;
- 'Grows' a region around this pixel by iteratively taking neighbouring high SNR pixels;
- Once there are no more high SNR neighbours jumps to the next high SNR pixel and repeat from step (iii). Each of these independent regions of high SNR pixels is uniquely identified and processed separately by the source photometry component.

For XID+, we first investigated using the most optimum and transparent way of segmenting the map. This involves breaking the map into distinct tiles but fitting for sources within and beyond the tiling region. Using the fact that, conditional on the fluxes, the data from each tile is independent, the posterior from the different tiles can be combined to give an overall posterior for the whole map using the following formula:

$$P(\mathbf{F}|D_1,..D_n,D_A) = a\frac{\prod_{i=1}^{n} P(\mathbf{F}|D_n,D_A)}{P(\mathbf{F}|D_A)^{n-1}} \tag{7}$$

where n is number of tiles and $P(\mathbf{F}|D_A)$ is our prior on the flux, given some ancillary data. As the number of dimen-

sions is highly dimensional, the only feasible way of multiplying highly dimensional probability distribution functions is to model them as multivariate Gaussians. We investigated fitting the fluxes in both normal and log space in an attempt to make the multivariate Gaussian approximation appropriate. Unfortunately, neither succeeded and so we have to abandon the optimum scheme.

We have settled on a tiling scheme that has tiles that overlap in the map. This makes sure sources that lie near the edge of a tile, will be nearer the centre of the adjacent tile. As before, for each tile, we fit to sources both in and beyond. For our final posterior, for each source, we take the posterior form the tile that is optimum, i.e. the tile in which the source lies closest to the centre.

3.3 Uncertainties and Covariances

If DESPHOT is assumed to be linear[1], then one can get a lower limit on the noise from $(\mathbf{A^T N_d^{-1} A})^{-1}$. This estimate only includes instrumental noise and degeneracies between sources. An estimate of the remaining residual confusion noise is is calculated by taking the standard deviation of the residual map pixels σres and removing the average instrumental noise in these pixels in quadrature, $\sigma^2_{conf} = \sigma^2_{res} - \sigma^2_{pix}$, where σ_{pix} is calculated directly from the exposure time per pixel. The total noise σ_{tot} for a point source is then calculated from both the instrumental noise (and confusion noise from the known sources), $\sigma_i = \sqrt{\mathrm{diag}((\mathbf{A^T N_d^{-1} A})^{-1})}$, and confusion noise from the unknown sources in the residual map σ_{conf} via $\sigma^2_{tot} = \sigma^2_i + \sigma^2_{conf}$.

For XID+ we are sampling directly from the posterior which give the uncertainty of the flux given the data. This includes the uncertainty from instrumental noise and confusion (I think). Unlike XID, we are not solving $\mathbf{f} = (\mathbf{A^T N_d^{-1} A})^{-1} \mathbf{A^T N_d^{-1} d}$, we are solving equation 1 and so variations in pixel flux from sources not in the prior list, i.e. from confusion, will directly affect our flux estimates. This is advantageous as DESPHOT was well known to underestimate the uncertainties.

4 SIMULATIONS

In order to test and quantify XID+, we use simulated SPIRE maps, generated from the Lacey simulations. NEED MORE DETAIL

A mock r band input catalogue, similar to that expected from the GAMA survey, is generated by taking the mock catalogue and making a cut at an apparent magnitude of 19.8. We use this as our prior input catalogue for both XID+ and DESPHOT. In order to compare performance, we look at flux accuracy (i.e. $S_{obs} - S_{True}$), measured across all recovered sources. Figure 2 shows the flux accuracy for both XID+ and DESPHOT while Table **??** shows the mean flux accuracy for both methods.

It is also vital that our proposed method return reliable estimates of the flux density error, as for real applications we will not have knowledge of the true flux density of our sources. In Figure 3 we show the distribution of observed flux density error (i.e. $S_{obs} - S_{true}$), normalised by the error estimated by the photometric pipeline for the deep simulation.

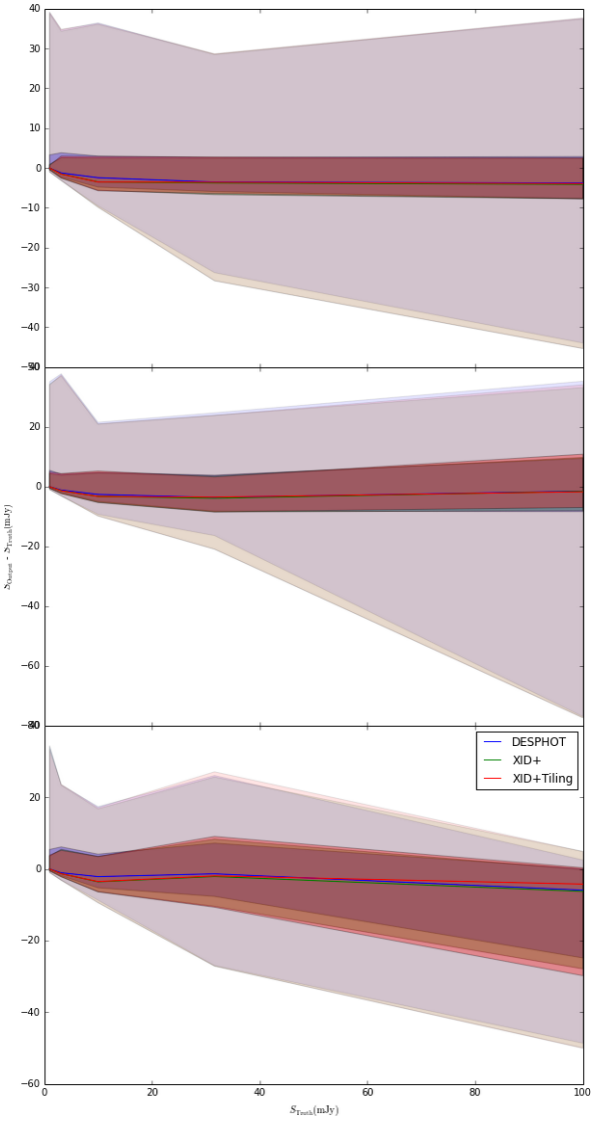

Figure 2. The Flux accuracy for DESPHOT, XID+ and XID+Tiling for the three SPIRE bands. The lines indicate median flux accuracy, the darker filled regions indicate the one sigma dispersion limit, while lighter filled regions indicate the three sigma limit of dispersion.

4.1 Performance with Priors

Adding prior information on fluxes Discuss how additional paper will focus on best way of providing prior information on fluxes.

[1] introducing LASSO and non-negative priors introduces a non-linearity

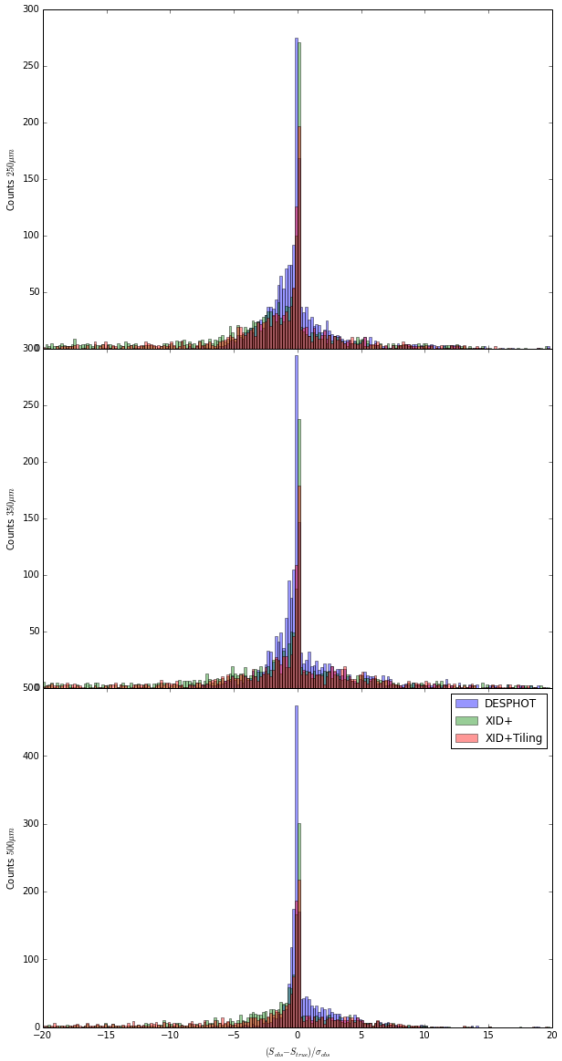

Figure 3. The Flux density error for DESPHOT, XID+ and XID+Tiling for the three SPIRE bands.

5 XID+SCIENCE

6 CONCLUSIONS

ACKNOWLEDGEMENTS

APPENDIX A

```
//Full Bayesian inference fit  XID
data {
  int<lower=0> nsrc;//number of sources
  //----PSW----
  int<lower=0> npix_psw;//number of pixels
  int<lower=0> nnz_psw; //number of non neg entries in A
  vector[npix_psw] db_psw;//flattened map
  vector[npix_psw] sigma_psw;//flattened uncertianty map (assuming no covariance between pixels)
  real bkg_prior_psw;//prior estimate of background
  real bkg_prior_sig_psw;//sigma of prior estimate of background
  vector[nnz_psw] Val_psw;//non neg values in image matrix
  int Row_psw[nnz_psw];//Rows of non neg valies in image matrix
  int Col_psw[nnz_psw];//Cols of non neg values in image matrix
  //----PMW----
  int<lower=0> npix_pmw;//number of pixels
  int<lower=0> nnz_pmw; //number of non neg entries in A
  vector[npix_pmw] db_pmw;//flattened map
  vector[npix_pmw] sigma_pmw;//flattened uncertianty map (assuming no covariance between pixels)
  real bkg_prior_pmw;//prior estimate of background
  real bkg_prior_sig_pmw;//sigma of prior estimate of background
  vector[nnz_pmw] Val_pmw;//non neg values in image matrix
  int Row_pmw[nnz_pmw];//Rows of non neg valies in image matrix
  int Col_pmw[nnz_pmw];//Cols of non neg values in image matrix
  //----PLW----
  int<lower=0> npix_plw;//number of pixels
  int<lower=0> nnz_plw; //number of non neg entries in A
  vector[npix_plw] db_plw;//flattened map
  vector[npix_plw] sigma_plw;//flattened uncertianty map (assuming no covariance between pixels)
  real bkg_prior_plw;//prior estimate of background
  real bkg_prior_sig_plw;//sigma of prior estimate of background
  vector[nnz_plw] Val_plw;//non neg values in image matrix
  int Row_plw[nnz_plw];//Rows of non neg valies in image matrix
  int Col_plw[nnz_plw];//Cols of non neg values in image matrix

}
parameters {
  vector<lower=0.0,upper=300> [nsrc] src_f_psw;//source vector
  real bkg_psw;//background
  vector<lower=0.0,upper=300> [nsrc] src_f_pmw;//source vector
  real bkg_pmw;//background
  vector<lower=0.0,upper=300> [nsrc] src_f_plw;//source vector
  real bkg_plw;//background

}

model {
  vector[npix_psw] db_hat_psw;//model of map
  vector[npix_pmw] db_hat_pmw;//model of map
  vector[npix_plw] db_hat_plw;//model of map

  vector[nsrc+1] f_vec_psw;//vector of source fluxes and background
  vector[nsrc+1] f_vec_pmw;//vector of source fluxes and background
  vector[nsrc+1] f_vec_plw;//vector of source fluxes and background



  bkg_psw ~normal(bkg_prior_psw,bkg_prior_sig_psw);
  bkg_pmw ~normal(bkg_prior_pmw,bkg_prior_sig_pmw);
  bkg_plw ~normal(bkg_prior_plw,bkg_prior_sig_plw);

  src_f_psw ~cauchy(0,10);
  src_f_pmw ~cauchy(0,10);
```

```
src_f_plw ~cauchy(0,10);


for (n in 1:nsrc) {
  f_vec_psw[n] <- src_f_psw[n];
  f_vec_pmw[n] <- src_f_pmw[n];
  f_vec_plw[n] <- src_f_plw[n];


}
f_vec_psw[nsrc+1] <-bkg_psw;
f_vec_pmw[nsrc+1] <-bkg_pmw;
f_vec_plw[nsrc+1] <-bkg_plw;


#src_f ~cauchy(0,10); // set cauchy distribution for fluxes i.e. expect lower

for (k in 1:npix_psw) {
  db_hat_psw[k] <- 0;
}
for (k in 1:nnz_psw) {
  db_hat_psw[Row_psw[k]+1] <- db_hat_psw[Row_psw[k]+1] + Val_psw[k]*f_vec_psw[Col_psw[k]+1];
    }

for (k in 1:npix_pmw) {
  db_hat_pmw[k] <- 0;
}
for (k in 1:nnz_pmw) {
  db_hat_pmw[Row_pmw[k]+1] <- db_hat_pmw[Row_pmw[k]+1] + Val_pmw[k]*f_vec_pmw[Col_pmw[k]+1];
    }

for (k in 1:npix_plw) {
  db_hat_plw[k] <- 0;
}
for (k in 1:nnz_plw) {
  db_hat_plw[Row_plw[k]+1] <- db_hat_plw[Row_plw[k]+1] + Val_plw[k]*f_vec_plw[Col_plw[k]+1];
    }


db_psw ~ normal(db_hat_psw,sigma_psw);
db_pmw ~ normal(db_hat_pmw,sigma_pmw);
db_plw ~ normal(db_hat_plw,sigma_plw);

  }
```