Creating a Non-Negative Matrix Factorization Deblender

January 27, 2017

1 Introduction

We begin with a set of images

$$A = \begin{pmatrix} A_g \\ A_r \\ A_i \\ A_z \\ A_y \end{pmatrix},$$

where A_λ is the flattened calibrated exposure using filter λ. This matrix can be decomposed into two matrices

$$A = WH,$$

with

$$\begin{aligned} W &= \begin{pmatrix} W_{g1} & \cdots & W_{gK} \\ W_{r1} & \cdots & W_{gK} \\ W_{i1} & \cdots & W_{iK} \\ W_{z1} & \cdots & W_{zK} \\ W_{y1} & \cdots & W_{yK} \end{pmatrix}, \\ H &= \begin{pmatrix} H_{11} & \cdots & H_{1N} \\ \vdots & & \vdots \\ H_{K1} & \cdots & H_{KN} \end{pmatrix} \end{aligned}$$

where W is a matrix with K columns, one for each source, such that each column is a normalized (to one) SED for each source; and H is matrix of intensities, where each row in H is the intensity of source k for each pixel (from 1 to N) in the images. This assumes that the images are calibrated such that they are astrometrically aligned and have been convolved to the same PSF, so that a single source in each image will have the same intensity profile, scaled by its SED in that band. The problem then becomes given a set of images A, can we find W, H?

The basic algorithm is outlined in Berry et. al 2006 but the main idea is that if our data is non-zero (which may require an inital projection which can later be undone) we can use a iterative multiplicative formula to update W, and H such that they are always non-negative. This involves minimizing the function

$$F = |A - WH|_F^2 + \alpha J_W(W) + \beta J_H(H),$$

where J_1 and $J_2$are constraints on W and H. With the appropriate choice of step functions, this becomes a gradient descent update algorithm such that

$$\begin{aligned} W_{ij} &\rightarrow W_{ij} \frac{AH^T}{(WHH^T)+\alpha\frac{\partial F}{\partial W_{ij}}} \\ H_{ij} &\rightarrow H_{ij} \frac{W^T A}{(W^T W)H_{ij}+\alpha\frac{\partial F}{\partial H_{ij}}} \end{aligned} .$$

Choosing the appropriate constraints, conditions for stopping, and initial conditions is problem dependent and part of our task is to find the best implementation for the deblender.

2 Constraints

2.1 Normalized SEDs

One might notice that W and H are not unique, as any invertible matrix D can be inserted

$$A = WDD^{-1}H$$

that solves the same problem. One easy way to break this degeneracy, and ensure that the SEDs are reasonable, is to requre each column to sum to zero.

2.2 Symmetry

Instead of using the rigid symmetry of the SDSS deblender, we use a weaker version that penalizes each row in H for not being symmetric but does not strictly forbid it.We define a linear operator $\Sigma^{(k)}$ that symmetrizes peak k about it's center, so each peak will have a different $\Sigma^{(k)}$ and we define

$$H_{k,sym} = \Sigma^{(k)} H_k.$$

$\Sigma^{(k)}$is a block marix with dimension N^2 of the form

$$\Sigma^{(k)} = \left(\begin{array}{c|c|c} 0 & 0 & 0 \\ \hline 0 & \sigma^{(k)} & 0 \\ \hline 0 & 0 & 0 \end{array}\right),$$

where $\sigma^{(k)}$ is an anti-diagonal matrix with 1's and 0's to appropriately symmetrize H. This allows us to add the constraint

$$\begin{aligned} J_H(H) &= |H - H_{sym}|^2 \\ &= |I - \Sigma|^2 H^2, \end{aligned}$$

so that

$$\begin{aligned}\frac{\partial J_H}{\partial H} &= |I - \Sigma|^2 H \\ &= (I - 2\Sigma + \Sigma^2)H,\end{aligned}$$

where Σ^2 is the indentity matrix with $\Sigma_{ii} = 0$ for pixels whose symmetric partner is outside of the footprint.

2.3 Background

One difficulty in using real background subtracted data is that the data is not non-negative, but contains negative values in the background. A quick fix is to project the data by subtracting the minimum pixel value, but removing the offset once W and H has been completed becomes tricky. One way to take care of this is to add an extra source with a constant background value set to

$$\begin{aligned}H_{bkg} &= offset * \lambda_{count}, \\ W_{bkg} &= 1/\lambda_{count}\end{aligned}$$

where λ_{count} is the number of filters used, which normalizes W_{bkg} to one. As the algorithm runs, we penalize pixels in $H_{k=bkg}$that deviate from a constant offset using the constraint

$$J_H(H_{bkg}) = (H_{bkg} - B)^3,$$

where B is a constant array set to the initial value of $H_{bkg} = offset * \lambda_{count}$ and we cube the term so that it's derivative is always positive.

2.4 PSF Sources

For sources that closely match the PSF, while it has not been implemented or tested yet, an algorithm similar to the one used to model the background can be used.