SPECIFICATION OF WALLABY/DINGO SOURCE FINDING FOR BETA PIPELINE

This document outlines the BETA pipline source finding specifications for the WALLABY and DINGO SSPs. Below we summarise our recommendations, before providing more detail on our motivation (Section 2), specifications (Sections 3 and 4) and implementation tests (Section 5).

1 Summary of recommendations

- The Duchamp 3D à trous wavelet reconstruction should be replaced with the 2D-1D wavelet transform in Flöer & Winkel (2011). (Duchamp change.)
- Duchamp should implement independent spectral and spatial smoothing over multiple scales. (Duchamp change.)
- A mask optimisation tool should be used to improve the masks of Duchamp detections. (Addition to pipeline.)

2 Motivation

The WALLABY and DINGO SSPs use a joint TWG to investigate and develop source finders and source parameterisation methods. The core of this TWG consists of the following people: Tobias Westmeier (chair), Lars Floer, Amr Hassan, Russell Jurek, Attila Popping, Paolo Serra and Matthew Whiting. This TWG has tested and developed a variety of source finders and source parameterisation methods, which is described in detail in various papers in the PASA source finding special issue[1]. We are currently in the process of combining various techniques into a single source finding and parameterisation tool. For example, we have successfully combined the CNHI and Duchamp source finders, and we're actively investigating how to add the S+C source finder to this combination.

We want to use a modified version of Duchamp and an under development source parameteriser for the BETA pipeline. Our assessment of various source finders (Popping et al., 2012) has shown that the performance of an unmodified Duchamp is comparable to the current implementations of other source finders. Until future implementations of other source finders (or a combination of them) outperforms Duchamp in all situations, we expect to continue using Duchamp. We have identified several weaknesses in Duchamp though, that can be fixed to improve its performance. These weaknesses exist

[1] A list of the PASA source finding special issue papers with links can be found here, http://www.atnf.csiro.au/research/WALLABY/PASA-Special-Issue/

in the manner that Duchamp carries out wavelet decomposition and smoothing of HI data cubes.

Duchamp uses wavelet decomposition of HI data cubes to de-noise them prior to source finding. Wavelet decomposition provides a powerful method of measuring the spatial and spectral scales of structure in a data cube. This allows for noise to be selectively removed from the data cube either by thresholding the wavelet coefficients produced by the wavelet decomposition, or removing the small spatial and spectral scales where uncorrelated, Gaussian noise is expected to be.

Duchamp currently allows the user to carry out a 3D "à trous" wavelet reconstruction of the data cube and search for sources in the reconstructed cube instead of the original data cube. The effectiveness of the method is currently limited by the fact that Duchamp does not treat spatial and spectral dimensions independently. Instead, the software will increase spatial and spectral wavelet scales simultaneously in each iteration of the reconstruction. This is suitable for sources that have the same extent in the spatial and spectral domain. In other words, that look like 'spheres' in the data cube. Many sources in large HI surveys like WALLABY and DINGO, however, will be elongated in one or two dimensions, warranting an independent treatment of spatial and spectral axes. An obvious example are the spatially unresolved galaxies with broad spectral lines that are typically encountered at higher redshift.

The situation is similar for spatial and spectral smoothing, which is another method of enhancing the signal-to-noise ratio of sources. Duchamp currently only allows either spatial or spectral smoothing at only one particular smoothing scale. However, in a real HI data cube containing galaxies at different redshifts we expect to find sources on various spatial and spectral scales, again demanding independent spatial and spectral smoothing on multiple scales to maximise the number of detections.

We have developed and tested two new algorithms, one for separate spatial and spectral wavelet transform on multiple scales and the other one for independent spatial and spectral smoothing on multiple scales. Extensive testing has demonstrated that both algorithms are more effective at detecting faint galaxies of different sizes and morphologies in HI data cubes than the wavelet reconstruction and smoothing options currently implemented in Duchamp. Therefore, we propose to implement the new algorithms in the Duchamp source finder, either replacing or complementing the currently implemented algorithms. The following sections of this document will introduce and discuss the new algorithms in detail.

3 Duchamp changes

3.1 Specification: 2D-1D wavelet transform

A 2D-1D wavelet transform has been developed for the EBHIS survey, and is described in Flöer & Winkel (2011). The mathematical framework for this wavelet transform is presented in Starck et al. (2009). The 2D-1D wavelet transform is essentially two interleaved à trous wavelet reconstructions. A C++ implementation of this method (based on the à trous wavelet transform) is attached to this memo. The simple algorithm describing the 2D-1D wavelet transform is presented below:

---

for all spatial scales of interest **do**
  Smooth the datacube array spatially.
  Calculate the spatial wavelet coefficients by subtracting the smoothed datacube from the original datacube.
  for all spectral scales of interest **do**
    Smooth the spatial wavelet coefficients.
    Subtract the smoothed spatial wavelet coefficients from the pre-smoothing spatial wavelet coefficients to determine the 2D-1D coefficients.
  end for
end for

The benefit of using this transform over the existing 3D wavelet transform is illustrated in Figure 1. Sources with similar spatial and spectral sizes (measured in voxels) look like spheres in the datacube, and are better matches for the Duchamp 3D wavelet kernel, which results in more significant wavelet coefficients and a higher completeness. This is offset somewhat by sources becoming sufficiently compact either spatially or spectrally at the edges of the plot to lift the source peak S/N above the threshold. The result is two 'valleys' in the left plot of Figure 1, that are populated by sources that are the least spherical in voxel space. In contrast, the relatively flat completeness of the 2D-1D wavelet transform on the right of Figure 1, shows that the 2D-1D wavelet transforms flexibility allows it to achieve a better match to the shape and size of sources. We have deliberately plotted the results of Duchamp with a much more aggressive (lower) threshold than was used for the 2D-1D wavelet transform. We did this to illustrate that this can't be fixed by using a more aggressive threshold. This effect is a fundamental flaw in the current Duchamp wavelet transform.

3.2 Independent spatial and spectral smoothing over multiple scales

We have found that a minimalist set of filters can be used to improve the performance of intensity thresholding source finders, such as Duchamp. The key is to use spatial and spectral filters that are zeroth order approximations of sources and are independently

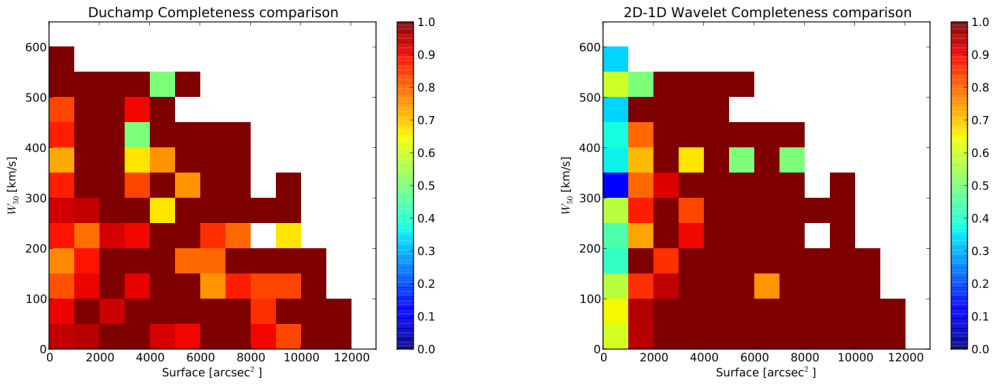

Figure 1: Completeness as a function of line width (w_{50}) and surface area for Duchamp with 3D "à trous" wavelet reconstruction (left) and the 2D–1D wavelet transform algorithm (right) based on source finding tests on model galaxies. Duchamp struggles to detect either extended sources with narrow lines or compact sources with broad lines, whereas the 2D–1D wavelet algorithm achieves 100% completeness over most of the parameter space shown. Please note that the low completeness of the 2D–1D wavelet algorithm for compact sources, irrespective of line width, is the result of using a much more conservative flux threshold in that particular test and does not imply that the algorithm is intrinsically worse than Duchamp at detecting such sources.

scaled. While this approach doesn't ensure the optimal S/N ratio for an arbitrary source, it does improve the S/N ratio sufficiently to improve source detection rates. An implementation of this approach is described in Serra et al. (2011) & Popping et al. (2012). These implementations use a top hat filter in the spectral dimension and a 2-D gaussian filter over both spatial dimensions.

The pseudo-code for this implementation is presented below. This implementation requires the following inputs, whose symbols (as used in the algorithm) are in parenthesis:

1. The input datacube (f).
2. The size of the datacube array dimensions (N_x, N_y, N_z).
3. An integer array to store the output mask (M).
4. The threshold for very bright voxels (T_0). Voxels above this threshold are automatically added to the output mask.
5. The threshold applied to the datacube after smoothing (T_1).
6. An array containing the spatial filters (k_i).
7. An array containing the spectral filtes (k_j).

8. The sizes of the spatial and spectral filter arrays (N_i, N_j).

This algorithm uses the dummy integer values (x,y,z) to traverse the f and M arrays. It also uses the dummy integer values i and j to index the spatial and spectral filters, respectively. The subscript ij is therefore used to indicate quantities corresponding to the cube smoothed with the i-th sky kernel and j-th frequency kernel. We use $i = 0$, $j = 0$ to indicate the original cube, f_{ij} to indicate a smoothed cube and k_{ij} to denote the kernel formed by a combination of spatial and spectral filter.

Explanatory comments are in parenthesis { . . . }.

{Create a temporary array that is used to store the results of intensity thresholding the datacube after smoothing with an arbitrary kernel, k_{ij}.}
Create the temporary mask array, M'.
{Create a temporary working array, f', that is used to store the various smoothed versions of the input datacube.}
Create the temporary array, f'.
Measure noise in the original cube, σ_{00}.
{Add very bright voxels to the output mask and clip these voxel values.}
for all $(x = 0;\ x < N_x;\ x++)$,$(y = 0;\ y < N_y;\ y++)$,$(z = 0;\ z < N_z;\ z++)$ **do**
  if $f_{00}(x, y, z) \geq T_0 \times \sigma_{00}$ **then**
    $M(x, y, z) \leftarrow 1.$
    $f_{00}(x, y, z) \leftarrow T_0 \times \sigma_{00}.$
  end if
end for
{For each kernel, k_{ij}, smooth the datacube, find the source voxels and add them to the output mask.}
for $i = 0;\ i < N_i;\ i++$ **do**
  if $i > 0$ **then**
    {Re-initialise the temporary working array, f', and spatially smooth.}Smooth data cube f'_{00} with i-th spatial filter: $f'_{i0} \leftarrow f_{00} * k_{i0}$.
  end if
  for $j = 0;\ j < N_j;\ j++$ **do**
    Initialise temporary mask, M' to zeroes.
    if $j > 0$ **then**
      Smooth data cube with j-th spectral filter: $f'_{ij} \leftarrow f'_{i0} * k_{0j}$.
    end if
    Measure noise level of smoothed datacube, σ_{ij}.
    {Generate the temporary mask, M', for this smoothing kernel, k_{ij}.}
    for all $(x = 0;\ x < N_x;\ x++)$,$(y = 0;\ y < N_y;\ y++)$,$(z = 0;\ z < N_z;\ z++)$ **do**
      if $f'_{ij}(x, y, z) \geq T_1 \times \sigma_{ij}$ **then**
        $M'(x, y, z) \leftarrow 1.$
      end if
    end for

{Apply a soft size filtering to the temporary mask, M', using a morphological opening with the current smoothing kernel, k_{ij}.}
for all $(x = 0; x < N_x; x++)$,$(y = 0; y < N_y; y++)$,$(z = 0; z < N_z; z++)$ **do**
$M'(x, y, z) \leftarrow M'(x, y, z) \circ k_{ij}$.
end for
{Update the output mask, M, with the temporary mask, M'.}
for all $(x = 0; x < N_x; x++)$,$(y = 0; y < N_y; y++)$,$(z = 0; z < N_z; z++)$ **do**
if $M'(x, y, z) = 1$ **then**
$M(x, y, z) \leftarrow 1$.
end if
end for
end for
end for
{Free the temporary arrays used as working spaces.}
Free M' array.
Free f' array.

---

4 Additional pipeline components

4.1 Specification: mask optimiser

We have identified a promising method for optimising the masks of spatially unresolved and marginally resolved detections. This method assumes that spatially unresolved and marginally resolved detections can be approximated as an elliptical cylinder in 3-D. We assume that this cylinder extends over $\pm W_{50}$, because Duchamp has been shown (Westmeier et al., 2011) to return accurate W_{50} measurements. The shape of the cylinders elliptic face is determined by fitting an ellipse to the moment-0 map. The size of this ellipse is determined by iteratively expanding until convergence is achieved. In this algorithm convergence is achieved when the total flux (sum of voxels within elliptical cylinder) divided by the square root of the number of voxels reaches its first maximum.

It should be noted that this method is still undergoing testing and development. We are requesting implementation at this stage so that a functional prototype will exist in the ASKAPsoft BETA pipeline. We expect this to facilitate easy, rapid updating in the future as we refine this method. As this method is still being refined, we would not be comfortable with other SSPs making use of this tool without first consulting us. This will ensure that they are aware of the results of future tests and impact of future refinements.

Attached to this document is an in-depth discussion and specification of the mask optimiser.

5 Testing the implementations

We have created three small, simple datasets containing a variety of sources. We intend to use these datasets and the C++ source finding framework that we have developed, to test the ASKAPsoft implementations in the following way:

1. The ASKAPsoft 2D-1D wavelet transform will be applied to each test dataset with a range of settings. Each output datacube will then be divided by the corresponding output datacube produced by Lars' existing implementation. If the ratio of these two outputs deviates from one at any point, then something has gone wrong. If something has gone wrong, then the 2D-1D wavelet coefficients produced by both implementations can be compared. This will reveal if their is an issue with original 2D-1D transform, thresholding of 2D-1D wavelet coefficients or the inverse 2D-1D transform.

2. Duchamp can be configured to produce a global output mask of the detections. This global mask can be fed into our C++ source finding framework to generate catalogues and sparse representations of the individual source masks. Using one of the tools in this source finding framework, we can compare the individual source masks produced by the ASKAPsoft implementation of intensity thresholding with independent spatial and spectral smoothing over multiple scales, with those produced by our existing implementation. This will allow us to check that for each test dataset and choice of input parameters, the same voxels are being flagged as source voxels by both implementations. In the event of a discrepancy, analogous to testing the ASKAPsoft 2D-1D wavelet transform, we can compare the various smoothed versions of the input dataset that are produced by both implementations.

3. We can test the mask optimiser in a simple way by checking that the ASKAPsoft implementation returns the same total fluxes as our implementation, when given the same inputs. In the event of a discrepancy, we can first check that each implementation starts off with the same seed shapes. If these are the same but the fluxes are different, then we can use our source finding framework to determine the differences in the optimised masks at the individual voxel level.

References

Flöer L., Winkel B., 2011, ArXiv e-prints (astro-ph/1112.3807)

Popping A., Jurek R., Westmeier T., Serra P., Floer L., Meyer M., Koribalski B., 2012, ArXiv e-prints (astro-ph/1201.3994)

Serra P., Jurek R., Floer L., 2011, ArXiv e-prints (astro-ph/1112.3162)

Starck J.-L., Fadili J. M., Digel S., Zhang B., Chiang J., 2009, A&A, 504, 641

Westmeier T., Popping A., Serra P., 2011, ArXiv e-prints (astro-ph/1112.3093)