Wavelet decomposition code

Alexey Vikhlinin
avikhlinin@cfa.harvard.edu

July 25, 1999

1   Introduction

The introduction to the wavelet analysis and its application to the astronomical images can be found elsewhere. We point the reader to the work of Grebenev et al. [1] who use the Mexican Hat to detect and analyze the small scale structure in A1367. Another interesting work is that of Slezak et al. [2]; they discuss an iterative image restoration using the *à trous* wavelet transform. Both works have useful bibliographies.
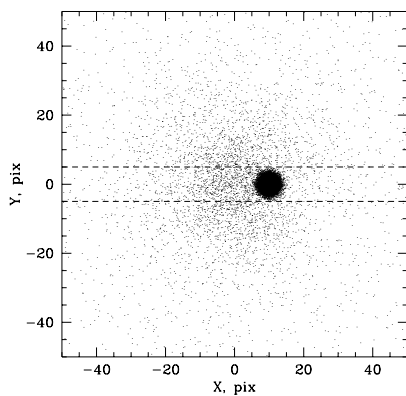
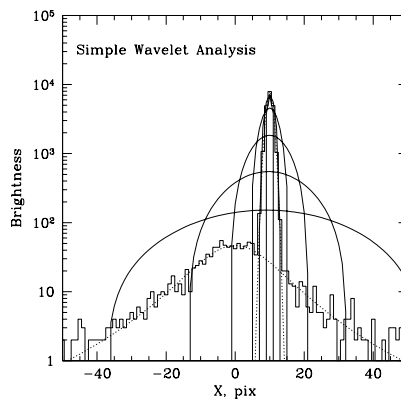## 2   Problem

Figure 1: Sample image

Figure 2: Simple wavelet analysis

The problem solved by the wavelet decomposition can be illustrated by a sample image containing two sources of very different size (Fig. 1), bright narrow "point source" and the fainter, wide "cluster". Our goal is to detect both sources and possibly decompose the image into the original components. These are the tasks that stimulated the use of the wavelet transform for the astronomical image analysis. In traditional variants of the wavelet analysis, one convolves
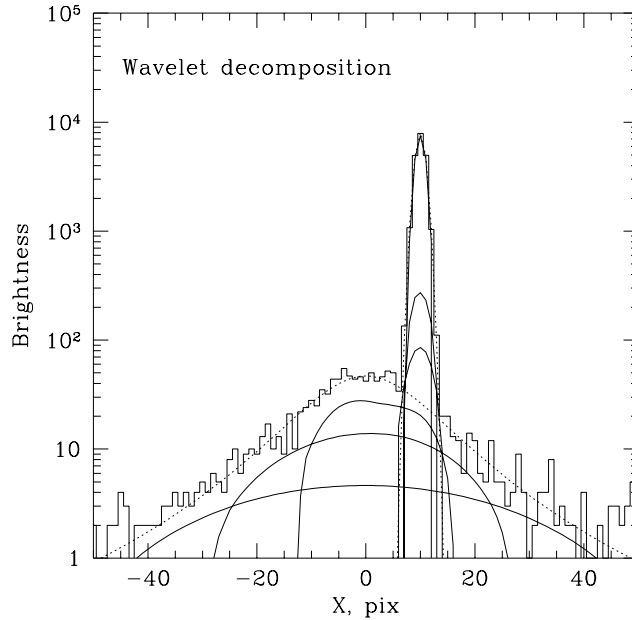
Figure 3: Wavelet decomposition. Solid curves show flux at scales from 1 (the narrowest) to 6 (the broadest)

the data with special filters designed to emphasize structures of a particular size. The traditional method works well in a simple case of isolated sources, but fails for the test image. Figure 2 shows the slice of the surface brightness of the test image convolved with the Mexican Hat filter of different scale (the slice is computed in a band shown by dashed lines in Fig. 1). The convolution is dominated by the point source, and the cluster remains undetected.

The problem of eliminating the contribution of narrow sources to the large-scale emission, which is being solved by our algorithm, can lead to significant improvements in understanding the structures in real astrophysical images. For example, a narrow filamentary structure in the Coma cluster was not detected by either isophotal analysis [3] or simple wavelet analysis [4], but was found [5] using the wavelet decomposition technique described here.

3 Wavelet Decomposition: The Idea

The wavelet kernel we use is called *á trous*. It is described in detail, for example, by Starck & Murtagh [6]. On scale i, this kernel (k_i) equals the difference of two functions, f_i and f_{i+1}, each *very approximately* represented by a Gaussian of width 2^{i-1} (f_1 is the δ-function). The convolution of an image with kernel k_i emphasizes structures with the characteristic size $\approx 2^{i-1}$.

On the largest scale, n the kernel $k_n = f_n$. The original image $I(x,y)$ is
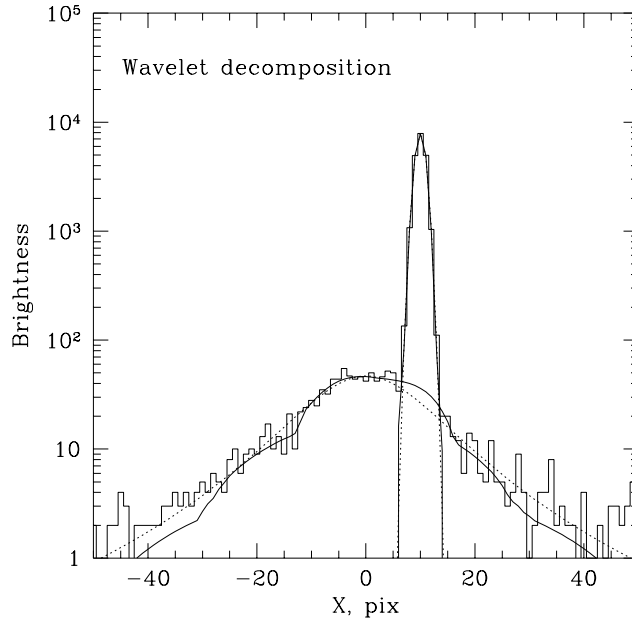
Figure 4: Restoration of the original image. Solid curves show the combined scales 1–3 and 4–6

very simply related to its convolutions, w_i (called "wavelet planes" below), with kernels k_i:

$$I(x, y) = \sum_{i=1}^{n} w_i(x, y). \tag{1}$$

Therefore, we can speak of w_i as images containing "flux" on scale i; sum of all fluxes yields the original image. This is the basis of the wavelet decomposition algorithm. The image is convolved with the wavelet of the first scale. All insignificant features are zeroed, and the remaining, significant, ones are subtracted from the data. After subtraction, we turn to the next scale. In other words, we subtract all significant small-scale features from the data before working at large scales. Clearly, this algorithm greatly reduces the interference of the point source in Fig. 1 with the large-scale kernels. The example is shown in Fig. 3. Small-scale kernels are sensitive to the point source. It gets almost completely subtracted at scale 3. At larger scales, only cluster flux is left in the data. The cluster is therefore easily detected by large-scale kernels (scales 4, 5, and 6).

The described algorithm involved some cleaning of the wavelet planes, i.e. zeroing all features that are below some predefined threshold. Interestingly, equation (1) remains almost valid if thresholds are chosen correctly. Moreover, we can restore individual structures of different sizes by combining the appropriate wavelet planes. For example, Fig. 4 shows that the sum of scales 1–3 and

4–6 almost perfectly restores the original components in the data.

4 Thresholding

4.1 Detection thresholds

The detection thresholds are conceptually very simple. Roughly speaking, we convolve the image with a filter and want to know the level above which all maxima in the convolution are statistically significant. This level is called the detection threshold. To derive the detection threshold, one should calculate the level that is rarely exceeded in the convolved images containing just noise.

4.1.1 Gaussian noise

The simplest case is the noise with the Gaussian distribution and constant rms across the image. First, consider the Gaussian noise with $rms=1$. The distribution of image values upon convolution with the kernel k_i is also Gaussian with the rms σ_i. The values of σ_i for each scale i can be calculated analytically as

$$\sigma_i = \left(\sum_{x,y} k_i^2(x,y) \right)^{1/2} . \tag{2}$$

For the *à trous* kernels, the values of σ_i are given in Table.

i	σ_i
1	0.89080
2	0.20066
3	0.08551
4	0.04122
5	0.02042
...	...
$j+1$	$\approx \sigma_j/2$

Next, consider the uniform Gaussian noise with the rms level n. Since the convolution with the wavelet kernel is a linear transformation, the rms level in the convolved image is $n\sigma_i$. The t sigma significance detection threshold is at the level of $tn\sigma_i$. Thus, the detection threshold determination is straightforward for the uniform Gaussian noise.

Suppose now that the image noise is Gaussian, but its rms varies across the image, $n = n(x,y)$. In this case one can still use the $t\hat{n}\sigma_i$ relation to calculate the detection threshold, where $\hat{n}$ is related to the convolution of the noise pattern with the wavelet kernel as

$$\hat{n}(x,y) = \left(k_i^2 \otimes n^2(x,y) \right)^{1/2} . \tag{3}$$

In practice, it is inefficient to use this equation because the convolution with k_i^2 cannot be performed with the fast algorithm that is used for convolution with k_i. Instead, we convolve $n^2(x,y)$ with the Gaussian with $\sigma = 2^i$ that approximates the negative part of the wavelet kernel.

4.1.2 Poisson noise

This type of noise is usually found in the X-ray astronomy. The general solution to the problem of finding detection thresholds in the case of Poisson noise is given

by Starck & Pierre in [7]. Their approach will be implemented in future releases of the code. Currently, we use the following approximate approach. The 1-σ error on N counts can be approximated as $1 + (n + 0.75)^{1/2}$ [8]. We use this approximation to estimate the error map as

$$\hat{n}(x,y) = \frac{1 + \sqrt{b(x,y)S + 0.75}}{S},\tag{4}$$

where S is the effective area covered by the wavelet kernel, $S = 1/\sigma_i^2$, and $b(x,y)$ is the background. The detection threshold is given, as for the Gaussian noise, by $t\hat{n}\sigma_i$. The background is estimated by convolving the data with the function representing the negative part of the wavelet kernel.

4.2 Filtering thresholds

The detection threshold should be high to prevent false detections. To allow just one false source in a 512×512 image, one should should use the 4.5σ detection threshold. However, the use this high threshold for image filtering leads to a significant loss of useful signal.

This point is illustrated in Fig. 5. The convolution with the wide kernel reveals a source which is $> 4.5\sigma$ significant. However, only the very top of the source is above this significance level. If we set all regions below the

Figure 5:

4.5σ level to zero, we would loose most of the source flux. Fortunately, we know that the wavelet kernel was wide and therefore the detected structure *must be broad*. We, therefore, can keep some region (say, above 2σ) around the detected maximum, and still be sure that the signal kept is real.

The code realizes this scheme as follows. The region above the filtering threshold (also known as the minimum threshold) $t_{\min}$ is kept if and only if it contains a local maximum above the detection threshold (also, the maximum threshold) $t_{\max}$.

This algorithm has a drawback if $t_{\min}$ is set too low, as illustrated in Fig. 6. The secondary maximum has only $\approx 3\sigma$ significance, and normally should not be detected. However, it is kept because it is within the "2σ" region around the main peak. The so-

Figure 6:

lution will be to implement an additional constraint: the region kept should be above $t_{\min}$ *and* within some radius of the $> t_{\max}$ region. In practice, the situation in Fig. 6 is rarely found if $t_{\min} > 2.5 - 3\sigma$.
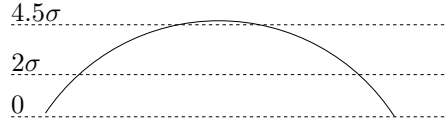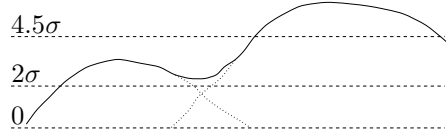
5 Implementation

5.1 Wavelet kernels

Wvdecomp supports two families of the wavelet kernels — the *à trous* kernels [6] and "gauss" kernels. Both families can be thought of as the difference of two Gaussians of unit normalization. On scale 1, the positive part is the δ-function, while the negative part is approximately a Gaussian with $sigma = 1$ pixel. On scale 2, the positive part is approximated by a $\sigma = 1$ pixel, and the negative part by a $\sigma = 2$ pixel Gaussians; the widths increase by a factor of 2 on each subsequent scale.

While the approximation by the two Gaussians is close to the truth, it is not the whole truth. The *à trous* family uses a complicated (but fast) algorithm to compute the kernel. For the "gauss" family, the Gaussian convolution is approximated by three subsequent rectangular box convolutions. Both families produce very similar results[1].

5.2 The code

The program is invoked as `wvdecomp image output [options]`. `Image` is the input image and `output` is the output keyword. The command line options are explained below.

`kernel=gauss/atrous` Set the wavelet kernel; `kernel=atrous` is the default.

`scalemin=smin` Set the scale at which the decomposition process starts (see §3). Default value is 1.

`scalemax=smax` Set the scale at which the decomposition stops. Default is `scalemax=6`.

The program writes a set of output images, `output` and `output.i` where $\texttt{smin} \leq \texttt{i} \leq \texttt{smax} + 1$. `output.i` contains flux detected at scale `i` (see §3 and §4.2), except for `i=smax+1`. $\texttt{output} \equiv \sum_{i=\text{smin}}^{\text{smax}} \texttt{output.i}$. `output.[smax+1]` contains is the residual image (e.g., `image-output`) smoothed with the kernel at the largest scale (§3).

`threshold=t` Set the detection threshold, in sigma (§4.1). Default is `threshold=3`. However, this is insufficient for most applications; for example, this allows for many false detections in a 512×512 image. `threshold=4.5-5` is a reasonable choice for most applications.

It is possible to set individual detection threshold for each scale. For that, the `threshold=ta,tb,...` syntax should be used, where `ta` is the threshold at `smin` and so on.

`thresholdmin=tmin` This parameter, the filtering threshold, is explained in §4.2. Its default value equals the detection threshold at the given scale. `thresholdmin=2.5`

---

[1]I usually use the *à trous* kernel, but the 160 deg^2 cluster survey data were reduced using the "gaussian" family

or more is a good choice. It is possible to set individual `tmin`'s for all scales with the same syntax as for the `threshold` parameter.

iter=niter Set the number of iterations at each scale. The decomposition process is repeated at the same scale until no new structures are detected or until `niter` iterations are made. This parameter defaults to 1. A reasonable choice is `iter=5`.

errimg=error_image You can supply the image containing standard deviations in the data at each pixel. If this parameter is not set, the noise will be estimated from the data assuming either Poisson or Gaussian statistics (the statistics used can be controlled by the `stat` parameter).

stat=poisson,gauss Set the statistics of the image noise. This parameter defaults to `poisson`; however, if `errimg` is supplied, `stat=gauss` is the default.

bg=bgimage Sometimes it is practical to subtract the known structures from the data before the wavelet decomposition. For example, in the case of the ROSAT PSPC data, one should subtract the normalized exposure map to get rid of spurious detections around the PSPC window support structure. If the image noise is Gaussian, this "background" can be subtracted in advance. However, if it is desirable to use the Poisson statistics, this should be done internally in the code. This is the purpose of the bg parameter.

exp=expimage Similar to `bg`, but uses the exposure map to correct the data. Image is divided by the exposure map before convolution with the wavelet kernel, and multiplied back by exposure upon the convolution. This preserves Poisson statistics, but eliminates false detections in the region of strongly varying exposure.

All other parameters listed the the `wvdecomp` help file are experimental and should be used with caution.

There is no way to change the default values of parameters except changing the code. However, your usual setting can be specified in a parameter file which has a form like

```
threshold=5              # detection threshold
thresholdmin=3           !  filtering threshold
iter=5                   - the number of iterations
```

This parameter file can be supplied to the `wvdecomp` as `@par.file` (e.g., `wvdecomp img out @par.file`).

6 Examples

The use of wavelet decomposition for separation of structures of different size was already discussed (§2, 3 and Fig. 1– 4). Other applications, which require the ability to detect sources and combine structures of different size are numerous. Just two examples are shown below.

6.1 Calculation of the background map

The best approximation to the background is the smoothed data from which all detectable sources were subtracted. This is exactly what the largest scale of the wavelet decomposition contains. So, the image `bg.7` created by

```
wvdecomp image bg scalemin=1 scalemax=6 threshold=3 \
thresholdmin=2 iter=10
```

is a good approximation to the background map. Note that in this example, thresholds were set low, because we were interested not in the confident source detection but rather in a complete source subtraction.

6.2 Subtraction of point sources to study the large scale brightness gradients

This is similar to the previous example, with the two exceptions. First, we should stop the wavelet decomposition at small scale, because we would like to subtract only point sources. Second, we may not want to smooth the resulting image. Suppose, scale 3 is already larger than the point source, but still much smaller than the structures of interest.

```
wvdecomp image point scalemin=1 scalemax=3 threshold=3 \
thresholdmin=2 iter=10
```

creates the image of point sources, `point`. Subtraction of `point` from the `image` produces the desired map.

References

[1] Grebenev, S. A., Forman, W., Jones, C., & Murray, S. 1995, *Wavelet transform analysis of the small-scale X-ray structure of the cluster Abell 1367*, ApJ, 445, 607

[2] Slezak, E., Durret, F., & Gerbal, D. 1994, *A wavelet analysis search for substructures in eleven X-ray clusters of galaxies*, AJ, 108, 1996

[3] White, S. D. M., Briel, U. G., & Henry, J. P. 1993, *X-ray archaeology in the Coma cluster*, MNRAS, 261, L8

[4] Vikhlinin, A., Forman, W., & Jones, C. 1994, *Mass concentrations associated with extended X-ray sources in the core of the Coma cluster*, ApJ, 435, 162

[5] Vikhlinin, A., Forman, W., & Jones, C. 1997, *Another Collision for the Coma Cluster*, ApJ, 474, L7

[6] Starck, J.-L. & Murtagh, F. 1994, *Image restoration with noise suppression using the wavelet transform*, A&A, 288, 342

[7] Starck, J. L. & Pierre, M. 1998, *Structure detection in low intensity X-ray images*, A&AS, 128, 397

[8] Gehrels, N. 1986, *Confidence limits for small numbers of events in astrophysical data*, ApJ, 303, 336