A quick guide to wedgelets

Laurent Demaret, Hartmut Führ, Felix Friedrich
Institute of Biomathematics and Biometry
GSF Research Center for Environment and Health
D–85764 Neuherberg

March 1, 2005

Abstract

We give a short introduction to wedgelet approximations, and describe some of the features of the implementation available at the website www.wedgelets.de. Here we only give a short account aiming to provide a first understanding of the algorithm and its features, and refer to [1, 2] for details.

Wedgelet approximations

Wedgelet approximations were introduced by Donoho [1], as a means to efficiently approximate piecewise constant images. Generally speaking, these approximations are obtained by partitioning the image domain adaptively into disjoint sets, followed by computing an approximation of the image on each of these sets. Optimal approximations are defined by means of a certain functional weighing approximation error against complexity of the decomposition. The optimization can be imagined as a game of puzzle: The aim is to approximate the image by putting together a number of pieces from a fixed set, possibly using a minimal number of pieces.

As can be imagined, the efficient computation of such an optimal approximation is a critical issue, depending on the particular class of partitions under considerations. Donoho proposed to use *wedges*, and to study the associated wedgelet approximations. For the sake of notational convenience, we fix that images are understood as elements of the function space $\mathbb{R}^I$, where $I = \{0, \dots, 2^J - 1\} \times \{0, \dots, 2^J - 1\}$. Other image sizes can be treated by suitable adaptation, at the cost of a more complicated notation.

The wedgelet approach can be described by a two-step procedure:

1. Decompose the image domain I into a disjoint union of wedge-shaped sets, $I = \bigcup_{w \in \mathcal{P}} w$.
2. On each set $w \in \mathcal{P}$, approximate the image by a constant.

Here the w are required to be elements of a fixed set $\mathcal{W}$ of wedges. A *wedgelet approximation* of an image f associated to the regularization parameter γ is by definition a minimizer of the functional

$$H_{\gamma,f}(\mathcal{P}, f_{\mathcal{P}}) = \gamma|\mathcal{P}| + \|f - f_{\mathcal{P}}\|_2^2 \quad . \tag{1}$$

Here $f_{\mathcal{P}}$ is a function which is constant on each wedge $w \in \mathcal{P}$, and $\|\cdot\|_2$ denotes the ℓ^2-norm on $\mathbb{R}^I$,

$$\|g\|_2^2 = \sum_{x \in I} |g(x)|^2 \quad .$$

The regularization parameter γ can be interpreted as a scale: As γ ranges from 0 to ∞, the minimizer $\widehat{f}_\gamma$ of (1) runs through a stack of images, starting from the data (for $\gamma = 0$) to a constant image (for $\gamma = \infty$).

A minimizer $(\widehat{\mathcal{P}}_\gamma, \widehat{f}_\gamma)$ of (1) can be viewed as an optimal approximation to f with a prescribed complexity: If $N = |\widehat{\mathcal{P}}_\gamma|$, let X_N denote the set of images in $\mathbb{R}^I$ which are constant on a partition of I into at most N wedges. Then $\widehat{f}_\gamma$ is an element of X_N with minimal ℓ^2-distance to f.

A further useful observation is that since we consider constant approximation, $\widehat{f}_\gamma$ is easily obtained from the optimal partition $\widehat{\mathcal{P}}_\gamma$ by computing mean values of f over the elements of $w \in \mathcal{P}$. Hence the minimization procedure boils down to finding the optimal partition.

This allows yet another interpretation of a minimizer $(\widehat{\mathcal{P}}_\gamma, \widehat{f}_\gamma)$: Let $\epsilon = \|f - \widehat{f}_\gamma\|_2$. Then, among all wedge partitions incurring an approximation error of at most ϵ, $\widehat{\mathcal{P}}_\gamma$ is the one with the smallest number of elements.

Hence, a minimization result of (1) can either be viewed as optimal approximation with (at most) a fixed number $N(\gamma)$ of pieces, or as partition with a minimal number of pieces among those partitions that incur (at most) a fixed approximation error $\epsilon(\gamma)$.

Clearly the properties of this scheme depend on the precise definition of the set $\mathcal{W}$ of admissible wedges. One possible choice could be to take the set $\mathcal{Q}$ of dyadic squares contained in I,

$$\mathcal{Q} = \{[2^j k, 2^j(k+1)[\times[2^j m, 2^j(m+1)[\ : \ 0 \le j \le J \ , \ 0 \le k, m < 2^{J-j}\} \quad .$$

Strictly speaking, the "wedges" appearing in this set do not deserve the name, and approximations based on dyadic squares have been studied for quite some time, usually under the name of "quadtree decompositions". The wedgelet construction proposed by Donoho can be seen as a refinement of this: Then elements of $\mathcal{W}$ are obtained by splitting an element $q \in \mathcal{Q}$ along a suitable straight line, yielding $q = w_1 \cup w_2$. For each dyadic square the lines are prescribed according to a suitable scheme (more on that in the next section). Figure 1 shows examples of optimal approximations using wedges and dyadic squares, with the same number of pieces. Since coding wedges requires more information per piece, the comparison is not exactly fair; but the images manage to convey the motivation behind the construction of wedgelets, which do a much better job at resolving diagonal edges.

With these definitions, an algorithm for the minimization of (1) can be broken down into the following two steps:

1. **Local minimization:** For each dyadic square, determine the wedge split $q = w_1 \cup w_2$ which yields the minimal local approximation error $\|f_p - \widehat{f}_p\|_2^2$. Here f_p is the restriction of f to p, and $\widehat{f}_p$ is the function that assigns each element of w_i the mean value of f on w_i. Store the optimal split and associated approximation error.

2. **Global minimization:** Given γ, compute the optimal wedgelet decomposition $\mathcal{P}$ from the data stored in the first step.

This way of organizing the minimization procedure is based on the following observations. We refer to [1, 2] for more details.

1. From a computational point of view, local minimization dominates the algorithm performance. We are required to compute mean values and ℓ^2-errors for a large number of wedges, and a naive approach to this problem results in huge computation times. An effective solution of this problem is one of the main contributions of our implementation.

2. In view of the previous remarks, it is important to note that the local minimization does not depend on the parameter γ. The quadtree structure associated to dyadic squares allows a fast implementation of the global minimization step, yielding arbitrary access to the minimizers of (1) almost in real time.

Our implementation

The algorithms available on this website depend on a technique that allows a convenient control of the angular resolution of the wedgelet scheme. Given an arbitrary angle θ, we consider a decomposition of the image domain into parallel digitized lines all having angle θ with the horizontal axis. Then the (costly) first step of the minimization procedure is performed for all wedges arising by a split along one of these lines, using lookuptables. Angles can be treated consecutively, which keeps the allocation efforts within reasonable bounds. The number of angles enters linearly into the computation time. Thus our implementation allows an arbitrary transition

Let us quickly give an overview of the features, as they are accessible in the panel. The main purpose of the panel is to provide a platform for the convenient experimentation with wedgelets, with a rapid and convenient visualization of the results. We now give a short description of the panel; confer the page http://www.antsinfields.de/wedgelet/screenshots.html for a screenshot.

- Section "open pgm (ascii) file": This button has the obvious function. Currently we only support pgm (ascii) format; freely available software such as IrfanView (Windows) or GIMP (Linux) can be used for the conversion from other file formats.

- The "create new tree" button allocates the necessary memory for the following steps.

- The buttons in the "add model" section allow to perform a local minimization for the loaded image. Note that there is an array of such buttons. Here the different rows correspond to different choices of the underlying sets of wedges, in increasing order of complexity: "dyadic rectmodel" corresponds to taking dyadic squares as wedges. "wedge model explicit" allows the definition of angles in a direct manner; "wedge model" uses an incremental way of prescribing the angles. The "adaptive wedge model" row refers to a set of admissible angles that depends on the size of the dyadic interval. Clearly a dyadic square of $2^j \times 2^j$ can only resolve $O(2^j)$ angles, and the "adaptive wedge model" scheme reflects this observation. Thus we obtain a considerably better angular resolution than in the "wedge model", at a comparable computational cost.

  The columns of the array of buttons correspond to different ways of locally approximating the image. In addition to piecewise constant approximation of the image, we have also implemented piecewise affine and piecewise quadratic approximation. The generalization to piecewise linear approximation was already suggested by Willett and Nowak [3], who coined the term "platelets" for the resulting system of local approximants. Figure 2 below shows a comparison of locally constant vs. locally linear approximation.

  Since increase in model complexity necessarily implies a better approximation behaviour, the functional that is minimized in the more general setting is given by

  $$H_{\gamma,f}(\mathcal{P}, f_{\mathcal{P}}) = \gamma \#(\text{Parameters }) + \|f - f_{\mathcal{P}}\|_2^2 \quad , \tag{2}$$

  where #(Parameters) is the number of real parameters needed to code the different local approximations, i.e., 1 for constant, 3 for affine, and 6 for quadratic approximation. The price to pay is an obvious increase in computational complexity; in the quadratic case there are also some numerical stability issues to consider (see [2]). On the other hand, for images containing smooth color gradients, the use of higher order models allows a better approximation, as can be seen in Figure 2 below.

  An important additional feature of the implementation is that the different schemes can be used simultaneously. Note that the functional (2) allows to treat varying models within an image. In our implementation, the result of each additional local minimization step is stored separately, and considered later on in the global minimization procedure.

- The minimization section allows to fix the parameter γ. Pressing a button from the "open output" section then allows to view the result. The displayed minimization result is automatically updated after a new choice of γ. The parameter can be prescribed numerically or via sliders. Moving

the "heavy drag slider" around while pressing the left mouse button allows to view the family of minimizers $(\widehat{\mathcal{P}}_\gamma, \widehat{f}_\gamma)$ for different γ essentially in real time.

- The "open output" section allows the display of the minimization results. The "show array" button displays $\widehat{f}_\gamma$, the "show tree" button displays the partition $\widehat{\mathcal{P}}_\gamma$, and "show both" superimposes $\widehat{\mathcal{P}}_\gamma$ onto $\widehat{f}_\gamma$.
- The "report" button allows to view computation times and properties of the most recently computed global minimization step, such as psnr (in comparison to the original image) and number of active pieces. The button "remove models" removes the results of the local minimization steps, to allow starting over with a new model.

Websites with related contents and implementations

We are aware of the following websites containing implementations and literature concerned with wedgelet approximation, or more generally, with geometric multiscale image approximation.

- The matlab toolbox BeamLab200 was developed by Donoho and his collaborators; it is available at http://www-stat.stanford.edu/∼beamlab/ . The toolbox contains an implementation of a wedgelet approximation algorithm. In addition to that BeamLab200 contains a host of other algorithms performing ridgelet, curvelet and beamlet transforms. Also, preprint version of related papers can be downloaded.
- The website http://www-ece.rice.edu/∼willett/Research/software.html contains matlab implementations of greedy, coarse-to-fine wedgelet and platelet approximation algorithms, mostly in a denoising context.
- A website containing contourlet transforms, a digital implementation of the curvelet system, can be found under http://www.ifp.uiuc.edu/∼minhdo/software/ .

References

[1] D. Donoho. *Wedgelets: nearly minimax estimation of edges,* Ann. Statist. 27, no. 3, 1999, pp. 859–897.

[2] L. Demaret, F. Friedrich, H. Führ and K. Wicker. *Discrete Green's theorem for polygonal domains, with an application to rapid wedgelet approximation.* Preprint 2005, available at http://ibb.gsf.de/preprints.php.

[3] R.M. Willett and R.D. Nowak. *Platelets: A multiscale approach for recovering edges and surfaces in photon-limited medical imaging,* IEEE Transactions on Medical Imaging **22** 2003, pp. 332–350.

Figure 1: `IBB North.` 640×480 (a) Original image, (b) approximation using 1000 dyadic squares, and (c) approximation using 1000 wedges.

Figure 2: `IBB North.` 640×480 (a) Original image, (b) (b) locally constant approximation using 1000 wedges (c) locally affine approximation using 1000 parameters.