The *dwtrans2d* package

Managing 2-dimensional dyadic wavelet transforms
Computation of the corresponding extrema
Reconstruction of the original image from the extrema

Emmanuel Bacry
CMAP, Ecole polytechnique, 91128 Palaiseau Cedex, France
email : lastwave@cmap.polytechnique.fr
web : http://www.cmap.polytechnique.fr/˜ bacry/LastWave

The *owtrans2d* package was co-written by

- **E.Bacry** *CMAP, Ecole Polytechnique, 91128 Palaiseau Cedex France.*
- **J.Fraleu** *CMAP, Ecole Polytechnique, 91128 Palaiseau Cedex France.*
- **W.L.Hwang** *Institute of Information Science,Academia Sinica,Taipei, Taiwan.*
  email : whwang@iis.sinica.edu.tw
- **J.Kalifa** *LetItWave, Ecole Polytechnique, 91128 Palaiseau Cedex France.*
- **E. Le Pennec** *CMAP, Ecole Polytechnique, 91128 Palaiseau Cedex France.*
- **S. Mallat** *CMAP, Ecole Polytechnique, 91128 Palaiseau Cedex France.*
- **S. Zhong**, *Department of Civil Engineering and Operations Research, Princeton University, E220 Engineering Quadrangle Princeton NJ 08544.*
  email : szhong@chelsea.princeton.edu

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION
Version 2, June 1991

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".
Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.
1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.
You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.
2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
- c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.
Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.
In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.
3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail

to address new problems or concerns.
Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.
10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.
12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Contents

Part I

Documentation

Chapter 1

Using the 2d dyadic wavelet transform and extrema (`dwtrans2d`) package

1.1 Introduction

This package allows to perform dyadic wavelet transforms of images and computation of the corresponding extrema. Moreover, it allows the recontruction of the original image from these extrema and lets you manipulate these extrema so that you can perform high level denoising. This denoising procedure and the theory corresponding to the dyadic wavelet transform extrema can be found in the article *Characterization of signals from multiscale edges* by Stephane Mallat and Sifen Zhong, which appeared in IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 14, No. 7, p. 710-732 in July 1992.

1.2 loading the `&dwtrans2d` package

As for any package, you must load the `dwtrans2d` package for using 1d-wavelet transforms. This is done by the original startup file or directly by typing `package load dwtrans2d` in the terminal.

1.3 The `&dwtrans2` structure

1.3.1 Introduction

In LastWave, a dyadic wavelet transform is stored in a structure called a *dwtrans2* structure and corresponds to the type `&dwtrans2`. (In the present version of this package, there is no type defined for extrema related structure). The resulting images as well as the original images are stored in a 2d array of images. The first index corresponds to the octave number `oct`. Only $oct \geq 1$ are used by a wavelet transforms. Consequently the images with `oct=0` are working images available for the user. The second index, that we will referred to as i could be one of

- $i = 0$: the image corresponding to the smoothed image at the corresponding octave number `oct` (i.e., the projection on the corresponding V space)
- $i = 1$: the image corresponding to the x-wavelet image at the corresponding octave number `oct` (i.e., the projection on the corresponding W_x space)
- $i = 2$: the image corresponding to the y-wavelet image at the corresponding octave number `oct` (i.e., the projection on the corresponding W_y space)
- $i = 3$: the image corresponding to the modulus wavelet image at the octave number `oct` (i.e., the modulus of the complex image which real part is the x-wavelet image and which imaginary part is the y-wavelet image)
- $i = 4$: the image corresponding to the phase wavelet image at the octave number `oct` (i.e., the phase in $[0, 2\pi]$ of the complex image which real part is the x-wavelet image and which imaginary part is the y-wavelet image)

1.3.2 The fields of `&dwtrans2`

The main fields of the `dwtrans2` structure are

- `noct` : the number of octaves used for the decomposition. An octave value of 1 means that the image is decomposed on a single level corresponding to scale 2. The dyadic wavelet transform at this scale corresponds to 2 images of the same size as the original image. One image corresponds to the variation along the x-coordinate (the x-wavelet image) and the other one to the y-coordinate (the y-wavelet image). The number of octaves *noct* can take any value between 1 and $\log_2 N + 1$ where N is the number of row of the original **square** image
- `[oct,i]` : an 2d array of images as explained in the previous section.

1.4 A simple example of wavelet decomposition/reconstruction

In this section, we basically describe the demo file `scripts/dwtrans2d/DemoDWtrans2d` (the `DemoDWtrans2dWT` command). We want to perform the dyadic wavelet transform of the *lenna* image. Loading the `dwtrans2d` package creates 2 variables `a2d` and `b2d` of type `&dwtrans2`. Moreover, it also defines two commands `a2d` (resp. `b2d`) which sets the current object to be the variable `a2d` (resp. `b2d`). Indeed, in order to access the image `[<oct>,<i>]` of `a2d`,one can either use the regular syntax

```
a2d[<oct>,<i>]
```

or the abreviated syntax

```
<oct><i>a2d
```

or, in the case `a2d` is the current object, the syntax

```
a2d> <oct><i>
```

We are going to use the variable `a2d` for all our computations. Thus it is convenient to set the current object to `a2d` :

```
(&wtrans) a> a2d
(&dwtrans2) a2d>
```

Since we are dealing with black and white images, we should set the current colormap to be the `grey` colormap (which should have been defined in your `startup` file) :

```
(&dwtrans2) a2d> colormap current 'grey'
```

Then we read the "lenna" image as the original image to be decomposed (i.e., in the image `[0,0]` of the `a2d` variable). The file is in the directory `image/` of the original scripts directory. The original script directory is stored in the global variable `_scriptDir`. Thus

```
(&dwtrans2) a2d> iread 0 '$_scriptDir/image/lenna.char' -c
```

Then we perform the dyadic wavelet decomposition on 5 octaves and display it :

```
(&dwtrans2) a2d> dwt2d 5
(&dwtrans2) a2d> dw2disp
```

The `dw2disp` is a script command that is defined in the `scripts/dwtrans2d/dwtrans2d.pkg` file. It displays the x-wavelet transform image at the different octaves in the first row. The y-wavelet transform images are displayed in the second row. The modulus in the third and the phases in the fourth. The current colormap (i.e., the `grey` one) is used and all the images are displayed using the normalization method `max` (see Section on images in the main LastWave manual) except the modulus images which are coded using the default normalization method (`+max`) (see Section on images in the main LastWave manual) using the inversed default colormap (i.e., if the current colormap is the `grey` colormap, black pixels will correspond to high modulus values).

Since the reconstruction command reconstructs the image in the image `[0,0]`, in order not to loose the original image we are going to make a copy into the image `[0,1]` :

```
(&dwtrans2) a2d> 1 = 0a2d
```

then perform the reconstruction (by default it is set in in `[0,0]`)

```
(&dwtrans2) a2d> dwt2r
```

Then we can display in the same window the original image, the reconstructed image and the error image :

```
(&dwtrans2) a2d> disp {0 1 0a2d-1a2d} -title "Original/Reconstruction/Error" -pos 10 40
```

Let us note that the `dwt2f` command lets you choose the filter you want to use for decomposition/reconstruction. However, for now, only one filter (named `p3`) can be used.

1.5 A simple example of extrema decomposition and reconstruction

In this section, we basically describe the demo file `scripts/dwtrans2d/DemoDWtrans2d` (the `DemoDWtrans2dExtrema` command). We want to perform the dyadic wavelet transform of the "lenna" image, compute the corresponding extrema and reconstruct the image from the extrema or from the thresholded extrema.

In order to perform the decomposition, we just do as in the previous section :

```
(&wtrans) a> a2d
(&dwtrans2) a2d> colormap current 'grey'
(&dwtrans2) a2d> iread 0 '$_scriptDir/image/lenna.char' -c
(&dwtrans2) a2d> dwt2d 5
```

Then in order to compute the extrema, one just needs to call the `extrema2` command :

```
(&dwtrans2) a2d> extrema2
```

There is no way to display the extrema as a specific graphic object. In order to dispaly them, you need to use an image graphic object. The `e2image` command fills up 2 images. The first one (the modulus image) is filled up with zeros where there are no extrema and with the extrema modulus values where there are extrema. The second one is filled up with zeros where there are no extrema and with the extrema phase values where there are extrema. Thus, for instance, if you want to set the images `[0,1]` and `[0,2]` with the extrema modulus and phases of the extrema at octave 3, you should type

```
(&dwtrans2) a2d> e2image 1 1 2 -..view.* -cm '_'
(&dwtrans2) a2d> disp {1 2}
```

Let us note that the `-..view.* -cm '_'` option allows to set the colormap of the two images to be the inversed current colormap. In our case the current colormap if the `grey` one. Thus the high modulus values will be displayed using black whereas white will correspond to pixels where there are no extrema.

If you just want to get a binary image (indicating just the position of the extrema and not their values), you should use the `-p` option

```
(&dwtrans2) a2d> e2image 1 1 2 -p
(&dwtrans2) a2d> disp 1 -..1 -cm '_'
```

Before performing the extrema reconstruction (using 10 iterations) we must save the original image (in `[0,2]` for instance)

```
(&dwtrans2) a2d> 2=0a2d
(&dwtrans2) a2d> e2recons 10
```

Let us copy the result in the image `[0,1]`

```
(&dwtrans2) a2d> 1=0a2d
```

Actually before performing the extrema reconstruction, we could erase the extrema at all octaves whose modulus are small (e.g., for compression or for denoising). This is done using the `e2thresh` command

```
(&dwtrans2) a2d> e2thresh 0 20
```

The first `0` argument means that extrema at all octaves should be thresholded (otherwise you should have specified an octave number instead of 0) and `20` sets the threshold (it assumes a power law scaling across scales, read the corresponding help). Then we can perform the reconstruction

```
(&dwtrans2) a2d> e2recons 10
```

and display in the same window the reconstructed image using the thresholded extrema, the reconstructed image using all the extrema and the original image :

```
(&dwtrans2) a2d> disp {0 1 2} -title "Threshold/reconstruct/original"
```

Let us note that there is a another algorithm for extrema reconstruction using the conjugate gradient method and which corresponds to the command `e2reconsgrad`.

1.6 Other use of the `dwtrans2d` package (including denoising)

In the `scripts/dwtrans2d/WaveletTour` directory, there are 4 files which (when sourced) display figures that where use as examples in the book *A wavelet tour of signal processing* by S.Mallat (Academic Press, 1998). The figures are the following

- Figure 6.9 (file `fig69`) : It corresponds to a dyadic wavelet decomposition and extrema of a simple "circle" image,
- Figure 6.10 (file `fig610`) : It corresponds to a dyadic wavelet decomposition and extrema of the "lenna" image and reconstruction using all the extrema or just the strongest extrema,
- Figure 6.11 (file `fig611`) : Same as figure 6.11 except that the extrema reconstruction method is the conjugate gradient method.

- Figure 10.6 (file `fig106`) : Denoising of the "pepper" image using a very sophisticated algorithm described in the paper *Characterization of signals from multiscale edges* by Stephane Mallat and Sifen Zhong, which appeared in IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 14, No. 7, p. 710-732 in July 1992. It chains the extrema along scales and then denoise the noisy image using the `denoise2` interactive command which deals directly with these chains (not with individual extrema). The `denoise2` command will ask you questions about which chains you want to keep. In our example, all the answered to those questions have been stored in a file called `param` (in the same directory) and we just redirected the standard input of the command. In a future version, we will give more details about this procedure.

Part II

Reference

Chapter 2

Package `dwtrans2d` 2.0

Package allowing to perform 2d dyadic wavelet transform decomposition, reconstruction and extrema reconstruction
*** Authors and Copyright : E.Bacry, J.Fraleu, J.Kalifa, E.LePennec, S.Mallat, W.L.Hwang, S.Zhong*

2.1 Defined types

2.1.1 Type `&dwtrans2`

This type is the basic type for 2d dyadic wavelet transforms. It contains a 2d array of images. The first index 'oct' corresponds to the octave number and the second index is called 'orient'. When a wavelet transform is performed, the analyzed image must be in [0,0]. The projection on the V_oct spaces are stored in [oct,0]. [oct,1] corresponds to the vertical details at octave oct and [oct,2] corresponds to the horizontal details at octave oct. [oct,3] corresponds to the norm of ([oct,1],[oct,2]) and [oct,4] to its phase. All the other images are not used by the wavelet transform and can be used as working images.

- `&dwtrans2 [oct,orient]`
  it returns the image which corresponds to octave <oct> and orientation <orient>. The image [0,0] corresponds to the analyzed image. The projection on the V_oct spaces are stored in [oct,0]. [oct,1] corresponds to the vertical details at octave oct and [oct,2] corresponds to the horizontal details at octave oct. [oct,3] corresponds to the norm of ([oct,1],[oct,2]) and [oct,4] to its phase. All the other images are not used by the wavelet transform and can be used as working images.
- `&dwtrans2.name [= <name>]`
  Sets/Gets the name of a dwtrans2
- `&dwtrans2.noct [= <noct>]`
  Gets the number of octave of a 2d dyadic wavelet transform.
- `&dwtrans2.wavelet`
  Gets the analyzing wavelet used for the dyadic wavelet transform.

2.2 Commands related to wavelet 2D transform

• **dw2read** `[<dwtrans2>=objCur] (<filename> | <stream>)`
Reads a wavelet transform from a file named <filename> or a <stream>.

• **dw2write** `[<dwtrans2>=objCur] (<filename> | <stream>)`
Writes a wavelet transform in a file named <filename> or a <stream>.

• **dwt2d** `[<dwtrans2>=objCur] <noct> [-N] [-m]`
Performs a dyadic wavelet decomposition on the current <dwtrans2> on <noct> octaves. The image to be analyzed is supposed to be in the image 0 of the <dwtrans2>. If the number of columns and the number of rows are N = 2ˆ p, then <noct> must range from 1 to p+1. When <oct> is equal to p+1, the coarsest is a constant. Options are :

• **dwt2f** `<filterFileName>`
Specifies the filename of the filters that must be used in case a new dyadic wavelet decomposition is performed (using the 'dwt2d' command). The filter directory (defined in the file 'scripts/dwtrans2d/dwtrans2d.pkg') contains all the filter available files. The filenames ending by the '.1' suffixes correspond to decomposition filters and the ones ending with the '.2' suffixes correspond to reconstruction filters. For this command, you must specify the name without the suffix (it will load both the reconstruction and decomposition filters at the same time). For now, only one pair of decomposition/reconstruction filters are available? Its name is 'p3'. More details about filters can be found in the Appendix A of the paper :
Characterization of signals from multiscale edges
by Stephane Mallat and Sifen Zhong,
IEEE Transactions on Pattern Analysis and Machine Intelligence,
Vol. 14, No. 7, p. 710-732, July 1992.

• **dwt2r** `[<dwtrans2>=objCur] [-N]`
Performs a dyadic wavelet reconstruction from a dyadic wavelet decomposition (made with the 'dwt2d' command). It reconstructs the image which corresponds to the dyadic wavelet transform stored in <dwtrans2>. It uses the same number of octaves as the decomposition has been made with. The reconstructed image is put in image 0 of <dwtrans2>.

2.3 Commands related to ext used in wavelet 2D transform

• **e2image** `[<dwtrans2>=objCur] <octaveNumber> <modImage> <phaseImage> [-p]`
Copies the modulus and phase of extrema at octave number <octaveNumber> to the images <modImage> for the modulus and <phaseImage> for the phase. If '-p' is set then the <modImage> is just made of 1 (to indicate an extrema) or 0.

• **e2read** `[<dwtrans2>=objCur] (<file> | <stream>)`
Reads an extrema representation in a <file> or a <stream>

• **e2recons** `[<dwtrans2>=objCur] <iteration> [-N][-i][-n]`
Reconstructs the image from the extrema of the wavelet transform (computing using the 'extrema2' command). The algorithm reconstructs an approximation of the original image,

by alternatively projecting between two affine spaces. The reconstructed image is in image 0 of <dwtrans2>. The parameter <iteration> gives the number of alternative projection desired for the reconstruction. 20 iterations are sufficient to reconstruct an approximation with an SNR larger than 25 db. Options are :
-i : Does not initialize the reconstruction algorithm. It begins the iterations from an image (in 0) obtained after <n> iterations previously performed and from the associated extrema. The resultant image is stored in 0 and corresponds to an approximation of the original image after <n>+<iteration> iterations.

• **e2reconsgrad** `[<dwtrans2>=objCur] <iterations> [<threshold>=0]`
Reconstruction from the wavelet maxima with the conjugate gradient algorithm. Before the reconstruction is performed, the extrema are thresholded using <threshold>.

• **e2thresh** `[<dwtrans2>=objCur] <octaveNumber> <threshold>`
Thresholds the extrema at scale <octaveNumber> and whose modulus are smaller than <threshold>2ˆ (<octaveNumber>/2)/3. If <octaveNumber> is 0 then all the scales are thresholded.

• **e2write** `[<dwtrans2>=objCur] (<file> | <stream>)`
Writes an extrema representation in a <file> or a <stream>

• **extrema2** `[<dwtrans2>=objCur] [-o] [-n] [-N]`
Computes the extrema representation of the current wavelet transform. The extrema are defined as the modulus local maxima along some gradient direction. The command returns the number of extrema found. Options are :
-n : Due to the discretization of the filter values, the wavelet modulus maxima of a step edge do not have the same amplitude at all scales. In the extrema computation, we have correlated the extremas modulus such that the extremas of a step edge in each scale will have the same values. We called this process the 'normalization process'. With -n, we eliminate the normalization process in the computation of the extrema.
-o : The extrema are defined as the modulus local maxima in any gradient direction. It is possible that, for example, the modulus at location (x,y) is bigger than the modulus at (x-1,y) but less than that at (x+1,y), however the gradient at location (x+1,y) is opposite to the gradient at (x,y). We would expect in this case that the intensity at the location (x,y) is a local maxima and include the (x,y) as a extrema location. With -o, we will eliminate this inclusion.Properties of the extrema are described in the paper :

2.4 Commands related to the noise in pictures

• **denoise2** `[<dwtrans2>=objCur]`
Removes the white noise in images by thresholding extrema chains corresponding to particular values of Lipschitz exponents. Lipschitz exponents are calculated from the decay of the wavelet transform modulus local extrema along chains. The denoising process proceeds as follows :
1. thresholds all the extrema in a chain whose Lipschitz exponent is smaller than a given threshold.

2. thresholds all the chains at any level according to their lengths or average amplitudes.
3. removes all the extrema not propagating to a given scale.
4. rechains the chains such that all the chains in a fine scale corresponding to the same coarser chain are grouped together. We know that a contour has chains at many scales. Moreover, the white noise's energy distributes mostly in fine scales. Thus, the chains in fine scales are distorted more severely than those in coarse scales. This is the reason why we need to rechain the chains (in fine scales) by making use of the chains in a coarse scales.
5. smoothes the abscissa and amplitude along the chains at some scales by a Gaussian kernel with a given standard deviation.
6. predicts a chain's phases in fine scales from the abscissa of the chain and smoothes the resultant phases.

2.5 Script Commands

• **dw2disp** (in file `scripts/dwtrans2d/dwtrans2d.pkg`) `[<owtrans2>=objCur]`
This function displays a dyadic 2d wavelet transform (i.e., a variable of type '&dwtrans2') in a window. If no <dwtrans2> is specified then the current object is used. The display is organized as follows : each row corresponds to a different scale from small scales (top) to large scales (bottom). The first column corresponds to the horizontal wavelet component. The second one corresponds to the vertical component. The third one to the modulus and the last one to the phase. Except the modulus (which uses a '+max' normalization and the inverted current colormap), all the other images use a 'max' normalization and the current colormap. Let us note that you can us the mouse on each image of the display in order to perform zooms.

2.6 Demos

Here is a list of all the Demo files and for each of them all the corresponding Demo commands. To try a Demo command, you should first source the corresponding Demo file then run the command. (When sourcing the Demo file, LastWave tells you about all the commands included in this file).
The Demo files corresponding to this package are :

Demo file **DemoDWtrans2d**

• **DemoDWtrans2dExtrema** (in file `scripts/dwtrans2d/DemoDWtrans2d`)

Demo command that computes the extrema of the dyadic wavelet transform of the 'lenna' image on 5 octaves. It displays the original image, the reconstruction image (reconstructed from the extrema) as well as the error image.

• **DemoDWtrans2dWT** (in file `scripts/dwtrans2d/DemoDWtrans2d`)

Demo command that computes the dyadic wavelet transform of the 'lenna' image on 5 octaves and displays it. It also displays the reconstruction image as well as the error image.

Demo file **DemoDWtrans2dWT**

• **DemoDWtrans2dWT_10_6** (in file `scripts/dwtrans2d/DemoDWtrans2dWT`)

Demo command that reproduces the figure 10.6 of the book 'A Wavelet Tour in Signal Processing' by S. Mallat.

• **DemoDWtrans2dWT_6_10** (in file `scripts/dwtrans2d/DemoDWtrans2dWT`)

Demo command that reproduces the figure 6.10 of the book 'A Wavelet Tour in Signal Processing' by S. Mallat.

• **DemoDWtrans2dWT_6_11** (in file `scripts/dwtrans2d/DemoDWtrans2dWT`)

Demo command that reproduces the figure 6.11 of the book 'A Wavelet Tour in Signal Processing' by S. Mallat.

• **DemoDWtrans2dWT_6_9** (in file `scripts/dwtrans2d/DemoDWtrans2dWT`)

Demo command that reproduces the figure 6.9 of the book 'A Wavelet Tour in Signal Processing' by S. Mallat.

Index