About WaveLab

Jonathan Buckheit, Shaobing Chen,
David Donoho, Iain Johnstone
Stanford University
&
Jeffrey Scargle
NASA-Ames Research Center

Version .850
December, 2005

Abstract

Wavelab is a library of Matlab routines for wavelet analysis, wavelet-packet analysis, cosine-packet analysis and matching pursuit. The library is available free of charge over the Internet. Versions are provided for Macintosh, UNIX and Windows machines. Downloading and installation instructions are given here.

Wavelab has over 1200 .m files which are documented, indexed and cross-referenced in various ways. In this document we suggest several ways to get started using Wavelab: (a) trying out a point-and-click browser, which allows one to interactively select datasets and compute their wavelet transforms; (b) running various demonstrations, which illustrate topics ranging from the visual appearance of various wavelets to the wavelet compression of certain images,(c) browsing the extensive collection of source files, which are self-documenting and (d) reproducing the figures from the book *Wavelet Tour of Signal Processing* by Stephan Mallat [30].

Wavelab makes available, in one package, all the code to reproduce all the figures in our published wavelet articles. The interested reader can inspect the source code to see exactly what algorithms were used, and how parameters were set in producing our figures, and can then modify the source to produce variations on our results. Wavelab has been developed, in part, because of exhortations by Jon Claerbout of Stanford that computational scientists should engage in "really reproducible" research.

This document helps with installation and getting started, as well as describing the philosophy, limitations and rules of the road for this software.

Acknowledgment of Support. This work was partially supported by NSF DMS 92-09130 and 95-05151 (Stanford), by the NASA Astrophysics Data Program (through NASA-Ames Grant NCC2-5100), ASOR MURI, by DARPA BAA-98-04, By NSF KDI, NSF DMS 00-77261, NSF DMS 01-40698 and by other sponsors.

Contents

1 Introduction **3**

2 Access and Installation **4**
2.1 Platform-Specific Information . . . 4
2.2 WEB Acess . . . 4
2.3 Installation . . . 4
2.4 Pathnames . . . 6
2.5 Checklists . . . 6
2.5.1 UNIX Checklist . . . 6
2.5.2 Macintosh Checklist . . . 7
2.5.3 PC Checklist . . . 9
2.6 Success . . . 10

3 Getting Started **11**
3.1 Snooping . . . 11
3.1.1 Contents Files . . . 11
3.1.2 Help for Functions . . . 13
3.1.3 Source Browsing . . . 13
3.1.4 Documentation Directory . . . 14
3.1.5 Dataset Documentation . . . 15
3.2 Demos . . . 15
3.2.1 Short Course Demo . . . 16
3.2.2 Demo Inventory . . . 19
3.3 Browser . . . 19
3.3.1 Example: De-Noising . . . 21
3.3.2 Example: Thresholding Caruso . . . 21
3.4 Toons . . . 22
3.4.1 Outline of Toons . . . 22
3.4.2 Example: Plotting Wavelets . . . 25
3.4.3 Example: Looking at a 2-d Wavelet Transform . . . 27
3.4.4 Example: 2-d Compression Methods . . . 27
3.5 Books . . . 30
3.5.1 Devil's staircase and its wavelet transform . . . 30
3.5.2 Example-Linear and nonlinear approximation for Lena's image . . . 31
3.6 Reproducible Research . . . 31
3.7 Freeware . . . 33

4 Fine Print **35**
4.1 Dependence on Matlab . . . 35
4.2 Registration – Wavelab Registration . . . 35
4.3 Limitations . . . 35
4.4 Support . . . 35
4.5 No Charge – No Charge for Wavelab Software . . . 36
4.6 No Warranty – No Warranty on WaveLab software . . . 36
4.7 Copyright – Wavelab Copying Permissions . . . 36
4.8 Thanks – Thanks to contributors . . . 37

1 Introduction

Wavelab is a library of Matlab routines for wavelet analysis, wavelet-packet analysis, cosine-packet analysis and matching pursuit. This library has been used in teaching courses on the Wavelet Transform and related Time-Frequency transforms at Berkeley and at Stanford. The library is also the basis for wavelet research by the authors, and may be used to reproduce the figures in their published articles, and to redo those figures with variations in the parameters.

The library is available free of charge over the Internet by WWW access; instructions are given below. The material is, however, copyrighted, so that advance permission is required for any commercial use.

The package agrees in philosophy with the "orthogonal wavelet transforms" school, and with associated orthogonal time-frequency transforms, such as wavelet packets and cosine packets. The less-mathematically inclined reader is encouraged to study Y. Meyer's book published by SIAM in English [28]; This will give the reader a conceptual background for the whole package. Other references for the philosophy expressed here include the book of Ingrid Daubechies [14] (for background on orthogonal wavelet transforms), the papers of Coifman, Meyer and Wickerhauser [12] (for background on wavelet and cosine packets and adaptive choice of time-frequency bases), the book of Wickerhauser [29] and the paper by S. Mallat and S. Zhang on matching pursuit [27] and the book of S. Mallat [30] (.850 version of Wavelab includes almost all the scripts that produce the figures in Mallat's book. Exploring and running the scripts is a good way to understand the book itself).

In addition to routines implementing basic wavelet transforms for finite data sets (both periodic transforms and boundary-corrected transforms), wavelet-packet analysis, cosine-packet analysis ($\equiv$ local cosine analysis of Coifman and Meyer) and matching pursuit, the library contains scripts which give a quick guide to wavelets, wavelet packets, cosine packets, matching pursuit and related concepts and which perform elementary data compression and de-noising tasks. We believe that by studying these scripts one can quickly learn the practical aspects of wavelet analysis and one can learn how to use the Wavelab software library.

In this guide we give information which will help you access and install the software on your machine and get started in exploring the resources contained in the Wavelab distribution. We also explain the philosophy which underlies our distribution of the software, and some of the fine print associated with the software.

There are other resources for obtaining information about Wavelab. First, there is the *WaveLab Reference*, a rather long document giving details about all the functions and scripts contained in the package. Second, there is a *WaveLab Architecture* guide which gives details about how Wavelab is constructed and maintained.

This body of software is under continuing development by a team of researchers supported by a grant from the NASA Astrophysics Data program, and from other sponsors. The main aim is research – to develop specific tools for specific goals in adaptive wavelet analysis. We conduct our research with the idea, from the beginning, that we will implement our tools in Wavelab. We believe that the discipline this entails makes our research of a higher quality than otherwise possible.

We welcome your suggestions for further enhancements, and any contributions you might make.

2 Access and Installation

The Wavelab library contains .m files (Matlab code), .mex files (compiled dynamically loadable code), datasets, documentation, scripts and workouts (both also .m files) for reproducing the figures in articles by the authors.

The whole library consists of over 1200 files and 50 subdirectories. It requires more than 2MB and less than 5Mb space on disk once it is downloaded, decompressed and installed.

This documentation refers to Version .850 of Wavelab.

2.1 Platform-Specific Information

Wavelab is available for use in Matlab 6.X or 7 on three different platforms: Windows XP or 2000 , UNIX and Macintosh. The package is made available as a compressed archive, in a .zip format.

You do have to know about one convention used in the documentation. We always use the UNIX pathname conventions rather than PC or Macintosh, e.g. `Matlab/Toolbox/Wavelab` rather than `Matlab\Toolbox\Wavelab` or `Matlab:Toolbox:WaveLab`. You have to transliterate what we say into the version appropriate for your platform.

2.2 WEB Acess

To download the compressed archive from the web, point your web browser to http://www-stat.stanford.edu/˜ wavelab to access the Wavelab web-page. Once there, mouse click under the section "How to Download?".

2.3 Installation

In this section we first describe the installation process in narrative form, and later give a step-by-step checklist.

Once the appropriate compressed archive has been transferred to your machine, it should be decompressed and installed. You will need an appropriate software to decompress .zip file Wavelab850.zip. On a personal computer (Macintosh or Windows), the archives should be decompressed and installed as a subdirectory of the `Toolbox` directory inside the `matlab` folder. On a UNIX workstation or server, the archives could either be installed in the systemwide `matlab` directory, if you have permission to do this, or in your own personal `matlab` directory, if you do not.

Once the actual files are installed, you should have a number of files and subdirectories in the directory Wavelab. If you look in the files Contents.m inside of the Wavelab directory, you will see a plan of what is inside:

```
% WaveLab Main Directory, Version 850
%
% This is the main directory of the WaveLab package; the full package
% contains over 1200 files, consisting of .m files, .mex files and datasets.
%
%              .m files in this directory
%
%   Contents.m          -    This file
%   startup.m           -    Sample Startup file
%   WavePath.m          -    Sets up global variables and pathnames
%   InstallMEX.m        -    Install MEX files
%
%              Subdirectories
%
%   Biorthogonal/       -    Bi-Orthogonal Wavelet Transform tools
%   Books               -    figures for books
```

```
%     /WaveTour          -    Figures of the book "A Wavelet Tour of Signal Processing"
%   Browsers/            -    WaveLab Browsers
%     /One-D             -    One-D Signal Browser
%     /WaveTour          -    Figures of the book "A Wavelet Tour of Signal Processing"
%   Continuous/          -    Continuous Wavelet Transform tools
%   Datasets/            -    Data for use with WaveLab
%   DeNoising/           -    Wavelet Shrinkage tools
%   Documentation/       -    System-Wide Documentation
%   FastAlgorithms/      -    Tools for Fast Algorithms
%   Fractals/            -    Fractal Analysis
%   Interpolating/       -    Refinement schemes & Bi-Orthogonal Wavelet Transforms
%   Invariant/           -    Invariant Wavelet Transform tools
%   Median/              -    Median Interpolating Pyramid Transform Tools
%   MEXSource/           -    c reservoir for mex files.
%   Meyer/               -    Periodic Meyer Wavelet Transform tools
%   Orthogonal/          -    Standard Orthogonal Wavelet Transforms
%   Packets/             -    Wavelet Packets, Cosine Packets, Best Basis Algorithm
%     /One-D             -    1-d Wavelet Packets, Cosine Packets, Best Basis
%     /Two-D             -    2-d Wavelet Packets, Cosine Packets, Best Basis
%   Pursuit/             -    Matching Pursuit in Wavelet Packet/Cosine Packet Dict.
%   Stationary/          -    Stationary Wavelet Transform tools
%   TimeFrequency/       -    Time-Frequency Distribution Tools
%   Utilities/           -    System-Wide scripting utilities
%
%   Papers/              -    Scripts recreating figures in published articles:
%
%     /Adapt             -    figures for ``Adapting to Unknown Smoothness via
%                                            Wavelet Shrinkage''
%     /Asymp             -    figures for ``Wavelet Shrinkage: Asymptopia?''
%     /Blocky            -    figures for ``Smooth Wavelet Decompositions with
%                                            Blocky Coefficient Kernels''
%     /Correl            -    figures for ``Wavelet Threshold Estimators for
%                                            Data with Correlated Noise''
%     /Ideal             -    figures for ``Ideal Spatial Adaptation via
%                                            Wavelet Shrinkage''
%     /MinEntSeg         -    figures for ``On Minimum Entropy Segmentation''
%     /MIPT              -    figures for ``Nonlinear Wavelet Transforms
%                                            based on Median-Interpolation''
%     /RiskAnalysis      -    figures for ``Exact Risk Analysis of Wavelet Regression''
%     /ShortCourse       -    figures for ``Nonlinear Wavelet Methods for Recovery
%                                            of Signals, Densities, Spectra and
%                                            Images from Incomplete and Noisy Data''
%     /CSpinCycle        -    figures for ``Translation-Invariant De-Noising''
%     /Tour              -    figures for ``Wavelet Shrinkage and W.V.D.
%                                            -- a Ten-Minute Tour''
%     /VillardDeLans     -    figures for ``WaveLab and Reproducible Research''
%
%   Workouts/            -    Scripts giving WaveLab features a workout
%
%     /BestOrthoBasis    -    Best Basis workout
%     /MatchingPursuit   -    Matching Pursuit workout
%     /MultiFractal      -    Continuous Wavelet Transform workout
%     /Toons             -    Cartoon Guide to Wavelets
%
% Other
```

```
%
%   README                 -    General information blurb
%
```

Make an actual directory listing to see if your hard disk actually has these files and subdirectories.

2.4 Pathnames

Matlab can automatically, at startup time, make all the Wavelab software available. The script WavePath.m is provided as part of Wavelab to enable this feature. It should be invoked from the user's Startup.m file.

PC Startup.m is located in the `matlab\local` directory on MS-Windows. Insert the line `WavePath` in that file, and put a copy of WavePath.m in that directory.

Mac Startup.m may be located anywhere inside the `Matlab` directory on Macintosh. Insert the line `WavePath` in that file. Since Wavelab contains a Startup.m file, if you have no other Startup.m file, there is nothing to do once Wavelab is installed.

Unix This file is located in the `matlab` subdirectory of your home directory on UNIX. If you don't have such a subdirectory, use `mkdir ~/matlab` to make one. Create a file named Startup.m and insert the line `WavePath` in that file. Then put a copy of WavePath.m in that directory.

2.5 Checklists

To reinforce the above points, we furnish here step-by-step installation checklists.

2.5.1 UNIX Checklist

1. Binary Download the archive to the directory you want Wavelab to reside.
2. Uncompress the archive: `Wavelab850.zip`
3. Decide where you want the Wavelab directory to reside. It will have a number of subdirectories and occupy at least 2MB disk space.
4. After you decompress the file for your machine, you should have the following directory structure.
   Wavelab850
   Wavelab850/ Biorthogonal
   Wavelab850/ Books
   Wavelab850/ Books / WaveTour
   Wavelab850/ Browsers
   Wavelab850/ Browsers / One-D
   Wavelab850/ Browsers / WaveTour
   Wavelab850/ Continuous
   Wavelab850/ Datasets
   Wavelab850/ DeNoising
   Wavelab850/ Documentation
   Wavelab850/ FastAlgorithms
   Wavelab850/ Fractals
   Wavelab850/ Interpolating
   Wavelab850/ Invariant
   Wavelab850/ Median
   Wavelab850/ Median / HigherDegree
   Wavelab850/ Mexsource

Wavelab850/ Meyer
Wavelab850/ Orthogonal
Wavelab850/ Packets
Wavelab850/ Packets / One-D
Wavelab850/ Packets / Two-D
Wavelab850/ Papers
Wavelab850/ Papers / Adapt
Wavelab850/ Papers / Asymp
Wavelab850/ Papers / Blocky
Wavelab850/ Papers / Correl
Wavelab850/ Papers / Ideal
Wavelab850/ Papers / MinEntSeg
Wavelab850/ Papers / MIPT
Wavelab850/ Papers / RiskAnalysis
Wavelab850/ Papers / ShortCourse
Wavelab850/ Papers / SpinCycle
Wavelab850/ Papers / Tour
Wavelab850/ Papers / VillardDelans
Wavelab850/ Pursuit
Wavelab850/ TimeFrequency
Wavelab850/ Utilities
Wavelab850/ Workouts
Wavelab850/ Workouts / BestOrthoBasis
Wavelab850/ Workouts / MatchingPursuit
Wavelab850/ Workouts / MultiFractal
Wavelab850/ Workouts / Toons

5. Copy all the Wavelab files from the place you put the original Wavelab archive (for example /tmp) to their final destination, for example in your home directory ũser/matlab/Wavelab850.

6. Lunch Matlab; In Matlab set the current path to matlabroot/toolbox/Wavelab850 or alternatively copy the file WavePath.m from < MatlabToolboxPath > /Wavelab850 to <MatlabToolboxPath> /local

7. Run WavePath.m; If the default pathname is not right the program will ask you to enter the correct path.

8. Type `installMEX` to compile and install the .mex files.

Trouble-Shooting UNIX: Compare the output of `ls -r WaveLab850` with `Documentation` to see if you have all the files. Compare the output of the Matlab command `path` with the list above to see if you have all the directories in your path.

2.5.2 Macintosh Checklist

To follow these instructions you will need:

(1) A Macintosh running MacOS 10.3 or later.

(2) A program which can unzip .zip file.

(3) Matlab 6.x or 7 for Mac.

(4) In certain special circumstances, you may need to have the C compiler to compile Mex files.

Steps:

1. Binary Download the file `Wavelab850.zip` to your Macintosh.
2. Extract the archive to the Toolbox folder of your Matlab folder. After you extract the file you should have the following subdirectory structure:
   Wavelab850
   Wavelab850/ Biorthogonal
   Wavelab850/ Books
   Wavelab850/ Books / WaveTour
   Wavelab850/ Browsers
   Wavelab850/ Browsers / One-D
   Wavelab850/ Browsers / WaveTour
   Wavelab850/ Continuous
   Wavelab850/ Datasets
   Wavelab850/ DeNoising
   Wavelab850/ Documentation
   Wavelab850/ FastAlgorithms
   Wavelab850/ Fractals
   Wavelab850/ Interpolating
   Wavelab850/ Invariant
   Wavelab850/ Median
   Wavelab850/ Median / HigherDegree
   Wavelab850/ Mexsource
   Wavelab850/ Meyer
   Wavelab850/ Orthogonal
   Wavelab850/ Packets
   Wavelab850/ Packets / One-D
   Wavelab850/ Packets / Two-D
   Wavelab850/ Papers
   Wavelab850/ Papers / Adapt
   Wavelab850/ Papers / Asymp
   Wavelab850/ Papers / Blocky
   Wavelab850/ Papers / Correl
   Wavelab850/ Papers / Ideal
   Wavelab850/ Papers / MinEntSeg
   Wavelab850/ Papers / MIPT
   Wavelab850/ Papers / RiskAnalysis
   Wavelab850/ Papers / ShortCourse
   Wavelab850/ Papers / SpinCycle
   Wavelab850/ Papers / Tour
   Wavelab850/ Papers / VillardDelans
   Wavelab850/ Pursuit
   Wavelab850/ TimeFrequency
   Wavelab850/ Utilities
   Wavelab850/ Workouts
   Wavelab850/ Workouts / BestOrthoBasis
   Wavelab850/ Workouts / MatchingPursuit
   Wavelab850/ Workouts / MultiFractal
   Wavelab850/ Workouts / Toons
3. Lunch Matlab; In Matlab set the current path to matlabroot/toolbox/Wavelab850 or alternatively copy the file WavePath.m from < MatlabToolboxPath >/Wavelab850 to <MatlabToolboxPath>/local
4. Run WavePath.m at the command prompt to start Wavelab. You will see a "Welcome to Wavelab" message as shown in the section Success below.

Note:

1. If you want to automatically load Wavelab850 upon the start-up copy the file WavePath.m from the folder Wavelab850 to the folder Matlab/Toolbox/local. Determine if you have any file named startup.m besides the one that is in Wavelab850 directory. If you don't, go to step 3.

2. if you have Startup.m , then copy the contents of WavePath.m into this file.

3. If you don't have any Startup.m , then copy the file Startup.m from Wavelab850 directory to <MatlabToolboxPath>/local

2.5.3 PC Checklist

To follow these instructions you will need:

(1) An Intel Platform running Windows 2000 or XP.

(2) A program such as Winzip which can unzip .zip file.

(3) Matlab 6.x or 7 for Windows.

(4) In certain special circumstances, you may need to have the C compiler to compile Mex files.

1. Binary Download the file `Wavelab850.zip` to your PC.

2. Extract the archive to the Toolbox folder of your Matlab folder. After you extract the file you should have the following subdirectory structure:
   Wavelab850
   Wavelab850\Biorthogonal
   Wavelab850\Books
   Wavelab850\Books \WaveTour
   Wavelab850\Browsers
   Wavelab850\Browsers \One-D
   Wavelab850\Browsers \WaveTour
   Wavelab850\Continuous
   Wavelab850\Datasets
   Wavelab850\DeNoising
   Wavelab850\Documentation
   Wavelab850\FastAlgorithms
   Wavelab850\Fractals
   Wavelab850\Interpolating
   Wavelab850\Invariant
   Wavelab850\Median
   Wavelab850\Median \HigherDegree
   Wavelab850\Mexsource
   Wavelab850\Meyer
   Wavelab850\Orthogonal
   Wavelab850\Packets
   Wavelab850\Packets \One-D
   Wavelab850\Packets \Two-D
   Wavelab850\Papers
   Wavelab850\Papers \Adapt
   Wavelab850\Papers \Asymp
   Wavelab850\Papers \Blocky
   Wavelab850\Papers \Correl

Wavelab850\Papers \Ideal
Wavelab850\Papers \MinEntSeg
Wavelab850\Papers \MIPT
Wavelab850\Papers \RiskAnalysis
Wavelab850\Papers \ShortCourse
Wavelab850\Papers \SpinCycle
Wavelab850\Papers \Tour
Wavelab850\Papers \VillardDelans
Wavelab850\Pursuit
Wavelab850\TimeFrequency
Wavelab850\Utilities
Wavelab850\Workouts
Wavelab850\Workouts \BestOrthoBasis
Wavelab850\Workouts \MatchingPursuit
Wavelab850\Workouts \MultiFractal
Wavelab850\Workouts \Toons

3. Lunch Matlab; In Matlab set the current path to matlabroot\toolbox \Wavelab850 or alternatively copy the file WavePath.m from < MatlabToolboxPath > \Wavelab850 to <MatlabToolboxPath> \local

4. Run WavePath.m at the command prompt to start Wavelab. You will see a "Welcome to Wavelab" message as shown in the section Success below.

Note:

1. If you want to automatically load Wavelab850 upon the start-up copy the file WavePath.m from the folder Wavelab850 to the folder Matlab \Toolbox \local. Determine if you have any file named startup.m besides the one that is in Wavelab850 directory. If you don't go to step 3.

2. if you have Startup.m then copy the contents of WavePath.m into this file.

3. If you don't have any Startup.m then copy the file Startup.m from Wavelab850 directory to <MatlabToolboxPath> \local

2.6 Success

When you have a successful installation, you should see something like the following when you invoke Matlab:

```
Welcome to WaveLab v 850 Setting Global Variables

WAVELABPATH =

C:\MATLAB6p5\toolbox\Wavelab850\

Pathnames Successfully Set global WAVELABPATH =
"C:\MATLAB6p5\toolbox\Wavelab850\" global PATHNAMESEPARATOR = "\";
global MATLABVERSION = 6.5 global PREFERIMAGEGRAPHICS = 1 WaveLab v
850 Setup Complete

Available Demos - Figures from the following papers:
  AdaptDemo  - ‘‘Adapting to Unknown Smoothness via Wavelet Shrinkage’’
  AsympDemo  - ‘‘Wavelet Shrinkage: Asymptopia?’’
```

```
  BlockyDemo - ``Smooth Wavelet Decompositions with Blocky Coefficient Kernels''
  CorrelDemo - ``Wavelet Threshold Estimators for Data with Correlated Noise''
  IdealDemo  - ``Ideal Spatial Adaptation via Wavelet Shrinkage''
  MESDemo    - ``Minimum Entropy Segmentation''
  MIPTDemo  - ``Nonlinear Wavelet Transforms based on Median-Interpolaton''
  RiskDemo  - ``Exact Risk Analysis of Wavelet Regression''
  SCDemo     - ``Nonlinear Wavelet Methods for Recovery of Signals, Densities
                 and Spectra from Indirect and Noisy Data''
  CSpinDemo   - ``Translation-Invariant De-Noising''
  TourDemo   - ``Wavelet Shrinkage and W.V.D. -- A Ten-Minute Tour''
  VdLDemo    - ``WaveLab and Reproducible Research''

Available Workouts:
  BBWorkout  - Workouts for Best Basis
  MPWorkout  - Workouts for Matching Pursuit
  MultiFrac  - Workouts for Continuous Wavelet Transform
  Toons      - The Cartoon Guide to Wavelets

Available Book(s):
  WaveTour   - ``WaveLet Tour of Signal Processing''
```

3 Getting Started

There are several ways to get started with Wavelab. First, you can snoop around the directory structure to see what's there. Second, you can try running some of the demos to see what they do. Third, you can try the Browser to do some point and-click wavelet operations on "canned" signals. Fourth, you can run the Toons to get individual figures.

3.1 Snooping

If you just snoop around in the Wavelab file structure, you will notice many directories and a great range of different information about the system itself and what it can do. We list here some basic facts.

3.1.1 Contents Files

Each directory has a `Contents.m` file, which explains the contents and purpose of that directory. The directory `Orthogonal` contains the central wavelet transform tools; its `Contents.m` file looks as follows:

```
% Orthogonal:Contents  v.850 -- Orthogonal Wavelet Transform Tools
%
%   The routines in this directory perform periodic- and boundary-corrected
% wavelet analysis of 1-d and 2-d signals. The main tools for all-purpose
% use are FWT_PO and IWT_PO.
%
%          Wavelet Transforms
%
% FWT_PO            -  Forward Wavelet Transform, Periodized, Orthogonal
% IWT_PO            -  Inverse Wavelet Transform, Periodized, Orthogonal
%
% FWT_CDJV          -  Forward Wavelet Transform, Boundary-Corrected
% IWT_CDJV          -  Inverse Wavelet Transform, Boundary-Corrected
%
% FWT2_PO           -  Forward Wavelet Transform, 2-d MRA, Periodized,
```

```
%                          Orthogonal
% IWT2_PO           -  Inverse Wavelet Transform, 2-d MRA, Periodized,
%                          Orthogonal
%
% FTWT2_PO          -  Forward Wavelet Transform, 2-d Tensor, Periodized,
%                          Orthogonal
% ITWT2_PO          -  Inverse Wavelet Transform, 2-d Tensor, Periodized,
%                          Orthogonal
%
%          Wavelet Transform Displays
%
% ContourMultiRes   -  Multi-Resolution Mesh Display of 1-d Wavelet Transform
% DisplayMutltiRes  -  Mesh, Contour or Image Plot of Multi-Resolution
% DisplayWaveCoeff  -  Mesh, Contour or Image Plot of Wavelet Coefficients
% PlotMultiRes      -  Display Mallat-style Multiresolution Decomposition
% PlotWaveCoeff     -  Spike Plot of Wavelet Coefficients
%
%          Filter and Wavelet Generators
%
% MakeCDJVFilter    -  Generate Filters for CDJV Boundary-Corrected Transform
% MakeONFilter      -  Generate Filters for Daubechies, Coiflets, Symmlets, Haarlets
% MakeWavelet       -  Make periodized orthogonal wavelet
% Make2dWavelet     -  Make 2-d wavelet
%
%          Two-Scale Operators
%
% UpDyadHi          -  Upsampling Hi Pass operator   (used in IWT_PO)
% UpDyadLo          -  Upsampling Lo Pass operator   (used in IWT_PO)
% DownDyadHi        -  Downsampling Hi Pass operator (used in FWT_PO)
% DownDyadLo        -  Downsampling Lo Pass operator (used in FWT_PO)
% CDJVDyadDown      -  Downsampling operator         (used in FWT_CDJV)
% CDJVDyadUp        -  Upsampling operator           (used in IWT_CDJV)
%
%          Utilities
%
% aconv             -  Filtering by periodic convolution of x with
%                        time reverse of f
% iconv             -  Filtering by periodic convolution of x with f
% dyad              -  Access entire j-th dyad of 1-d transform
% dyad2ix           -  Convert (j,k) index to linear index
% dyadlength        -  Length and Dyadic Length of 1-d array
% quad2ix           -  Convert (j,k) index to linear index
% quadlength        -  Length and Dyadic Length of 2-d array
% rshift            -  Circulant right shift
% lshift            -  Circulant left shift
% MirrorFilt        -  Apply (-1)^t modulation
% reverse           -  Reverse order of samples
% UpSampleN         -  Interpolate zeros between samples
% PlotSpikes        -  Plot an array as as spikes on baseline
% UpDyadHi          -  Hi-Pass Upsampling operator; periodized
% UpDyadLo          -  Lo-Pass Upsampling operator; periodized
%
```

3.1.2 Help for Functions

Each function in Wavelab has help documentation. For example, `FWT_PO` is a basic wavelet transform routine. If you are in Matlab and type `help FWT_PO`, Matlab will type out the following documentation:

```
 FWT_PO -- Forward Wavelet Transform (periodized, orthogonal)
  Usage
    wc = FWT_PO(x,L,qmf)
  Inputs
    x     1-d signal; length(x) = 2^J
    L     Coarsest Level of V_0;  L << J
    qmf   quadrature mirror filter (orthonormal)
  Outputs
    wc    1-d wavelet transform of x.

  Description
    1. qmf filter may be obtained from MakeONFilter
    2. usually, length(qmf) < 2^(L+1)
    3. To reconstruct use IWT_PO

  See Also
    IWT_PO, MakeONFilter
```

3.1.3 Source Browsing

All the algorithms in Wavelab are available for inspection – even those that are actually implemented by fast complied C code as .mex files. For example, if you are in Matlab and type `FWT_PO` you get the following documentation:

```
function wcoef = FWT_PO(x,L,qmf)
% FWT_PO -- Forward Wavelet Transform (periodized, orthogonal)
%  Usage
%    wc = FWT_PO(x,L,qmf)
%  Inputs
%    x     1-d signal; length(x) = 2^J
%    L     Coarsest Level of V_0;  L << J
%    qmf   quadrature mirror filter (orthonormal)
%  Outputs
%    wc    1-d wavelet transform of x.
%
%  Description
%    1. qmf filter may be obtained from MakeONFilter
%    2. usually, length(qmf) < 2^(L+1)
%    3. To reconstruct use IWT_PO
%
%  See Also
%    IWT_PO, MakeONFilter
%
  [n,J] = dyadlength(x) ;
  wcoef = zeros(1,n) ;
  beta = ShapeAsRow(x);  %take samples at finest scale as beta-coeffts
  for j=J-1:-1:L
       alfa = DownDyadHi(beta,qmf);
       wcoef(dyad(j)) = alfa;
       beta = DownDyadLo(beta,qmf) ;
```

```
  end
  wcoef(1:(2^L)) = beta;
  wcoef = ShapeLike(wcoef,x);

%
% Copyright (c) 1993. Iain M. Johnstone
%
```

Notice that the source contains information about the author and date of compilation, as well as copyright, of the routine. Also, the help information is built in as the first thing following the function header. Notice also that the wavelet transform routine depends on other routines, such as `DownDyadHi` and `dyad`, which are also part of Wavelab and can also be inspected at source level.

3.1.4 Documentation Directory

The Wavelab system also has extensive built-in documentation about the system itself. If you look in the directory `Documentation`, you will find several files of general interest:

```
WlAlphaHelpListing      -  all help files arranged by function name
WLAlphaSynopsisListing  -  one-line synopses arranged by function name
WLContentsListing       -  all Contents.m files
WLFiles                 -  listing of all WaveLab files arranged by directory
WLHelpHeaders.m         -  listing of all first lines of help headers
WLHelpListing           -  all help files arranged by directory
```

Two extracts:

WLAlphaSynopsisListing: This file is helpful for quick reference when writing code – when you know what function to use, and can't remember exactly the calling sequence. Part of the file near the letter "C":

```
%    sqtree = Calc2dStatTree(TFType,img,D,TFPar,ent[,EntPar])
%    heights = Calc2dTreeHeight(stree,D)
%    Ent = CalcEntropy(object,ent[,par])
%    stree = CalcStatTree(pkt,ent[,par])
%    [maxheight, cost] = CalcTreeHeight(stree,D)
%    CalcWPLocation(d,b,k,qmf,n)
%    blo = CDJVDyadDown(bhi,F,LEF,REF)
%    bhi = CDJVDyadUp(blo,F,LEF,REF)
%    bestlev = CompareStdBases(stree,D)
%    PlotMultiRes(wc,L,scal,qmf)
%    [bb,stats,coef] = CP2dTour(img,MaxDeep,titlestr)
%    sig = CPAtomicSynthesis(atoms,cp,bell)
%    [clean,bb,st] = CPDeNoise(x,D,bell)
%    dcp = CPImpulse(cp,d,b,k,bell)
%    atomic = CPPursuit(x,D,bell,natom,frac,show)
%    atomic = CPPursuitBF(x,D,bell,natom,frac,show)
%    [cp, btree] = CPPursuitTour(Format,x,D,ball[,natom,title])
```

WLHelpHeaders.m: This lists the first line of each help file in the system. It is handy for browsing when you don't know what you need and are searching for a handle. Here is a segment dealing with the `Utilities` directory.

```
%*** WaveLab:Utilities ***
% Utilities:Contents v.850: Utilties for Writing Scripts
```

```
% AppendTitle -- Utility to Build Title String
% AutoImage -- image display of object assuming arbitrary values
% CutDyad -- Truncate signal to Dyadic length
% GrayImage -- standard gray-scale image display
% HitAnyKey -- Tool for pausing in scripts
% LockAxes -- Version-independent axis command
% PadDyad - Zero-fill signal to Dyadic length
% MakeTiledfigures -- Tile the screen with figures
% RegisterPlot -- add legend with file name, date, flag
% ShapeAsRow -- Make signal a row vector
% ShapeLike -- Make 1-d signal with given shape
% UnlockAxes -- version-independent axis command
% versaplot -- version-independent plot routine
% WaitUntil -- Burn up CPU cycles until sec seconds elapse from oldclock
% WhiteNoise -- version-independent white noise generator
```

3.1.5 Dataset Documentation

Datasets are also documented. If you look in the directory `Datasets`, you will find that each dataset (.raw or .asc) is accompanied by a .doc file. The dataset `daubechies.raw` is accompanied by the file `daubechies.doc`, which contains the following:

```
daubechies.raw -- Gray-scale image of Ingrid Daubechies

Access
   Ingrid = ReadImage('Daubechies');

Size
   256 by 256

Gray Levels
   256

Description
   Ingrid Daubechies is a very active researcher in the field of
   wavelet analysis and author of the book "Ten Lectures on Wavelets",
   SIAM, 1992.  She is inventor of smooth orthonormal wavelets
   of compact support.

Source
   Photograph of Ingrid Daubechies at the 1993 AMS winter meetings
   in San Antonio, Texas. Taken by David Donoho with Canon XapShot
   video still frame camera.
```

3.2 Demos

After browsing around to see what files Wavelab contains, it's time to see what Wavelab can do!

The subdirectory `Wavelab850/Papers` itself contains several subdirectories; each one of these contains scripts that were used to produce figures in our published articles.

As new articles are written by members of our group, we will add new subdirectories.

Each subdirectory contains a "demo" file (e.g. `SCDemo.m` in directory `ShortCourse`, `TourDemo.m` in directory `Tour`) and so on. This file allows you to reproduce the figures in the corresponding article.

When you invoke that file in Matlab by typing its name (without the .m extension), a new window will appear on the screen. If you mouse-click on the push button `Show All Figures` you will see, in sequence, each figure in the corresponding article. As each figure appears in

Matlab's figure window, the command window will contain narrative explaining what you see in the figure window.

3.2.1 Short Course Demo

For example, in `SCDemo`, one gets the figures from a short course presented at the American Mathematical Society. When one runs `SCDemo`, the following banner appears in the command window:

```
SCIntro -- Info for SCDemo

  The .m files in this firectory can reproduce the figures
  in the article

      Nonlinear wavelet methods for recovery of
        Signals, Densities and Spectra from
              Indirect and Noisy Data
                        by
                  David L. Donoho

   to appear in Proc. Symp. Appl. Math. 1993 Edited by
   Ingrid Daubechies, American Math. Soc., Providence RI

  These figures illustrate the application of thresholding in the
  wavelet, wavelet packet and cosine packet domains to recovery
  of objects from noisy and incomplete data. References are given to
  recent work of Donoho, Johnstone, Kerkyacharian, Picard, Gao and
  researchers in other groups.
```

All in all, this Demo consists of 29 figures. Here is a sort of table of contents, made by extracting from the file `Documentation/WLHelpHeaders.m`:

```
scfig01: Short Course 01 -- De-Noising of NMR Signal
scfig02: Short Course 02 -- Noisy Deconvolution
scfig03: Short Course 03 -- Comparison of Wavelet and Packet DeNoising
scfig04: Short Course 04 -- Four Spatially Inhomogenoeus Signals
scfig05: Short Course 05 -- Noisy Versions of Four Signals
scfig06: Short Course 06 -- Wavelet Shrinkage of the Four Noisy Signals
scfig07: Short Course 07 -- AutoSpline Reconstructions from Noisy Data
scfig08: Short Course 08 -- AutoTrunc Truncated Fourier Reconstructions
scfig09: Short Course 09 -- WaveShrink of object yBlocks in Haar Basis
scfig10: Short Course 10 -- Comparing Compression Abilities
scfig11: Short Course 11 -- Compare Wavelet and DCT Partial Reconstructions
scfig12: Short Course 12 -- DeNoising a 2-d object
scfig13: Short Course 13 -- Smoothing Counts data by square roots -- ESCA data
scfig14: Short Course 14 -- Estimating Time Series Spectrum
scfig15: Short Course 15 -- Noisy Differentiation by WVD
scfig16: Short Course 16 -- Noisy Differentiation by WVD in Wavelet Domain
scfig17: Short Course 17 -- Noisy Differentiation by Ideal Fourier Damping
scfig18: Short Course 18 -- Depict Deconvolution in Wavelet Domain
scfig19: Short Course 19 -- Display Vaguelette Kernels
scfig20: Short Course 20 -- Four Time-Frequency test signals
scfig21: Short Course 21 -- Four Noisy Time-Frequency test signals
scfig22: Short Course 22 -- Wavelet Packet DeNoising
scfig23: Short Course 23 -- Compare Four DeNoising methods
```

```
scfig24: Short Course 24 -- Compare Segmented and Ordinary Refinement
scfig25: Short Course 25 -- DeNoising a segmented transform
scfig26: Short Course 26 -- Sine signal with Cauchy Contamination
scfig27: Short Course 27 -- Linear Smoothing of Cauchy Noise
scfig28: Short Course 28 -- Robust De-Noising
```

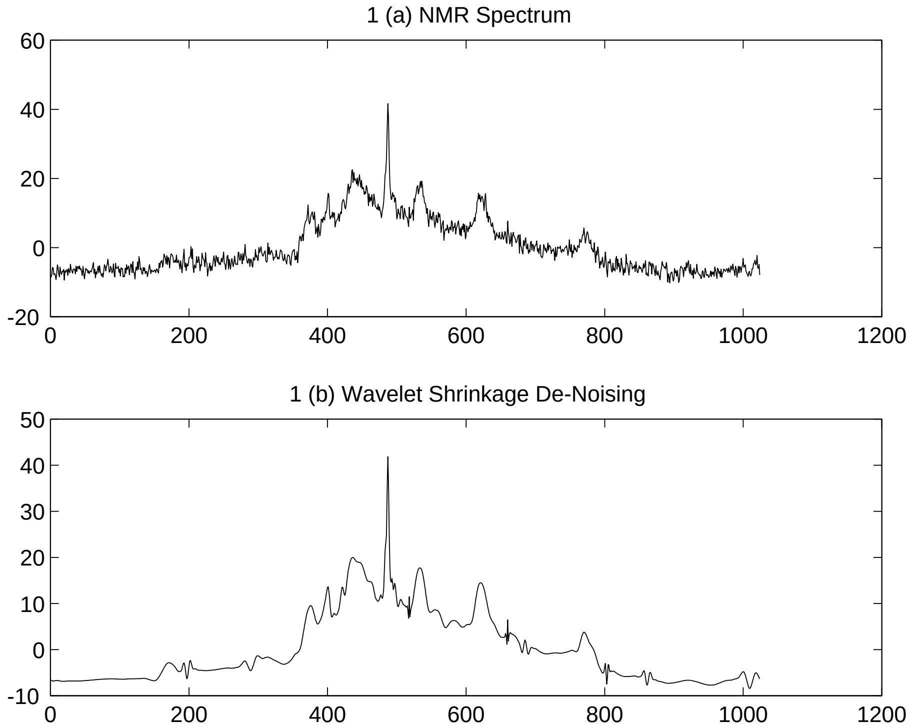


Figure 1: First Figure of Short Course

The first figure to appear is as follows:

As this figure appears, the following text appears in the command window:

```
scfig01: Short Course 01 -- De-Noising of NMR Signal

Here a noisy NMR signal is denoised by a simple three-step
recipe:
        1.  Transform to Wavelet Domain.
        2.  Apply thresholding to set coeficients
            ``at the noise level'' to zero
        3.  Return to the original domain.

Evidently, the noise is suppressed without broadening the peaks.
More traditional smoothing methods either suppress the noise and
broaden peaks or don't suppress the noise and leave peaks narrow.

These data were kindly supplied by Chris Raphael, an NSF postdoctoral
fellow in the Statistics Department at Stanford.
```

Note that if you ever become interested in *how* a certain effect is achieved, this is available simply by inspecting the code at source level. An extract from the output given by Matlab in response to `type scfig01`:

```
nmrsignal  = ReadSignal('RaphaelNMR')';
QMF8       = MakeONFilter('Symmlet',8);
scalednmr  = NormNoise(nmrsignal,QMF8);
[xh,wcoef] = WaveShrink(y,'Visu',5,QMF8);
tnmr       = 1:length(nmrsignal);
```

```
%
    clg;
    versaplot(211,tnmr,y, [],' 1 (a) NMR Spectrum',[],[])
    versaplot(212,tnmr,xh,[],' 1 (b) Wavelet Shrinkage De-Noising',[],[])
```

This code fragment reads in a dataset, normalizes it, calls `WaveShrink` to de-noise it and then displays, in two panels, the result.

3.2.2 Demo Inventory

Here is an up-to-date listing of demos in version .850, and the articles to which they correspond:

```
AdaptDemo    ``Adapting to Unknown Smoothness via Wavelet Shrinkage''
AsympDemo    ``Wavelet Shrinkage: Asymptopia?''
BlockyDemo   ``Smooth Wavelet Decompositions with Blocky Coefficient Kernels''
CorrelDemo   ``Wavelet Threshold Estimators for Data with Correlated Noise''
IdealDemo    ``Ideal Spatial Adaptation via Wavelet Shrinkage''
MESDemo      ``On Minimum Entropy Segmentation''
MIPTDemo     ``Nonlinear Wavelet Transforms based on Median-Interpolaton''
RiskDemo     ``Exact Risk Analysis of Wavelet Regression''
SCDemo       ``Nonlinear Wavelet Methods for Recovery of Signals, Densities,
               Spectra and Images from Incomplete and Noisy Data''
CSpinDemo    ``Translation-Invariant De-Noising''
TourDemo     ``Wavelet Shrinkage and W.V.D. -- A Ten-Minute Tour''
VdLDemo      ``WaveLab and Reproducible Research''
```

3.3 Browser

Now you might be interested in *interacting* with wavelets a bit. For this purpose, consider the 1-d signal browser `WLBrowser`.

The `Contents.m` file for directory `Browsers/One-D` says: "The routines in this directory implement a point-and-click browser that allows the user to select signals, select wavelet, wavelet packet, discrete cosine transforms, etc. Some of the options, including WTCompress, WPCompress and CPCompress, contain other possibilities for interaction, such as interactive wavelet thresholding – changing a threshold interactively and watching the effects upon reconstruction."

Invoke this browser by simply typing `WLBrowser` at the Matlab prompt. After the browser is initialized, you will see four new figure windows tiling the screen, and you will see several menu items at the top of the window at the upper left of the screen:

```
File Edit Window *Data *Signal *Transform *Options *Action
```

The starred items are new items installed by the browser. If you click the mouse button while pointing at the `*Data` item, a pull-down menu will appear with the names of datasets which can be accessed by the Browser.

At the same time, documentary text will scroll by in the command window:

```
**********************************************
*           WaveLab  Browser                *
**********************************************

This WaveLab MATLAB program lets you try out standard
Wavelet Analysis techniques on standard or synthetic data sets.

Here is a summary of how to use the menus:

---------------------------------------------------
Data:    Load data from WaveLab's standard data sets:
         (Caruso, Laser, Sunspots, Seismic, ESCA, HochNMR, RaphaelNMR)

---------------------------------------------------
Signals:  Use a built-in artificial signal
         (Bumps, Blocks, LinChirp, TwoChirp, ...)

Xforms:
         Sqrt     - square root transformation
         Log      - log transformation
         Anscombe - square root transformation
         Wahba    - log transformation of periodogram
         Pgram    - periodogram
         Add Noise- apply the currently defined noise to the data & plot
         Normalize- apply the WaveLab routine 'NormNoise' to the data and plot it.
---------------------------------------------------
Actions:
         Plot_WT    - Plot Wavelet Transforms
         Plot_MRA   - Plot Multi-Resolution Analysis
         Plot_DCT   - Plot Discrete Cosine Transforms
         WPTour     - Wavelet Packet & Best Basis Analysis
         CPTour     - Cosine Packets & Best Basis Analysis
         WTCompress - Wavelet Compression
         WTDeNoise  - Wavelet DeNoising
         WPCompress - Wavelet Packet Compression
         CPCompress - Cosine Packet Compression
---------------------------------------------------
Params:
         Wavelet      - Select the wavelet for the wavelet transform
         Bell         - Select the bell to be used in  cosine packets
         Nonlinearity - type of thresholding to use in denoising
         Threshold Selector - method of selecting the threshold
         Noise Type   - type of noise to use when adding noise
         Noise Level  - amplitude of noise when adding noise
         Signal Length- signal length when fabricating artificial signal
---------------------------------------------------
KEEP THIS TEXT WINDOW HANDY AND YOU WILL SEE COMMENTS
AND THE MATLAB COMMANDS DISPLAYED AS THEY ARE EXECUTING,
FROM TIME TIME TO TIME YOU WILL HAVE TO ENTER DATA HERE.
---------------------------------------------------
```

This gives a sketch of the functions available in the Browser.

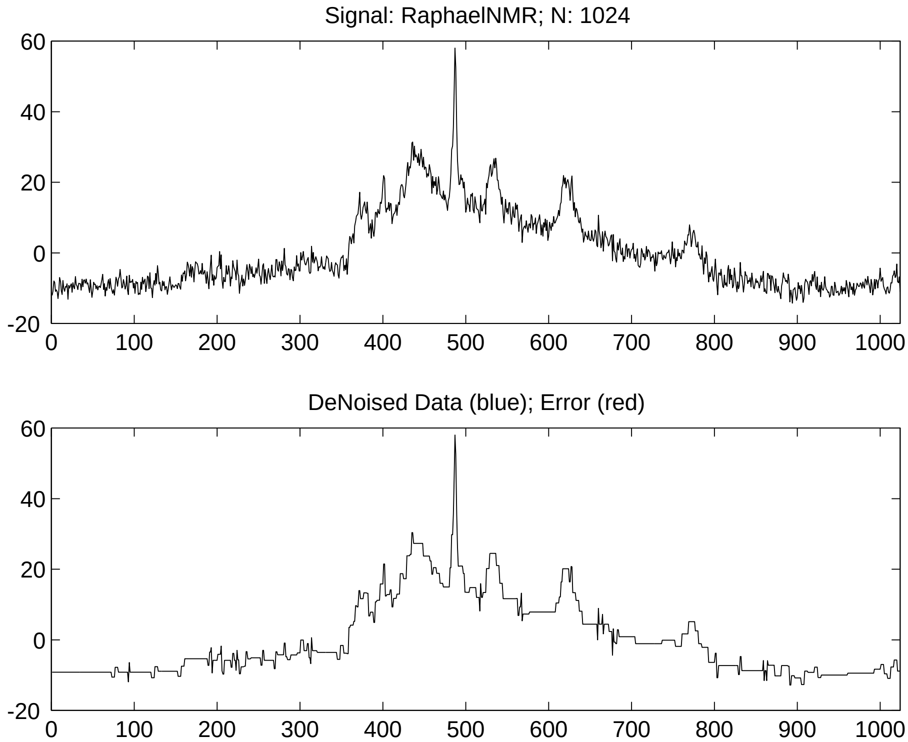


Figure 2: Denoising NMR Data

3.3.1 Example: De-Noising

Try pulling down the `*Data` menu, and selecting `RaphaelNMR`. In the upper left window of the screen, you will then see a display of some noisy NMR data; these are the same data that appeared earlier, in our example `scfig01`.

The window has a scroll bar at the bottom and several clickable buttons. Those have little use in this example.

In the command window, documentation about the data will scroll by.

Hold down the mouse button while pointing at the `*Actions` item; a pull-down menu will appear, with the names of actions. Try selecting `WTDeNoise`. In the '#3' window at the lower left of the screen, you will see a display of the wavelet transform of the noisy data, and a display of the transform after thresholding.

In the '#2' window at the upper right of the screen, you will see a display of the noisy data, and of the inverse transform of the thresholded coefficients. The display shows that a considerable amount of noise has been removed:

The '#4' window at the lower right displays the power spectra of the data and the reconstruction.

The blocky appearance of the bottom pane in window '#2' is due to the default wavelet used by the system – the Haar Wavelet. You can change the default wavelet by pulling down the `*Params` menu and selecting `Wavelet>Symmlet`. That will generate a display comparable to that of `scfig01`.

3.3.2 Example: Thresholding Caruso

Try pulling down the `*Data` menu, and selecting `Caruso`. In the upper left window of the screen, you will then see a display of some data obtained by digitizing a Caruso recording (Figure 3). The dataset, as loaded by the Browser, has 32,768 samples. The window displays a segment of length 4096.

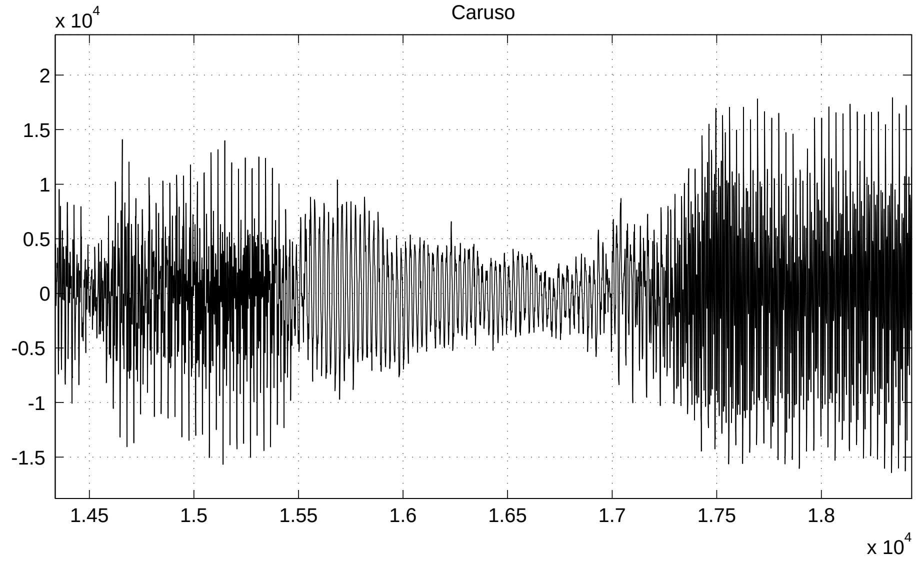


Figure 3: Caruso Signal

The window has a scroll bar at the bottom and several clickable buttons. If you click `Play`, and you are on a machine that supports sound, the visible segment is played. The speech was sampled at 8192 Hz, so you'll hear only a half a second of sound. Try clicking `*2`. This will cause the visible segment of data to double in length, to 8192 samples. If you now click `Play`, you will hear the phrase *La Bella*.

Hold down the mouse button while pointing at the `*Actions` item; a pull-down menu will appear, with the names of actions. Try selecting `WPCompress`. In the '#3' window at the lower left of the screen, you will see a display of the wavelet packet coefficients of the signal, and a display of the coefficients after thresholding (Figure 4).

In the '#2' window at the upper right of the screen, you will see a display of the signal, and of the inverse wavelet packet transform of the thresholded coefficients. The display shows that a considerable amount of noise has been removed (Figure 5).

The '#4' window at the lower right displays the sorted wavelet packet coefficients and two slider bars (Figure 6).

By adjusting the sliders, you change the height of the threshold, or equivalently, the number of coefficients discarded. Each change is followed by an automatic update of the other windows. In this way you can interactively set the threshold level and inspect the results.

3.4 Toons

The `Workouts/Toons` directory contains more than 100 scripts which exercise various features of Wavelab. These range from `toon0131` which depicts wavelets at various scales, to `toon0541-toon0548`, which compare 2-d wavelet transforms and 2-d Fourier transforms as methods of image compression, to `toon1611-toon1613` which illustrate fingerprint compression.

We again point out that one can not only view the figures that these scripts generate; one can inspect and modify the underlying code.

3.4.1 Outline of Toons

Underlying the production of Toons is a topic outline, currently located in the `Contents.m` file in the `Workouts/Toons` directory. For each topic in this outline, there are (or will ultimately be) one or several .m files which generate figures illustrating the given topic.

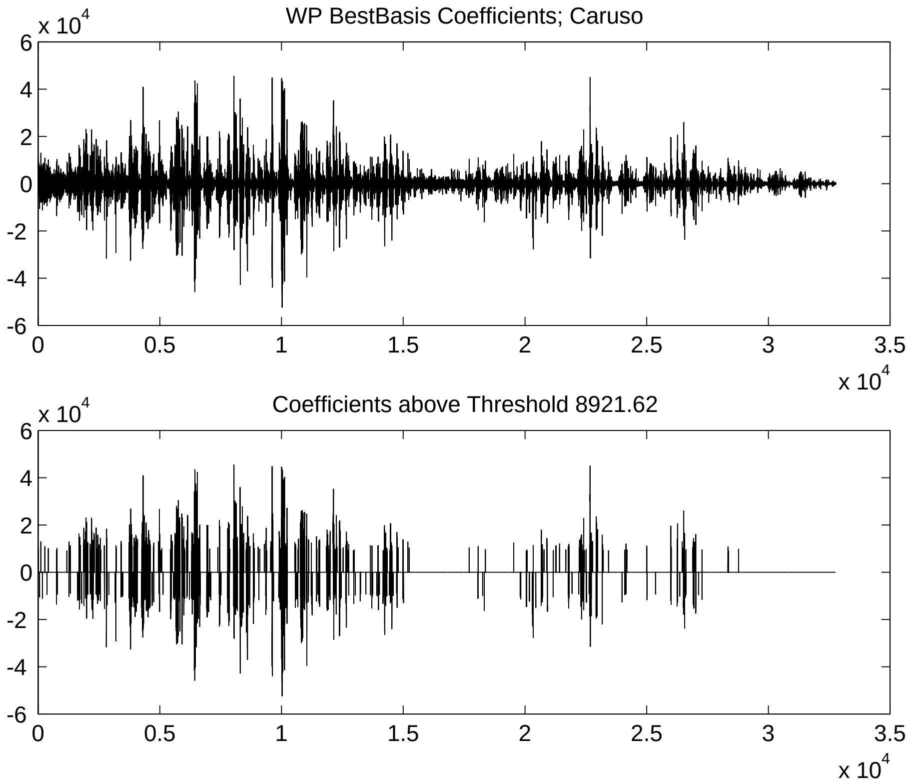


Figure 4: WP Coefficients, Caruso Signal

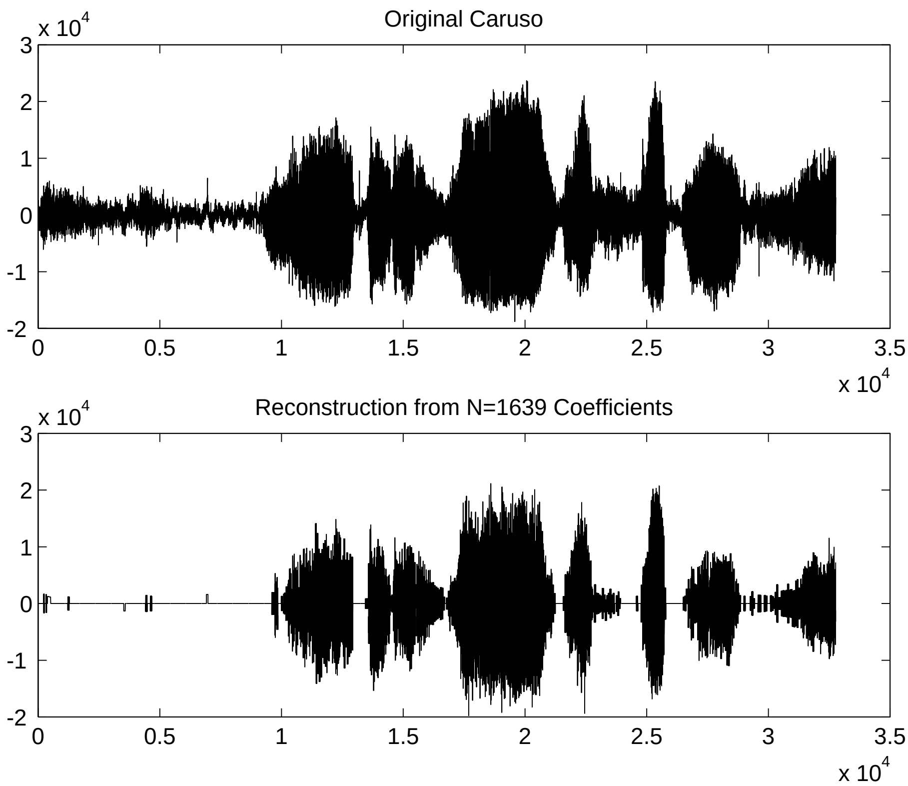


Figure 5: Denoised Caruso Signal

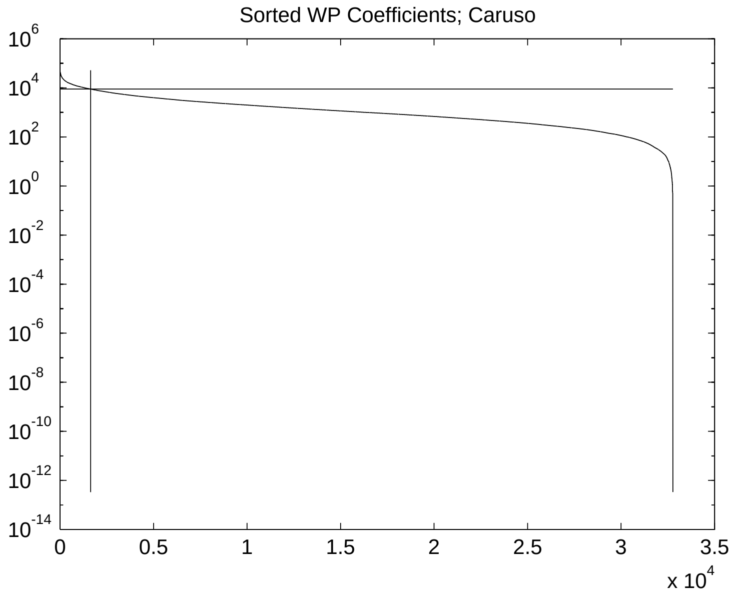


Figure 6: Threshold Selection, Caruso Signal

```
1.0 Wavelets
  1.1 Types of Wavelets
  1.2 Gender of Wavelets
  1.3 Scale Families of Wavelets
  1.4 Daubechies D4 Wavelets
  1.5 Wavelet Analysis of Functions
  1.6 MRA Analysis of Functions
  1.7 Smoothness of Wavelets
  1.8 Frequency Content of Wavelets
2.0 2-d Wavelets
  2.1 Mesh Plots of Four Wavelets
  2.2 Image Plots of Four Wavelets
  2.3 2-d Wavelet Analysis of Synthetic Objects
3.0 Wavelet Analysis
  3.1 Analysis of Smooth Signals
  3.2 Analysis of Piecewise Polynomials
  3.3 Analysis of Singularities
  3.4 Comparison of Wavelet Types
4.0 Wavelet Synthesis
  4.1 Partial Reconstructions
  4.2 Comparison of Two Wavelets
  4.3 Compression Number
  4.4 Comparison with Other Transforms
  4.5 Unconditional Basis Property
5.0 Applications
  5.1 Data Expansion
  5.2 Progressive Transmission
  5.3 Data Compression 1-d
  5.4 Data Compression 2-d Ingrid Image
```

```
  5.5 De-Noising
  5.6 Fast Algorithms

     <skipping...>

16.0 Applications of 2-d Wavelet Packet Analysis
  16.1 FBI Fingerprint
    16.1.1 Display fingerprint and basis tree
    16.1.2 First- and best-  5% reconstructions
    16.1.3 First- and best- 10% reconstructions
```

For example, in the above list, associated with topic 1.1 are four files `toon0111.m`–`toon0114` which show graphically the various types of wavelets. By extracting from the file `WLHelpHeaders.m`, we get the following:

```
% toon0111 -- Wavelet Families
% toon0112 -- Interpolating Wavelets
% toon0113 -- Average Interpolating Wavelets
% toon0114 -- Meyer Wavelets
% toon0121 -- Wavelets Come in Genders
% toon0131 -- Scale Families of Wavelets
% toon0132 -- Wavelets come at all different scales and positions
% toon0140 -- Illustrating Boundary Wavelets
% toon0141 -- Illustrating Boundary Wavelets
% toon0142 -- Illustrating Boundary Wavelets
% toon0151 -- Visualize wavelet decomposition of ramp.
% toon0152 -- Visualize wavelet decomposition of Doppler.
% toon0161 -- Visualize multi-resolution decomposition
% toon0171 -- Illustrate smoothness of wavelets
```

This shows that associated with topic 1.1 there are four figures, with topic 1.2 only one, with topic 1.4 there are three figures, etc. In general, the association of topic numbers to figures is in the scheme topic number AA.B $\rightarrow$ AABN, where N ranges from 1 to 9.

Most figures in toons are stand-alone figures, meaning they can be run independently of all other figures. However, in some cases they belong to sequences of figures that should be run all in a row. For example, `toon0541-toon0548` make up a sequence of eight figures that should be run consecutively.

3.4.2 Example: Plotting Wavelets

We now consider a few simple examples of what the toons contain. The file `toon0111` contains the following help header, accessible by typing `help toon0111`

```
% toon0111 -- Wavelet Families
%
%   Wavelet analysis begins by choosing a specific family of wavelets
%   to work with.
%
%   The family is specified by a father and a mother wavelet, and
%   these generate a basis by translation and dilation.
%
%   Here we illustrate four specific Mother wavelets
%
%       Haar           -- the first wavelet; a square-wave wavelet
%
%       Daubechies D4 -- the first continuous, compactly supported
%                         orthonormal wavelet family
```

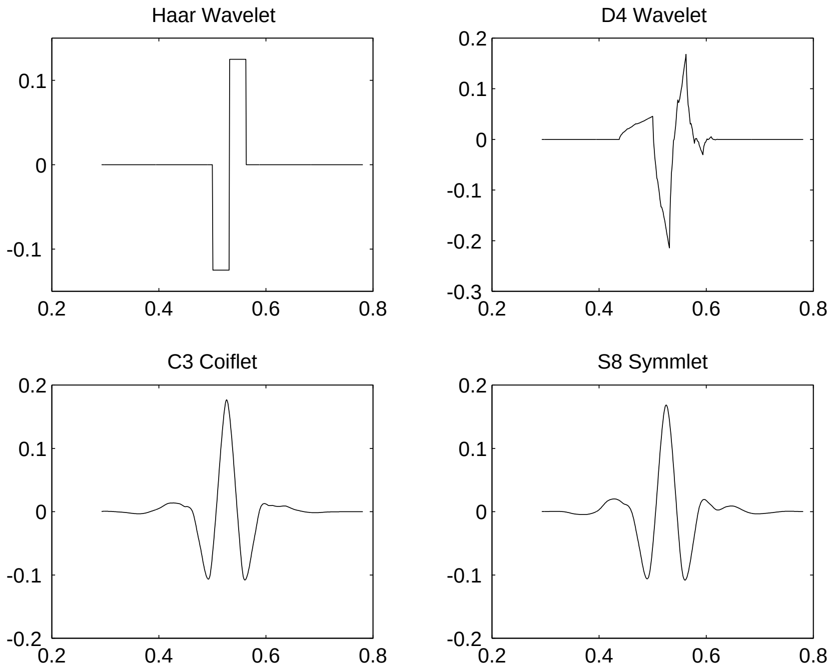


Figure 7: Toon 0111

```
%
%       Coiflet C3      -- orthonormal wavelets system where both father and
%                          mother have special vanishing moments properties
%
%       Symmlet S8      -- nearly-symmetric orthogonal wavelet of
%                          compact support with 8 vanishing moments.
%
```

When we run this .m file, by typing `toon0111` at the Matlab prompt, we get the following figure. The figure displays wavelets of compact support with various degrees of smoothness:

Inspecting the source of this figure reveals how it was made:

```
  wave = MakeWavelet(4,8,'Haar',[],'Mother',1024);
  subplot(221);
  t = (1:1024)./1024;
  plot(t(300:800),wave(300:800)); title(' Haar Wavelet ');
%
  wave = MakeWavelet(4,8,'Daubechies',4,'Mother',1024);
  subplot(222);
  plot(t(300:800),wave(300:800)); title(' D4 Wavelet ');
%
  wave = MakeWavelet(4,8,'Coiflet',3,'Mother',1024);
  subplot(223);
  plot(t(300:800),wave(300:800)); title(' C3 Coiflet ');
%
  wave = MakeWavelet(4,8,'Symmlet',8,'Mother',1024);
  subplot(224);
  plot(t(300:800),wave(300:800)); title(' S8 Symmlet ');
```

The secret: use of the `MakeWavelet` routine, along with some standard plotting commands.

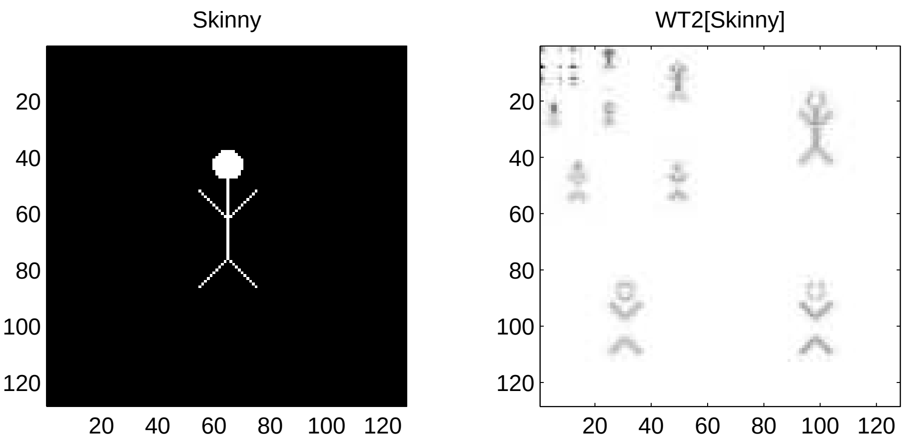


Figure 8: Toon 0231

3.4.3 Example: Looking at a 2-d Wavelet Transform

The file `toon0231` is associated with the outline segment

```
2.3 2-d wavelet analysis of synthetic objects
  2.3.1 Stick Figure
```

The file contains the following help header, accessible by typing `help toon0231`:

```
% toon0231 -- Visualize Wavelet Xform
%
%  Here we display a 2-d image (a stick figure)
%  and its 2-d wavelet transform.
%
```

When we run this .m file, by typing `toon0231` at the Matlab prompt, we get the following figure:

Inspecting the source of this figure reveals how it was made:

```
    stick = Make2dSignal('StickFigure', 128);
%
    clg; subplot(121);
    AutoImage(stick);
    title('Skinny');
%
    Q = MakeONFilter('Coiflet',2);
    wstick = FWT2_PO(stick,3,Q);
%
    subplot(122);
    zmat = sqrt(abs(wstick));
    zmat = 256-3.8*zmat;
    AutoImage(zmat);
    title('WT2[Skinny]');
```

The secret: use of the `Make2dSignal` routine to access the `StickFigure` image, use of `FWT2_PO` to calculate the 2d wavelet transform and use of `AutoImage` to display the images on the screen.

3.4.4 Example: 2-d Compression Methods

The file `toon0548` is associated with the outline segment

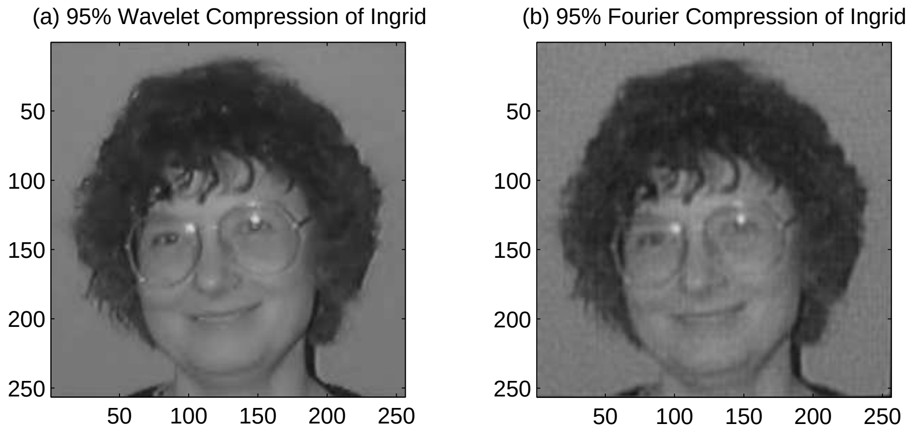


Figure 9: Compression of Ingrid Daubechies

```
5.4 Data Compression 2-d Ingrid image
  2-d Ingrid Image        toon0541
  FWT Imgrid Image        toon0542
  Nonzero Patterns        toon0543
  Co/Dec 95% FWT          toon0544
  Co/Dec 95% DCT          toon0545
  Compression Numbers     toon0546
  Error Comparisons       toon0547
  Side-by-Side            toon0548
```

All eight files should be accessed in sequence. The file `toon0548` contains the following help header, accessible by typing `help toon0548`:

```
% toon0548 -- Data Compression of Ingrid Daubechies
%
%  A side-by-side comparison of the 95% wavelet
%  and Fourier compressions.
%
```

When we run this .m file by typing `toon0548` at the Matlab prompt, we get the following figure:

Inspecting the source of this figure reveals how it was made:

```
    subplot(121);
    GrayImage(icw_ingrid,256);
    title('(a) 95% Wavelet Compression of Ingrid');
%
    subplot(122)
    GrayImage(icf_ingrid,256);
    title('(b) 95% Fourier Compression of Ingrid');
```

The secret, in this case, is hidden, because this file only uses the results of earlier calculations. To track down the earlier calculations, we have to inspect the source of other figures in the sequence. The command `type toon0542` shows how the image is transformed into the wavelet domain using `FWT2_PO`:

```
% toon0542 -- Data Compression of Ingrid Daubechies
%
%  Take Ingrid into the Wavelet Domain.
%
```

```
    qmf = MakeONFilter('Coiflet',2);
    wingrid = FWT2_PO(ingrid,3,qmf);
%
    zmat = abs(wingrid);
    AutoImage(zmat);
    title('Wavelet Transform of Ingrid Daubechies');
```

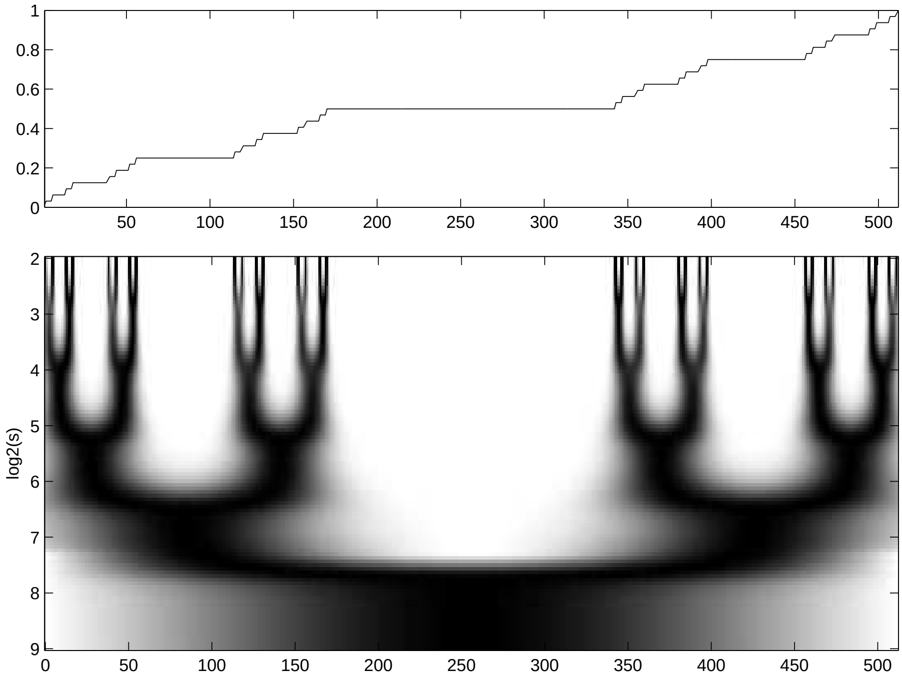
The command `type toon0543` reveals how the wavelet-domain object is operated upon, setting 95% of the coefficients to zero (ellipses indicate omissions):

```
% toon0543 -- Data Compression of Ingrid Daubechies
%
%  Investigate Sparsity in the Wavelet Transform of Daubechies.
%
    wcsort = sort(abs(wingrid(:)));
 ...
%
%  Sparsify Image
%
    wthresh = wcsort(floor(.95*65536));
    cw_ingrid = wingrid .* (abs(wingrid) > wthresh);
  ...
  ...
```

The command `type toon0544` reveals how the sparse object is then transformed back into the original domain, using `IWT2_PO`:

```
% toon0544 -- Data Compression of Ingrid Daubechies
%
%  Reconstruct Daubechies from 5% of her coefficients.
%
    icw_ingrid = IWT2_PO(cw_ingrid,3,qmf);
    AutoImage(icw_ingrid);
    title('95% Wavelet Co/Dec of Daubechies');
```

In this case, then, figuring out “the secret” may require following the flow of Matlab execution across several .m files.



3.5 Books

The Books/WaveTour directory contains a collection of scripts which reproduce the figures in Mallat's book "*A Wavelet Tour of Signal Processing*" [30]. The scripts are devided into 8 subdirectories, WTCH02, WTCH04, ..., WTCH10, corresponding to the book chapters in which the figures appear. Each subdirectory contains a Demo file (e.g. WTCH04Demo in directory WTCH04 and so on.) This file allows you to reproduce the figures in the corresponding chapter of the book. When you invoke the file in Matlab by typing its name (without .m extension), a menu bar will appear on the screen. You can see each one of the figures from the corresponding chapter by mouse-clicking on the approprite menu item. As each figure appears, the Matlab command window will contain narrative explaining what you see in the figure window. The borwser- WTBrowser enables you to run any of the Demo files by clicking on the desired chapter. To run this browser just type its name in the Matlab command window.
Another way to view a single figure is by invoking its scripts directly. Each chapter's subdirectory includes the chapter's figures individual scripts. These scripts are named according to the chapter and figure number in the book. (e.g. wt06fig11.m corresponds to figure 6.11 in the book which can be found in page 196). All .m files can be inspected and changed.

3.5.1 Devil's staircase and its wavelet transform

Consider Figure [?], which appears as figure 6.16 on page 204 of [30]. This figure presents the wavelet transform of a Devils's staircase with equal weights. To reproduce the figure, type WTBrowser, choose "Chapter 6: Wavelet Zoom" and, from Chapter 6 menu bar choose figure 16.

An alternative way to view figure is by entering wt06fig16 into the Matlab command window. The following short description will appear in the Matlab window:

```
Figure 6.16 Window 1: The top signal is devil's staircase,
calculated by integrating a Cantor measure constructed with equal
weights p_1 = p_2 = 0.5. The wavelet transform is calculated with a
```

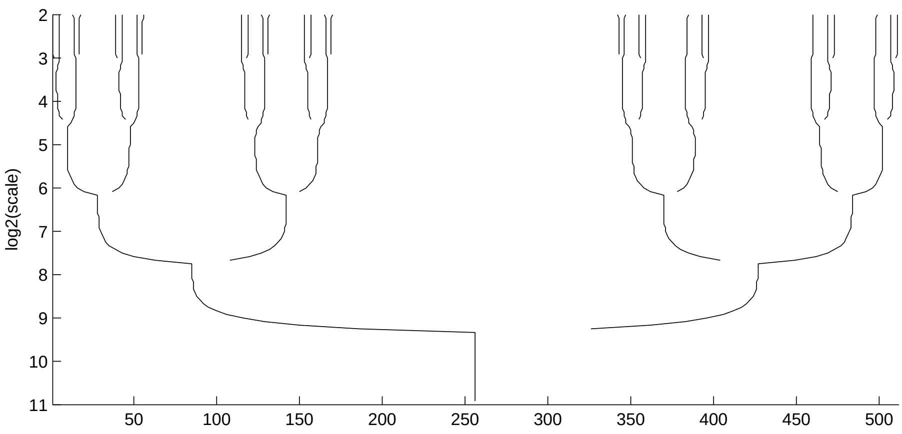


Figure 10: Devil's Staircase and its wavelet transform

Figure 11: Linear approximation of Lena using wavelets

`wavelet which is the first derivative of a Gaussian. Window 2: Wavelet transform modulus maxima. We briefly describe in this section some underlying philosophical ideas which have guided us in the construction of this software library.`

3.5.2 Example-Linear and nonlinear approximation for Lena's image

Figure [?] appears as figure 9.2 on page 382, presents Lena's image and linear approximation by using coarse scale wavelets. Neglecting the fine scale wavelet coefficients blurs the image, especially in the neighborhood of edges. To view the figure on your monitor, you can use the browser again, this time choosing Figure 2 of Chapter 9. Figure [?], which appears as Figure 9.4 in the book, presents a nonlinear approximation of the same image and the corresponding wavelet coefficients matrix.

3.6 Reproducible Research

Jon Claerbout, a distinguished exploration geophysicist at Stanford, has in recent years championed the concept of *really reproducible research* in the "Computational Sciences."

Figure 12: Non-linear approximation of Lena

The "Computational Sciences" he has in mind are fields in which mathematical and computer science heuristics may suggest algorithms to be tried on scientific signal processing and imaging problems, but mathematical analysis alone is not able to predict fully the behavior and suitability of algorithms for specific datasets. Therefore experiments are necessary and such experiments ought, in principle, to be reproducible, just as experiments in other fields of science.

Some background information may help the reader. Suppose we are working in an area like exploration seismology where the goal is an image of the subsurface, and computational science aims to produce better images. However, the deliverable is not an image itself, but instead the software environment that, applied in the right way, produces the image, and which, hopefully, could be applied to other datasets to produce equally nice images. The scientific findings may turn out to be a *knowledge of parameter settings* for this complex software environment that seem to lead to good results on real datasets.

With this as background, reproducibility of experiments requires having the complete software environment available in other laboratories and the full source code available for inspection, modification, and application under varied parameter settings.

Reducing this to a slogan: *An article about computational science in a scientific publication is* **not** *the scholarship itself, it is merely* **advertising** *of the scholarship. The actual scholarship is the complete software development environment and the complete set of instructions which generated the figures.*

The advantage of reproducibility for the progress of the discipline is clear. When a really good idea is found, everyone else can be using it right away. When a mistaken finding is reported, it is rooted out almost immediately, etc.

The barriers to sharing complete software environments are also clear: if you are the developer of a nice piece of software, you may not want to give other people the benefit of your investment of time simply for the benefit of an abstract principle such as scientific progress. Even if you are altruistic enough to make your work available to others in this way, it's a lot of extra work to generate code clean enough for others to look at; most people prefer to make the figures for their articles using quick-and-dirty undocumented code, and move on to the next project.

Claerbout and his colleagues have developed a discipline for building their own software, so that from the start, they expect it to be made available to others as part of the publication of their work. Specifically, they publish CD-ROMs (available from Stanford University Press) which contain the text of their books along with a special viewer that makes those books *interactive documents*, where as one reads the document, each figure is accompanied by the possibility of pop-up windows which allow one to interact with the code that generated the figure, to "burn" the illustration (i.e., erase the postscript file supplied with the distribution), and to rebuild the figure from scratch on one's own machine. By following the discipline of planning to publish in this way from the beginning, they maintain all their work in a form which is easy to make available to others at any point in time.

Why do we think this concept is significant for the wavelet community? It is our perception that as we approach specific applications using wavelets, we are becoming a computational science like seismic imaging. Arguments over data compression have everything to do with specifics: exactly what was done (which wavelets, which coders, which corpus of data was compressed) with exactly what parameters. In this setting, publishing figures or results without the complete software environment could be compared to publishing an announcement of a mathematical theorem without giving the proof.

With the recent rapid spread of Internet facilities worldwide and the standardization of scientific computing on about five machine architectures, most of which are UNIX-based, it becomes feasible and timely to actually experiment with protocols implementing the goal of reproducible research. We may forget that in other sciences, the canonical form for scientific articles was arrived at incrementally. For example, it has been said that Pasteur introduced the notion of publishing a full description of methods, materials and laboratory procedures.

The publication of Wavelab is a modest step in this direction. Partly to indicate our esteem for the work of Claerbout and the Stanford Exploration Project, we are making available code which will allow the interested Internet-nik to reproduce the figures in our articles and to study the exact parameter settings and algorithms which were used in those articles. This arrangement does not, at the moment, conform to the Stanford Exploration Project's idea of interactive document, in which a special TeX viewer is tied to special code resources that can rebuild and vary the figures in a paper. However, at a primitive level, it provides the interested researcher with the TeX files of the documents and everything necessary (sauf Matlab) to rebuild and modify the figures in those articles.

We hope to publish code in this way for several forthcoming articles and also to learn more about the concepts of *Interactive Document* and reproducible research.

3.7 Freeware

Richard Stallman and others associated with the GNU project have pioneered the idea of free software – software that can freely be redistributed by users. (This does not mean free of cost; it means the rights any one person has over the software are the same as those of any other). We have been influenced by this, but obviously cannot completely follow them since we require of users that they have Matlab, which is not freely redistributable. In fact, Richard Stallman told one of us that making our software available in Matlab was the worst thing we could do, as this might encourage people to buy Matlab!

In our opinion, the Freeware concept is useful and has had a major impact. However, the Freeware concept has limits, and the Matlab example shows this clearly. Modern scientific computing depends on *quantitative programming environments* like Matlab, S-PLUS, MATHEMATICA, X-MATH, IDL and so on. These are widely available, widely understood high-level languages in which key concepts of scientific discourse (Fast Fourier Transform, etc.) are available as built-in, easily usable features. At the moment, a complete Freeware implementation of one's computational experiments requires, more or less, emulating Jon Claerbout's example and writing all one's tools from scratch in C or Fortran. Claerbout is forced to do this because of the massive size of the datasets he uses, which exceed the bounds of Matlab or other QPE's. Most working scientists have smaller-scale datasets than Claerbout, and so they can use modern QPE's. Moreover they are busy, and view scientific computing only as a sideshow. They cannot be expected to start from scratch and develop all their code in C when they can get, much more quickly and reliably, the same results in a very high-level language. Thus the temptation to work in a QPE is almost irresistible.

Stallman might retort that there are now freeware QPE's. Octave is a Matlab work-alike in many ways, developed by John Eaton at the University of Texas Chemical Engineering department. Octave is developed strictly within the GNU philosophy, and so makes available a Stallman-acceptable QPE. In the latest release, version 1.1, it comes close to Matlab 4.X in many key ways, so that a large fraction of what we do will run under Octave. Perhaps one day an Octave port of Wavelab will be feasible.

But where does a working scientist spend his/her effort? On ports to noncommercial systems that will satisfy the urges towards binary liberation? Perhaps, but only with a lower priority than other activities. It is our impression that, at the moment, there are relatively few scientists who would complain about lack of access to Wavelab due to the cost of Matlab, and far fewer who would complain that we fall short of full reproducibility by our dependence on a commercial tool at one stage. Sorry to let you down, Richard.

4 Fine Print

In making available our software, the authors have tried to follow some simple guidelines. We spell them out in this section to avoid misunderstandings about what we are offering, why we are offering it, what rights we give you and what rights we retain for ourselves.

The directory `WaveLab/Documentations` contains the following files:

```
ADDINGNEWFEATURES      -  How to Add New Features to WaveLab
BUGREPORT              -  How to report bugs about WaveLab
COPYING                -  WaveLab Copying Permissions
DATASTRUCTURES         -  Basic data structures in WaveLab
FEEDBACK               -  Give feedback about WaveLab
GETTINGSTARTED         -  Ideas for getting started with WaveLab
INSTALLATION           -  Installation of WaveLab
LIMITATIONS            -  WaveLab known limitations
PAYMENT                -  No Charge for WaveLab Software
READING                -  Sources for further reading about wavelets
REGISTRATION           -  WaveLab Registration
SUPPORT                -  WaveLab Support
THANKS                 -  Thanks to contributors
VERSION                -  Part of WaveLab Version v$VERSION$
WARRANTY               -  No Warranty on WaveLab software
```

These describe the philosophy and limitations of our package. The key points are summarized here, by reprinting the contents of some of those files.

4.1 Dependence on Matlab

Matlab is a product of The Mathworks, a successful company based in Natick, Massachusetts. Their product is expensive. You need it to run our software. We do not offer it. You must get it from them. You need version 6.X or later. We have no connection with them. They have no connection with us.

4.2 Registration – Wavelab Registration

(from file `REGISTRATION.m`)

Please Register yourself as a user of Wavelab so that we can send you e-mail about upgrades and enhancements. To Register: e-mail wavelab@stat.stanford.edu with the subject line "registration." If you like, please include information about the version of Matlab you are using and about the type of machine you are using.

4.3 Limitations

This package has been designed to reproduce the figures in our research. Accordingly, it may not solve the problems you have in mind. Please see the file `LIMITATIONS.m` before complaining. Perhaps we are already aware of the problem you have and are seeking to fix it!

4.4 Support

(from file `SUPPORT.m`)

This software has been developed as part of the research effort of the authors under various federally supported grants. If you find that it does not work correctly, please e-mail a notification of your problem to Wavelab. Use the format indicated in the file `BUGREPORTS.m`.

To the extent that we can isolate the problem and develop a solution, and to the extent that it fits in with our schedule with releasing a new version, we will attempt to fix the problem.

4.5 No Charge – No Charge for Wavelab Software

(From file Payment.m)

Wavelab software is available at NO CHARGE from the wavelab website at

http://www-stat.stanford.edu/~ wavelab.

The software is copyrighted. For permissions to copy, see the file copying.m or you may e-mail wavelab@stat.stanford.edu. If you use this software to produce scientific articles, we would appreciate being informed of the article's title, authors, topic, and place of appearance. If this software played a major enabling role in your scientific work, and you are so moved, you might acknowledge us in your article.

4.6 No Warranty – No Warranty on WaveLab software

(From file Warranty.m)

There is no warranty attached to the Wavelab software.

If you find that it does not work correctly, use the file BUGREPORTS.m as a template to compile a description of the problem, including as far as possible a complete m-file script that generates the error. E-mail the description to: wavelab@stat.stanford.edu. Resources permitting, an effort will be made to correct the problem for a future release.

FORMAL LEGAL DISCLAIMER OF WARRANTY

We make no warranties, explicit or implicit, that the programs contained in this collection are free of error, or are consistent with any particular standard of accuracy, or that they will meet your requirements for any particular application. They should not be relied on for any purpose where incorrect results could result in loss of property or personal injury. If you do use programs for any such purpose it is at your own risk. The authors disclaim all liability of any kind, either direct or consequential, resulting from your use of these programs.

4.7 Copyright – Wavelab Copying Permissions

(From file copying.m)

This software was developed with partial support of the National Science Foundation Grant DMS 92-09130, DMS 95-05151, DMS 00-77261, DMS 01-40698 and of NASA Astrophysics Data Program, and of other sponsors. Contributors to this effort include Jonathan Buckheit, Shaobing Chen, David Donoho, Iain Johnstone, and Eric Kolaczyk, all from Stanford and Jeffrey Scargle, from NASA-Ames.

In EVERY case, the software is COPYRIGHTED by the original author. For permissions, e-mail wavelab@stat.stanford.edu.

COPYLEFT NOTICE

Permission is granted to make and distribute verbatim copies of this entire software package, provided that all files, directories, and subdirectories are copied *together* as a *unit*.

Here *copying as a unit* means: all the files listed in the file `WLFiles` are copied, so that among other things the thanks.m notice, and this permission notice accompany all copies of the full software, and every ".m" file continues to contain a copy of the file `VERSION.m` inside it.

Permission is granted to make and distribute modified copies of this software, under the conditions for verbatim copying, provided that the entire resulting derived work is distributed under the terms of a permission notice identical to this one. Names of new authors, their affiliations and information about their improvements may be added to the files copying.m and thanks.m.

The purpose of this permission notice is to allow you to make copies of the software and distribute them to others subject to the constraint that you maintain the collection of tools here as a unit. This enables people to whom you give the software to be aware of its origins, to ask questions of us by e-mail, to request improvements, obtain later releases, etc.

If you seek permission to copy and distribute translations of this software into another language, please e-mail a specific request to wavelab@stat.stanford.edu. If you seek permission to excerpt a *part* of the software library for example to appear in a scientific publication, please e-mail a specific request to wavelab@stat.stanford.edu.

4.8 Thanks – Thanks to contributors

(from file thanks.m)

This software was developed with support of the National Science Foundation grant DMS 92-09130 It is further developed with support of National Aeronautics and Space Administration, Astrophysics Data Program, NSF DMS 95 05151, NSF DMS 00-77261, NSF DMS 01-40698 . Contributors to this software include Jonathan Buckheit, Stanford; Shaobing Chen, Stanford; Maureen Clerc, Ecole Polytechnique; James Crutchfield, Berkeley & Santa Fe Institute, David Donoho, Stanford; Mark Duncan, Stanford; Hong-Ye Gao, Berkeley (now at Statistical Science); Xiaoming Huo, Stanford (now at Georgia Tech); Jermo Kalifa, Ecole Polytechnique; Ofer Levi, Stanford; Stephan Mallat, Ecole Polytechnique; Iain Johnstone, Stanford; Eric Kolaczyk, Stanford (now at Boston University); Jeffrey Scargle, NASA-Ames; Karl Young, Nara Ames and Thomas Yu, Stanford.

Further contributions are welcome, for info e-mail wavelab@stat.stanford.edu.

Students: We would like to thank students enrolled in Statistics 319, Stanford University, Fall 1992; Statistics 260, University of California, Berkeley, Fall 1993; Statistics 371, Stanford University, Spring 1994; Statistics 343, Stanford University, Fall 1995.

Contributors of Data: The following have contributed data which is part of our system. Jean-Paul Bibérian, Université de Marseille, Luminy; Chris Brislawn, Los Alamos National Labs; R.R. Coifman, Yale University; Ingrid Daubechies, AT&T Bell Labs; Paul Donoho, Chevron; Jeffrey Hoch, Rowland Institute; Doug Jones, Univ. Illinois; Jitendra Malik, UC Berkeley; Stephane Mallat, Courant Institute; Adrian Maudsley, VA Medical Center, San Francisco; Chris Raphael, Stanford University; Jan-Olov Stromberg, University of Tromso; and Zhifeng Zhang, Courant Institute.

References

[1] Antonini, M., M. Barlaud, P. Mathieu and I. Daubechies (1992). "Image coding using wavelet transforms." *IEEE Trans. Image Proc.*, **1**(2), 205-20.

[2] Buckheit, J.B. and D.L. Donoho (1995). "A Cartoon Guide to Wavelets." Technical Report, Department of Statistics, Stanford University.
`ftp://playfair.stanford.edu/pub/buckheit/toons.ps`.

[3] Buckheit, J.B. and D.L. Donoho (1995). "WaveLab Architecture."
`ftp://playfair.stanford.edu/pub/wavelab/WaveLabArch.ps`.

[4] Buckheit, J.B, S. Chen, D.L. Donoho, I.M. Johnstone and J.D. Scargle (1995). "WaveLab Reference Manual." `ftp://playfair.stanford.edu/pub/wavelab/WaveLabRef.ps`.

[5] Buckheit, J.B. and D.L. Donoho (1995). "WaveLab and Reproducible Research." In A. Antoniadis and G. Oppenheim (Eds.), *Wavelets and Statistics* (pp. 55-81). New York: Springer-Verlag. `ftp://playfair.stanford.edu/pub/buckheit/wavelab.ps.Z`.

[6] Chen, S. and D.L. Donoho. (1994). "On Basis Pursuit." Technical Report, Department of Statistics, Stanford University.
`ftp://playfair.stanford.edu/pub/chen_s/asilomar.ps.Z`.

[7] Claerbout, Jon (1994). Hypertext Documents about Reproducible Research.
`http://sepwww.stanford.edu`.

[8] Cohen, A., I. Daubechies and J.C. Feauveau (1992). "Biorthogonal bases of compactly supported wavelets." *Comm. Pure Appl. Math.*, **45**, 485-560.

[9] Cohen, A., I. Daubechies, B. Jawerth and P. Vial (1992). "Multiresolution analysis, wavelets, and fast algorithms on an interval." *Comptes Rendus Acad. Sci. Paris A*, **316**, 417-421.

[10] Coifman, R.R. and D.L. Donoho (1995). "Translation-Invariant De-Noising." In *Wavelets and Statistics.* In A. Antoniadis and G. Oppenheim (Eds.), *Wavelets and Statistics* (pp. 125-150). New York: Springer-Verlag.

[11] Coifman, R.R. and Y. Meyer (1991). "Remarques sur l'analyse de Fourier à fenêtre." *Comptes Rendus Acad. Sci. Paris Series A*, **312**, 259-261.

[12] Coifman, R.R., Y. Meyer and M.V. Wickerhauser (1992). "Wavelet analysis and signal processing." In M.B. Ruskai et al. (Eds.), *Wavelets and Their Applications* (pp. 153-178). Boston: Jones and Bartlett.

[13] Coifman, R.R. and M.V. Wickerhauser (1992). "Entropy-based algorithms for best-basis selection." *IEEE Trans. Info. Theory*, **38**(2), 713-718.

[14] Daubechies, I. (1992) *Ten Lectures on Wavelets.* Philadelphia: SIAM.

[15] DeVore, R.A., B. Jawerth and B.J. Lucier (1992). "Image compression through wavelet transform coding." *IEEE Trans. Info Theory*, **38**(2), 719-746.

[16] Donoho, D.L. (1992). "Interpolating Wavelet Transforms." Technical Report, Department of Statistics, Stanford University.
`ftp://playfair.stanford.edu/pub/donoho/interpol.ps.Z`.

[17] Donoho, D.L. (1993). "Nonlinear Wavelet Methods for Recovery of Signals, Images, and Densities from Noisy and Incomplete Data." In I. Daubechies (Ed.), *Different Perspectives on Wavelets* (pp. 173-205). Providence, RI: American Mathematical Society.
`ftp://playfair.stanford.edu/pub/donoho/ShortCourse.ps.Z`.

[18] Donoho, D.L. (1994). "On Minimum Entropy Segmentation." In C.K. Chui, L. Montefusco and L. Puccio (Eds.), *Wavelets: Theory, Algorithms and Applications* (pp. 233-269). San Diego: Academic Press.
`ftp://playfair.stanford.edu/pub/donoho/MES_TechReport.ps.Z`.

[19] Donoho, D.L. (1993). "Smooth Wavelet Decompositions with Blocky Coefficient Kernels." In L. Schumaker and F. Ward (Eds.), *Recent Advances in Wavelet Analysis* (pp. 259-308). San Diego: Academic Press. `ftp://playfair.stanford.edu/pub/donoho/blocky.ps.Z`.

[20] Donoho, D.L. (1993). "Unconditional Bases are Optimal Bases for Data Compression and for Statistical Estimation." *Applied and Computational Harmonic Analysis*, **1**, 100-115.
`ftp://playfair.stanford.edu/pub/donoho/UBRelease.ps.Z`.

[21] Donoho, D.L. (1993). "Wavelet Shrinkage and W.V.D. – A Ten-Minute Tour." In Y. Meyer and S. Roques (Eds.), *Progress in Wavelet Analysis and Applications* (pp. 109-128). Gif-sur-Yvette: Éditions Frontières.
`ftp://playfair.stanford.edu/pub/donoho/toulouse.ps.Z`.

[22] Donoho, D.L. and I.M. Johnstone (1994). "Adapting to Unknown Smoothness by Wavelet Shrinkage." To appear, *J. Amer. Stat. Assoc.*
`ftp://playfair.stanford.edu/pub/donoho/ausws.ps.Z`.

[23] Donoho, D.L. and I.M. Johnstone (1994). "Ideal Spatial Adaptation via Wavelet Shrinkage." *Biometrika*, **81**, 425-455. `ftp://playfair.stanford.edu/pub/donoho/isaws.ps.Z`.

[24] Donoho, D.L. and I.M. Johnstone (1994). "Ideal Time-Frequency Denoising." Technical Report, Department of Statistics, Stanford University.
`ftp://playfair.stanford.edu/pub/donoho/tfdenoise.ps.Z`.

[25] Donoho, D.L., I.M. Johnstone, G. Kerkyacharian and D. Picard (1993). "Wavelet Shrinkage: Asymptopia." *J. Roy. Statist. Soc. Series B*, **2**, 301-369. `ftp://playfair.stanford.edu/pub/donoho/asymp.ps.Z`.

[26] Kolaczyk, E. (1994). *WVD Solution of Inverse Problems.* Doctoral Dissertation, Stanford University Dept. of Statistics (D.L. Donoho, supervisor).

[27] Mallat, S. and S. Zhang (1993). "Matching Pursuits with Time-Frequency Dictionaries." *IEEE Transactions on Signal Processing,* **41**(12), 3397-3415.

[28] Meyer, Y. (1993). *Wavelets: Algorithms and Applications.* Philadelphia: SIAM.

[29] Wickerhauser, M.V. (1994). *Adapted Wavelet Analysis, from Theory to Software.* Boston: AK Peters.

[30] Mallat, S. (1997). *A Wavelet Tour of Signal Processing.* Academic Press