Introduction to Wavelets in Scilab

Anuradha Amrutkar

Indian Institute of Technology Bombay
anuradha.amrutkar@gmail.com

National Workshop on Scilab
Sinhagad College of Engineering, Wadgaon, Pune
January 5, 2011
The word WAVELET literally means small wave. Wavelets are localised waves and they extend not from $-\infty$ to $+\infty$ but only for a finite duration of time.
The word WAVELET literally means small wave.

Wavelets are localised waves and they extend not from $-\infty$ to $+\infty$ but only for a finite duration of time.
Since waves extend over the entire space, they do not need any shift parameter. Thus, a Fourier Transform maps 1-D time signals to 1-D frequency signals, whereas the wavelet transform maps 1-D time signals to 2-D scale (frequency) and shift parameter signals.
Since waves extend over the entire space, they do not need any shift parameter.

Thus, a Fourier Transform maps 1-D time signals to 1-D frequency signals, whereas

The wavelet transform maps 1-D time signals to 2-D scale(frequency) and shift parameter signals.
Example 1

Let us see a program which finds out the approximate coefficients and detailed coefficients of a given signal.

Anuradha Amrutkar
Introduction to Wavelets in Scilab
Let us see a program which finds out the approximate coefficients and detailed coefficients of a given signal.
Example1: dwt.sce

In this Example:

1. \( x = \text{linspace}(-\pi, \pi, 10000) \);
2. \( s = \sin(x) \);
   // Constructs an elementary sine wave signal
3. \([ca1, cd1] = \text{dwt}(s, 'haar') \);
   // Performs single-level discrete wavelet transform of "s" by "haar".
4. The graph of approximate coefficients (cA) and detailed coefficients (cD) is plotted using the plot() command
5. The above procedure is repeated for "db2" type of wavelet.

Anuradha Amrutkar
Introduction to Wavelets in Scilab
In this Example:

1. \( x=\text{linspace}(-\pi,\pi,10000); \)
2. \( s=\text{sin}(x); \) //Constructs and Elementary sine wave signal
3. \([\text{ca1,cd1}] = \text{dwt}(s,'\text{haar}'); \) // Perform single-level discrete wavelet transform of \( s \) by \( \text{haar} \).
4. The Graph of Approximate co-efficients\((cA)\) and Detailed co-efficient\((cD)\) is Plotted using the \text{plot()}\ command
5. The above procedure is repeated for \( \text{db2} \) type of wavelet.
In this Example:

1. Steps 1, 2 and 3 are same as above.

2. `ss = idwt(ca1, cd1, 'haar');` //Perform single-level inverse discrete wavelet transform, illustrating that idwt is the inverse function of dwt.

3. The Graph of Approximate co-efficients (cA) and Detailed co-efficient (cD) is Plotted using the plot() command.
In this Example:

1. Steps 1, 2 and 3 are same as above.

2. `ss = idwt(ca1, cd1, 'haar');` // Perform single-level inverse discrete wavelet transform, illustrating that `idwt` is the inverse function of `dwt`.

3. The Graph of Approximate co-efficients (cA) and Detailed co-efficient (cD) is Plotted using the `plot()` command
Commands: dwt & idwt

- **dwt**: Discrete Fast Wavelet Transform
  - For discrete fast wavelet transform with the signal extension method optional argument.
  - As output it gives values of $c_A$: Approximate coefficients and $c_D$: Detailed coefficients.
  - For syntax detailed help see type "help dwt"

- **idwt**: Inverse Discrete Fast Wavelet Transform
  - For inverse discrete fast wavelet transform.
  - Coefficient could be void vector as \(['\emptyset']\) for $c_A$ or $c_D$.
  - As output it gives a reconstructed vector.
  - For syntax detailed help see type "help idwt"
Commands: `dwt` & `idwt`

→ **dwt**: Discrete Fast Wavelet Transform
   - `dwt` is for discrete fast wavelet transform with the signal extension method optional argument.
   - As output it gives values of `cA`: Approximate co-efficients and `cD`: Detailed co-efficients

   - For Syntax Detailed help see type "help dwt"

→ **idwt**: Inverse Discrete Fast Wavelet Transform
   - `idwt` is for inverse discrete fast wavelet transform.
   - Coefficient could be void vector as `[]` for `cA` or `cD`.
   - As output it gives a Reconstructed Vector

   - For Syntax Detailed help see type "help idwt"
Example3:wavelet.sce

Let us Revise the Decomposition Diagram for the wavelets:

Anuradha Amrutkar
Introduction to Wavelets in Scilab
Let us Revise the Decomposition Diagram for the wavelets:

![Decomposition Diagram for Wavelets]
Example3: wavelet.sce

In this Example:

1. \( s = [1:100] \);
2. \( l = \text{length}(s) \);
3. \( a = \sin(2\pi s/100) + \sin(3\pi s/100) \);

// Constructs elementary sine wave signal

Anuradha Amrutkar
Introduction to Wavelets in Scilab
In this Example:

1. \( s = [1:100]; \)
2. \( l_s = \text{length}(s) \)
3. \( a = \sin(2\times\pi\times s/100) + \sin(3\times\pi\times s/100); \) //Constructs and Elementary sine wave signal
The coefficients of all the components of a third-level decomposition (that is, the third-level approximation and the first three levels of detail) are returned concatenated into one vector, C. Vector L gives the lengths of each component.

\[ [C, L] = \text{wavedec}(a, 3, 'haar') \]
The coefficients of all the components of a third-level decomposition (that is, the third-level approximation and the first three levels of detail) are returned concatenated into one vector, \( C \).

Vector \( L \) gives the lengths of each component.

4. \([C,L] = \text{wavedec}(a,3,'haar');\)
To extract the level 3 approximation coefficients from C, type:
\[ cA3 = \text{appcoef}(C, L, 'haar', 3); \]
To extract the levels 3, 2, and 1 detail coefficients from C, type:
\[ cD3 = \text{detcoef}(C, L, 3); \]
\[ cD2 = \text{detcoef}(C, L, 2); \]
\[ cD1 = \text{detcoef}(C, L, 1); \]
The above can be written in one command as:
\[ [cD1, cD2, cD3] = \text{detcoef}(C, L, [1, 2, 3]); \]
To extract the level 3 approximation coefficients from C, type:

5. \( cA_3 = \text{appcoef}(C, L, 'haar', 3); \)

To extract the levels 3, 2, and 1 detail coefficients from C, type

6. \( cD_3 = \text{detcoef}(C, L, 3); \)
7. \( cD_2 = \text{detcoef}(C, L, 2); \)
8. \( cD_1 = \text{detcoef}(C, L, 1); \)

The above can be written in one command as:

9. \([cD_1, cD_2, cD_3] = \text{detcoef}(C, L, [1, 2, 3]);\)
Example3:wavelet.sce

To reconstruct the level 3 approximation from C, type:

```
A3 = wrcoef('a',C,L,'haar',3);
```

To reconstruct the details at levels 1, 2, and 3, from C, type:

```
D1 = wrcoef('d',C,L,'haar',1);
D2 = wrcoef('d',C,L,'haar',2);
D3 = wrcoef('d',C,L,'haar',3);
```
To reconstruct the level 3 approximation from C, type

10. \( A_3 = \text{wrcoef}('a',C,L,'haar',3); \)

To reconstruct the details at levels 1, 2, and 3, from C, type

11. \( D_1 = \text{wrcoef}('d',C,L,'haar',1); \)
12. \( D_2 = \text{wrcoef}('d',C,L,'haar',2); \)
13. \( D_3 = \text{wrcoef}('d',C,L,'haar',3); \)
Example1: wavelet.sce

Display the results of a multilevel decomposition

Anuradha Amrutkar
Introduction to Wavelets in Scilab
Example1:wavelet.sce

Display the results of a multilevel decomposition

Approximation A3

Detail D1

Detail D2

Detail D3
To reconstruct the original signal from the wavelet decomposition structure, type:

```
A3 = waverec(C,L,'haar');
```

Of course, in discarding all the high-frequency information, we've also lost many of the original signal's sharpest features.
To reconstruct the original signal from the wavelet decomposition structure, type

\[ A3 = \text{waverec}(C,L,'haar'); \]

Of course, in discarding all the high-frequency information, we’ve also lost many of the original signal’s sharpest features.
To compare the approximation to the original signal, type
To compare the approximation to the original signal, type
Commands Used

The commands used in the Multi-level Decomposition and Construction of Approximate and Detailed Coefficients are:

- `wavedec`: Multiple Level Discrete Fast Wavelet Transform
- `waverec`: Multiple Level Inverse Discrete Fast Wavelet Transform
- `appcoef`: One Dimension Approximation Coefficient Reconstruction
- `detcoef`: One Dimension Detail Coefficient Extraction
- `wrcoef`: Reconstruction from single branch from multiple level
The commands used in the Multi-level Decomposition and Construction of Approximate and Detailed Coefficients are:

- `wavedec`: Multiple Level Discrete Fast Wavelet Transform
- `waverec`: Multiple Level Inverse Discrete Fast Wavelet Transform
- `appcoef`: One Dimension Approximation Coefficient Reconstruction
- `detcoef`: One Dimension Detail Coefficient Extraction
- `wrcoef`: Restruction from single branch from multiple level
Help

Please type help command name to see the Usage, Description and Examples for that particular command.
Please type `help command_name` to see the Usage, Description and Examples for that particular command.
Further Exploration

Optimal de-noising requires a more subtle approach called thresholding. This involves discarding only the portion of the details that exceeds a certain limit.
Further Exploration

Optimal de-noising requires a more subtle approach called thresholding.
Further Exploration

Optimal de-noising requires a more subtle approach called thresholding.
This involves discarding only the portion of the details that exceeds a certain limit.
Thank You!!
Thank You!!

- Thank You!