User's Guide

FOR THE

Modelica Wavelet Library

AUTHOR: **Dr. Jianbo GAO**
Technische Universität München
November, 2013

1 Introduction

This document describes the first release version of the Modelica Wavelet Library.

Users have to acquire the basic knowledge about wavelet transform in order to use this library.

The library is released with the following files.

File Name	Description
Wavelet.mo	The main part of wavelet library saved in Modelica standard format. This is a text file.
fft_c.c	An external C-file. It is necessary for generating the Meyer wavelet. The location of this C-file has to be known (set as the current working directory or set in the searching path list) to the Modelica environment software, e.g. Dymola, in order that the Modelica compiler can access this file.
testSignal1.mat	A testing signal for the wavelet examples. This is a simulation data file generated by Dymola. This file can be automatically generated by running the model “testSignal1” delivered with this library.
testSignal2.mat	A second testing signal.
Users_guide.pdf	This document in PDF format. It provides an overall description of the library.
Library_reference.pdf	A library reference in PDF format. This is actually an extraction of the description texts embedded in the library.
Help folder	This folder includes multiple files. It is saved in HTML format and contains the same information as the user’s guide in PDF format. The main file to access the help information is “Wavelet.html “ in this folder.

During running the wavelet library some other files will be generated by the Modelica environment software.

The library is organized in the following classes.

Name	Description
Examples	Some examples showing how to use this library
MRA	A wavelet application toolbox for Multi-Resolution Analysis (MRA)
Denoising	A wavelet application toolbox for denoising
Transform	Wavelet transform, including the algorithms for continuous wavelet transform, discrete wavelet transform and inverse wavelet transform

Families	Definition of wavelet families, including functions to generate the wavelet filter banks, the scaling and the wavelet functions of all available wavelet families in this library
General	General purpose functions
Records	Modelica classes for defining parameters and user interfaces
Types	Modelica classes for defining parameters

This wavelet library is specifically designed for post-processing of the Modelica simulation results. It cannot be executed with simulation models.

The execution of this library requires several public libraries, including **ModelicaServices**, **Modelica_LinearSystem2** and **Plot3D**. All of them are free libraries.

This library is programmed and tested in Dymola demo version 2013 (32-bit). To load the library, open the file Wavelet.mo.

2 Features of the Wavelet Library

The features of this Modelica wavelet library are summarized as follows:

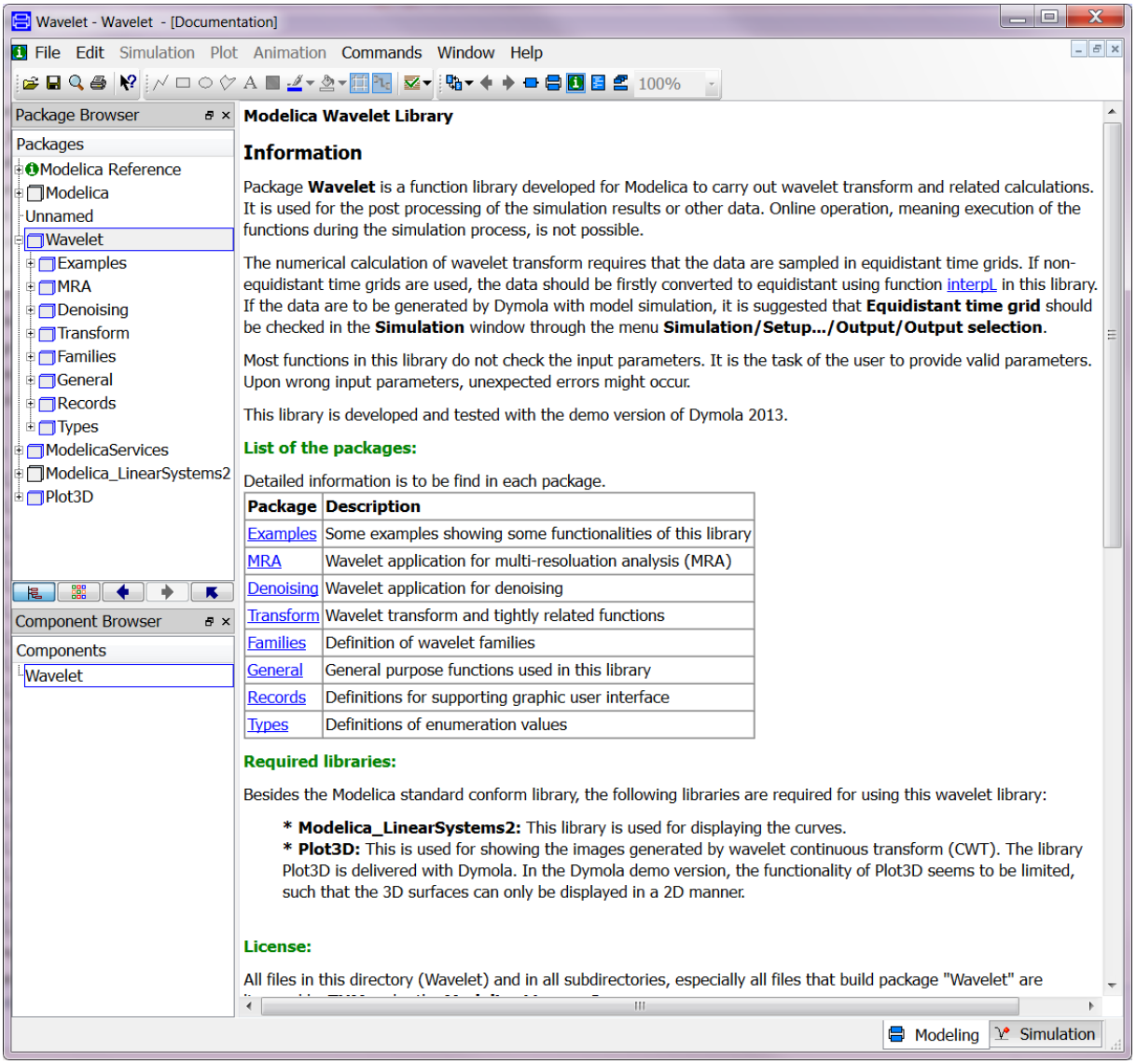
- Fifteen wavelet families, including orthogonal, biorthogonal, real and complex wavelets;
- One dimensional continuous wavelet transform;
- One dimensional discrete wavelet transform and inverse transform;
- One dimensional multi-level wavelet decomposition and reconstruction;
- One dimensional wavelet multi-resolution analysis;
- One dimensional wavelet denoising, could also be used for data compression;
- Graphic user interface (GUI) for parameter input;
- GUI in the forms of curve and image for showing calculation results;
- The combination of GUIs for input and output realising a simple interactive working process;
- Reading simulation data directly from the hard disk (in Dymola environment);
- Versatile examples showing the functionalities of the library;
- Fully open source under Modelica License 2.

These features fulfils the project requirements.

3 Content of the Wavelet Library

3.1 User's Guide

The user's guide is embedded in every single Modelica class with documentation. A specific package for the user's guide is not provided. This information explains the basic concept, structure and usage of this library. The screen shot of the document page of the library in the highest level is given in the following figure. Through the hypo-links the user can be guided to the documentation of all sub-packages.

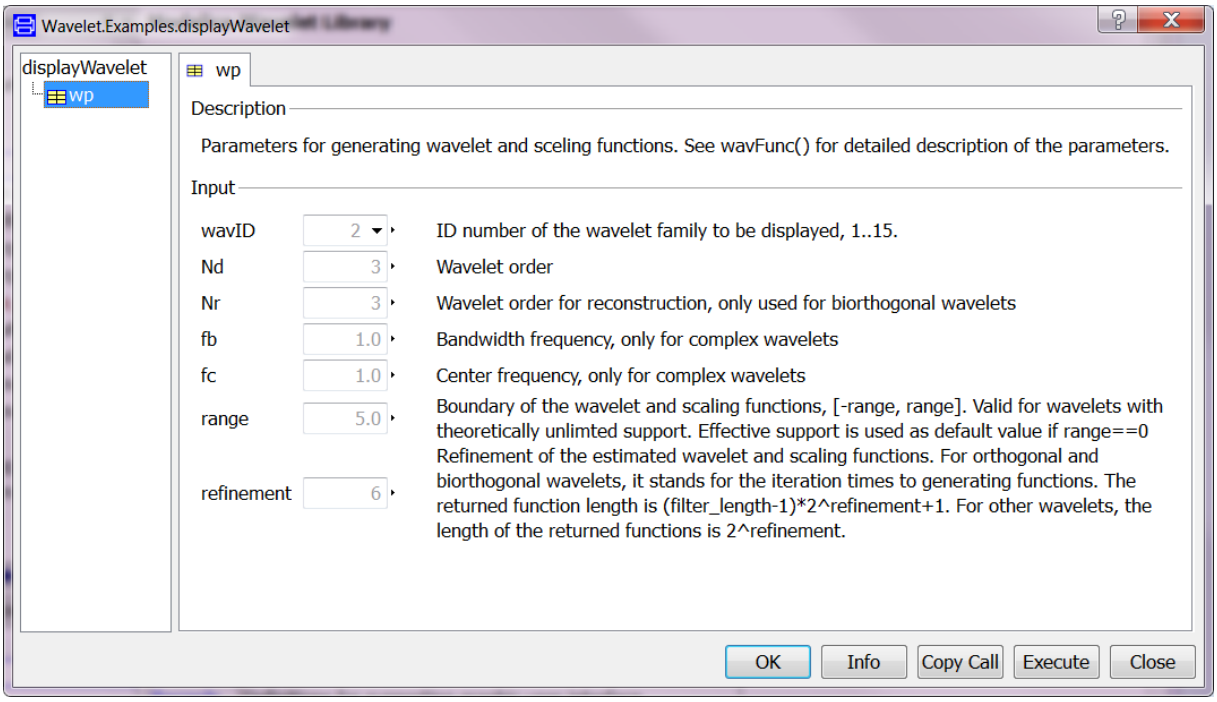


3.2 Examples

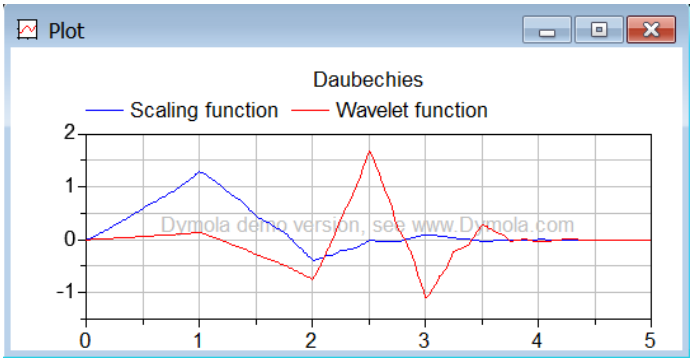
The functionalities of this library are illustrated graphically with the examples in this sub-package. The examples also provide the users a start point to build up their own algorithms. Six examples are delivered with the library.

3.2.1 Example 1: displayWavelets

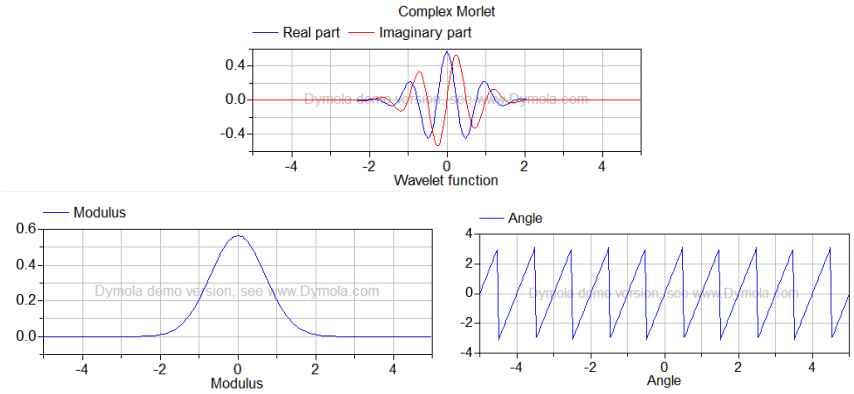
This example shows all available wavelet functions included in this library with graphical user interface (GUI). The GUI for the input parameters is illustrated as follows.



By clicking the button “OK” or “Execute” without changing any parameters, the second order Daubechies scaling and wavelet functions will be displayed, as follows:

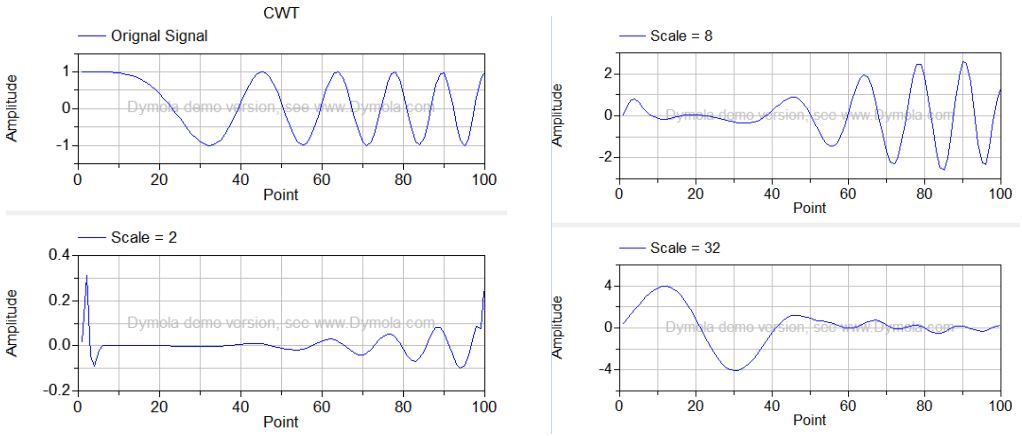


By changing the parameters, other wavelet functions can be displayed. Since different wavelets has different properties, the curves shown for different wavelets will be different. For example, complex Morlet wavelet has no scaling function, only the complex wavelet function will be shown in two different forms, real / imaginary parts and modulus / angle values. An example is given as follows.



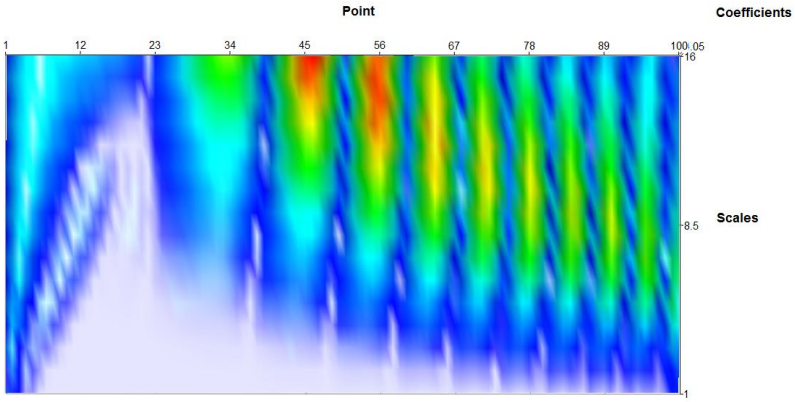
3.2.2 Example 2: cwtChirpCurve

This example calculates the continuous wavelet transform (CWT) of a chirp signal and displays the results of scales 2, 8 and 32 with curves. The original chirp signal is displayed, too.



3.2.3 Example 3: cwtChirpImage

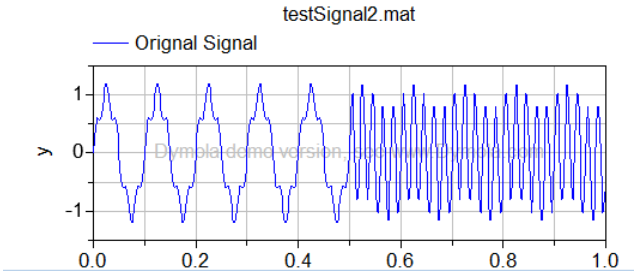
This example calculates the continuous wavelet transform (CWT) of a chirp signal and displays the results of scales 1:16 with an image. The original chirp signal is displayed, too. The following images shows the transform result. The display of images is realized using the library, plot3D, delivered with Dymola. With a full version of Dymola a surface with 3 dimensional effect can be shown. In the demo version, the third dimension is omitted.



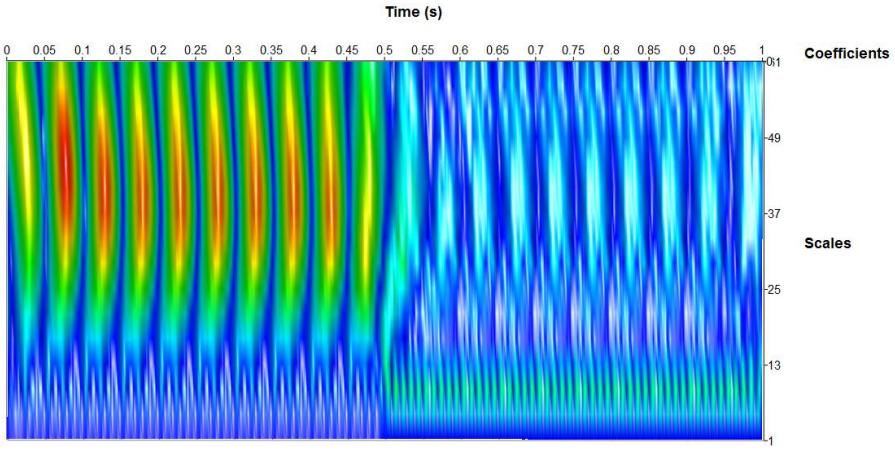
3.2.4 Example 4: fileData_cwtn

This example reads in the simulation result from the file testSingal2.mat, carries out continuous wavelet transform (CWT) and displays the results with an image in pseudo-color. Before running this example, the simulation data have to be available. This can be done by running the model “testSignal2”. The user can select other data for analysis or change wavelets through the GUI.

The signal of the model is:

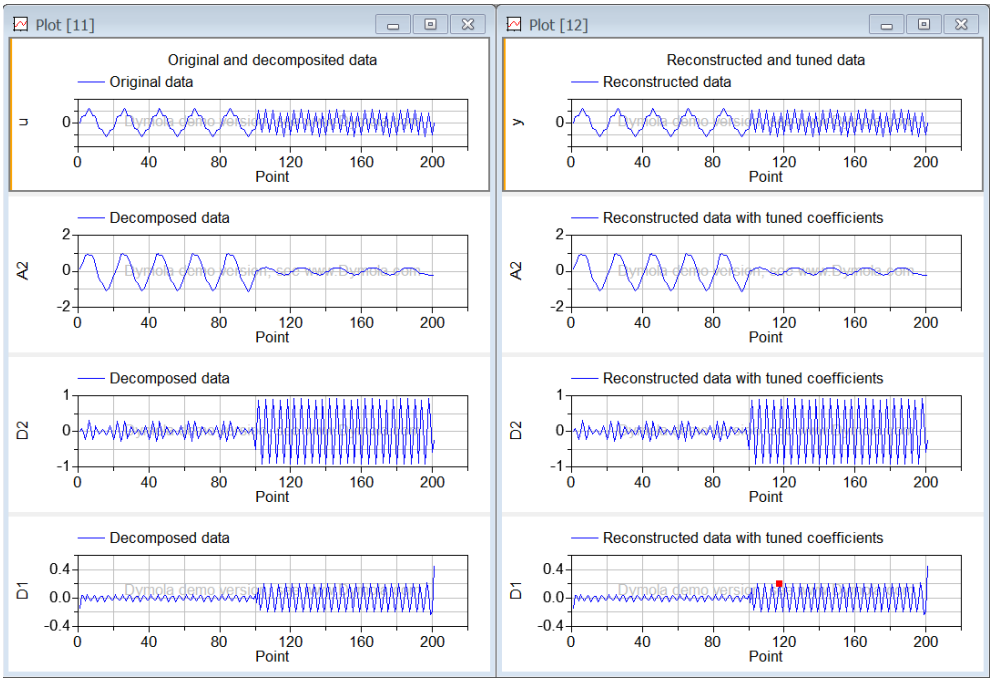


It will take some time to finish the calculation and the CWT result with the default input values will be shown as:



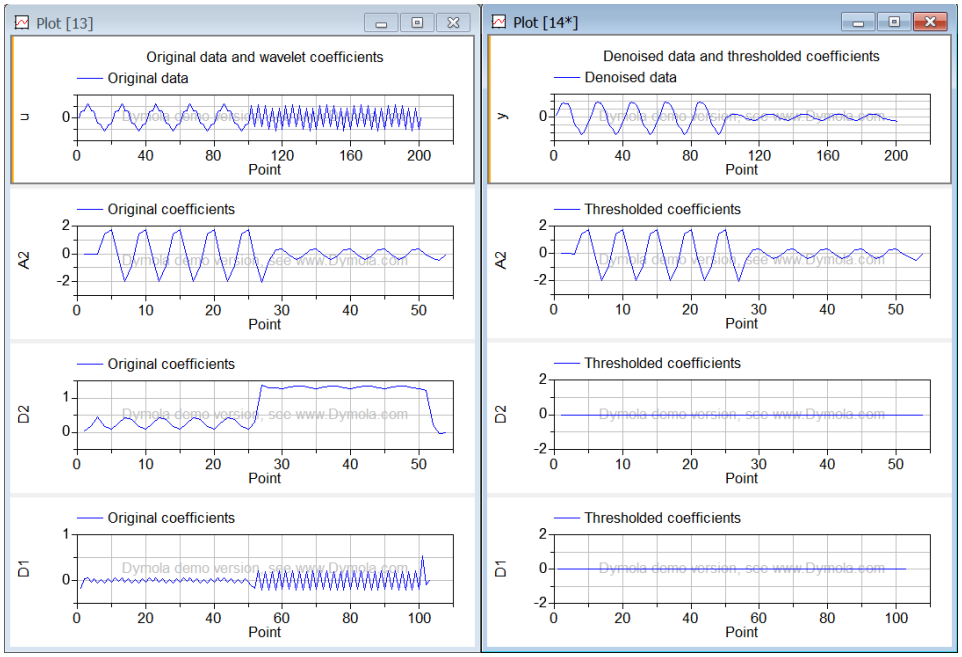
3.2.5 Example 5: fileDataMRA

This example reads in the simulation result from the file testSingal2.mat, carries out two-level wavelet MRA and displays the results with curves. Before running this example, the simulation data have to be available. This can be done by running the model “testSignal2”. The user can select other data for analysis or change wavelets through the GUI. In the example, the wavelet coefficients are not tuned before reconstruction. Therefore, the reconstructed data are identical to the original data.



3.2.6 Example 6: fileDataDenoising

This example reads in the simulation result from the file testSingal2.mat, carries out wavelet denoising with two-level decomposition and displays the results with curves. Before running this example, the simulation data have to be available. This can be done by running the model “testSignal2”. The user can select other data for analysis or change wavelets through the GUI.



3.3 MRA

The MRA package provides a tool to carry out wavelet Multi-Resolution Analysis. The function mraGUI starts a GUI for parameter input and displays the results in curves. The function, mra, carries out MRA without GUI. The data are output with values. The third function, tuneCoef, is used to tune the wavelet coefficients, supporting the MRA functionality.

The example, fileDataMRA, shows how to use this toolbox.

3.4 Denoising

The Denoising package provides a tool to carry out wavelet denoising. The function denoisGUI starts a GUI for parameter input and displays the results in curves. The function, denois, carries out denoising without GUI. The data are output with values. The rest two functions are used to calculate and apply thresholds for denoising.

It has to be noted that the noise in the signal must be normal distributed white noise with zero mean and one standard deviation if the automatic method, thSelect, is to be used. Otherwise, the thresholds have to be selected or calculated manually.

The same data processing for denoising could also be used for data compression. The idea is based on such consideration: The wavelet coefficients with small magnitudes contain few information about the original signal. By removing these coefficients the data amount can be greatly reduced while not much information is lost. If a suitable wavelet is used,most coefficients might have very small magnitudes. Thus, a large compression ratio can be achieved.

The example, fileDataDenoising, shows how to use this toolbox.

3.5 Transform

This Modelica class includes the core functions of the wavelet library. They are listed in the following table. The outputs of these functions are numeric values.

Function	**Description**
cwt	One-dimensional continous wavelet transform with a given wavelet function
cwtn	One-dimensional continous wavelet transform with a given wavelet name
dwt	One-dimensional discrete wavelet transform (one-level decomposition)
idwt	One-dimensional inverse discrete wavelet transform (one-level reconstruction)
wavDec	Wavelet multilevel decomposition
wavRec	Wavelet multilevel reconstruction
wavRec1	Wavelet reconstruction using coefficients from a single level
wavCoef1	Extract the wavelet coefficients of a single level

3.6 Families

This Modelica class includes functions to define the wavelets in this library. Fifteen wavelet families are realized in this library. Detailed descriptions are included in the library. The wavelet families are listed here with short descriptions:

wavName	**ID**	**type**	**Description**
Haar	1	1	Haar wavelet
Daubechies	2	1	Daubechies wavelet

Symlets	3	1	Symlets wavelet
Coiflets	4	1	Coiflets wavelet
BiorSpline	5	2	Biorthogonal spline wavelet
RBiorSpline	6	2	Reverse biorthogonal spline wavelet
DiscreteMeyer	7	3	Discrete Meyer wavelet
Meyer	8	3	Meyer wavelet
Gaussian	9	4	Gaussian wavelet
MexicanHat	10	4	Mexican hat wavelet
Morlet	11	4	Morlet wavelet
ComplexGaussian	12	5	Complex Gaussian wavelet
ComplexMorlet	13	5	Complex Morlet wavelet
ComplexShannon	14	5	Complex Shannon wavelet
ComplexFBSpline	15	5	Complex frequency B-Spline wavelet

All wavelet families can be categorized into five types:

Type	**Description**
1	orthogonal wavelets
2	biorthogonal wavelets
3	non-(bi)orthogonal wavelets with scaling function
4	non-(bi)orthogonal wavelets without scaling function
5	complex wavelets without scaling function

3.7 General

This Modelica class includes the general functions that are used by other packages. In this release the following functions are included:

Function	**Description**
_fft	Fast Fourier transform. External C code is called
cumSum	Cumulative sum of a vector. Data type is Real.
cumSumInt	Cumulative sum of a vector. Data type is Integer
diff	Difference between every two adjacent elements of a vector
fft	Fast fourier transform
fftShift	Shift zero-frequency component to center of spectrum
findIndex	Find the location of a value in a monotone vector

filterBank	Get the four wavelet filters based on a given scaling filter
ifft	Inverse fast fourier transform
innerProduct	Inner product of two same length vectors
interPL	One dimensional linear interpolation
midVector	Extract the middle part of a vector
nStdIfft	Inverse non-standard 1-D fast Fourier transform
quadReverse	Quadrature mirror of a given vector
sinc	Sinc function
upsample	Upsampling of a vector (insert a zero after every element except the last one)
wavConv	Fully convolving of a data vector and a filter vector for wavelet transform

3.8 Records

The following Modelica records are defined to record structuralized data and provide the possibility for graphical user interface.

Recrod	**Description**
denoisParameters	Parameters for 1D wavelet denoising
mraParameters	Parameters for multi-resolution analysis (MRA)
wavFuncOut	Output values of the wavFunc() function
waveletDefinition	All parameters for defining a wavelet

3.9 Types

The Modelica types define enumeration values of parameters, including

Type	**Description**
mraDisplay	Selection of display data for MRA
threshMethod	Methods for calculating threshold for denoising
waveletID	Definition of wavelet indentifiers

4 Testing of the library

The algorithms of this wavelet library were completely tested with the black-box method. Testing vectors have been defined for each algorithm. The function outputs were checked visually and

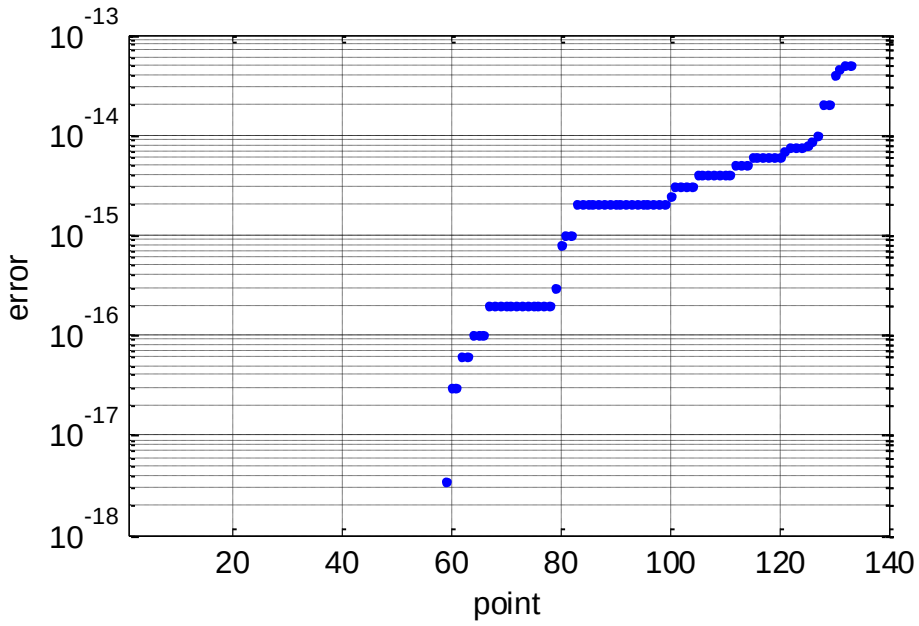
numerically. The numeric check was carried out with the assistance of MATLAB because most algorithms written in the Modelica wavelet library are included in the MATLAB wavelet toolbox. For these algorithms, the same input parameters were given to the functions of the Modelica wavelet library and the corresponding commands in MATLAB. The calculation results of both tools were compared. An algorithm of the library is considered correct if the decrepancy is below a certain limit.

All testing results were recorded and summarized. As an example, the following table shows the summary of the testing results of the function “Wavelet.Families.wavCoiflets”, which is used to generate the wavelet filters associated with Coiflets wavelets.

order	**F**	**lod**	**hid**	**lor**	**hir**	**Result**
1	-	-	-	-	-	Error = 0
2	-	-	-	-	-	Error = 0
3	-	-	-	-	-	Max. error < 1e-16
4	-	-	-	-	-	Max. error < 1e-16
5	-	-	-	-	-	Error = 0

The first six columns of the table records the input and output values of the function. Since the data amount of the output values is too large, the data are stored in a separate file. The last column “Result”, records the errors of the outputs obtained by the Modelica function under test compared with the data calculated by the same algorithm written in MATLAB.

All other functions in the library were tested with the same process and altogether 133 numeric testing results were recorded. The errors are illustrated in the following figure. Since the values are shown in logarithm axis, the first 58 points with zero value are not plotted. The testing showed that the more than half of the tests have the errors below 2e-16, which is around the digital error of a double precision floating number. Other errors with the values up to 5e-14 might be caused by the precision of the direct constant numbers and detailed difference of the coding in the algorithms.



In addition, visual check was carried out for all functions that have graphical inputs and outputs. All results coincided with the specifications. For the functions outputting curves and images, the graphic data are actually numerical results and also tested numerically.