The `mw` Toolbox

Fritz Keinert
Department of Mathematics
Iowa State University
Ames, IA 50011
keinert@iastate.edu

February 20, 2004

1 Overview

This toolbox provides routines for working with multiwavelets (which includes scalar wavelets as a special case). It requires the toolbox `mpoly` (for matrix Laurent polynomials). This toolbox uses object-oriented programming, so it requires MATLAB® version 5 or higher to run. Please report any problems or other suggestions to keinert@iastate.edu.

A multiwavelet is given by a function vector

$$\boldsymbol{\phi}(x) = \begin{pmatrix} \phi_1(x) \\ \vdots \\ \phi_r(x) \end{pmatrix}$$

which satisfies a recursion relation

$$\boldsymbol{\phi}(x) = \sqrt{m} \sum_{k=k_0}^{k_1} H_k \, \boldsymbol{\phi}(mx - k).$$

r is called the *multiplicity* of $\boldsymbol{\phi}$; the integer $m \geq 2$ is the *dilation factor*. The *recursion coefficients* H_k are $r \times r$ matrices.

It is assumed that the user is familiar with the basic concepts of wavelets and multiwavelets, so I won't repeat them here. This toolbox is intended to go with my book *Wavelets and Multiwavelets*, Studies in advanced mathematics, vol. 42, Chapman & Hall/CRC Press, Boca Raton, FL, ISBN 1-58488-304-9. You can find explanations of all the terms and techniques used in these routines in the book.

It is also assumed that you have read the description of the `mpoly` toolbox first.

The coefficient matrices H_k can be either numerical or symbolic. Implementing symbolic matrices has been more of a headache than anything else; if anything is not working correctly in this toolbox, it will most likely relate to symbolic matrices. Also, symbolic calculations are noticeably slower than numerical calculations, and MATLAB® may need some help in simplifying the result.

This toolbox is a work in progress. Check back periodically for updates and extensions. My own primary interest lies in the theory of multiwavelets, so the application aspect (pre- and postfiltering, performing wavelet transforms) is not very developed yet. I am planning to concentrate my future efforts first in this direction, in particular towards more kinds of boundary handling. At the moment, only the periodic transform is implemented.

2 Installation

Install the routines, including the subdirectories `mpoly`, `@double` and `@sym`, in a directory in the MATLAB® path.

There are several test routines in the `mw` directory. You should run `test_all` to verify that everything is working. Alternatively, you can run `test_double`, `test_sym`, `test_mpoly_numerical`, `test_mpoly_symbolic`, `test_mw` individually.

Notes:

1. Normally MATLAB® will notice new routines that are added while it is running. However, this does not seem to work for `@`-directories. Remove the installation directory from the path, and add it again.

2. Don't worry if the test routines seem to take a long time to run. They use symbolic computation extensively, which is in fact quite slow, and MATLAB® also needs to compile each routine in the toolbox the first time it is run. Numerical computations, after the initial compilation step, will run much faster.

3 The `mpoly` data type

A matrix Laurent polynomial is a polynomial of the form

$$P(z) = P_{k_0} z^{k_0} + P_{k_0+1} z^{k_0+1} + \cdots + P_{k_1} z^{k_1},$$

where the coefficients P_k are two-dimensional matrices, all of the same size. The exponents are integers, possibly negative. Equivalently, $P(z)$ is a matrix with (Laurent) polynomial entries.

Internally, this toolbox represents a matrix polynomial as a structure with the fields

`coef`	A three-dimensional array; `coef(:,:,1)` is P_{k_0}, `coef(:,:,2)` is $P_(k_0+1)$, etc.
`min`	Starting exponent k_0.
`type`	one of `''`, `'symbol'`, `'polyphase'`
`m`	dilation factor
`r`	multiplicity

A multiwavelet is represented by either its symbol or its polyphase matrix, both of which are `mpoly` objects. You can create many standard wavelets using the `wavelet` subroutine, and convert between symbol and polyphase representations by using the `symbol` and `polyphase` routines.

All routines accept multiwavelet arguments in either of the two forms, and will convert them if necessary.

4 List of Routines

A list of routines is given in table 1. For details about the routines, use the `help` function inside MATLAB®. It is recommended that you run routine `example_mw` to get an overview. You can also look at the test routines for further examples.

Notes:

1. You can add more types of wavelets to the `wavelet` routine yourself, or build them from scratch using the `mpoly` command.

   The `symbol` and `polyphase` routines are in the `mpoly` toolbox, so they are not listed here, but they certainly pertain to the `mw` toolbox.

2. The testing and example routines are not necessary for the operation of the toolbox. You can remove them after installation, if you want.

Table 1: List of Routines

Constructor Routine	`wavelet`	create one of the standard wavelets or multiwavelets (see note 1.)
Properties of Wavelets	`approximation_order`	find the approximation order
	`continuous_moment`	find continuous moments
	`correlation`	find recursion coefficients of correlation of two wavelets
	`discrete_moment`	find discrete moments
	`refinement_matrix`	compute the refinement matrix
	`sobolev_estimate`	quick estimate of Sobolev exponent
	`sobolev_exponent`	slower, but better estimate of Sobolev exponent
	`transition_matrix`	find the transition matrix
Manipulating Wavelets	`itst`	inverse two-scale similarity transform
	`projection_factorization`	factorize polyphase matrix
	`projection_shift`	shift support of dual wavelet
	`tst`	two-scale similarity transform
Computations	`dwt`	discrete wavelet transform
	`idwt`	inverse discrete wavelet transform
	`point_values`	find point values of scaling and wavelet functions
	`postfilter`	postfilter reconstructed signal
	`prefilter`	prefilter signal
Supporting Routines	`nullspace`	nullspace of a matrix
	`projection_factor`	build a projection factor
	`range`	range of a matrix
Testing/Example Routines	`example_mpoly`	see note 2.
	`example_mw`	
	`test_double`	
	`test_mpoly_numerical`	
	`test_mpoly_symbolic`	
	`test_mw`	
	`test_sym`	