dxf — *polymorphic wavelet filter*

v2.0(β) / 24.02.2023

Mihai-Tiberiu Dima, Maria Dima

leader Yu.N. Pepelyshev
Frank Neutron Physics Lab, JINR-Russia

Header: `#include "dxf.hh"`

Libraries: `libdxf.a` (static) linux-3.10.0-1160.88.1.el7.x86_64
`libdxf.so` (dynamic) gcc-9.3.1 20200408

Description

The ***dxf*** class is a wavelet filter that takes as input either a time-series of `double` from a file, or an `auto` (polymorph, non-abelian) type from a vector.

It performs the filtering using one to the following windowing functions:

- Gaussian (`GS`), centered, falloff 90% @ interval ends
- Poisson (`PS`), centered, symmetrical, falloff 90% @ interval ends
- `O4`, the O4 apodisation function
- Blackman (`BA`), the Blackman apodisation function
- Hamming (`HM`), the Hamming apodisation function
- `F0`, the Cooley-Tukey (box) apodisation function
- `F1`, the FoxLima Fourier space apodisation function F1 (similar to Welch)
- `F2`, -- " -- -- " -- F2 (-- " -- Welch^2)
- `F3`, -- " -- -- " -- F3 (-- " -- Welch^3)
- `F4`, -- " -- -- " -- F4 (-- " -- Welch^4)
- `F5`, -- " -- -- " -- F5 (-- " -- Welch^5)

Parameters of the functions:

	apodis./space	spec.leak	sgm	lobe
classical				
PS	Poisson /TIME	-6dB/oct	1.10	-20dB
GX	Gauss -"-	-6 -"-	1.50	-42dB
HM	Hamming -"-	-6 -"-	1.50	-42dB
O4	O4 -"-	-6 -"-	1.70	-52dB
BA	Blackman -"-	-18 -"-	1.90	-60dB
FoxLima				
F0	F0 / NATIVE	-6dB/oct	1.00	-13dB
F1	F1 / FOURIER	-12 -"-	1.30	-22dB
F2	F2 / -"-	-18 -"-	1.60	-28dB
F3	F3 / -"-	-24 -"-	1.80	-34dB
F4	F4 / -"-	-30 -"-	2.00	-40dB
F5	F5 / -"-	-36 -"-	2.20	-44dB

For the polymorphic part the class comes instantiated with:

- `scalar`
- `cpx<scalar>`
- `vu2<scalar>, vu2<cpx<scalar>>`
- `su2<scalar>, su2<cpx<scalar>>`

where `scalar = int, float, double, long double.`

The filter works in 2 modes:

- *file-2-file* ... designed for reading huge files, filtering them and writing to new files. This version works with `double`'s.

```
dxf filter(0.30, 0.02, "file_in", "file_out", "F9");
// filter frequency f0 from the data, with a sigma = s0
//
// 0.30 = f0 * delta_sampling
// 0.02 = s0 * delta_sampling
//
// use - apodisation method FoxLima F9
//     - input  file = file_in    (column #1 = value of  sample )
//     - output file = file_out (column #1 = index nr. sample,
//                                       #2 = cos center interv.
//                                       #3 = sin       -- " --
//                                       #4 = sample  -- " --     )
```

- *vec-2-vec* .... designed for reading from a polymorphic vector and writing the results to a vector of the same type. The code comes instantiated with the following types: `scalar`, `cpx<scalar>`, `vu2<scalar>`, `vu2<cpx<scalar>>`, `su2<scalar>`, `su2<cpx<scalar>>` - where `scalar` = `int`, `float`, `double`, `long double`. The types `vu2` and `su2` are SU(2) vectors, respectively matrices; `cpx` is the *complex* type.

```
using dtype = float                                  ;
int   NN    = 10000                                  ;
// ... define + fill vec with data
auto vec = new vu2<cpx<dtype>>[NN]                   ;
// ... define output vectors
auto csx = new vu2<cpx<dtype>>[NN]                   ;
auto snx = new vu2<cpx<dtype>>[NN]                   ;
// ... call filter
dxf filter(0.30, 0.02, vec, csx, snx, NN, "GX")  ;
//     - input  vector = vec of vu2<cpx<float>>, complex SU(2) vectors
//     - output vector = csx ... cos component
//                       snx ... sin component
```

The ***makefile*** is banale, however with full pfledged functionality: `make libs`, `make test`, `make run`, `make clean`.

The class comes with 4 examples and 1 application example.