Digital Filters library

Generated by Doxygen 1.9.8

1 DigitalFilters Library **1**

1.1 About the project 1

1.1.1 Features 2

1.1.2 Requirements 2

1.1.3 Project structure 2

1.1.4 Licence 3

2 Hierarchical Index **3**

2.1 Class Hierarchy 3

3 Class Index **3**

3.1 Class List 3

4 File Index **4**

4.1 File List 4

5 Class Documentation **4**

5.1 JK::BodeData Struct Reference 4

5.1.1 Detailed Description 5

5.2 JK::Filter Class Reference 5

5.2.1 Detailed Description 7

5.2.2 Constructor & Destructor Documentation 7

5.2.3 Member Function Documentation 7

5.2.4 Friends And Related Symbol Documentation 11

5.3 JK::FilterCoeffs Struct Reference 11

5.3.1 Detailed Description 11

5.3.2 Member Function Documentation 11

5.4 JK::FIR Class Reference 12

5.4.1 Detailed Description 14

5.4.2 Constructor & Destructor Documentation 14

5.4.3 Member Function Documentation 15

5.5 JK::Hamming Class Reference 16

5.5.1 Detailed Description 18

5.5.2 Constructor & Destructor Documentation 18

5.6 JK::Hanning Class Reference 18

5.6.1 Detailed Description 20

5.6.2 Constructor & Destructor Documentation 20

5.7 JK::IIR Class Reference 21

5.7.1 Detailed Description 23

5.7.2 Constructor & Destructor Documentation 23

5.7.3 Member Function Documentation 23

5.8 JK::Window Class Reference 24

5.8.1 Detailed Description 26

5.8.2 Constructor & Destructor Documentation 26

5.8.3 Member Function Documentation . . . 26
5.8.4 Friends And Related Symbol Documentation . . . 28

6 File Documentation 28

6.1 include/DigitalFilters/filter.hpp File Reference . . . 28
6.1.1 Detailed Description . . . 29
6.2 filter.hpp . . . 29
6.3 include/DigitalFilters/fir.hpp File Reference . . . 30
6.3.1 Detailed Description . . . 31
6.4 fir.hpp . . . 31
6.5 include/DigitalFilters/hamming.hpp File Reference . . . 32
6.5.1 Detailed Description . . . 33
6.6 hamming.hpp . . . 33
6.7 include/DigitalFilters/hanning.hpp File Reference . . . 33
6.7.1 Detailed Description . . . 34
6.8 hanning.hpp . . . 34
6.9 include/DigitalFilters/iir.hpp File Reference . . . 34
6.9.1 Detailed Description . . . 35
6.10 iir.hpp . . . 35
6.11 include/DigitalFilters/plots.hpp File Reference . . . 36
6.11.1 Detailed Description . . . 36
6.11.2 Function Documentation . . . 37
6.12 plots.hpp . . . 37
6.13 include/DigitalFilters/window.hpp File Reference . . . 37
6.13.1 Detailed Description . . . 38
6.14 window.hpp . . . 39

Index 41

1 DigitalFilters Library

A C++ library for the design and analysis of digital signal filters (FIR, IIR). It enables easy creation of low-pass and high-pass filters, application of time windows, and generation of time response and frequency response (Bode) plots.

The library was created by Jakub Kubuszek.

1.1 About the project

The library is my final project for the "Object-oriented programming languages" course at AGH in Cracow.

1.1.1 Features

- **FIR Filters:** Design using the window method (LowPass, HighPass), support for custom coefficients.
- **IIR Filters:** Support for custom coefficients (feedforward/feedback).
- **Time Windows:** Available Hamming and Hanning windows.
- **Analysis:** Generation of data for Bode plots (magnitude and phase).
- **Visualization:** Built-in plotting functions (requires Gnuplot).

1.1.2 Requirements

To build and use the library, you need:

- A C++ compiler supporting the **C++17** standard.
- **CMake** version 3.14 or newer.
- **Gnuplot** (installed on the system) – required for generating PDF plots.

Gnuplot Installation (Linux/Ubuntu)

sudo apt update sudo apt install gnuplot

Building and installation

The project uses CMake. To build the library follow these steps:

1. Create build directory
   mkdir build && cd build
2. Configure project
   cmake ..
3. Build the library
   make
4. (optional) install the library
   sudo make install

1.1.3 Project structure

- `include/DigitalFilters/` - library header files
- `src/` - library source files
- `examples/` - use examples
- `libs/` - external libraries (sciplot)
- `docs/` - doxygen documentation

1.1.4 Licence

This project uses MIT sharing license. More in `LICENSE`

2 Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

- **JK::BodeData** 4
- **JK::Filter** 5
  - **JK::FIR** 12
  - **JK::IIR** 21
- **JK::FilterCoeffs** 11
- **JK::Window** 24
  - **JK::Hamming** 16
  - **JK::Hanning** 18

3 Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

JK::BodeData
Structure for holding all the parameters needed for Bode plots 4

JK::Filter
Filter class representing a digital filter 5

JK::FilterCoeffs
Structure for holding both feedback (a) and feedforward (b) coefficients 11

JK::FIR
FIR class, FIR is a Finite Impulse Response filter 12

JK::Hamming
Hamming window class 16

JK::Hanning
Hanning window class 18

JK::IIR
IIR class, IIR is a Infinite Impulse Response filter 21

JK::Window
Window class 24

4 File Index

4.1 File List

Here is a list of all documented files with brief descriptions:

include/DigitalFilters/filter.hpp
This file handles the base filter class 28

include/DigitalFilters/fir.hpp
This file handles the FIR class, which is a child class of Filter 30

include/DigitalFilters/hamming.hpp
This file handles the Hamming class, which is a child class of Window 32

include/DigitalFilters/hanning.hpp
This file handles the Hanning class, which is a child class of Window 33

include/DigitalFilters/iir.hpp
This file handles the IIR class, which is a child class of Filter 34

include/DigitalFilters/plots.hpp
This file contains plotting functions that use sciplot library to generate pdfs with plots 36

include/DigitalFilters/window.hpp
This file handles the Window class 37

5 Class Documentation

5.1 JK::BodeData Struct Reference

a structure for holding all the parameters needed for Bode plots

```
#include <filter.hpp>
```

Public Attributes

- std::vector< double > **freq**
- std::vector< double > **mag_db**

  frequency vector in Hz
- std::vector< double > **phase_deg**

  magnitude in decibels corresponding to each element from freq vector

5.1.1 Detailed Description

a structure for holding all the parameters needed for Bode plots

The documentation for this struct was generated from the following file:

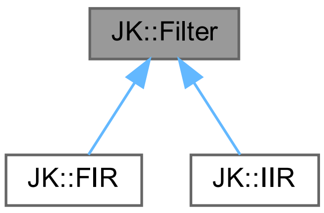
- include/DigitalFilters/filter.hpp

5.2 JK::Filter Class Reference

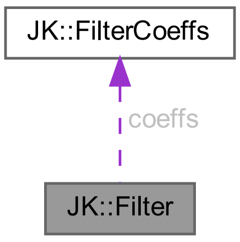
Filter class representing a digital filter.

```
#include <filter.hpp>
```

Inheritance diagram for JK::Filter:



Collaboration diagram for JK::Filter:



Public Member Functions

- bool operator== (const Filter &other) const

  overloaded == operator for comparing filter type and their coefficients sets
- bool operator!= (const Filter &other) const

  overloaded != operator for comparing filter type and their coefficients sets
- Filter (std::string n, const std::vector< double > &b, const std::vector< double > &a)

  class constructor supporting custom coefficients
- **Filter** ()

  default constructor
- virtual ~**Filter** ()

  virtual destructor, for memory safety when using polymorphism
- virtual FilterCoeffs getCoeffs () const =0

  getter for filter coefficients
- virtual std::vector< double > response (const std::vector< double > &x, const int y_len=-1)=0

  method for calculating filter's response to a given signal
- void responsePlot (const std::vector< double > &x, const int y_len=-1)

  method for plotting the filter's response to a given signal
- void applyWindow (const Window &win)

  method for applying a window to filter's coefficients
- BodeData bode (double fs, int plot_points=200) const

  method for calculating filter's frequency response
- void bodePlot (double fs, int plot_points=200) const

  method for plotting filter's frequency response, generates 2 pdf files, one for magnitude plot and one for phase plot
- virtual void setCoeffs (const std::vector< double > &b, const std::vector< double > &a)=0

  setter for filter's coefficients
- std::string getName () const

  getter for filter's name
- void setName (std::string n)

  setter for filter's name

Protected Member Functions

- virtual void print (std::ostream &os) const =0

  helper method for overloaded << operator

Protected Attributes

- std::string **name**

  filter' name
- FilterCoeffs **coeffs**

  structure holding filter's coefficients

Friends

- std::ostream & operator<< (std::ostream &os, const Filter &f)

  overloaded << operator for printing all the coefficients into ouput stream

5.2.1 Detailed Description

Filter class representing a digital filter.

5.2.2 Constructor & Destructor Documentation

Filter()

```
JK::Filter::Filter (
             std::string n,
             const std::vector< double > & b,
             const std::vector< double > & a )
```

class constructor supporting custom coefficients

Parameters

n	name of the filter
b	feedforward coefficients
a	feedback coefficients

5.2.3 Member Function Documentation

applyWindow()

```
void JK::Filter::applyWindow (
             const Window & win )
```

method for applying a window to filter's coefficients

Parameters

win	object of the Window class, representing a Hamming/Hanning window

bode()

```
BodeData JK::Filter::bode (
             double fs,
             int plot_points = 200 ) const
```

method for calculating filter's frequency response

Parameters

fs	sampling frequency
plot_points	number of desired plot points, defaults to 200 if not specified

Returns

returns frequency indexes and both magnitude in decibels and phase in degrees corresponding to each frequency index

bodePlot()

```
void JK::Filter::bodePlot (
             double fs,
             int plot_points = 200 ) const
```

method for plotting filter's frequency response, generates 2 pdf files, one for magnitude plot and one for phase plot

Parameters

fs	sampling frequency
plot_points	number of desired plot points, defaults to 200 if not specified

getCoeffs()

```
virtual FilterCoeffs JK::Filter::getCoeffs ( ) const  [pure virtual]
```

getter for filter coefficients

Returns

returns a structure holding both feedforward and feedback coefficients

Implemented in JK::FIR, and JK::IIR.

getName()

```
std::string JK::Filter::getName ( ) const
```

getter for filter's name

Returns

returns a string containing the name

operator"!=()

```
bool JK::Filter::operator!= (
             const Filter & other ) const
```

overloaded != operator for comparing filter type and their coefficients sets

Parameters

other	other Filter Class object

Returns

returns 0 if filters coeficients sets are the same, 1 if they are different

operator==()

```
bool JK::Filter::operator== (
            const Filter & other ) const
```

overloaded == operator for comparing filter type and their coefficients sets

Parameters

other	other Filter Class object

Returns

returns 1 if filters coeficients sets are the same, 0 if they are different

print()

```
virtual void JK::Filter::print (
            std::ostream & os ) const  [protected], [pure virtual]
```

helper method for overloaded << operator

Parameters

os	output stream

Implemented in JK::FIR, and JK::IIR.

response()

```
virtual std::vector< double > JK::Filter::response (
            const std::vector< double > & x,
            const int y_len = -1 )  [pure virtual]
```

method for calculating filter's response to a given signal

Parameters

x	vector containing signal samples
y_len	desired output length, defaults to length of the signal vector if not specified

Returns

returns a vector with calculated response of the filter

Implemented in JK::FIR, and JK::IIR.

responsePlot()

```
void JK::Filter::responsePlot (
             const std::vector< double > & x,
             const int y_len = -1 )
```

method for plotting the filter's response to a given signal

Parameters

x	vector containing signal samples
y_len	desired output length, defaults to length of the signal vector if not specified

setCoeffs()

```
virtual void JK::Filter::setCoeffs (
             const std::vector< double > & b,
             const std::vector< double > & a )  [pure virtual]
```

setter for filter's coefficients

Parameters

b	feedforward coefficients
a	feedback coefficients

Implemented in JK::FIR, and JK::IIR.

setName()

```
void JK::Filter::setName (
             std::string n )
```

setter for filter's name

Parameters

n	name

5.2.4 Friends And Related Symbol Documentation

operator<<

```
std::ostream & operator<< (
             std::ostream & os,
             const Filter & f )  [friend]
```

overloaded << operator for printing all the coefficients into ouput stream

Parameters

os	ouput stream
f	Filter Class object

Returns

returns the output stream

The documentation for this class was generated from the following files:

- include/DigitalFilters/filter.hpp
- src/filter.cpp

5.3 JK::FilterCoeffs Struct Reference

a structure for holding both feedback (a) and feedforward (b) coefficients

`#include <filter.hpp>`

Public Member Functions

- bool operator== (const FilterCoeffs &other) const

  feedback coefficients, equals [1.0] for fir
- bool operator!= (const FilterCoeffs &other) const

  overloaded != operator for comparing filter coefficients sets

Public Attributes

- std::vector< double > **b**
- std::vector< double > **a**

  feedfoward coefficients

5.3.1 Detailed Description

a structure for holding both feedback (a) and feedforward (b) coefficients

5.3.2 Member Function Documentation

operator"!=()

```
bool JK::FilterCoeffs::operator!= (
             const FilterCoeffs & other ) const  [inline]
```

overloaded != operator for comparing filter coefficients sets

Parameters

other	other FilterCoeffs structure

Returns

returns 0 if coeficients sets are the same, 1 if they are different

operator==()

```
bool JK::FilterCoeffs::operator== (
            const FilterCoeffs & other ) const  [inline]
```

feedback coefficients, equals [1.0] for fir

overloaded == operator for comparing filter coefficients sets

Parameters

other	other FilterCoeffs structure

Returns

returns 1 if coeficients sets are the same, 0 if they are different

The documentation for this struct was generated from the following file:

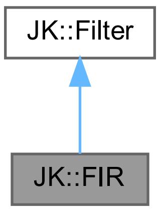
- include/DigitalFilters/filter.hpp

5.4 JK::FIR Class Reference

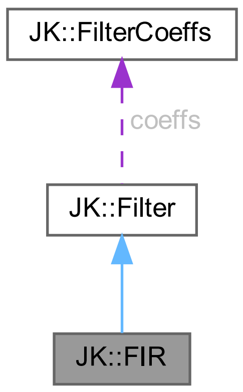
FIR class, FIR is a Finite Impulse Response filter.

```
#include <fir.hpp>
```

Inheritance diagram for JK::FIR:



Collaboration diagram for JK::FIR:



Public Types

- enum class FilterType { **LowPass** , **HighPass** }

  variable containing types od filters for one of the constructors

Public Member Functions

- FIR (const std::vector< double > &b)

  constructor supporting custom coefficients
- **FIR** ()

  default constructor
- FIR (FilterType type, int order, double cutoff_f, double fs)

  advanced constructor that designs a low- or high-pass filters
- FilterCoeffs getCoeffs () const override

  getter for filter coefficients
- void print (std::ostream &os) const override

  helper method for overloaded << operator
- void setCoeffs (const std::vector< double > &b, const std::vector< double > &a) override

  setter for filter's coefficients
- std::vector< double > response (const std::vector< double > &x, const int y_len=-1) override

  method for calculating filter's response to a given signal

Public Member Functions inherited from JK::Filter

- bool operator== (const Filter &other) const

  overloaded == operator for comparing filter type and their coefficients sets
- bool operator!= (const Filter &other) const

  overloaded != operator for comparing filter type and their coefficients sets
- Filter (std::string n, const std::vector< double > &b, const std::vector< double > &a)

class constructor supporting custom coefficients
- **Filter** ()

  default constructor
- virtual ∼**Filter** ()

  virtual destructor, for memory safety when using polymorphism
- void responsePlot (const std::vector< double > &x, const int y_len=-1)

  method for plotting the filter's response to a given signal
- void applyWindow (const Window &win)

  method for applying a window to filter's coefficients
- BodeData bode (double fs, int plot_points=200) const

  method for calculating filter's frequency response
- void bodePlot (double fs, int plot_points=200) const

  method for plotting filter's frequency response, generates 2 pdf files, one for magnitude plot and one for phase plot
- std::string getName () const

  getter for filter's name
- void setName (std::string n)

  setter for filter's name

Additional Inherited Members

Protected Attributes inherited from JK::Filter

- std::string **name**

  filter' name
- FilterCoeffs **coeffs**

  structure holding filter's coefficients

5.4.1 Detailed Description

FIR class, FIR is a Finite Impulse Response filter.

5.4.2 Constructor & Destructor Documentation

FIR() [1/2]

```
JK::FIR::FIR (
             const std::vector< double > & b )
```

constructor supporting custom coefficients

Parameters

b	feedforward coefficients

FIR() [2/2]

```
JK::FIR::FIR (
                FilterType type,
                int order,
                double cutoff_f,
                double fs )
```

advanced constructor that designs a low- or high-pass filters

Parameters

type	desired filter type (LowPass, Highpass)
order	order
cutoff↩ _f	cutoff frequency
fs	sampling frequency

5.4.3 Member Function Documentation

getCoeffs()

```
FilterCoeffs JK::FIR::getCoeffs ( ) const  [override], [virtual]
```

getter for filter coefficients

Returns

returns a structure holding both feedforward and feedback coefficients

Implements JK::Filter.

print()

```
void JK::FIR::print (
                std::ostream & os ) const  [override], [virtual]
```

helper method for overloaded << operator

Parameters

os	output stream

Implements JK::Filter.

response()

```
std::vector< double > JK::FIR::response (
                const std::vector< double > & x,
                const int y_len = -1 )  [override], [virtual]
```

method for calculating filter's response to a given signal

Parameters

x	vector containing signal samples
y_len	desired output length, defaults to length of the signal vector if not specified

Returns

returns a vector with calculated response of the filter

Implements JK::Filter.

setCoeffs()

```
void JK::FIR::setCoeffs (
            const std::vector< double > & b,
            const std::vector< double > & a )  [override], [virtual]
```

setter for filter's coefficients

Parameters

b	feedforward coefficients
a	feedback coefficients, since it is a FIR, they are defaulted to {1.0} no matter the input

Implements JK::Filter.

The documentation for this class was generated from the following files:

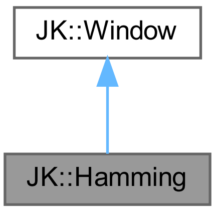
- include/DigitalFilters/fir.hpp
- src/fir.cpp

5.5 JK::Hamming Class Reference

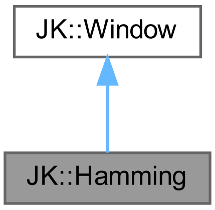
Hamming window class.

```
#include <hamming.hpp>
```

Inheritance diagram for JK::Hamming:



Collaboration diagram for JK::Hamming:



Public Member Functions

- Hamming (size_t N)

  constructor supporting a custom number of coefficients
- Hamming (const Filter &f)

  constructor supporting a number of coefficients based on a referenced Filter class object
- **Hamming** ()

  default constructor

Public Member Functions inherited from JK::Window

- bool operator== (const Window &other) const

  overloaded == operator for comparing windows coefficients sets
- bool operator!= (const Window &other) const

  overloaded != operator for comparing windows coefficients sets
- Window (std::string n)

  constructor supporting custom name
- **Window** ()

  default constructor
- std::vector$<$ double $>$ getCoeffs () const

  getter for window coefficients
- std::string getName () const

  getter for window's name
- void setName (std::string n)

  setter for filter's name
- size_t size () const

  method for getting the window's size (the size of the coefficients vector)
- virtual ∼**Window** ()

  virtual destructor, for memory safety when using polymorphism

Additional Inherited Members

Protected Attributes inherited from JK::Window

- std::vector$<$ double $>$ **coeffs** = {}

  vector containing window coefficients
- std::string **name** = ""

  window's name

5.5.1 Detailed Description

Hamming window class.

5.5.2 Constructor & Destructor Documentation

Hamming() [1/2]

```
JK::Hamming::Hamming (
            size_t N )
```

constructor supporting a custom number of coefficients

Parameters

N	number of coefficients

Hamming() [2/2]

```
JK::Hamming::Hamming (
            const Filter & f )
```

constructor supporting a number of coefficients based on a referenced Filter class object

Parameters

f	Filter class object

The documentation for this class was generated from the following files:

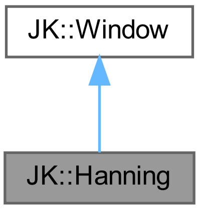
- include/DigitalFilters/hamming.hpp
- src/hamming.cpp

5.6 JK::Hanning Class Reference

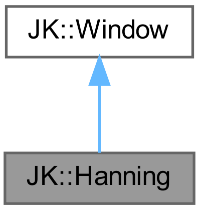
Hanning window class.

```
#include <hanning.hpp>
```

Inheritance diagram for JK::Hanning:



Collaboration diagram for JK::Hanning:



Public Member Functions

- Hanning (size_t N)

  constructor supporting a custom number of coefficients
- Hanning (const Filter &f)

  constructor supporting a number of coefficients based on a referenced Filter class object
- **Hanning** ()

  default constructor

Public Member Functions inherited from JK::Window

- bool operator== (const Window &other) const

  overloaded == operator for comparing windows coefficients sets
- bool operator!= (const Window &other) const

  overloaded != operator for comparing windows coefficients sets
- Window (std::string n)

  constructor supporting custom name
- **Window** ()

  default constructor

- std::vector< double > getCoeffs () const

  getter for window coefficients
- std::string getName () const

  getter for window's name
- void setName (std::string n)

  setter for filter's name
- size_t size () const

  method for getting the window's size (the size of the coefficients vector)
- virtual ∼**Window** ()

  virtual destructor, for memory safety when using polymorphism

Additional Inherited Members

Protected Attributes inherited from JK::Window

- std::vector< double > **coeffs** = {}

  vector containing window coefficients
- std::string **name** = ""

  window's name

5.6.1 Detailed Description

Hanning window class.

5.6.2 Constructor & Destructor Documentation

Hanning() [1/2]

```
JK::Hanning::Hanning (
            size_t N )
```

constructor supporting a custom number of coefficients

Parameters

N	number of coefficients

Hanning() [2/2]

```
JK::Hanning::Hanning (
            const Filter & f )
```

constructor supporting a number of coefficients based on a referenced Filter class object

Parameters

f	Filter class object

The documentation for this class was generated from the following files:

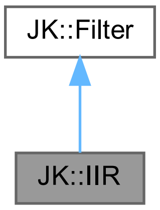
- include/DigitalFilters/hanning.hpp
- src/hanning.cpp

5.7 JK::IIR Class Reference

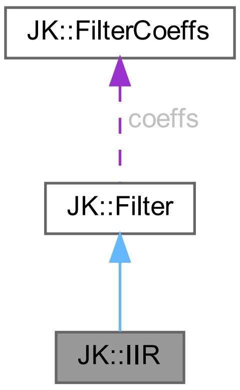
IIR class, IIR is a Infinite Impulse Response filter.

```
#include <iir.hpp>
```

Inheritance diagram for JK::IIR:



Collaboration diagram for JK::IIR:



Public Member Functions

- IIR (const std::vector< double > &b, const std::vector< double > &a)

    constructor supporting custom coefficients
- **IIR** ()

    default constructor
- void print (std::ostream &os) const override

    helper method for overloaded << operator
- FilterCoeffs getCoeffs () const override

    getter for filter coefficients
- std::vector< double > response (const std::vector< double > &x, const int y_len=-1) override

    method for calculating filter's response to a given signal
- void setCoeffs (const std::vector< double > &b, const std::vector< double > &a) override

    setter for filter's coefficients

Public Member Functions inherited from JK::Filter

- bool operator== (const Filter &other) const

    overloaded == operator for comparing filter type and their coefficients sets
- bool operator!= (const Filter &other) const

    overloaded != operator for comparing filter type and their coefficients sets
- Filter (std::string n, const std::vector< double > &b, const std::vector< double > &a)

    class constructor supporting custom coefficients
- **Filter** ()

    default constructor
- virtual ∼**Filter** ()

    virtual destructor, for memory safety when using polymorphism
- void responsePlot (const std::vector< double > &x, const int y_len=-1)

    method for plotting the filter's response to a given signal
- void applyWindow (const Window &win)

    method for applying a window to filter's coefficients
- BodeData bode (double fs, int plot_points=200) const

    method for calculating filter's frequency response
- void bodePlot (double fs, int plot_points=200) const

    method for plotting filter's frequency response, generates 2 pdf files, one for magnitude plot and one for phase plot
- std::string getName () const

    getter for filter's name
- void setName (std::string n)

    setter for filter's name

Additional Inherited Members

Protected Attributes inherited from JK::Filter

- std::string **name**

    filter' name
- FilterCoeffs **coeffs**

    structure holding filter's coefficients

5.7.1 Detailed Description

IIR class, IIR is a Infinite Impulse Response filter.

5.7.2 Constructor & Destructor Documentation

IIR()

```
JK::IIR::IIR (
            const std::vector< double > & b,
            const std::vector< double > & a )
```

constructor supporting custom coefficients

Parameters

b	feedforward coefficients
a	feedback coefficients

5.7.3 Member Function Documentation

getCoeffs()

```
FilterCoeffs JK::IIR::getCoeffs ( ) const  [override], [virtual]
```

getter for filter coefficients

Returns

returns a structure holding both feedforward and feedback coefficients

Implements JK::Filter.

print()

```
void JK::IIR::print (
            std::ostream & os ) const  [override], [virtual]
```

helper method for overloaded << operator

Parameters

os	output stream

Implements JK::Filter.

response()

```
std::vector< double > JK::IIR::response (
            const std::vector< double > & x,
            const int y_len = -1 )  [override], [virtual]
```

method for calculating filter's response to a given signal

Parameters

x	vector containing signal samples
y_len	desired output length, defaults to length of the signal vector if not specified

Returns

returns a vector with calculated reponse of the filter

Implements JK::Filter.

setCoeffs()

```
void JK::IIR::setCoeffs (
            const std::vector< double > & b,
            const std::vector< double > & a )  [override], [virtual]
```

setter for filter's coefficients

Parameters

b	feedforward coefficients
a	feedback coefficients

Implements JK::Filter.

The documentation for this class was generated from the following files:

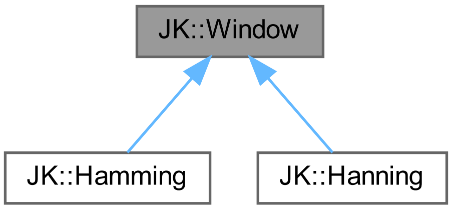
- include/DigitalFilters/iir.hpp
- src/iir.cpp

5.8 JK::Window Class Reference

Window class.

```
#include <window.hpp>
```

Inheritance diagram for JK::Window:



Public Member Functions

- bool operator== (const Window &other) const

  overloaded == operator for comparing windows coefficients sets
- bool operator!= (const Window &other) const

  overloaded != operator for comparing windows coefficients sets
- Window (std::string n)

  constructor supporting custom name
- **Window** ()

  default constructor
- std::vector< double > getCoeffs () const

  getter for window coefficients
- std::string getName () const

  getter for window's name
- void setName (std::string n)

  setter for filter's name
- size_t size () const

  method for getting the window's size (the size of the coefficients vector)
- virtual ∼**Window** ()

  virtual destructor, for memory safety when using polymorphism

Protected Attributes

- std::vector< double > **coeffs** = {}

  vector containing window coefficients
- std::string **name** = ""

  window's name

Friends

- std::ostream & operator<< (std::ostream &os, const Window &w)

  overloaded << operator for printing all coefficients into output stream

5.8.1 Detailed Description

Window class.

5.8.2 Constructor & Destructor Documentation

Window()

```
JK::Window::Window (
             std::string n )
```

constructor supporting custom name

Parameters

n	name

5.8.3 Member Function Documentation

getCoeffs()

```
std::vector< double > JK::Window::getCoeffs ( ) const
```

getter for window coefficients

Returns

returns a vector containing window coefficients

getName()

```
std::string JK::Window::getName ( ) const
```

getter for window's name

Returns

returns a string containing filter's name

operator"!=()

```
bool JK::Window::operator!= (
             const Window & other ) const
```

overloaded != operator for comparing windows coefficients sets

Parameters

other	other Window Class object

Returns

returns 0 if window coeficients sets are the same, 1 if they are different

operator==()

```
bool JK::Window::operator== (
             const Window & other ) const
```

overloaded == operator for comparing windows coefficients sets

Parameters

other	other Window Class object

Returns

returns 1 if window coeficients sets are the same, 0 if they are different

setName()

```
void JK::Window::setName (
             std::string n )
```

setter for filter's name

Parameters

n	name

size()

```
size_t JK::Window::size ( ) const
```

method for getting the window's size (the size of the coefficients vector)

Returns

returns a size_t variable containing window's size

5.8.4 Friends And Related Symbol Documentation

operator<<

```
std::ostream & operator<< (
             std::ostream & os,
             const Window & w )  [friend]
```

overloaded << operator for printing all coefficients into output stream

Parameters

os	output stream
w	Window class object

Returns

returns the output stream

The documentation for this class was generated from the following files:

- include/DigitalFilters/window.hpp
- src/window.cpp

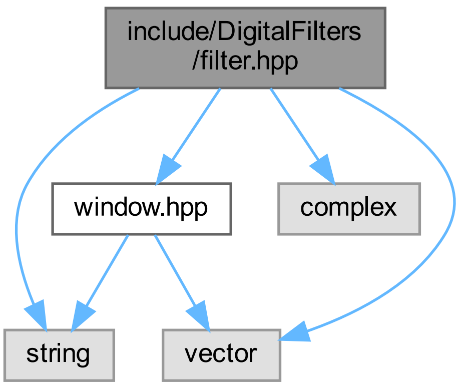
6 File Documentation

6.1 include/DigitalFilters/filter.hpp File Reference

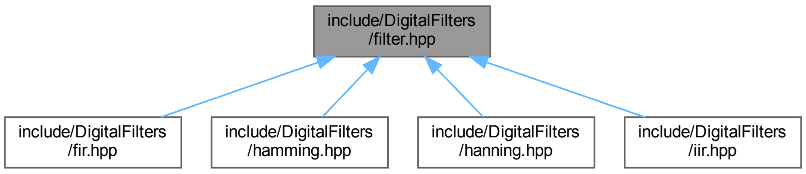
This file handles the base filter class.

```
#include <string>
#include <vector>
#include <complex>
#include "window.hpp"
```

Include dependency graph for filter.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- struct JK::FilterCoeffs

  a structure for holding both feedback (a) and feedforward (b) coefficients
- struct JK::BodeData

  a structure for holding all the parameters needed for Bode plots
- class JK::Filter

  Filter class representing a digital filter.

6.1.1 Detailed Description

This file handles the base filter class.

Author

JK

Date

2026-01-02

6.2 filter.hpp

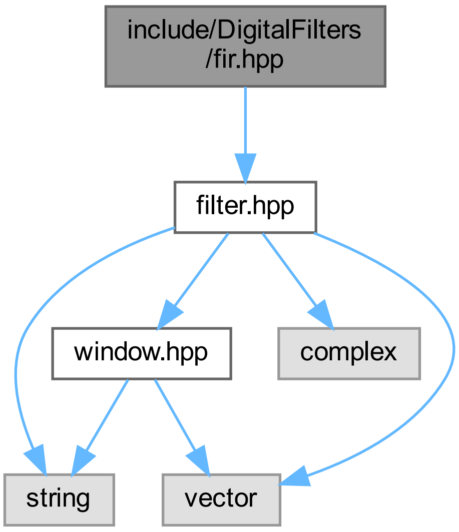
Go to the documentation of this file.

```
#pragma once
#define _USE_MATH_DEFINES
#include <string>
#include <vector>
#include <complex>
#include "window.hpp"

namespace JK{
    struct FilterCoeffs{

        std::vector<double> b;
        std::vector<double> a;

        bool operator==(const FilterCoeffs& other) const {
            return b == other.b && a == other.a;
        }

        bool operator!=(const FilterCoeffs& other) const {
            return b != other.b || a != other.a;
        }
    };
    struct BodeData{
        std::vector<double> freq;
        std::vector<double> mag_db;
        std::vector<double> phase_deg;
    };
    class Filter{
    protected:
        virtual void print(std::ostream& os) const = 0;

        std::string name;

        FilterCoeffs coeffs;
    public:

        bool operator==(const Filter& other) const;

        bool operator!=(const Filter& other) const;

        friend std::ostream& operator«(std::ostream& os, const Filter& f);

        Filter(std::string n, const std::vector<double> &b,  const std::vector<double> &a);

        Filter();

        virtual ~Filter();



        virtual FilterCoeffs getCoeffs() const = 0;

        virtual std::vector<double> response(const std::vector<double> &x, const int y_len = -1) = 0;

        void responsePlot(const std::vector<double> &x, const int y_len = -1);

        void applyWindow(const Window &win);

        BodeData bode(double fs, int plot_points = 200) const;

        void bodePlot(double fs, int plot_points = 200) const;

        virtual void setCoeffs(const std::vector<double> &b, const std::vector<double> &a)  = 0;

        std::string getName() const;

        void setName(std::string n);
    };
}
```

6.3 include/DigitalFilters/fir.hpp File Reference

This file handles the FIR class, which is a child class of Filter.

`#include "filter.hpp"`
Include dependency graph for fir.hpp:



Classes

- class JK::FIR

  FIR class, FIR is a Finite Impulse Response filter.

6.3.1 Detailed Description

This file handles the FIR class, which is a child class of Filter.

Author

JK

Date

2026-01-02

6.4 fir.hpp

Go to the documentation of this file.

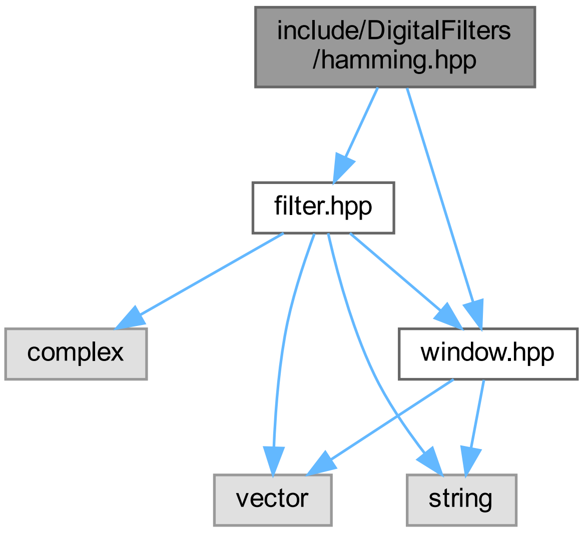
```
#pragma once
#include "filter.hpp"

namespace JK{
    class FIR : public Filter{
    private:
        void design_low_pass(int order, double cutoff_f, double fs);

        void design_high_pass(int order, double cutoff_f, double fs);

    public:
        enum class FilterType {LowPass, HighPass};

        FIR(const std::vector<double> &b);

        FIR();

        FIR(FilterType type, int order, double cutoff_f, double fs);

        FilterCoeffs getCoeffs() const override;

        void print(std::ostream& os) const override;

        void setCoeffs(const std::vector<double> &b, const std::vector<double> &a)override;

        std::vector<double> response(const std::vector<double> &x, const int y_len = -1) override;
    };
}
```

6.5 include/DigitalFilters/hamming.hpp File Reference

This file handles the Hamming class, which is a child class of Window.

```
#include "window.hpp"
#include "filter.hpp"
```

Include dependency graph for hamming.hpp:



Classes

- class JK::Hamming

    Hamming window class.

6.5.1 Detailed Description

This file handles the Hamming class, which is a child class of Window.

Author

JK

Date

2026-01-02

6.6 hamming.hpp

Go to the documentation of this file.

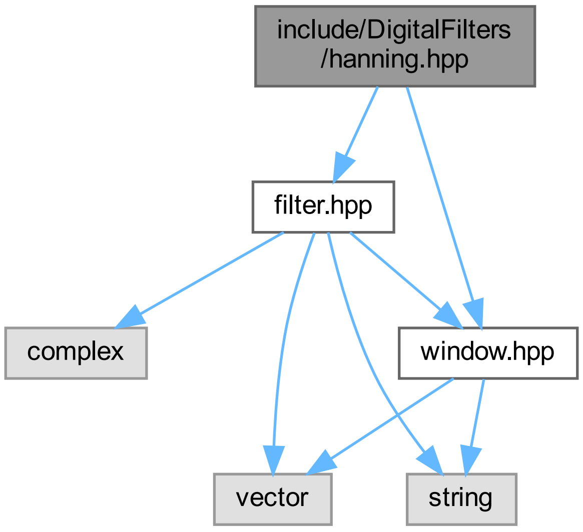
```

#pragma once
#include "window.hpp"
#include "filter.hpp"

namespace JK{
    class Hamming : public Window{
    public:
        Hamming(size_t N);

        Hamming(const Filter &f);

        Hamming();
    };
}
```

6.7 include/DigitalFilters/hanning.hpp File Reference

This file handles the Hanning class, which is a child class of Window.

```
#include "window.hpp"
#include "filter.hpp"
```

Include dependency graph for hanning.hpp:



Classes

- class JK::Hanning

  Hanning window class.

6.7.1 Detailed Description

This file handles the Hanning class, which is a child class of Window.

Author

JK

Date

2026-01-02

6.8 hanning.hpp

Go to the documentation of this file.

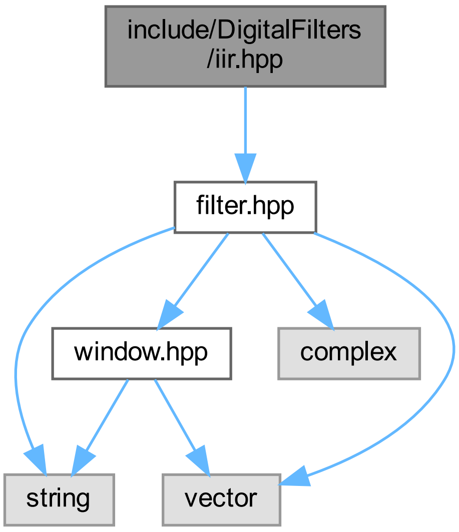
```
#pragma once
#include "window.hpp"
#include "filter.hpp"

namespace JK{
    class Hanning : public Window{
    public:
        Hanning(size_t N);

        Hanning(const Filter &f);

        Hanning();
    };
}
```

6.9 include/DigitalFilters/iir.hpp File Reference

This file handles the IIR class, which is a child class of Filter.

```
#include "filter.hpp"
```

Include dependency graph for iir.hpp:



Classes

- class JK::IIR

  IIR class, IIR is a Infinite Impulse Response filter.

6.9.1 Detailed Description

This file handles the IIR class, which is a child class of Filter.

Author

JK

Date

2026-01-02

6.10 iir.hpp

Go to the documentation of this file.

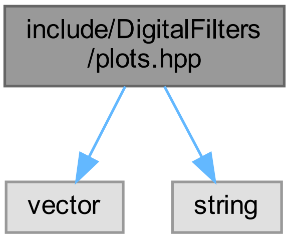
```
#pragma once
#include "filter.hpp"

namespace JK{
    class IIR : public Filter{
    public:
        IIR(const std::vector<double> &b, const std::vector<double> &a);

        IIR();

        void print(std::ostream& os) const override;

        FilterCoeffs getCoeffs() const override;

        std::vector<double> response(const std::vector<double> &x, const int y_len = -1) override;

        void setCoeffs(const std::vector<double> &b, const std::vector<double> &a)override;
    };
}
```

6.11 include/DigitalFilters/plots.hpp File Reference

This file contains plotting functions that use sciplot library to generate pdfs with plots.

```
#include <vector>
#include <string>
```

Include dependency graph for plots.hpp:



Functions

- void JK::plot2d (const std::vector< double > &y, std::string name)

    overloaded 2d plotting function, that defaults the x axis to just sample numbers
- void JK::plot2d (const std::vector< double > &x, const std::vector< double > &y, std::string name)

    overloaded 2d plotting function, that supports custom x axis

6.11.1 Detailed Description

This file contains plotting functions that use sciplot library to generate pdfs with plots.

Author

JK

Date

2026-01-02

6.11.2 Function Documentation

plot2d() [1/2]

```
void JK::plot2d (
            const std::vector< double > & x,
            const std::vector< double > & y,
            std::string name )
```

overloaded 2d plotting function, that supports custom x axis

Parameters

x	x axis vector
y	input vector
name	desired pdf file name

plot2d() [2/2]

```
void JK::plot2d (
            const std::vector< double > & y,
            std::string name )
```

overloaded 2d plotting function, that defaults the x axis to just sample numbers

Parameters

y	input that will be plotted
name	desired pdf file name

6.12 plots.hpp

Go to the documentation of this file.

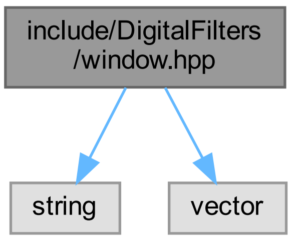
```
#pragma once
#include <vector>
#include <string>

namespace JK{
    void plot2d(const std::vector<double> &y, std::string name);

    void plot2d(const std::vector<double> &x, const std::vector<double> &y, std::string name);
}
```

6.13 include/DigitalFilters/window.hpp File Reference

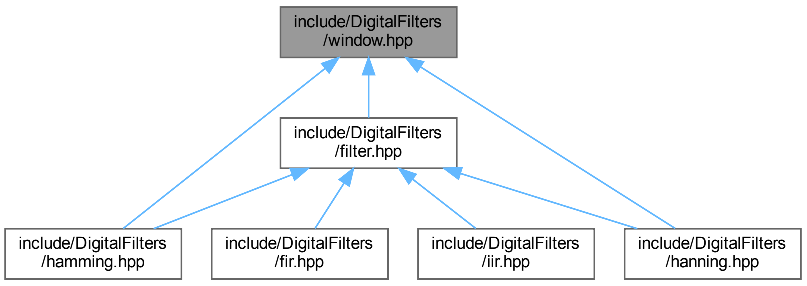
This file handles the Window class.

```
#include <string>
#include <vector>
```

Include dependency graph for window.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class JK::Window

    Window class.

6.13.1 Detailed Description

This file handles the Window class.

Author

JK

Date

2026-01-02

6.14 window.hpp

Go to the documentation of this file.

```

#pragma once
#include <string>
#include <vector>

namespace JK{
    class Window{
    protected:
        std::vector<double> coeffs = {};
        std::string name = "";

    public:
        bool operator==(const Window& other) const;

        bool operator!=(const Window& other) const;

        Window(std:: string n);

        Window();

        friend std::ostream& operator«(std::ostream& os, const Window& w);

        std::vector<double> getCoeffs() const;

        std::string getName() const;

        void setName(std::string n);

        size_t size() const;

        virtual ~Window();
    };
}
```

Index

applyWindow
    JK::Filter, 7

bode
    JK::Filter, 7
bodePlot
    JK::Filter, 8

DigitalFilters Library, 1

Filter
    JK::Filter, 7
FIR
    JK::FIR, 14

getCoeffs
    JK::Filter, 8
    JK::FIR, 15
    JK::IIR, 23
    JK::Window, 26
getName
    JK::Filter, 8
    JK::Window, 26

Hamming
    JK::Hamming, 18
Hanning
    JK::Hanning, 20

IIR
    JK::IIR, 23
include/DigitalFilters/filter.hpp, 28, 29
include/DigitalFilters/fir.hpp, 30, 31
include/DigitalFilters/hamming.hpp, 32, 33
include/DigitalFilters/hanning.hpp, 33, 34
include/DigitalFilters/iir.hpp, 34, 35
include/DigitalFilters/plots.hpp, 36, 37
include/DigitalFilters/window.hpp, 37, 39

JK::BodeData, 4
JK::Filter, 5
    applyWindow, 7
    bode, 7
    bodePlot, 8
    Filter, 7
    getCoeffs, 8
    getName, 8
    operator!=, 8
    operator<<, 11
    operator==, 9
    print, 9
    response, 9
    responsePlot, 10
    setCoeffs, 10
    setName, 10
JK::FilterCoeffs, 11
    operator!=, 11
    operator==, 12
JK::FIR, 12
    FIR, 14
    getCoeffs, 15
    print, 15
    response, 15
    setCoeffs, 16
JK::Hamming, 16
    Hamming, 18
JK::Hanning, 18
    Hanning, 20
JK::IIR, 21
    getCoeffs, 23
    IIR, 23
    print, 23
    response, 23
    setCoeffs, 24
JK::Window, 24
    getCoeffs, 26
    getName, 26
    operator!=, 26
    operator<<, 28
    operator==, 27
    setName, 27
    size, 27
    Window, 26

operator!=
    JK::Filter, 8
    JK::FilterCoeffs, 11
    JK::Window, 26
operator<<
    JK::Filter, 11
    JK::Window, 28
operator==
    JK::Filter, 9
    JK::FilterCoeffs, 12
    JK::Window, 27

plot2d
    plots.hpp, 37
plots.hpp
    plot2d, 37
print
    JK::Filter, 9
    JK::FIR, 15
    JK::IIR, 23

response
    JK::Filter, 9
    JK::FIR, 15
    JK::IIR, 23
responsePlot
    JK::Filter, 10

setCoeffs

JK::Filter, 10
JK::FIR, 16
JK::IIR, 24
setName
JK::Filter, 10
JK::Window, 27
size
JK::Window, 27

Window
JK::Window, 26