Faculty of Informatics, Masaryk University in Brno
Centre for Biomedical Image Analysis

FTutor1D

User manual

October 11, 2016

CBIA
CENTRE FOR BIOMEDICAL
IMAGE ANALYSIS

Contents

1 Preface

Dear reader,

thank you for expressing your interest in Fourier transform and FTutor1D, a free software for visualisation of one dimensional discrete Fourier transform. I hope my application will help you undestand the transform in detail: see the results of applying the transform to signals of different lengths, understand the meaning of the Fourier coefficients, experience the impact of modifying the signal in the frequency spectrum and comparing the result of inverse transform with the original input signal.

This application was created as a semestral project at Centre of Biomedical Image Analysis (CBIA) at the Faculty of Informatics, Masaryk University in Brno (FI MU). The aim was to provide an educational tool to help teachers explain and to help students understand the transform. The application was required to respond in real time, have simple but expressing user interface, be multiplatform and extensible in terms of adding custom language versions and loading custom signals without the need of modyfing the source code. I believe I achieved these goals and you will enjoy the time spent by using FTutor1D.

This document shall explain how to use the FTutor1D. Technical details on how the application is programmed are covered in another document, the Technical Overview, which should be available together with the source codes. The first chapter explains how to run the application for the first time, after either downloading the binary for Windows operating systems, or downloading the source codes to compile and run the application on Linux. The compulsory directory structure, which must be preserved, is also explained there. The second chapter explains the user interface of the application main window. The next chapter is focused on manipulating the signals: signal creation, loading and storing, but also modification of the signal in frequency domain and editing the original signal in the time domain. The fourth chapter is about filtering, what filters are available in FTutor1D and how to apply them. The fifth chapter explains how to define custom language versions for the application.

2 First execution

In this chapter, I will explain how to run the application properly.

2.1 Starting with Windows binary

I assume you have downloaded the Windows binary, which is provided as a zip package. Unzip the archive using an arbitrary tool. There is no need to install any software, clicking *FTutor1D.exe* should be enough to run the application.

2.2 Building from source codes

To build the application from the source codes, download and install the Qt framework. Qt is available on `https://www.qt.io/download/`. You can use the Qt for private use and open-source distribution free of charge. If you want to download the single packages using the repository (common way in most of Linux distributions), download the following (with all dependencies): `libqt5core5a, libqt5widgets5, libqt5printsupport5, libqt5xml5, libqt5xmlpatterns5-dev`. You also need to have a C++ compiler and either *cmake* or *qmake* (packages `cmake` or `qt5-qmake`).

Open the command line, navigate to the root directory with the source codes and type the following:

1. `qmake FTutor1D.pro` or `cmake .`
2. `make`

In case the step one failed, provide the path to the missing libraries, or install the missing packages.

2.3 Compulsory directory structure

You must preserve the directory structure as listed in Figure 1 to have FTutor1D run correctly.

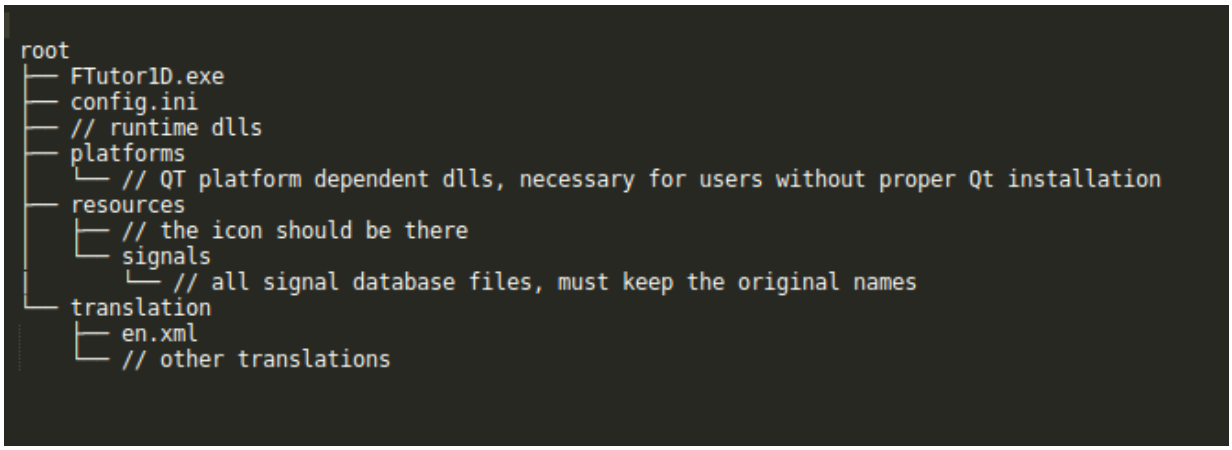

Figure 1: Directory tree for FTutor1D.

The `resources` folder is the folder where all the predefined signals and images are stored, the `translation` folder is where the language versions are loaded from. The `platforms` folder is important only for Windows users without Qt installation. Executable name, *FTutor1D.exe*, is the file to run, it should be called the same on Linux, just without the *.exe* suffix. You probably have *config.ini* file missing. This is OK, it is generated after you first run the application. It searches for the resources at the predefined paths and puts those paths into the *config.ini*. If you do not want to keep the described directory structure, open *config.ini* file in a text editor and change the paths to the application icon, to the folder with translations and to the predefined signals. However, modified *config.ini* must still accompany the executable in the directory so that it could be loaded.

3 User interface

After running the application, you are welcomed by the main window. The window is mostly empty at first, until you create or load a signal (this is explained in the next chapter). The Figure 2 shows the appearance of the main window, with already having loaded one of the signals.

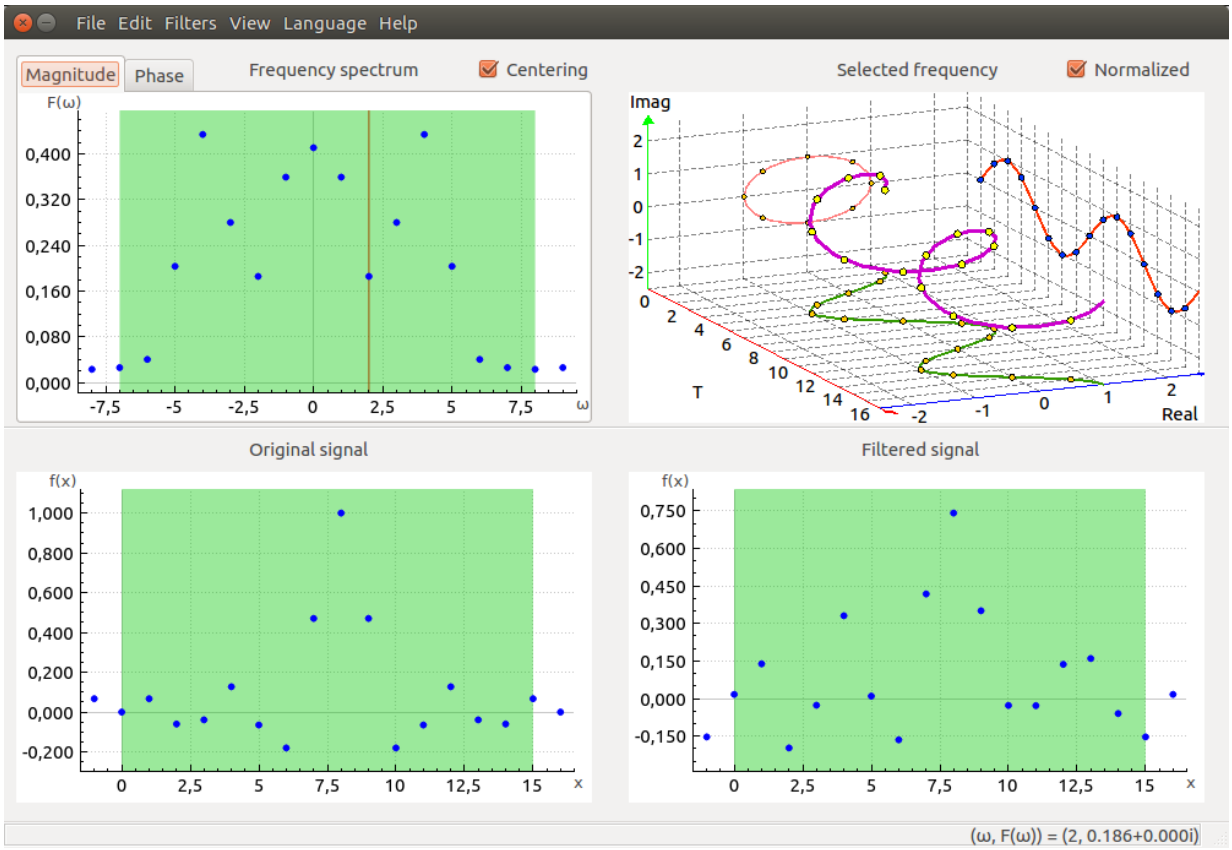

Figure 2: Main Window of FTutor1D.

As you can see, there are four plots in the main window. Read them in clockwise order, starting in the lower left corner.

The first plot is the original signal graph. This is the signal you have created or loaded.

The next graph, in the upper left corner, displays the original signal transformed in the frequency spectrum. There are two tabs: in one you can visualize magnitudes of the computed Fourier coefficients (also a scale of the corresponding basis function), in the other there is a plot of phases (also the time shift of the basis function).

The next graph displays the basis function corresponding to the point in the fourier spectrum you are currently pointing at with the mouse. This is a 3D graph, where x-axis (red) denotes time, y-axis (green) the imaginary compound and the z-axis (blue) the real compound of the Fourier basis function. The purple curve is the basis function itself, the other curves are projections into imaginary, real and time planes.

Last graph, in the lower-right corner, shows the filtered signal. This is a time domain graph, which was created by applying the inverse transform to the Fourier coefficients. Modifying signal in the frequency spectrum (magnitude, phase of a single or more points or applying a filter) you can see the immediate effects on the resulting signal.

Below the graphs, in the status bar, you can see the value of the point you are currently pointing at with the mouse. This applies to all the graphs except for the basis function visualization.

At the top, you can find the application main menu. The action File provides option to create, load and store a signal or to exit the application. In Edit action, you can choose to undo one edit or to reload the original signal (all changes in frequency domain will be stashed). In Filters action, you can choose and apply one of ideal, Gaussian or Butterworth filters. You can use View menu to rescale all the graphs to the default scale, enable displaying with lines (simple connecting the discrete points into polylines) or to forbid automatic scaling. Language menu is populated based on available translation files, select your own language to localize the application properly. Help menu provides a small guide and information about the application.

4 Manipulating signals

This section provides a step-by-step guide on how to manipulate with signals in FTutor1D.

4.1 Loading a predefined signal

FTutor1D provides a small library of 16 predefined signals that you can load and investigate. These are eight discrete functions, provided in both 16 and 32 samples versions:

- sinus
- cosinus
- Gaussian
- unit impulse
- step function
- ramp function
- constant function
- sinc function

To load one of the predefined functions, navigate to the menu and click *File* → *Open predefined signal*. A new window will be opened, as illustrated in 4. The content of the window is only visible if you have correctly set up the directory tree, as explained in the *First execution* chapter. Choose the function you want to load, select 16 or 32 samples version hitting the corresponding radio button under the graph and click the graph to load the function. If you changed your mind and wish to close the window without loading the function, click *Cancel* button.

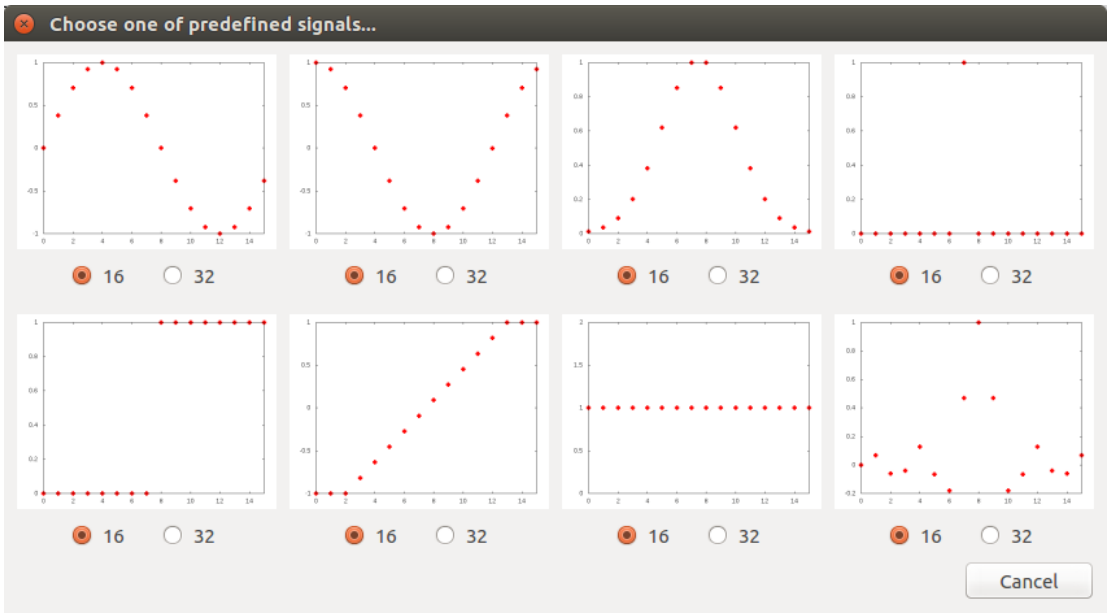

Figure 3: FTutor1D signals library

4.2 Loading a custom signal

If you want to load your own signal file, in the menu click *File* and *Open*. Choose the file you want to load using the system dialog.

4.2.1 Signal file format

The file format for FTutor1D signals is extremely simple, as it can only contain the coordinates of discrete points. On one line, there can be the coordinate of only one point, divided by white characters (for example spaces). Lines starting with `#` are considered comments and are skipped. Nothing else is supported, there cannot be more coordinates on one line or coordinates followed by a comment. If the file loader finds an unexpected sequence of characters, it refuses to load the signal. The Figure **??** shows a sample file, which is used for the 16 samples discrete cosinus. Also note that the signal you want to load must be uniformly sampled in the x-axis.

```
1   0   1.0000e+00
2   1   9.2388e-01
3   2   7.0711e-01
4   3   3.8268e-01
5   4   6.1232e-17
6   5  -3.8268e-01
7   6  -7.0711e-01
8   7  -9.2388e-01
9   8  -1.0000e+00
10  9  -9.2388e-01
11  10  -7.0711e-01
12  11  -3.8268e-01
13  12  -1.8370e-16
14  13   3.8268e-01
15  14   7.0711e-01
16  15   9.2388e-01
17
```

Figure 4: Discrete cosinus with 16 samples. Note that the first number denotes the line number, it is not part of the file.

4.3 Saving a signal

Click *File* and *Save*, if you want to save the signal. A system dialog is opened, select a directory and a name for your signal. Submit the dialog to finish the action. The signal, which is saved, is always the one displayed in the Filtered graph in the lower-right corner. You can later use the signal you have saved and load it into the programme.

4.4 Creating a new signal

If you want to create your own signal directly using FTutor1D, click *File* and *New*. All the graphs are inactive, except for what used to be the original signal graph, which is now highligted by the red border. This graph is now in so called *Edit mode*. By clicking the left mouse button, you can add a point. Clicking the right mouse button removes a point.

Pressing the left mouse button and moving the mouse you can drag the point up or down.

The important restriction we have made is that the uniform sampling must be preserved in order for the Fourier Transform to make sense. The default spacing is set to 1, so you can only add points at integer coordinates. If you skip some positions, say that you have added a point at position 1 and then at position 6, all the positions between (2,3,4,5) are automatically added and are set to zero. Similarly, you can only delete points from the beginning or the end of the signal. If you try to remove the point at position, which is not the first nor the last, it is not removed, but set to zero. If you wish to accept the signal you have created, click *Finish editing*. If you want to abandon the changes you have made, click *Cancel*.

4.5 Editing the original signal

You can also edit the original signal that you have previously created or you have loaded from a file. Click the original signal graph using the right mouse button and choose *Open edit mode*. You are already familiar with the edit mode if you have read the previous section. Note that if the original signal is uniformly sampled with different distance between the two samples, this is preserved and you can only add points to multiples of the sampling distance.

4.6 Modifying the signal in the frequency domain

There is no need to open any special mode to edit magnitude and phase signal, it is always enabled. Press the left mouse button while being over one of the points and drag it to change its y-coordinate. In the magnitude graph, you cannot drag the point to values below zero. Because magnitude is defined as the absolute value of the complex number (and therefore is always positive by the definition), it would not make any sense. Also, in the phase graph, you can only set the value of a point in the interval $[-\pi, \pi]$.

4.7 Going a step back

FTutor1D maintains the history of your edits to signals. The point at which the history starts recording is when you load a signal or submit the edit mode. Using menu *Edit* and *Revert to original*, you can empty the history and recompute the frequency spectrum (and also filtered signal) from the signal, which is currently displayed in the original signal graph. Menu *Edit* $\rightarrow$ *Undo* is for going one step back. It works in both *Edit mode*, when creating or editing a signal, or with Frequency spectrum changes (both changes made by manipulating the signal with mouse or applying a filter).

5 Filtering

FTutor1D enables you to apply ideal, Gaussian or Butterworth filters to the frequency domain signals and display the result in the filtered signal graph. Both low and high pass versions of all filters are available. The low pass filters have the effect of smoothening the input signal, keeping the low frequencies and suppressing the high frequencies. The high pass filters work the other way round, keeping the details and suppressing the low frequencies.

Filters in the frequency domain are applied using the simple multiplication. A new signal of the same length as the input signal, also called filter, is generated. Values of filter and magnitude on the corresponding positions are multiplied to obtain the result of filtering. To emphasize the fact of applying the filters to the magnitude signal, Filters menu is unavailable if the frequency specturum is switched to the *Phase* graph. When the filter is applied, both previous signal and its change are displayed in the graphs, as you can see in 5. Manipulating with the graph removes the shadow of the previous signal.

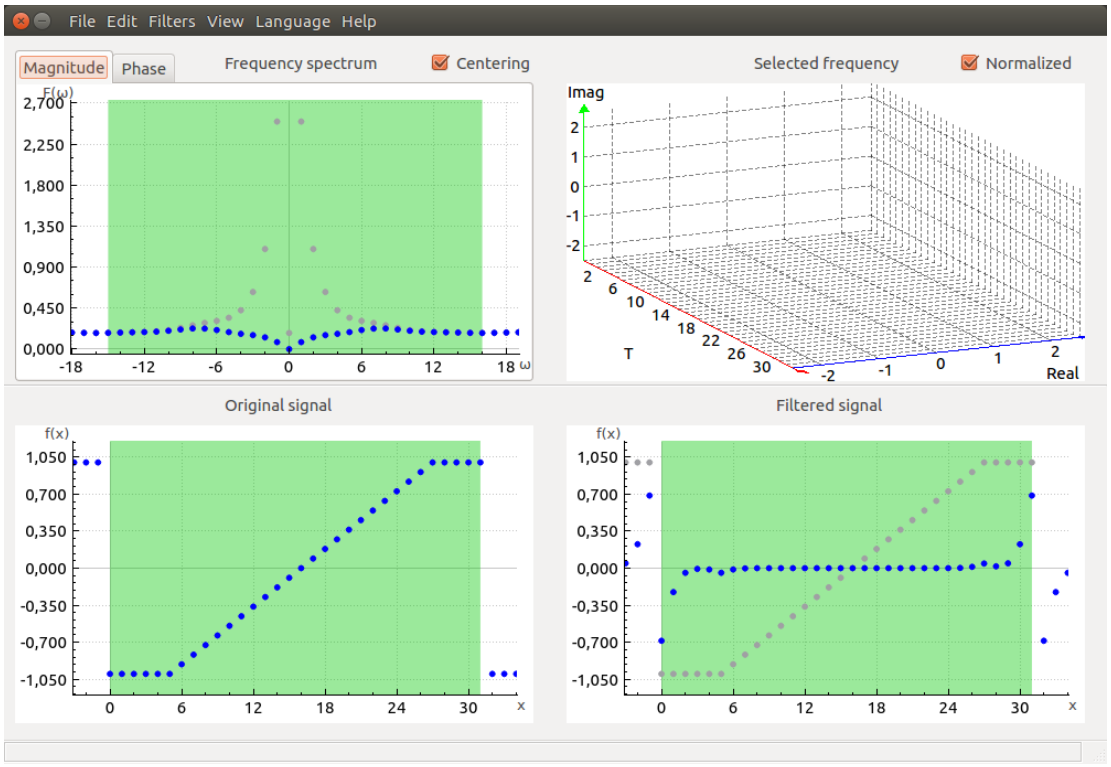

Figure 5: Gaussian high-pass filter applied to ramp signal. Both modified coefficients (blue) and shadow of the signal before applying the filter (grey) are displayed in the frequency spectrum and the filtered signal graph.

5.1 Ideal low-pass filter

The ideal low-pass filter (ILPF) is defined as follows:

$$G(\omega) = \begin{cases} 1, & \text{if } \omega \leq \omega_0 \\ 0, & \text{otherwise,} \end{cases}$$

where ω_0 denotes the cut-off frequency (the last frequency to be kept).

To apply the ILPF, click *Filter* and *Ideal low-pass*. Set the ω_0 by using the slider and click *OK* to apply. Use *Cancel* button to abort the filtering.

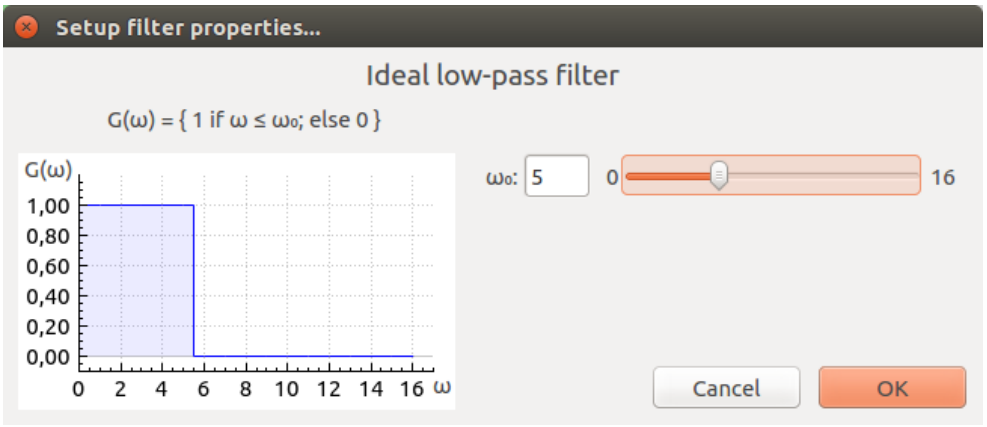

Figure 6: The dialog for setting the ideal low-pass filter.

5.2 Ideal high-pass filter

The ideal high-pass filter (IHPF) is defined as follows:

$$G(\omega) = \begin{cases} 1, & \text{if } \omega \geq \omega_0 \\ 0, & \text{otherwise,} \end{cases}$$

where ω_0 denotes the cut-off frequency (the first frequency to be kept).

To apply the IHPF, click *Filter* and *Ideal high-pass*. Set the ω_0 by using the slider and click *OK* to apply. Use *Cancel* button to abort the filtering.

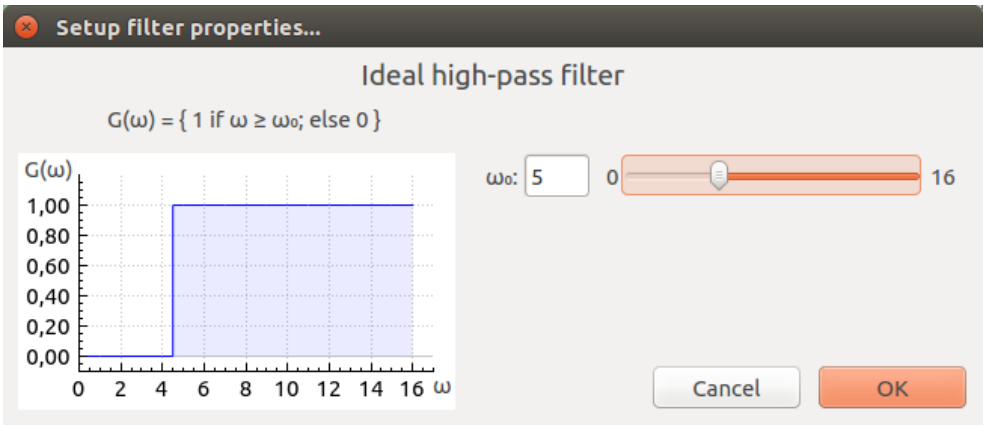

Figure 7: The dialog for setting the ideal high-pass filter.

5.3 Ideal band-pass filter

The ideal band-pass filter (IBPF) is a combination of ILPF and IHPF. Using two cut-off frequencies, it is defined asfollows:

$$G(\omega) = \begin{cases} 1, & \text{if } \omega_1 \leq \omega \leq \omega_2 \\ 0, & \text{otherwise,} \end{cases}$$

where ω_1 is the first frequency to keep and the ω_2 is the first frequency to keep.

To apply the IBPF, click *Filter* and *Band-pass*. Set the ω_1 and ω_2 by using the sliders and click *OK* to apply. Use *Cancel* button to abort the filtering.

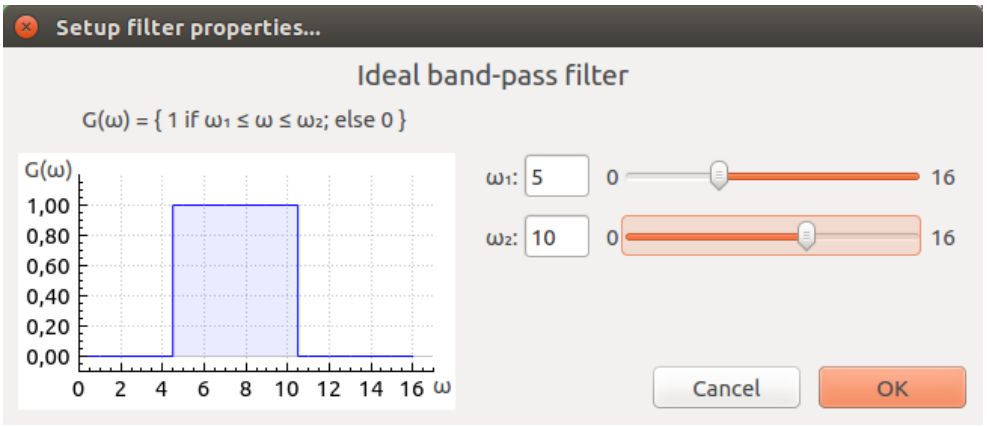

Figure 8: The dialog for setting the ideal band-pass filter.

5.4 Gaussian low-pass filter

The Gaussian low-pass filter (GLPF) is defined as follows:

$$G(\omega) = \begin{cases} 0, & \text{if } \omega_0 = 0 \\ e^{\frac{-\omega^2}{2\omega_0^2}}, & \text{otherwise.} \end{cases}$$

This filter (and the following filters) is not binary, but defines weights with which to multiply each frequency of the input signal. Parameter ω_0 affects the width of the "Gaussian hat" function, the bigger it ω_0, the wider the Gaussian. Similarly to ideal filtes, it can also be regarded as the cut-off frequency: if $G(\omega) = \omega_0$, the GLPF is down to 0.607 of its maximum value [1]. Also, ω_0^2 is the variance of the Gaussian.

To apply the GLPF, click *Filter* and *Gaussian low-pass*. Set the ω_0 by using the slider and click *OK* to apply. Use *Cancel* button to abort the filtering.

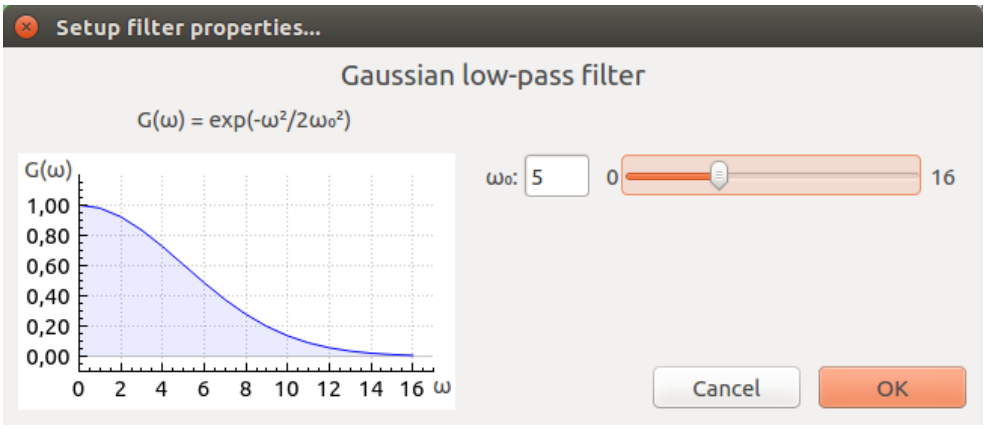

Figure 9: The dialog for setting the Gaussian low-pass filter.

5.5 Gaussian high-pass filter

The Gaussian high-pass filter (GHPF) is defined as follows:

$$G(\omega) = \begin{cases} 1, & \text{if } \omega_0 = 0 \\ 1 - e^{\frac{-\omega^2}{2\omega_0^2}}, & \text{otherwise.} \end{cases}$$

GHPF is basically a constant unit signal, from which the GLPF is subtracted. Parameter ω_0 is the cut-off frequency, as explain at GLPF.

To apply the GHPF, click *Filter* and *Gaussian high-pass*. Set the ω_0 by using the slider and click *OK* to apply. Use *Cancel* button to abort the filtering.

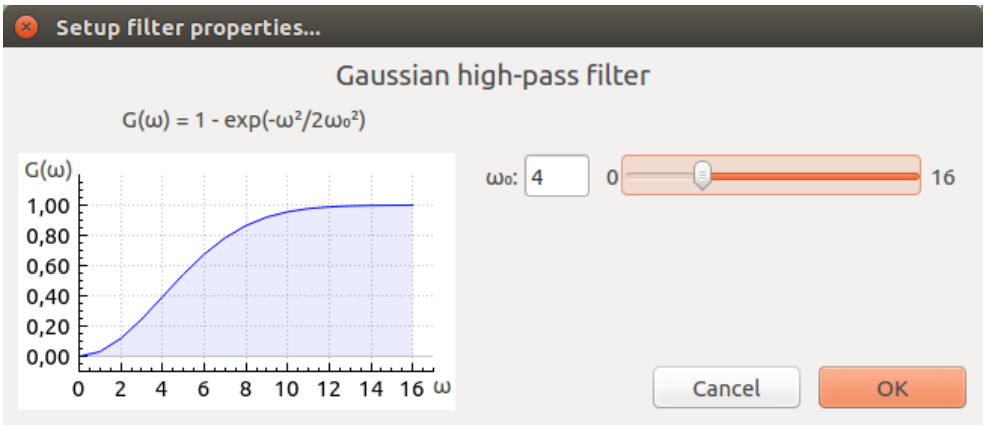

Figure 10: The dialog for setting the Gaussian high-pass filter.

5.6 Butterworth low-pass filter

The Butterworth low-pass filter (BLPF) is defined as follows:

$$G(\omega) = \begin{cases} 0, & \text{if } \omega_0 = 0 \\ \dfrac{1}{1 + \left(\dfrac{\omega}{\omega_0}\right)^{2n}}, & \text{otherwise.} \end{cases}$$

BLPF allows us to control the steepness of the filter fuction with the parameter n, which is the order of the Butterworth filter. ω_0 is the cut-off frequency. When $F(\omega) = \omega_0$, the filter is down to 50% of its maximum value of 1 [1].

To apply the BLPF, click *Filter* and *Butterworth low-pass*. Set the ω_0 and n and click *OK* to apply the filter. Use *Cancel* button to abort the filtering.

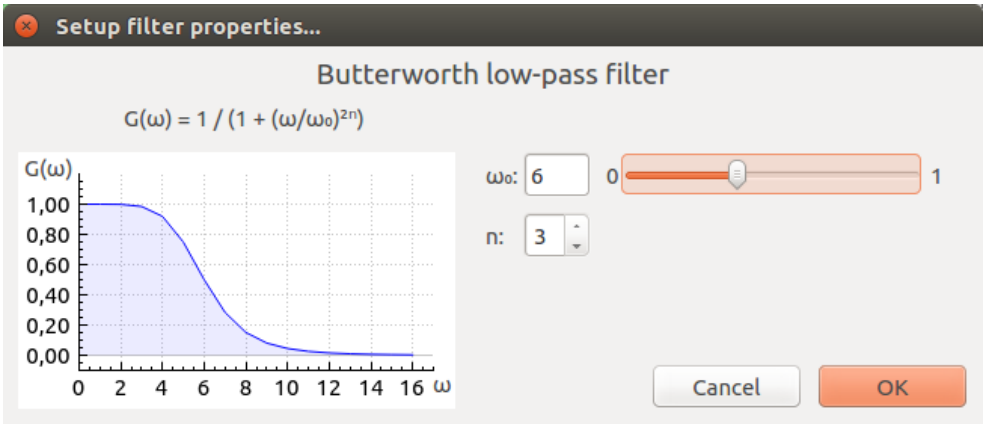

Figure 11: The dialog for setting the Butterworth low-pass filter.

5.7 Butterworth high-pass filter

The Butterworth high-pass filter (BHPF) is defined as follows:

$$G(\omega) = \begin{cases} 1, & \text{if } \omega_0 = 0 \\ \dfrac{1}{1 + \left(\dfrac{\omega_0}{\omega}\right)^{2n}}, & \text{otherwise.} \end{cases}$$

Similarly as in BLPF, n is the order of the Butterworth filter and ω_0 is the cut-off frequency.

To apply the BHPF, click *Filter* and *Butterworth high-pass*. Set the ω_0 and n and click *OK* to apply the filter. Use *Cancel* button to abort the filtering.

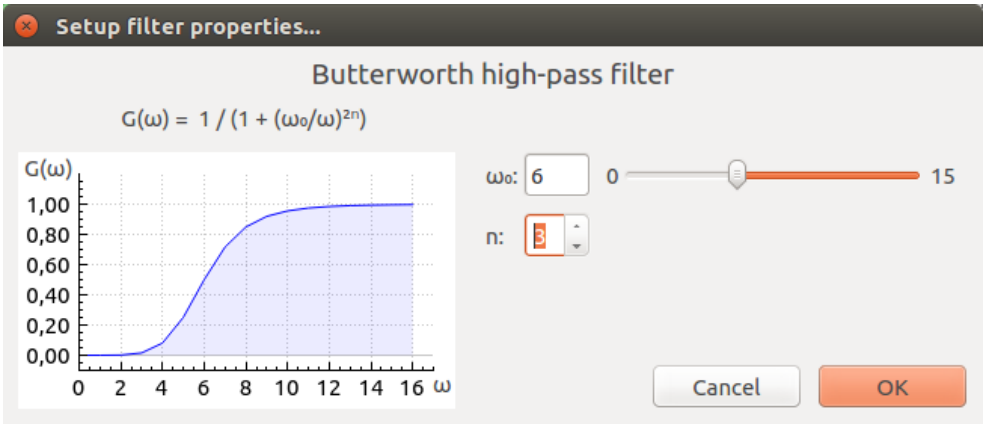

Figure 12: The dialog for setting the Butterworth high-pass filter.

6 Visualization options

This chapter explains how to manipulate with the visible area of the graphs. Note that the actions applied to the original signal graph have the same effect on the filtered signal graph. Similarly for magnitude and phase graphs.

6.1 Zooming

To zoom the graph, roll the mouse button over the graph. X and Y axes zoom simulateously. When zooming out, at first both axes zoom together, but later on, because the range in the X axis is limited (because of the performance reasons), only Y axis zooms out. The best way to return back is to use the *Default scale* option.

If you want to zoom just one of the axes, click the desired axis with the left mouse button first. The axis turns blue, which means this mode is enabled. You can cancel it by clicking somewhere else in the graph.

6.2 Translating the visible area

To translate the visible area, press the left mouse button somewhere over the graph (be careful not to select any point). Moving the mouse you edit the visible area of the graph. You finish the action by releasing the button.

Translating the visible area works differently in the *Edit mode*. Left mouse button could not be used, because it addds new points. Therefore, press the mouse middle button instead of left mouse button to achieve the same functionality.

6.3 Default-scaling

Default scaling rescales the graph to just contain only the original part of the signal. You could notice that all the signals are copied. This is because of the periodicity of the discrete Fourier transform. If you want to apply the default scale to all the graphs, use menu *View* and *Default scale*. If you want to apply the default scale individually, use the mouse and right click on the graph to show its pop-up menu. Select the option *Default scale for this graph*.

6.4 Displaying with lines

Sometimes it is difficult to guess the original function seeing only the mess of discrete points. Displaying with lines connects these discrete points into a polyline, so it is easier for a human to read the graph. To enable displaying with lines in all the graphs, use menu *View* and *Display with lines*. To enable it individually, right click the graph to show the pop-up menu and select *Display with lines in this graph*.

6.5 Automatic scaling

When you load a new signal, create or edit a signal, or apply a filter, all the graphs are automatically default-scaled. This behaviour might not be appreciated by the user. To enable or to disable the autoscaling for all the graphs, use menu *View* and *Allow autoscaling*. You can toggle autoscaling for the individual graphs also, by right clicking on the graph to call its pop-up menu and selecting *Automatic scaling*.

6.6 Centering and normalization

6.6.1 Centering

The raw result of the Fourier transform is a signal, in which the positions denote frequencies. At position 0 we have the zero frequency, then the frequencies increase until the middle of the signal, where we have the highest frequency (we are not able to cover higher frequencies due to the sampling), and then the frequencies drop again. Normally we have the highest coefficients in the lower frequencies and smaller in the higher ones, which would mean the graph would look like the letter U. You can display such signal when disabling the centering. When the centering is enabled, the visible area is shifted to the side, so that the higher coefficients are in the middle. Such display is usually easier to read.

6.6.2 Normalization

In the Selected frequency plot, you can see the shape of the basis fourier functions. However, all of these have the magnitude equal to 1 and phase equal to zero, so they do not reflect the computed coefficients in the frequency spectrum. By disabling the Normalization (unchecking the Normalized checkbox) you can see the basis functions properly scaled and shifted.

7 Language versions

FTutor1D offers the option to use your own language versions. These are in form of special XML files and are located inside the `translation` folder. The name of the file is irrelevant, more important is the content, which must be well-formed. The best approach to define your own language version is to copy and translate one of the files, which are provided by default. You can open the .xml files in an arbitrary text editor.

```
1  <?xml version="1.0" encoding="utf-8"?>
2
3  <FTutor1DLocalization xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:
   noNamespaceSchemaLocation="localization.xsd"
4      languageName="English">
5
6      <window name="MainWindow">
7          <title>FTutor1D: 1D Fourier Transform Tutor</title>
8          <UIElement name="menuBar">
9              <UIElement name="menuFile">
10                 <text>File</text>
11                 <UIElement name="actionOpen">
12                     <text>Open</text>
13                     <UIElement name="FileDialog">
14                         <text>Open signal...</text>
15                     </UIElement>
16                 </UIElement>
17                 <UIElement name="actionOpenPredefined">
18                     <text>Open predefined signal</text>
19                 </UIElement>
20                 <UIElement name="actionSave">
21                     <text>Save</text>
22                     <UIElement name="FileDialog">
23                         <text>Save signal...</text>
24                     </UIElement>
25                 </UIElement>
```

Figure 13: The beginning of the English translation file.

Figure 13 illustrates the beginning of the English localization file. When defining your own localization, make sure to translate (change) only the following:

- `languageName` attribute value of the `FTutor1DLocalization` tag, this is the name that is listed in the Languages menu in the application
- the text enclosed in the `text` tags
- the text enclosed in the `title` tags

Everything else must remain the same.

References

[1] GONZALEZ, Rafael C.; WOODS, Richard E. Digital image processing. Prentice hall Upper Saddle River, NJ, USA: 3rd edition. 2008.

A Discrete Fourier Transform

The discrete Fourier transform of the signal f of length N is defined as follows:

$$F(k) = f \cdot \varphi_k = \frac{1}{\sqrt{N}} \sum_{m=0}^{N-1} f(m) e^{-\frac{2\pi i m k}{N}},$$

where $i = \sqrt{-1}$. The transform is the projection of the signal into the Fourier basis functions. Fourier basis functions are of the form:

$$\varphi_k(m) = \frac{1}{\sqrt{N}} e^{\frac{2\pi i m k}{N}} = \frac{1}{\sqrt{N}} \left(\cos\left(\frac{2\pi m k}{N}\right) + i \sin\left(\frac{2\pi m k}{N}\right) \right)$$

The inverse transform is defined as:

$$f(m) = F \cdot \bar{\varphi_k} = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} F(k) e^{\frac{2\pi i m k}{N}},$$