The Large Time Frequency Analysis Toolbox

Torino, 10 September 2023

What is LTFAT?

- A free (GPLv3) toolbox for the
  - Study of
  - Education on
  - Construction of

Time Frequency representations

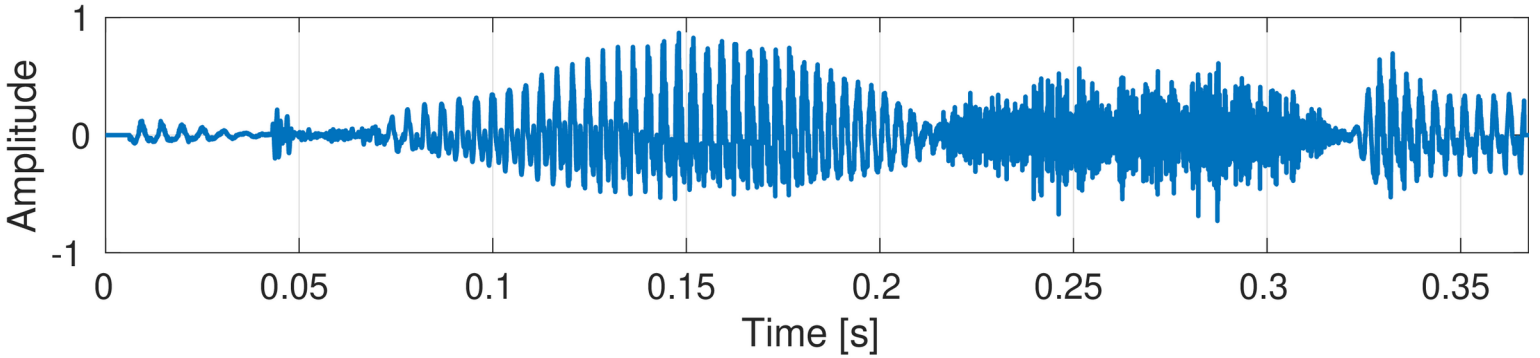
  - And their applications, such as e.g.
    - Tonal-Transient-Residual separation
    - Time-variant filtering
    - Phase reconstruction and vocoding (in PHASERET)

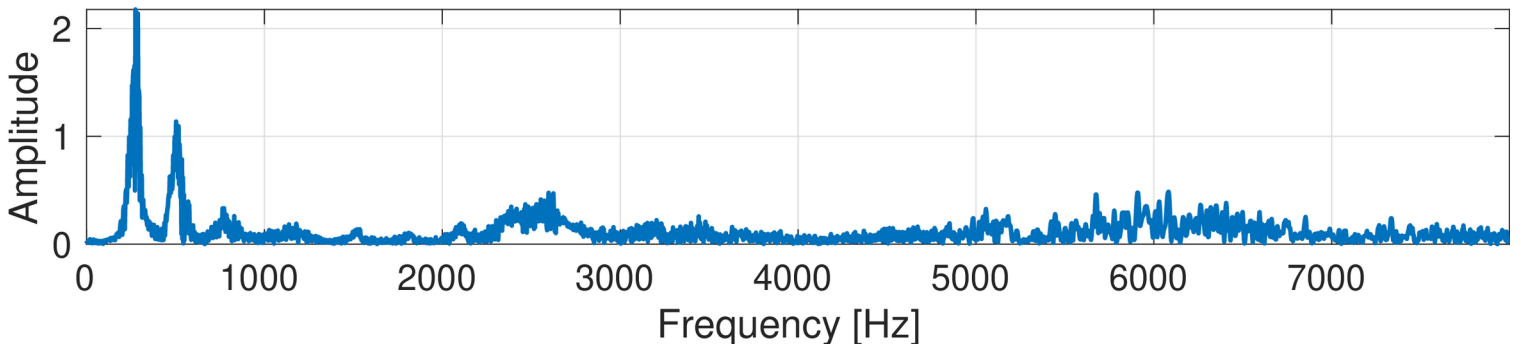
- In Matlab/Octave, with a C/C++-backend

Overview

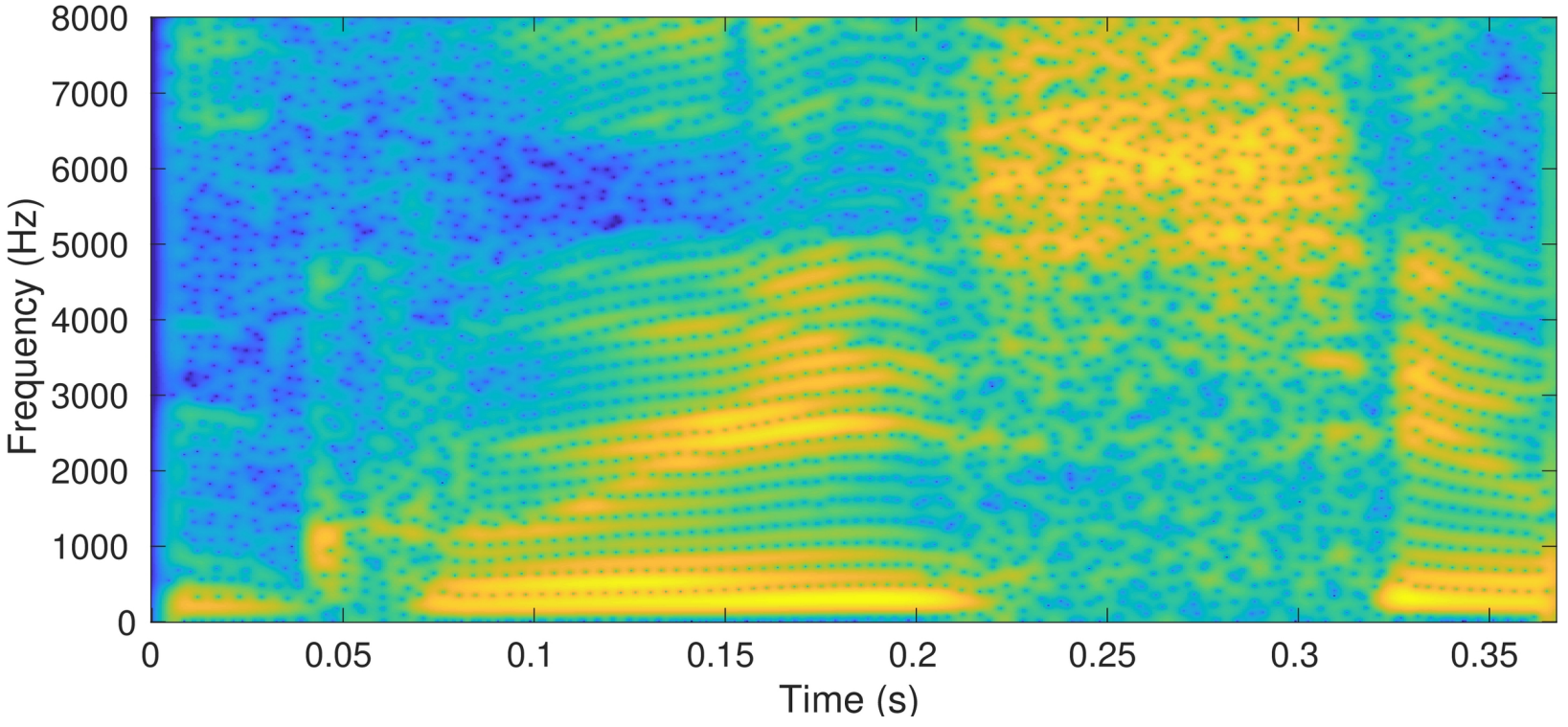
- **Introduction**
  - Why a toolbox for time-frequency representations?
  - LTFAT Structure and Resources
  - How to install
- **Filter banks**
  - Filter banks and frames: why use LTFAT?
  - Designing an invertible constant-Q filter bank
  - Designing an invertible gammatone filter bank
  - Some further processing: reassignment and phase retrieval

Time and Frequency representations





Time ~~and~~ Frequency representation~~s~~



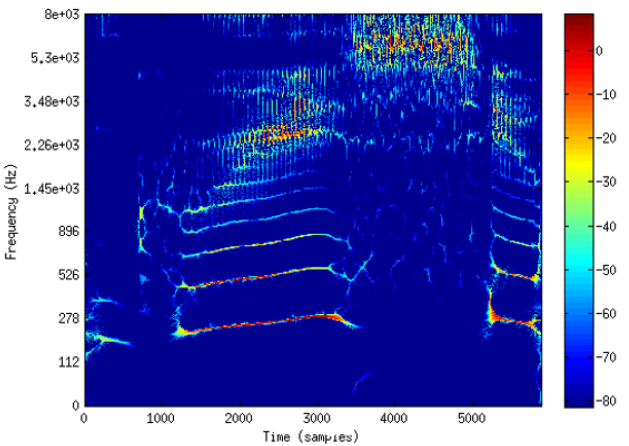
LTFAT Resources - Homepage

ltfat.org

LTFAT The Large Time-Frequency Analysis Toolbox

-All your frame are belong to us -

Home | Documentation | Notes | Team | PHASERET



The Large Time/Frequency Analysis Toolbox (LTFAT) is a Matlab/Octave toolbox for working with time-frequency analysis and synthesis. It is intended

LTFAT structure

LTFAT Modules

- fourier
- gabor
- wavelets
- filterbank
- nonstatgab
- frames
- operators
- block processing
- quadratic
- sigproc
- auditory
- demos
- signals

Typical module structure:

- Transforms & core functionality
- Window/Filter construction
- Frame bounds
- Auxiliary functions
- Applications
- Plotting functions

Backend written in C

LTFAT Resources - Documentation

View the code

Go to function dgt

Categories
- base
- fourier
- gabor
- wavelets
- filterbank
- nonstatgab
- frames
- operators
- block processing
- quadratic
- sigproc
- auditory
- demos
- signals
- deprecated

See also:
- idgt
- gabwin
- dwilt
- gabdual
- phaselock

DGT - Discrete Gabor transform

Usage

```
c=dgt(f,g,a,M);
c=dgt(f,g,a,M,L);
c=dgt(f,g,a,M,'lt',lt);
[c,Ls]=dgt(...);
```

Input parameters

f	Input data.
g	Window function.
a	Length of time shift.
M	Number of channels.
L	Length of transform to do.
lt	Lattice type (for non-separable lattices).

Output parameters

c	\(M \times N\) array of coefficients.
Ls	Length of input signal.

Description

`dgt(f,g,a,M)` computes the Gabor coefficients (also known as a windowed Fourier transform) of the input signal *f* with respect to the window *g* and parameters *a* and *M*. The output is a vector/matrix in a rectangular layout.

The length of the transform will be the smallest multiple of *a* and *M* that is larger than the signal. *f* will be zero-extended to the length of the transform. If *f* is a matrix, the transformation is applied to each column. The length of the transform done can be obtained by `L=size(c,2)*a;`

The window *g* may be a vector of numerical values, a text string or a cell array. See the help of gabwin for more details.

View the help

Go to function dgt

Categories
- base
- fourier
- gabor
- wavelets
- filterbank
- nonstatgab
- frames
- operators
- block processing
- quadratic
- sigproc
- auditory
- demos
- signals
- deprecated

See also:
- idgt
- gabwin
- dwilt
- gabdual
- phaselock

DGT - Discrete Gabor transform

Program code:

```
function [c,Ls,g]=dgt(f,g,a,M,varargin)
%DGT  Discrete Gabor transform
%   Usage:  c=dgt(f,g,a,M);
%           c=dgt(f,g,a,M,L);
%           c=dgt(f,g,a,M,'lt',lt);
%           [c,Ls]=dgt(...);
%
%   Input parameters:
%         f     : Input data.
%         g     : Window function.
%         a     : Length of time shift.
%         M     : Number of channels.
%         L     : Length of transform to do.
%         lt    : Lattice type (for non-separable lattices).
%   Output parameters:
%         c     : M xN array of coefficients.
%         Ls    : Length of input signal.
%
%   DGT(f,g,a,M) computes the Gabor coefficients (also known as a windowed
%   Fourier transform) of the input signal f with respect to the window
%   g and parameters a and M. The output is a vector/matrix in a
%   rectangular layout.
%
%   The length of the transform will be the smallest multiple of a and M*
%   that is larger than the signal. f will be zero-extended to the length of
%   the transform. If f is a matrix, the transformation is applied to each
%   column. The length of the transform done can be obtained by
%   L=size(c,2)*a;
%
%   The window g may be a vector of numerical values, a text string or a
%   cell array. See the help of GABWIN for more details.
%
```

LTFAT Resources – Notes

057 Grid-Based Decimation for Wavelet Transforms With Stably Invertible Implementation — 2023 article
Nicki Holighaus, Günther Koliander, Clara Hollomey, Friedrich Pillichshammer
Webpage

056 Accelerating Matching Pursuit for Multiple Time-Frequency Dictionaries — 2020 conference
Zdeněk Průša, Nicki Holighaus, Peter Balazs
Cite this paper

055 Non-iterative phaseless reconstruction from wavelet transform magnitude — 2019 conference
Nicki Holighaus, Günther Koliander, Zdeněk Průša, Luis Daniel Abreu
Cite this paper
Webpage

053 Characterization of Analytic Wavelet Transforms and a New Phaseless Reconstruction Algorithm — 2018 article
Nicki Holighaus, Günther Koliander, Zdeněk Průša, Luis Daniel Abreu
Cite this paper
Webpage

052 Fast Matching Pursuit with Multi-Gabor Dictionaries — 2021 article
Zdeněk Průša,Nicki Holighaus, Peter Balazs
Cite this paper
Download poster, MATLAB/octave code

051 Non-iterative Filter Bank Phase (Re)Construction — 2017 conference
Zdeněk Průša, Nicki Holighaus
Cite this paper

Reproducible research:
The following archive ltfatnote057.zip contains scripts and data for reproducing figures and tables from the paper, as well as the reported experiments. The audio files for running Experiment 3 can be downloaded here, the audio files for running Experiment 2 can be downloaded here .

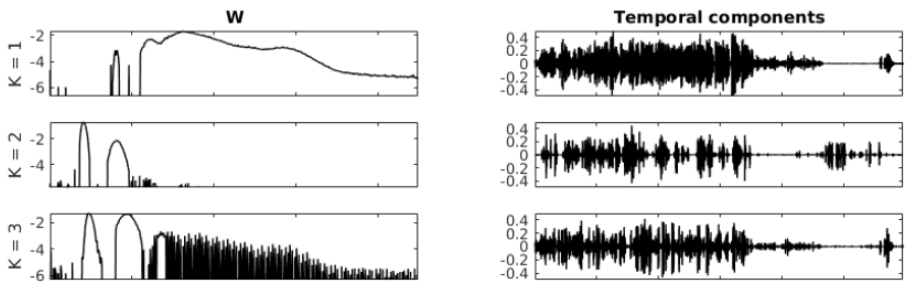
Please note that LTFAT toolbox (version>=2.5.0 or the current development version, available here) must be installed in order to run the scripts and reproduce the data.

Sound examples - Experiment 1 (Non-negative matrix factorization): The playback can be started by selecting one of the table cells (the cells turn yellow when the cursor hovers over them). Your browser must support HTML5 audio player. Alternativelly, the file path is shown below the player and it can be downloaded by Save Link As ...

0:00 / 0:00

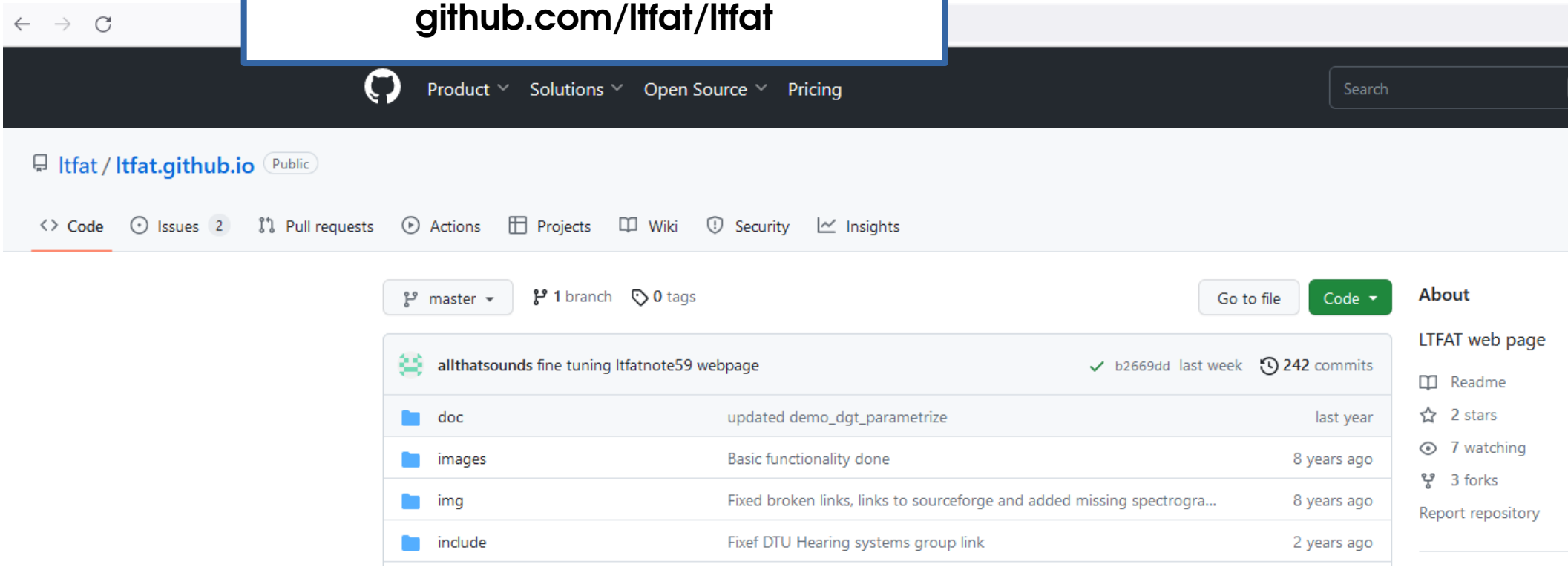
Loaded file: None

Non-negative matrix factorization signals:														
original	lead	accompaniment	denoised	upmix	comp 1	comp 2	comp 3	comp 4	comp 5	comp 6	comp 7	comp 8	comp 9	comp 10



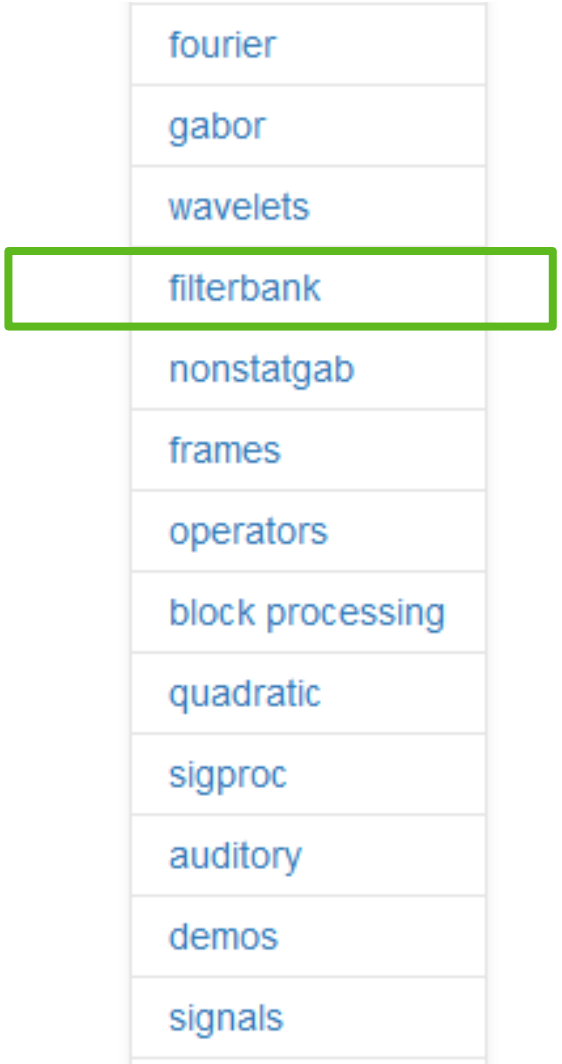
LTFAT Resources – The source code

github.com/ltfat/ltfat



How to install

- **`https://github.com/ltfat/ltfat/releases/tag/v2.6.0`**
- Download
  - MATLAB:
    - Windows: **`ltfat-2.6.0-win64.zip`**
    - MacOS & Linux: **`ltfat-2.6.0-src.tar.gz`**
  - Octave:
    - **`ltfat-2.6.0-of.tar.gz`**
- Extract the folder locally
- Open MATLAB/GNU Octave and navigate to the extracted folder
- Type **`ltfatstart`**
- Type **`ltfatmex`** (not on Windows, on MacOS: Xcode cmd line tools)

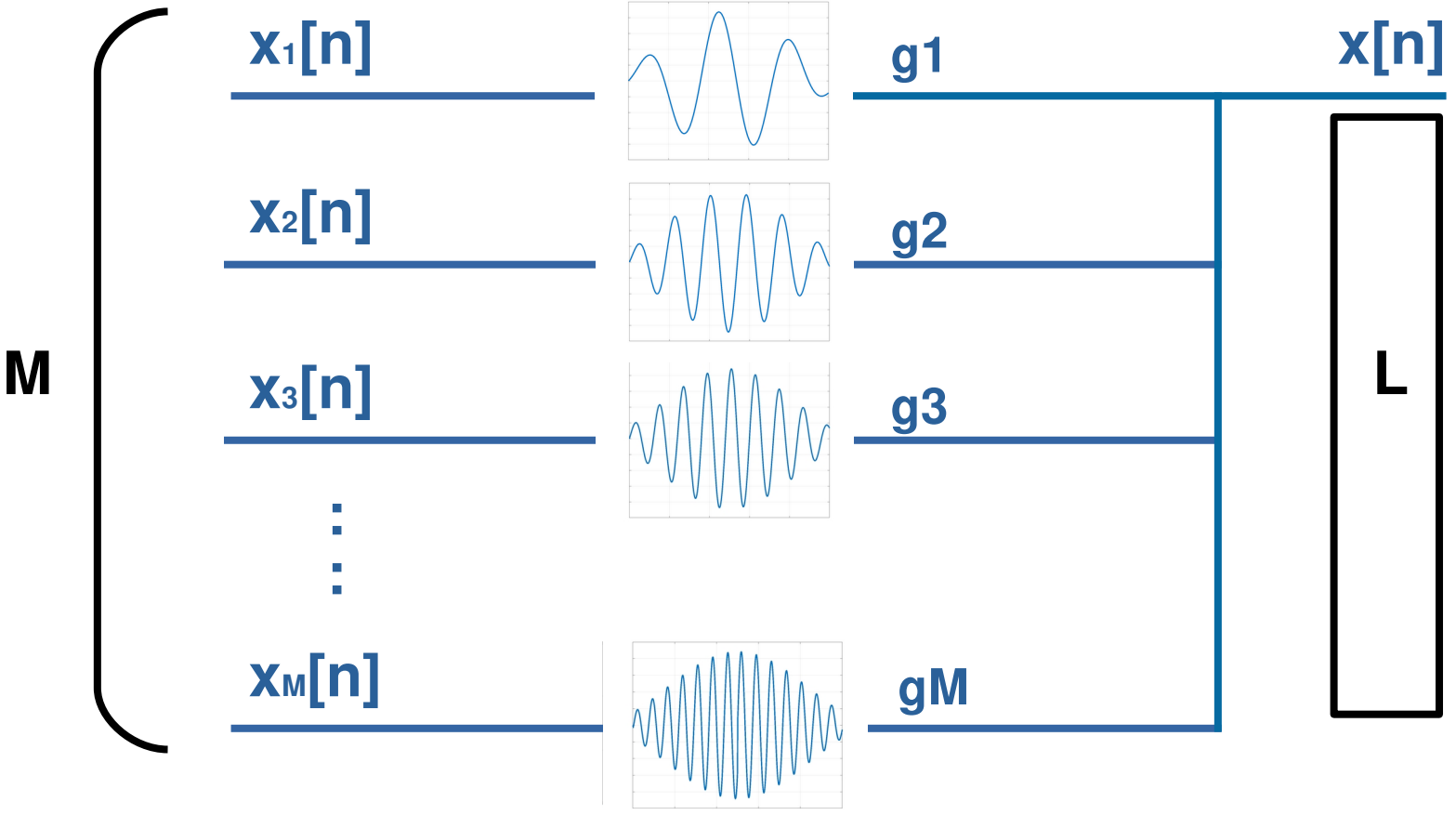


Filter banks

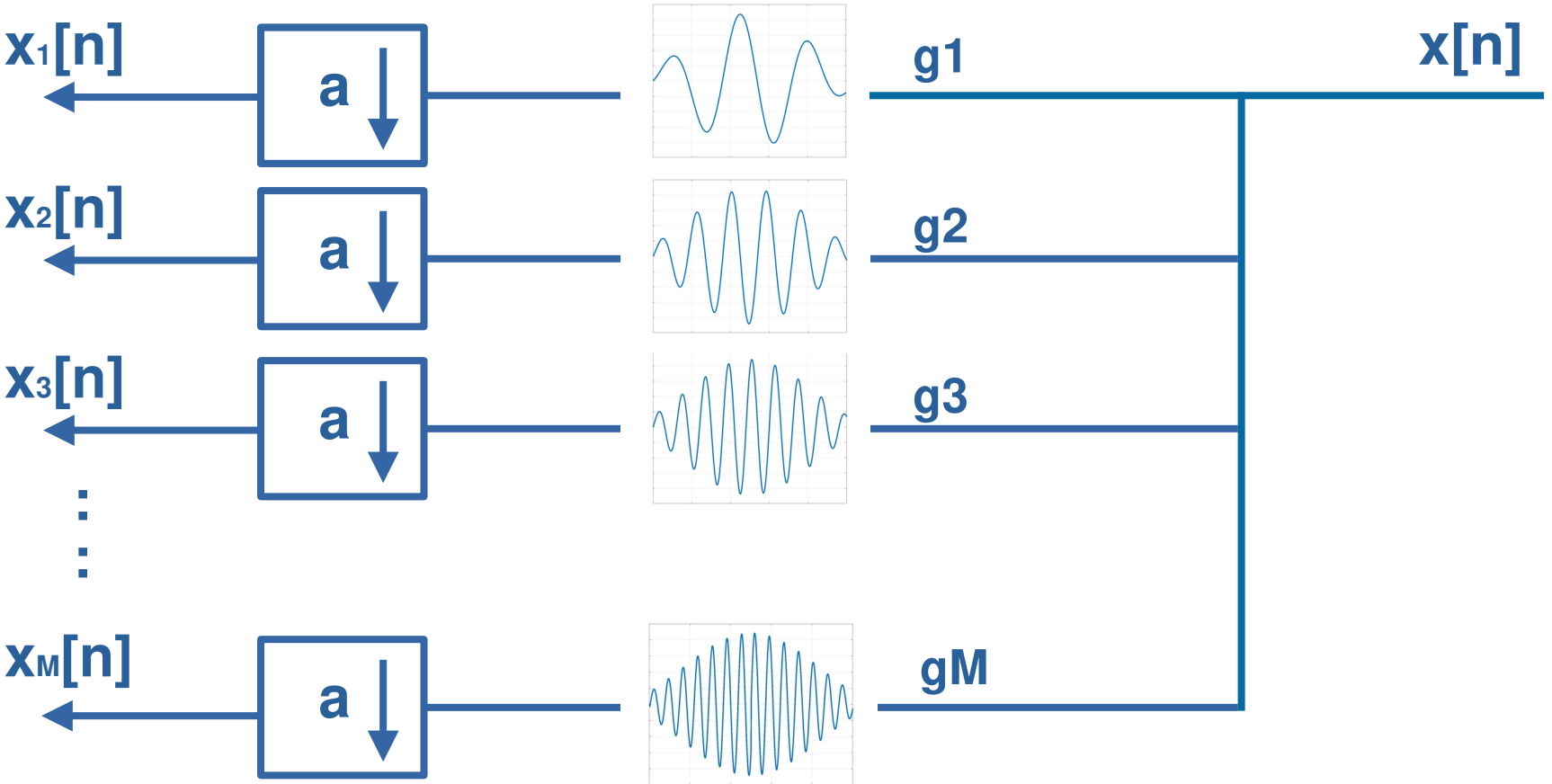
...an array of bandpass filters that separates the input signal into multiple components, each one carrying a single frequency sub-band of the original signal.

[https://en.wikipedia.org/wiki/Filter_bank]

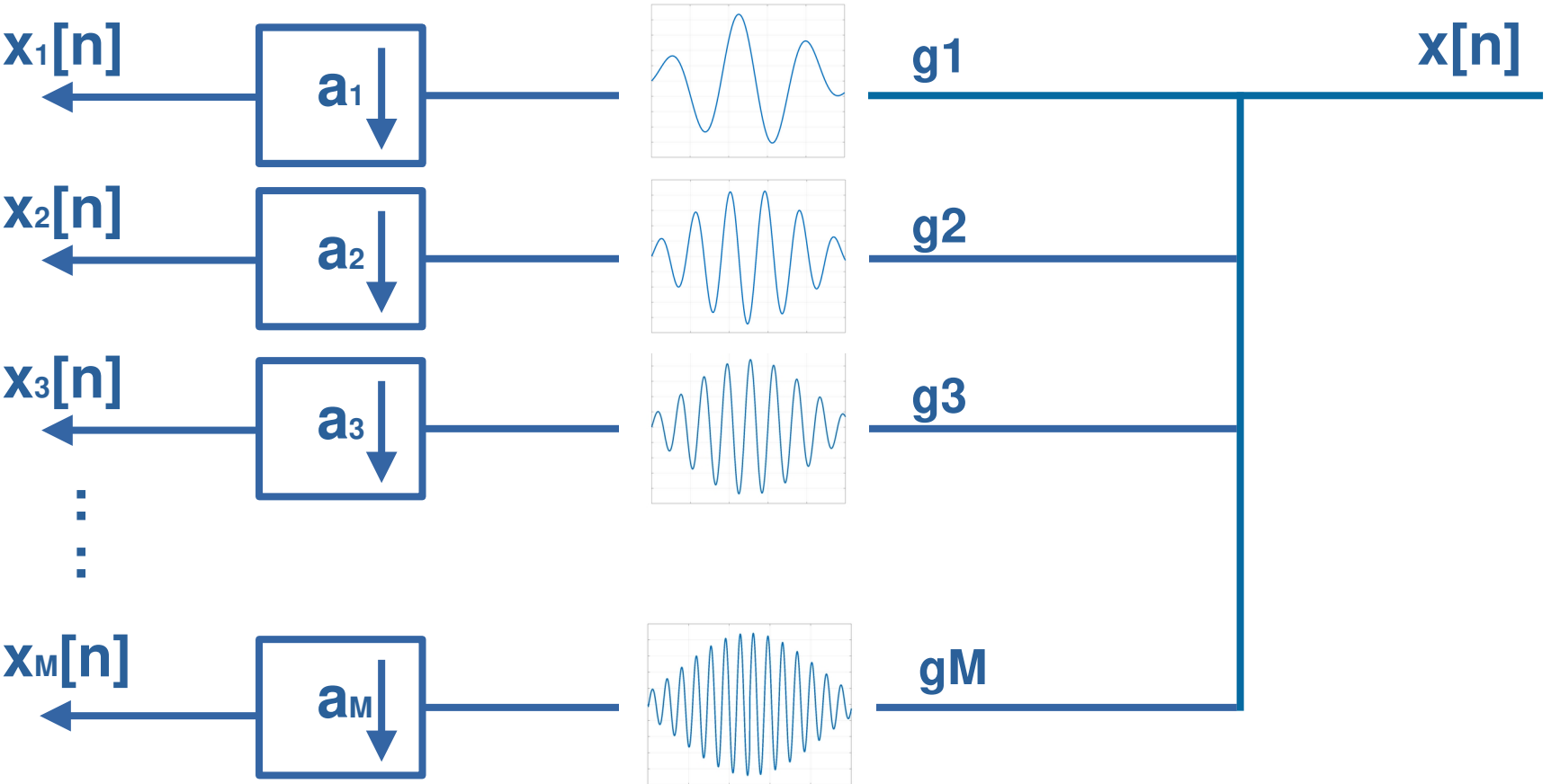
An array of band pass filters



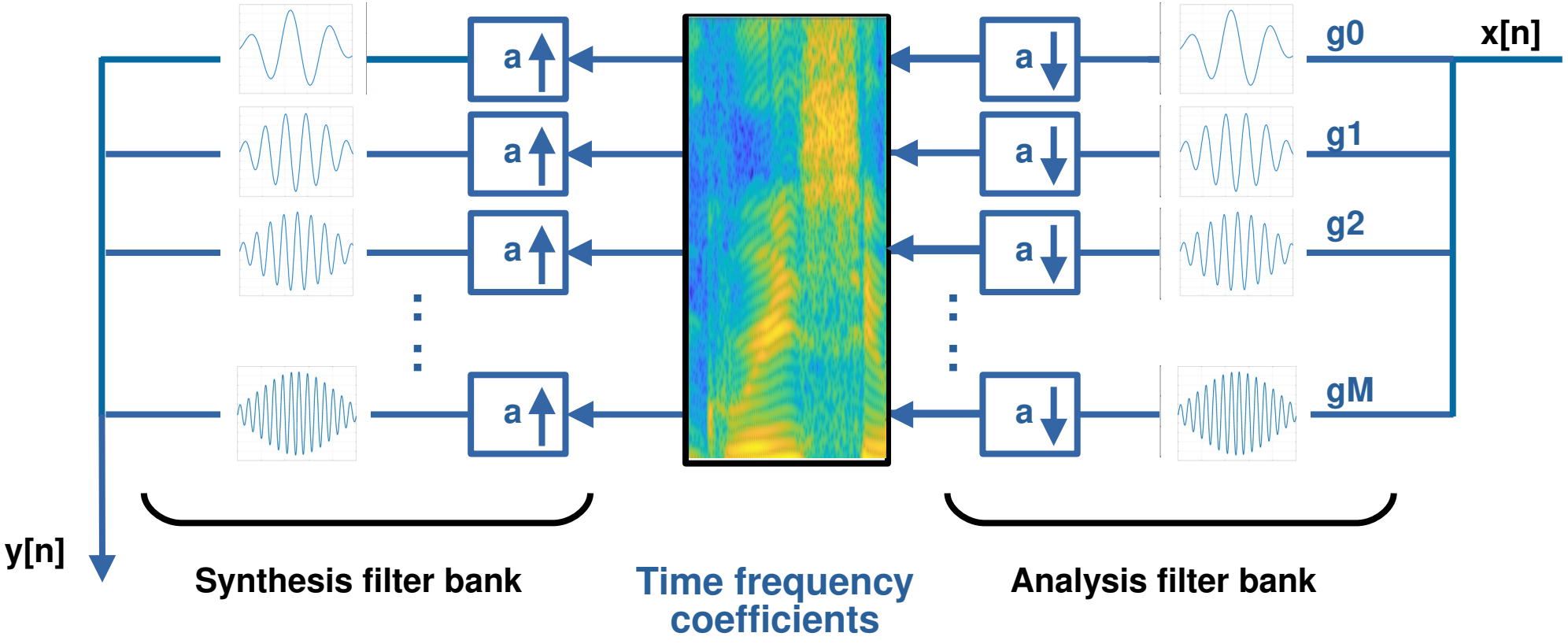
But this is not the whole truth...



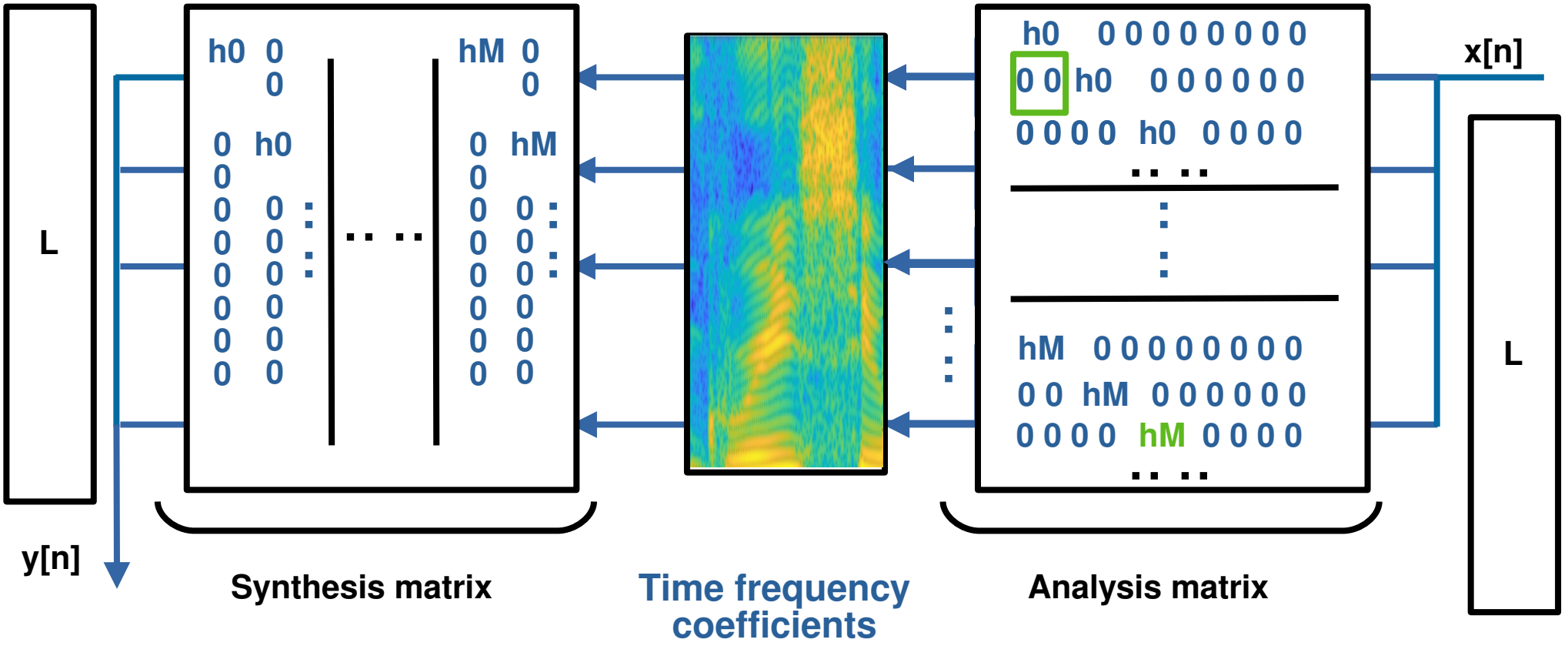
...and this is still not the whole truth.



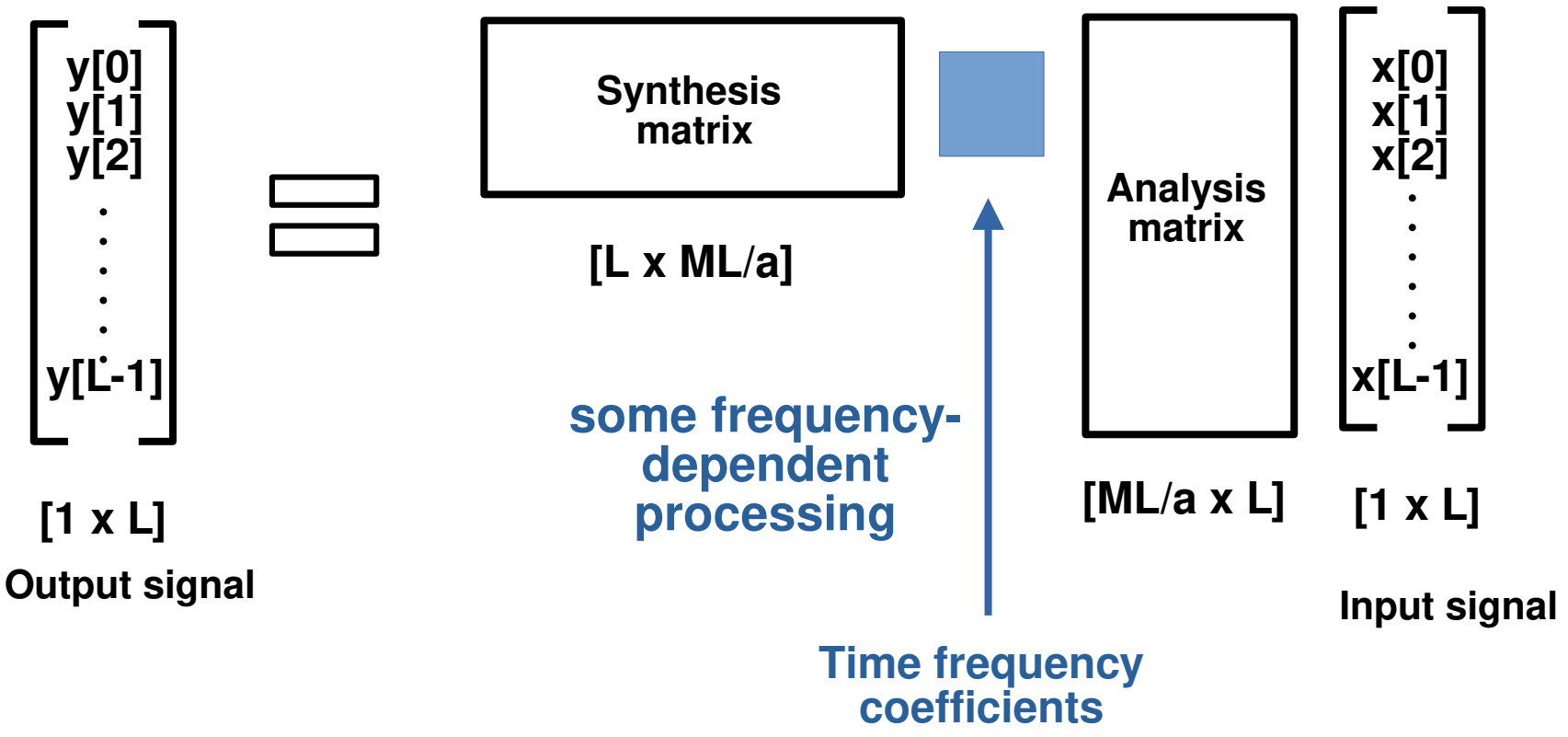
The full filter bank system



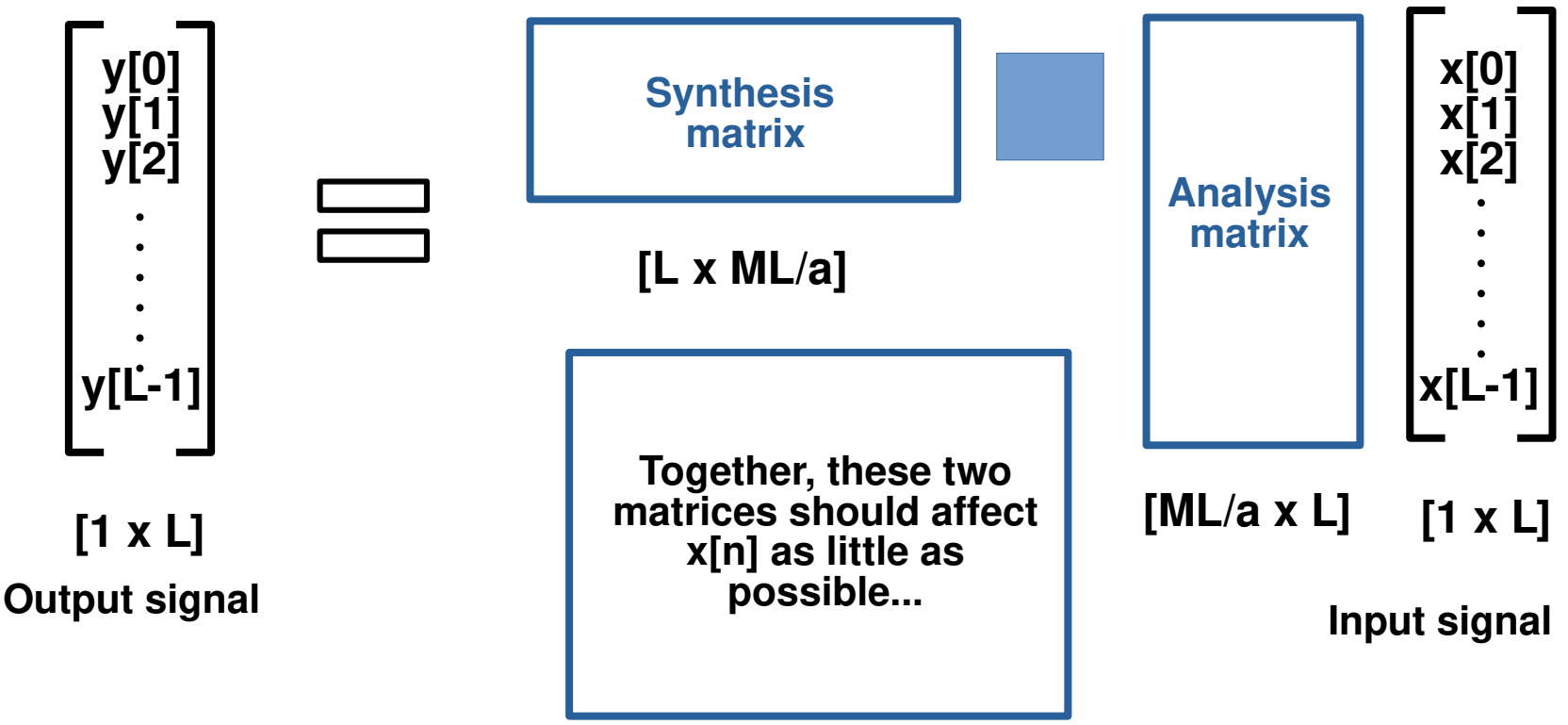
The full filter bank system



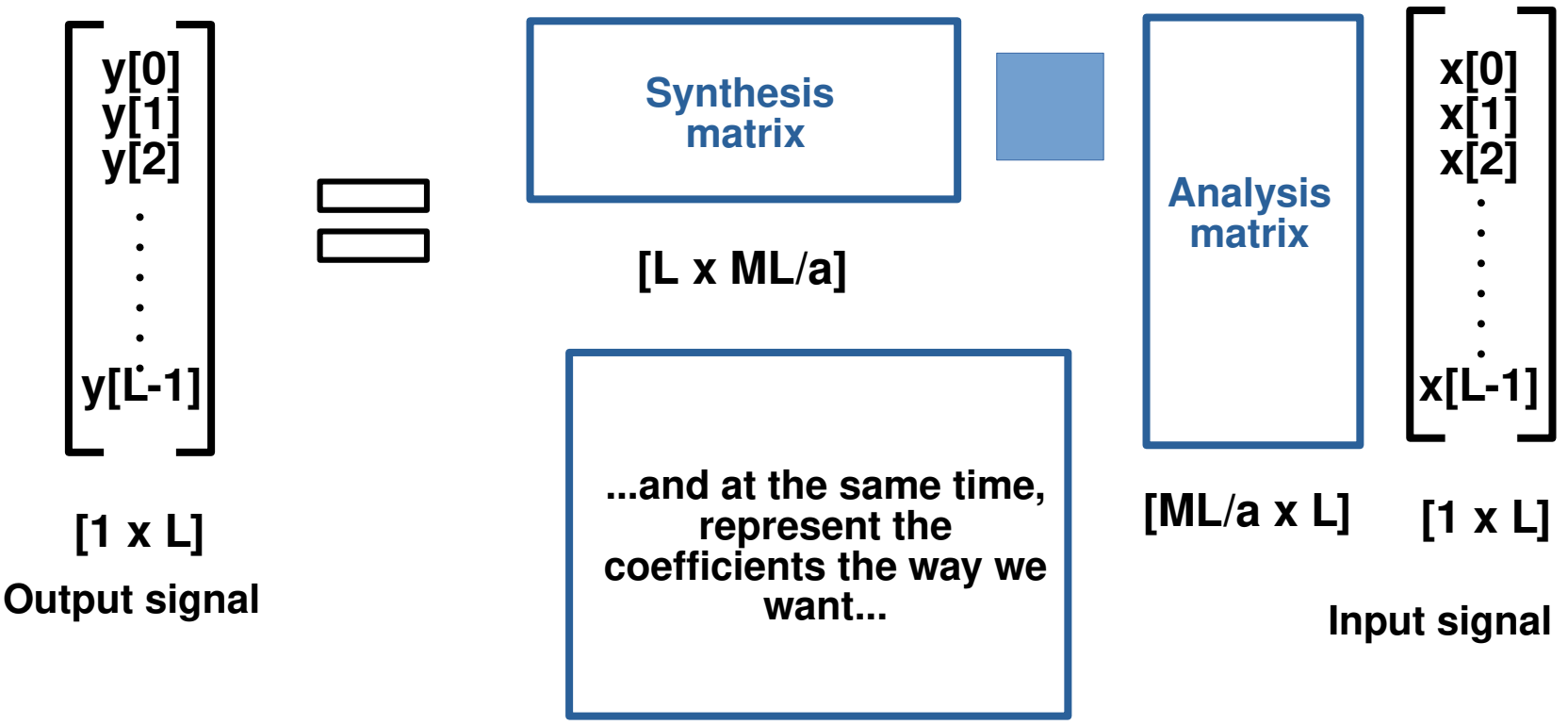
The full filter bank system of equations



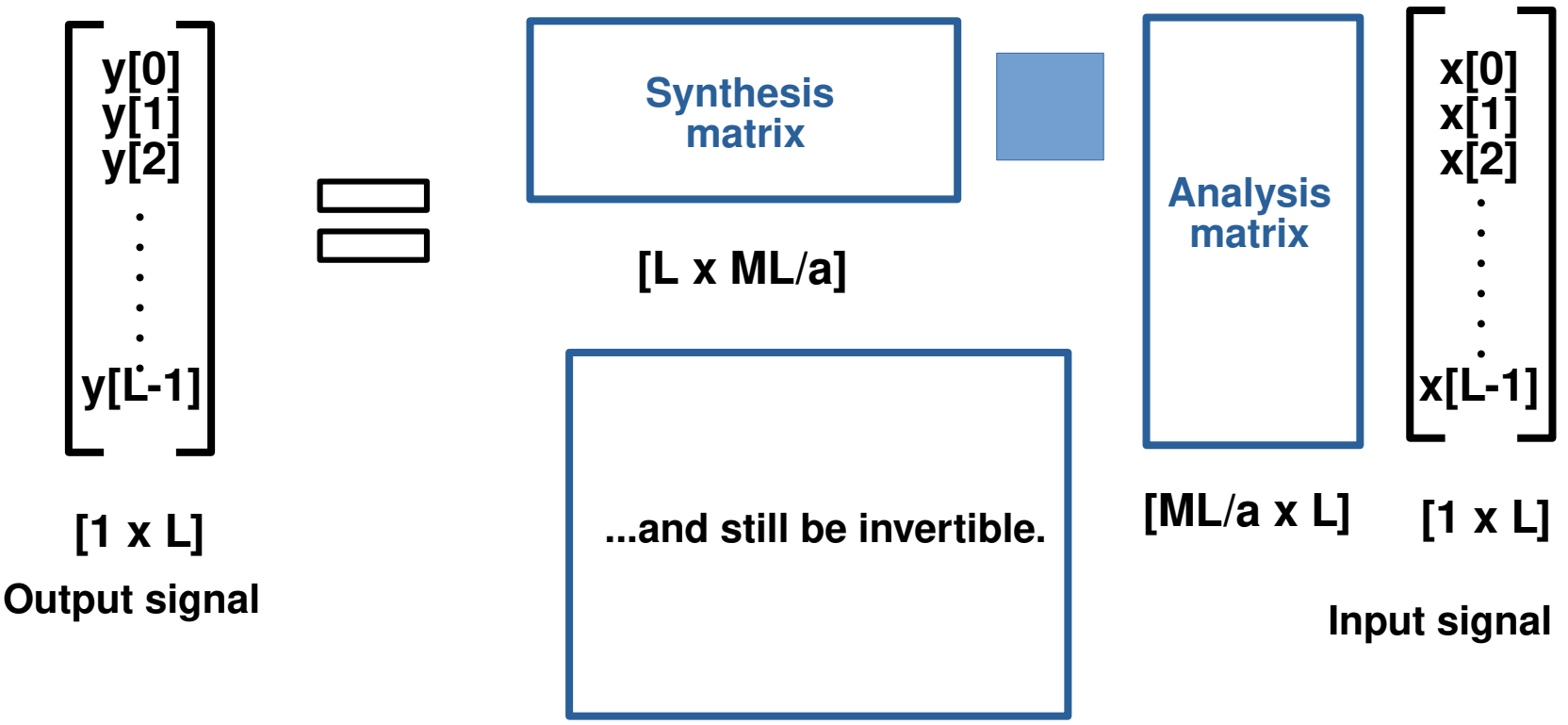
The full filter bank system of equations



The full filter bank system of equations



The full filter bank system of equations



Construct an invertible filter bank

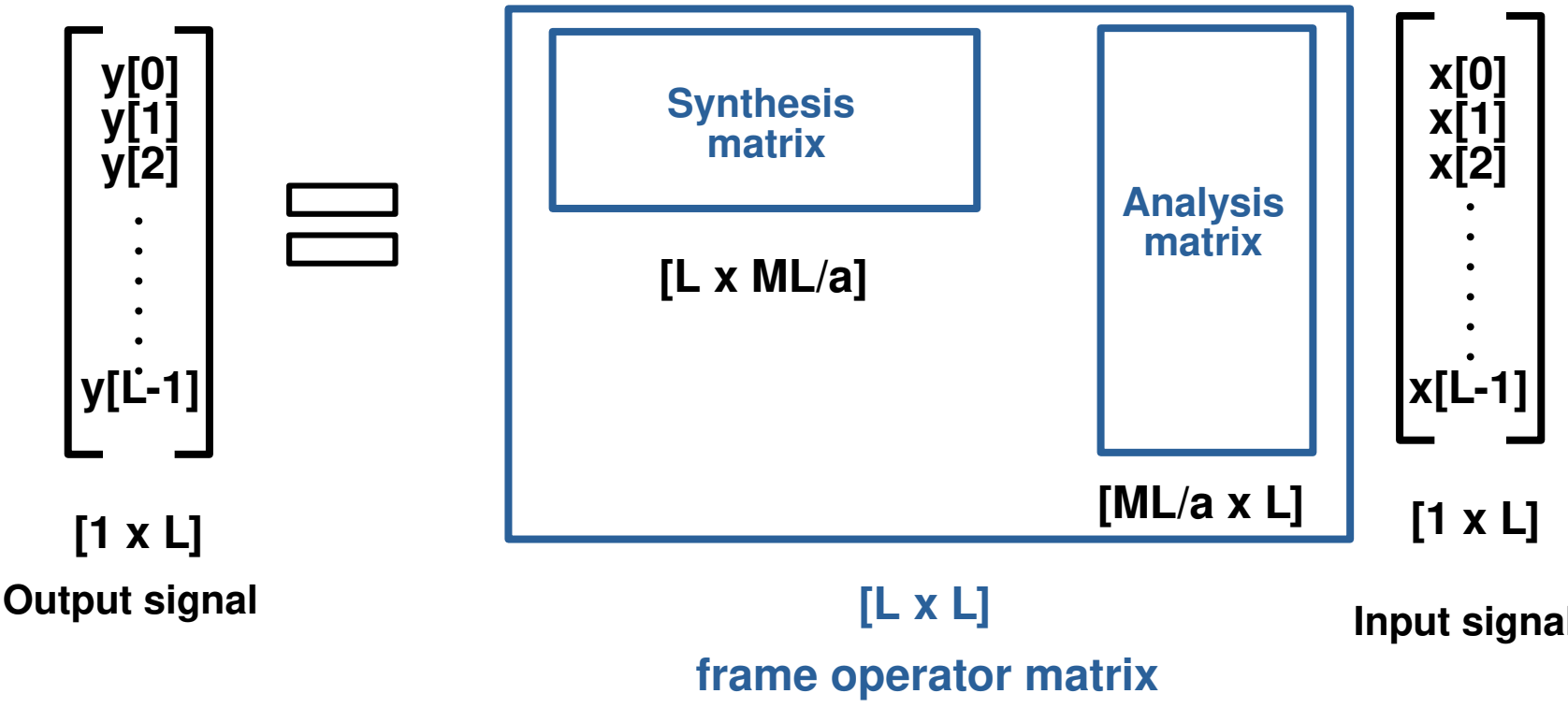
Run `ltfatworkshop_script1.m`

- Initialize the filters `g`, the downsampling factor `a`, the channel number `M`
- Match `L`, `g`, `a` and `M` such that they form an invertible filterbank: `gabfilters()`
- Construct the filter bank: `ufilterbank()`
- Invert it: `ifilterbank()`

`ltfatworkshop_script1.m` : output

- `Figure 1:` filter bank coefficients (complex)
- `Figure 2:` filter center frequencies (should be linearly spaced)
- `Figure 3:` frequency response single filters
- `Figure 4:` frequency response (should be constant)
- `Figure 5:` input and output signal
- `Figure 6:` the difference between input and output signal

(Discrete) frame theory 1/4



(Discrete) frame theory 2/4

LTFAT functions:

- `filterbankbounds`
- `filterbankrealbounds`
- `filterbankdual`
- `filterbankrealdual`
- `filterbanktight`
- `filterbankrealtight`

http://ltfat.org/doc/filterbank/

Mathematically speaking, a collection of vectors $\{g_k\}_{k\in\mathfrak{I}} \subseteq H$ is a frame for a Hilbert space H if there exist two constants $0 < A \leqslant B < \infty$, for which all $x \in H$

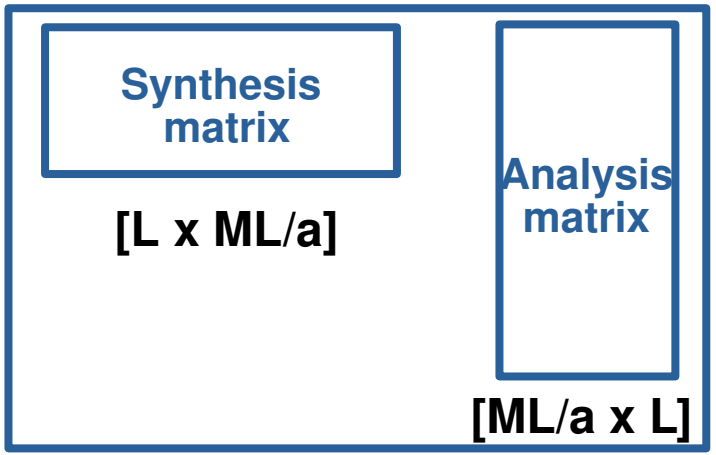
$$A\|x\|^2 \leq \sum_{k\in\mathfrak{I}} |\langle x, g_k\rangle|^2 \leq B\|x\|^2$$

A, B ...the frame bounds, the smallest and largest eigenvalues of the frame operator matrix

(Discrete) frame theory 3/4

Practically speaking, a matrix needs to

- (have full rank), and
- its smallest eigenvalue needs to be larger than zero, and
- its largest eigenvalue needs to be smaller than infinity

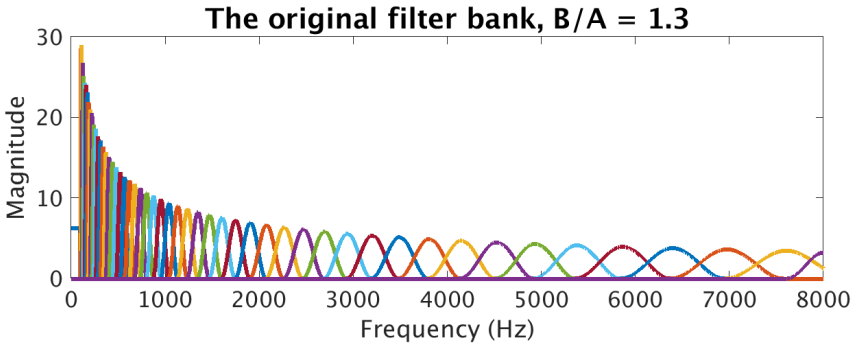


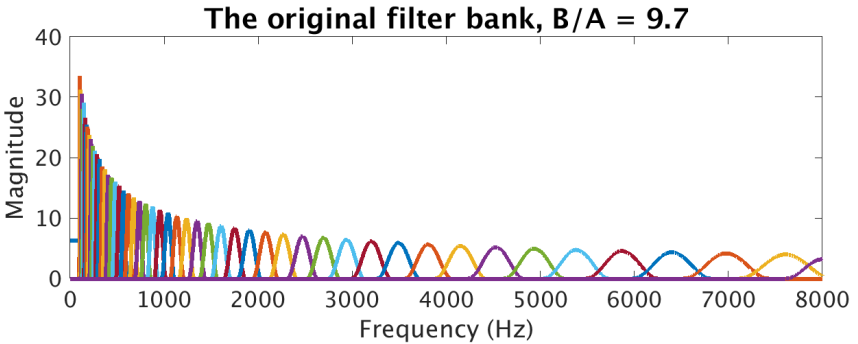
Condition number: $\kappa(S) = \frac{B}{A}$

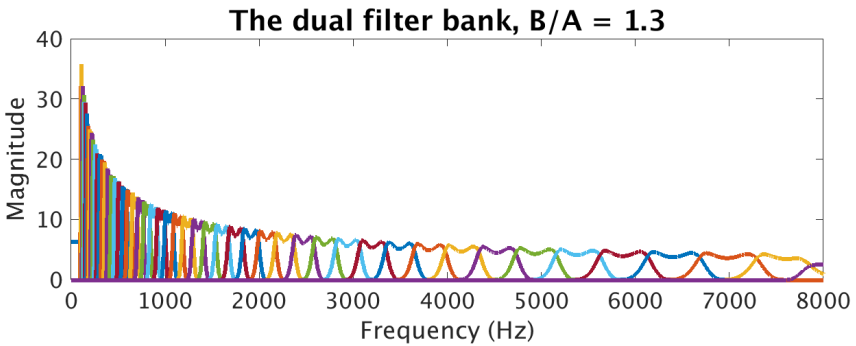
Advantages:

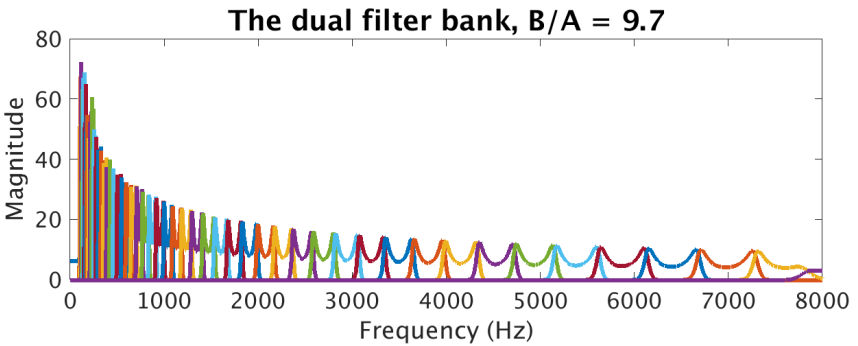
- robustness against noise and artefacts
- freedom in filter bank design

(Discrete) frame theory 4/4





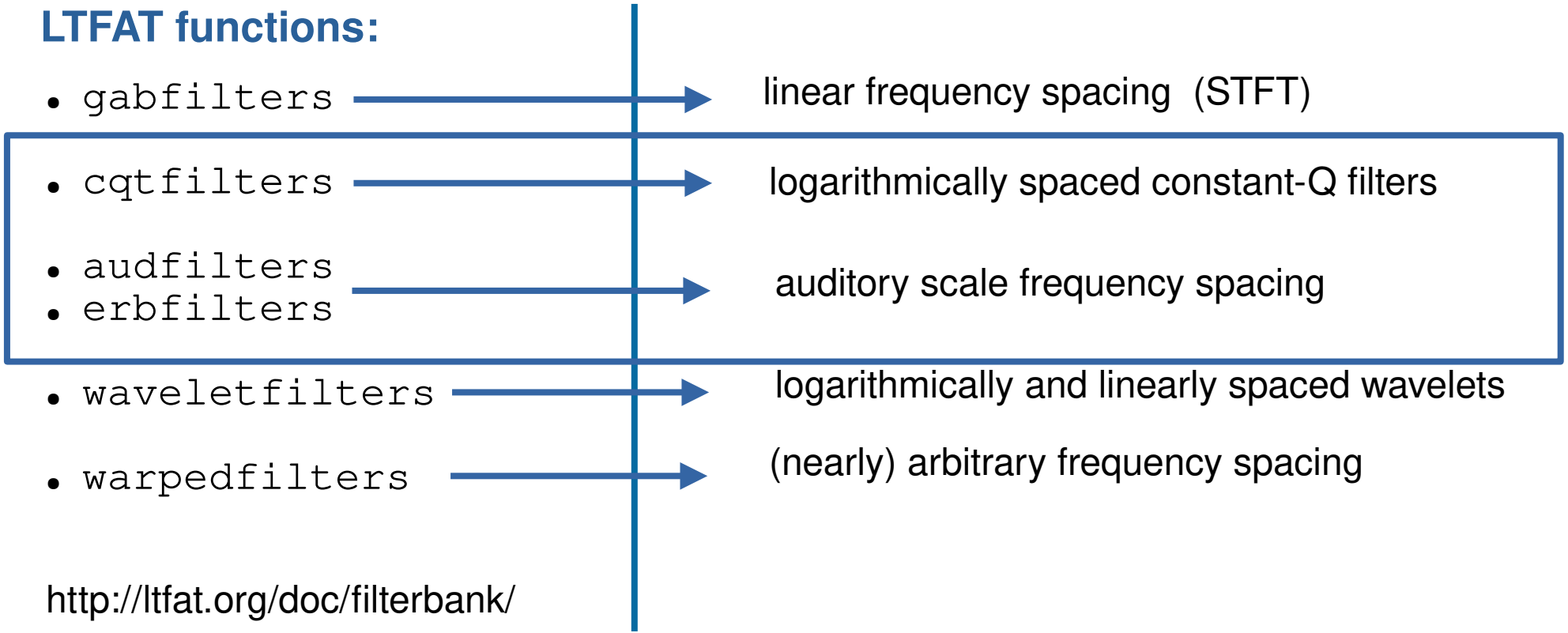




What you should remember

- Filter banks comprise an **analysis** and a **synthesis** part, each with
  - An array of band pass filters
  - Down/Upsampling units
- They can be written as the multiplication of two matrices.
- Together, hey should
  - Affect the processing chain as little as possible, yet
  - Yield the desired **time-frequency representation**, and
  - Be **invertible** (each).
- The **frame bounds** (smallest/largest eigenvalue) indicate whether the filter bank system fulfills these requirements.

How to build a filter bank for audio?



Filter banks for audio: requirements

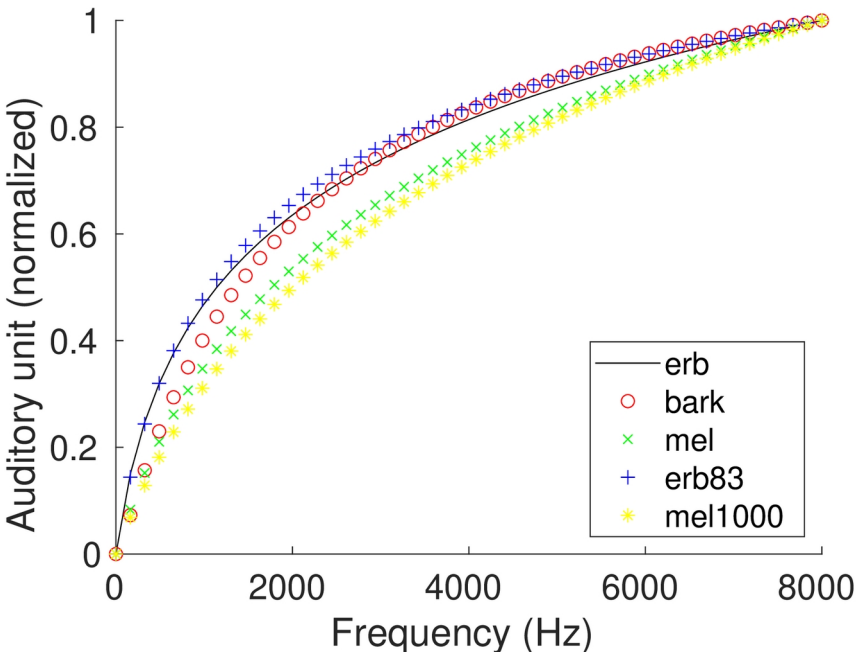
LTFAT functions:

Run

`ltfatworkshop_script2.m`

http://ltfat.org/doc/signals/

http://ltfat.org/doc/filterbank/



Filter banks for audio: requirements

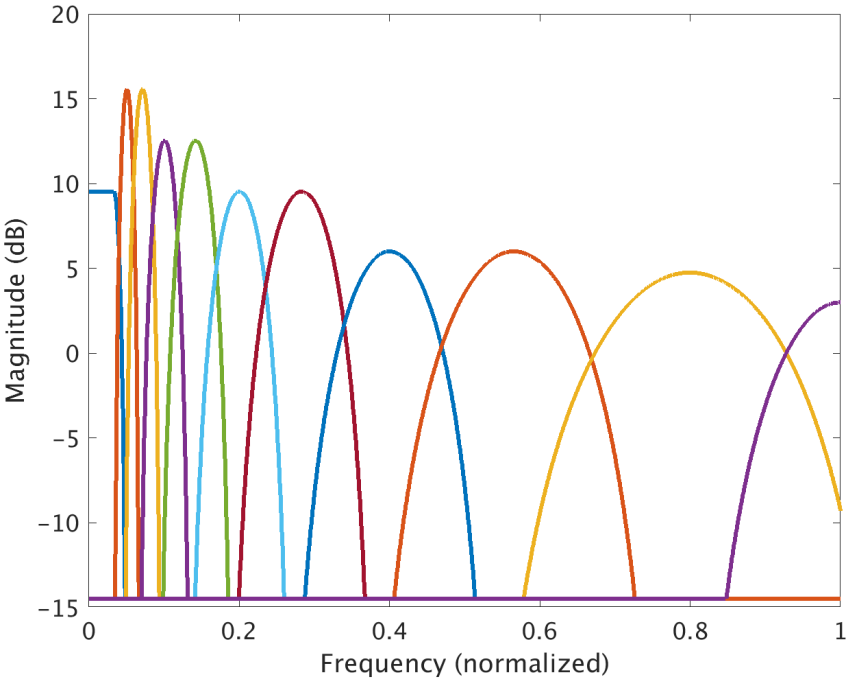
LTFAT functions:

Run

`ltfatworkshop_script2.m`

http://ltfat.org/doc/signals/

http://ltfat.org/doc/filterbank/



Designing a constant-Q filter bank

LTFAT functions:

Run

`ltfatworkshop_script3.m`

http://ltfat.org/doc/signals/

http://ltfat.org/doc/filterbank/

```
[g,a,fc L]=cqtfilters(fs,fmin,fmax,…
          bins,Ls, varargin);
```

- fs...sampling frequency of audio
- fmin...minimum frequency
- fmax...maximum frequency
- bins...number of frequency bins (per channel)
- Ls...length of audio
- varargin...next slide

- g...filters
- a...downsampling factors
- fc...filter center frequencies
- L...filter bank length

Position-dependent arguments

Run

`ltfatworkshop_script4.m`

```
[g,a,fc]=cqtfilters(fs,fmin,fmax,…
          bins,Ls, varargin );
```

- Key-value pairs:
  - `'Qvar',Qvar`
  - `'min_win',min_win`
  - `'redmul',redmul`
- Flags:
  - `'complex'`
  - `'nosubprec'`
  - `'regsampling'`
  - `'uniform'`
  - `'fractional'`
  - `'fractionaluniform'`

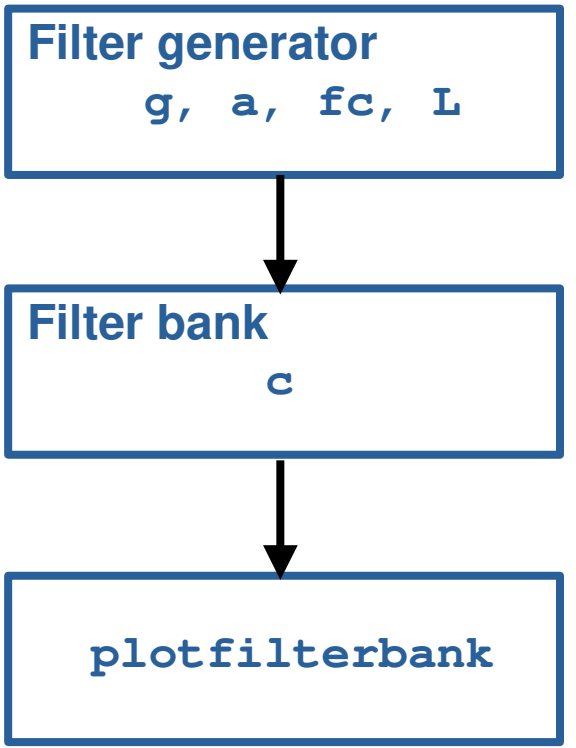
downsampling methods

Output parameters

LTFAT functions:

- `filterbank`
- `ufilterbank`

http://ltfat.org/doc/filterbank/



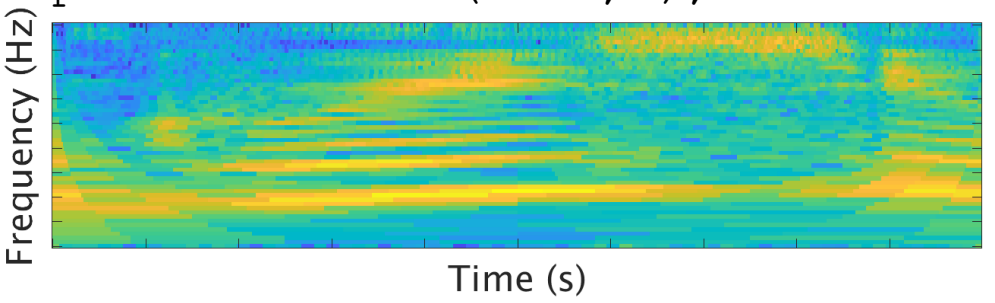
Graphic display

LTFAT functions:

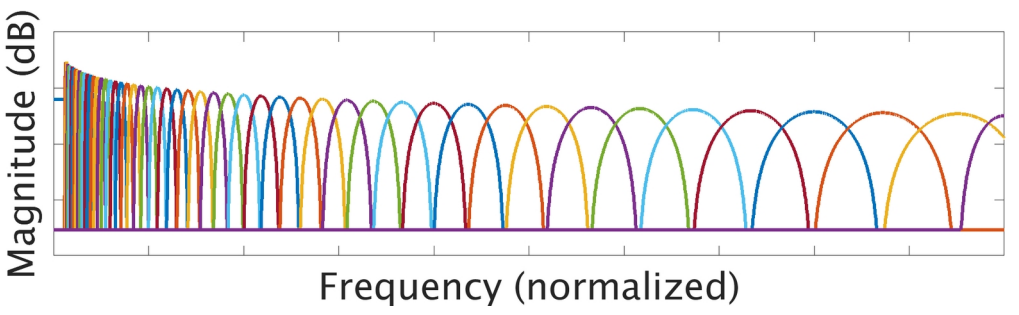
- `plotfilterbank`
- `filterbankfreqz`
- `filterbankresponse`

http://ltfat.org/doc/filterbank/

- `plotfilterbank(coef,a);`



- `filterbankfreqz(g,a,L);`



Filter bank inversion and reconstruction

Run
`ltfatworkshop_script5.m`

http://ltfat.org/doc/filterbank/

- `[A,B]=filterbankrealbounds(g,a,L);`
- `gd=filterbankrealdual(g,a,L);`
- `fout=ifilterbank(c, gd, a, 'real');`
- `rerr = norm(f-fout);`

Design an invertible gammatone filter bank

LTFAT functions:

- `audfilters`
- `filterbankrealbounds`
- `filterbank`
- `ifilterbank`
- `plotfilterbank`
- `filterbankfreqz`
- `filterbankresponse`
- `freqwin`

http://ltfat.org/doc/filterbank/audfilters.html

- Design a „nice looking“ (very redundant) auditory filter bank.
- Design the least redundant filter bank that is still invertible.
- Calculate the redundancies, reconstruction error, and framebounds. Plot their frequency responses.

Filter bank processing - reassignment

LTFAT functions:

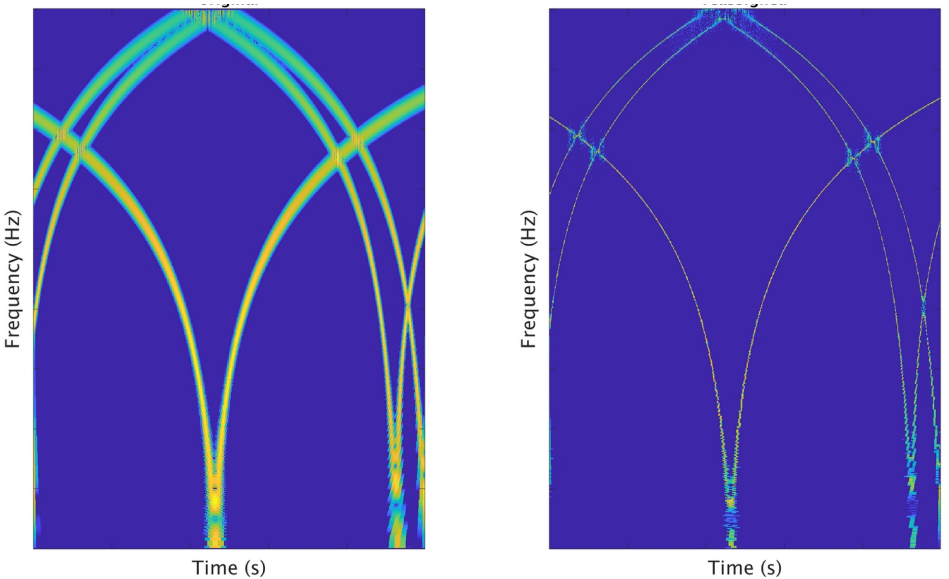
- `filterbankreassign`
- `demo_filterbank...`
  `synchrosqueeze`
- `filterbankphasegrad`

Reassign your filter bank

http://ltfat.org/notes/ltfatnote041.pdf

http://ltfat.org/notes/ltfatnote044.pdf

Goal: obtain a sharper time-frequency representation



Filter bank processing – phase retrieval

LTFAT functions:

- `im2double (Octave function)`
- `plotfilterbank`
- `filterbankconstphase`
- `filterbankrealbounds`
- `filterbankdual`
- `ifilterbank`

Repo: ltfat/signals/ltfattext.png

http://ltfat.org/doc/signals/

Download the audio files.
run
`ltfatworkshop_script7`

Filter bank processing – phase retrieval

LTFAT functions:

- `im2double (Octave function)`
- `plotfilterbank`
- `filterbankconstphase`
- `filterbankrealbounds`
- `filterbankdual`
- `ifilterbank`

Repo: ltfat/signals/ltfattext.png

http://ltfat.org/doc/signals/

`filterbankconstphase` parameters for image:

- `s...converted image`
- `a...ones(size (ltfattext,2),1)`
- `fc...[1:size (ltfattext,2)]`
- `tfr...1`

- fourier
- gabor
- wavelets
- filterbank
- nonstatgab
- frames
- operators
- block processing
- quadratic
- sigproc
- auditory
- demos
- signals

What else can you do with LTFAT?

PHASERET

`http://ltfat.org/phaseret`

A collection of phase retrieval algorithms
for STFT-like time frequency representations

```
mulaclab('insig.wav')
```

Time-variant filters

Summary

- **Homepage:**
  ltfat.org
- **Code & Discussions:**
  github/ltfat/ltfat
- **Octave:**
  https://gnu-octave.github.io/packages/ltfat/

Dependencies:

- fftw
- playrec
- portaudio
- GNU compiler chain
- polyboolclipper
- JAVA

- **Beyond filter banks:**
  - DGT/STFT, with varying windows and temporal sampling
  - wavelet transforms
  - phase retrieval and more...
- **Detailed help:**
  ltfat.org/doc
- **Citable:**
  ltfat.org/notes
- **Like & subscribe:**
  github/ltfat/ltfat

LTFAT