The LTFAT exercises

Peter L. Søndergaard, Jordy van Velthoven

This document contains several exercises to become acquainted with some of the functions and their computation in the Linear Time-frequency Analysis Toolbox (LTFAT). Each section starts with some theory on the functions used in the exercises of that section. For an introduction to the functions in the LTFAT the LTFAT tutorial can be consulted. The LTFAT tutorial can be downloaded from: `http://ltfat.sourceforge.net/notes/ltfatnote004.pdf` For a complete overview of the toolbox the on-line documentation or the LTFAT reference manual can be consulted. The on-line documentation of the toolbox is available from: `http://ltfat.sourceforge.net/doc/start` and the reference manual can be downloaded from: `http://ltfat.sourceforge.net/doc/ltfat.pdf`.

Contents

1 The Discrete Fourier Transform

The normalized discrete Fourier transform (DFT) is computed by

$$c\,(k+1) = \frac{1}{\sqrt{L}} \sum_{l=0}^{L-1} f\,(l+1)\, e^{-2\pi i k l/L}.$$

A discrete Fourier transform can be computed by the command `c=fft(f)`. However, this will compute an un-normalized DFT. The toolbox includes the normalized DFT (`dft`).

The inverse normalized discrete Fourier transform (IDFT) is computed by

$$f(l+1) = \frac{1}{\sqrt{L}} \sum_{k=0}^{L-1} c\,(k+1)\, e^{2\pi i k l/L}$$

The inverse discrete Fourier transforms are computed by `ifft` and `idft`.

A finite, discrete Gaussian is computed by

$$\varphi_w(l+1) = \left(\frac{wL}{2}\right)^{-1/4} \sum_{k\in\mathbb{Z}} e^{-\pi\left((l+c_t)/\sqrt{L} - k\sqrt{L}\right)^2/w}.$$

This is a sampling, periodization and dilation of the continuous Gaussian

$$\varphi(x) = e^{-\pi x^2}.$$

The finite, discrete Gaussian is computed by `pgauss(L,w);`, where L the the length of the Gaussian. If w is omitted, then a Gaussian is computed with $w = 1$.

The DFT of φ_w is $\varphi_{1/w}$.

Exercise 1

Compute the inverse discrete Fourier transform of a diagonal matrix:

```
f = ifft(eye(15));
```

Plot `f`:

```
plot(f);
```

What is the result? Try replacing 15 by odd and even numbers. When the number is even, what generates the straight, horizontal line in the middle?

Exercise 2

Plot a Gaussian by the following command:

```
plot(pgauss(40));
```

This plot shows how the Gaussian is centered, but it does not make a nice plot. The following is a better way to plot it:

```
plot(fftshift(pgauss(40)));
```

The fftshift command does nothing but move the beginning of the function to the middle of the plot. To see the faster exponential decay, use:

```
semilogy(fftshift(pgauss(40)));
```

Exercise 3

Compute three Gaussians by the commands:

```
g1=pgauss(40,.5);
g2=pgauss(40,1);
g3=pgauss(40,2);
```

What happens when w (the second argument to `pgauss`) gets larger?

Compute the DFTs of the three functions. What is the norm of the imaginary parts of the DFTs, e.g.

```
norm(imag(dft(g1)));
```

Check that the DFT of `g1` is `g3` and that `g2` is invariant.

2 Short-time Fourier transform

The Short time Fourier transform is computed by

$$c(m+1, n+1) = \sum f(k) e^{-2\pi i m k / L} \overline{g(k-n+1)}$$

A STFT of a signal f can be plotted by `sgram(f);` This will plot the spectrogram of f, this is simply the square of the absolute value of the coefficients c. The function `sgram` accepts some useful flags, including `'tc'` to move the beginning of the signal to the middle of the plot, `'nf'` to show negative frequencies even for real-valued signals and `'dynrange'` to limit the dynamic range.

Exercise 4

Compute the spectrogram of the bat signals through the command:

```
sgram(bat);
```

Compute the spectrograms of some Gaussians using additional parameters of `sgram`:

```
sgram(pgauss(100));
sgram(pgauss(100),'tc','nf');
sgram(pgauss(100,5),'tc','nf');
sgram(pgauss(100,1/5),'tc','nf');
```

Compute the spectrograms of some other functions:

```
sgram(pherm(100,8),'tc','nf');
sgram(pchirp(100,2),'nf');
sgram(shah(144,12),'nf');
```

Compute spectrograms of random values from the uniform and normal distributions:

```
sgram(rand(100,1),'nf');
sgram(randn(100,1),'nf');
```

Is there a visible difference between those spectrograms? Are the distributions stationary?

Exercise 5

Define the following short chirp:

```
f=zeros(80,1);
f(1:20)=exp(-pi*i*(2*((0:19)/20)).^2);
```

Compute the follow spectrograms:

```
sgram(f,'tc');
sgram(dft(f),'tc');
sgram(dft(dft(f)),'tc');
sgram(dft(dft(dft(f))),'tc');
sgram(dft(dft(dft(dft(f)))),'tc');
```

What happens? Try also using `idft` instead of `dft`.

3 Discrete Gabor Transform

A finite, discrete Gabor transform (DGT) is computed by

$$\mathbf{c}\,(m+1,n+1) = \sum_{l=0}^{L-1} f(l+1)e^{2\pi im/M}\overline{g\,(l-na+1)}, \tag{1}$$

for $m = 0,\ldots,M-1$ and $n = 0,\ldots,N-1$. To calculate the Gabor coefficients c of a signal f use `c=dgt(c,f,a,M);` The redundancy of the Gabor system is given by M/a.

Exercise 6

The time-frequency plane computed by (1) is highly redundant. For example, the two images generated by the code below contains the same information.

```
g=pgauss(400);
c1=dgt(bat,g,1,400);
c2=dgt(bat,g,20,25);
figure(1);
imagesc(abs(c1));
figure(2);
imagesc(abs(c2));
```

What is the size of `c1` and `c2`? How redundant are the two representations?

3.1 Inverse Discrete Gabor transform

The inverse discrete Gabor transform (IDGT) is computed by:

$$f(k+1) \quad = \quad \sum_{n=0}^{N-1}\sum_{m=0}^{M-1} cr(m+1,n+1)e^{2\pi imbk/L}g\,(k-an)$$

This is the inverse of the DGT, but only if the windows used are dual windows. Canonical dual and tight windows of a Gabor system with window g, time-shift a and M channels are computed by

```
gd=gabdual(g,a,M);
gd=gabtight(g,a,M);
```

Exercise 7

Compute the canonical dual `gd` and canonical tight `gt` window of a Gabor system with a Gaussian window `g` of length $L = 120$, time-shift $a = 10$ and $M = 12$ channels. Plot `g`, `gd` and `gt` using one of the following commands:

```
plot(fftshift(gd));
semilogy(fftshift(gd));
```

See what happens if you choose the initial window to wide:

```
g1=pgauss(120,2);
```

or too narrow

```
g2=pgauss(120,1/2);
```

Compare the Discrete Fourier Transform of the canonical dual window of `g1` with the canonical dual window of `g2`. Is there a resemblance? Why?

Exercise 8

`idgt` can be applied to a 3-dimensional array, which it will interpret as a collection of Gabor coefficients, so `c(:,:,1)` is the first set of coefficients, `c(:,:,2)` is the second set and so on. The output from such an `idgt` is a matrix, where the first column corresponds to `c(:,:,1)` the second to `c(:,:,2)` and so on.

Define:

```
c=reshape(eye(25),5,5,25);
```

What does `c(:,:,1)`, `c(:,:,2)`, etc. consist of?

Compute:

```
F=idgt(c,pgauss(15),3,5);
```

What does the colums of `F` consist of?

Plot `F`:

```
plot(F);
```

and compare the result with exercise 1.

4 Discrete wavelet transform

The J-level discrete wavelet transform of a function f with length L is given by

$$\alpha_j(n+1) = \langle f, g_j \rangle = \sum_{l=0}^{L-1} f(l+1)\bar{g_j}(l+1-2^j n+1),$$

$$\beta_j(n+1) = \langle f, h_j \rangle = \sum_{l=0}^{L-1} f(l+1)\bar{h_J}(l+1-2^J n+1)$$

here $\alpha_j(n)$ with $n \in \{0, ..., \frac{L}{2^j} - 1\}$ and $j \in \{1, ..., J\}$ are called the wavelet coefficients and $\beta_j(n)$ the scaling coefficients. The functions g_j and h_J are called the scaling sequence respectively the wavelet sequence and are recursively obtained from g_1 and h_1 as

$$g_{j+1}(l+1) = \sum_{k=0}^{L/2^j-1} g_1(k+1)h_j(l+1-2^j k+1)$$

$$h_{J+1}(l+1) = \sum_{k=0}^{L/2^j-1} h_1(k+1)h_j(l+1-2^j k+1)$$

where g_1 denotes a scaling function and h_1 a wavelet function.

The discrete wavelet transform is implemented in the LTFAT as the function `fwt`. The input of `fwt` is a function `f`, wavelet definition `w` and number of levels `J`. The output of `fwt` are the wavelet coefficients `c`.

Exercise 9

Compute the wavelet coefficients of the *greasy* signal obtained through a 7-level discrete wavelet transform with the Daubechies 4 (D4) wavelet function:

```
f = bat;
[c, info] = fwt(f, 'db4', 10);
```

Plot the wavelet coefficients through the `plotwavelets` function assuming a sampling rate of 44100 Hz from the `greasy` signal:

```
plotwavelets(c, info, 44100, 'dynrange', 80);
```

Plot now the Gabor coefficients of the `greasy` signal obtained through the Gabor transform:

```
cg = dgt(f, 'gauss', 60, 40, 'dynrange', 80);
```

What are the visible differences between the wavelet coefficients and the Gabor coefficients? Which properties does the wavelet transform have that the Gabor transform has not? In which cases could those properties be useful and in which not?