THE ANSI C
Time-Frequency Toolbox

Manuel Davy, Emmanuel Roy

November 15, 2000

Contents

1 Introduction

The *ANSI C Time-Frequency toolbox* is a package aimed at computing time-frequency representations. One the one hand, it is an extension of the *Time-Frequency toolbox for use with Matlab* `http://crttsn.univ-nantes.fr/ auger/tftb.html`, since it provides some accelerated functions, and new ones. On the other hand, this package provides C language sources that can be included in any program.

Most of the functions are similar to those programmed in the *Time-Frequency toolbox for use with Matlab*, and can be used in a similar way. As it is programmed in C language, its use with Matlab requires a ANSI C language compiler, compatible with your version of Matlab.

WARNING: THIS DISTRIBUTION IS A BETA VERSION, WHICH MAY HAVE BUGS! THE AUTHORS DECLINE RESPONSIBILITY FOR ANY PROBLEM CAUSED BY THIS PACKAGE

2 Installation

When you have uncompressed the file `Ctftb.zip` or `Ctftb.tar.gz`, the directory `Ctftbx` has been created. This directory contains this documentation file, the compilation script `tftbinst.m` and two directories containing the source `src` and help files `hlpfiles`.

2.1 Installation for use with Matlab

The installation for use with Matlab requires a ANSI C language compiler, compatible with Matlab. Before starting the installation of the *ANSI C Time-Frequency toolbox*, you need to know how the installation program will proceed.

1. The file `system.h` will be automatically generated in `src`. It will contain a string specifying which system you use, such as

   ```
   #define SYSTEME  UNIX
   ```

   if you use UNIX or LINUX. If you use Windows or MacOs, `system.h` will contain

   ```
   #define SYSTEME  WINDOWS
   ```

2. a directory, `Ctftb`, will be created in the current directory `Ctftbx`.

3. The C files will be compiled. This results in a set of files with a system-specific extension, such as `.dll` for Windows, *.mexlx* for Linux, ... (The default extension of mex files on your system, is displayed by the Matlab function `mexext.m`).

4. The generated mex files will be moved to the directory `Ctftb`. The help files (Files with the same name and the corresponding mex file — e.g. `filename.dll` — but with a `.m` extension — e.g. `filename.m`) will also be copied in `Ctftb`.

5. This ends the compilation step.

To proceed to the installation, do as follows:

1. Uncompress this package.

2. If you have never compiled C language programs under Matlab, then start Matlab and type `mex -setup`. If you have any problem at this step, please consult the Matlab documentation. It is recommended to try to compile an example program, such as `yprime.c` (typing mex yprime.c), provided in your Matlab distribution in `$MATLABROOT/extern/examples/mex/`, where `$MATLABROOT` is the directory in which Matlab is installed on your computer (type `matlabroot` under Matlab). Try to run the compiled file. If any problem occurs at this step, please consult your Matlab documentation.

3. Change directory to `Ctftbx`.

4. Type `tftbinst`. You will first have to indicate which system you use. The toolbox is compiled. The executable files and help files are moved to `Ctftb`.

5. Before the installation is completed, the files contained in the directory `Ctftb` must be moved to their final location. There are several possibilities:

   Local installation Move all the files in `Ctftb` in you Matlab preference directory (type `prefdir` under Matlab to know the name of this directory). The *C Time-Frequency toolbox* is now ready to use.

   Multi-User installation Move the directory `Ctftb` in `$MATLABROOT/toolbox/` (you must have write authorization for this step — e.g. on Unix systems, you must be root). Then, still with write authorization, start Matlab, and type

   ```
   addpath(fullfile(matlabroot,'toolbox','Ctftb'))
   path2rc
   ```

   This last command saves the new path, and should return 0 to indicate that the path has been saved.

6. The installation is completed.

7. Run `Ctftbdemo` for a demo.

2.2 Installation of the source files

The C language source files are in the directory `src`. You may copy it wherever you need. There is no other installation to do in the present version of the toolbox.

3 Functions description

This toolbox provides two types of functions:

- some functions, already existing in the *Time-Frequency toolbox for use with Matlab*. This functions have been re-programmed in C language for the sake of computational efficiency

- Some new functions

3.1 Translated functions

Most of the functions in this package are similar to their Matlab version (in the *Time-Frequency toolbox for use with Matlab*). However, there are two main differences:

- The `Trace` possibility is not implemented.
- The computation of cross-Time-frequency representations is not yet available.

Some other differences concern `Ctfrsp`, `Ctfrstft`, `Ctfrrsp`, ... Please read the corresponding help files. Table 1 displays the correspondence between Matlab and C Time-Frequency functions. Please report to *Time-Frequency toolbox for use with Matlab Reference guide* and to the corresponding help files.

Function name in this toolbox	**Equivalent function name in the Matlab toolbox**
`Cambifunb`	`ambifunb`
`Chtl`	`htl`
`Ctfrgrd`	`tfrgrd`
`Ctfrbj`	`tfrbj`
`Ctfrbud`	`tfrbud`
`Ctfrcw`	`tfrwv`
`Ctfrmh`	`tfrmh`
`Ctfrmhs`	`tfrmhs`
`Ctfrmmce`	`tfrmmce`
`Ctfrpage`	`tfrpage`
`Ctfrpmh`	`tfrpmh`
`Ctfrppage`	`tfrppage`
`Ctfrpwv`	`tfrpwv`
`Ctfrri`	`tfrri`
`Ctfrridb`	`tfrridb`
`Ctfrridbn`	`tfrridbn`
`Ctfrridh`	`tfrridh`
`Ctfrridt`	`tfrridt`
`Ctfrrsp`	`tfrrsp`
`Ctfrsp`	`tfrsp`
`Ctfrspwv`	`tfrspwv`
`Ctfrstft`	`tfrstft`
`Ctfrwv`	`tfrwv`
`Ctfrzam`	`tfrzam`
`Cwindow`	`window`

Table 1: Correspondence between the function programmed in this toolbox, and functions in the *Time-Frequency toolbox for use with Matlab*

3.2 New functions

Some new Functions have been created in this toolbox. Table 2 on the next page displays the name and a brief description of these time-frequency functions. These functions are described in Appendix A on the following page.

Name of the function	**Brief description**
`Caf2tfr`	Given the ambiguity function of a signal, and a TFR kernel (in the ambiguity plane), compute the corresponding TFR
`Ctfrdist`	Computes Time-Frequency distances between two TFRs
`Ctfrker`	Creates a TFR kernel, in the ambiguity plane
`Ctfrreas`	Reassigns a TFR, given a field reassignment vectors

Table 2: New time-frequency functions in this toolbox.

A Reference guide for the new functions

`Caf2tfr`

Purpose
Given a TFR kernel and the narrow-band ambiguity function of a signal, this function computes the corresponding TFR.

Synopsis
`TFR=Caf2tfr(AF,KERNEL);`

Description
`Caf2tfr` computes the double Fourier transform of the product ambiguity function × kernel:

$$\mathcal{C}_x^{\phi}(t,f) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} \mathcal{A}_x(\xi,\tau)\phi(\xi,\tau)e^{j2\pi(\tau f-\xi t)}\,d\xi d\tau$$

where $\mathcal{C}_x^{\phi}(t,f)$ is the TFR of the signal x, with kernel ϕ, $\mathcal{A}_x$ is the ambiguity function of x, ξ is the Doppler, τ is the delay (lag).

Name	Description	Default value
`AF`	Ambiguity function of the signal (size : `N` × `N`, where `N` is the number of points in the analyzed signal)	
`KERNEL`	matrix containing the TFR kernel (same size as the ambiguity function)	
`TFR`	Time-frequency representation, with the frequency bins stored in the rows and the time bins stored in the columns (same size as the ambiguity function and the kernel matrices)	

The ambiguity Matrix MUST be computed as shown in example, with the following lag vector and number of frequency bins: `AF=Cambifunb(x,-N/2+1:N/2,N)`. Other way of computing the TFR would produce an inaccurate result.

Example
Computation of a the TFR of a FM signal, with a Radially Gaussian kernel

```
t=1:128;
sig=exp(sqrt(-1)*2*pi*(0.3+0.2*t+0.001*t.^2))+randn(1,128);
af=Cambifunb(sig,-63:64,128);
kernel=Ctfrker(128,128,'rgk',[.2 1 1]);
TFR=Caf2tfr(af,kernel);
imagesc(1:128,linspace(0,0.5,128),TFR);
axis xy; xlabel('time'); ylabel('frequency')
```

`Ctfrdist`

Purpose

Computes time-frequency distance measures between TFRs.

Synopsis

```
dist=Ctfrdist(TFR1,TFR2,dist_name)
dist=Ctfrdist(TFR1,TFR2,dist_name,dist_coef)
```

Description

`Ctfrdist` computes several time-frequency distance measures between two TFRs

Name	Description	Default value
`TFR1`	First TFR	
`TFR2`	Second TFR, with the same number of columns and rows as `TFR1`	
`dist_name`	Name of the distance measure to use	
`distcoef`	distance parameters, required for some distance measures	
dist	Computed distance	

Possible distance names are: Where $\mathcal{C}_1(t,f)$ is the first TFR, $\mathcal{C}_2(t,f)$ is the second TFR,

Name	parameter	Expression
`'Lq'`	q	$\left[\int\int \lvert\mathcal{C}_1(t,f)-\mathcal{C}_2(t,f)\rvert^q \; dtdf\right]^{\frac{1}{q}}$
`'Quadratic'`	—	$\int\int \lvert\mathcal{C}_1(t,f)-\mathcal{C}_2(t,f)\rvert^2 \; dtdf$
`'Correlation'`	—	$1-\dfrac{2\iint \mathcal{C}_1(t,f)\,\mathcal{C}_2(t,f)\,dtdf}{\lVert\mathcal{C}_1\rVert^2+\lVert\mathcal{C}_2\rVert^2}$
`'Kolmogorov'`	—	$\int\int \lvert\mathbf{N}\mathcal{C}_1(t,f)-\mathbf{N}\mathcal{C}_2(t,f)\rvert \; dtdf$
`'Kullback'`	—	$\int\int \left(\mathbf{N}\mathcal{C}_1(t,f)-\mathbf{N}\mathcal{C}_2(t,f)\right)\log\frac{\mathbf{N}\mathcal{C}_1(t,f)}{\mathbf{N}\mathcal{C}_2(t,f)} \; dtdf$
`'Chernoff'`	m	$-\log\left[\int\int \mathbf{N}\mathcal{C}_1(t,f)^m\cdot\mathbf{N}\mathcal{C}_2(t,f)^{1-m} \; dtdf\right]$
`'Matusita'`	—	$\left[\int\int \left\lvert\mathbf{N}\mathcal{C}_1(t,f)^{1/m}-\mathbf{N}\mathcal{C}_2(t,f)^{1/m}\right\rvert^m \; dtdf\right]^{\frac{1}{m}}$
`'NLq'`	q	$\left[\int\int \lvert\mathbf{N}\mathcal{C}_1(t,f)-\mathbf{N}\mathcal{C}_2(t,f)\rvert^q \; dtdf\right]^{\frac{1}{q}}$
`'LSD'`	q	$\left[\int\int \lvert\log\mathbf{N}\mathcal{C}_1(t,f)-\log\mathbf{N}\mathcal{C}_2(t,f)\rvert^q \; dtdf\right]^{\frac{1}{q}}$
`'Jensen'`	m	see [1]

and $\mathbf{N}\mathcal{C}(t,f)$ denotes the normalization of the TFR as

$$\mathbf{N}\mathcal{C}(t,f) \;=\; \frac{\lvert\mathcal{C}(t,f)\rvert}{\displaystyle\int\int \lvert\mathcal{C}(s,\nu)\rvert \; dsd\nu}$$

Example

Distance between two Wigner-Ville distribution of the same FM signal, with different noise realizations

```
t=1:128;
sig=exp(sqrt(-1)*2*pi*(0.3+0.2*t+0.001*t.^2))+randn(1,128);
```

```
TFR1=Ctfrwv(sig);
sig=exp(sqrt(-1)*2*pi*(0.3+0.2*t+0.001*t.^2))+randn(1,128);
TFR2=Ctfrwv(sig);
d=Ctfrdist(TFR1,TFR2,'Kolmogorov')
d=Ctfrdist(TFR1,TFR2,'Jensen',3)
d=Ctfrdist(TFR1,TFR2,'Chernoff',0.5)
```

Reference

[**1**] O. Michel, R.G. Baraniuk and P. Flandrin, "Time-Frequency Based Distance and Divergence Measure," *IEEE Int. Symp. on TFTS*, 1994, pp. 64-67

[**2**] M. Basseville, "Distance measures for signal processing and pattern recognition," *Signal Processing*, No. 4, Vol. 18, December 1989, pp.349-369

[**3**] M. Davy, C. Doncarli and G.F. Boudreaux-Bartels: "Improved Optimization of Time-Frequency based Signal Classifiers", *IEEE Signal Processing Letters*, January 2001.

Ctfrker

Purpose

Generates a TFR kernel in the ambiguity plane, given a kernel shape.

Synopsis

`KERNEL=Ctfrker(NDOPPLER,NDELAY,KER_name); KERNEL=Ctfrker(NDOPPLER,NDELAY,KER_name,parameters);`

Description

`Ctfrker` generates a TFR kernel matrix in the ambiguity plane, given a kernel shape. This matrix can be used in e.g. `Caf2tfr`

Name	Description	Default value
`NDOPPLER`	Number of Doppler bins, i.e. number of rows in the output matrix	
`KERNEL`	Number of delay bins, i.e. number of columns in the output matrix	
`KER_name`	name of the kernel shape	
`parameters`	kernel parameters	
`KERNEL`	Output matrix, containing the kernel, with the Doppler bins stored in the rows and the delay bins stored in the columns.	

Possible kernel names are:

Name	parameters	Expression	Ref
`'WV'`	—	1	
`'spectro'`	window h	$[\mathcal{A}_h(\xi,\tau)]^\star$	
`'rgk'`	c $a_1,\ldots,a_n$ $b_1,\ldots,b_n$	$e^{-\frac{\rho^2}{2\sigma(\theta)^2}}$ where $\sigma(\theta)=c+\sum_{p=1}^{n}[a_p\,\cos(2p\theta)+b_p\,\sin(2p\theta)]$	[1]
`'gmcwk'`	$\sigma,\theta_1\ldots\theta_n$	$e^{-\frac{1}{\sigma}\prod_{k=1}^{n}(\xi\cos\theta_k+\tau\sin\theta_k)^2}$	[2]
`'mtek'`	α,β,γ r,τ_0,ν_0,λ	$e^{-\pi\left\{\left[\left(\frac{\tau}{\tau_0}\right)^2\left(\frac{\xi}{\xi_0}\right)^{2\alpha}+\left(\frac{\tau}{\tau_0}\right)^{2\alpha}\left(\frac{\xi}{\xi_0}\right)^2+2rA\right]^2\right\}^\lambda}$ where $A=\left([\tau\xi/\tau_0\xi_0]^\beta\right)^\gamma$	[3]

where θ and ρ are the polar coordinates in the ambiguity plane ($\rho^2=\xi^2+\tau^2, \tan(\theta)=\xi/\tau$), with τ the delay and ξ the Doppler.

Example

See the function `Caf2tfr`

Reference

[1] M. Davy, C. Doncarli and G.F. Boudreaux-Bartels: "Improved Optimization of Time-Frequency based Signal Classifiers", *IEEE Signal Processing Letters*, January 2001.

[2] X.-G. Xia. Y. Owechko, B. H. Soffer and R. M. Matic, "On Generalized-Marginals Time-Frequency Distributions," *IEEE Trans. on Signal Processing*, No. 11, Vol. 44, 1996, pp.2882-2886.

[**3**] H. Costa and G.F. Boudreaux-Bartels, "Design of Time-Frequency Representations Using a Multiform, Tiltable Exponential Kernel," *IEEE Trans. on Signal Processing*, No. 10, Vol. 43, 1995, pp.2283-2301.

Ctfrreas

Purpose
Reassigns the pixels of a TFR, given a field of reassignment vectors

Synopsis
`TFR_R=Ctfrreas(TFR,field_time,field_freq);`

Description
`Ctfrreas` reassigns the TFR pixels according to a field of reassignment vectors. The pixel that should be reassigned outside the TFR matrix are, in the time direction, left along the edges (time=0 or time = max_time); and in the frequency direction, a circular rotation is done.

Name	Description	Default value
`TFR`	TFR to be reassigned	
`field_time`	Time component of the field of reassignment vectors	
`field_freq`	Frequency component of the field of reassignment vectors	
`TFR_R`	Reassigned Time-frequency representation, with the frequency bins stored in the rows and the time bins stored in the columns	

Example
Alternative method for spectrogram reassignment

[1] E. Chassande-Mottin, F. Auger, P. Flandrin, "On the statistics of spectrogram reassignment vectors," Multidimensional Systems and Signal Processing, Vol. 9, No. 4, pp. 355-362, 1999

B Source files

Help files (.m files)

Caf2tfr.m	Ctfrdist.m	Ctfrgrd.m	Ctfrpmh.m	Ctfrridbn.m
Ctfrstft.m	Cambifunb.m	Ctfrker.m	Ctfrppage.m	Ctfrridh.m
Ctfrwv.m	Chtl.m	Ctfrmh.m	Ctfrpwv.m	Ctfrridt.m
Ctfrzam.m	Ctfrbj.m	Ctfrmhs.m	Ctfrreas.m	Ctfrrsp.m
Cwindow.m	Ctfrbud.m	Ctfrmmce.m	Ctfrri.m	Ctfrsp.m
Ctfrcw.m	Ctfrpage.m	Ctfrridb.m	Ctfrspwv.m	Contents.m

Installation program (.m file)

tftbinst.m

Include files (.h files)

tftb.h system.h

C/Matlab Interface programs (.c files)

Caf2tfr.c	Ctfrdist.c	Ctfrgrd.c	Ctfrpmh.c	Ctfrridbn.c
Ctfrstft.c	Cambifunb.c	Ctfrker.c	Ctfrppage.c	Ctfrridh.c
Ctfrwv.c	Chtl.c	Ctfrmh.c	Ctfrpwv.c	Ctfrridt.c
Ctfrzam.c	Ctfrbj.c	Ctfrmhs.c	Ctfrreas.c	Ctfrrsp.c
Cwindow.c	Ctfrbud.c	Ctfrmmce.c	Ctfrri.c	Ctfrsp.c
Ctfrcw.c	Ctfrpage.c	Ctfrridb.c	Ctfrspwv.c	

Computation programs (.c files)

File name	Matlab interface	Purpose
af.c	Cambifunb.c	Computes the ambiguity function
af2tfr.c	Caf2tfr.c	Given a kernel and an ambiguity function, computes the corresponding TFR
bj.c	Ctfrbj.c	Computes the Born-Jordan distribution
bud.c	Ctfrbud.c	Computes the Butterworth distribution
create_window.c	Cwindow.c	Creates a window of given shape
cw.c	Ctfrcw.c	Computes the Choi-Williams distribution
distance.c	Ctfrdist.c	Computes Time-Frequency distance measures between two TFRs
divers.c	—	A set of various basic functions
gradient.c	—	Computes the gradient of a matrix
grd.c	Ctfrgrd.c	Computes the generalized Rectangular distribution
hough.c	Chtl.c	Computes the Hough transform of an image
kernel.c	Ctfrker	Creates a kernel of a given type in the ambiguity plane
mh.c	Ctfrmh.c	Computes the Margenau-Hill distribution
mhs.c	Ctfrmhs.c	Computes the Margenau-Hill-Spectrogram distribution
mmce.c	Ctfrmmce.c	Computes the Minimum mean cross-entropy combination of spectrograms
page.c	Ctfrpage.c	Computes the Page distribution
pmh.c	Ctfrpmh.c	Computes the pseuso Margenau-Hill distribution
ppage.c	Ctfrppage.c	Computes the pseudo Page distribution
pwv.c	Ctfrpwv.c	Computes the pseudo Wigner-Ville distribution
reas_spectro.c	Ctfrrsp.c	Computes the reassigned spectrogram
reassign.c	Ctfrreas.c	Reassigns a TFR, given the field of reassignment vectors
ri.c	Ctfrri.c	Computes the Rihacek distribution
ridb.c	Ctfrridb.c	Computes the reduced Interference Distribution with Bessel kernel
ridbn.c	Ctfrridbn.c	Computes the reduced Interference Distribution with binomial kernel
ridh.c	Ctfrridh.c	Computes the reduced Interference Distribution with Hanning kernel
ridt.c	Ctfrridt.c	Computes the reduced Interference Distribution with Triangular kernel
sp.c	Ctfrsp.c	Computes the Spectrogram
spwv.c	Ctfrspwv.c	Computes the smoothed pseudo Wigner-Ville distribution
stft.c	Ctfrstft	Computes the Short Time Fourier Transform
wv.c	Ctfrwv.c	Computes the Wigner-Ville distribution
zam.c	Ctfrzam.c	Computes the Zao-Atlas-Marks distribution