ISSN 2039-7941
Anno 2011_Numero 214

rapporti tecnici INGV

Multi-Taper Spectra: a *Mathematica*8 TM Function

214

Istituto Nazionale di Geofisica e Vulcanologia

ISSN 2039-7941
Anno 2011_Numero 214

rapporti tecnici INGV

MULTI-TAPER SPECTRA: A *MATHEMATICA*^{8} ^{TM} FUNCTION

Edoardo Del Pezzo and Francesca Bianco

INGV (Istituto Nazionale di Geofisica e Vulcanologia, Sezione di Napoli - Osservatorio Vesuviano)

214

Index

Abstract 5

1. Introduction 5

2. A brief outline of MTA 5
   2.1 Method outline 5
   2.2 Definition of Resolution 6
   2.3 Definition of Variance 6
   2.4 Multi-Taper-Algorithm in *Mathematica* [8] 7
   2.5 The Routines MTS and MTSH 7

3. Application to synthetic data 8
   3.1 Square wave 8
   3.2 Application of the spectral routines MTS and MTSH to square wave example and comparison with FOURIER 9

4. Discussion and comments 11

Appendix. Other examples 13

Acknowledgments 18

References 18

Abstract

Two simple functions written in *Mathematica* $^{8\ TM}$ are here presented for a rapid multitaper spectral estimation. Multitaper ensures an optimal spectral smoothing, as it reduces the variance of the amplitude spectrum estimates using the technique of tapering the time series with a set of orthogonal functions, and then optimally averaging the FFT of the tapered signals. The first Function uses the Slepian n_ prolate functions as tapers, while the second Function uses Hermite Functions as tapers. Both Routines are tested using a synthetic time serie as input. Multitaper smoothing is often used in the spectral analysis of seismic and other geophysical data. The present functions may be useful for geophysicists who are *Mathematica* $^{8\ TM}$ users.

1. Introduction

In numerous sectors of Geophysics spectral analysis is a routine tool for many applications. FFT is a standard, implemented as internal routine in many languages like MATLAB, *Mathematica* or Maple. Accuracy in frequency and amplitude of the spectral estimates becomes important in several geophysical problems, like for istance the determination of the corner frequency of the seismic spectra, or the seeking of particular frequency peaks of signal embedded into the seismic noise. Examples are so numerous that citing becomes impractical. Problems that arises are a) that the computed spectra of time series are affected by spectral leakage, caused the physical limitation of the time series and b) that frequency resolution and variance of the spectral estimate are directly proportional. Smoothing the spectra becomes thus necessary when one wants to focus into the definition of the spectral shape of the signal under study. In this case, the smoothing procedure is equivalent to filter out the spectral roughness, or the spectral-amplitude fluctuations with frequency. Smoothing reduces variance but on the other hand reduces the frequency resolution, and a compromise should be searched in order to achieve the best possible frequency resolution with the lowest possible variance. Classical smoothing of the spectral estimates is generally and easily carried out using running average procedures or filtering. Multi-taper algorithm (MTA) is a special smoothing, designed to optimally reduce the bias resulting from leakage and in the same time reduce the variance of the spectral estimates [Xu et al., 1999 and references therein].

In this paper we describe two *Mathematica* ^{8} functions which use the multitaper algorithm to calculate the smoothed spectrum of a time serie, based on two different ways of generating the tapering functions in time domain, and test both functions with synthetic data. The function itself is made freely downloadable by any *Mathematica* user.

2. A brief outline of MTA

2.1 Method outline

Among the many papers on this argument, the method is well described for specific applications in geophysics [Lees and Park 1995] and in applications in medicine [Xu et al. 1999]. The multi-taper method consists in tapering the single data series with a set of K tapers, Fourier transforming the K tapered signals and finally averaging them in order to obtain its optimally leakagereduced maximum-resolved spectral estimate. The tapers can be calculated in two ways, producing almost equivalent results.

The first (and most used) is based on the calculation a Slepian sequence of K tapers. A Slepian sequence of tapers is given by the eigenvectors of the tri-diagonal matrix S, given by its diagonal elements

$$((N-1-2n)/2)^2\mathrm{Cos}[2\pi\ \ p/N],\ n=0,\ \ldots\ldots,\ N-1 \qquad (1)$$

and off-diagonal elements

$$n(N-n)/2,\ n=1,\ \ldots\ldots N-1 \qquad (2)$$

where N are the data samples and p is an integer. The user selects a suitable frequency band, B, for the data serie sampled at *sps* rate, greater or equal to the minimum frequency resolution, $\frac{1}{T}$, where T is the time duration of the data series. B will be given by

$$B = \frac{2p \cdot \mathrm{sps}}{N} \quad (3)$$

It can be demonstrated [Xu et al., 1999] that the number K of tapers should be set at

$$K < 2p - 1 \quad (4)$$

Tipical values of p and K in practical analysis are 3 and 5. For $p = K = 1$ the method yields the single taper (non-smoothed) Fourier Transform of the signal.

Once that eigenvalues λ_k and corresponding eigenvectors w^k of S have been calculated, the MTA "eigenspectra" can be estimated with

$$Y_k(f) = \sum_{n=1}^{N} w_n{}^k x_n e^{\mathrm{i}2\pi \mathrm{fn/sps}} \quad (5)$$

and their average, weighted for the respective eigenvalues, is an estimate of MTA spectrum

$$Y(f) = \sqrt{\frac{\Sigma_{k=0}^{K-1} \lambda^{-1}{}_k |Y_k(f)|^2}{\Sigma_{k=0}^{K-1} \lambda^{-1}{}_k}} \quad (6)$$

The second way to calculate tapers is using the Hermite functions. The k-th order Hermite function, $h_k(t)$; is defined by

$$h_k(t) = \pi^{-1/4} (2^k k!)^{-1/2} \left(t - \frac{d}{dt} \right)^k e^{-t^2/2} \quad (7)$$

From equation (7) it is easy to calculate the discrete $h^k{}_n$ corresponding to $w^k{}_n$ used in equation (5). Differently from using Slepian functions, the estimate of the MTA spectrum using Hermite functions, correspondent to equation (6), is done with no weights in the average, and is given by

$$Y_H(f) = \sqrt{\frac{\Sigma_{k=0}^{K-1} |Y_k(f)|^2}{K}} \quad (8)$$

2.2 Definition of Resolution

It is well known from spectral analysis that frequency resolution, R_f, in case of no smoothing is given by

$$R_f = \frac{1}{T} \quad (9)$$

where T is the duration of the time series sample. Effective resolution in case of smoothing is KR_f

2.3 Definition of Variance

Variance can be defined as the sum of two terms:

$$\theta = \left| \frac{\Sigma_{k=1}^{K} U_k{}^*(0) Y_k(f_0)}{\Sigma_{k=1}^{K} |U_k(0)|^2} \right|^2 \sum_{k=1}^{K} |U_k(0)|^2 \quad (10)$$

and

$$\psi = \sum_{k=1}^{K} \left| Y_k(f_0) - \left| \frac{\sum_{k=1}^{K} U_k^*(0) Y_k(f_0)}{\sum_{k=1}^{K} |U_k(0)|^2} \right| U_k(0) \right|^2 \tag{11}$$

where U_k is the Fourier Transform of the k-th taper.

The random variable

$$F(f) = (K-1)\frac{\theta}{\psi} \tag{12}$$

obeys a Fisher law with 2K–2 degrees of freedom. Uncertainty can be thus estimated calculating the quantity in equation 12.

2.4 Multi-Taper-Algorithm in *Mathematica* [8]

In this paper we describe two *Mathematica* [8] functions which use the multitaper algorithm to calculate the smoothed spectrum of a time serie, based on both ways of generating the tapering functions (Slepian or Hermite) in time domain, and test both functions with synthetic data. The two functions are both freely downloadable by any *Mathematica* user as explained in the following of the present report.

2.5 The Routines MTS and MTSH

Two routines are here defined: MTS and MTSH. The first uses Slepian tapers, the second hermite functions. Both are single-channel routines. All the command lines are commented, and hence self-explicatory.

```
Clear["Global‘*"];
Hermite[k_,t_]:=*Exp[-0.5*t^2]*HermiteH[k-1,t];
(* The Hermite functions, based upon Hermite Polynomials*)
MTS[FileData_,p_,K_,sps_]:=Module[{NN=Length[FileData],pp=p,KK=K,fsamp=sps},
Factor1=(NN)^0.5/(*fsamp);FFmin1=Fourier[FileData]*Factor1;
AAA=DiagonalMatrix[Table[((NN-1-2*n)/2)2*Cos[2*Pi*pp/NN],
{n,0,NN-1}]]+DiagonalMatrix[Table[n*(NN-n)/2,
{n,1,NN-1}],1]+DiagonalMatrix[Table[n*(NN-n)/2,{n,1,NN-1}],-1];
weights=Eigenvalues[N[AAA],KK];Slep=Eigenvectors[N[AAA],KK];
Slepian=Table[Slep[[k,All]]/Max[Slep[[k,All]]],{k,1,KK}];
SpectralMatrix=Transpose[Table[Factor1*Fourier[FileData*Slepian[[k,All]]],{k,1,KK}]];
Spectrum=Table[\[Sqrt](()/()),{j,1,Length[SpectralMatrix]}];
RatioSm=Mean[Abs[FFmin1]]/Mean[Abs[Spectrum]];
Spectrum*RatioSm];(*Slepian Taper Routine*)
MTSH[FileData_,p_,K_,sps_]:=(*number of data should be odd!!!!*)
Module[{NN=Length[FileData],pp=p,KK=K,fsamp=sps},Factor1=(NN)^0.5/(*fsamp);
FFmin1=Fourier[FileData]*Factor1;
Hermitians=Table[Hermite[k,n/sps],{n,-(NN-1)/2,(NN-1)/2},{k,1,KK}];
mmm=Table[Max[Hermitians[[All,k]]],{k,1,KK}];
Hrmtns=Transpose[Table[Hermitians[[All,k]]/mmm[[k]],{k,1,KK}]];
SpectralMatrix=Transpose[Table[Factor1*Fourier[FileData*Hrmtns[[All,k]]],{k,1,KK}]];
Spectrum=Table[\[Sqrt](),{j,1,Length[SpectralMatrix]}];
RatioSm=Mean[Abs[FFmin1]]/Mean[Abs[Spectrum]];
Spectrum*RatioSm](*Hermite functions routine*)
```

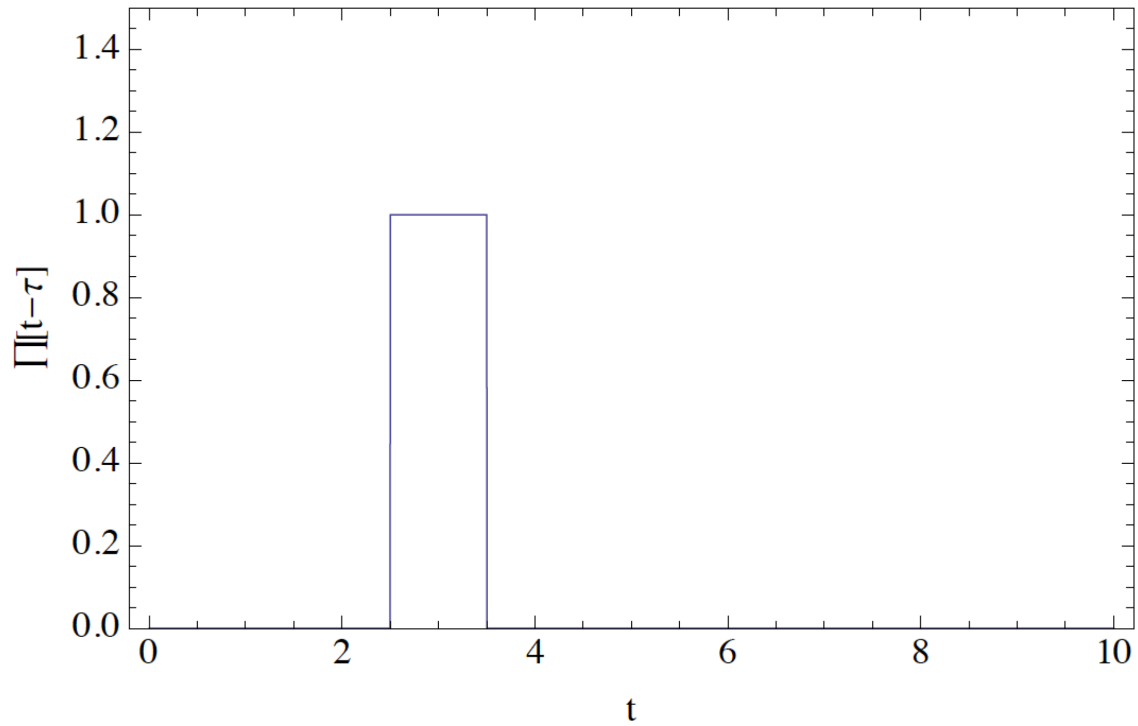

Figure 1. Plot of the UnitBox function (or $\Pi[t-\tau]$) as a function of the independent variable, t. In the present case τ=3 (Lag=3).

3. Application to synthetic data

3.1 Square wave

This data example is a square wave type function (Figure 1) given by the *Mathematica* [8] function UnitBox[x-Lag], or $\Pi[t-\tau]$ where Lag is the time Lag respect to the origin time. In this example, Lag is 3 s. We remark that in the following Figures all the y-coordinates of the Fourier spectra are expressed as "amplitude density" for a generic "amplitude" of the input signal, accounting for the Fourier Transform definition.

```
Lag=3;
Plot[UnitBox[t-Lag],{t,0,10},Frame->True,PlotRange->{0,1.5},
LabelStyle->Medium,FrameLabel->{"t","\[Product][t-τ]"}]
SS[t_,L_]:=UnitBox[t-L];(* Definition of the example function*)
Table[SS[t,3],{t,0,10,0.01}];(*Numerical table of the example function,
at a sampling period of 0.01 s*)
MaxF=Max[Table[SS[t,3],{t,0,10,0.01}]](* max of the example function*)
1
MinF=Min[Table[SS[t,3],{t,0,10,0.01}]](* min of the example function*)
0
PPP=Plot[SS[t,3],{t,0,10},PlotLabel->"TestSquare-Wave",LabelStyle-
>{FontFamily->"Times",16},
PlotStyle->{Black,Thickness[0.005]},PlotRange->{2*MinF,2*MaxF}];
Tewa=Show[PPP,Frame->True,FrameLabel->{"Time (s)","Amplitude"},LabelStyle-
>{FontFamily->"Helvetica",12}];
FourierTransform[UnitBox[t-L],t,ω]
(*MATHEMATICA-performed Fourier Transform of the example
function using the "standard" definition*)
```

```
F1[f_,L_]:=(EI L 2*Pi*f Sinc[(2*Pi*f)/2])/;
(* Theoretical definition of the Fourier Transform of the example
function*)
FTW=Plot[Abs[F1[f,Lag]],{f,0,30},PlotRange->{{0,30},{0,0.5}},
PlotLabel->"Fourier Spectrum of Square-Wave",
LabelStyle->{FontFamily->"Helvetica",12},PlotStyle->{Black,Thickness[0.005]}];
WaFt=Show[FTW,Frame->True,FrameLabel->{"Frequency (Hz)","Amplitude
density"},
LabelStyle->{FontFamily->"Helvetica",12}];
(*Sampling the Test Function at 100 sps for 10 seconds (1001
points)*)
Ndati=1001;
fsampling=100;(*Ndati is the number of samples*)
X=Table[i/fsampling,{i,1,Ndati}];(*X is the vector containing the
sampling times*)
FF1=Table[SS[i/fsampling,Lag],{i,1,Ndati}];(*FF1 is the sampled
function at X. sps=100*)
FF1F=Transpose[Join[{X},{FF1}]] //N;
(* FF1F is the two-columns vector, containing times and corresponding
samples of the example function*);
Sawa=ListPlot[{FF1F},PlotRange->{2*MinF,2*MaxF},Frame->True,
Joined->True,FrameLabel->{"Time (s)","Wavelet Amplitude","Sampled
SquareWave"},
LabelStyle->{FontFamily->"Helvetica",12},PlotStyle->{Black,Thickness[0.005]},
PlotRange->{2*MinF,2*MaxF}](*The Signal Plot*);
GraphicsRow[{Tewa,WaFt}]

GraphicsRow[{Sawa,WaFt}];(*This plot can be activated to check if the sampled signal corresponds to the theoretical one*)
```

3.2 Application of the spectral routines MTS and MTSH to square wave example and comparison with FOURIER

As well known, the Fourier Transform can be defined in several ways. In the present example we define the Fourier transform of a function $f(t)$; the quantity $\frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty} f(t)\mathrm{e}^{i\ \omega\ t}dt$. Numerical Fourier transform is very often carried out using the so called Cooley-Tukey algorithm universally known as FFT. This algorithm is implemented in the *Mathematica* [8] routine FOURIER, which calculates, $\frac{1}{\sqrt{n}}\sum_{r=1}^{n} u_r e^{2\pi i(r-1)(s-1)/n}$ where u_r is the r-th sample of the signal under study, n is the number of samples and s is the transformed variable (frequency if the signal is time-sampled). $\frac{1}{\sqrt{n}}$ is a coefficient used by *Mathematica* [8]. Other softwares may use different normalizations. To re-obtain the correct definition of Fourier transform it is necessary to multiply the result obtained by FOURIER by the following quantity:

$$Factor1 = \sqrt{Ndati} \,/\, fsampling \qquad (13)$$

The normalization factor for the two routines MTS and MTSH is chosen in such a way that the smoothed spectrum, $|F_{\mathrm{Sm}}(\omega)|$ and the non-smoothed spectrum, $|\mathrm{F}(\omega)|$, have the same energy:

$$\int_0^{\omega_n} |F_{\mathrm{Sm}}(\omega)| \mathrm{d}\omega = \int_0^{\omega_n} |F(\omega)| \mathrm{d}\omega$$

where ω_n is the Nyquist frequency.

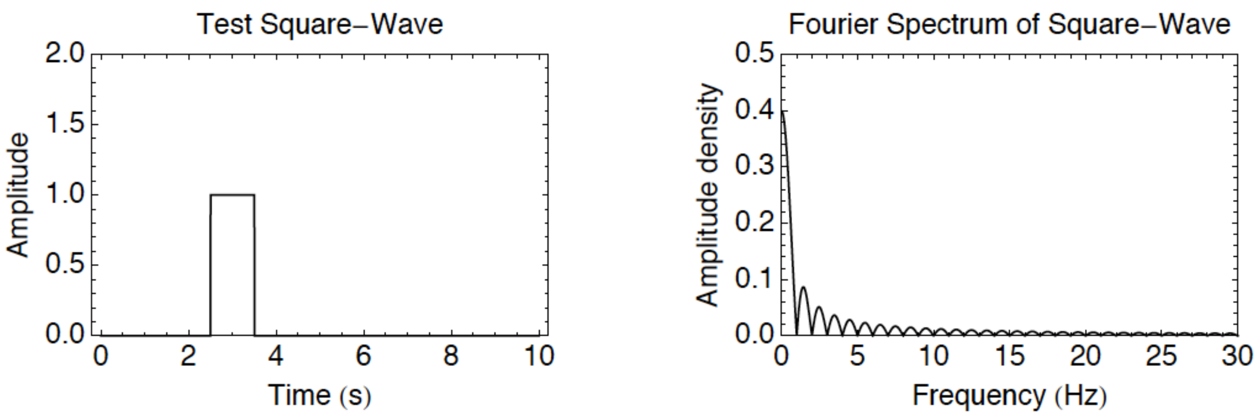

Figure 2. The square wave (left panel) pulse, plotted as a function of time, t, and its Fourier spectrum (right panel) plotted as a function of frequency (Hz).

```
Factor1=(Ndati)^0.5/(*fsampling);
(* This is the scale factor utilized by MATHEMATICA in the
Routine FOURIER*)
FFmin1=Fourier[FF1]*Factor1;
7
(* Normalized numerical Fourier Trasform*)
FFmts=MTS[FF1,3,5,100];(* Normalized Smoothed (Slepian) Fourier
Transform*)
FFmtsH=MTSH[FF1,3,5,100];(* Normalized Smoothed (Hermite)
Fourier Transform*)
fre=Table[fsampling*(n-1)/Ndati,{n,1,Ndati}];(*Frequencies [Hz]*)
Dimensions[FFmts]
{1001}
Dimensions[fre]
{1001}
Dimensions[FFmts]
{1001}
FFmin1F=Transpose[Join[{fre},{Abs[FFmin1]}]] //N;
(*Files utilized for plotting results*)
FFsmooth=Transpose[Join[{fre},{Abs[FFmts]}]]//N;
(*Files utilized for plotting results*)
FFsmoothH=Transpose[Join[{fre},{Abs[FFmtsH]}]]//N;
(*Files utilized for plotting results*)
Spe=ListPlot[FFmin1F,PlotRange->{{0,10},{0,1}},PlotRange->Full,Joined-
>True,
PlotLabel->"Numerical Fourier Transform of Square Pulse",
LabelStyle->{FontFamily->"Times",16},PlotStyle->{Green,Blue,Thickness[0.005]}];
smooth=ListPlot[{FFsmooth,FFsmoothH},PlotRange->{{0,10},{0,0.5}},
Joined->True,Frame->True,FrameLabel->{"Frequency (Hz)","Amplitude
Density"},
PlotLabel->"Smoothed Fourier Spectrum of Square Pulse",
LabelStyle->{FontFamily->"Times",16},PlotStyle->{Red,Black,Thickness[0.005]}];
Teo=Plot[{0.02+Abs[F1[f,Lag]]},{f,0,10},PlotRange->{0,0.5},
PlotLabel->"Analytical Fourier Transform of Square Pulse",
LabelStyle->{FontFamily->"Times",16},PlotStyle->{Orange,Thickness[0.005]}];
teospe=Show[{Teo,Spe},Frame->True,FrameLabel->{"Frequency
(Hz)","Amplitude Density"},
```

LabelStyle->{FontFamily->"Times",16},PlotLabel->"Comparison
between Theoretical and numerical"];
GraphicsRow[{smooth,teospe}]

Show[{teospe,smooth},PlotLabel->"Comparison between Theoretical,
numerical and smoothed"]

4. Discussion and comments

The two Routines MTS and MTSH are here implemented in *Mathematica* [8]. They should work also in previous *Mathematica* releases, but this has been not tested. A different normalization (the present saves the spectral energy respect to that calculated with the non smoothed version) can be easily achieved modifying the last two rows:

RatioSm=Mean[Abs[FFmin1]]/Mean[Abs[Spectrum]];Spectrum*RatioSm

Instead of the mean (which is proportional to the square root of Energy) one can calculate other normalization factors. One that saves the amplitude is given by

RatioSm=Max[Abs[FFmin1]]/Max[Abs[Spectrum]];Spectrum*RatioSm

The user may substitute this last two rows into the two MTS and MTSH routines.

The coefficient of the Fourier Transform utilized in the present paper is $\frac{1}{\sqrt{2\pi}}$, taken from the default definition $F(\omega)=\frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty} f(t)\mathrm{e}^{i\ \omega\ t}dt$. If the user needs to change definition, he/she should use a normalization factor for the internal routine FOURIER different from Factor1.

An user could a) copy the bold part of the present report directly into a *Mathematica* notebook or b) ask to the authors the .nb file writing at the following e-mail address: edoardo.delpezzo@ov.ingv.it.

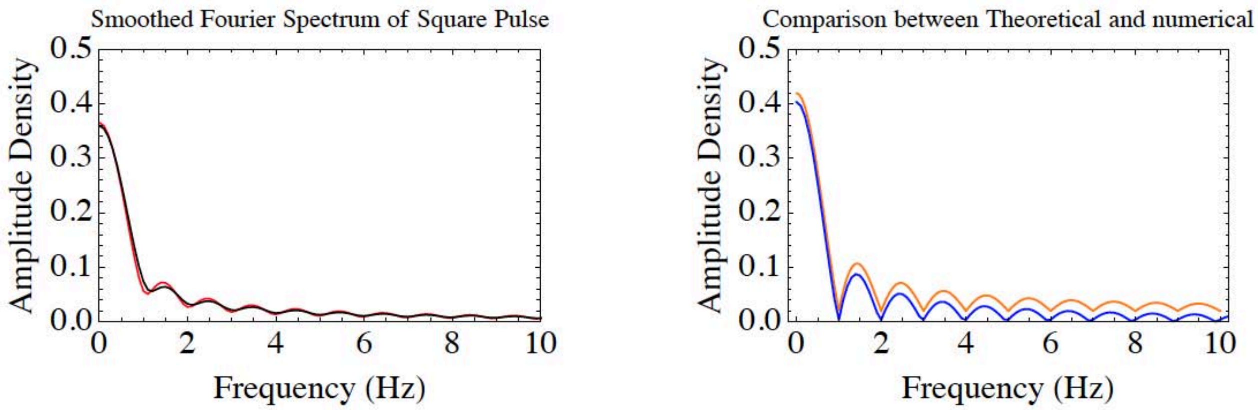

Figure 3. Left panel shows the smoothed Fourier spectrum of the Square Pulse. Red curve - Slepian tapers. Black curve - Hermite tapers. Right Panel shows the comparison between theoretical (Orange curve) and numerical (Blue curve) non-smoothed Fourier spectrum of the square pulse.

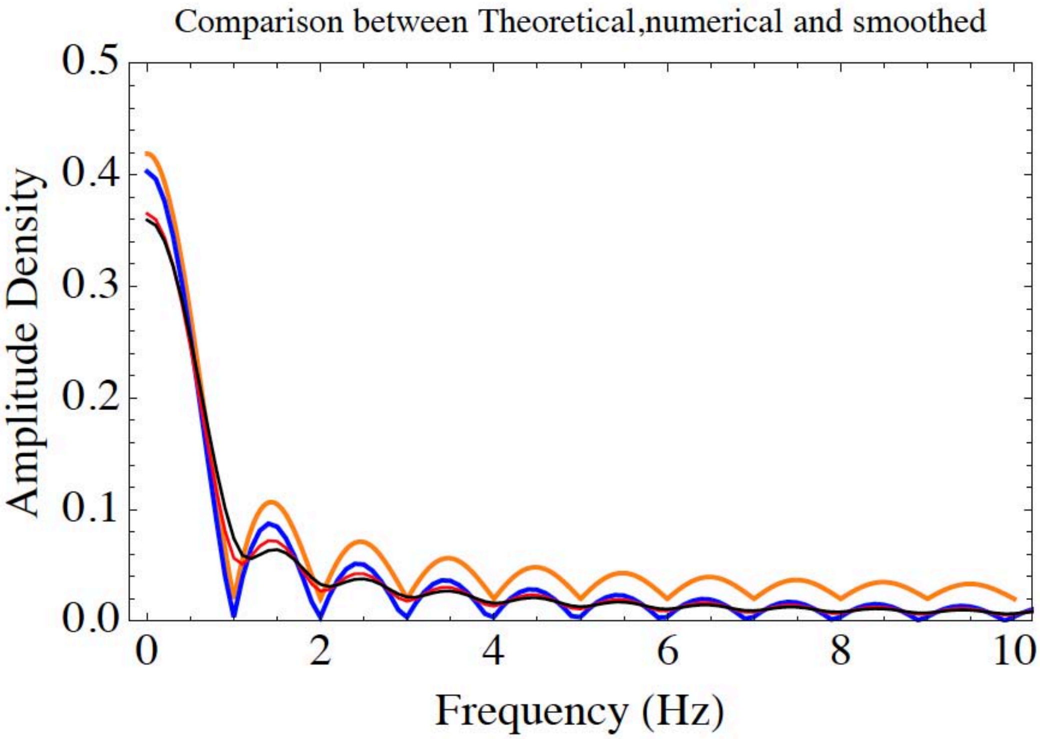

Figure 4. Smoothed Fourier spectrum of the Square Pulse, overimposed to the theoretical and numerical non-smoothed Fourier spectrum. As in Figure 3, the red curve represents Slepian tapers, the black curve the Hermite tapers, the orange curve is the theoretical Fourier spectrum, whilst the blue curve is the numerical one. The shift between orange and blue curve ($\simeq 0.01$) is added for clarity in visualization.

Appendix. Other examples

Other examples are here presented. Synthetic data are generated sampling a) a Sine wave function, b) a Sine-Type wavelet function.

a) Sine wave function given by $S(t) = 3\mathrm{Sin}\,[2\pi f_1 t]+5\mathrm{Sin}\,[2\,\pi f_2 t]$

```
f1=2;
f2=12; SS[t_]:=3*Sin[2*Pi*f1*t]+5*Sin[2*Pi*f2*t];(* Definition of
the example function*)
Table[SS[t],{t,0,10,0.01}];(*Numerical table of the example function,
at a sampling period of 0.01 s*)
MaxF=Max[Table[SS[t],{t,0,10,0.01}]](* max of the example function*)
7.60845
MinF=Min[Table[SS[t],{t,0,10,0.01}]](* min of the example function*)
-7.60845
PPP=Plot[SS[t],{t,0,10},PlotLabel->"TestSine-Function",LabelStyle-
>{FontFamily->"Times",16},
PlotStyle->{Black,Thickness[0.005]},PlotRange->{2*MinF,2*MaxF},
Epilog->{Dashing[{0.015,0.015}],Line[{{0,MaxF},{10,MaxF}}],Line[{{0,MinF},{10,MinF}}]}];
Tewa=Show[PPP,Frame->True,FrameLabel->{"Time (s)","Amplitude"},
LabelStyle->{FontFamily->"Helvetica",12}];
FourierTransform[3*Sin[2*Pi*fa*t]+5*Sin[2*Pi*fb*t],t,ω]
(*MATHEMATICA-performed Fourier Transform of the example function using the "standard" definition*)
3 I DiracDelta[-2 fa π+ω]-3 I DiracDelta[2 fa π+ω]+5 I DiracDelta[-
2 fb π+ω]-5 I DiracDelta[2 fb π+ω]
F1[f_,fa_,fb_]:=3 I DiscreteDelta[-2 fa π+2*Pi*f]-3 I DiscreteDelta[2
fa π+2*Pi*f]
+5 I DiscreteDelta[-2 fb π+2*Pi*f]-5 I DiscreteDelta[2 fb π+2*Pi*f];
(* Definition of the Fourier Transform of the example function*)
FTW=ListPlot[Table[{f,Abs[F1[f,f1,f2]]},{f,0,50,0.01}],PlotRange-
>{0,10},
Joined->True,PlotLabel->"Fourier Spectrum of Sine-Function",
LabelStyle->{FontFamily->"Helvetica",12},PlotStyle->{Black,Thickness[0.005]}];
```

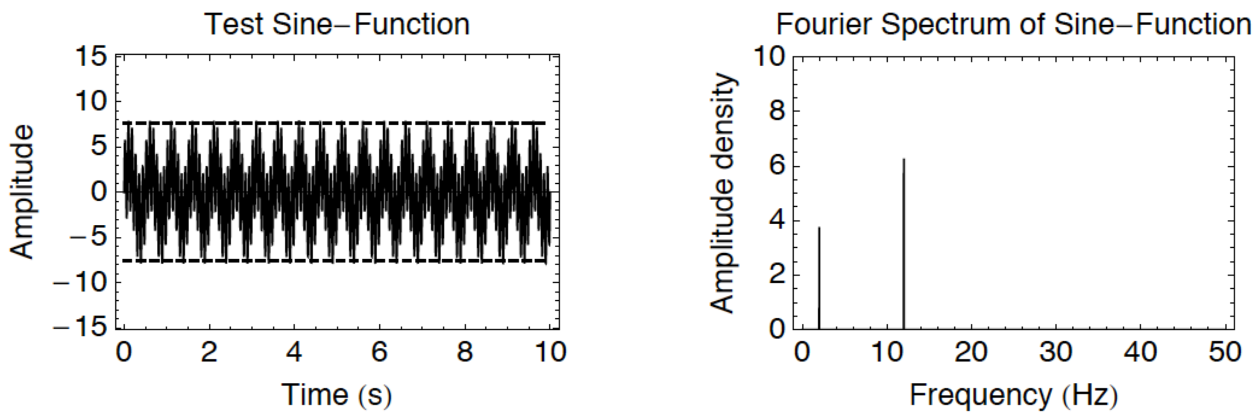

Figure 5. Left panel: Plot of the Sine function as function of the independent variable. Right Panel: Theoretical Fourier Spectrum of the function plotted at left.

```
WaFt=Show[FTW,Frame->True,FrameLabel->{"Frequency (Hz)","Amplitude
density"},
LabelStyle->{FontFamily->"Helvetica",12}];
GraphicsRow[{Tewa,WaFt}]

Ndati=1001;
fsampling=100;
X=Table[i/fsampling,{i,1,Ndati}];
FF1=Table[SS[i/fsampling],{i,1,Ndati}];
FF1F=Transpose[Join[{X},{FF1}]] //N;
Sawa=ListPlot[{FF1F},PlotRange->{2*MinF,2*MaxF},Frame->True,Joined-
>True,
FrameLabel->{"Time (s)","Sine Amplitude","Sampled Sine"},
LabelStyle->{FontFamily->"Helvetica",12},PlotStyle->{Black,Thickness[0.005]},
PlotRange->{2*MinF,2*MaxF},Epilog->{Dashing[{0.015,0.015}],
Line[{{0,MaxF},{10,MaxF}}],Line[{{0,MinF},{10,MinF}}]}](*FF
è il segnale da analizzare*);
GraphicsRow[{Sawa,Tewa}];
Factor1=(Ndati)^0.5/(*fsampling);(* This is the scale factor utilized
by MATHEMATICA in the Routine FOURIER*)
FFmin1=Fourier[FF1]*Factor1;(* Normalized numerical Fourier Trasform*)
FFmts=MTS[FF1,3,5,100];(* Normalized Smoothed (Slepian) Fourier
Transform*)
FFmtsH=MTSH[FF1,3,5,100];(* Normalized Smoothed (Hermite)
Fourier Transform*)
fre=Table[fsampling*(n-1)/Ndati,{n,1,Ndati}];
(*Frequencies [Hz]*)
Dimensions[FFmts]
{1001}
Dimensions[fre]
{1001}
Dimensions[FFmts]
{1001}
FFmin1F=Transpose[Join[{fre},{Abs[FFmin1]}]] //N;
(*Files utilized for plotting results*)
FFsmooth=Transpose[Join[{fre},{Abs[FFmts]}]]//N;
(*Files utilized for plotting results*)
FFsmoothH=Transpose[Join[{fre},{Abs[FFmtsH]}]]//N;
(*Files utilized for plotting results*)
Spe=ListPlot[FFmin1F,PlotRange->{{0,20},{0,10}},PlotRange->Full,
Joined->True,PlotLabel->"Numerical Fourier Spectrum of Sine-
Function",
LabelStyle->{FontFamily->"Times",16},PlotStyle->{Red,Thickness[0.005]}];
smooth=ListPlot[{FFsmooth,FFsmoothH},PlotRange->{{0,20},{0,10}},
Joined->True,Frame->True,FrameLabel->{"Frequency (Hz)","Amplitude
density"},
PlotLabel->"Smoothed Fourier Spectrum of Sine function",
LabelStyle->{FontFamily->"Times",16},PlotStyle->{Green,Blue,Thickness[0.005]}];
Teo=ListPlot[Table[{f,Abs[F1[f,f1,f2]]},{f,0,20,0.01}],
PlotRange->{0,10},Joined->True,PlotLabel->"Fourier Spectrum
of Sine-Function",
LabelStyle->{FontFamily->"Times",16},PlotStyle->{Black,Thickness[0.005]}];
teospe=Show[{Teo,Spe},Frame->True,FrameLabel->{"Frequency
(Hz)","Amplitude density"},
LabelStyle->{FontFamily->"Times",16},
PlotLabel->"Comparison between Theoretical and Numerical"];
```

```
GraphicsRow[{smooth,teospe}]
Show[{teospe,smooth},PlotLabel->"Comparison between Theoretical,
Numerical and Smoothed"]
```

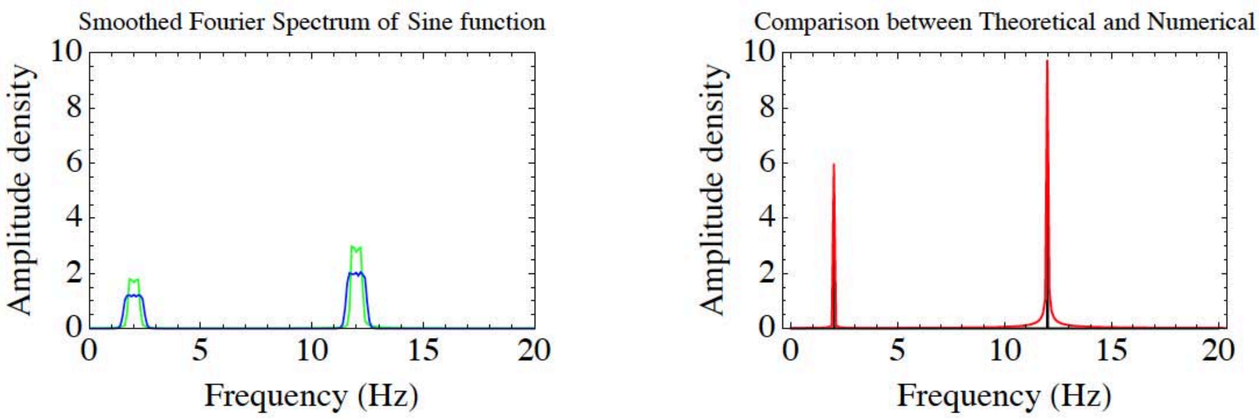

Figure 6. Smoothed numerical spectra (Blue - Hermite smoothing, Green - Slepian smoothing) of Sine function (left panel). A comparison between the theoretical spectrum (black) and the non smoothed numerical spectrum (red) in the right panel.

b) a Sine-Type wavelet function

This is given by $-\dfrac{2e^{-\frac{(t-\tau)^2}{\sigma^2}}(t-\tau)}{\sigma^2}$, where σ is associated with the time duration and τ is the time of the first zero-crossing. In this example, σ is 0.6 s and τ=3 s.

```
sigma=0.6;t0=3;
SS[t_]:=-((2 (t-t0))/sigma2);
(* Definition of the example function*)
Table[SS[t],{t,0,10,0.01}];(*Numerical table of the example function,
at a sampling period of 0.01 s*)
MaxF=Max[Table[SS[t],{t,0,10,0.01}]](* max of the example function*)
1.42946
MinF=Min[Table[SS[t],{t,0,10,0.01}]](* min of the example function*)
-1.42946
PPP=Plot[SS[t],{t,0,10},PlotLabel->"Test Sine-Wavelet",LabelStyle-
>{FontFamily->"Times",16},
PlotStyle->{Black,Thickness[0.005]},PlotRange->{2*MinF,2*MaxF},
Epilog->{Dashing[{0.015,0.015}],Line[{{0,MaxF},{10,MaxF}}],
Line[{{0,MinF},{10,MinF}}],Line[{{t0,MinF},{t0,MaxF}}]}];
Tewa=Show[PPP,Frame->True,FrameLabel->{"Time (s)","Amplitude"},
LabelStyle->{FontFamily->"Times",16}];
FourierTransform[-((2 (t-τ))/σ2),t,ω]
(*MATHEMATICA-performed Fourier Transform of the example
function using the "standard" definition*)
-((I (1/σ^2)3/2 σ4 ω)/)
F1[f_,sig_,to_]:=-((I (1/sig^2)3/2 sig4 2*Pi*f)/ );
(* Definition of the Fourier Transform of the example function*)
FTW=Plot[Abs[F1[f,sigma,t0]],{f,0,10},PlotRange->Full,
```

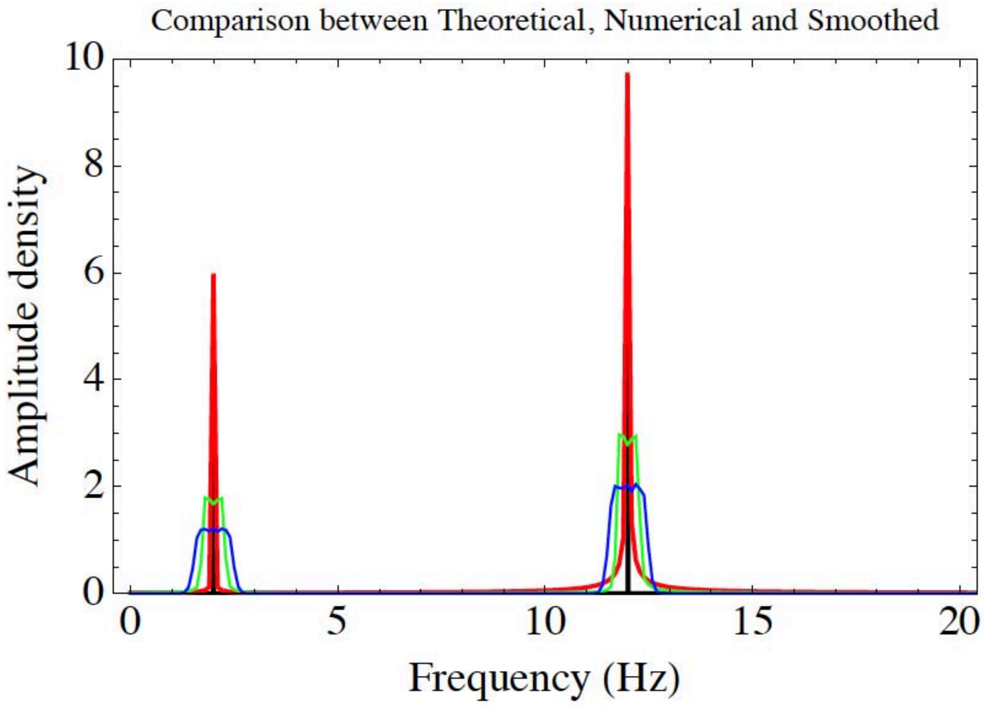

Figure 7. The spectra of Figure 6 superimposed one to each other: smoothed numerical spectra (Blue - Hermite smoothing, Green - Slepian smoothing) of Sine function superimposed to the theoretical spectrum (black) and to the non smoothed numerical spectrum (red).

```
PlotLabel->"Fourier Spectrum of Sine-Wavelet",
LabelStyle->{FontFamily->"Helvetica",12},PlotStyle->{Black,Thickness[0.005]}];
WaFt=Show[FTW,Frame->True,
FrameLabel->{"Frequency (Hz)","Amplitude density"},
LabelStyle->{FontFamily->"Helvetica",12}];
GraphicsRow[{Tewa,WaFt}]
Ndati=1001;
fsampling=100;
X=Table[i/fsampling,{i,1,Ndati}];
FF1=Table[SS[i/fsampling],{i,1,Ndati}];
FF1F=Transpose[Join[{X},{FF1}]] //N;
Sawa=ListPlot[{FF1F},PlotRange->{2*MinF,2*MaxF},
Frame->True,Joined->False,FrameLabel->{"Time (s)","Wavelet Amplitude","
Sampled Wavelet"},
LabelStyle->{FontFamily->"Helvetica",12},
PlotStyle->{Black,Thickness[0.005]},PlotRange->{2*MinF,2*MaxF},
Epilog->{Dashing[{0.015,0.015}],Line[{{0,MaxF},{10,MaxF}}],
Line[{{0,MinF},{10,MinF}}],Line[{{t0,MinF},{t0,MaxF}}]}]
(*FF è il segnale da analizzare*); Factor1=(Ndati)^0.5/(*fsampling);
(* This is the scale factor utilized by MATHEMATICA in the Routine FOURIER*)
FFmin1=Fourier[FF1]*Factor1;(* Normalized numerical Fourier Trasform*)
FFmts=MTS[FF1,3,5,100];(* Normalized Smoothed (Slepian) Fourier
Transform*)
FFmtsH=MTSH[FF1,3,5,100];(* Normalized Smoothed (Hermite)
Fourier Transform*)
fre=Table[fsampling*(n-1)/Ndati,{n,1,Ndati}];(*Frequencies [Hz]*)
Dimensions[FFmts]
{1001}
```

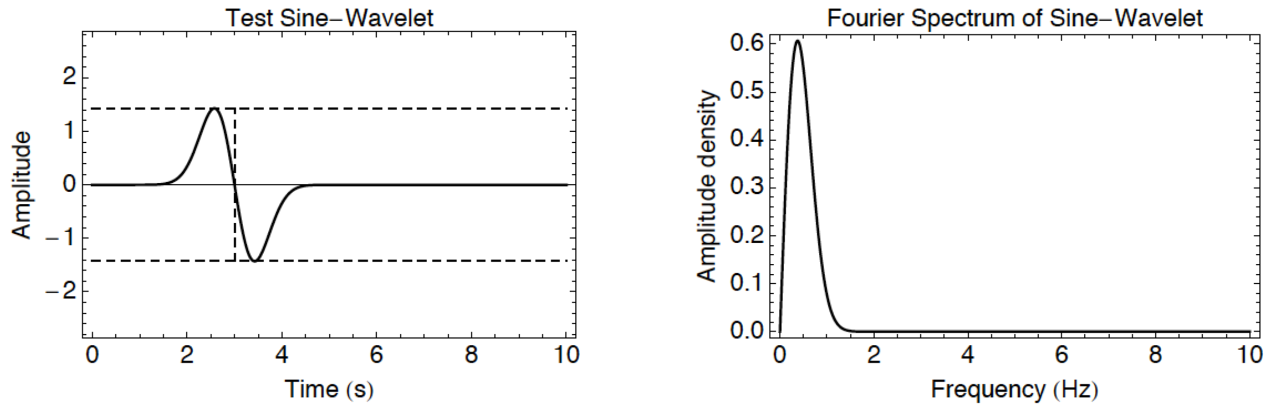

Figure 8. Left panel: plot of the Sine-wavelet function, as a function of time. Right panel: its Fourier spectrum (Theoretical).

```
Dimensions[fre]
{1001}
Dimensions[FFmts]
{1001}
FFmin1F=Transpose[Join[{fre},{Abs[FFmin1]}]] //N;
(*Files utilized for plotting results*)
FFsmooth=Transpose[Join[{fre},{Abs[FFmts]}]]//N;
(*Files utilized for plotting results*)
FFsmoothH=Transpose[Join[{fre},{Abs[FFmtsH]}]]//N;
(*Files utilized for plotting results*)
Spe=ListPlot[FFmin1F,PlotRange->{{0,10},{0,1}},PlotRange->Full,
Joined->True,PlotLabel->"Numerical Fourier Transform of Cosine
Wavelet",
LabelStyle->{FontFamily->"Times",16},PlotStyle->{Red,Thickness[0.005]}];
smooth=ListPlot[{FFsmooth,FFsmoothH},PlotRange->{{0,10},{0,1}},
```

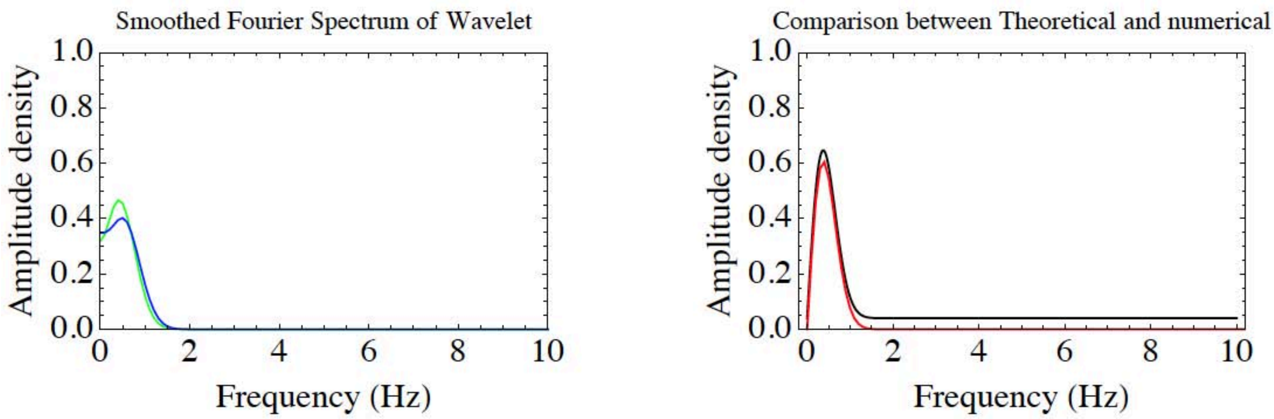

Figure 9. Left panel: Smoothed (Blue Hermite smoothing. Green Slepian smoothing) Fourier spectrum of the Sine-wavelet. Righ panel: Theoretical (Red) and numerical (Black, non smoothed) Fourier spectrum of Sine-wavelet.

```
Joined->True,Frame->True,FrameLabel->{"Frequency (Hz)","Amplitude
density"},
PlotLabel->"Smoothed Fourier Spectrum of Wavelet",
LabelStyle->{FontFamily->"Times",16},PlotStyle->{Green,Blue,Thickness[0.005]}];
Teo=Plot[{0.04+Abs[F1[f,sigma,t0]]},{f,0,10},PlotRange->{0,1},
PlotLabel->"Analytical Fourier Spectrum of Wavelet",
LabelStyle->{FontFamily->"Times",16},PlotStyle->{Black,Thickness[0.005]}];
teospe=Show[{Teo,Spe},Frame->True,FrameLabel->{"Frequency
(Hz)","Amplitude density"},
LabelStyle->{FontFamily->"Times",16},
PlotLabel->"Comparison between Theoretical and numerical"];
GraphicsRow[{smooth,teospe}]
```

Acknowledgments

Mathematica [8] license in use from the authors has been financed by project UNREST (INGV-DPC). The code (.nb) can be freely downloadable at the following site: http://istituto.ingv.it/l-ingv/produzione-scientifica/rapporti-tecnici-ingv/copy_of_numeri-pubblicati-2010/2011-12-16.0210434567/download.

References

Yan Xu, Simon Haykin, and Ronald J. Racine, (1999). *Multiple Window Time-Frequency Distribution and Coherence of EEG Using Slepian Sequences and Hermite Functions*, IEEE TRANSACTIONS ON BIOMEDICAL ENGINEERING, VOL. 46, NO. 7. pp 861-866.

Jonathan Lees and Jeffrey Park (1995). *Multiple Taper Spectral Analysis: a Stand-Alone C-Subroutine*. Computer and Geosciences, 21, 2, 199-236 19.

Coordinamento editoriale e impaginazione
Centro Editoriale Nazionale | INGV

Progetto grafico e redazionale
Daniela Riposati | Laboratorio Grafica e Immagini | INGV

Via di Vigna Murata, 605
00143 Roma
Tel. +39 06518601 Fax +39 065041181

http://www.ingv.it

Istituto Nazionale di Geofisica e Vulcanologia