FFT's, Ensembles and Correlations

B. A. Grierson*
Department of Applied Physics and Applied Mathematics
Columbia University, New York, NY 10027
(Dated: August 30, 2006)

Statistics using the FFT and Correlation Analysis. The FFT (in IDL) takes a time series, and attempts to decompose it into a sum of sine and cosine functions. This monograph investigates the FFT as implemented in the programming language IDL. The simple FFT is then extended to looking at ensemble spectra, cross coherence, etc ... The Correlation function is also examined in relation to the time series created by spatially separated diagnostics.

I. INTRODUCTION

Fourier Analysis is the most standard signal processing technique. It transforms a function via an integral (or sum) of sine and cosine function which can represent the function in an orthogonal basis. A periodic function can be represented by

$$f(x) = \frac{A_0}{2} + \sum_{m=1}^{\infty} A_m \cos mkx + \sum_{m=1}^{\infty} B_m \sin mkx \tag{1}$$

The coefficients are computed using

$$A_m = \frac{2}{\lambda}\int_0^{\lambda} f(x)\cos mkx\ dx \tag{2}$$

$$B_m = \frac{2}{\lambda}\int_0^{\lambda} f(x)\sin mkx\ dx \tag{3}$$

The Discrete Fourier Transform of an N-element, one-dimensional function is given by

$$F(u) = \frac{1}{N}\sum_{x=0}^{N-1} f(x)\exp[-j2\pi ux/N] \tag{4}$$

and is the method most often used numerically.

II. THE FFT

The FFT is one of the crowning achievements of computational mathematics. It allows a fast, accurate method of extracting the dominant frequencies of a time series. Classically, the Fourier Transform of a signal `signal=SIN(2 * !PI * f0)` will give a delta function in frequency space. The delta function exists at $\pm$ `f0` with an amplitude of 0.5 at each peak, and the spectrum is symmetric. The FFT method will be implemented as follows in the IDL language.

- The time series will be represented by the variable `signal` with corresponding time vector `time`
- The number of points in the time series will be denoted by `nt`
- The time step will be defined as `dt=time(1)-time(0)`
- The Digitizing Frequency is `df=1.0/dt`
- The Nyquist Frequency is `Nyquist=df/2.0`
- The signal must have a zero mean, accomplished by `signal=signal-MEAN(signal)`
- The signal will be windowed using a simple triangular window, which is like [0,...,1,...,0], to eliminate edge effects

  `window=[FINDGEN(nt/2)/(nt/2),REVERSE(FINDGEN(nt/2-1))/(nt/2)]`

  Then `signal=signal*window`

Now the FFT is taken, and this is where is gets tricky. The function `result=FFT(X)` in IDL returns a complex valued array, with a real and complex part. The real part has a peak of 0.25 at ± `f0`, while the imaginary part is antisymmetric. So, one quarter of the power is in each part. One quarter in the real part at + `f0`, one quarter in the real part at − `f0`,... etc. Because the function is symmetric we only take the positive half. Then we take the absolute value `ABS(Z)`≡`SQRT(Z*CONJ(Z))`. This function is multiplied by `4.0` to achieve power of `1.0` at the peak.

It is of note that we are defining the amount of power at one frequency for a pure, infinite sinusoid with amplitude on [-1,1] to be `1.0`. If the sinusoid amplitude is different, then the power will reflect that.

Also, because we are not working with an infinite time series, the FFT cannot resolve frequencies lower than $1/\Delta t$ where Δt is the length of the time sample, and these values are thrown out.

III. ABSOLUTE VALUE FFT CODE

The code for this function is included:

```
FUNCTION MY_ABS_FFT,signal,time,FVALS=fvals,FATMAXP=fatmaxp
; Returns the absolute value of the FFT in kHz
; called by
; fspec=MY_ABS_FFT(sig,nt,dt,FVALS=fvals,FATMAXP=fmax)
nt=N_ELEMENTS(time)
delT=time(nt-1)-time(0)
dt=time(1)-time(0)
lowFreqCutoff=1.0/delT/1000.0 ;in kHz
window=[FINDGEN(nt/2)/(nt/2),REVERSE(FINDGEN(nt/2-1))/(nt/2)]
fvalues=FINDGEN(nt/2)/(dt*nt)/1000.0 ;in kHz
fsignal=signal-MEAN(signal)
fsignal=signal*window
ff=FFT(fsignal,-1)
ff=ABS(ff)    ;Comment out this line for MY_FULL_FFT
ff2=4.0*ff(0:nt/2-1)
ff2(WHERE(fvalues LE lowFreqCutoff))=0.0*ff2(WHERE(fvalues LE lowFreqCutoff))
fatmaxp=fvalues(WHERE(ff2 eq MAX(ff2))) ;in kHz
return,ff2
END
```

IV. ENSEMBLE AVERAGES

When one is interested in the statistics of a fluctuating quantity over a long period of time, it can be advantageous to examine 'Ensemble Averages' of functions like the FFT. An ensemble average of a quantity A (usually denoted as $< A >$) can be expressed as

$$< A >= \frac{1}{M} \sum_{i=1}^{M} A \tag{5}$$

where A can be a function of one or more variables. An ensemble can be calculated 'in place', or 'after the fact'. I prefer the latter, because it allows contour plots on the function in time.

For example, the ensemble spectrum of a time series can be performed two equally valid ways. Say we have a time series on [0,1] seconds. We could perform 100 FFTs, add them up, and divide by 100. This would be accomplished by the following code.

```
delT=0.01
tStart=0.0
tStop=delT

FOR i=0,99 DO BEGIN

frameTime=time(WHERE(time GE tStart AND time LE tStop))
```

```
frameSignal=signal(WHERE(time GE tStart AND time LE tStop))

frameFFT=MY_ABS_FFT(frameSignal,frameTime,FVALS=fvals)

IF i EQ 0 THEN ensembleFFT=frameFFT/100.0 ELSE ensembleFFT+=frameFFT/100.0

tStart=tStart+delT
tStop=tStop+delT
ENDFOR
END
```

This is a completely valid method, but I prefer to store each FFT in a rectangular grid rather than adding to a vector. Storing each FFT and contour plotting creates a spectrograph, or TFD (time-frequency domain plot), where the power is the z-axis of the contour. The spectrograph (Fig.1) will reveal any frequency-sweeping, which is a time evolution of the frequency where the power of the signal resides. One example of such a phenomena is a Hot Electron Interchange burst in a dipole confined plasma. The mode creates quickly rising tones, which the ensemble spectrum would 'smear out' when adding.

Once the TFD grid has been formed, the ensemble spectrum can be formed by taking the mean of each FFT at the specific frequency, i.e., integrating in time. Furthermore, the grid can be refined by taking small steps in time, less than the time window `delT`, and create quite beautiful plots.

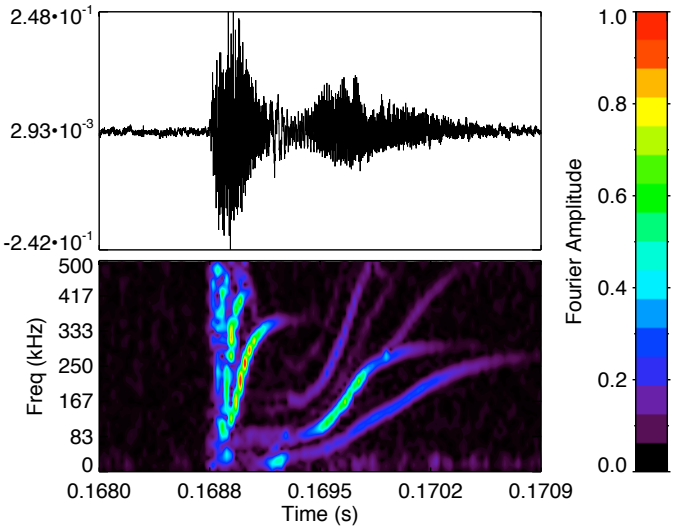

FIG. 1: A time-frequency domain (TFD) contour plot, consisting of individual FFTs placed in a rectangular array.

V. PHASE INFORMATION

When one wishes to extract phase information when comparing two signals, the full FFT is needed. The phase between two signals is defined as

$$\alpha = \tan^{-1}(\Im(Z)/\Re(Z)) \tag{6}$$

Numerically, Z is a function, and not a simple ordered pair of values. Therefore α is a **function in frequency space**. In order to extract the phase difference between two signals, we need a way to pick out which values of $\varphi(f)$ correspond to a **high correlation** in the frequency domain. This is accomplished as follows.

Two signals, `signal1` and `signal2` are needed to extract phase information. We also need the full FFT spectra for each signal, which means taking out the `ABS( )` in the FFT code. This retains the real and imaginary part of each spectrum.

Next the cross power frequency spectrum (Z) is formed by `specCorr=spectrum1*CONJ(spectrum2)`.

The phase function is formed by `phaseFun=ATAN(IMAGINARY(specCorr),REAL_PART(specCorr))`

The code for extracting the phase information from two signals is given below

VI. PHASE CODE

```
FUNCTION GET_PHASE,signal1,signal2,time,$
FVALS=fvals,$
ABSSPECCORR=AbsPowerCorr,$
PHVAL=phaseValue,$
PHASEVALX=phaseValueX,$
PHASEVALAMP=phaseValueAmp,$

dt=time(1)-time(0)
delT=time(N_ELEMENTS(time)-1)-time(0)
lowFreqCutoff=1.0/delT/1000.0 ;In kHz

spec1=MY_FULL_FFT(signal1,time,FVALS=fvals)
spec2=MY_FULL_FFT(signal2,time,FVALS=fvals)

spec1(WHERE(fvals LE lowFreqCutoff))=0.0*spec1(WHERE(fvals LE lowFreqCutoff))
spec2(WHERE(fvals LE lowFreqCutoff))=0.0*spec2(WHERE(fvals LE lowFreqCutoff))

powerCorr=spec1*CONJ(spec2)
AbsPowerCorr=ABS(powerCorr)
phaseFun=-ATAN(IMAGINARY(powerCorr),REAL_PART(powerCorr))
gdIndex=WHERE( AbsPowerCorr EQ MAX(AbsPowerCorr) )
phaseValue=phaseFun(gdIndex)
phaseValueX=fvals(gdIndex)
phaseValueAmp=MAX(powerCorr)

RETURN,phaseFun
END
```

The code will return the value of the phase between the signals, in radians, to the value `phaseValue`, occurring at frequency `phaseValueX`, named so for plotting purposes, with cross power amplitude `phaseValueAmp`. These are the three variables to track phase shifts in time, and record their relative strengths.

In practice, when signals are not pure sinusoids, an average needs to be taken of the phase values near where the power correlation is a maximum. Therefore, one could grab the phase values where `powerCorr` is grater than 90% of it's maximum value, and use and average of those values to define the phase.

If the process being analyzed is quasi-stationary, then new, more sophisticated tools need to be used to extract the phase shift and other relevant statistics in time. We define the following notation and statistical quantities:

$$\text{Fast Fourier Transform} \qquad S(t) \rightarrow_{FFT} \rightarrow \hat{S}(\omega) \tag{7}$$

$$\text{Power Correlation} \qquad \hat{C}_{1,2}(\omega) \equiv \hat{S}_1 \hat{S}_2^* \tag{8}$$

$$\text{Cross Phase} \qquad \alpha_{1,2} \equiv \tan^{-1}\left(\frac{\Im[\hat{C}_{1,2}(\omega)]}{\Re[\hat{C}_{1,2}(\omega)]}\right) \tag{9}$$

$$\text{Power Weighted Ensemble Cross Phase} \qquad < \alpha_{1,2} > \equiv \frac{\int \alpha_{1,2}(\omega)|\hat{C}_{1,2}(\omega)|d\omega}{\int |\hat{C}_{1,2}(\omega)|} \tag{10}$$

$$\text{Power Weighted Ensemble Cross Coherence} \qquad < \gamma_{1,2}^2 > \equiv \frac{< |\hat{C}_{1,2}^2| >}{< |\hat{S}_1| >< |\hat{S}_2| >} \tag{11}$$

$$\text{Bispectrum} \qquad \hat{B}(\omega_1, \omega_2) = \hat{S}(\omega_1)\hat{S}(\omega_2)\hat{S}^*(\omega_1 + \omega_2) \tag{12}$$

$$\text{Bicoherence} \qquad \hat{b}^2(\omega_1, \omega_2) = \frac{< |B(\omega_1, \omega_2)|^2 >}{| < \hat{S}(\omega_1)\hat{S}^*(\omega_1) > |^2 | < \hat{S}(\omega_1, \omega_2) > |^2} \tag{13}$$

VII. CORRELATION ANALYSIS

The Correlation Function is a function of one signal (auto) or between signals (cross) which allows extraction of **lag time** and **correlation time**. The method integrates one signal (the refrence signal), and another signal shifted in time. The 'shift' parameter is defined as τ, and is the variable which the function $C_{1,2}$ is plotted against. The equation is

$$C_{1,2} = \int S_1(t)S_2(t-\tau)dt \tag{14}$$

and can be normalized, bounding $C_{1,2}(\tau) \in [-1,1]$, via

$$C_{1,2} = \frac{\int S_1(t)S_2(t-\tau)dt}{\sqrt{\int S_1^2(t)dt \int S_2^2(t)dt}} \tag{15}$$

Here we have defined the 'Cross Correlation Function', where the 'Auto Correlation Function' is defined as $C_{1,1}(\tau)$.

Consider the following illustrative example: If one wished to measure the velocity of a light source moving in a straight line, and one only had two photodiodes, how would one accomplish the task?

Placing the photodiodes in the path of the light source and recording the light intensity in time, together with correlation analysis could yield the velocity. Two photodiodes, placed Δx apart, and two light intensity traces with lag time τ_{Lag} from cross correlation, gives the velocity of the source as $\Delta x/\tau_{Lag}$. The sample principle can be applied to any sensors separated in space.

The correlation time τ_{Corr} defines the time over which the correlation function decays in amplitude by half, and can determine the 'lifetime' of an 'event' in an oscillating time series.

The method for creating the correlation function is defined as follows:

- Two signals are read into the routine, using `signal1` as the 'refrence' signal.
- The *middle half* (from `nt/2` to `3*nt/2`) of `signal1` times the *middle half* of `signal2` is integrated over the time window which `signal1` exists. This is the **correlation coefficient**.
- Now, `signal2` is shifted forward and backward in time, by the digitizing time step, and integrated at each shift.
- It is of note that this is **not** a circular shifting routine, because we cannot assume that the function is circular or strictly periodic. We are 'padding' the function with more real data, hence the oversampling by $\times 2$
- Because only the middle half of the signal sample is used, the time width **must** be much longer than the characteristic frequency of fluctuations (`1/delT`> few $\times$ characteristic frequency). If the fluctuations reside near 1kHz, than a time window of 5-10ms would be sufficient not to 'miss' the event.

As an example of auto and cross correlation, consider the oscillatory, enveloped functions

$$\begin{aligned} S_1(t) &= \cos[2\pi f_1 t]\exp[-50(t-0.5)^2] \\ S_2(t) &= \cos[2\pi f_1(t-0.05)]\exp[-50((t-0.05)-0.5)^2] \end{aligned} \tag{16}$$

where $S_2(t)$ has been shifted forward in time 0.05 time units, i.e. there is a lag time between S_1 and S_2. If S_1 is taken as reference, then S_2 lags S_1, and $\tau_{Lag} > 0$. This is a very important definition, because it can determine **direction** of propagation. The signals (Fig.2) and correlation functions (Fig.3) are plotted below.

The correlation analysis, just as the Fourier Spectrum, can be extended to examine fluctuations in time. Included in Fig.4 are plots of the auto and cross correlation for two probes spatially separated measuring floating potential in a plasma device. The correlation functions are for the same instability bursts seen in the above TFD Fig.1.

---

* Electronic mail: bag2107@columbia.edu; `http://www.ap.columbia.edu/ctx/ctx.html`; We gratefully acknowledge the support of your mom.

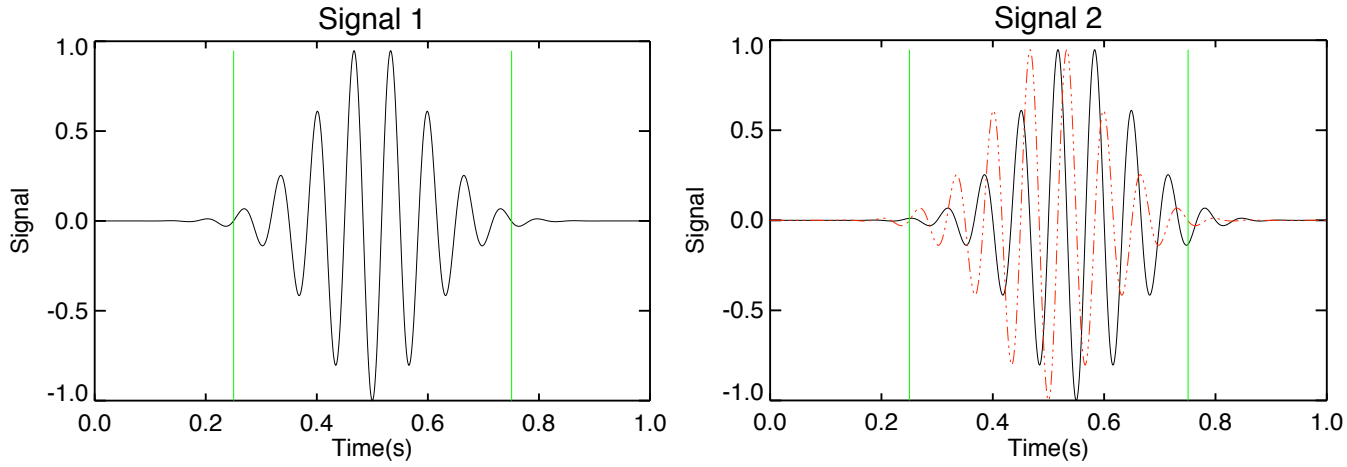

FIG. 2: $S_1(t)$ and $S_2(t)$: S_1 is overplotted against S_2 in red. The green bars determine the middle half of the time window. It can be seen from the plot that S_2 lags S_1, i.e. occurs at a later time.

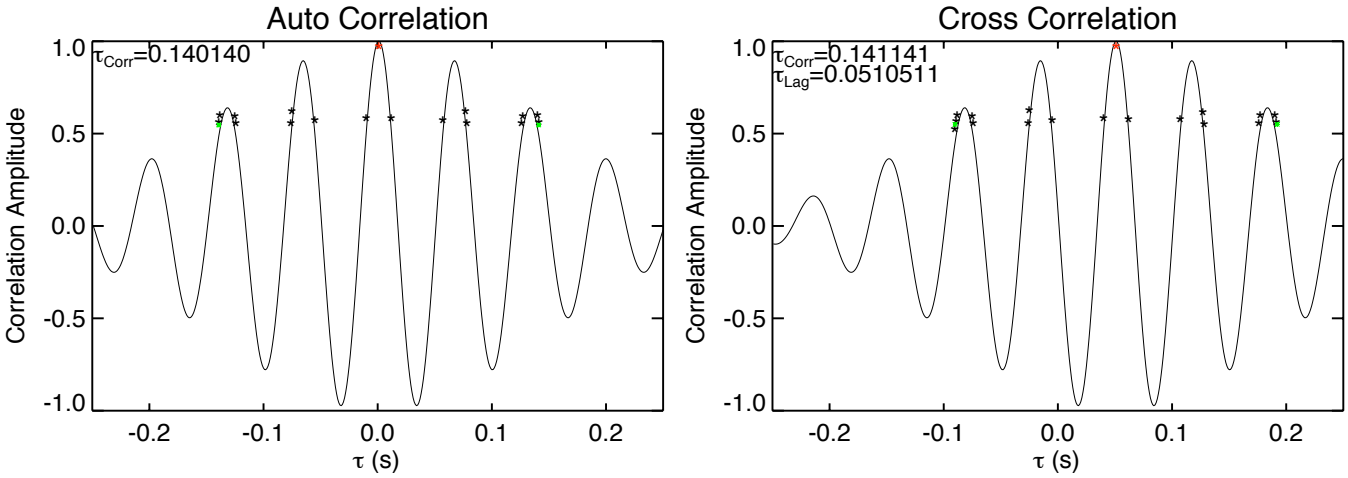

FIG. 3: Correlation Functions: The autocorrelation lag time $\tau^A_{Lag} \equiv 0$, whereas the cross correlation lag time $\tau^C_{Lag} \approx +0.05$, as expected. Furthermore, $\tau^C_{Lag} < \tau^A_{Corr}, \tau^C_{Corr}$, which guarantees we are looking at the same 'event'.

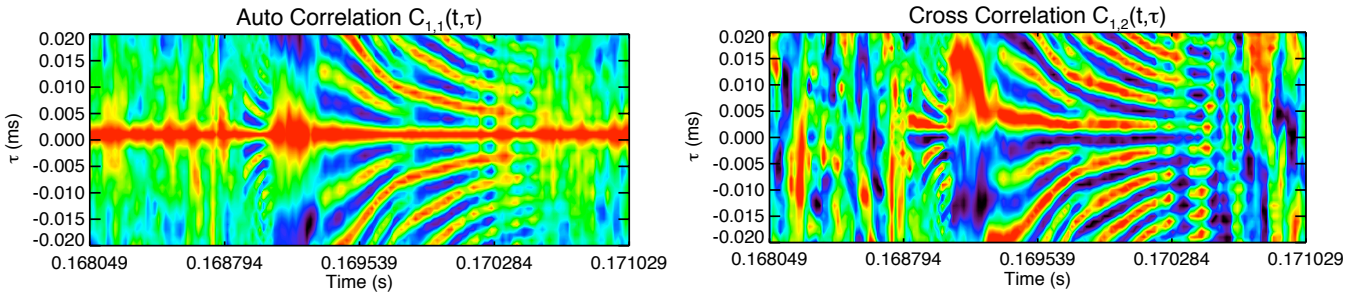

FIG. 4: Auto (left) and Cross (right) Correlation functions in time. It can be seen in the second figure that there exists a peak(red) which evolves in time, but maintains a positive τ^C_{Corr} for the duration of the instability burst. The second probe was located 'downstream' of the first probe, and the positive lag time confirms the direction of 'propagation'.