The FrequenC Library

W.J. Riley
Hamilton Technical Services

● Introduction

The FrequenC Library© is a collection of over 100 C functions for the analysis of frequency stability. It solves the problem of researching, developing, testing, documenting, and validating custom algorithms and computer code to perform such operations as calculating the normal, modified and total versions of the Allan and Hadamard variances. It is the basis of most of the underlying functionality of the widely-used Stable32 program for frequency stability analysis. The FrequenC functions are available both as C source code and as a 32-bit dynamic link library (DLL) that can be linked with a Microsoft Windows® program

● FrequenC Functions

The FrequenC Library includes many of the specialized functions needed to analyze time and frequency data. Included are functions for conversion between time and frequency data, conversion between time and frequency domains, and the calculation of various Allan, Hadamard and total variances from phase or frequency data. The library also includes functions for drift calculation and removal, common statistical functions, and several special functions for identifying noise types and determining confidence intervals. These functions use an array data format that defines analysis limits and can include gaps. The following is an alphabetical list of the functions in the current Version 3.0 of the FrequenC Library:

```
ACFNoiseID()
AddPSD()
AddSigma()
AutocorrelationCalc()
AverageFreqData()
AveragePhaseData()
BasScale()
BreakDate()
CalcBias1()
CalcBias2()
CalcBias3()
CalcBias()
CalcBisectionDrift()
CalcChiSqrProb()
CalcDegFree()
CalcDiffusionFreqDrift()
CalcDomain()
CalcFastMTIE()
CalcFastModSigma()
CalcFirstDiff()
CalcFreqHadamardDev()
CalcFreqModSigma()
CalcFreqOffset()
CalcFreqOverlapHadamardDev()
CalcFreqOverlapSigma()
CalcFreqSigma()
```

```
CalcFreqStdDev()
CalcGreenhallModSigma()
CalcHadamardB1()
CalcHadamardDev()
CalcInvChiSqr()
CalcLinFreqDrift()
CalcLogFreqDrift()
CalcMTIE()
CalcMean()
CalcNormalProb()
CalcPhaseHadamardDev()
CalcPhaseModSigma()
CalcPhaseOverlapHadamardDev()
CalcPhaseOverlapSigma()
CalcPhaseSigma()
CalcPhaseStdDev()
CalcQuadraticDrift()
CalcRatio()
CalcSecondDiff()
CalcStarB1()
CalcThreePointDrift()
CheckFrequenC()
CombinedEDF()
ConvDomain()
ConvFreqToPhase()
ConvPhaseToFreq()
ConvPhaseToFreqUsingTimetags()
CountEqualTimetags()
CountGaps()
DateConv()
DateToJulian()
DateToMJD()
DayOfWeek()
EDF()
FillFloatGaps()
FillGaps()
FindFreqOutliers()
FindMedian()
FindMinMax()
FindPlotScale()
GenNoise()
GetMJD()
HadTotvarBias()
HadTotvarCalc()
HadTotvarEDF()
HadamardEDF()
HistoCalc()
JulianToDate()
MJDToDate()
MJDtoDOY()
MJDtoGPS()
MakeDate()
MedDev()
ModTotvarBias()
ModTotvarCalc()
```

```
ModTotvarEDF()
MultiTaperSpectrumCalc()
NoiseID()
NormalizeData()
RemoveDiffusionFreqDrift()
RemoveFreqOffset()
RemoveLinFreqDrift()
RemoveLogFreqDrift()
RemoveQuadraticDrift()
RoundAxes()
ScaleData()
SpectrumCalc()
TIErms()
Theo1()
Theo1Bias()
Theo1BiasToAlpha()
Theo1EDF()
TotvarBias()
TotvarCalc()
TotvarEDF()
```

- **Function Descriptions**

The following tables describe the FrequenC Library functions:

- **Allan Variance Functions**

Allan Variance Functions	
CalcPhaseSigma	Calculate Allan deviation for phase data
CalcFreqSigma	Calculate Allan deviation for frequency data
CalcPhaseOverlapSigma	Calculate Allan deviation for phase data using overlapping samples
CalcFreqOverlapSigma	Calculate Allan deviation for frequency data using overlapping samples
CalcPhaseModSigma	Calculate modified Allan deviation for phase data
CalcFreqModSigma	Calculate modified Allan deviation for frequency data
CalcFastModSigma	Quickly calculate modified Allan deviation for gapless phase data
CalcGreenhallModSigma	Calculate mod sigma for gapless phase data using Greenhall method

- **Hadamard Variance Functions**

Hadamard Variance Functions	
CalcHadamardDev	Calculate the 3-sample Hadamard deviation for phase data
CalcFreqHadamardDev	Calculate the 3-sample Hadamard deviation for frequency data
CalcPhaseHadamardDev	Calculate the Hadamard deviation for phase data
CalcPhaseOverlapHadamardDev	Calculate the Hadamard deviation for phase data using overlapping samples
CalcFreqOverlapHadamardDev	Calculate the Hadamard deviation for freq data using overlapping samples

- ## Total Variance Functions

Total Variance Functions	
TotvarCalc	Calculate the total Allan deviation for phase data
ModTotvarCalc	Calculate the total modified Allan deviation for phase data
HadTotvarCalc	Calculate the total Hadamard deviation for phase data

- ## Thêo1 Functions

Thêo1 Functions	
Thêo1	Calculate Thêo1 for phase data

- ## Other Stability Functions

Other Stability Functions	
TIErms	Calculate TIErms for phase data
CalcMTIE	Calculate MTIE for phase data
CalcFastMTIE	Calculate MTIE for phase data using the fast method

- ## Autocorrelation Functions

Autocorrelation Functions	
AutocorrelationCalc	Calculate the autocorrelation function

- ## Noise Generation and Identification Functions

Noise Generation and Identification Functions	
GenNoise	Generate pseudo-random power law noise data
ACFNoiseID	Identify the power law noise type using the lag 1 autocorrelation
NoiseID	Identify the power law noise type using the B1 ratio

- ## Drift Functions

Drift Functions	
CalcQuadraticDrift	Calculate least-squares quadratic fit to phase data
CalcFirstDiff	Calculate average of 1st differences of frequency data
CalcSecondDiff	Calculate average of 2nd differences of phase data
CalcThreePointDrift	Calculate frequency drift using first, middle and last phase data points
CalcLinFreqDrift	Calculate least-squares linear fit for frequency data
CalcBisection Drift	Calculate freq drift using averages of first & last halves of freq data
CalcLogFreqDrift	Calculate least-squares log fit for frequency data
CalcDiffusionFreqDrift	Calculate least-squares diffusion fit for frequency data
RemoveQuadraticDrift	Remove quadratic (frequency) drift from phase data
RemoveLinFreqDrift	Remove linear drift from frequency data
RemoveLogFreqDrift	Remove log drift from frequency data
RemoveDiffusionFreqDrift	Remove diffusion drift from frequency data

- ## Plot Scale Functions

Plot Scale Functions	
FindPlotScale	Find scale factor for plotting phase or freq data
BasScale	Find scale factor for plotting phase or freq data using Bas method
RoundAxes	Round scale axis values

- ## Conversion Functions

Conversion Functions	
ConvFreqToPhase	Convert from frequency data to phase data
ConvPhaseToFreq	Convert from phase data to frequency data
ConvPhaseToFreqUsingTimetags	Convert from phase to frequency data using timetags as measurement interval
CalcDomain	Calculate the time and frequency domain noise parameter h(α)
ConvDomain	Convert between time and frequency domains
AddPSD	Add PSD values for a domain conversion
AddSigma	Add sigma values for a domain conversion

- ## Power Spectrum Functions

Power Spectrum Functions	
SpectrumCalc	Calculate a periodogram power spectral density
MultitaperSpectrumCalc	Calculate a multitaper power spectral density

- ## Common Statistical Functions

Common Statistical Functions	
FindMinMax	Find min and max of phase or frequency data
FindMedian	Find median of phase or frequency data
MedDev	Calculate the median deviation of phase or frequency data
CalcMean	Calculate average of phase or frequency data
CalcFreqOffset	Calculate the frequency offset of phase data
CalcPhaseStdDev	Calculate the standard deviation for phase data
CalcFreqStdDev	Calculate the standard deviation for frequency data
CalcNormalProb	Calculate the value of the normal deviate
CalcChiSqrProb	Calculate the area under the chi squared distribution
CalcInvChiSqr	Calculate value of chi squared for certain # of degrees of freedom
HistoCalc	Calculate a histogram of phase or frequency data

- ## Data Inspection and Modification Functions

Data Inspection and Modification Functions	
ScaleData	Scale phase or frequency data by a+bx
NormalizeData	Normalize phase or freq data to zero mean
AveragePhaseData	Do averaging of phase data
AverageFreqData	Do averaging of frequency data
RemoveFreqOffset	Remove the frequency offset of phase data
CountGaps	Count # gaps in phase or frequency data
FillGaps	Fill double gaps in phase or frequency data
FillFloatGaps	Fill float gaps in phase or frequency data
CountEqualTimetags	Count the number of equal timetags for phase or frequency data
FindFreqOutliers	Find, and optionally remove, outliers in frequency data

- ## Date Functions

Date Functions	
DateToMJD	Convert calendar date to Modified Julian Date
MJDToDate	Convert Modified Julian Date to calendar date
MJDToDOY	Convert Modified Julian Date to Day of Year
MJDToGPS	Convert Modified Julian Date to GPS Day #
DateConv	Convert month & year to Day of Week, DOY, MJD, and find # days in month
DayOfWeek	Find Day of Week for a calendar date
DateToJulian	Convert a calendar date to Julian date format
JulianToDate	Convert a Julian date to calendar date format
GetMJD	Get current Modified Julian Date from system clock
MakeDate	Make Gregorian date (yyyymmdd) from month, day and year
BreakDate	Break Gregorian date (yyyymmdd) into month, day and year

- **EDF Functions**

EDF Functions	
EDF	Calc estimated # of degrees of freedom for the modified Allan variance
HadamardEDF	Calc estimated # of degrees of freedom for the Hadamard variance
ModTotvarEDF	Calc estimated # of degrees of freedom for the modified total variance
HadTotvarEDF	Calc estimated # of degrees of freedom for the total Hadamard variance
TotvarEDF	Calc estimated # of degrees of freedom for the total variance
Thêo1EDF	Calc estimated # of degrees of freedom for Thêo1
CombinedEDF	Calc edf for the normal, overlapping and modified Allan and Hadamard variances

- **Bias Functions**

Bias Functions	
CalcBias	Calculate the value of the B1 bias function for zero deadtime & integer mu
CalcBias1	Calculate the value of the B1 bias function
CalcBias2	Calculate the value of the B2 bias function
CalcBias3	Calculate the value of the B3 bias function
CalcStarB1	Calculate the value of the *B1 bias function
Calc HadamardB1	Calculate the value of the Hadamard B1 bias function
TotvarBias	Calculate the value of the total variance bias
ModTotvarBias	Calculate the value of the modified total variance bias
HadTotvarBias	Calculate the value of the Hadamard total variance bias
Thêo1Bias	Calculate the value of the Thêo1 bias
Thêo1BiasToAlpha	Convert Thêo1 bias to power law noise exponent alpha
CalcRatio	Calculate the value of the R(n) function

- **Special Functions**

Special Functions	
CalcDegFree	Calculate # degrees of freedom for overlapping frequency data
CheckFrequenC	Check FrequenC version #

- **Documentation**

The FrequenC functions are documented with forms that describe their purpose, arguments, and usage as shown in the following example. Purchase of the FrequenC Library source code includes a complete User Manual with source code listings and details about the library implementation and usage.

The FrequenC Library

NAME:	**FUNCTION:**
AutocorrelationCalc	Calculate autocorrelation function of phase or frequency data

SYNOPSIS: `int AutocorrelationCalc(F_TYPE fZ[], float a[], int nLags, int nAF, BOOL bType, BOOL bMethod, BOOL bZero)`

F_TYPE fZ[]	Phase or frequency time domain data array:
	Z[0] = # data points
	Z[1] = analysis start point
	Z[2] = analysis end point

float a[]	Autocorrelation results (preallocated array)
int nLags	# lags to perform calc for <= # analysis points
int nAF	Averaging factor
BOOL bType	Input data type (0=phase, 1=freq)
BOOL bMethod	Calculation type (0=direct, 1=FFT)
BOOL bZero	Lag zero flag for direct method (0=skip, 1=do)
RETURN: int	The # of autocorrelation points, or -1 if error

REMARKS:

Averaging factor must be >0 and ≤ # data points.
Any gaps in data are temporarily filled during calculation.
Output array must be preallocated to # analysis points

EXAMPLE:

```
#include "frequenc.h"                              /* FrequenC header file */
F_TYPE y[512+ARRAY_OFFSET-1];                      /* frequency data array */
float a[512+ARRAY_OFFSET-1];              /* autocorrelation results array */
int ret;                                          /* function return value */
.
.
ret=AutocorrelationCalc(y, a, 1, 512, 0, 0, 0,);            /* calc ACF */
if(ret==-1)                                           /* check for error */
{
    printf("\nError");                                  /* error message */
}
else
{
    printf("\n# ACF Data Points = %d", num);        /* display # ACF pts */
}
```

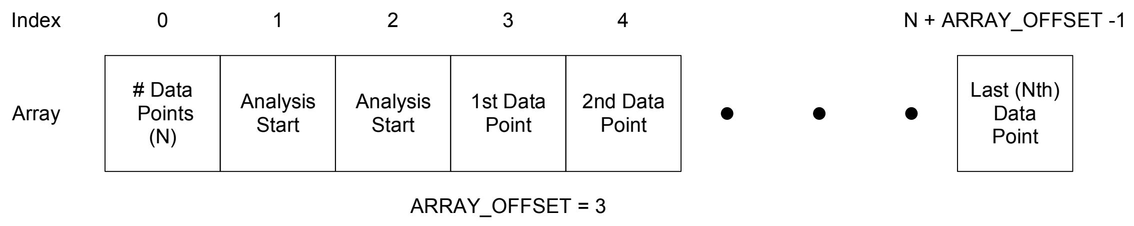
SEE ALSO: ACFNoiseID()

REFERENCE: D.B. Percival and A.T. Walden, *Spectral Analysis for Physical Applications*, Cambridge University Press, 1993, ISBN 052143541.

● Data Format

Many of the The FrequenC Library functions use an one-dimensional array format for phase or frequency data where the first three elements hold the number of data points, and the start and end analysis limits. The data are assumed to represent equally-spaced points, with gaps denoted by the value zero. All zeros are considered gaps in frequency data, while only embedded zeros are considered gaps in phase data. The sizes of the data arrays are determined by the calling program. The same data array format is also used to store timetags.

The FrequenC phase, frequency, and timetag double data arrays contain "headers" that hold the number of data points and the (1-based) start and end analysis limits in their first three array elements (0-based indices 0, 1 and 2):



● Availability

The FrequenC Library, its functions, and their documentation are available for purchase by special arrangement with Hamilton Technical Services.

The FrequenC dynamic link library, FrequenC.dll, is distributed with the Stable32 software package for frequency stability analysis. Using it with other applications requires the FrequenC.lib import library and FrequenC.h header files, along with documentation describing the various functions and their usage. Please contact Hamilton Technical Services about purchasing those items.

The FrequencyC Library source files may also be available under licensing and royalty agreements with Hamilton Technical Services. Again, please contact Hamilton Technical Services for further information.

● FrequenC License

A license to use the FrequenC Library with Stable32 is included with that software package.

Using the FrequenC library with other applications requires another license that is included with the purchase of the FrequenC documentation, import library and header files. No additional run time royalty payment is required for its personal use within the immediate user group that purchased it. Use of the FrequenC Library for commercial purposes requires a mutually beneficial royalty agreement with Hamilton Technical Services.

Licensing and royalty agreements can also be negotiated for use of the FrequenC source code

● Other Licenses

No other licenses are required to use the FrequenC.dll with another application. However, portions of the FrequenC Library source code contain utilize code adapted from other sources, and the user is required to obtain licenses from those vendors, as necessary, before utilizing it. Those vendors and products are as follows:

IPC-TC-006 Science and Engineering ToolsQuinn-Curtis, Inc.
18 Hearthstone Drive
Medfield, MA 02052
508-359-6639
www.quinn-curtis.com

Numerical Recipes in C
Cambridge University Press
40 West 20th Street
New York, NY 10011
www.nr.com

File: The FrequenC Library.doc
W.J. Riley
Hamilton technical Services
01/24/07