PAUL SCHERRER INSTITUT

PSI Bericht Nr. 01-07
August 2001
ISSN 1019-0643

Department Logistic and Marketing

# A Novel Auto-Correlation Function Method and FORTRAN Codes for the Determination of the Decay Ratio in BWR Stability Analysis

K. Behringer



Paul Scherrer Institut
CH - 5232 Villigen PSI
Telefon 056 310 21 11
Telefax 056 310 21 99

PLEASE BE AWARE THAT
ALL OF THE MISSING PAGES IN THIS DOCUMENT
WERE ORIGINALLY BLANK

PSI Bericht Nr. 01-07
August 2001
ISSN 1019-0643

# A Novel Auto-Correlation Function Method and FORTRAN Codes for the Determination of the Decay Ratio in BWR Stability Analysis

K. Behringer (*)

Paul Scherrer Institut
Department Logistic and Marketing

CH-5232 Villigen PSI

(*) Guest Scientist at PSI

## ABSTRACT

A novel auto-correlation function (ACF) method has been investigated for determining the oscillation frequency and the decay ratio in BWR stability analyses. The report describes not only the method but also documents comprehensively the used and developed FORTRAN codes. The neutron signals are band-pass filtered to separate the oscillation peak in the power spectral density (PSD) from background. Two linear second-order oscillation models are considered. The ACF of each model, corrected for signal filtering and with the inclusion of a background term under the peak in the PSD, is then least-squares fitted to the ACF estimated on the previously filtered neutron signals, in order to determine the oscillation frequency and the decay ratio. The procedures of filtering and ACF estimation use fast Fourier transform techniques with signal segmentation. Gliding 'short-time' ACF estimates along a signal record allow the evaluation of uncertainties. Some numerical results are given which have been obtained from neutron signal data offered by the recent Forsmark I and Forsmark II NEA benchmark project. They are compared with those from other benchmark participants using different other analysis methods.

**ZUSAMMENFASSUNG**

Eine neuartige Auto-Correlationsfunktion (ACF) Methode für die Bestimmung der Oscillationsfrequenz und des Abklingverhältnisses in BWR Stabilitätsanalysen wurde untersucht. Der Bericht beschreibt nicht nur die Methode, sondern dokumentiert ausführlich auch die FORTRAN Programme, welche benützt und entwickelt wurden. Die Neutronensignale werden mit einem Bandpass gefiltert, um den Oscillationspeak in der spektralen Leistungsdichte (PSD) vom Untergrund zu separieren. Zwei lineare Oscillatormodelle zweiter Ordnung werden betrachtet. Die ACF von jedem Modell, welches inbezug auf die Signalfilterung korregiert wird und einen Untergrundterm unter dem Peak in der PSD berücksichtigt, wird mit einem Least-Squares Fit an die ACF angepasst, welche an dem zuvor gefilterten Neutronensignal geschätzt wurde, um die Oscillationsfrequenz und das Abklingverhältnis zu bestimmen. Die Verfahren der Signalfilterung und der ACF Schätzung benützen die schnelle Fourier Transformation mit Signalsegmentierung. Gleitende "Kurzzeit" ACF Schätzungen entlang einer Signalaufzeichnung erlauben die Evaluierung von Unsicherheiten. Es werden einige numerische Resultate gegeben, welche an Neutronensignaldaten des kürzlichen Forsmark I und Forsmark II NEA Benchmarkprojektes erhalten wurden. Sie werden mit jenen von anderen Teilnehmern am Benchmarkprojekt, welche verschiedene andere Analysenmethoden benützten, verglichen.

**TABLE OF CONTENTS**

# 1. INTRODUCTION AND METHODOLOGY

In the stability analysis of boiling water reactors (BWRs), most experimental methods use parametric modelling of the neutron signals. The auto-correlation function (ACF) method is non-parametric, but requires the relationship to a theoretical model for interpretation. The classical ACF method is based on the linear second-order damped oscillator model. If it is applied to the digital neutron signal data offered by the recent NEA benchmark project on BWR stability analysis (Conde et al., 1999), the determination of the oscillation frequency and the decay ratio from the first few minima and maxima in ACF estimates failed. It required the investigation of a more sophistical ACF method. The reason is a background problem. The signal data must be band-pass filtered to separate the interesting peak in the power spectral density (PSD). The choice of the filter cutoff frequencies requires at first a PSD estimation. The selected frequency window should not be too small, because filtering involves a loss of information, also of that which is useful. Furthermore, a background under the peak in the PSD must be taken into account. Signal filtering and the inclusion of the PSD background term require a correction of the assumed model ACF. The corrected model ACF is then least-squares fitted to the ACF estimated on the previously filtered signal data. The fitting procedure is a relatively complicate one and contains a variable lag time range (fit range) parameter. Special techniques have been applied to the filtering procedure and the ACF estimation. They are both based on signal segmentation and the use of the fast Fourier transform (FFT). A FFT filter has been developed which approximates the shape of an ideal rectangular band-pass filter. The ACF estimation uses the indirect method with zero-padding. A modification of this method allows a ACF estimation over an arbitrary number of subsequent signal segments. If this segment number is small, then 'short-time' ACFs are obtained. A gliding segment analysis (GLSA) has been introduced. 'Short-time' ACF estimates move over the signal length, each forward shifted by one segment. The least-squares fitting is then repetetively applied to each ACF estimate. Average values of the oscillation frequency and the decay ratio and their standard deviations can be calculated for each specified fit range. A criterion has been found which allows the determination of the optimum fit range. The GLSA permits an uncertainty analysis in the estimate of each fit parameter and other data derived from them.

Two oscillator models have been considered. The report gives the necessary mathematical background. But the main aim of this report concerns the documentation of the used and developed FORTRAN codes for further improvements and refinements of the investigated analysis method. All codes are home-made. As far as possible, each code description is kept independent. Only the explanation of the variable notations used in the theoretical models, is not repeated.

The mathematical background to each code description had to be restricted to the necessary minimum of information with references to articles, papers and textbooks where details can be found. Copies of cited internal EIR or PSI reports are normally available from the PSI library on request.

The report includes also numerical results obtained from the analysis of several benchmark records. For comparison, corresponding results from other benchmark participants who apply different other methods, are given.

### 1.1 Reference

Conde J.M., Recio M., Verdu′ G., Ginestar D., Munoz-Cobo J.L., Navarro J., Palomo M.J., Sartori E., Lansaker P. (1999). Proceedings of Forsmark 1 & 2 BWR Stability Benchmark Time Series Analysis Methods for Oscillations during BWR Operation. M&C Conference, Madrid 1999, Vol. 1, 513.

## 2. THEORETICAL CONSIDERATIONS AND MODELS

The dynamic behaviour of a BWR is nonlinear. Following the summery given in a recent paper by Hennig (1999), the feedbacks of the thermal-hydraulic variables (volumetric steam content, fuel temperature etc.) to the neutron density lead to additional nonlinear terms in the describing system of equations, because one of the hydraulic balance equation is itself nonlinear. In particular, the nonlinear feedbacks can induce oscillatory solutions under certain system parameter combinations. Such phenomena can appear as global or regional decaying oscillations. For characterizing these oscillations the decay ratio (DR) is normally used. The auto-regressive moving-average (ARMA) modelling of measured neutron signals is a linear parametric approach. The optimal model order can be obtained from the application of an extended Akaike criterion which was originally developed for auto-regressive (AR) modelling. Under stable operating conditions, all poles of the transfer function must lie within the unit cycle in the z-transform. For the determination of the DR one must not presume that the system response can be approached by simple periodic functions. It is only necessary that the pulse response of the system is an oscillatory function decaying in time. This is described in the frequency domain by a broken rational transfer function. The DR is determined from the smallest decay constant. Near to a stable fixed state point in the system equations, nonlinear systems behave similarly as linear ones (Guggenheimer and Holmes, 1984), and the DR can principally be used for characterizing the stability properties.

The auto-correlation function (ACF) method is based on the approach of extracting the revelant informations from a simple periodic function. Power spectral density (PSD) estimations on the benchmark records showed mostly distortions around the main peak, in particular on the left-hand side. Such distortions appear then also in ACF estimations and make impossible to read the DR simply from the first few minima and maxima. The analysis method used here, is based on signal filtering and taking a PSD background term into account. Two oscillator models are represented which have been applied to least-squares fitting the model ACF to the experimental ACF estimated on previously filtered signal data.

The oscillation frequency in all investigated benchmark data is near to 0.5 Hz. The digital benchmark signal data have been sampled with 25 Hz and decimated to an effective sampling frequency of 12.5 Hz. Hence, the Nyquist cutoff frequency is 6.25 Hz.

## 2.1 Model A : Linear Second-Order Damped Oscillator

The application of this model to BWR stability analysis is as old as the BWR since its development period (Thie, 1981). It has been used in the frequency domain by PSD estimations in a rather engineering sense. The transfer function is given by

$$H(\omega) = \frac{1}{\omega_0^2 - \omega^2 + 2i\lambda\omega} \tag{2.1.1}$$

The frequencies $\omega$ and $\omega_0$ are denoted in angular units. $\omega_0$ is the undamped resonance frequency, and $\lambda$ (>0) is the damping constant which appears in the model ACF as decay constant. $H(\omega)$ shows resonance behaviour with the oscillation frequency $\omega_c$ , if

$$\omega_c^2 = \omega_0^2 - \lambda^2 > 0 \tag{2.1.2}$$

If one assumes that the oscillator is driven by random forces which are white over the resonance frequency region, the ideal PSD follows from

$$S_{xx}(\omega) = \frac{A_0}{(\omega^2 - \omega_0^2)^2 + 4\lambda^2\omega^2} \tag{2.1.3}$$

Where $A_0$ is a constant and represents the PSD of the driving forces. However, the peak maximum appears at a third characteristic frequency. The peak resonance frequency $\omega_r$ is given by

$$\omega_r^2 = \omega_0^2 - 2\lambda^2 > 0 \tag{2.1.4}$$

The existence of a real $\omega_r$ requires a stronger condition for resonance behaviour than given by equation (2.1.2).

One can calculate the half-power bandwidth $\Delta\omega$ of $S_{xx}(\omega)$. If $\lambda^2 << \omega_0^2$ , $\Delta\omega$ is given by $\Delta\omega \approx 2\lambda$. However, for small values of $\lambda$ , the peak is biased in the PSD estimate. In this case, one can only very subjectively estimate the magnitude of $\lambda$.

The ideal ACF, $R_{xx}^{(I)}(\tau)$ follows from equation (2.1.3) by the inverse Fourier transform.

$$R_{xx}^{(I)}(\tau) = C_0 e^{-\lambda|\tau|}\left[\cos(\omega_c\tau) + \frac{\lambda}{\omega_c}\sin(\omega_c|\tau|)\right] \tag{2.1.5}$$

where $\tau$ is the lag time and $C_0$ is a constant. There is the relationship between $A_0$ and $C_0$:

$$A_0 = 4\lambda\omega_0^2 C_0 \tag{2.1.6}$$

The decay ratio is defined by

$$DR = e^{-2\pi\lambda/\omega_c} \tag{2.1.7}$$

If the signal data are filtered with an ideal rectangular bandpass filter of unity gain, which has the lower cutoff frequency $\omega_L$ and the upper cutoff frequency $\omega_H$ under the condition $\omega_L<\omega_r<\omega_H$ , the ACF to be corrected for signal filtering, $R_{xx}^{(C)}(\tau)$, follows from

$$R_{xx}^{(C)}(\tau) = \frac{1}{\pi}\int_{\omega_L}^{\omega_H} d\omega S_{xx}(\omega)\cos(\omega\tau) \tag{2.1.8}$$

The integral must be solved numerically. An attempt of an analytical integration leads to exponential integral functions (or associated functions respectively). The resulting formulae did not invite for programming. Since for high DR cases, $S_{xx}(\omega)$ peaks strongly at $\omega_r$ , numerical integration routines cannot solve the problem. One has to rewrite equation (2.1.8) as

$$R_{xx}^{(C)}(\tau) = R_{xx}^{(I)}(\tau) - \frac{1}{\pi}\left(\int_0^{\omega_L} + \int_{\omega_H}^{\infty}\right) d\omega S_{xx}(\omega)\cos(\omega\tau) \tag{2.1.9}$$

The integral terms represent the correction to be made on the ideal ACF.

For establishing the fit ACF, $R_{xx}^{(F)}(\tau)$, a background ACF, $B(\tau)$, is added.

$$R_{xx}^{(F)}(\tau) = R_{xx}^{(C)}(\tau) + B(\tau) \tag{2.1.10}$$

where

$$B(\tau) = \frac{1}{\pi}\int_{\omega_L}^{\omega_H} d\omega B(\omega)\cos(\omega\tau) \tag{2.1.11}$$

Most estimated PSDs showed an exponentially decreasing background in the frequency region around the peak, suggesting for $B(\omega)$ the form :

$$B(\omega) = B_0 e^{-\alpha(\omega-\omega_L)}; \omega_L \le \omega \le \omega_H, \alpha > 0 \tag{2.1.12}$$

$B_0$ and $\alpha$ are constants. One obtains then for $B(\tau)$

$$B(\tau) = \frac{B_0}{\pi}\left\{\frac{\alpha}{\alpha^2+\tau^2}\left[\cos(\omega_L\tau) - e^{-\alpha(\omega_H-\omega_L)}\cos(\omega_H\tau)\right]\right.$$

$$\left. - \frac{\tau}{\alpha^2+\tau^2}\left[\sin(\omega_L\tau) - e^{-\alpha(\omega_H-\alpha_L)}\sin(\omega_H\tau)\right]\right\}; \tau>0 \tag{2.1.13}$$

$$= \frac{B_0}{\pi\alpha}\left(e^{-\alpha\omega_L} - e^{-\alpha\omega_H}\right); \tau=0 \tag{2.1.14}$$

The case $\alpha = 0$ (constant background under the peak in the PSD) gives physically no sense. It leads only to a peaking of $R_{xx}^{(F)}(\tau)$ at $\tau = 0$, and approaches to the Dirac-Delta function for $\omega_L \rightarrow 0$ and $\omega_H \rightarrow \infty$.

$R_{xx}^{(F)}(\tau)$ is the model ACF which has to be least-squares fitted to the experimental ACF resulting from previously filtered signal data. The digital band-pass filter must be very steep at the corner frequencies. Its shape must well correspond to the assumptions made for the fit ACF. Natural fit parameters are $C_0$, $\omega_c$, $\lambda$, $B_0$ and $\alpha$. From the fit data of $\omega_c$ and $\lambda$ one obtains then an estimate of the DR.

## 2.2 Model B : Modified Second-Order Oscillator

The appearance of three characteristic frequencies in model A is mathematically correct under the assumption of driving forces which have white noise characteristics (Bendat and Piersol, 1971). It happened many times in cases with small DR values that imaginary values of $\omega_r$ occurred in the fit procedure. Model B avoids this problem by considering a slightly modified version of an active second-order resonance filter given in the textbook of Moschytz and Horn (1983). The transfer function reads

$$H(\omega) = \frac{1}{\frac{1}{Q_R} + \frac{i\omega}{\omega_0} + \frac{\omega_0}{i\omega}} \tag{2.2.1}$$

$\omega_0$ is the resonance frequency, and $Q_R$ is the resonance quality factor. The filter shows resonance behaviour if $Q_R > 0.5$ . Again, if one assumes that the driving forces are random and white in the resonance frequency region, the ideal PSD is given by

$$S_{xx}(\omega) = A_0 \frac{\omega_0^2 \omega^2}{(\omega^2 - \omega_0^2)^2 + 4\lambda^2 \omega^2} \tag{2.2.2}$$

where one can identify

$$\lambda = \frac{\omega_0}{2Q_R} \tag{2.2.3}$$

Note that the constant $A_0$ has a different scale against model A.

$$A_{0(\text{model B})} \Leftrightarrow A_{0(\text{model A})} / \omega_0^4$$

A remarkable property of this PSD is that the peak amplitude appears exactly at $\omega_0$ and is not affected by $Q_R$ . Hence, $\omega_r = \omega_0$ in this model. The half-power frequency points $\omega_1$ , $\omega_2$ are given by

$$\omega_{1,2}^2 = \frac{\omega_0^2}{2Q_R}(1 + 2Q_R^2 \mp \sqrt{1 + 4Q_R^2}) \tag{2.2.5}$$

from which one obtains the half-power bandwidth $\Delta\omega$ to a very good approximation (for $Q_R > 2$) by

$$\Delta\omega = \omega_2 - \omega_1 \cong \frac{\omega_0}{Q_R} = 2\lambda \tag{2.2.6}$$

Note that $S_{xx}(\omega = 0) = 0$ in contrast to model A.

The transfer functions of model A and model B have the same (conjugate complex) poles. The difference is that in model A $H(\omega)$ corresponds to a AR(2) model, while in model B $H(\omega)$ is related to a ARMA(2,1) model (in the backward difference approximation by the z-transform).

The ideal ACF follows from Fourier transforming equation (2.2.2) and reads

$$R_{xx}^{(I)}(\tau) = C_0 e^{-\lambda|\tau|}\left[\cos(\omega_c\tau) - \frac{\lambda}{\omega_c}\sin(\omega_c|\tau|)\right] \quad (2.2.7)$$

$\omega_c$ has the same expression as given by equation (2.1.2). Equation (2.2.7) differs from equation (2.1.5) only by the negative sign of the second term. With respect to the minima and maxima of $R_{xx}^{(I)}(\tau)$, regarding only the right-hand sided part for $\tau > 0$, the extrema appear in model A exactly at multiples of $\pi/\omega_c$ . In model B, they range at down-shifted values of the lag time, and are given by

$$\tau_n^{(min)} = \frac{2(n-1)\pi + \eta}{\omega_c}; n = 1,... \quad (2.2.8)$$

$$\tau_n^{(max)} = \frac{(2n-1)\pi + \eta}{\omega_c}; n = 1,... \quad (2.2.9)$$

where

$$\eta = 2\arccos\frac{1}{2Q_R} \quad (2.2.10)$$

The values of $\tau_n^{(min)}$ and $\tau_n^{(max)}$ do not deviate very much from those of model A if DR> 0.2, and suggest to use the same definition of the DR as given by equation (2.1.7). $Q_R$ is then related to the DR by

$$Q_R = \frac{1}{2}\sqrt{1 + (\frac{2\pi}{\ln DR})^2} \quad (2.2.11)$$

In order to show a few numerical data of these lag time deviations, lag time shift factors can be introduced, defined by

$$r_n^{(min)} = \tau_{n(model-B)}^{(min)} / \tau_{n(model-A)}^{(min)} = \frac{2(n-1)\pi + \eta}{(2n-1)\pi} \quad (2.2.12)$$

$$r_n^{(max)} = \tau_{n(model-B)}^{(max)} / \tau_{n(model-A)}^{(max)} = \frac{(2n-1)\pi + \eta}{2n\pi} \quad (2.2.13)$$

Data for the worst case (n=1) are listed in Table 2.1.

**Table 2.1:** $r_1^{(min)}$ and $r_1^{(max)}$ as Function of the DR

| DR | $r_1^{(min)}$ | $r_1^{(max)}$ |
|---|---|---|
| 0.99 | 0.9990 | 0.9905 |
| 0.9 | 0.9893 | 0.9947 |
| 0.6 | 0.9484 | 0.9742 |
| 0.4 | 0.9078 | 0.9539 |
| 0.2 | 0.8404 | 0.9202 |

All further considerations with respect to the corrected ACF and the fit ACF follow from model A. In equation (2.1.9) one has to replace $R_{xx}^{(I)}(\tau)$ by equation (2.2.7) and $S_{xx}(\omega)$ by equation (2.2.2).

## 2.3 Gradient Functions of the Fit ACF

The least-squares fit codes ACFIT6 and ACFIT7, described in Section 10, require gradient functions of the fit ACF for $\tau \geq 0$ with respect to the fit parameters $C_0$ , $\omega_c$ , $\lambda$ , $B_0$ and $\alpha$ . For completeness, a listing of these functions will be given.

The following three derivative functions refer to both models :

$$C_0 \frac{\partial R_{xx}^{(F)}(\tau)}{\partial C_0} = R_{xx}^{(C)}(\tau) \tag{2.3.1}$$

$$B_0 \frac{\partial R_{xx}^{(F)}(\tau)}{\partial B_0} = B(\tau) \tag{2.3.2}$$

$$\frac{\partial R_{xx}^{(F)}(\tau)}{\partial \alpha} = \frac{\partial B(\tau)}{\partial \alpha} \tag{2.3.3}$$

$$\frac{\partial B(\tau)}{\partial \alpha} = \frac{B_0}{\pi}\left\{ \frac{\tau^2 - \alpha^2}{(\alpha^2 + \tau^2)^2}\left[\cos(\omega_L \tau) - e^{-\alpha(\omega_H - \omega_L)}\cos(\omega_H \tau)\right]\right.$$

$$+\frac{2\alpha\tau}{(\alpha^2 + \tau^2)^2}\left[\sin(\omega_L \tau) - e^{-\alpha(\omega_H - \omega_L)}\sin(\omega_H \tau)\right]$$

$$\left. + \frac{\omega_H - \omega_L}{\alpha^2 + \tau^2} e^{-\alpha(\omega_H - \omega_L)}\left[\alpha\cos(\omega_H \tau) - \tau\sin(\omega_H \tau)\right] \right\}; \tau, \alpha > 0 \tag{2.3.4}$$

For $\tau$=0, equation (2.3.4) reduces to

$$\left(\frac{\partial B(\tau)}{\partial\alpha}\right)_{\tau=0} = \frac{B_0}{\pi}\left\{-\frac{1}{\alpha^2}\left[1-e^{-\alpha(\omega_H-\omega_L)}\right]+\frac{\omega_H-\omega_L}{\alpha}e^{-\alpha(\omega_H-\omega_L)}\right\} \quad (2.3.5)$$

and for $\tau$=0 and $\alpha\to 0$

$$\left(\frac{\partial B(\tau)}{\partial\alpha}\right)_{\tau=0,\alpha\to 0} = -\frac{B_0}{\pi}(\omega_H-\omega_L)^2 \quad (2.3.6)$$

The derivative functions with respect to $\omega_c$ and $\lambda$ are different between the models. Generally, one can write

$$\frac{\partial R_{xx}^{(F)}(\tau)}{\partial\omega_c} = \frac{\partial R_{xx}^{(C)}(\tau)}{\partial\omega_c} = \frac{\partial R_{xx}^{(I)}(\tau)}{\partial\omega_c} - \frac{\partial Q(\tau)}{\partial\omega_c} \quad (2.3.7)$$

$$\frac{\partial R_{xx}^{(F)}(\tau)}{\partial\lambda} = \frac{\partial R_{xx}^{(C)}(\tau)}{\partial\lambda} = \frac{\partial R_{xx}^{(I)}(\tau)}{\partial\lambda} - \frac{\partial Q(\tau)}{\partial\lambda} \quad (2.3.8)$$

Model A :

$$R_{xx}^{(I)}(\tau) = C_0 e^{-\lambda\tau}\left[\cos(\omega_c\tau)+\frac{\lambda}{\omega_c}\sin(\omega_c\tau)\right] \quad (2.3.9)$$

$$Q(\tau) = \frac{4\lambda\omega_0^2 C_0}{\pi}\left(\int_0^{\omega_L} + \int_{\omega_H}^{\infty}\right)d\omega\frac{\cos(\omega\tau)}{(\omega^2-\omega_0^2)^2+4\lambda^2\omega^2} \quad (2.3.10)$$

where $\omega_0^2 = \omega_c^2 + \lambda^2$

$$\frac{\partial R_{xx}^{(I)}(\tau)}{\partial\omega_c} = C_0 e^{-\lambda\tau}\left[\frac{\lambda\tau}{\omega_c}\cos(\omega_c\tau) - (\tau+\frac{\lambda}{\omega_c^2})\sin(\omega_c\tau)\right] \quad (2.3.11)$$

$$\frac{\partial Q(\tau)}{\partial\omega_c} = \frac{8\lambda\omega_c C_0}{\pi}\left\{\left(\int_0^{\omega_L} + \int_{\omega_H}^{\infty}\right)d\omega\frac{\cos(\omega\tau)}{(\omega^2-\omega_0^2)^2+4\lambda^2\omega^2}\right.$$

$$+2\omega_0^2\left(\int_0^{\omega_L}+\int_{\omega_H}^{\infty}\right)d\omega\frac{(\omega^2-\omega_0^2)\cos(\omega\tau)}{\left[(\omega^2-\omega_0^2)^2+4\lambda^2\omega^2\right]^2}\ \Big\} \tag{2.3.12}$$

$$\frac{\partial R_{xx}^{(I)}(\tau)}{\partial\lambda}=-C_0e^{-\lambda\tau}\left[\tau\cos(\omega_c\tau)+\frac{\lambda\tau-1}{\omega_c}\sin(\omega_c\tau)\right] \tag{2.3.13}$$

$$\frac{\partial Q(\tau)}{\partial\lambda}=\frac{4C_0}{\pi}\Big\{(\omega_0^2+2\lambda^2)\left(\int_0^{\omega_L}+\int_{\omega_H}^{\infty}\right)d\omega\frac{\cos(\omega\tau)}{(\omega^2-\omega_0^2)^2+4\lambda^2\omega^2}$$

$$-4\lambda^2\omega_0^2\left(\int_0^{\omega_L}+\int_{\omega_H}^{\infty}\right)d\omega\frac{(\omega^2+\omega_0^2)\cos(\omega\tau)}{\left[(\omega^2-\omega_0^2)^2+4\lambda^2\omega^2\right]^2}\ \Big\} \tag{2.3.14}$$

Model B :

$$R_{xx}^{(I)}(\tau)=C_0e^{-\lambda\tau}\left[\cos(\omega_c\tau)-\frac{\lambda}{\omega_c}\sin(\omega_c\tau)\right] \tag{2.3.15}$$

$$Q(\tau)=\frac{4\lambda C_0}{\pi}\left(\int_0^{\omega_L}+\int_{\omega_H}^{\infty}\right)d\omega\frac{\omega^2\cos(\omega\tau)}{(\omega^2-\omega_0^2)+4\lambda^2\omega^2} \tag{2.3.16}$$

$$\frac{\partial R_{xx}^{(I)}(\tau)}{\partial\omega_c}=-C_0e^{-\lambda\tau}\left[\frac{\lambda\tau}{\omega_c}\cos(\omega_c\tau)+(\tau-\frac{\lambda}{\omega_c^2})\sin(\omega_c\tau)\right] \tag{2.3.17}$$

$$\frac{\partial Q(\tau)}{\partial\omega_c}=\frac{16\lambda\omega_0C_0}{\pi}\left(\int_0^{\omega_L}+\int_{\omega_H}^{\infty}\right)d\omega\frac{\omega^2(\omega^2-\omega_0^2)\cos(\omega\tau)}{\left[(\omega^2-\omega_0^2)^2+4\lambda^2\omega^2\right]^2} \tag{2.3.18}$$

$$\frac{\partial R_{xx}^{(I)}(\tau)}{\partial\lambda}=-C_0e^{-\lambda\tau}\left[\tau\cos(\omega_c\tau)+\frac{1-\lambda\tau}{\omega_c}\sin(\omega_c\tau)\right] \tag{2.3.19}$$

$$\frac{\partial Q(\tau)}{\partial\lambda}=\frac{4C_0}{\pi}\Big\{\left(\int_0^{\omega_L}+\int_{\omega_H}^{\infty}\right)d\omega\frac{\omega^2\cos(\omega\tau)}{(\omega^2-\omega_0^2)^2+4\lambda^2\omega^2}$$

$$-4\lambda^2\left(\int_0^{\omega_L}+\int_{\omega_H}^{\infty}\right)d\omega\frac{\omega^2(\omega^2+\omega_0^2)\cos(\omega\tau)}{\left[(\omega^2-\omega_0^2)^2+4\lambda^2\omega^2\right]^2}\ \} \qquad (2.3.20)$$

## 2.4 References

Bendat S.J.and Piersol A.G.(1971). Random Data : Analysis and Measurements Procedures, Wiley-Interscience, New York.

Guggenheimer J.and Holmes P.(1984). Nonlinear Oscillation, Dynamical Systems and Bifurcations in Vector Fields, Applied Mathematical Sciences 42, Springer Verlag.

Hennig D.(1999). Nucl.Technology, 126, 10.

Moschytz G.S.and Horn P.(1983). Handbuch zum Entwurf aktiver Filter, Oldenbourg.

Thie J.A.(1981). Power Reactor Noise, American Nuclear Society, La Grange Park, Illinois.

## 3. GENERAL PROGRAMMING FEATURES

All codes have been written in VAX FORTRAN-EXTENDED under the VMS operating system. They are fully compatible with the modern DEC compiler COMPAQ (FORTRAN 95). In normal precion, real variables are represented as REAL*4 (32 bits), in double precision as REAL*8 (64 bits). The COMPAQ compiler interprets automatically real variables in codes which are implicitly declared for double precision, in G_FLOATING structure. Integer variables have the precision of INTEGER*4 by default.

The codes have a lot of common features, but they are not like an unique package, because they have been developed at different time periods

All files used are of the sequential type. Real numbers on data input and data output files are principally formatted in the E- or F-specification. Output data from double precision codes are normally not accepted in D-specification by graphical codes. Real numbers on parameter data input files must be written according to the given code precision. The file denotation has the structure : file name (up to 9 characters).type (up to 3 characters);version number. The file name and the type can be extended by underscores. The name of files which are directly related to the code in question, is taken over from the program name. The type declaration follows generally the scheme :

.IN parameter data input file (for non-interactive codes)

.PRT file determined for printing with text

.DAT, .PLO data output file, the second declaration specially related for plotting.

All file OPEN statements contain the keyword DEFAULTFILE='DIRINPUT'. By a preceding ASSIGN command another subdirectory can be chosen where the required old files are stored and the new files should be created. For batch operation an ASSIGN command must be given in the SUBMIT command file.

Most codes are strongly modularly designed. Programming is based on flow sheets. In the main program, there is an unique style in the use of the label numbers. Labels of up to two digits are reserved for DO loops. Labels for branching have three digits. Format labels have four or exceptionally five digits, whereby the first or the first two digits refer to the file unit number. Labeled formats are put at the program begin after the declarations. Short formats are sometimes included in the READ or WRITE statements.

As far as possible and reasonable, the input parameter data are assured against rough mistakes and have internal range restrictions. These restrictions are given in the parameter data input list of each code description. As far as external library routines are required, they are exceptionless taken from IMSL.

The code listings have line numbers with leading zeros which have been inserted in the identification field (column 73-80).

# 4. PROGRAM CPSDES3

Precision : double

Operation : interactive

Required auxiliary routines : none

Purpose of the program :

Univariate and Bivariate Spectral Analysis of Stationary Noise Signals by the Welch Method.

Feature Summary :

- Direct estimation procedure by FFT-techniques of the
  - power spectral densities (PSDs) in channel A and channel B,
  - cross-power spectral density (CPSD),
  - transfer function (TF),
  - squared coherence function (SCF).
- Options for 3 signal window functions, selectable degree of segment overlap.
- Process alignment available for transport time analysis.
- Within a run, a practically unlimited number of signal pairs from different files, each case called a job, can be analyzed.

## 4.1 Mathematical Background

The realized estimation procedure of the PSDs and the CPSD refers to the direct method with FFT techniques. It is based on the Welch method (Welch, 1967) using signal windowing with overlapped segmentation. The principles of the method have been summarized in a report by Behringer (1988).

Any signal window (SW) function can be represented by a Fourier series. There is a category of SWs which are characterized by a small finite number of terms in the Fourier series expansion (Harris, 1978). Three types, which have strongly different properties, have been selected from this category of SWs and implemented in the code :

- the one-term uniform or rectangular SW,
- the two-term Hannig SW,

- the four-term –74 db Blackman-Harris SW. The coefficients used for this SW are values revised by Nuttall (1981).

The Hanning SW and the Blackman-Harris SW are implemented in the code by a convolution in the frequency domain.

The uniform SW gives the best frequency resolution, while the Blackman-Harris SW shows the worst frequency resolution (very smooth estimate). The SWs behave the other way around with regard to the highest side lobe level. The Hanning SW is a compromise between. It is believed that, with these 3 available SWs, the requirements might be covered for most practical spectral analysis cases. Data of the properties of these SWs are given in the mentioned report by Behringer (1988).

The rectangular SW does not require segment overlap. But a smooth weighting of signal data without overlap wastes available information. The procedure for obtaining the optimal fractional segment overlap consists in minimizing the variance of the PSD data estimation. With regard to practical applications, one is interested in selecting the fractional overlap somewhat below the optimal values, in order to save CPU time. A good choice is 50 % fractional overlap for the Hanning SW, and 60 % fractional overlap for the Blackman-Harris SW. It is to note that variance minimalization considerations must be restricted to the PSD estimation. They cannot simply be extended to the CPSD estimation, since in the variance of a CPSD estimate the coherence function appears, which is, in effect, an additional free parameter. It is usually assumed that the parameter data selected for a good PSD estimation should simultaneously give a good CPSD estimate.

In the estimation of the TF, channel A is considered as input and channel B as output. The code contains the option to represent the complex functions, CPSD and TF, by the real and imaginary parts, or by the magnitude and phase. The phase is defined in the interval $-180^0$ to $+180^0$ . With respect to transport time analysis, a routine is implemented which tries to calculate additionally a continued phase.

Transport time analysis of stochastic processes is an important but special field of noise analysis. It will not work in every case and sometimes requires tricky solutions. Usually, one has the simple case that the two detector signals x(t) and y(t), both assumed to be stationary, contain a common component c(t) which is delayed in the signal y(t) by the amount $\tau_c$ . This delay time is the quantity to be determined.

$$x(t) = c(t) + b_x(t)$$

$$y(t) = c(t - \tau_c) + b_y(t)$$

The components $b_x(t)$ and $b_y(t)$ are assumed to be uncorrelated broad-band background noise. c(t) is assumed to be the dominant component. In the estimation procedure of the CPSD there is the segment length T. If $\tau_c$ is larger than T, the transport phenomenon is lost and one cannot measure $\tau_c$ . If T cannot be adapted for any reason, there is a first requirement for a signal alignment, i.e. to compensate the delay of the signal component c(t) in y(t) by a corresponding time shift of the signal x(t). An exact alignment leads to the zero-phase line of the CPSD. If $\tau_c < T$, the SCF can be utilized for determining the frequency range over which the evaluation of a plot of the CPSD phase versus frequency may give a reliable value of $\tau_c$ . The SCF is a computational quantity following from the estimates of the PSDs and the CPSD. One can show that the estimated CPSD magnitude and the SCF will be biased. Hence, even if $\tau_c < T$, there is a second requirement for aligning the signals x(t) and y(t) in order to transfer bad estimations of the CPSD and SCF into good ones. The code CPSDES3 contains the possibility for process alignment, when, in a first trial, an approximative value of $\tau_c$ can be determined.

The code does not make use of FFT routines available from the IMSL. It is independent of any routine requirements from an external library. There are several tricky applications of FFT techniques in order to reach an efficient utilization. Two real signal records can be combined into a complex signal and their data in a segment can simultaneously be transformed into the frequency domain by one FFT. This method has not been applied, since in mixing the signals and in decomposing the Fourier coefficients, rounding errors could influence the results by cross-effects under unfavourable numerical conditions. The signal data are treated separately in each channel. If $N_s$ is the number of real signal data in a segment, which must be a power of 2, these data can be shuffled into a sample of $N_s/2$ complex data. The application of the FFT requires then only the transform size $N_T = N_s/2$ . The FFT routine implemented is a copy from the textbook of Brigham (1974), where also the shuffling procedure is described. The code was originally written in single precision. The present version, CPSDES3, represents an adaptation to double precision, which gives better PSD estimates over a larger amplitude range.

### 4.2 References

Behringer K. (1988). The Code CPSDES1 for Univariate and Bivariate Spectral Analysis by the Welch Method, Internal PSI Report TM-41-88-20.

Brigham E.O. (1974). The Fast Fourier Transform, Prentice-Hall Inc..

Harris F.J. (1978). Proc. IEEE 66, 51.

Nuttall A.H. (1981). IEEE Trans.Acoustics, Speech and Signal Processing ASSP-29, 84.

Welch P.D. (1967). IEEE Trans.Audio Electroacoustics 15, 20.

### 4.3 Files

There are 4 files :

- 'FINPUTA'

This file contains the signal data of channel A.

- 'FINPUTB'

This file contains the signal data of channel B. 'FINPUTA' and 'FINPUTB' are formal parameters. The file names must be interactively specified for each job. The code assumes that the signal data are of the REAL*4 type, are written as column vector, and have the same format in both channels within a run.

- CPSD.PRT

This file is meant for printing. It contains (with text) all input parameter data and messages from the computation progress. Furthermore, signal variance values are given, which are calculated from PSD integration. There are two values. The first one refers to the complete integration of the PSD spectrum from zero-frequency up to the Nyquist cutoff frequency. The second one leaves out the first and the last spectral points in the PSD integration. It is usually significantly smaller if a DC component is present in the signal. The presence of a DC component leads theoretically to a Dirac-Delta function of the PSD at zero-frequency. The PSD data point at zero-frequency is mostly of no interest. Optionally, a PSD average value over a given frequency interval can be output.

The data of the spectral functions can optionally be written on this file. For this reason, the set of the common input parameter data is completely rewritten for each job.

- CPSD0'''.PLO

This file is optionally opened and contains the data of the estimated spectral functions of a job. It serves for plotting by a graphical code. When this option has been chosen, then for each job a separate file with a current number of up to 3 digits is opened. The initial number for the first job must be interactively specified. There are 8 columns with the line format (1P,8E10.3). The data in these columns depend on the selected option for representing the complex spectral functions.

a) Representation of the complex spectral functions by the real and imaginary parts :

column

| | | |
|---|---|---|
| 1 | : | frequency in Hz |
| 2 | : | PSD of channel A |
| 3 | : | PSD of channel B |
| 4 | : | real part of the CPSD |
| 5 | : | imaginary part of the CPSD |
| 6 | : | real part of the TF |
| 7 | : | imaginary part of the TF |
| 8 | : | SCF |

b) Representation of the complex spectral functions by magnitude and phase :

column

| | | |
|---|---|---|
| 1 | : | frequency in Hz |
| 2 | : | PSD of channel A |
| 3 | : | PSD of channel B |
| 4 | : | magnitude of the CPSD |
| 5 | : | magnitude of the TF |
| 6 | : | phase of the CPSD and the TF respectively in degrees between $-180^0$ and $+180^0$ |
| 7 | : | continued phase in degrees |
| 8 | : | SCF |

## 4.4 Parameter Data Input

There is an elaborated interactive procedure for the input of the parameter data with text and format indications on the terminal screen. Protection is provided against typing errors. Each data typed in is immediately rewritten to the screen. It can be corrected if necessary, and must be verified for final acceptance. If in a number field an unallowed character is erroneously typed in, the code repeats the question. In a few cases, erroneous data are rejected by return to the question. There are common and individual parameter data. The common parameter data refer to all jobs to be treated within a run. The individual parameter data must be given for each job.

Common Parameter Data Input :

- format (A), RUN

A string of max. 40 characters for run identification.

- format (I4), NT

Number of signal data points in a segment (FFT transform size disregarding the shuffling procedure).

Internal restriction : The value must be a power of 2 within the range $2^5$ to $2^{13}$.

- format (I4), NAV

Number of averages (including segment overlap). For convenience, an exact number must not be given, if one wishes to scan over the available record length for each job. In this case, the maximum value (NAV=1000) can be given. The code checks on the EOF mark in each signal channel. It reduces NAV automatically to the maximum possible value (with completely filled segments) and continues the analysis with this new value, provided that this value is ≥2. The same procedure applies to a detected read-error. In both cases, messages are sent to the terminal and the print file, specifying the channel, the current average number, the relative data point number in the segment, and the absolute data point number (counted from where the record begins), and where the EOF mark or the read-error has been found.

Internal restrictions : IF(NAV.LT.1) NAV=1 (default value)

IF(NAV.GT.1000) NAV=1000

- format (I4), NRED

Number of data points to be overlapped. The fractional overlap is given by FLOAT(NRED)/FLOAT(NT).

Internal restrictions : IF(NRED.LT.0) NRED=0 (default value)

IF(NRED.GE.NT) NRED=NT-1

Example : If NT=256 and the Hanning SW is used, one should set NRED=128.

- format (I1), MODE

Parameter for selecting the SW.

MODE = 1 : uniform SW

2 : Hanning SW

3 : Blackman-Harris SW

Internal restrictions : IF(MODE.LT.1) MODE=1 (default value)

IF(MODE.GT.3) MODE=3

- format (D10.3), SFR

Sampling frequency in Hz.

Internal restriction : IF(SFR.LE.0.) question will be repeated.

- Representation of the complex spectral functions by magnitude and phase ?

If one wishes this type of representation, one has to answer with YES, otherwise the representation will be given by the real and imaginary parts.

- format (I1), IOUT

Parameter for directing the data output of the spectral functions to the print file and/or to the plot files :

IOUT = 0 : CPSD0'''.PLO

1 : CPSD.PRT

2 : CPSD0'''.PLO + CPSD.PRT

Internal restrictions : IF(IOUT.LT.0) IOUT=0 (default value)

IF(IOUT.GT.2) IOUT=2

- format (I3), IPLOT

Initial integer number of the file CPSD0'''.PLO for the first job. For each further job within a run this number is incremented by 1.

Internal restrictions : IF(IPLOT.LT.1) IPLOT=1 (default value)

IF(IPLOT.GT.990) IPLOT=990

If the run starts with the highest allowed number, the code assumes that the run does not encompasses more than 10 jobs, otherwise the run will normally terminate after job 10 with a preceding message to the terminal. The question for IPLOT is surpressed if IOUT=1. IPLOT allows a grouping of numbers of the plot files, if sets of signal pairs are to be analyzed by more than one run.

- format (A), IFORM

A string of max. 20 characters for specifying the single-read signal data format. The format must be given in E- or F-specification including the brackets. The code assumes the same data format in both channels and for all jobs.

Individual Parameter Data Input :

- format (A), FINPUTA

A string of max. 40 characters for the file name of the signal data of channel A. If after a third trial this file cannot be opened, the code stops with a message.

- format (A), FINPUTB

A string of max. 40 characters for the file name of the signal data of channel B. If after a third trial this file cannot be opened, the code stops with a message.

If the code is used only for PSD estimations with the same signal data as in channel A, one must before make a copy of the file of the signal data of channel A to an other file name (e.g. to the next higher file version number), and assign this other file name to FINPUTB.

- format (I7), NAS

This number specifies the first signal data point in channel A where analysis starts. If it is set > 1, then NAS-1 initial data points are passed over.

Default value NAS=1, press return !

- format (D10.3), XA0

DC value to be subtracted from the signal data in channel A.

Default value XA0=0., press return !

- format (D10.3), GA

Conversion factor for the signal data of channel A. Its use is explained later at channel B.

Internal restriction : IF(GA.LE.0.) GA=1.

Default value GA=1., press return !

- format (I7), NBS

As NAS, this number specifies the first signal data point in channel B where analysis starts.

Default value NBS=1, press return !

By a proper setting of the parameters NAS and NBS, one has the possibilities :

- to leave out an initial part of signal data without a prior deleting.
- To perform process alignment in transport time analysis. It is obvious that the accuracy of the alignment is limited by the sampling interval.

- format (D10.3), XB0

DC value to be subtracted from the signal data points in channel B.

Default value XB0=0., press return !

- format (D10.3), GB

Conversion factor for the signal data of channel B.

Internal restriction : IF(GB.LE.0.) GB=1.

Default value GB=0., press return !

The conversion factors $g_x$ and $g_y$ transform the signal data x(t) and y(t) into normalized time series by

$$x'(t) = (x(t) - x_0)/g_x$$

$$y'(t) = (y(t) - y_0)/g_y$$

where $x_0$ and $y_0$ are the DC values. Values for $g_x$ and $g_y$ can be used for unit conversion or for normalizing the signal amplitudes to the input of the amplifiers. However, they are not directly applied to the measured signal data, but to the final estimates of the spectral functions. An estimation of the DC components is not provided in the code. They should only be applied, if they are very dominant. Values can be obtained via the code ACCF1 (Section 7).

- format (I4), NFL

- format (I4), NFU

NFL and NFU are frequency numbers for specifying the closed interval [NFL,NFU] over which average PSD values in both channels are estimated. NFL is the lower frequency number, and NFU is the upper frequency number.

Internal restrictions : IF(NFL.LT.0) NFL=0

IF(NFU.GT.NT/2) NFU=NT/2

IF(NFL.GT.NFU) NFL=0, NFU=0

Default values : NFL=0, press return !

NFU=0, press return !

By using the default values, the calculation of the average PSD values is not carried out.

## 4.5 Branchings

After a job has terminated, there will be a question concerning continuing with the next job. This question must be acknowledged with YES or NO. If there is another job, the code then asks for the next individual input parameter data. A job which has been started, terminates, if the available average number NAV has been found to be less than 1. If IOUT≠1, the previously opened plot file is deleted.

# LISTING

## CPSDES3

```
      PROGRAM CPSDES3
C
C     SPECTRAL ANALYSIS OF TWO DIGITAL NOISE RECORDS.
C     ESTIMATION OF THE
C          POWER SPECTRAL DENSITY IN EACH CHANNEL,
C          CROSS-POWER SPECTRAL DENSITY,
C          TRANSFER FUNCTION (CHANNEL A = INPUT, CHANNEL B = OUTPUT),
C          SQUARED COHERENCE FUNCTION.
C
C     THE ESTIMATION PROCEDURE IS BASED ON THE WELCH METHOD USING
C     SIGNAL WINDOWING WITH OVERLAPPED SEGMENTATION.
C
C     THE SIGNAL DATA ARE ASSUMED TO BE FORMATTED IN SINGLE PRECISION.
C     THE PLOT FILE DATA ARE GIVEN IN SINGLE PRECISION.
C
C     CODE WRITTEN BY K.BEHRINGER, MAY 1986.
C
C     CODE REVISED BY K.BEHRINGER, SEPTEMBER 1988; REWRITTEN IN MAY 1996
C     EXTENDED TO A TRANSFORM SIZE OF 8192 DATA POINTS IN SEPTEMBER 1996
C     TRANSLATED IN DOUBLE PRECISION, APRIL 1997.
C
C     INTERACTIVE VERSION FOR THE VAX COMPUTER SYSTEM.
C
C     COMPILATION COMMAND : FORTRAN/CONTINUATION=24 CPSDES3
C     LINK COMMAND        : LINK CPSDES3
C     (NO IMSL ROUTINE REQUIRED).
C
C
C     DECLARATIONS
C
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*4 XNA,XNB
      CHARACTER RUN*40,IFORM*20,FINPUTA*40,FINPUTB*40,FPRINT*20,
     1FPLOT*20,LY,RY
      BYTE MODE,MODES,IPOLAR,IOUT
      DIMENSION F(4097),XASPEC(8194),XBSPEC(8194),YASPEC(8194),
     1YBSPEC(8194),PSDA(4097),PSDB(4097),CPSDR(4097),CPSDI(4097),
     2TFR(4097),TFI(4097),CPSD(4097),TF(4097),PHASE(4097),
     3PHASEC(4097),COH(4097),XA(8192),XB(8192)
      EQUIVALENCE (XASPEC(1),TFR(1)),(XASPEC(4098),TFI(1)),
     1(XBSPEC(1),CPSD(1)),(XBSPEC(4098),TF(1)),(YASPEC(1),PHASE(1)),
     2(YASPEC(4098),PHASEC(1)),(YBSPEC(1),COH(1))
      DATA MODES,F(1),F(2),F(3)/3,1.,2.666667D0,3.44644D0/
      DATA JOB,IPLOT,IPOLAR,LY,FPRINT,FPLOT/2*0,1,'Y',
     1'CPSD.PRT            ','CPSD0000.PLO        '/
C
C     STATEMENT FUNCTION
C
      NERROR(M1,M2,M3,M4,M5) =
     1M5+IABS(JZEXT(M4.GT.0))*(M1-1+(M2-M3)*(M4-1))
C
C     FORMATS
C
```

```
3000 FORMAT(1H1,40X,'P R O G R A M',5X,'C P S D E S 3',9X,
    1'(VERSION FOR THE VAX COMPUTER SYSTEM)'///1X,'RUN DENOTATION',
    236X,'RUN',5X,'=',1X,A//1X,'JOB-NUMBER',40X,'JOB',5X,'=',I5
    3//1X,'TRANSFORM SIZE',36X,'NT',6X,'=',I5
    4//1X,'NUMBER OF AVERAGES',32X,'NAV',5X,'=',I5
    5//1X,'NUMBER OF OVERLAPPED DATA POINTS',18X,'NRED',4X,'=',I5
    6//1X,'SIGNAL WINDOW FUNCTION',28X,'MODE',4X,'=',I5
    7//1X,'SAMPLING FREQUENCY (HZ)',27X,'SFR',5X,'=',1PD13.3
    8/1X,'SAMPLING INTERVAL (SEC)',27X,'DT',6X,'=',D13.3
    9/1X,'NYQUIST CUTOFF FREQUENCY (HZ)',21X,'CFR',5X,'=',D13.3
    A/1X,'NYQUIST CO-INTERVAL (HZ)',26X,'DF',6X,'=',D13.3
    B//1X,'REPRESENTATION OF THE COMPLEX OUTPUT FUNCTIONS',4X,'IPOLAR',
    C2X,'=',I5/3X,'(0 : REAL+IMAGINARY PARTS, 1 : MAGNITUDE+PHASE)'
    D//1X,'SELECTED DATA OUTPUT-FILES',24X,'IOUT',4X,'=',I5
    E/3X,'(0 : FPLOT, 1 : FPRINT, 2 : FPLOT+FPRINT)'
    F//1X,'NUMBER OF FPLOT',35X,'IPLOT',3X,'=',I5
    G//1X,'SIGNAL DATA INPUT FORMAT',26X,'IFORM',3X,'=',1X,A
    H//1X,'SIGNAL DATA FILES',33X,'FINPUTA =',1X,A
    I/51X,'FINPUTB =',1X,A
    K//1X,'EVALUATION STARTS AT DATA POINT NUMBER  NAS',5X,'=',I8,13X,
    L'NBS',5X,'=',I8
    M//1X,'SIGNAL DC TO BE SUBTRACTED',14X,'XA0',5X,'=',D16.3,5X,
    N'XB0',5X,'=',D16.3
    O//1X,'SIGNAL NORMALISATION FACTOR',13X,'GA',6X,'=',D16.3,5X,
    P'GB',6X,'=',D16.3)
3001 FORMAT(///1X,'READ-ERROR IN CHANNEL ',A,', AVERAGE NR. ',I4,
    1', DATA POINT NR. ',I4,3X,'=',3X,'POINT NR.',I9,
    2' ON THE DATA FILE.')
3002 FORMAT(///1X,'EOF IN CHANNEL ',A,', AVERAGE NR. ',I4,
    1', DATA POINT NR. ',I4,3X,'=',3X,'POINT NR.',I9,
    2' ON THE DATA FILE.')
3003 FORMAT(//1X,'IF CURRENT AVERAGE NR.GT.1 ,'/1X,
    1'EVALUATION OF THIS JOB CONTINUES WITH NAV = CURRENT AVERAGE NR.'
    2,' -1.')
3004 FORMAT(//1X,'SIGNAL VARIANCES',4X,'N = (1,NP)',10X,'VARA1',3X,
    1'=',1PD16.3,5X,'VARB1',3X,'=',D16.3
    2/21X,'N = (2,NP-1)',8X,'VARA2',3X,'=',D16.3,5X,'VARB2',3X,'=',
    3D16.3/)
3005 FORMAT(1H1,'RUN = ',A,4X,'JOB = ',I4,1P,(//2X,'N',7X,'F (HZ)',8X,
    1'PSDA',9X,'PSDB',8X,'CPSDR',8X,'CPSDI',9X,'TFR',10X,'TFI',10X,
    2'COH'/50(/1X,I3,8D13.3)/1H1))
3006 FORMAT(1H1,'RUN = ',A,4X,'JOB = ',I4,1P,(//2X,'N',7X,'F (HZ)',8X,
    1'PSDA',9X,'PSDB',9X,'CPSD',10X,'TF',9X,'PHASE',7X,'PHASEC',9X,
    2'COH'/50(/1X,I4,D12.3,7D13.3)/1H1))
3007 FORMAT(////1X,'E N D')
3010 FORMAT(1X,'PSD AVERAGES'/)
3015 FORMAT(6X,'NO REQUEST (NFL=NFU=0)')
3020 FORMAT(6X,'NFL =',I5,4X,'FL =',1PD10.3/6X,
    1'NFU =',I5,4X,'FU =',D10.3/41X,
    2'PSDAAV  =',D16.3,5X,'PSDBAV  =',D16.3)
4000 FORMAT(I3)
4001 FORMAT(1P,(8E10.3))
5000 FORMAT(A)
5001 FORMAT(I4)
5002 FORMAT(2X,I2)
5003 FORMAT(D10.3)
5004 FORMAT(I7)
```

```
6000 FORMAT(1H1,'PROGRAM CPSDES3',10X,'(VERSION FOR THE VAX COMPUTER)'
    1/1X,'SPECTRAL ANALYSIS OF TWO DIGITAL NOISE RECORDS,'
    2/1X,'ESTIMATION OF THE'
    3/6X,'POWER SPECTRAL DENSITY IN EACH CHANNEL,'
    4/6X,'CROSS-POWER SPECTRAL DENSITY,'
    5/6X,'TRANSFER FUNCTION (CHANNEL A = INPUT, CHANNEL B = OUTPUT),'
    6/6X,'SQUARED COHERENCE FUNCTION.'
    7//1X,'THE CODE HAS BEEN WRITTEN FOR INTERACTIVE OPERATION.'
    8/1X,'FOLLOW THE INSTRUCTION RULES FOR THE PARAMETER DATA INPUT !')
6001 FORMAT(//1X,'RUN DENOTATION ? (MAX.40 CH.)'/1X,
    1'^^^^^^^^^*^^^^^^^^^*^^^^^^^^^*^^^^^^^^^*')
6002 FORMAT(1X,'YOU SPECIFIED :'/1X,A)
6003 FORMAT('$CORRECT ? (Y/N) ')
6004 FORMAT(/1X,'TRANSFORM SIZE NT ? (I4,(32,8192))'/1X,'^^^^')
6005 FORMAT(1X,'INCORRECT VALUE, REPEAT !')
6006 FORMAT(1X,'YOU SPECIFIED :'/1X,I4)
6007 FORMAT(/1X,'NUMBER OF AVERAGES NAV ? (I4,MAX.1000)'/1X,'^^^^')
6008 FORMAT(/1X,'NUMBER OF DATA POINTS NRED TO BE OVERLAPPED ?',1X,
    1'(I4,(0,NT-1))'/1X,'^^^^')
6009 FORMAT(/1X,'TYPE OF SIGNAL WINDOW, MODE ? (SEE LIST ! (I1)'/1X,
    1'---^')
6010 FORMAT(/1X,'SAMPLING FREQUENCY SFR IN HZ ? (D10.3)'/1X,
    1'^^.^^^D^^^')
6011 FORMAT(1X,'YOU SPECIFIED :'/1X,1PD10.3)
6012 FORMAT(1X,'YOUR SPECIFICATIONS RESULT IN'
    1/6X,'SAMPLING INTERVAL (SEC)',7X,'DT  =',1PD11.3
    2/6X,'NYQUIST CUTOFF FREQUENCY (HZ) CFR =',D11.3
    3/6X,'NYQUIST CO-INTERVAL (HZ)',6X,'DF  =',D11.3)
6013 FORMAT(/1X,'DO YOU WISH REPRESENTATION OF THE COMPLEX SPECTRAL',
    11X,'FUNCTIONS BY MAGNITUDE'/1X,
    2'AND PHASE ? (Y/N). OTHERWISE THEY WILL BE REPRESENTED BY THE',
    31X,'REAL AND'/'$IMAGINARY PARTS. ')
6014 FORMAT(/1X,'DATA OUTPUT MODE, IOUT ? (I1)'/1X,
    1'(0 : PLOTFILES, 1 : PRINTFILES, 2 : PLOTFILES+PRINTFILE)'/1X,
    2'---^')
6015 FORMAT(/1X,'INITIAL NUMBER OF FPLOT, IPLOT ? (I3,(0,990))'/1X,
    1'-^^^')
6016 FORMAT(/1X,'SIGNAL DATA INPUT FORMAT IFORM ? (MAX.20 CH.,',1X,
    1'INCL.(...) )'/1X,'(SINGLE-READ FORMAT IN E- OR F-SPECIFICATION',
    21X,'EQUAL FOR BOTH CHANNELS)'/1X,'^^^^^^^^^*^^^^^^^^^*')
6017 FORMAT(/1X,'SIGNAL DATA FILE OF CHANNEL ',A,' ? (MAX.40 CH.)'/1X,
    1'^^^^^^^^^*^^^^^^^^^*^^^^^^^^^*^^^^^^^^^*')
6018 FORMAT(1X,'OPEN-ERROR, REPEAT !')
6019 FORMAT(/1X,'SPECIFY FOR CHANNEL ',A,' :')
6020 FORMAT(1X,'THE DATA POINT NUMBER NS WHERE EVALUATION SHOULD',1X,
    1'START (I7)'/1X,'^^^^^^^')
6021 FORMAT(1X,'THE SIGNAL DC X0 TO BE SUBTRACTED (D10.3)'/1X,
    1'^^.^^^D^^^')
6022 FORMAT(1X,'THE SIGNAL NORMALISATION FACTOR G (D10.3)'/1X,
    1'^^.^^^D^^^')
6023 FORMAT(1X,'YOU SPECIFIED :'/1X,I7/1X,1PD10.3/1X,D10.3)
6024 FORMAT(/1X,'READ-ERROR IN CHANNEL ',A,' !')
6025 FORMAT(/1X,'EOF IN CHANNEL ',A,' !')
6026 FORMAT(6X,'AVERAGE NR.',I8/6X,'DATA POINT NR.',I5/6X,
    1'= POINT NR.',I8,' ON THE DATA FILE.')
6027 FORMAT(/1X,'SIGNAL VARIANCES :'/6X,
    1'CHANNEL A',5X,'VARA1 =',1PD11.3,5X,'VARA2 =',D11.3/6X,
    2'CHANNEL B',5X,'VARB1 =',D11.3,5X,'VARB2 =',D11.3)
```

```
 6028 FORMAT(/'$NEXT JOB ? (Y/N) ')
 6029 FORMAT('$VERIFY ! (Y/N) ')
 6030 FORMAT(/1X,'CURRENT PLOT FILE NUMBER IPLOT = 1000; YOU CANNOT',
     11X,'CONTINUE !')
 6031 FORMAT(//1X,'JOB NR.',I5,'  STARTS !')
 6040 FORMAT(/1X,'LOWER FREQUENCY NUMBER NFL FOR PSD AVERAGES ?'/1X,
     1'(I4,(0,4096))'/1X,'^^^^')
 6045 FORMAT(/1X,'UPPER FREQUENCY NUMBER NFU FOR PSD AVERAGES ?'/1X,
     1'(I4,(0,4096))'/1X,'^^^^')
 6050 FORMAT(/1X,'YOU SPECIFIED :'/1X,'NFL = ',I4/1X,'NFU = ',I4)
 6055 FORMAT(1X,'IF NFL=NFU=0, NO PSD AVERAGES ARE CALCULATED !')
 6060 FORMAT(1X,'USE A DIFFERENT FILE NAME FOR FINPUTB !')
C
C     PARAMETER INPUT
C
      WRITE(6,6000)
      OPEN(UNIT=3,FILE=FPRINT,STATUS='NEW',ERR=100)
  110 WRITE(6,6001)
      READ(5,5000) RUN
      WRITE(6,6002) RUN
      WRITE(6,6003)
      READ(5,5000) RY
      IF(RY.NE.LY) GO TO 110
  115 WRITE(6,6004)
      READ(5,5001,ERR=115) NT
      DO 1 NU=5,13
      IF(NT.EQ.2**NU) GO TO 120
    1 CONTINUE
      WRITE(6,6005)
      GO TO 115
  120 WRITE(6,6006) NT
      WRITE(6,6003)
      READ(5,5000) RY
      IF(RY.NE.LY) GO TO 115
  125 WRITE(6,6007)
      READ(5,5001,ERR=125) NAV
      IF(NAV.LT.1) NAV=1
      IF(NAV.GT.1000) NAV=1000
      WRITE(6,6006) NAV
      WRITE(6,6003)
      READ(5,5000) RY
      IF(RY.NE.LY) GO TO 125
  130 WRITE(6,6008)
      READ(5,5001,ERR=130) NRED
      IF(NRED.LT.0) NRED=0
      IF(NRED.GE.NT) NRED=NT-1
      WRITE(6,6006) NRED
      WRITE(6,6003)
      READ(5,5000) RY
      IF(RY.NE.LY) GO TO 130
  135 WRITE(6,6009)
      READ(5,5002,ERR=135) MODE
      IF(MODE.LT.1) MODE=1
      IF(MODE.GT.MODES) MODE=MODES
      WRITE(6,6006) MODE
      WRITE(6,6003)
      READ(5,5000) RY
```

```
      IF(RY.NE.LY) GO TO 135
  140 WRITE(6,6010)
      READ(5,5003,ERR=140) SFR
      IF(SFR.GT.0.D0) GO TO 145
      WRITE(6,6005)
      GO TO 140
  145 WRITE(6,6011) SFR
      WRITE(6,6003)
      READ(5,5000) RY
      IF(RY.NE.LY) GO TO 140
      NTH = NT/2
      NU = NU-1
      INV = 0
      NP = NTH+1
      DT = 1.D0/SFR
      CFR = 5.D-1*SFR
      DF = CFR/DFLOTJ(NTH)
      PNORM = DT*F(MODE)/DFLOTJ(NT)
      WRITE(6,6012) DT,CFR,DF
  146 WRITE(6,6013)
      READ(5,5000) RY
      WRITE(6,6029)
      READ(5,5000) LY
      IF(RY.NE.LY) GO TO 146
      LY = 'Y'
      IF(RY.NE.LY) IPOLAR=0
  150 WRITE(6,6014)
      READ(5,5002,ERR=150) IOUT
      IF(IOUT.LT.0) IOUT=0
      IF(IOUT.GT.2) IOUT=2
      WRITE(6,6006) IOUT
      WRITE(6,6003)
      READ(5,5000) RY
      IF(RY.NE.LY) GO TO 150
      IF(IOUT.EQ.1) GO TO 160
  155 WRITE(6,6015)
      READ(5,5001,ERR=155) IPLOT
      IF(IPLOT.LT.1) IPLOT=1
      IF(IPLOT.GT.990) IPLOT=990
      WRITE(6,6006) IPLOT
      WRITE(6,6003)
      READ(5,5000) RY
      IF(RY.NE.LY) GO TO 155
  160 WRITE(6,6016)
      READ(5,5000) IFORM
      WRITE(6,6002) IFORM
      WRITE(6,6003)
      READ(5,5000) RY
      IF(RY.NE.LY) GO TO 160
      DO 2 N=1,NP
    2 F(N) = DF*DFLOTJ(N-1)
  165 WRITE(6,6040)
      READ(5,5001,ERR=165) NFL
  166 WRITE(6,6045)
      READ(5,5001,ERR=166) NFU
      IF(NFL.LT.0) NFL=0
      IF(NFU.GT.NTH) NFU=NTH
      IF(NFL.LE.NFU) GO TO 170
```

```
      NFL = 0
      NFU = 0
  170 WRITE(6,6050) NFL,NFU
      WRITE(6,6055)
      WRITE(6,6003)
      READ(5,5000) RY
      IF(RY.NE.LY) GO TO 165
      FL = F(NFL+1)
      FU = F(NFU+1)
C
C     LOOP SECTION
C
C     PARAMETER INPUT
C
  200 JOB = JOB+1
      WRITE(6,6031) JOB
      DO 3 N=1,3
  205 RY = 'A'
      WRITE(6,6017) RY
      ACCEPT 5000,FINPUTA
      WRITE(6,6002) FINPUTA
      WRITE(6,6003)
      READ(5,5000) RY
      IF(RY.NE.LY) GO TO 205
      OPEN(UNIT=1,FILE=FINPUTA,STATUS='OLD',ERR=215)
      GO TO 210
  215 WRITE(6,6018)
    3 CONTINUE
      STOP 'OPEN-ERROR FINPUTA'
  210 DO 4 N=1,3
  220 RY = 'B'
      WRITE(6,6017) RY
      ACCEPT 5000,FINPUTB
      WRITE(6,6002) FINPUTB
      WRITE(6,6003)
      READ(5,5000) RY
      IF(RY.NE.LY) GO TO 220
      IF(FINPUTB.NE.FINPUTA) GO TO 221
      WRITE(6,6060)
      GO TO 220
  221 OPEN(UNIT=2,FILE=FINPUTB,STATUS='OLD',ERR=230)
      GO TO 225
  230 WRITE(6,6018)
    4 CONTINUE
      STOP 'OPEN-ERROR FINPUTB'
  225 IF(IOUT.EQ.1) GO TO 235
      ENCODE(3,4000,FPLOT(6:8)) IPLOT
      OPEN(UNIT=4,FILE=FPLOT,STATUS='NEW',ERR=240)
  235 K = 0
  245 RY = 'A'
      WRITE(6,6019) RY
  246 WRITE(6,6020)
      READ(5,5004,ERR=246) NAS
  247 WRITE(6,6021)
      READ(5,5003,ERR=247) XA0
  248 WRITE(6,6022)
      READ(5,5003,ERR=248) GA
```

```
      IF(NAS.LT.1) NAS=1
      IF(GA.LE.0.D0) GA=1.D0
      WRITE(6,6023) NAS,XA0,GA
      WRITE(6,6003)
      READ(5,5000) RY
      IF(RY.NE.LY) GO TO 245
  250 RY = 'B'
      WRITE(6,6019) RY
  251 WRITE(6,6020)
      READ(5,5004,ERR=251) NBS
  252 WRITE(6,6021)
      READ(5,5003,ERR=252) XB0
  253 WRITE(6,6022)
      READ(5,5003,ERR=253) GB
      IF(NBS.LT.1) NBS=1
      IF(GB.LE.0.D0) GB=1.D0
      WRITE(6,6023) NBS,XB0,GB
      WRITE(6,6003)
      READ(5,5000) RY
      IF(RY.NE.LY) GO TO 250
      WRITE(3,3000) RUN,JOB,NT,NAV,NRED,MODE,SFR,DT,CFR,DF,IPOLAR,IOUT,
     1IPLOT,IFORM,FINPUTA,FINPUTB,NAS,NBS,XA0,XB0,GA,GB
      IF(IOUT.NE.1) IPLOT=IPLOT+1
C
C     DATA INPUT
C
      DO 5 N=1,NAS
    5 READ(1,IFORM,ERR=300,END=305) XNA
      DO 6 N=1,NBS
    6 READ(2,IFORM,ERR=301,END=306) XNB
      XA(1) = DBLE(XNA)-XA0
      XB(1) = DBLE(XNB)-XB0
      N1 = 2
      N2 = NT-NRED
      N3 = 0
      DO 7 N=1,NP
      PSDA(N) = 0.D0
      PSDB(N) = 0.D0
      CPSDR(N) = 0.D0
    7 CPSDI(N) = 0.D0
      DO 10 K=1,NAV
      IF(N3.EQ.0) GO TO 310
      DO 11 N=1,NRED
      NS = N+N2
      XA(N) = XA(NS)
   11 XB(N) = XB(NS)
  310 DO 12 N=N1,NT
      READ(1,IFORM,ERR=300,END=305) XNA
      READ(2,IFORM,ERR=301,END=306) XNB
      XA(N) = DBLE(XNA)-XA0
   12 XB(N) = DBLE(XNB)-XB0
      DO 9 N=1,NT
      XASPEC(N) = XA(N)
    9 XBSPEC(N) = XB(N)
C
C     CALCULATIONS
C
      CALL XSFFT2(XASPEC,YASPEC,NTH,NU,INV)
```

```
      CALL XSFFT2(XBSPEC,YBSPEC,NTH,NU,INV)
      YASPEC(1) = 0.D0
      YASPEC(NP) = 0.D0
      YBSPEC(1) = 0.D0
      YBSPEC(NP) = 0.D0
      GO TO (320,321,322),MODE
C     CALLS FOR WINDOW FUNCTIONS. IF FURTHER WINDOW FUNCTIONS ARE
C     REQUIRED EXTEND THE GO-TO-STATEMENT AND THE FIRST DATA DECLARATION
  321 CALL HANW1(XASPEC,YASPEC,NP)
      CALL HANW1(XBSPEC,YBSPEC,NP)
      GO TO 320
  322 CALL BLHAW1(XASPEC,YASPEC,NP)
      CALL BLHAW1(XBSPEC,YBSPEC,NP)
  320 DO 13 N=1,NP
      XA1 = XASPEC(N)
      YA1 = YASPEC(N)
      XB1 = XBSPEC(N)
      YB1 = YBSPEC(N)
      PSDA(N) = PSDA(N)+XA1**2+YA1**2
      PSDB(N) = PSDB(N)+XB1**2+YB1**2
      CPSDR(N) = CPSDR(N)+XA1*XB1+YA1*YB1
   13 CPSDI(N) = CPSDI(N)+XA1*YB1-XB1*YA1
      IF(K.GT.1) GO TO 10
      N1 = NRED+1
      N3 = NRED
   10 CONTINUE
      K = NAV
      GO TO 400
  300 RY = 'A'
      NS = NAS
      GO TO 350
  301 RY = 'B'
      NS = NBS
  350 WRITE(6,6024) RY
      NS = NERROR(NS,NT,NRED,K,N)
      WRITE(6,6026) K,N,NS
      WRITE(3,3001) RY,K,N,NS
      GO TO 360
  305 RY = 'A'
      NS = NAS
      GO TO 355
  306 RY = 'B'
      NS = NBS
  355 WRITE(6,6025) RY
      NS = NERROR(NS,NT,NRED,K,N)
      WRITE(6,6026) K,N,NS
      WRITE(3,3002) RY,K,N,NS
  360 WRITE(6,3003)
      WRITE(3,3003)
      IF(K.LE.1) GO TO 500
      K = K-1
  400 XA0 = PNORM/DFLOTJ(K)
      XA1 = XA0/GA**2
      XB1 = XA0/GB**2
      XB0 = XA0/(GA*GB)
      DO 14 N=1,NP
      PSDA(N) = XA1*PSDA(N)
```

```
      PSDB(N) = XB1*PSDB(N)
      CPSDR(N) = XB0*CPSDR(N)
   14 CPSDI(N) = XB0*CPSDI(N)
      XA1 = 0.D0
      XB1 = 0.D0
      DO 15 N=2,NTH
      XA1 = XA1+PSDA(N)
   15 XB1 = XB1+PSDB(N)
      YA1 = XA1+PSDA(1)+PSDA(NP)
      YB1 = XB1+PSDB(1)+PSDB(NP)
      XA0 = 2.D0*DF
      XA1 = XA0*XA1
      YA1 = XA0*YA1
      XB1 = XA0*XB1
      YB1 = XA0*YB1
      WRITE(6,6027) YA1,XA1,YB1,XB1
      WRITE(3,3004) YA1,YB1,XA1,XB1
      WRITE(3,3010)
      IF(NFL.EQ.0.AND.NFU.EQ.0) GO TO 401
      PSDAAV = 0.D0
      PSDBAV = 0.D0
      DO 20 N=NFL+1,NFU+1
      PSDAAV = PSDAAV+PSDA(N)
   20 PSDBAV = PSDBAV+PSDB(N)
      XA0 = DFLOTJ(NFU-NFL+1)
      PSDAAV = PSDAAV/XA0
      PSDBAV = PSDBAV/XA0
      WRITE(3,3020) NFL,FL,NFU,FU,PSDAAV,PSDBAV
      GO TO 402
  401 WRITE(3,3015)
  402 DO 16 N=1,NP
      XA1 = PSDA(N)
      XB1 = PSDB(N)
      XA0 = CPSDR(N)
      XB0 = CPSDI(N)
      IF(XA1.GT.0.) GO TO 405
      TFR(N) = 0.D0
      TFI(N) = 0.D0
      YA1 = 0.D0
      GO TO 410
  405 TFR(N) = XA0/XA1
      TFI(N) = XB0/XA1
      YA1 = (XA0**2+XB0**2)/XA1
  410 IF(XB1.GT.0.D0) GO TO 415
      COH(N) = 0.D0
      GO TO 16
  415 COH(N) = YA1/XB1
   16 CONTINUE
C
C     READOUT
C
      K = IOUT-1
      IF(IPOLAR.GT.0) GO TO 420
      IF(K.LT.0) GO TO 425
      WRITE(3,3005) RUN,JOB,(N,F(N),PSDA(N),PSDB(N),CPSDR(N),
     1CPSDI(N),TFR(N),TFI(N),COH(N),N=1,NP)
      IF(K.EQ.0) GO TO 505
  425 WRITE(4,4001) (SNGL(F(N)),SNGL(PSDA(N)),SNGL(PSDB(N)),
```

```
     1SNGL(CPSDR(N)),SNGL(CPSDI(N)),SNGL(TFR(N)),SNGL(TFI(N)),
     2SNGL(COH(N)),N=1,NP)
      GO TO 435
  420 DO 17 N=1,NP
   17 TF(N) = DSQRT(TFR(N)**2+TFI(N)**2)
      CALL POLAR1V(CPSDR,CPSDI,CPSD,PHASE,PHASEC,NP)
      IF(K.LT.0) GO TO 430
      WRITE(3,3006) RUN,JOB,(N,F(N),PSDA(N),PSDB(N),CPSD(N),TF(N),
     1PHASE(N),PHASEC(N),COH(N),N=1,NP)
      IF(K.EQ.0) GO TO 505
  430 WRITE(4,4001) (SNGL(F(N)),SNGL(PSDA(N)),SNGL(PSDB(N)),
     1SNGL(CPSD(N)),SNGL(TF(N)),SNGL(PHASE(N)),SNGL(PHASEC(N)),
     2SNGL(COH(N)),N=1,NP)
  435 CLOSE(UNIT=4,STATUS='KEEP')
      GO TO 505
  500 IF(IOUT.EQ.1) GO TO 505
      CLOSE(UNIT=4,STATUS='DELETE')
  505 CLOSE(UNIT=1,STATUS='KEEP')
      CLOSE(UNIT=2,STATUS='KEEP')
  510 WRITE(6,6028)
      READ(5,5000) RY
      WRITE(6,6029)
      READ(5,5000) LY
      IF(RY.NE.LY) GO TO 510
      LY = 'Y'
      IF(RY.NE.LY) GO TO 515
      IF(IPLOT.LT.1000) GO TO 200
      WRITE(6,6030)
  515 WRITE(3,3007)
      STOP
  100 STOP 'OPEN-ERROR FPRINT'
  240 STOP 'OPEN-ERROR FPLOT'
      END
C===============================================
      SUBROUTINE POLAR1V(X,Y,GAIN,PHASE,PHASEC,NPTS)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION X(1),Y(1),GAIN(1),PHASE(1),PHASEC(1)
      EQUIVALENCE (CR,PH),(CI,PHS),(G,DPH)
      DATA CON/5.729577951D1/
      DO 1 I=1,NPTS
      CR = X(I)
      CI = Y(I)
      GAIN(I) = DSQRT(CR**2+CI**2)
      G = GAIN(I)
      IF(G.GT.0.D0) GO TO 105
      PHASE(I) = 0.D0
      GO TO 1
  105 PHASE(I) = DSIGN(CON*DACOS(CR/G),CI)
    1 CONTINUE
      PH = 0.D0
      PHASEC(1) = PHASE(1)
      DO 2 I=2,NPTS
      PHS = PHASE(I)
      DPH = PHS-PHASE(I-1)
      IF(DPH.GT.1.80D2) PH=PH-3.60D2
      IF(DPH.LT.-1.80D2) PH=PH+3.60D2
    2 PHASEC(I) = PHS+PH
```

```
      RETURN
      END
C==================================================
      SUBROUTINE XSFFT2(X,Y,N,NU,INV)
C     XSFFT2 IS A FAST FOURIER TRANSFORM PROCEDURE FOR THE TIME SERIES
C     X (INV=0) AND THE INSTANTANEOUS SPECTRUM S (INV=1) USING THE
C     FFT-SUBROUTINE OF E.O.BRIGHAM. IT IS A MODIFIED VERSION OF ITFFT2.
C
C     N MUST BE A NUMBER EQUAL 2**NU (NO INTERNAL CHECK),
C     DIMENSION X(2*N+1),Y(2*N+1)
C
C     INV=0 : FORWARD TRANSFORM, 2*N DATA POINTS OF X
C
C     INV=1 : INVERSE TRANSFORM, N+1 DATA POINTS OF S
C                               IMAG(S(1)),IMAG(S(N+1)) ARE SET = 0.
C
C                INPUT  X          Y         OUTPUT  X          Y
C
C     INV=0             X     INTERNALLY       REAL(S)   IMAG(S)
C                                USED
C     INV=1         REAL(S)   IMAG(S)                 X     INTERNALLY
C                                          X(2*N+1)=X(1)    USED
C
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION X(1),Y(1)
      COMMON/FFTC/ARG1
      DATA PI/3.1415926535898D0/
      N1 = N+1
      N2 = 2*N1
      ARG = PI/DFLOTJ(N)
      ARG1 = 2.D0*ARG
      FAC = 5.D-1
      IF(INV.GE.1) GO TO 110
      DO 2 I=1,N
      K = 2*I
      Y(I) = X(K)
    2 X(I) = X(K-1)
      CALL FFT(X,Y,N,NU)
      X(N1) = X(1)
      Y(N1) = Y(1)
      GO TO 120
  110 FAC = FAC/DFLOTJ(N)
      Y(1) = 0.D0
      Y(N1) = 0.D0
      DO 3 K=2,N
   3  Y(K) = -Y(K)
  120 DO 4 K=1,N
      I = N2-K
      X(I) = X(K)
    4 Y(I) = Y(K)
      DO 5 K=1,N1
      I = N+K
      X1 = X(K)
      X2 = X(I)
      Y1 = Y(K)
      Y2 = Y(I)
      XS = X1+X2
      XD = X1-X2
```

```
      YS = Y1+Y2
      YD = Y1-Y2
      X2 = ARG*DFLOTJ(K-1)
      X1 = DSIN(X2)
      X2 = DCOS(X2)
      X(K) = FAC*(XS+YS*X2-XD*X1)
    5 Y(K) = FAC*(YD-XD*X2-YS*X1)
      IF(INV.LE.0) RETURN
      CALL FFT(X,Y,N,NU)
      N2 = N2-1
      DO 6 K=1,N
      II = N1-K
      I = N2-2*K
      X(I) = X(II)
    6 X(I+1) = -Y(II)
      X(N2) = X(1)
      RETURN
      END
C===============================================
      SUBROUTINE FFT(XREAL,XIMAG,N,NU)
C     FFT-PROCEDURE COPIED FROM E.O.BRIGHAM, "THE FAST FOURIER TRANS-
C     FORM" ,PRENTICE-HALL INC., 1974.
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION XREAL(1),XIMAG(1)
      COMMON/FFTC/ARG1
      N2 = N/2
      NU1 = NU-1
      K = 0
      DO 1 L=1,NU
   10 DO 2 I=1,N2
      P = IBITR(K/2**NU1,NU)
      ARG = ARG1*P
      C = DCOS(ARG)
      S = DSIN(ARG)
      K1 = K+1
      K1N2 = K1+N2
      TREAL = XREAL(K1N2)*C+XIMAG(K1N2)*S
      TIMAG = XIMAG(K1N2)*C-XREAL(K1N2)*S
      XREAL(K1N2) = XREAL(K1)-TREAL
      XIMAG(K1N2) = XIMAG(K1)-TIMAG
      XREAL(K1) = XREAL(K1)+TREAL
      XIMAG(K1) = XIMAG(K1)+TIMAG
    2 K = K+1
      K = K+N2
      IF(K.LT.N) GO TO 10
      K = 0
      NU1 = NU1-1
    1 N2 = N2/2
      DO 3 K=1,N
      I = IBITR(K-1,NU)+1
      IF(I.LE.K) GO TO 3
      TREAL = XREAL(K)
      TIMAG = XIMAG(K)
      XREAL(K) = XREAL(I)
      XIMAG(K) = XIMAG(I)
      XREAL(I) = TREAL
      XIMAG(I) = TIMAG
```

```
    3 CONTINUE
      RETURN
      END
C==================================================
      FUNCTION IBITR(J,NU)
      J1 = J
      IBITR = 0
      DO 1 I=1,NU
      J2 = J1/2
      IBITR = IBITR*2+(J1-2*J2)
    1 J1 = J2
      RETURN
      END
C==================================================
      SUBROUTINE HANW1(X,Y,NP)
C     HANNING WINDOW (FREQUENCY DOMAIN)
      IMPLICIT REAL*8 (A-H,O-Z)
      COMPLEX*16 Z1,Z2,Z3,Z
      DIMENSION X(1),Y(1)
      NP1 = NP-1
      X1 = X(1)
      X2 = X(2)
      Z1 = DCMPLX(X1,Y(1))
      Z2 = DCMPLX(X2,Y(2))
      DO 1 N=2,NP1
      N1 = N+1
      Z3 = DCMPLX(X(N1),Y(N1))
      Z = 5.D-1*Z2-2.5D-1*(Z1+Z3)
      X(N) = DREAL(Z)
      Y(N) = DIMAG(Z)
      Z1 = Z2
    1 Z2 = Z3
      X(1) = 5.D-1*(X1-X2)
      X(NP) = 5.D-1*DREAL(Z2-Z1)
      RETURN
      END
C==================================================
      SUBROUTINE BLHAW1(X,Y,NP)
C     BLACKMAN-HARRIS 4-TERM WINDOW (-74 DB WINDOW, FREQUENCY DOMAIN)
      IMPLICIT REAL*8 (A-H,O-Z)
      COMPLEX*16 Z0,Z1,Z2,Z3,Z4,Z5,Z6,Z,CF,Y0,Y1,Y2,Y3,Y4,Y5,Y6
      DIMENSION X(1),Y(1)
      DATA A0,A1,A2,A3/4.0217D-1,2.4852D-1,4.946D-02,9.4D-04/
      CF(Y0,Y1,Y2,Y3,Y4,Y5,Y6) = A0*Y0-A1*(Y1+Y2)+A2*(Y3+Y4)-A3*(Y5+Y6)
      NP1 = NP-3
      Z1 = DCMPLX(X(1),Y(1))
      Z2 = DCMPLX(X(2),Y(2))
      Z3 = DCMPLX(X(3),Y(3))
      Z4 = DCMPLX(X(4),Y(4))
      Z5 = DCMPLX(X(5),Y(5))
      Z6 = DCMPLX(X(6),Y(6))
      Z = CF(Z1,Z2,DCONJG(Z2),Z3,DCONJG(Z3),Z4,DCONJG(Z4))
      X(1) = DREAL(Z)
      Z = CF(Z2,Z3,Z1,Z4,DCONJG(Z2),Z5,DCONJG(Z3))
      X(2) = DREAL(Z)
      Y(2) = DIMAG(Z)
      Z = CF(Z3,Z4,Z2,Z5,Z1,Z6,DCONJG(Z2))
      X(3) = DREAL(Z)
```

```
      Y(3) = DIMAG(Z)
      DO 1 N=4,NP1
      Z0 = Z1
      Z1 = Z2
      Z2 = Z3
      Z3 = Z4
      Z4 = Z5
      Z5 = Z6
      N1 = N+3
      Z6 = DCMPLX(X(N1),Y(N1))
      Z = CF(Z3,Z4,Z2,Z5,Z1,Z6,Z0)
      X(N) = DREAL(Z)
    1 Y(N) = DIMAG(Z)
      Z = CF(Z4,Z5,Z3,Z6,Z2,DCONJG(Z5),Z1)
      X(NP-2) = DREAL(Z)
      Y(NP-2) = DIMAG(Z)
      Z = CF(Z5,Z6,Z4,DCONJG(Z5),Z3,DCONJG(Z4),Z2)
      X(NP-1) = DREAL(Z)
      Y(NP-1) = DIMAG(Z)
      Z = CF(Z6,DCONJG(Z5),Z5,DCONJG(Z4),Z4,DCONJG(Z3),Z3)
      X(NP) = DREAL(Z)
      RETURN
      END
```

# 5. PROGRAM FFTF2

Precision : double

Operation : interactive

Required auxiliary routines : from IMSL : DF2TRB, DF2TRF, DFFTRI

Purpose of the program :

Band-Pass Filtering of Noise Records by FFT-Techniques.

Feature Summary :

- Record segmentation.
- Rectangular filter characteristics.
- 2 options for the removal of the DC component (at low-pass filtering).
- 2 options for output data smoothing.
- Within a run, a practically unlimited number of signal data files, each case called a job, can be treated.

## 5.1 Mathematical Background

In principle, for band-pass filtering of signal data any digital filter type which has smooth characteristics and is very steep at the corner frequencies can be used. However, the simplest digital filter which approximates the shape of an ideal rectangular filter is obtained by FFT-techniques. This filter shape is assumed in the application of the oscillator models to the determination of the decay ratio in BWR stability analysis via the auto-correlation function (ACF) method. The filtering method is based on signal segmentation. The number of data points in a segment must be a power of 2. The data in a segment are at first Fourier transformed. According to the given filter boundaries $f_L$ and $f_H$, where $f_L$ is the lower filter cutoff frequency and $f_H$ is the upper filter cutoff frequency, the code searches for the nearest values $f_{Lp}$ and $f_{Hp}$ which are multiples of the Nyquist co-interval. These values have then to be transferred as input parameter data to the fit codes (ACFIT6, ACFIT7, ACFIT7SA). The complex spectral coefficients are retained in the closed frequency interval $[f_{Lp},f_{Hp}]$, the other coefficients outside this interval are set equal zero. The inverse FFT gives then the filtered signal data in a segment. In this way, the data are treated segment by segment. The code allows filtering of a practically unlimited number of data records from different files within a run. The data output files have identification numbers. Common input parameters are only the segment length (number of data points in a segment) and the sampling frequency. All other input parameters, like the data file name, data format, number of segments, the filter cutoff frequencies, the option for the DC component removal and the option for data smoothing are individual. If one gives a high value for the number of segments to be treated, the code searches for the maximum available number of completely filled segments. One can also apply only low-pass filtering. For this case, there are two options available for the DC component removal, either with a DC value calculated over all segments (global removal), or with DC values calculated for each segment separately (local removal, low frequency trend elimination). The FFT filter is not universal and is not listed in the modern filter handbook by Chen (1995). It picks out the frequency components at multiples of the Nyquist co-interval within the given cutoff frequencies. It is specially related to the ACF estimation procedure on previously filtered signal data by the code ACCF1 (Section 7) or ACCF2 (Section 8). These ACF estimation codes are based on the application of FFT techniques.

One may asume that the original signal data exhibit the digital image of a continuous record, and, hence, the filtered data can be regarded as beeing continuous within each segment. But on the edges between succeeding segments discontinuities appear. They lead to distortions in a PSD estimation with the same segment length and without use of segment overlap (code CPSDES3), but with time-shifted segments, i.e. if one starts scanning e.g. in the middle of the first segment. For this reason, an option for smoothing of the filtered signal data has been provided. Behringer et al. (1986) have developed a smoothing method with a two-sequence window operator in connection with a similar problem which arrised at the artificial generation of coloured Gaussian random noise by the Rice formula (code RICE3, Section 14). The procedure preserves the signal variance. Two window operators are known : the two-sequence cosine window and the two-sequence square root window. Both windows have practically equivalent properties. They are available in the code as options. However, their efficiencies have not fully be tested on actual neutron signals in the BWR stability analysis, because there was no need with respect to the application of this FFT filter to the code ACCF1 or ACCF2. The smoothing procedure requires at least two succeeding segments of filtered signal data.

## 5.2 References

Behringer K.,Nishihara H. and Spiekerman G.(1986). Ann.Nucl.Energy 13, 443, EIR Report 601.

Chen Wai-Kai (1995). The Circuits and Filters Handbook, CRC Press, Boca Raton, Florida, USA.

## 5.3 Files

There are 4 files :

- 'FINPUT'

This file contains the signal data to be filtered. The file name must be interactively specified for each job. The code assumes that the signal data are of the REAL*4 type and are written as column vector.

- FFTF2.PRT

This file is meant for printing. It contains (with text) all input parameter data, additionally calculated data and messages from the computation progress.

- FFTF2_'''.DAT

This file contains the filtered signal data. For each job a separate file with a current extension number of up to 3 digits is opened. The initial extension number for the first job must be interactively specified. There are 5 columns with the line format (1P,2(1X,I8,E13.5),E15.7) :

column

1 : relative data point number, starting with the value 1,
2 : relative time in sec, starting with the value 0.,
3 : absolute data point number, referring to the record begin with the value 1, if an initial part of the input signal data is passed over,
4 : absolute time in sec, starting with the value 0. at the record begin,
5 : filtered signal data.

The 4$^{th}$ file is used as an unformatted scratch file and serves for intermediate data storage. The name must be adapted by the user (DATA declaration of line 35 in the listing). It is deleted at the termination of a run. It is not declared as a proper file of the SCRATCH type because of the number of blocks limited by the operating system.

## 5.4 Parameter Data Input

The interactive procedure for the input of the parameter data with text and format indications on the terminal screen is very similar to that of the code CPSDES3. Protection is provided against typing errors. Each data typed in is immediately rewritten to the screen. It can be corrected if necessary, and must be verified for final acceptance. If in a number field an unallowed character is erroneously typed in, the code repeats the question. There are common and individual parameter data. The common parameter data refer to all jobs to be treated within a run. The individual parameter data must be given for each job.

Common Parameter Data Input :

- format (A), RUN

A string of max. 50 characters for run identification.

- format (D10.3), SFR

Sampling frequency in Hz.

- format (I4), NT

Number of signal data points in a segment (equivalent to the FFT transform size).

Internal restriction : The value must be a power of 2 within the range of $2^5$ to $2^{10}$.

- format (I3), IDAT

Initial extension number of the file FFTF2_'''.DAT for the first job. For each further job within a run, this number is incremented by 1.

Internal restrictions: IF(IDAT.LT.1) IDAT=1 (default value)

IF(IDAT.GT.990) IDAT=990

If the run starts with the highest allowed number, the code assumes that the run does not encompasses more than 10 jobs, otherwise the run will normally terminate after job 10 with a preceding message to the terminal. IDAT allows grouping of the numbers of the data output files, if sets of data input files are to be treated by more than one run.

Individual Parameter Data Input :

- format (A), FINPUT

A string of max. 50 characters for the file name of the signal data to be filtered. If after a third trial this file cannot be opened, the code stops with a message.

- format (A), IFORM

A string of max. 20 characters for specifying the single-read format of the signal data to be filtered. The format must be given in E- or F-specification including the brackets.

The question for a new value of IFORM in the next job can be by-passed, retaining the value used in the previous job.

- format (I5), NSR

Requested number of segments. This number refers to that part of signal data points which should be filtered, if an initial part of data points is passed over by setting the parameter NSTART greater than 1.

Internal restrictions: IF(NSR.LT.2) NSR=2 (default value)

IF(NSR.GT.10000) NSR=10000

For convenience, an exact number must not be given, if one wishes to scan until the record end. In this case, the maximum allowed value can be given. The code checks on the EOF mark and searches for the maximum possible value NSA, which is the accepted number of completely filled segments. If NSA<2, the job is cancelled and the code continues asking for the next job. In the case of a detected read-error, the code stops. The point number of the EOF mark or the point number of the read-error, both counted from the record begin, will be output.

- format (I7), NSTART

The number specifies the first data point number where filtering starts. If it is set >1, then NSTART-1 initial data points are passed over.

Default value : NSTART=1

- format (D10.3,1X,D10.3), FRL,FRH

Requested values for the lower and the upper cutoff frequencies of the filter in Hz. The code searches for the nearest possible values which are multiples of the Nyquist co-interval. These values define a closed interval of the filter bandwidth. One can also filter out only one frequency component.

Internal restrictions: IF(FRL.LT.0.) FRL=0.

IF(FRL.GT.CFR) FRL=CFR

IF(FRH.GT.CFR) FRH=CFR

IF(FRH.LT.FRL) FRH=FRL

CFR is the value of the Nyquist cutoff frequency in Hz (calculated by the code). The question for new values of FRL and FRH in the next job can be by-passed.

- format (I1), IOPTDC

Parameter for the DC component removal.

IOPTDC = 0: No DC component removal, recommended option, if FRL>0..

1: Local DC component subtraction. The DC value is estimated in each segment separately.

2: Global DC component subtraction. The DC value is estimated over all accepted segments. It is written on the print file.

Internal restrictions : IF(IOPTDC.LT.0) IOPTDC=0 (default value)

IF(IOPTDC.GT.2) IOPTDC=2

- format (I1), IOPTSM

Parameter for output data smoothing.

IOPTSM = 0: No smoothing

1: Smoothing with the two-sequence cosine window.

2: Smoothing with the two-sequence square-root window.

Internal restrictions : IF(IOPTSM.LT.0) IOPTSM=0 (default value)

IF(IOPTSM.GT.2) IOPTSM=2

The question for new values of IOPTDC and IOPTSM in the next job can be by-passed. The question is common for both parameters.

## 5.5 Branchings

After a job has terminated, there will be a question concerning continuing with the next job. This question must be acknowledged with YES or NOT. If there is another job, the code returns to asking for the next individual input parameter data.

## 5.6 Numbered Stops

STOP 1: After a third trial of opening the file 'FINPUT'.

STOP 2: Read-error of a signal data.

STOP 3: Opening error of the file FFTF2_'''.DAT .

STOP 4: Opening error of file 4 (scratch file).

# LISTING

## F F T F 2

```
      PROGRAM FFTF2
C
C     THE CODE GENERATES A NEW DATA RECORD FILTERED BY FAST FOURIER
C     TRANSFORM FROM A GIVEN DATA RECORD. WITHIN A RUN A PRACTICALLY
C     UNLIMITED NUMBER OF DATA RECORDS CAN BE FILTERED. THE CODE
C     CONTAINS 3 OPTIONS FOR DC TREATMENT, AND 3 OPTIONS FOR OUTPUT
C     DATA SMOOTHING.
C     VERSION INCLUDING FILTER PHASES.
C
C     CODE WRITTEN BY K.BEHRINGER, AUGUST 1998, FOR OPERATION
C     ON THE DEC-ALPHA 2100 COMPUTER.
C
C     IMSL ROUTINES ARE REQUIRED.
C
C     LINK COMMAND : IMPORT IMSL
C                    LINK FFTF2,IMSLIBG_STATIC/OPT,IMSLPSECT/OPT
C
C     NOTE : IF ONE CHANGES THE FILE3 SPECIFICATION, THEN ONE HAS
C            TO ADAPT THE ENCODE STATEMENT ON LINE 331.
C
C
C     DECLARATIONS
C
      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER (NTMAX=1024)
      REAL*4 XSS
      CHARACTER*50 FINPUT,FILE1,FILE2,FILE3,FILE4,RUN
      CHARACTER RY,RYV,LY,IFORM*20
      DIMENSION X(NTMAX),XS(NTMAX),COEF(NTMAX),WFFTR(2*NTMAX+15)
      EQUIVALENCE (FILE1,FINPUT)
      DATA JOB,JOBS,LY/2*0,'Y'/,
     1FILE1/'<FINPUT>                                          '/,
     2FILE2/'FFTF2.PRT                                         '/,
     3FILE3/'FFTF2_^^^.DAT                                     '/,
     4FILE4/'DISK_216_DAT0:[BEHRINGER]FFTF2.SCR                '/
      EXTERNAL WINDOWA,WINDOWB
C
C     FORMATS
C
 2000 FORMAT('1',40X,'P R O G R A M     F F T F 2'///1X,
     1'FILES :'/1X,'DATA INPUT',5X,'FILE1 = ',A/1X,
     2'PRINT OUTPUT',3X,'FILE2 = ',A/1X,
     3'DATA OUTPUT',4X,'FILE3 = ',A/1X,
     4'SCRATCH FILE',3X,'FILE4 = ',A///1X,
     5'RUN DENOTATION',26X,'RUN',5X,'=',1X,A//1X,
     6'SAMPLING FREQUENCY (HZ)',17X,'SFR',5X,'=',1PD15.5//1X,
     7'TRANSFORM SIZE',26X,'NT',6X,'=',I5//1X,
     8'NYQUIST INTERVAL (SEC)',18X,'DT',6X,'=',D15.5/1X,
     9'NYQUIST CUTOFF FREQUENCY (HZ)',11X,'CFR',5X,'=',D15.5/1X,
     A'NYQUIST CO-INTERVAL (HZ)',16X,'DFR',5X,'=',D15.5/1X,
     B'TIME LENGTH OF A SEGMENT (SEC)',10X,'TS',6X,'=',D15.5//1X,
     C'INITIAL NUMBER OF DATA OUTPUT FILE',6X,'IDAT',4X,'=',I5)
```

```
2005 FORMAT('1JOB NUMBER',40X,'JOB',5X,'=',I9//1X,
    1'DATA INPUT FILE',35X,'FINPUT',2X,'=',1X,A//1X,
    2'DATA FORMAT',39X,'IFORM',3X,'=',1X,A//1X,
    3'REQUESTED NUMBER OF SEGMENTS',22X,'NSR',5X,'=',I9//1X,
    4'EVALUATION START POINT NUMBER',21X,'NSTART',2X,'=',I9/)
2010 FORMAT(1X,'EOF AT DATA POINT NUMBER',26X,'NEOF',4X,'=',I9)
2015 FORMAT(1X,'AVAILABLE NUMBER OF SEGMENTS',22X,'NSA',5X,'=',I9/1X,
    1'AVAILABLE NUMBER OF DATA POINTS',19X,'NPA',5X,'=',I9/)
2020 FORMAT(1X,'READ-ERROR AT DATA POINT NUMBER',19X,'NERROR',2X,'=',
    1I9)
2025 FORMAT(1X,'LOWER CUTOFF FREQUENCY :'/11X,
    1'REQUESTED (HZ)',26X,'FRL',5X,'=',1PD19.5/11X,
    2'NEAREST POSSIBLE (HZ)',19X,'FRLP',4X,'=',D19.5/11X,
    3'FREQUENCY NUMBER',24X,'KL',6X,'=',I9//1X,
    4'UPPER CUTOFF FREQUENCY :'/11X,
    5'REQUESTED (HZ)',26X,'FRH',5X,'=',D19.5/11X,
    6'NEAREST POSSIBLE (HZ)',19X,'FRHP',4X,'=',D19.5/11X,
    7'FREQUENCY NUMBER',24X,'KH',6X,'=',I9/)
2030 FORMAT(1X,'OPTION PARAMETER FOR DC-SUBTRACTION',15X,'IOPTDC',
    12X,'=',I9/11X,
    2'(0:NO,1:LOCAL,2:GLOBAL)'//1X,
    3'OPTION PARAMETER FOR OUTPUT DATA SMOOTHING',8X,'IOPTSM',
    42X,'=',I9/11X,
    5'(0:NO,1:COS-WINDOW,2:SQUARE-ROOT-WINDOW)'/11X,
    6'(FOR IOPTSM>0 : NSA=NSA-1,NPA=NPA-NT)'/)
2035 FORMAT(1X,'GLOBAL DC-MEAN VALUE',30X,'XMEAN',3X,'=',1PD19.5/)
2040 FORMAT(////11X,'E N D')
3000 FORMAT(1P,2(1X,I8,E13.5),E15.7)
6000 FORMAT('1',20X,'P R O G R A M     F F T F 2'//1X,
    1'FFTF2 GENERATES A NEW DATA RECORD FILTERED BY FAST FOURIER',
    21X,'TRANSFORM'/1X,
    3'FROM A GIVEN DATA RECORD. WITHIN A RUN A PRACTICALLY',1X,
    4'UNLIMITED'/1X,
    5'NUMBER OF DATA RECORDS CAN BE FILTERED.'//1X,
    6'COMMON PARAMETERS :')
6005 FORMAT(1X,'YOU SPECIFIED :')
6010 FORMAT('$CORRECT ? (Y/N) ')
6015 FORMAT('$VERIFY ! (Y/N) ')
6020 FORMAT(/1X,'RUN DENOTATION ? (MAX.50 CH.)'/1X,
    15('^^^^^^^^^*'))
6025 FORMAT(/1X,'SAMPLING FREQUENCY SFR ? (HZ)'/1X,
    1'^^.^^^D^^^')
6030 FORMAT(/1X,'TRANSFORM SIZE NT ? (32,1024)'/1X,
    1'^^^^')
6035 FORMAT(/1X,'INITIAL NUMBER IDAT OF DATA OUTPUT FILE ? (1-990)'/1X,
    1'^^^')
6040 FORMAT(/1X,'NYQUIST INTERVAL DT (SEC)',11X,1PD12.5/1X,
    1'NYQUIST CUTOFF FREQUENCY CFR (HZ)',3X,D12.5/1X,
    2'NYQUIST CO-INTERVAL DFR (HZ)',8X,D12.5/1X,
    3'TIME LENGTH OF A SEGMENT TS (SEC)',3X,D12.5)
6050 FORMAT(/1X,'INDIVIDUAL PARAMETERS FOR EACH JOB :')
6055 FORMAT(/1X,'YOU CANNOT PROCEED !')
6060 FORMAT(/1X,'YOU CANNOT CONTINUE !')
6200 FORMAT('1JOB',I5,1X,'STARTS !')
6205 FORMAT(/1X,'DATA INPUT FILE FINPUT ? (MAX.50 CH.)'/1X,
    15('^^^^^^^^^*'))
6210 FORMAT(/'$NEW DATA FORMAT ? (Y/N) ')
6215 FORMAT(/1X,'SINGLE-READ DATA FORMAT ?',1X,
```

```
     1'(MAX.20 CH.,E- OR F-SPECIFICATION)'/1X,
     22('^^^^^^^^^^*'))
6220 FORMAT(/1X,'REQUESTED NUMBER NSR OF SEGMENTS ? (MAX.10000)'/1X,
     1'^^^^^')
6225 FORMAT(/1X,'DATA POINT NUMBER NSTART, WHERE EVALUATION',1X,
     1'SHOULD START ?'/1X,'^^^^^^^^')
6230 FORMAT(/1X,'EOF AT DATA POINT NUMBER NEOF',11X,I12)
6235 FORMAT(/1X,'READ-ERROR AT DATA POINT NUMBER NERROR',2X,I12)
6240 FORMAT(/1X,'AVAILABLE NUMBER OF SEGMENTS NSA',8X,I12/1X,
     1'AVAILABLE NUMBER OF DATA POINTS NPA',5X,I12)
6245 FORMAT(/'$NEW CUTOFF FREQUENCIES ? (Y/N) ')
6250 FORMAT(/1X,'LOWER AND UPPER CUTOFF FREQUENCIES FRL,FRH ?'/1X,
     1'(0.LT.FRL.LE.FRH.LE.CFR)'/1X,
     2'^^.^^^D^^^ ^^.^^^D^^^')
6255 FORMAT(1X,'NEAREST POSSIBLE CUTOFF FREQUENCIES FRLP,FRHP',1X,
     1'AND FREQUENCY NUMBERS KL,KH :'/1X,1P,
     2D10.3,1X,D10.3/3X,I4,7X,I4)
6260 FORMAT(/'$NEW MODES FOR DC-SUBTRACTION AND DATA SMOOTHING ?',1X,
     1'(Y/N) ')
6265 FORMAT(/1X,'DC-SUBTRACTION IOPTDC ? (0:NO,1:LOCAL,2:GLOBAL)'/1X,
     1'^')
6270 FORMAT(/1X,'DATA SMOOTHING IOPTSM ? (0:NO,1:COS-WINDOW,',
     1'2:SQUARE-ROOT-WINDOW)'/1X,
     2'^')
6300 FORMAT(/'$NEXT JOB ? (Y/N) ')
C
C     INPUT COMMON PARAMETERS
C
      WRITE(6,6000)
  100 WRITE(6,6020)
      READ(5,'(A)') RUN
      WRITE(6,6005)
      WRITE(6,'(1X,A)') RUN
      WRITE(6,6010)
      READ(5,'(A)') RY
      IF(RY.NE.LY) GO TO 100
  105 WRITE(6,6025)
      READ(5,'(D10.3)',ERR=105) SFR
      WRITE(6,6005)
      WRITE(6,'(1X,1PD10.3)') SFR
      IF(SFR.LE.0.D0) GO TO 105
      WRITE(6,6010)
      READ(5,'(A)') RY
      IF(RY.NE.LY) GO TO 105
  110 WRITE(6,6030)
      READ(5,'(I4)',ERR=110) NT
      WRITE(6,6005)
      WRITE(6,'(1X,I4)') NT
      DO 1 N=5,10
      IF(NT.EQ.2**N) GO TO 120
    1 CONTINUE
      GO TO 110
  120 WRITE(6,6010)
      READ(5,'(A)') RY
      IF(RY.NE.LY) GO TO 110
  115 WRITE(6,6035)
      READ(5,'(I3)',ERR=115) IDAT
```

```
      IF(IDAT.LT.1) IDAT=1
      IF(IDAT.GT.990) IDAT=990
      WRITE(6,6005)
      WRITE(6,'(1X,I3)') IDAT
      WRITE(6,6010)
      READ(5,'(A)') RY
      IF(RY.NE.LY) GO TO 115
      DT = 1.D0/SFR
      CFR = 5.D-1*SFR
      DFR = SFR/DFLOTJ(NT)
      TS = DT*DFLOTJ(NT)
      WRITE(6,6040) DT,CFR,DFR,TS
      OPEN(UNIT=2,FILE=FILE2,STATUS='NEW',DEFAULTFILE='DIRINPUT')
      WRITE(2,2000) FILE1,FILE2,FILE3,FILE4,RUN,SFR,NT,DT,CFR,DFR,
     1TS,IDAT
      CALL DFFTRI(NT,WFFTR)
      OPEN(UNIT=4,FILE=FILE4,STATUS='NEW',FORM='UNFORMATTED',
     1DEFAULTFILE='DIRINPUT',ERR=116)
C
C     INPUT INDIVIDUAL PARAMETERS FOR EACH JOB
C
      WRITE(6,6050)
  200 JOB = JOB+1
      WRITE(6,6200) JOB
      DO 2 N=1,3
  205 WRITE(6,6205)
      READ(5,'(A)') FINPUT
      WRITE(6,6005)
      WRITE(6,'(1X,A)') FINPUT
      WRITE(6,6010)
      READ(5,'(A)') RY
      IF(RY.NE.LY) GO TO 205
      OPEN(UNIT=1,FILE=FINPUT,STATUS='OLD',DEFAULTFILE='DIRINPUT',
     1ERR=2)
      GO TO 215
    2 CONTINUE
      WRITE(6,6055)
      CLOSE(UNIT=4,STATUS='DELETE')
      STOP 1
  215 IF(JOBS.EQ.0) GO TO 220
  210 WRITE(6,6210)
      READ(5,'(A)') RY
      WRITE(6,6015)
      READ(5,'(A)') RYV
      IF(RY.NE.RYV) GO TO 210
      IF(RY.NE.LY) GO TO 225
  220 WRITE(6,6215)
      READ(5,'(A)') IFORM
      WRITE(6,6005)
      WRITE(6,'(1X,A)') IFORM
      WRITE(6,6010)
      READ(5,'(A)') RY
      IF(RY.NE.LY) GO TO 220
  225 WRITE(6,6220)
      READ(5,'(I5)',ERR=225) NSR
      IF(NSR.LT.2) NSR=2
      IF(NSR.GT.10000) NSR=10000
      WRITE(6,6005)
```

```
      WRITE(6,'(1X,I5)') NSR
      WRITE(6,6010)
      READ(5,'(A)') RY
      IF(RY.NE.LY) GO TO 225
  230 WRITE(6,6225)
      READ(5,'(I7)',ERR=230) NSTART
      IF(NSTART.LT.1) NSTART=1
      IF(NSTART.GT.NT*(NSR-1)) NSTART=NT*(NSR-1)
      WRITE(6,6005)
      WRITE(6,'(1X,I7)') NSTART
      WRITE(6,6010)
      READ(5,'(A)') RY
      IF(RY.NE.LY) GO TO 230
      WRITE(2,2005) JOB,FINPUT,IFORM,NSR,NSTART
      REWIND(UNIT=4)
      IF(NSTART.EQ.1) GO TO 235
      DO 3 N=1,NSTART-1
    3 READ(1,IFORM,ERR=240,END=241) XSS
      GO TO 235
  240 NERROR = N
  245 WRITE(6,6235) NERROR
      WRITE(2,2020) NERROR
      WRITE(6,6055)
      CLOSE(UNIT=4,STATUS='DELETE')
      STOP 2
  241 NSA = 0
      NPA = 0
      NEOF = N
      GO TO 255
  235 DO 4 NS=1,NSR
      DO 4 N=1,NT
      READ(1,IFORM,ERR=250,END=251) XSS
      X1 = DBLE(XSS)
    4 WRITE(4) X1
      NSA = NSR
      NPA = NT*NSA
      GO TO 260
  250 NERROR = NT*(NS-1)+N
      GO TO 245
  251 NSA = NS-1
      NPA = NT*NSA
      NEOF = NPA+N
  255 WRITE(6,6230) NEOF
      WRITE(2,2010) NEOF
  260 WRITE(6,6240) NSA,NPA
      WRITE(2,2015) NSA,NPA
      IF(NSA.LT.2) GO TO 265
      IF(JOBS.EQ.0) GO TO 270
  275 WRITE(6,6245)
      READ(5,'(A)') RY
      WRITE(6,6015)
      READ(5,'(A)') RYV
      IF(RY.NE.RYV) GO TO 275
      IF(RY.NE.LY) GO TO 285
      GO TO 270
  265 WRITE(6,6055)
      GO TO 335
```

```
  270 WRITE(6,6250)
      READ(5,'(D10.3,1X,D10.3)',ERR=270) FRL,FRH
      IF(FRL.LT.0.D0) FRL=0.D0
      IF(FRL.GT.CFR) FRL=CFR
      IF(FRH.GT.CFR) FRH=CFR
      IF(FRH.LT.FRL) FRH=FRL
      WRITE(6,6005)
      WRITE(6,'(1X,1PD10.3,1X,D10.3)') FRL,FRH
      KL = JIDNNT(FRL/DFR)
      KH = JIDNNT(FRH/DFR)
      FRLP = DFR*DFLOTJ(KL)
      FRHP = DFR*DFLOTJ(KH)
      WRITE(6,6255) FRLP,FRHP,KL,KH
      WRITE(6,6010)
      READ(5,'(A)') RY
      IF(RY.NE.LY) GO TO 270
      KL1 = KL+1
      KH1 = KH+1
  285 WRITE(2,2025) FRL,FRLP,KL,FRH,FRHP,KH
      IF(JOBS.EQ.0) GO TO 300
  295 WRITE(6,6260)
      READ(5,'(A)') RY
      WRITE(6,6015)
      READ(5,'(A)') RYV
      IF(RY.NE.RYV) GO TO 295
      IF(RY.NE.LY) GO TO 310
  300 WRITE(6,6265)
      READ(5,'(I1)',ERR=300) IOPTDC
      IF(IOPTDC.LT.0) IOPTDC=0
      IF(IOPTDC.GT.2) IOPTDC=2
      WRITE(6,6005)
      WRITE(6,'(1X,I1)') IOPTDC
      WRITE(6,6010)
      READ(5,'(A)') RY
      IF(RY.NE.LY) GO TO 300
  305 WRITE(6,6270)
      READ(5,'(I1)',ERR=305) IOPTSM
      IF(IOPTSM.LT.0.OR.NSA.LT.2) IOPTSM=0
      IF(IOPTSM.GT.2) IOPTSM=2
      WRITE(6,6005)
      WRITE(6,'(1X,I1)') IOPTSM
      WRITE(6,6010)
      READ(5,'(A)') RY
      IF(RY.NE.LY) GO TO 305
  310 WRITE(2,2030) IOPTDC,IOPTSM
C
C     CALCULATIONS AND OUTPUT
C
      ENCODE(3,'(I3)',FILE3(7:9)) IDAT
      OPEN(UNIT=3,FILE=FILE3,STATUS='NEW',DEFAULTFILE='DIRINPUT',
     1ERR=311)
      REWIND(UNIT=4)
      XMEAN = 0.D0
      IF(IOPTDC.LT.2) GO TO 315
      DO 5 N=1,NPA
      READ(4) X1
    5 CALL MEAN(N,X1,XMEAN)
      WRITE(2,2035) XMEAN
```

```
      REWIND(UNIT=4)
  315 DO 10 NS=1,NSA
      DO 6 N=1,NT
    6 READ(4) X(N)
      IF(IOPTDC.NE.1) GO TO 320
      XMEAN = 0.D0
      DO 11 N=1,NT
   11 CALL MEAN(N,X(N),XMEAN)
  320 DO 12 N=1,NT
   12 X(N) = X(N)-XMEAN
      CALL DF2TRF(NT,X,COEF,WFFTR)
      CALL FILTER(NT,KL1,KH1,COEF)
      CALL DF2TRB(NT,COEF,X,WFFTR)
      IC = NS-1
      IF(IOPTSM-1) 325,326,327
  326 CALL SMOOTH(NT,IC,WINDOWA,X,XS)
      GO TO 330
  327 CALL SMOOTH(NT,IC,WINDOWB,X,XS)
  330 IF(IC.EQ.0) GO TO 10
      IC = IC-1
  325 I1 = NT*IC
      I2 = I1+NSTART-1
      DO 13 N=1,NT
   13 WRITE(3,3000) N+I1,SNGL(DFLOTJ(N+I1-1)*DT),
     1N+I2,SNGL(DFLOTJ(N+I2-1)*DT),SNGL(X(N))
   10 CONTINUE
      CLOSE(UNIT=3,STATUS='KEEP')
  335 CLOSE(UNIT=1,STATUS='KEEP')
  400 WRITE(6,6300)
      READ(5,'(A)') RY
      WRITE(6,6015)
      READ(5,'(A)') RYV
      IF(RY.NE.RYV) GO TO 400
      IF(RY.NE.LY) GO TO 405
      JOBS = JOB
      IDAT = IDAT+1
      IF(IDAT.LE.999) GO TO 200
      WRITE(6,6060)
  405 WRITE(2,2040)
      CLOSE(UNIT=4,STATUS='DELETE')
      STOP
  116 WRITE(6,6055)
      STOP 4
  311 WRITE(6,6055)
      STOP 3
      END
C========================================================
      SUBROUTINE MEAN(NEW,XNEW,XMEAN)
C
C     THIS SUBROUTINE SERVES FOR CALCULATING THE ARITHMENTIC
C     MEAN VALUE OF THE INPUT DATA.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      A = DFLOTJ(NEW)
      B = A-1.D0
      XMEAN = (B/A)*XMEAN+XNEW/A
      RETURN
```

```
      END
C=========================================================
      SUBROUTINE FILTER(NT,KL1,KH1,COEF)
C
C     THIS SUBROUTINE PERFORMS SPECTRAL FILTERING.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION COEF(1)
      NTH = NT/2
      IF(1.LT.KL1) COEF(1)=0.D0
      IF(KH1.LT.NTH+1) COEF(NT)=0.D0
      DO 1 K=2,NTH
      IF(K.GE.KL1.AND.K.LE.KH1) GO TO 1
      COEF(2*(K-1)) = 0.D0
      COEF(2*K-1) = 0.D0
    1 CONTINUE
      DO 2 K=1,NT
    2 COEF(K) = COEF(K)/DFLOTJ(NT)
      RETURN
      END
C=========================================================
      SUBROUTINE SMOOTH(NT,IC,WSUB,X,XS)
C
C     A TWO-SEQUENCE WINDOW IS APPLIED TO THE FILTERED DATA.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION X(1),XS(1)
      IF(IC.GT.0) GO TO 100
      DO 1 N=1,NT
    1 XS(N) = X(N)
      RETURN
  100 DO 2 N=1,NT
      CALL WSUB(N,NT,W1,W2)
      X1 = XS(N)
      X2 = X(N)
      XS(N) = X2
    2 X(N) = X1*W1+X2*W2
      RETURN
      END
C=========================================================
      SUBROUTINE WINDOWA(N,NT,W1,W2)
C
C     TWO-SEQUENCE COSINE WINDOW.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      SAVE PIH
      DATA PIH/1.57079633D0/
      W1 = PIH*DFLOTJ(N-1)/DFLOTJ(NT)
      W2 = DSIN(W1)
      W1 = DCOS(W1)
      RETURN
      END
C=========================================================
      SUBROUTINE WINDOWB(N,NT,W1,W2)
C
C     TWO-SEQUENCE SQUARE-ROOT WINDOW.
C
      IMPLICIT REAL*8 (A-H,O-Z)
```

```
W1 = DFLOTJ(N-1)/DFLOTJ(NT)
W2 = DSQRT(W1)
W1 = DSQRT(1.D0-W1)
RETURN
END
```

## 6. PROGRAM DTBSP2

Precision : double

Operation : interactive

Required auxiliary routines : from IMSL : DBSLSQ, DBSVAL

Purpose of the program :

Detrending of Non-Stationary Noise Data by Spline Techniques.

Feature Summary :

- Separation of random noise data by a characteristic cutoff frequency into a part with low frequency components (trend signal data) and into a part with high frequency components (detrended signal data).
- Both signal parts are available.
- Least-squares spline approximation of the noise data with equally spaced breakpoints.
- The cutoff frequency is determined by the breakpoint distance.
- The steepness of the cutoff is controlled by the spline order.
- Band-pass filtering of stationary noise data by a twice application of the code. Specially suitable for filter cutoff frequencies which are small against the Nyquist cutoff frequency.
- Within a run a practicallly unlimited number of signal data files, each case called a job, can be treated.

### 6.1 Mathematical Background

The filtering method developed by Behringer (1989/1, 1989/2, 1998) is a simple a-posteriori procedure with equally spaced breakpoints in a least-squares spline approximation. All data in the data set selected from a given noise record are simultaneously included in the fit procedure. The method is mainly applicable to relatively short data sets. The maximum data set length allowed depends on the available memory size of the computer. If $x_n$, n=1,2,... are the data points in the data set of the original signal, and $\hat{x}_n$ are the smoothed data points from the spline approximation which exhibits the approximated trend, the residual or detrended data points $\hat{x}_n$ follow simply from

$$\hat{x}_n = x_n - \overset{\frown}{x}_n$$

The method makes available both signal parts.

The spline function is represented in the form of B-splines (Schumaker, 1981). For the least-squares B-spline approximation to the given signal data set, the IMSL routine DBSLSQ is used. The B-spline approximation is evaluated at each data point of the data set, using the IMSL routine DBSVAL. All data points to be processed are assumed to have equal weight (=1). The point numbers, starting with number 1, are internally handled as real quantities. The determination of the knot sequence is based on an equidistant segmentation of the data set with adjacent segments. The segment length (number of data points in a segment) is an input parameter. The last segment must be completely filled with signal data. The boundaries of the segments are midpoints on the data abscissa and are chosen as breakpoints. A maximum possible smoothness of the spline function across each of the interior breakpoints has been adopted. If K is the spline order (K=4 for a cubic spline), then all derivatives up to and including the (K-2)th derivative are continuous across each interior breakpoint.

There are two control parameters which affect the filtering action. These are the segment length (breakpoint distance) and the spline order. They have been found to be independent of each other.

The segment length determines the low frequency cutoff in the detrended signal. The spline approximation acts as a low-pass filter for obtaining the trend signal. The bandwidth increases with decreasing segment length. The method acts as a high-pass filter to separate the trend from the noise. The low frequency cutoff is shifted upwards with decreasing segment length. Considering that the spline approximation can follow frequency components with periods larger than the double segment length, one can define a low cutoff frequency point $f_c$ for the signal to be detrended by

$$f_c = \frac{1}{2N_s\Delta t}$$

where $N_s$ ($\geq 2$) is the number of data points in a segment, and $\Delta t$ is the sampling interval. The upper value of $N_s$ is allowed to correspond to the length of the data set (no interior breakpoint). The equation for $f_c$ gives values which seem to be closely related to the 3 db cutoff frequency point used in linear filter theory. The cutoff is very sharp. There is no distortion by any undershoot or overshoot in the vicinity of $f_c$. The filter has zero-phase shift. The method is especially suitable for chosing $f_c$ values which are small against the Nyquist cutoff frequency. The highest possible value of $f_c$ is given by the half of the Nyquist cutoff frequency.

An increase of the spline order makes the filter steeper. The steepness has been defined in db units simply by

$$\text{Steepness} = 10\log(\text{PSD}(0.5f_c)/\text{PSD}(f_c))$$

Where PSD denotes the power spectral density with respect to white noise. A plot of the dependence of the steepness on the spline order is given in the cited papers by Behringer. E.g. a spline order of K=10 shows a steepness of about –90 db/octave.

The method can also be applied to band-pass filtering of stationary noise data. But in the present version, the code must be run twice (for reasons of the data input format). If $f_L$ is the lower cutoff frequency and $f_H$ is the upper cutoff frequency, one can set $f_c \cong f_L$ in the first run. The detrended signal part is then used with $f_c \cong f_H$ in the second run. The trend signal part resulting from this second run is then the desired band-pass filtered signal. This procedure has alternatively been used for testing the effectiveness of the FFT filter (code FFTF2) with K=8. Auto-correlation functions estimated with the code ACCF1 on signal data previously filtered by both methods under equivalent parametric conditions showed practically identical results.

## 6.2 References

Behringer K.(1989/1). The Code DTBSP1 for Detrending Non-Stationary Noise Data by Spline Techniques, Internal PSI Report TM-41-89-05.

Behringer K.(1989/2). PSI Report 50, Paper presented at the 21th Informal Meeting on Reactor Noise (IMORN 21), Villigen PSI, September 20-22.

Behringer K.(1998). Ann.Nucl.Energy 25,889.

Schumaker L.L.(1981). Spline Functions : Basic Theory, J.Wiley & Sons, New Jork.

### 6.3 Files

Within a run, an unlimited number of noise records can be processed from different files and with different segmentation. Only the specified data input format is common for all jobs. The data resulting from the trend analysis are output to files which are separate for each job.

There are 3 files :

- 'FINPUT'

This file contains the original signal data. 'FINPUT' is a formal parameter. The file name must be interactively specified for each job. The code assumes that the signal data are of the REAL*4 type and are written as column vector.

- DTBSP2.PRT

This file is meant for printing. It contains (with text) all input parameter data, parameter data modified during processing, and additional data and messages from the computation progress.

- DTBSP20'''.DAT

This file contains the original signal data which are to be processed, and the processed data. For each job a separate file with a current number of up to 3 digits is opened. The initial number for the first job must be interactively input. There are 5 columns with the line format (1X,I<NDATAI>,I8,1P,3E14.6). In the present version NDATAI=4 (see line 33 in the listing).

column

1 : current data point number of the selected data set, starting with number 1,

2 : corresponding data point number from the begin of the original signal record,

3 : FDATA, selected original signal data,

4 : FD, data of the spline approximation (trend signal data),

5 : DFD=FDATA–FD, data of the residual signal (detrended signal data).

A time abscissa is not given. It would require the input of the sampling frequency value. For plotting, the data point numbers can be used and converted into time units by the graphical code.

### 6.4 Parameter Data Specified in the Code (line 33 in the listing)

- NDATAM : Max. even number of data points to be processed.
- NDATAI : Number of digits of NDATAM (variable format).
- KORDER : Spline order (for cubic spline KORDER=4), recommended values between 4 and 10.
- NCOEFM : Max. number of B-spline coefficients, NCOEFM=KORDER+NDATAM/2-1 .

### 6.5 Interactive Parameter Data Input

The interactive procedure for the input of the parameter data with text and format indications on the terminal screen is very similar to those of the codes CPSDES3 and FFTF2. Each data typed in is immediately rewritten to the screen. It can be corrected if necessary, and must be verified for final acceptance. In a few cases, erroneous data are rejected and must be corrected. There is no protection, if in a number field an unallowed character is typed in. In this case the code stops.

Common Parameter Data Input :

- format (A), RUN

A string of max. 40 characters for run identification.

- format (I3), IDAT

Initial number of the data output file for the first job. For each further job within a run, this number is incremented by 1 .

Internal restrictions : IF(IDAT.LT.1) IDAT=1 (default value)

IF(IDAT.GT.990) IDAT=990

If a run starts with the highest allowed number, the code assumes that the run does not encompasses more than 10 jobs, otherwise the run will normally terminate after job 10 with a preceding message. IDAT allows a grouping of the numbers of the data output files, if sets of signal records are to be processed by more than one run.

- format (A), IFORM

A string of max. 20 characters for specifying the single-read signal data format. The format must be given in E- or F-specification including the brackets.

Individual Parameter Data Input :

- format (A), FINPUT

A string of max. 40 characters for the file name of the signal data to be processed. If after a third trial this file cannot be opened, the code stops with a message.

- format (I6), NSTART

The number specifies the first signal data point where processing starts. If it is set > 1, then NSTART-1 initial data points are passed over.

Internal restriction : IF(NSTART.LT.1) NSTART=1 (default value)

- format (I<NDATAI>), ISEG

Number of requested segments.

Internal restrictions : IF(ISEG.LT.1) ISEG=1 (default value)

Further conditions see Section 6.6

The number of interior breakpoints is ISEG-1. If the signal record (or the data set selected by NSTART) should be scanned until its end, a sufficiently high number can be given, provided that the maximum allowed data set size (NDATAM, line 33 in the listing) is not exceeded. The code checks on the EOF mark and reduces the number given for ISEG to the maximum available number of completely filled segments. The same procedure applies to a detected read-error. In both cases, messages are sent to the terminal screen and the print file, indicating the type of action and the number of the erroneous data point. This data point number (NERROR) refers to the begin of the signal record.

- format (I<NDATAI>), NPSEG

Number of data points in a segment (segment length).

Internal restrictions : IF(NPSEG.LT.2) NPSEG=2 (default value)

Further conditions see Section 6.6

The size of the data set to be processed is defined by PNSEG*ISEG.

## 6.6 Branchings

After a job has terminated, there will be a question concerning continuing with the next job. The question must be acknowledged with YES or NOT. If there is another job, the code then asks for the individual input parameter data of the next job. There are two conditions for successful data processing. They concern the values of ISEG and NPSEG. The maximum allowed data set size is determined by the value attributed to the parameter NDATAM. It must be :

Condition a : NPSEG*ISEG≤NDATAM

The spline order is fixed by the value attributed to the parameter KORDER. It must be :

Condition b: KORDER+ISEG-1≤NPSEG*ISEG

The code checks on both conditions. If at the beginning of a started job, during the period of the individual parameter data input, either the one or the other condition is violated, the code simply returns and asks for other values of ISEG and/or NPSEG. A message is sent to the terminal screen. If, during the period of reading the signal data, the EOF mark is reached or a read-error is detected, the originally given (and accepted) value for ISEG is automatically reduced to its possible value. The violation of condition b leads to a message beeing sent to the terminal and the print file that this job cannot be processed. In this case, the previously opened data output file is deleted and the job terminates.

## 6.7 Numbered Stops

- STOP 100 : The print file cannot be opened.
- STOP 101 : After 3 allowed trials, the file FINPUT cannot be opened.
- STOP 102 : The data output file cannot be opened.

# LISTING

## DTBSP2

```
      PROGRAM DTBSP2
C
C     THE CODE DETRENDS A DIGITAL NOISE RECORD BY APPLYING A LEAST
C     SQUARES B-SPLINE APPROXIMATION WITH EQUIDISTANT KNOTS. THE
C     PROCEDURE IS BASED ON THE TWO IMSL ROUTINES DBSLSQ AND DBSVAL.
C     DOUBLE PRECISION VERSION OF DTBSP1 (INTERNAL CALCULATIONS ONLY).
C
C     THE SIGNAL DATA ARE ASSUMED TO BE FORMATTED (REAL*4 TYPE).
C     THE OUTPUT DATA ARE GIVEN IN SINGLE PRECISION.
C
C     CODE WRITTEN BY K.BEHRINGER, MAY, 1998.
C
C     INTERACTIVE VERSION FOR THE PSI VAX COMPUTER SYSTEM.
C
C     LINK COMMAND : IMPORT IMSL
C                    LINK DTBSP2
C
C
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*4 SY
      CHARACTER RUN*40,FINPUT*40,FPRINT*20,FOUTPUT*20,IFORM*20,LY,RY
C
C     ADJUSTABLE PARAMETER DATA :
C
C     NDATAM : MAX.EVEN NUMBER OF DATA POINTS TO BE PROCESSED
C              (>KORDER).
C     NDATAI : NUMBER OF DIGITS OF NDATAM (VARIABLE FORMAT).
C     KORDER : SPLINE ORDER (FOR CUBIC SPLINE : K=4).
C     NCOEFM : MAX.NUMBER OF B-SPLINE COEFFICIENTS.
C              SINCE THE MIN.NUMBER OF DATA POINTS IN A SEGMENT
C              HAS BEEN LIMITED TO 2, NCOEFM = KORDER+NDATAM/2-1 .
C
      PARAMETER (NDATAM=6000,NDATAI=4,KORDER=8,NCOEFM=3007)
C
      DIMENSION XDATA(NDATAM),FDATA(NDATAM),WEIGHT(NDATAM),
     1XKNOT(NCOEFM+KORDER),BSCOEF(NCOEFM),WK1((3+NCOEFM)*KORDER),
     2WK2(NDATAM),WK3(NDATAM),WK4(NDATAM),IWK(NDATAM)
      DATA JOB,LY,WEIGHT,EPS/0,'Y',NDATAM*1.D0,5.D-1/,
     1FPRINT /'DTBSP2.PRT          '/,
     2FOUTPUT/'DTBSP20000.DAT      '/
C
C     FORMATS
C
 2000 FORMAT('1',40X,'P R O G R A M     D T B S P 2',11X,
     1'(VERSION FOR VAX COMPUTER)'////11X,'PARAMETER INPUT',4X,
     2': INTERACTIVE'//1X,'FILES :',3X,'SIGNAL DATA INPUT',2X,
     3': FILE ''FINPUT'''/11X,'PRINT OUTPUT',7X,
     4': FILE ''FPRINT'' (DTBSP2.PRT)'/11X,'SIGNAL DATA OUTPUT',1X,
     5': FILE ''FOUTPUT'' (DTBSP20''''''.DAT)'///1X,
     6'RUN DENOTATION',26X,'RUN',4X,'=',2X,A//1X,
     7'SPLINE ORDER',28X,'KORDER =',I5//1X,
     8'INITIAL NUMBER OF FOUTPUT',15X,'IDAT',3X,'=',I5//1X,
```

```
     9'DATA INPUT FORMAT',23X,'IFORM',2X,'=',2X,A)
2001 FORMAT('1JOB NUMBER',40X,'JOB',5X,'=',I11///1X,
     1'DATA INPUT FILE',35X,'FINPUT',2X,'=',1X,A//1X,
     2'NUMBER OF DATA POINT WHERE EVALUATION STARTS',6X,'NSTART',2X,
     3'=',I11//1X,'NUMBER OF REQUESTED SEGMENTS',22X,'ISEG',4X,'=',I11
     4//1X,'NUMBER OF DATA POINTS IN A SEGMENT',16X,'NPSEG',3X,'=',I11
     5//1X,'NUMBER OF REQUESTED DATA POINTS TO BE PROCESSED',3X,'NDATA',
     63X,'=',I11//1X,'NUMBER OF REQUESTED B-SPLINE COEFFICIENTS',9X,
     7'NCOEF',3X,'=',I11)
2002 FORMAT(//1X,'NUMBER OF AVAILABLE SEGMENTS',22X,'ISEG',4X,'=',I11,
     1//1X,'NUMBER OF AVAILABLE DATA POINTS TO BE PROCESSED',3X,'NDATA',
     23X,'=',I11//1X,'NUMBER OF B-SPLINE COEFFICIENTS',19X,'NCOEF',3X,
     3'=',I11)
2003 FORMAT(/1X,'DATA OUTPUT FILE',34X,'FOUTPUT =',1X,A)
2004 FORMAT(////1X,'E N D')
5000 FORMAT(A)
5001 FORMAT(I3)
5002 FORMAT(I6)
5003 FORMAT(I<NDATAI>)
6000 FORMAT('1PROGRAM DTBSP2',11X,'(VERSION FOR VAX COMPUTER)'//1X,
     1'THE CODE DETRENDS A DIGITAL NOISE RECORD'/1X,
     2'BY APPLYIMG A LEAST SQUARES B-SPLINE APPROXIMATION.'//1X,
     3'IT HAS BEEN WRITTEN FOR INTERACTIVE OPERATION.'/1X,
     4'FOLLOW THE INSTRUCTION RULES FOR THE PARAMETER DATA INPUT !'/)
6001 FORMAT(/1X,'RUN DENOTATION ? (MAX.40 CH.)'/1X,4('^^^^^^^^^*'))
6002 FORMAT(1X,'YOU SPECIFIED :'/1X,A)
6003 FORMAT('$CORRECT ? (Y/N) ')
6004 FORMAT(/1X,'INITIAL NUMBER IDAT OF THE DATA FILE OUTPUT ?',1X,
     1'(I3,1-990)'/1X,'^^^')
6005 FORMAT(1X,'YOU SPECIFIED :'/1X,I3)
6006 FORMAT(1X,'SIGNAL DATA INPUT FORMAT IFORM ?',1X,
     1'(MAX.20 CH., INCL.(...) )'/1X,
     2'(SINGLE-READ FORMAT IN E- OR F-SPECIFICATION)'/1X,
     32('^^^^^^^^^*'))
6007 FORMAT(//1X,'JOB NR.',I5,2X,'STARTS !'/)
6008 FORMAT(/1X,'SIGNAL DATA FILE FINPUT ? (MAX.40 CH.)'/1X,
     14('^^^^^^^^^*'))
6009 FORMAT(1X,'OPEN-ERROR, REPEAT !')
6010 FORMAT(/1X,'DATA POINT NUMBER NSTART WHERE SEGMENTATION STARTS ?',
     11X,'(I6,(>0))'/1X,6('^'))
6011 FORMAT(1X,'YOU SPECIFIED :'/1X,I6)
6012 FORMAT(/1X,'NUMBER OF SEGMENTS ISEG ? (I*,(>0))'/1X,
     1<NDATAI>('^'))
6013 FORMAT(1X,'YOU SPECIFIED :'/1X,I<NDATAI>)
6014 FORMAT(/1X,'NUMBER OF DATA POINTS NPSEG IN A SEGMENT ?',
     11X,'(I*,(>1))'/1X,<NDATAI>('^'))
6015 FORMAT(/1X,'NDATA ='I<2*NDATAI+2>/1X,
     1'THE VALUE EXCEEDS THE ALLOWED RANGE OF ',I<NDATAI>,1X,'.')
6016 FORMAT(/1X,'NDATA = ',I<NDATAI+1>/1X,'NCOEF = ',I<NDATAI+1>/1X,
     1'THE NUMBER OF B-SPLINE COEFFICIENTS'/1X,
     2'EXCEEDS THE NUMBER OF DATA POINTS.')
6017 FORMAT(1X,'GIVE OTHER VALUES FOR ISEG OR/AND NPSEG !')
6018 FORMAT(/1X,'READ-ERROR AT INPUT DATA POINT NERROR =',I12)
6019 FORMAT(/1X,'EOF MARK   AT INPUT DATA POINT NERROR =',I12)
6020 FORMAT(/1X,'NUMBER OF AVAILABLE SEGMENTS     ISEG  = ',I<NDATAI>
     1/1X,       'NUMBER OF AVAILABLE DATA POINTS NDATA = ',I<NDATAI>
     2/1X,       'NUMBER OF B-SPLINE COEFFICIENTS NCOEF = ',I<NDATAI>)
6021 FORMAT(/1X,'JOB CANNOT BE PROCESSED !')
```

```
 6022 FORMAT(/'$NEXT JOB ? (Y/N) ')
 6023 FORMAT('$VERIFY ! (Y/N) ')
 6024 FORMAT(/1X,'NUMBER OF FOUTPUT IDAT=1000, YOU CANNOT CONTINUE !')
 7000 FORMAT(1X,I<NDATAI>,I8,1P,3E14.6)
C
C      INPUT OF COMMON PARAMETER DATA
C
      WRITE(6,6000)
      OPEN(UNIT=2,FILE=FPRINT,STATUS='NEW',ERR=100)
  110 WRITE(6,6001)
      READ(5,5000) RUN
      WRITE(6,6002) RUN
      WRITE(6,6003)
      READ(5,5000) RY
      IF(RY.NE.LY) GO TO 110
  120 WRITE(6,6004)
      READ(5,5001) IDAT
      IF(IDAT.LT.1) IDAT=1
      IF(IDAT.GT.990) IDAT=990
      WRITE(6,6005) IDAT
      WRITE(6,6003)
      READ(5,5000) RY
      IF(RY.NE.LY) GO TO 120
  130 WRITE(6,6006)
      READ(5,5000) IFORM
      WRITE(6,6002) IFORM
      WRITE(6,6003)
      READ(5,5000) RY
      IF(RY.NE.LY) GO TO 130
      DO 1 N=1,NDATAM
    1 XDATA(N) = DFLOTJ(N)
      WRITE(2,2000) RUN,KORDER,IDAT,IFORM
C
C      INPUT OF INDIVIDUAL PARAMETER DATA
C
  200 JOB = JOB+1
      WRITE(6,6007) JOB
      DO 2 N=1,3
  210 WRITE(6,6008)
      READ(5,5000) FINPUT
      WRITE(6,6002) FINPUT
      WRITE(6,6003)
      READ(5,5000) RY
      IF(RY.NE.LY) GO TO 210
      OPEN(UNIT=1,FILE=FINPUT,STATUS='OLD',ERR=220)
      GO TO 230
  220 WRITE(6,6009)
    2 CONTINUE
      STOP 101
  230 WRITE(6,6010)
      READ(5,5002) NSTART
      IF(NSTART.LT.1) NSTART=1
      WRITE(6,6011) NSTART
      WRITE(6,6003)
      READ(5,5000) RY
      IF(RY.NE.LY) GO TO 230
  240 WRITE(6,6012)
```

```
      READ(5,5003) ISEG
      IF(ISEG.LT.1) ISEG=1
      WRITE(6,6013) ISEG
      WRITE(6,6003)
      READ(5,5000) RY
      IF(RY.NE.LY) GO TO 240
  250 WRITE(6,6014)
      READ(5,5003) NPSEG
      IF(NPSEG.LT.2) NPSEG=2
      WRITE(6,6013) NPSEG
      WRITE(6,6003)
      READ(5,5000) RY
      IF(RY.NE.LY) GO TO 250
      NDATA = NPSEG*ISEG
      IF(NDATA.GT.NDATAM) GO TO 260
      NCOEF = KORDER+ISEG-1
      IF(NCOEF.GT.NDATA) GO TO 270
      WRITE(2,2001) JOB,FINPUT,NSTART,ISEG,NPSEG,NDATA,NCOEF
      ENCODE(3,5001,FOUTPUT(8:10)) IDAT
      OPEN(UNIT=7,FILE=FOUTPUT,STATUS='NEW',ERR=290)
      IDAT = IDAT+1
C
C     INPUT OF SIGNAL DATA
C     COMPUTATION OF THE KNOT SEQUENCE
C
      I = -1
      IF(NSTART.EQ.1) GO TO 300
      DO 3 N=1,NSTART-1
    3 READ(1,IFORM,ERR=310,END=320) SY
  300 X0 = XDATA(1)-EPS
      DO 4 N=1,KORDER
    4 XKNOT(N) = X0
      DO 5 I=1,ISEG
      DO 6 N=1,NPSEG
      READ(1,IFORM,ERR=330,END=340) SY
    6 WK2(N) = DBLE(SY)
      N0 = NPSEG*(I-1)
      DO 7 N=1,NPSEG
    7 FDATA(N+N0) = WK2(N)
    5 XKNOT(I+KORDER) = XDATA(N0+NPSEG)+EPS
      I = ISEG
      GO TO 400
  330 I = I-1
      GO TO 310
  340 I = I-1
      GO TO 320
  310 ASSIGN 6018 TO N0
      GO TO 350
  320 ASSIGN 6019 TO N0
  350 NERROR = N+IABS(I.GE.0)*(NSTART-1+NPSEG*I)
      WRITE(6,N0) NERROR
      WRITE(2,N0) NERROR
      IF(I.LT.0) I=0
      NDATA = NPSEG*I
      NCOEF = KORDER+I-1
  400 WRITE(6,6020) I,NDATA,NCOEF
      WRITE(2,2002) I,NDATA,NCOEF
      IF(NCOEF.GT.NDATA) GO TO 410
```

```
      N0 = KORDER+I
      X0 = XDATA(NDATA)+EPS
      DO 8 N=1,KORDER-1
    8 XKNOT(N+N0) = X0
      WRITE(2,2003) FOUTPUT
C
C     OUTPUT OF KNOT SEQUENCE
C
C     WRITE(2,2010)  (N,XKNOT(N),N=1,NCOEF+KORDER)
C2010 FORMAT('1',1P,<NDATA>(/1X,I<NDATAI>,D12.5))
C
C
C     COMPUTATION OF THE LEAST SQUARES SPLINE APPROXIMATION
C
      CALL DB2LSQ(NDATA,XDATA,FDATA,WEIGHT,KORDER,XKNOT,NCOEF,
     1BSCOEF,WK1,WK2,WK3,WK4,IWK)
      NSTART = NSTART-1
      DO 9 N=1,NDATA
      FD = DBSVAL(XDATA(N),KORDER,XKNOT,NCOEF,BSCOEF)
      DFD = FDATA(N)-FD
    9 WRITE(7,7000) N,N+NSTART,SNGL(FDATA(N)),SNGL(FD),SNGL(DFD)
      CLOSE(UNIT=7,STATUS='KEEP')
      GO TO 420
C
C     NEXT JOB ?
C
  410 WRITE(6,6021)
      WRITE(2,6021)
      CLOSE(UNIT=7,STATUS='DELETE')
  420 CLOSE(UNIT=1,STATUS='KEEP')
  430 WRITE(6,6022)
      READ(5,5000) RY
      WRITE(6,6023)
      READ(5,5000) LY
      IF(RY.NE.LY) GO TO 430
      LY = 'Y'
      IF(RY.NE.LY) GO TO 440
      IF(IDAT.LT.1000) GO TO 200
      WRITE(6,6024)
  440 WRITE(2,2004)
      STOP
C
  100 STOP 100
  290 STOP 102
  260 WRITE(6,6015) NDATA,NDATAM
      GO TO 280
  270 WRITE(6,6016) NDATA,NCOEF
  280 WRITE(6,6017)
      GO TO 240
      END
```

# 7. PROGRAM ACCF1

Precision : single

Operation : foreground, background

Required auxiliary routines : from IMSL : F2TRF, F2TRB, FFTRI

Purpose of the program :

Univariate and Bivariate Correlation Analysis of Stationary Noise Data in the Time Domain

Feature Summary :

- Indirect estimation procedure by FFT techniques of the
  - auto-correlation functions (ACFs) in channel A and B,
  - cross-correlation function (CCF).
- Especially suitable for the estimation of the initial parts of the ACFs and the CCF.
- Modified averaging method by signal segmentation.
- 2 options available for the DC component removal.
- Process alignment available for transport time analysis.
- Within a run, a practically unlimited number of signal pairs from different files, each case called a job, can be processed.

## 7.1 Mathematical Background

In the direct ACF estimation procedure, an unbiased estimate, assuming $N_s$ data samples, is obtained by

$$R_{xx}(n) = \frac{1}{N_s - n} \sum_{m=0}^{N_s-n-1} x(n)x(n+m); 0 \le n \le N_s - 1$$

The statistical accuracy of the estimate decreases with increasing lag values, simply because the number of signal data products in averaging decreases. To be on the safe side, one should restrict the use of an ACF estimate to about 50 % of the maximum available lag value. This consideration also holds for the following indirect estimation method.

The estimation procedure realized in the code ACCF1, is based on the indirect method with FFT techniques and zero-padding. It is described in the textbook by Bendat and Piersol (1971). It is not suitable for very large values of $N_s$, which must be a power of 2 in the indirect method, because this would require a large FFT transform size. The FFT transform size $N_T$ which has to be applied, is $2N_s$.

The indirect estimation method is especially advantageous for estimating the mostly interesting initial parts of the ACFs and the CCF. However, in order to make full use of the available length of the signal records, the method has been modified by signal segmentation (Behringer, 1988). In each of $N_{av}$ succeeding segments of length $N_s$ (number of data points in a segment) raw instantaneous power spectral densities (PSDs) and the raw instantaneous cross-power spectral density (CPSD) are calculated by the FFT with double segment length. The first part of this double segment is the normal segment with $N_s$ signal data points. The second part is virtual and filled with $N_s$ zero's. The instantaneous PSDs and the instantaneous CPSDs are averaged over the $N_{av}$ succeeding segments at each frequency point. The inverse FFT is then applied to calculating the circular ACF and CCF estimates, from which the true ACF estimates in both channels and the true CCF estimate follow by decomposition. If one gives for $N_{av}$ a very large value, the code scans over all completely filled signal segments which are commonly available in both channels. Within a run, a practically unlimited number of record pairs can be treated. For each signal record a start point number from where analysis begins, must be given. The data output files have identification numbers.

Two options for the DC component removal in each channel are provided, either with a DC value calculated over all segments (global removal) or with DC values calculated for each segment separately (local removal, low frequency trend elimination). With the DC component subtracted from the signal data, the ACFs and the CCF represent then estimates of the auto-covariance functions and the cross-covariance function respectively.

If the code is used for ACF estimations on signal data previously filtered by the code FFTF2, the segments must exactly agree in length and position.

## 7.2 References

Behringer K.(1988). The Code ACCF1 for Bivariate Correlation Analysis in the Time Domain, Internal PSI Report TM-41-88-06.

Bendat J.S.and Piersol A.G.(1971). Random Data : Analysis and Measurement Procedures, Wiley-Interscience, New York.

### 7.3 Files

There are 5 files :

- ACCF.IN

This file contains all input parameter data.

- ACCF.PRT

This file is meant for printing. It contains (with text) all input parameter data, parameter data modified during processing, additional data, like e.g. the signal variances, and messages from the computation progress.

- 'FINPUTA'

This file contains the signal data of channel A.

- 'FINPUTB'

This file contains the signal data of channel B. 'FINPUTA' and 'FINPUTB' are formal parameters. The file names must be specified for each job on the file ACCF.IN. The code assumes that the signal data are of the REAL*4 type, are written as column vector, and have the same format in both channels within a run.

- ACCF0'''.PLO

This file contains the data of both ACFs and the CCF. It can directly be used for plotting the data by a graphical code. There are 5 columns with the line format (1X,I4,1P,4E12.4) :

column

1 : Current integer number from 1 to $2N_s+1$. The value 1 corresponds to the lag number $-N_s$, and the value $2N_s+1$ to the lag number $N_s$. The value $N_s+1$ refers to the zero lag number.

2 : Lag times in sec. from $-N_s\Delta t$ until $+N_s\Delta t$. $\Delta t$ is the sampling interval.

3 : Estimated ACF data in channel A.

4 : Estimated ACF data in channel B.

5 : Estimated CCF data.

The estimated ACFs are completely represented on both sides. Since an ACF is a symmetric function of the lag time, ordinarily the right-hand side is used for plotting.

For each job a separate file with a current number of exactly 3 digits with leading zero's is opened. The initial number for the first job must be specified on the file ACCF.IN.

## 7.4 Parameter Data Input

The parameter data must be written on the file ACCF.IN line by line.

Common Parameter Data :

- line 1, format (A), RUN

A string of max. 50 characters for run identification.

- line 2, format (I4), NS

Number of signal data points in a segment (segment length).

Internal restriction : The value must be a power of 2 within the range of $2^5$ to $2^{10}$; otherwise the code will stop.

- line 3, format (I4), NAV

Requested number of averages or number of succeeding segments to be included in the analysis.

Internal restrictions : IF(NAV.LT.1) NAV=1 (default value)

IF(NAV.GT.1000) NAV=1000

The code checks whether the requested number is available in each channel and for each job or not. If not, the number of completely filled segments beeing available in both channels is taken. This value is output on the file ACCF.PRT.

- line 4, format (E10.3), SFR

Sampling frequency in Hz.

Internal restriction : IF(SFR.LT.0.) run will be aborted.

- line 5, format (I1), IOPTDC

Parameter for signal DC component removal.

IOPTDC = 0 : No removal, recommended option in the case of previously band-pass filtered signal data.

1 : Global DC component removal. The mean computed in each channel over the requested or available number of segments is subtracted.

2 : Local DC component removal. The mean computed in each segment is subtracted in each channel.

Internal restrictions : IF(IOPTDC.LT.0) IOPTDC=0

IF(IOPTDC.GT.2) IOPTDC=2

- line 6, format (I3), IPLOT

Initial number of the file ACCF0'''.PLO for the first job. For each further job within a run this number is incremented by 1.

Internal restrictions : IF(IPLOT.LT.1) IPLOT=1 (default value)

IF(IPLOT.GT.990) IPLOT=990

If the run starts with the highest allowed number, the code assumes that the run does not encompasses more than 10 jobs; otherwise the run will terminate after job 10 with a message to the print file. IPLOT allows a grouping of the numbers of the plot files, if sets of signal data are to be processed by more than one run.

- line 7, format (A), IFORM

A string of max. 20 characters for specifying the single-read signal data format. The format must be given in E- or F-specification including the brackets. The code assumes the same data format in both channels and for all jobs.

Individual Parameter Data :

- line 8, format (A), FINPUTA

A string of max. 50 characters for the file name of the signal data of channel A.

- line 9, format (I5), NAS

The number specifies the first signal data point in channel A where analysis starts. If it is set >1, then NAS-1 initial data points are passed over.

Default value : NAS=1

- line 10, format (A), FINPUTB

A string of max. 50 characters for the file name of the signal data of channel B.

- line 11, format (I5), NBS

As NAS, this number specifies the first signal data point in channel B where analysis starts. With NAS≠NBS a transport peak in the CCF can be aligned to the region of zero lag time, where the statistical estimation accuracy is highest.

Default value : NBS=1

- further 4 lines for job 2.
- further 4 lines for job 3, etc.. The code accepts a next job only, if the individual parameter data are completely given.

## 7.5 Numbered Stops

- STOP 100 : Values for the segment length or the sampling frequency have not correctly been specified. A message is given.
- STOP 101 : Errors have been detected in reading the individual job parameter data.
- STOP 102 : The signal data file of channel A cannot be opened.
- STOP 103 : The signal data file of channel B cannot be opened.
- STOP 104 : The value specified for NAS is too large. The signal data are passed over to the EOF mark.
- STOP 105 : The value specified for NBS is too large.
- STOP 106 : The file ACCF0'''.PLO cannot be opened.

## 7.6 Application Possibilities in BWR Stability Analysis

The code is very fast. If it is used only for univariate ACF estimations, one should take the same signal data in channel A and B, in order to avoid confusion with the identification numbers of the data output files, i.e. to make copies of the data input files before.

The code allows three kinds of the decay ratio determination in combination with the fit codes (ACFIT6, ACFIT7, ACFIT7SA).

Possibility 1 : For a given large value of $N_{av}$, the ACF estimates are briefly called average ACFs. One can estimate average ACFs over all available segments.

Example : The signal data have been filtered by the code FFTF2 with the segment length $N_s$=256. The data output files are : FFTF2_31,DAT.1, FFTF2_32.DAT;1 etc., and the copies FFTF2_31.DAT;2, FFTF2_32.DAT;2 etc.. One has to write on the file ACCF1 :

```
....
NS = 256
NAV = 1000
IPLOT = 31 (gives the corresponding files ACCF0031.PLO, ACCF0032.PLO etc.)
....
FFTF2_31.DAT;1 (job 1)
1
FFTF2_31.DAT;2
1
FFTF2_32.DAT;1 (job 2)
1
FFTF2_32.DAT;2
1
....                 (job 3)
```

By fitting either model A or model B to the data of an average ACF one does not get any information about the statistical uncertainty of the fit parameter data.

Possibility 2 : With $N_{av}$=1 one can estimate segment or instantaneous ACFs along one filtered signal record which contains $N_{seg}$ segments. The start point numbers NAS and NBS are used for selecting the segments consecutively. Considering the previous example, one has to set :

....

NS = 256

NAV = 1

....

FFTF2_31.DAT;1 (job 1, ACF of segment 1)

1

FFTF2_31.DAT;2

1

FFTF2_31.DAT;1 (job 2, ACF of segment 2)

257

FFTF2_31.DAT;2

257

FFTF2_31.DAT;1 (job 3, ACF of segment 3)

513

FFTF2_31.DAT;2

513

....

.... (job $N_{seg}$, ACF of last segment)

The application of the fit procedure to each instantaneous ACF, which is briefly called a segment analysis (SA), gives then a set of fit parameter data from which average values and standard deviations can be calculated. However, instantaneous ACF estimates on actual (filtered) BWR neutron signals showed mostly very large scattering with often strange ACF shapes from segment to segment. It happened seldom that all the instantaneous ACF estimates could be fitted. One had to leave out non-fittable ACFs in this kind of uncertainty analysis.

The used denotation of a segment ACF as instantaneous is related to the old indirect PSD estimation method where a PSD estimate is obtained from Fourier transforming the ACF estimate (with the use of an appropriate correlation function window). The FFT applied to an instantaneous ACF estimate gives an instantaneous PSD estimate, where, for random noise data, the statistical error may be as large as the PSD amplitude at each frequency point.

Possibility 3 : One can treat a gliding segment analysis (GLSA). 'Short-time' ACFs are estimated with a small fixed value of $N_{av}$ (e.g. $N_{av}$=5). The first ACF estimate is obtained from the signal data in the first succeeding $N_{av}$ segments. Each following ACF estimate is based on taking the signal data in the next following segment and retaining the data in the $N_{av}$-1 preceding segments. One obtains a set of (strongly correlated) ACFs which move with some averaging over the signal length. The parameter data input as given for the example of possibility 2 is the same, with the exception that one has to write the desired value for NAV. The number of ACF estimates is given by $N_{seg} - N_{av} + 1$. Attention must be paid to the specification of the last job. If one specify one shift more, the code reduces NAV to the value NAV-1 for the last ACF.

Fits of these ACFs showed promising results with significantly lower standard deviations of the average fit parameter data than observed in the SA.

# LISTING

## A C C F 1

```
      PROGRAM ACCF1
C
C     CORRELATION ANALYSIS OF TWO DIGITAL NOISE RECORDS.
C
C     ACCF1 ESTIMATES
C          AUTOCORRELATION FUNCTION IN EACH CHANNEL
C          CROSS-CORRELATION FUNCTION.
C
C     THE PROCEDURE IS BASED ON FFT TECHNIQUES.
C
C     CODE WRITTEN BY K.BEHRINGER, JULY 1986,
C     RETYPED JULY 1998.
C
C     VERSION FOR THE PSI VAX COMPUTER SYSTEM.
C
C     IMSL ROUTINES ARE REQUIRED.
C
      CHARACTER RUN*50,FINPUTA*50,FINPUTB*50,IFORM*20,FPLOT*20
      REAL*8 XAML,XBML,XAMSQL,XBMSQL
      DIMENSION WFFTR(4111),X(1024),XAML(1000),XBML(1000),
     1XAMSQL(1000),XBMSQL(1000),PSDA(1025),PSDB(1025),
     2CPSDR(1025),CPSDI(1025),SEQ(2048),
     3COEFA(2048),COEFB(2048),SPECA(1025),SPECB(1025),
     4T(2049),ACFA(2049),ACFB(2049),CCF(2049)
      EQUIVALENCE (COEFA(1),XAMSQL(1)),(COEFB(1),XBMSQL(1)),
     1(ACFA(1),SPECA(1)),(ACFB(1),SPECB(1)),(X(1),SEQ(1),CCF(1))
      DATA JOB,FPLOT/0,'ACCF0000.PLO         '/
C
 1000 FORMAT(A/I4/I4/E10.3/I1/I3/A)
 1001 FORMAT(A/I5/A/I5)
 2000 FORMAT(1H1,40X,'P R O G R A M     A C C F 1',13X,
     1'(VERSION FOR VAX COMPUTER)'///1X,'FILES :    PARAMETEER INPUT',
     28X,': FILE ACCF.IN'/11X,'SIGNAL DATA CHANNEL A',3X,
     316H: FILE 'FINPUTA'/11X,'SIGNAL DATA CHANNEL B',3X,
     416H: FILE 'FINPUTB'/11X,'PRINT OUTPUT',12X,': FILE ACCF.PRT'/11X,
     5'DATA OUTPUT (FOR PLOTS)',1X,29H: FILE 'FPLOT' (ACCF0'''.PLO)////
     61X,'RUN DENOTATION',26X,'RUN',4X,'=',2X,A//1X,
     7'SAMPLE SIZE',29X,'NS',5X,'=',I6//1X,
     8'TRANSFORM SIZE',26X,'NT',5X,'=',I6//1X,
     9'NUMBER OF AVERAGES (REQUESTED)',10X,'NAVR',3X,'=',I6//1X,
     A'SAMPLING FREQUENCY (HZ)',17X,'SFR',4X,'=',1PE14.3/1X,
     B'SAMPLING INTERVAL (SEC)',17X,'DT',5X,'=',E14.3/1X,
     C'NYQUIST CUTOFF FREQUENCY (HZ)',11X,'CFR',4X,'=',E14.3/1X,
     D'NYQUIST CO-INTERVAL (HZ)',16X,'DF',5X,'=',E14.3//1X,
     E'DC-SUBTRACTION OPTION',19X,'IOPTDC =',I6,1X,'( 0 : NO',1X,
     F'SUBTRACTION, 1 : GLOBAL SUBTRACTION, 2 : LOCAL SUBTRACTION)'
     G//1X,'INITIAL NUMBER OF FPLOT',17X,'IPLOT  =',I6//1X,
     H'SIGNAL DATA INPUT FORMAT',16X,'IFORM  =',2X,A)
 2001 FORMAT(////11X,'STOP, CHECK PARAMETER DATA !')
 2002 FORMAT(////11X,'E N D')
 2005 FORMAT('1JOB NUMBER',30X,'JOB',5X,'=',I7///1X,'SIGNAL DATA FILE',
     124X,'FINPUTA =',2X,A/41X,'FINPUTB =',2X,A//1X,
```

```
     2'EVALUATION STARTS AT DATA POINT NUMBER',2X,'NAS',5X,'=',I7/41X,
     3'NBS',5X,'=',I7/)
 2006 FORMAT(1X,'NUMBER OF AVERAGES (AVAILABLE)',10X,'NAV',5X,'=',I7/)
 2007 FORMAT(1X,'SIGNAL DC',31X,'XAMG',4X,'=',1PE15.3/41X,
     1'XBMG',4X,'=',E15.3//1X,'SIGNAL VARIANCE',25X,'VARA',4X,'=',E15.3/
     241X,'VARB',4X,'=',E15.3/)
 2008 FORMAT(1X,'DATA OUTPUT FILE',24X,'FPLOT',3X,'=',2X,A)
 2009 FORMAT(////11X,'CHECK PARAMETER DATA OF THIS JOB !')
 7000 FORMAT(1X,I4,1P,4E12.4)
 7001 FORMAT(I3.3)
C
      OPEN(UNIT=1,FILE='ACCF.IN',STATUS='OLD',DEFAULTFILE='DIRINPUT',
     1ERR=100)
      OPEN(UNIT=2,FILE='ACCF.PRT',STATUS='NEW',DEFAULTFILE='DIRINPUT')
      READ(1,1000,ERR=105,END=105) RUN,NS,NAVR,SFR,IOPTDC,IPLOT,IFORM
      DO 1 N=5,10
      IF(NS.EQ.2**N) GO TO 110
    1 CONTINUE
      NT = 0
      NS = -1
      GO TO 115
  100 STOP
  105 CLOSE(UNIT=2,STATUS='DELETE')
      STOP
  110 NT = 2*NS
  115 IF(NAVR.LT.1) NAVR=1
      IF(NAVR.GT.1000) NAVR=1000
      IF(SFR.GT.0.) GO TO 120
      DT = 0.
      CFR = 0.
      DF = 0.
      GO TO 125
  120 DT = 1./SFR
      CFR = 0.5*SFR
      DF = CFR/FLOAT(NS)
  125 IF(IOPTDC.LT.0) IOPTDC=0
      IF(IOPTDC.GT.2) IOPTDC=2
      IF(IPLOT.LT.1) IPLOT=1
      IF(IPLOT.GT.990) IPLOT=990
      WRITE(2,2000) RUN,NS,NT,NAVR,SFR,DT,CFR,DF,IOPTDC,IPLOT,IFORM
      IF(NT.GT.0.AND.SFR.GT.0.) GO TO 130
      WRITE(2,2001)
      STOP 100
  130 CALL FFTRI(NT,WFFTR)
      NS1 = NS+1
      DO 2 N=1,NT+1
    2 T(N) = DT*FLOAT(N-NS1)
  200 READ(1,1001,ERR=205,END=210) FINPUTA,NAS,FINPUTB,NBS
      JOB = JOB+1
      IF(NAS.LT.1) NAS=1
      IF(NBS.LT.1) NBS=1
      WRITE(2,2005) JOB,FINPUTA,FINPUTB,NAS,NBS
      OPEN(UNIT=3,FILE=FINPUTA,STATUS='OLD',DEFAULTFILE='DIRINPUT',
     1ERR=215)
      OPEN(UNIT=4,FILE=FINPUTB,STATUS='OLD',DEFAULTFILE='DIRINPUT',
     1ERR=220)
      CALL DATST(NAS,NBS,IFORM)
      I = 3
```

```
      CALL MEANL(I,IFORM,NS,NAVR,X,N1,XAML,XAMSQL)
      I = 4
      CALL MEANL(I,IFORM,NS,NAVR,X,N2,XBML,XBMSQL)
      NAV = MIN(N1,N2)
      WRITE(2,2006) NAV
      IF(NAV.LT.1) GO TO 300
      CALL MEANG(NAV,XAML,XAMSQL,XAMG,VARA)
      CALL MEANG(NAV,XBML,XBMSQL,XBMG,VARB)
      WRITE(2,2007) XAMG,XBMG,VARA,VARB
      IF(IOPTDC.GT.0) GO TO 225
      XAMG = 0.
      XBMG = 0.
  225 REWIND 3
      REWIND 4
      CALL DATST(NAS,NBS,IFORM)
      DO 5 N=NS1,NT
    5 SEQ(N) = 0.
      DO 6 N=1,NS1
      PSDA(N) = 0.
      PSDB(N) = 0.
      CPSDR(N) = 0.
    6 CPSDI(N) = 0.
      DO 10 I=1,NAV
      A1 = FLOAT(I-1)
      A2 = A1+1.
      A1 = A1/A2
      A2 = 1./A2
      IF(IOPTDC.LT.2) GO TO 230
      XAMG = SNGL(XAML(I))
      XBMG = SNGL(XBML(I))
  230 READ(3,IFORM) (SEQ(N),N=1,NS)
      DO 11 N=1,NS
   11 SEQ(N) = SEQ(N)-XAMG
      CALL F2TRF(NT,SEQ,COEFA,WFFTR)
      READ(4,IFORM) (SEQ(N),N=1,NS)
      DO 12 N=1,NS
   12 SEQ(N) = SEQ(N)-XBMG
      CALL F2TRF(NT,SEQ,COEFB,WFFTR)
      CALL PSDIS(NS,COEFA,SPECA)
      CALL PSDIS(NS,COEFB,SPECB)
      DO 13 N=1,NS1
      PSDA(N) = A1*PSDA(N)+A2*SPECA(N)
   13 PSDB(N) = A1*PSDB(N)+A2*SPECB(N)
      CALL CPSDIS(NS,COEFA,COEFB,SPECA,SPECB)
      DO 14 N=1,NS1
      CPSDR(N) = A1*CPSDR(N)+A2*SPECA(N)
   14 CPSDI(N) = A1*CPSDI(N)+A2*SPECB(N)
   10 CONTINUE
      A1 = 1./FLOAT(NT)
      DO 20 N=1,NS1
      PSDA(N) = A1*PSDA(N)
      PSDB(N) = A1*PSDB(N)
      CPSDR(N) = A1*CPSDR(N)
   20 CPSDI(N) = A1*CPSDI(N)
      CALL COEFAC(NS,PSDA,COEFA)
      CALL F2TRB(NT,COEFA,COEFB,WFFTR)
      CALL CORRF(NS,COEFB,ACFA)
```

```
      CALL COEFAC(NS,PSDB,COEFA)
      CALL F2TRB(NT,COEFA,COEFB,WFFTR)
      CALL CORRF(NS,COEFB,ACFB)
      CALL COEFCC(NS,CPSDR,CPSDI,COEFA)
      CALL F2TRB(NT,COEFA,COEFB,WFFTR)
      CALL CORRF(NS,COEFB,CCF)
      DO 21 N=2,NT
      ACFA(N) = A1*ACFA(N)
      ACFB(N) = A1*ACFB(N)
   21 CCF(N) = A1*CCF(N)
      ENCODE(3,7001,FPLOT(6:8)) IPLOT
      WRITE(2,2008) FPLOT
      OPEN(UNIT=7,FILE=FPLOT,STATUS='NEW',DEFAULTFILE='DIRINPUT',
     1ERR=305)
      WRITE(7,7000) (N,T(N),ACFA(N),ACFB(N),CCF(N),N=1,NT+1)
      CLOSE(UNIT=7,STATUS='KEEP')
      IPLOT = IPLOT+1
      IF(IPLOT.GT.999) GO TO 210
  310 CLOSE(UNIT=3,STATUS='KEEP')
      CLOSE(UNIT=4,STATUS='KEEP')
      GO TO 200
  300 WRITE(2,2009)
      GO TO 310
  210 WRITE(2,2002)
      STOP
  205 STOP 101
  215 STOP 102
  220 STOP 103
  305 STOP 106
      END
C============================================================
      SUBROUTINE DATST(NAS,NBS,IFORM)
C
C     DATST MOVES THE SIGNAL DATA FILES UNTIL THE GIVEN DATA NUMBERS
C     WHERE EVALUATION STARTS.
C
      CHARACTER IFORM*20
      IF(NAS.EQ.1) GO TO 105
      DO 1 N=1,NAS-1
    1 READ(3,IFORM,END=100) X
  105 IF(NBS.EQ.1) GO TO 115
      DO 2 N=1,NBS-1
    2 READ(4,IFORM,END=110) X
  115 RETURN
  100 STOP 104
  110 STOP 105
      END
C============================================================
      SUBROUTINE MEANL(IUNIT,IFORM,NS,NAVR,X,NAVP,XM,XMSQ)
C
C     MEANL COMPUTES THE LOCAL MEAN AND THE LOCAL MEAN SQUARE OF
C     THE SIGNAL DATA IN A CHANNEL FOR EACH SAMPLE.
C
      CHARACTER IFORM*20
      REAL*8 XM,XMSQ,XN,A1,A2,SM,SMSQ
      DIMENSION X(1),XM(1),XMSQ(1)
      DO 1 M=1,NAVR
      READ(IUNIT,IFORM,END=100) (X(N),N=1,NS)
```

```
      SM = 0.D0
      SMSQ = 0.D0
      DO 2 N=1,NS
      A1 = DFLOAT(N-1)
      A2 = A1+1.D0
      A1 = A1/A2
      A2 = 1.D0/A2
      XN = DBLE(X(N))
      IF(DABS(XN).LT.1.D-15) XN=0.D0
      SM = A1*SM+A2*XN
    2 SMSQ = A1*SMSQ+A2*XN**2
      XM(M) = SM
    1 XMSQ(M) = SMSQ
      NAVP = NAVR
      RETURN
  100 NAVP = M-1
      RETURN
      END
C============================================================
      SUBROUTINE MEANG(NAV,XM,XMSQ,XMG,VAR)
C
C     MEANG COMPUTES THE GLOBAL MEAN AND THE GLOBAL VARIANCE OF
C     THE SIGNAL DATA IN A CHANNEL FOR THE SPECIFIED RECORD LENGTH.
C
      REAL*8 XM,XMSQ,A1,A2,SM,SMSQ
      DIMENSION XM(1),XMSQ(1)
      SM = 0.D0
      SMSQ = 0.D0
      DO 1 N=1,NAV
      A1 = DFLOAT(N-1)
      A2 = A1+1.D0
      A1 = A1/A2
      A2 = 1.D0/A2
      SM = A1*SM+A2*XM(N)
    1 SMSQ = A1*SMSQ+A2*XMSQ(N)
      XMG = SNGL(SM)
      VAR = SNGL(SMSQ-SM**2)
      RETURN
      END
C============================================================
      SUBROUTINE PSDIS(NS,COEF,SPEC)
C
C     PSDIS COMPUTES THE INSTANTANEOUS PSD FROM THE FOURIER
C     COEFFICIENTS OF A SIGNAL DATA SAMPLE INCLUDING ZERO-PADDING.
C
      DIMENSION COEF(1),SPEC(1)
      SPEC(1) = COEF(1)**2
      SPEC(NS+1) = COEF(2*NS)**2
      DO 1 N=2,NS
    1 SPEC(N) = COEF(2*(N-1))**2+COEF(2*N-1)**2
      RETURN
      END
C============================================================
      SUBROUTINE CPSDIS(NS,COEFA,COEFB,SPECR,SPECI)
C
C     CPSDIS COMPUTES THE INSTANTANEOUS CPSD FROM THE FOURIER
C     COEFFICIENTS OF THE SIGNAL DATA SAMPLES OF THE CHANNELS A AND B
```

```
C     INCLUDING ZERO-PADDING.
C
      DIMENSION COEFA(1),COEFB(1),SPECR(1),SPECI(1)
      N1 = 2*NS
      N2 = NS+1
      SPECR(1) = COEFA(1)*COEFB(1)
      SPECR(N2) = COEFA(N1)*COEFB(N1)
      SPECI(1) = 0.
      SPECI(N2) = 0.
      DO 1 N=2,NS
      N1 = 2*(N-1)
      N2 = N1+1
      SPECR(N) = COEFA(N1)*COEFB(N1)+COEFA(N2)*COEFB(N2)
    1 SPECI(N) = COEFA(N1)*COEFB(N2)-COEFB(N1)*COEFA(N2)
      RETURN
      END
C=========================================================
      SUBROUTINE COEFAC(NS,PSD,COEF)
C
C     COEFAC COMPUTES THE FOURIER COEFFICIENTS REQUIRED FOR THE
C     INVERSE FOURIER TRANSFORM OF THE PSD.
C
      DIMENSION PSD(1),COEF(1)
      COEF(1) = PSD(1)
      COEF(2*NS) = PSD(NS+1)
      DO 1 N=2,NS
      COEF(2*(N-1)) = PSD(N)
    1 COEF(2*N-1) = 0.
      RETURN
      END
C=========================================================
      SUBROUTINE COEFCC(NS,CPSDR,CPSDI,COEF)
C
C     COEFCC COMPUTES THE FOURIER COEFFICIENTS REQUIRED FOR THE
C     INVERSE FOURIER TRANSFORM OF THE CPSD.
C
      DIMENSION CPSDR(1),CPSDI(1),COEF(1)
      COEF(1) = CPSDR(1)
      COEF(2*NS) = CPSDR(NS+1)
      DO 1 N=2,NS
      COEF(2*(N-1)) = CPSDR(N)
    1 COEF(2*N-1) = CPSDI(N)
      RETURN
      END
C=========================================================
      SUBROUTINE CORRF(NS,CFC,CF)
C
C     CORRF COMPUTES THE CORRELATION FUNCTION FROM THE CIRCULAR
C     CORRELATION FUNCTION ANS SHUFFLES THE DATA FOR ASCENDANT
C     TIME LAG NUMBERS. ZERO TIME LAG CORRESPONDS TO THE DATA
C     NUMBER NS+1. THE DATA AT THE NUMBERS 1 AND 2*NS+1 ARE SET
C     EQUAL ZERO.
C
      DIMENSION CFC(1),CF(1)
      NS1 = NS+1
      NT = 2*NS
      CF(1) = 0.
      CF(NT+1) = 0.
```

```
      CF(NS1) = 2.*CFC(1)
      DO 1 N=2,NS
      CF(N) = FLOAT(NT)/FLOAT(N-1)*CFC(N+NS)
    1 CF(N+NS) = FLOAT(NT)/FLOAT(NS1-N)*CFC(N)
      RETURN
      END
```

# 8. PROGRAM ACCF2

Precision : single

Operation : foreground, background

Required auxiliary routines : from IMSL : F2TRF, F2TRB, FFTRI

Purpose of the program :

Estimation of Auto-Correlation Functions for the Gliding Segment Analysis

Feature Summary :

- Modified version of the code ACCF1 for automatic segment shifts.
- No provision for the DC component removal, assuming signal data previously band-pass filtered by the code FFTF2.
- Within a run, a limited number of auto-correlation functions (ACFs) can be estimated on one signal pair. Each case is called a job.

## 8.1 Modifications Against the Code ACCF1

The individual parameter data input at the code ACCF1 for the application possibilities 2 and 3 (segment analysis (SA), gliding segment analysis (GLSA)) is lengthy. The code version ACCF2 avoids this work, but restricts processing to one signal pair for a run. The bivariate character of ACCF1 has been maintained, but it is understodd that one should take the same signal data in channel A and B.

In the GLSA 'short-time' ACFs are estimated over a basic set of $N_{av}$ segments. The first ACF estimate is obtained from the signal data in the first succeeding $N_{av}$ segments. The basic set is shifted to include the data of the next segment, therby removing the data of the first segment. A second ACF is then calculated. This shift procedure and the calculation of further ACFs continue according to the given number $N_{sh}$ of shifts. The code can also be applied to the SA by setting $N_{av}$=1, or to evaluate an 'average' ACF by setting $N_{sh}$=0 and $N_{av}$ to a large value.

The code assumes that the ACFs are estimated on signal data previously filtered by the code FFTF2. A removal of the DC component is therefore not provided, because in the case of low-pass filtering the DC component can be removed there. It is obvious that the segments must exactly be scanned in length and position.

## 8.2 Files

There are 5 files as in the code ACCF1. They have the same meaning : ACCF2.IN, ACCF2.PRT, 'FINPUTA', 'FINPUTB', and ACCF0'''.PLO. The denotation of the data output file has not been changed with respect to its use in the fit codes.

## 8.3 Parameter Data Input (file ACCF2.IN)

- line 1, format (A), RUN (as in ACCF1)
- line 2, format (I4), NS (as in ACCF1)
- line 3, format (I4), NAVR

Requested number of averages. It defines the basis set of consecutive segments.

Internal restrictions : IF(NAVR.LT.1) NAVR=1 (default value)

IF(NAVR.GT.1000) NAVR=1000

If the requested number of averages is not available, the code searches for the maximum available number of averages. In this case, the value of the following parameter NSHR is set equal zero.

- line 4, format (I4), NSHR

Requested number of shifts of the basic segment set. The present code version assumes a shift operation by one segment.

Internal restrictions : IF(NSHR.LT.0) NSHR=0 (default value)

IF(NSHR.GT.100) NSHR=100

If the requested number of shifts of the basic segment set is not available, the code searches for the maximum available number of shifts.

The following lines are as in ACCF1 :

- line 5, format (E10.3), SFR
- line 6, format (I3), IPLOT
- line 7, format (A), IFORM
- line 8, format (A), FINPUTA

- line 9, format (I5), NAS
- line 10, format (A), FINPUTB
- line 11, format (I5), NBS

The numbered stops are the same as in the code ACCF1.

# LISTING

## A C C F 2

```
      PROGRAM ACCF2
C
C     ACCF2 ESTIMATES
C            AUTOCORRELATION FUNCTIONS IN EACH CHANNEL AND
C            CROSS-CORRELATION FUNCTIONS FOR GLIDING SEGMENT ANALYSIS.
C            THE SIGNAL DATA ARE ASSUMED TO BE BAND-PASS FILTERED.
C
C     THE PROCEDURE IS BASED ON FFT TECHNIQUES.
C
C     CODE WRITTEN BY K.BEHRINGER, MAY 2000.
C
C     VERSION FOR THE PSI DEC COMPUTER SYSTEM.
C
C     IMSL ROUTINES ARE REQUIRED.
C
      PARAMETER (NSHMAX=100)
      CHARACTER RUN*50,FINPUTA*50,FINPUTB*50,IFORM*20,FPLOT*20
      DIMENSION WFFTR(4111),PSDA(1025),PSDB(1025),
     1CPSDR(1025),CPSDI(1025),SEQ(2048),X(1024),
     2COEFA(2048),COEFB(2048),SPECA(1025),SPECB(1025),
     3T(2049),ACFA(2049),ACFB(2049),CCF(2049)
      EQUIVALENCE (ACFA(1),SPECA(1)),(ACFB(1),SPECB(1)),
     1(X(1),SEQ(1),CCF(1))
      DATA FPLOT/'ACCF0000.PLO        '/
C
 1000 FORMAT(A/I4/I4/I4/E10.3/I3/A)
 1001 FORMAT(A/I5/A/I5)
 2000 FORMAT(1H1,40X,'P R O G R A M     A C C F 2',13X,
     1'(VERSION FOR DEC COMPUTER)'///1X,'FILES :   PARAMETEER INPUT',
     28X,': FILE ACCF2.IN'/11X,'SIGNAL DATA CHANNEL A',3X,
     316H: FILE 'FINPUTA'/11X,'SIGNAL DATA CHANNEL B',3X,
     416H: FILE 'FINPUTB'/11X,'PRINT OUTPUT',12X,': FILE ACCF2.PRT'/11X
     5,'DATA OUTPUT (FOR PLOTS)',1X,29H: FILE 'FPLOT' (ACCF0'''.PLO)///
     61X,'RUN DENOTATION',26X,'RUN',4X,'=',2X,A//1X,
     7'SAMPLE SIZE',29X,'NS',5X,'=',I6//1X,
     8'TRANSFORM SIZE',26X,'NT',5X,'=',I6//1X,
     9'NUMBER OF AVERAGES (REQUESTED)',10X,'NAVR',3X,'=',I6//1X,
     A'NUMBER OF SHIFTS (REQUESTED)',12X,'NSHR',3X,'=',I6//1X,
     B'SAMPLING FREQUENCY (HZ)',17X,'SFR',4X,'=',1PE14.3/1X,
     C'SAMPLING INTERVAL (SEC)',17X,'DT',5X,'=',E14.3/1X,
     D'NYQUIST CUTOFF FREQUENCY (HZ)',11X,'CFR',4X,'=',E14.3/1X,
     E'NYQUIST CO-INTERVAL (HZ)',16X,'DF',5X,'=',E14.3//1X,
     F'INITIAL NUMBER OF FPLOT',17X,'IPLOT  =',I6//1X,
     G'SIGNAL DATA INPUT FORMAT',16X,'IFORM  =',2X,A/)
 2001 FORMAT(////11X,'STOP, CHECK PARAMETER DATA !')
 2002 FORMAT(////11X,'E N D')
 2005 FORMAT(1X,'SIGNAL DATA FILES',
     123X,'FINPUTA =',2X,A/41X,'FINPUTB =',2X,A//1X,
     2'EVALUATION STARTS AT DATA POINT NUMBERS',1X,'NAS',5X,'=',I7/41X,
     3'NBS',5X,'=',I7/)
 2006 FORMAT(1X,'NUMBER OF AVERAGES (AVAILABLE)',10X,'NAV',5X,'=',I7//
     11X,'NUMBER OF SHIFTS (AVAILABLE)',12X,'NSH',5X,'=',I7/)
```

```
 2007 FORMAT('1  JOB',21X,'NAS',17X,'NBS',12X,'FPLOT'/)
 2008 FORMAT(1X,I5,4X,2I20,10X,A)
 2009 FORMAT(////11X,'CHECK PARAMETER DATA OF THIS RUN !')
 7000 FORMAT(1X,I4,1P,4E12.4)
 7001 FORMAT(I3.3)
C
      OPEN(UNIT=1,FILE='ACCF2.IN',STATUS='OLD',DEFAULTFILE='DIRINPUT',
     1ERR=100)
      OPEN(UNIT=2,FILE='ACCF2.PRT',STATUS='NEW',DEFAULTFILE='DIRINPUT')
      READ(1,1000,ERR=105,END=105) RUN,NS,NAVR,NSHR,SFR,IPLOT,IFORM
      DO 1 N=5,10
      IF(NS.EQ.2**N) GO TO 110
    1 CONTINUE
      NT = 0
      NS = -1
      GO TO 115
  100 STOP
  105 CLOSE(UNIT=2,STATUS='DELETE')
      STOP
  110 NT = 2*NS
  115 IF(NAVR.LT.1) NAVR=1
      IF(NAVR.GT.1000) NAVR=1000
      IF(NSHR.LT.0) NSHR=0
      IF(NSHR.GT.NSHMAX) NSHR=NSHMAX
      IF(SFR.GT.0.) GO TO 120
      DT = 0.
      CFR = 0.
      DF = 0.
      GO TO 125
  120 DT = 1./SFR
      CFR = 0.5*SFR
      DF = CFR/FLOAT(NS)
  125 IF(IPLOT.LT.1) IPLOT=1
      IF(IPLOT.GT.990) IPLOT=990
      WRITE(2,2000) RUN,NS,NT,NAVR,NSHR,SFR,DT,CFR,DF,IPLOT,IFORM
      IF(NT.GT.0.AND.SFR.GT.0.) GO TO 130
      WRITE(2,2001)
      STOP 100
  130 CALL FFTRI(NT,WFFTR)
      NS1 = NS+1
      DO 2 N=1,NT+1
    2 T(N) = DT*FLOAT(N-NS1)
      READ(1,1001,ERR=205,END=210) FINPUTA,NAS,FINPUTB,NBS
      IF(NAS.LT.1) NAS=1
      IF(NBS.LT.1) NBS=1
      WRITE(2,2005) FINPUTA,FINPUTB,NAS,NBS
      OPEN(UNIT=3,FILE=FINPUTA,STATUS='OLD',DEFAULTFILE='DIRINPUT',
     1ERR=215)
      OPEN(UNIT=4,FILE=FINPUTB,STATUS='OLD',DEFAULTFILE='DIRINPUT',
     1ERR=220)
      CALL DATST(NAS,NBS,IFORM)
      I = 3
      CALL CHECK(X,I,IFORM,NS,NAVR,N1,NSHR,K1)
      I = 4
      CALL CHECK(X,I,IFORM,NS,NAVR,N2,NSHR,K2)
      NAV = MIN(N1,N2)
      NSH = MIN(K1,K2)
      WRITE(2,2006) NAV,NSH
```

```
      IF(NAV.LT.1) GO TO 300
      L = 1
      DO 30 JOB=1,NSH+1
      REWIND 3
      REWIND 4
      IF(JOB.EQ.L) THEN
      WRITE(2,2007)
      L = L+50
      END IF
      N1 = (JOB-1)*NS
      NAS1 = NAS+N1
      NBS1 = NBS+N1
      ENCODE(3,7001,FPLOT(6:8)) IPLOT
      WRITE(2,2008) JOB,NAS1,NBS1,FPLOT
      CALL DATST(NAS1,NBS1,IFORM)
      DO 5 N=NS1,NT
    5 SEQ(N) = 0.
      DO 6 N=1,NS1
      PSDA(N) = 0.
      PSDB(N) = 0.
      CPSDR(N) = 0.
    6 CPSDI(N) = 0.
      DO 10 I=1,NAV
      A1 = FLOAT(I-1)
      A2 = A1+1.
      A1 = A1/A2
      A2 = 1./A2
      READ(3,IFORM) (SEQ(N),N=1,NS)
      CALL F2TRF(NT,SEQ,COEFA,WFFTR)
      READ(4,IFORM) (SEQ(N),N=1,NS)
      CALL F2TRF(NT,SEQ,COEFB,WFFTR)
      CALL PSDIS(NS,COEFA,SPECA)
      CALL PSDIS(NS,COEFB,SPECB)
      DO 13 N=1,NS1
      PSDA(N) = A1*PSDA(N)+A2*SPECA(N)
   13 PSDB(N) = A1*PSDB(N)+A2*SPECB(N)
      CALL CPSDIS(NS,COEFA,COEFB,SPECA,SPECB)
      DO 14 N=1,NS1
      CPSDR(N) = A1*CPSDR(N)+A2*SPECA(N)
   14 CPSDI(N) = A1*CPSDI(N)+A2*SPECB(N)
   10 CONTINUE
      A1 = 1./FLOAT(NT)
      DO 20 N=1,NS1
      PSDA(N) = A1*PSDA(N)
      PSDB(N) = A1*PSDB(N)
      CPSDR(N) = A1*CPSDR(N)
   20 CPSDI(N) = A1*CPSDI(N)
      CALL COEFAC(NS,PSDA,COEFA)
      CALL F2TRB(NT,COEFA,COEFB,WFFTR)
      CALL CORRF(NS,COEFB,ACFA)
      CALL COEFAC(NS,PSDB,COEFA)
      CALL F2TRB(NT,COEFA,COEFB,WFFTR)
      CALL CORRF(NS,COEFB,ACFB)
      CALL COEFCC(NS,CPSDR,CPSDI,COEFA)
      CALL F2TRB(NT,COEFA,COEFB,WFFTR)
      CALL CORRF(NS,COEFB,CCF)
      DO 21 N=2,NT
```

```
      ACFA(N) = A1*ACFA(N)
      ACFB(N) = A1*ACFB(N)
   21 CCF(N) = A1*CCF(N)
      OPEN(UNIT=7,FILE=FPLOT,STATUS='NEW',DEFAULTFILE='DIRINPUT',
     1ERR=305)
      WRITE(7,7000) (N,T(N),ACFA(N),ACFB(N),CCF(N),N=1,NT+1)
      CLOSE(UNIT=7,STATUS='KEEP')
      IPLOT = IPLOT+1
      IF(IPLOT.GT.999) GO TO 210
   30 CONTINUE
  210 WRITE(2,2002)
      STOP
  300 WRITE(2,2009)
      STOP
  205 STOP 101
  215 STOP 102
  220 STOP 103
  305 STOP 106
      END
C=======================================================
      SUBROUTINE DATST(NAS,NBS,IFORM)
C
C     DATST MOVES THE SIGNAL DATA FILES UNTIL THE GIVEN DATA NUMBERS
C     WHERE EVALUATION STARTS.
C
      CHARACTER IFORM*20
      IF(NAS.EQ.1) GO TO 105
      DO 1 N=1,NAS-1
    1 READ(3,IFORM,END=100) X
  105 IF(NBS.EQ.1) GO TO 115
      DO 2 N=1,NBS-1
    2 READ(4,IFORM,END=110) X
  115 RETURN
  100 STOP 104
  110 STOP 105
      END
C=======================================================
      SUBROUTINE CHECK(X,IUNIT,IFORM,NS,NAVR,NAVP,NSHR,NSHP)
C
C     CHECK DETERMINES THE POSSIBLE NUMBERS OF AVERAGES
C     AND SHIFTS.
C
      CHARACTER*20 IFORM
      DIMENSION X(1)
      DO 1 M=1,NAVR
    1 READ(IUNIT,IFORM,ERR=100,END=100) (X(N),N=1,NS)
      NAVP = NAVR
      IF(NSHR.EQ.0) THEN
      NSHP = 0
      RETURN
      END IF
      DO 2 M=1,NSHR
    2 READ(IUNIT,IFORM,ERR=105,END=105) (X(N),N=1,NS)
      NSHP = NSHR
      RETURN
  100 NAVP = M-1
      NSHP = 0
      RETURN
```

```
  105 NSHP = M-1
      RETURN
      END
C========================================================
      SUBROUTINE PSDIS(NS,COEF,SPEC)
C
C     PSDIS COMPUTES THE INSTANTANEOUS PSD FROM THE FOURIER
C     COEFFICIENTS OF A SIGNAL DATA SAMPLE INCLUDING ZERO-PADDING.
C
      DIMENSION COEF(1),SPEC(1)
      SPEC(1) = COEF(1)**2
      SPEC(NS+1) = COEF(2*NS)**2
      DO 1 N=2,NS
    1 SPEC(N) = COEF(2*(N-1))**2+COEF(2*N-1)**2
      RETURN
      END
C========================================================
      SUBROUTINE CPSDIS(NS,COEFA,COEFB,SPECR,SPECI)
C
C     CPSDIS COMPUTES THE INSTANTANEOUS CPSD FROM THE FOURIER
C     COEFFICIENTS OF THE SIGNAL DATA SAMPLES OF THE CHANNELS A AND B
C     INCLUDING ZERO-PADDING.
C
      DIMENSION COEFA(1),COEFB(1),SPECR(1),SPECI(1)
      N1 = 2*NS
      N2 = NS+1
      SPECR(1) = COEFA(1)*COEFB(1)
      SPECR(N2) = COEFA(N1)*COEFB(N1)
      SPECI(1) = 0.
      SPECI(N2) = 0.
      DO 1 N=2,NS
      N1 = 2*(N-1)
      N2 = N1+1
      SPECR(N) = COEFA(N1)*COEFB(N1)+COEFA(N2)*COEFB(N2)
    1 SPECI(N) = COEFA(N1)*COEFB(N2)-COEFB(N1)*COEFA(N2)
      RETURN
      END
C========================================================
      SUBROUTINE COEFAC(NS,PSD,COEF)
C
C     COEFAC COMPUTES THE FOURIER COEFFICIENTS REQUIRED FOR THE
C     INVERSE FOURIER TRANSFORM OF THE PSD.
C
      DIMENSION PSD(1),COEF(1)
      COEF(1) = PSD(1)
      COEF(2*NS) = PSD(NS+1)
      DO 1 N=2,NS
      COEF(2*(N-1)) = PSD(N)
    1 COEF(2*N-1) = 0.
      RETURN
      END
C========================================================
      SUBROUTINE COEFCC(NS,CPSDR,CPSDI,COEF)
C
C     COEFCC COMPUTES THE FOURIER COEFFICIENTS REQUIRED FOR THE
C     INVERSE FOURIER TRANSFORM OF THE CPSD.
C
```

```
      DIMENSION CPSDR(1),CPSDI(1),COEF(1)
      COEF(1) = CPSDR(1)
      COEF(2*NS) = CPSDR(NS+1)
      DO 1 N=2,NS
      COEF(2*(N-1)) = CPSDR(N)
    1 COEF(2*N-1) = CPSDI(N)
      RETURN
      END
C===========================================================
      SUBROUTINE CORRF(NS,CFC,CF)
C
C     CORRF COMPUTES THE CORRELATION FUNCTION FROM THE CIRCULAR
C     CORRELATION FUNCTION ANS SHUFFLES THE DATA FOR ASCENDANT
C     TIME LAG NUMBERS. ZERO TIME LAG CORRESPONDS TO THE DATA
C     NUMBER NS+1. THE DATA AT THE NUMBERS 1 AND 2*NS+1 ARE SET
C     EQUAL ZERO.
C
      DIMENSION CFC(1),CF(1)
      NS1 = NS+1
      NT = 2*NS
      CF(1) = 0.
      CF(NT+1) = 0.
      CF(NS1) = 2.*CFC(1)
      DO 1 N=2,NS
      CF(N) = FLOAT(NT)/FLOAT(N-1)*CFC(N+NS)
    1 CF(N+NS) = FLOAT(NT)/FLOAT(NS1-N)*CFC(N)
      RETURN
      END
```

# 9. PROGRAM ACFOSC5 AND PROGRAM ACFOSC6

Precision : double

Operation : foreground, background

Required auxiliary routines : from IMSL : DQDAWO, DQDAWF

Purpose of the programs ;

Generation of Artificial Auto-Correlation Functions,

ACFOSC5 for Model A,

ACFOSC6 for Model B.

Feature summary :

- Calculation of the ideal auto-correlation function (ACF) with parameter data referring to the oscillator model A or B.
- Calculation of a set of ACFs corrected for band-pass filtered signal data. The band-pass filter is assumed to be rectangulsar with unit gain.
- Calculation of a set of fit ACFs which simulate the presence of background under the peak in the power spectral density (PSD).
- The resonance frequency of the peak in the PSD must lie within the filter cutoff frequencies.
- Within a run, a limited number of fit ACFs with different filter cutoff frequencies and different values of the two-parametric background under the peak in the PSD can be calculated. Each case is called a job.

## 9.1 Methodological Background

The two codes have been written for two tasks :

- for testing the fit codes, and
- for performing fit sensitivity studies under well defined conditions. The fit codes described in Section 10 contain a variable fit range parameter which determines the lag time up to which the model ACFs are to be fitted to the ACFs estimated on actual (filtered) signal data. Signal filtering, the presence of a PSD background, and a limited lag time range involve an information loss. Such studies allow the evaluation

of conditions where the fit parameter data do not reproduce the well determined parameter data of the artificial ACFs.

Both codes have the same structure. The code ACFOSC5 refers to the oscillator model A, and the code ACFOSC6 to the oscillator model B. The models are given in Section 2. Although these models are continuous in the variables, some digital features have been taken over from the indirect ACF estimation method by the code ACCF1 (Section 7). The main common input parameters for the calculation of the ideal ACF are the Nyquist cutoff frequency, a lag time resolution value, the resonance frequency of the undamped oscillator (model A) or the resonance frequency of the peak in the PSD (model B) respectively, and the decay ratio. The ideal ACF is normalized to unity amplitude at zero lag time. Sets of individual parameter data can be given for calculating ACFs corrected for assumed previously band-pass filtered signal data, and for calculating fit ACFs assuming in addition a two-parametric background under the peak in the PSD. These individual parameters will be explained in the list of the parameter data input.

## 9.2 Files

There are 4 files :

- ACFOSC5.IN, ACFOSC6.IN

These files contain the input parameter data. The data number, order and formats are the same for both codes.

- ACFOSC5.PRT, ACFOSC6.PRT

These files are meant for printing. They contain (with text) all input parameter data, parameter data modified during processing, additional data which are revelant for comparing with output data from the fit codes, and messages from the computation progress. In particular, the file ACFOSC6.PRT contains, in addition, the lag time values and the amplitude values of the first three minima and following maxima of the ideal ACF.

- ACFOSC5_'''.PLO, ACFOSC6_'''.PLO

These files contain right-hand sided ACF output data for plotting. There are 7 columns with the line format (1X,I4,3X,1P,6E12.5).

column

1 : N, current line number, N=1,LAGS+1
(for the definition of LAGS, see parameter data input list)

2 : TAU, lag time of the ACFs (sec), TAU=0,TAUMAX

3 : RXXI, ideal ACF (no signal filtering, no PSD background)

4 : ENV, upper envelope of RXXI

5 : -ENV, lower envelope of RXXI

6 : RXXC, corrected ACF due to signal filtering

7 : RXXF, ACF fit function = RXXC + Background

For each job a separate file is opened with a current extension number of up to 3 digits. The initial number for the first job must be specified on the parameter data input file. For each further job this number is then incremented by 1.

- ACFOSC5_'''.DAT, ACFOSC6_'''.DAT

The data on these files are thought for the input to the fit codes and contain the two-sided ACF fit function values RXXF (ACFOSC5_'''.DAT for the input to the code ACFIT6, ACFOSC6_'''.DAT for the input to the code ACFIT7 (Section 10)). There are 3 columns with the line format (1X,I4,1P,2E12.5). This format corresponds to the data output format in the code ACCF1 or ACCF2 respectively of channel A.

column

1 : N, current line number, N=1,2*LAGS+1

2 : TAU, lag time (sec), TAU= -TAUMAX,+TAUMAX

3 : RXXF, two-sided ACF fit function

The extension number of these files is the same as for the plot files.

### 9.3 Parameter Data Input (files ACFOSC5.IN, ACFOSC6.IN)

Common Parameter Data :

- line 1, format (A), RUN

A string of max. 50 characters for run identification.

- line 2, format (D12.5), CFR

Nyquist cutoff frenquency in Hz.

Internal restrictions : IF(CFR.LT.5.) CFR=5.

IF(CFR.GT.10.) CFR=10.

- line 3, format (I4), LAGS

Desired number of lags in the one-sided ACFs.

The codes assume a segment length (transform size NT) of 256 data points. This value has normally been used in filtering actual signal data by the code FFTF2 (Section 5) and the following ACF estimations by the code ACCF1 or ACCF2 respectively (Sections 7, 8). The value is given in the data declaration and can be changed. CFR and NT determine the lag time range TAUMAX of the ACF. According to the desired resolution of the ACF (for graphical plots) the incremental lag time step DTAU is given by TAUMAX/LAGS. For testing the fit codes, the value LAGS=256 is recommended.

Internal restrictions : IF(LAGS.LT.20) LAGS=20

IF(LAGS.GT.LAGMAX) LAGS=LAGMAX

LAGMAX is an internal parameter set equal 1000.

- line 4, format (D12.5), FR0

Model A (code ACFOSC5) : resonance frequency (Hz) of the undamped oscillator.

Model B (code ACFOSC6) : resonance frequency (Hz) of the peak in the PSD.

Internal restrictions : IF(FR0.LT.0.01*CFR) FR0=0.01*CFR

IF(FR0.GT.0.5*CFR) FR0=0.5*CFR

- line 5, format (D12.5), DR

Decay ratio.

The decay ratio refers to the ACF oscillation frequency FR0C.

The decay constant FLAMDA is calculated from DR and FR0.

Internal restrictions : IF(DR.LT.0.2) DR=0.2

IF(DR.GT.0.98) DR=0.98 ; code ACFOSC5

IF(DR.GT.0.99) DR=0.99 ; code ACFOSC6

- line 6. format (I3), IPLOT

Initial extension number of the plot and data output files for the first job. For each further job this number is incremented by 1.

Internal restrictions : IF(IPLOT.LT.1) IPLOT=1

IF(IPLOT.GT.1000-FRLHMAX) IPLOT=1000-FRLHMAX

The maximum number of jobs is given by FRLHMAX (declared as integer parameter) and is presently set FRLHMAX=10.

- line 7, format (I2), JOBMAX

Number of jobs within a run.

Internal restrictions : IF(JOBMAX.LT.1) JOBMAX=1

IF(JOBMAX.GT.FRLHMAX) JOBMAX=FRLHMAX

Individual Parameter Data :

- line 8 and the following lines according to the given value of JOBMAX :

format (4D12.5), FRL, FRH, BACKGR, SLOPE

FRL : Lower cutoff frequency (Hz) of the assumed rectangular signal filter.

FRH : Upper (higher) cutoff frequency (Hz) of the filter.

BACKGR : Percentage background amplitude admixture.

The background amplitude B is defined by
B=0.01*BACKGR*PEAKMAX,

where PEAKMAX is the peak amplitude in the PSD.

PEAKMAX=PSD(FR0R), where FR0R is the peak resonance frequency in the PSD. In model B, FR0R=FR0.

SLOPE : Background slope index. It is defined by

SLOPE=LOG10(background in the PSD at FR+1 Hz)/

background in the PSD at FR Hz)

E.g. a value of SLOPE=-2 means, that the background under the peak in the PSD has an exponential decay over 2 decades within 1 Hz. The background decay constant ALPHA is calculated from SLOPE.

Internal restrictions : 0.LE.FRL.LT.0.8*FR0

1.2*FR0.LT.FRH.LE.CFR

0.LE.BACKGR

ABS(SLOPE).LT.6

Violation of any one of these conditions stops further processing.

# LISTING

# ACFOSC5

```
      PROGRAM ACFOSC5
C
C     THE CODE REFERS TO THE DAMPED OSCILLATOR MODEL EXCITED BY RANDOM
C     WHITE NOISE. IT CALCULATES THE IDEAL AUTO-CORRELATION FUNCTION
C     (ACF) AND THE ACF WHICH RESULTS FROM PREVIOUSLY BAND-PASS FILTERED
C     SIGNALS. THE BAND-PASS FILTER IS ASSUMED TO BE RECTANGULAR.
C     THE (UNDAMPED) RESONANCE FREQUENCY MUST LIE WITHIN THE FILTER
C     CUTOFF FREQUENCIES.
C
C     THE PRESENT VERSION 5 CALCULATES THE CORRECTED AUTO-CORRELATION
C     FUNCTION FROM THE IDEAL AUTO-CORRELATION FUNCTION. IT ALLOWS THE
C     ADDITION OF BACKGROUND SPECIFIED IN THE FREQUENCY DOMAIN.
C     IN ADDITION TO VERSION 3, THE TWO-SIDED CORRECTED ACF INCLUDING
C     BACKGROUND IS READ OUT ON A SEPARATE FILE (FILE4) WITH THE SAME
C     FORMAT AS USED IN THE CODE ACCF1. IF ONE SETS LAGS = 256, THESE
C     DATA CAN BE USED FOR TESTING THE DIFFERENT VERSIONS OF THE CODE
C     ACFIT.
C
C     THE CODE ASSUMES THAT THE DIGITAL ESTIMATION OF THE ACF IS BASED
C     ON THE INDIRECT METHOD BY FAST FOURIER TRANSFORM TECHNIQUES WITH
C     ZERO-PADDING. IT REQUIRES THE INPUT OF THE NYQUIST CUTOFF
C     FREQUENCY CFR. THE TRANSFORM SIZE (SEGMENT LENGTH) NT IS GIVEN
C     IN THE DATA DECLARATION. CFR AND NT DETERMINE THE RANGE TAUMAX
C     OF THE ACF. FOR A HIGH RESOLUTION CALCULATION OF THE ACF THE
C     SAMPLING INTERVAL DTAU IS GIVEN BY TAUMAX/LAGS, WHERE LAGS IS
C     THE NUMBER OF LAGS.
C
C     WITHIN A RUN A LIMITED NUMBER OF ACF'S (CALLED JOBS) WITH
C     DIFFERENT PAIRS OF CUTOFF FREQUENCIES AND DIFFERENT VALUES
C     FOR A TWO-PARAMETRIC BACKGROUND CAN BE CALCULATED.
C
C     THE CODE REQUIRES IMSL ROUTINES.
C
C     CODE WRITTEN BY K.BEHRINGER, JUNE 1999, FOR BATCH OPERATION
C     ON THE DEC-ALPHA 2100 COMPUTER.
C
C     PARAMETERS :
C
C          LAGMAX  = MAXIMUM NUMBER OF LAGS
C          FRLHMAX = MAXIMUM NUMBER OF CUTOFF FREQUENCY PAIRS
C
C
C     DECLARATIONS
C
      IMPLICIT REAL*8 (A-H,O-Z)
      INTEGER*4 FRLHMAX
      PARAMETER (LAGMAX=1000,FRLHMAX=10)
      CHARACTER*50 RUN,FILE1,FILE2,FILE3,FILE4
      DIMENSION TAU(2*LAGMAX+1),RXXI(LAGMAX+1),ENV(LAGMAX+1),
     1RXXC(2*LAGMAX+1),RXXF(2*LAGMAX+1),FRL(FRLHMAX),FRH(FRLHMAX),
     2BACKGR(FRLHMAX),SLOPE(FRLHMAX)
      COMMON/PAR/A1,C,OMEGA0,OMEGAC,FLAMDA
```

```
      COMMON/FR/OMEGAL,OMEGAH
      COMMON/CONST/DPI
      DATA C,PI,NT/1.D0,3.141592654D0,256/,
     1FILE1/'ACFOSC5.IN                         '/,
     2FILE2/'ACFOSC5.PRT                        '/,
     3FILE3/'ACFOSC5_^^^.PLO                    '/,
     4FILE4/'ACFOSC5_^^^.DAT                    '/
C
C     FORMATS
C
 1000 FORMAT(A/D12.5/I4/2(D12.5/)I3/I2)
 1005 FORMAT(4D12.5)
 2000 FORMAT('1',40X,'P R O G R A M     A C F O S C 5'///1X,
     1'FILES : INPUT PARAMETER DATA :  FILE1 = ',A/9X,
     2'PRINT OUTPUT',9X,':  FILE2 = ',A/9X,
     3'PLOT OUTPUT',10X,':  FILE3 = ',A/9X,
     4'DATA OUTPUT',10X,':  FILE4 = ',A//)
 2005 FORMAT(1X,'RUN DENOTATION',36X,'RUN',5X,'=',1X,A//1X,
     1'NYQUIST CUTOFF FREQUENCY (HZ)',21X,'CFR',5X,'=',1PD15.5//1X,
     2'NUMBER OF LAGS',36X,'LAGS',4X,'=',I5//1X,
     3'UNDAMPED RESONANCE FREQUENCY (HZ)',17X,'FR0',5X,'=',D15.5//1X,
     4'DECAY RATIO',39X,'DR',6X,'=',D15.5//1X,
     5'INITIAL NUMBER OF PLOT FILE',23X,'IPLOT',3X,'=',I5//1X,
     6'GIVEN NUMBER OF JOBS',30X,'JOBMAX',2X,'=',I5)
 2010 FORMAT(1X,'ACCEPTED NUMBER OF JOBS',27X,'JOBMAX',2X,'=',I5/)
 2015 FORMAT(1X,'CUTOFF FREQUENCIES (HZ) :',5X,'JOB',6X,'FRL',11X,
     1'FRH',7X,'BACKGR (0/0)',5X,'SLOPE',
     21P/<FRLHMAX>(/32X,I2,4D14.5))
 2020 FORMAT('1DECAY CONSTANT (1/SEC)',28X,'FLAMDA',2X,'=',1PD15.5//1X,
     1'UNDAMPED RESONANCE FREQUENCY (HZ) (GIVEN)',9X,'FR0',5X,'=',
     2D15.5/1X,
     3'ACF RESONANCE FREQUENCY (HZ)',22X,'FR0C',4X,'=',D15.5/1X,
     4'PSD RESONANCE FREQUENCY (HZ)',22X,'FR0R',4X,'=',D15.5//1X,
     5'AMPLITUDE FACTOR ACF',30X,'C',7X,'=',D15.5//1X,
     6'AMPLITUDE FACTOR PSD',30X,'A',7X,'=',D15.5//1X,
     7'PEAK AMPLITUDE IN THE PSD',25X,'PEAKMAX =',D15.5/)
 2025 FORMAT(1X,'EQUIVALENCES IN THE ESTIMATION PROCEDURE :'//11X,
     1'TRANSFORM SIZE (SEGMENT LENGTH)',9X,'NT',6X,'=',I5/11X,
     2'SAMPLING FREQUENCY (HZ)',17X,'SFR',5X,'=',1PD15.5/11X,
     3'RANGE OF THE ACF (SEC)',18X,'TAUMAX',2X,'=',D15.5//1X,
     4'SAMPLING INTERVAL FOR ACF (SEC)',19X,'DTAU',4X,'=',D15.5)
 2030 FORMAT('1NUMBER OF JOB',37X,'JOB',5X,'=',I5//1X,
     1'PLOT OUTPUT FILE',34X,'FILE3',3X,'=',1X,A//1X,
     2'DATA OUTPUT FILE',34X,'FILE4',3X,'=',1X,A//1X,
     3'LOWER CUTOFF FREQUENCY (HZ)',23X,'FRL',5X,'=',1PD15.5//1X,
     4'UPPER CUTOFF FREQUENCY (HZ)',23X,'FRH',5X,'=',D15.5//1X,
     5'RELATIVE BACKGROUND AMPLITUDE ADMIXTURE (0/0)',5X,
     6'BACKGR',2X,'=',D15.5//1X,
     7'BACKGROUND SLOPE INDEX',28X,'SLOPE',3X,'=',
     8D15.5//1X,
     9'BACKGROUND AMPLITUDE (ABSOLUTE VALUE)',13X,'B',7X,'=',
     AD15.5//1X,
     B'BACKGROUND DECAY CONSTANT (SEC/RAD)',15X,'ALPHA',3X,'=',
     CD15.5)
 2100 FORMAT(////11X,'OPENING ERROR ! FILE = ',I2)
 2105 FORMAT(////11X,'READ ERROR IN FILE1 !')
 2110 FORMAT(////11X,'ERRANEOUS PARAMETER DATA !')
 2115 FORMAT(////11X,'E N D')
```

```
 3000 FORMAT(1X,I4,3X,1P,6E12.5)
 4000 FORMAT(1X,I4,1P,2E12.4)
C
C     INPUT PARAMETER DATA
C
      OPEN(UNIT=2,FILE=FILE2,STATUS='NEW',DEFAULTFILE='DIRINPUT')
      WRITE(2,2000) FILE1,FILE2,FILE3,FILE4
      NF = 1
      OPEN(UNIT=1,FILE=FILE1,STATUS='OLD',DEFAULTFILE='DIRINPUT',
     1ERR=100)
      READ(1,1000,ERR=105,END=105) RUN,CFR,LAGS,FR0,DR,IPLOT,JOBMAX
      IF(CFR.LT.5.D0) CFR=5.D0
      IF(CFR.GT.1.D1) CFR=1.D1
      IF(LAGS.LT.20) LAGS=20
      IF(LAGS.GT.LAGMAX) LAGS=LAGMAX
      IF(FR0.LT.1.D-2*CFR) FR0=1.D-2*CFR
      IF(FR0.GT.5.D-1*CFR) FR0=5.D-1*CFR
      IF(DR.LT.2.D-1) DR=2.D-1
      IF(DR.GT.9.8D-1) DR=9.8D-1
      IF(IPLOT.LT.1) IPLOT=1
      IF(IPLOT.GT.1000-FRLHMAX) IPLOT=1000-FRLHMAX
      IF(JOBMAX.LT.1) JOBMAX=1
      IF(JOBMAX.GT.FRLHMAX) JOBMAX=FRLHMAX
      WRITE(2,2005) RUN,CFR,LAGS,FR0,DR,IPLOT,JOBMAX
      DO 1 JOB=1,JOBMAX
    1 READ(1,1005,ERR=105,END=110) FRL(JOB),FRH(JOB),
     1BACKGR(JOB),SLOPE(JOB)
      GO TO 115
  110 JOBMAX = JOB-1
  115 WRITE(2,2010) JOBMAX
      IF(JOBMAX.LT.1) GO TO 105
      WRITE(2,2015) (JOB,FRL(JOB),FRH(JOB),BACKGR(JOB),
     1SLOPE(JOB),JOB=1,JOBMAX)
      DO 2 JOB=1,JOBMAX
      IF(FRL(JOB).LT.0.D0.OR.FRH(JOB).GT.CFR) GO TO 120
      IF(FRL(JOB).GT.8.D-1*FR0.OR.FRH(JOB).LT.1.2D0*FR0) GO TO 120
      IF(BACKGR(JOB).LT.0.D0.OR.DABS(SLOPE(JOB)).GT.6.D0) GO TO 120
    2 CONTINUE
      CLOSE(UNIT=1,STATUS='KEEP')
      DPI = 2.D0*PI
      OMEGA0 = DPI*FR0
      CALL LAMDA(DPI,OMEGA0,DR,FLAMDA)
      OMEGAC = DSQRT(OMEGA0**2-FLAMDA**2)
      OMEGAR = DSQRT(OMEGA0**2-2.D0*FLAMDA**2)
      FR0C = OMEGAC/DPI
      FR0R = OMEGAR/DPI
      A = 4.D0*C*FLAMDA*OMEGA0**2
      A1 = A/PI
      PEAKMAX = A*FINT(OMEGAR)
      WRITE(2,2020) FLAMDA,FR0,FR0C,FR0R,C,A,PEAKMAX
      SFR = 2.D0*CFR
      TAUMAX = DFLOTJ(NT)/SFR
      DTAU = TAUMAX/DFLOTJ(LAGS)
      WRITE(2,2025) NT,SFR,TAUMAX,DTAU
C
C     COMPUTATION OF THE IDEAL ACF
C
```

```
      DO 3 N=1,LAGS+1
      K = N+LAGS
      TAU(K) = DTAU*DFLOTJ(N-1)
    3 CALL RIDEAL(TAU(K),RXXI(N),ENV(N))
C
C     COMPUATION OF THE CORRECTED ACF -  DATA OUTPUT
C
      DO 10 JOB=1,JOBMAX
      NF = 3
      ENCODE(3,'(I3)',FILE3(9:11)) IPLOT
      OPEN(UNIT=3,FILE=FILE3,STATUS='NEW',DEFAULTFILE='DIRINPUT',
     1ERR=100)
      NF = 4
      ENCODE(3,'(I3)',FILE4(9:11)) IPLOT
      OPEN(UNIT=4,FILE=FILE4,STATUS='NEW',DEFAULTFILE='DIRINPUT',
     1ERR=100)
      OMEGAL = DPI*FRL(JOB)
      OMEGAH = DPI*FRH(JOB)
      CALL GROUND(BACKGR(JOB),SLOPE(JOB),PEAKMAX,B,ALPHA)
      B1 = B*DEXP(ALPHA*(OMEGAR-OMEGAL))/PI
      WRITE(2,2030) JOB,FILE3,FILE4,FRL(JOB),FRH(JOB),BACKGR(JOB),
     1SLOPE(JOB),B,ALPHA
      DO 11 N=1,LAGS+1
      K = N+LAGS
      CALL RCOR(TAU(K),RXXI(N),RXXC(K))
      RXXF(K) = RXXC(K)+B1*BGR(TAU(K),ALPHA)
   11 WRITE(3,3000) N,SNGL(TAU(K)),SNGL(RXXI(N)),SNGL(ENV(N)),
     1SNGL(-ENV(N)),SNGL(RXXC(K)),SNGL(RXXF(K))
      K = 2*LAGS+2
      DO 12 N=1,LAGS
      TAU(N) = -TAU(K-N)
      RXXC(N) = RXXC(K-N)
   12 RXXF(N) = RXXF(K-N)
      DO 13 N=1,2*LAGS+1
   13 WRITE(4,4000) N,SNGL(TAU(N)),SNGL(RXXF(N))
      IPLOT = IPLOT+1
      CLOSE(UNIT=3,STATUS='KEEP')
   10 CLOSE(UNIT=4,STATUS='KEEP')
      WRITE(2,2115)
      STOP
  100 WRITE(2,2100) NF
      STOP
  105 WRITE(2,2105)
      STOP
  120 WRITE(2,2110)
      STOP
      END
C=============================================================
      SUBROUTINE LAMDA(DPI,OMEGA0,DR,FLAMDA)
C
C     LAMDA CALCULATES FLAMDA FROM OMEGA0 AND DR.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      X1 = OMEGA0**2
      X2 = (DLOG(DR))**2
      FLAMDA = X1*X2/(DPI**2+X2)
      FLAMDA = DSQRT(FLAMDA)
      RETURN
```

```
      END
C=========================================================
      SUBROUTINE RIDEAL(TAU,RXXI,ENV)
C
C     RIDEAL CALCULATES THE IDEAL AUTO-CORRELATION FUNCTION
C     AND ITS ENVELOPE FOR A VALUE OF TAU.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/PAR/A1,C,OMEGA0,OMEGAC,FLAMDA
      SAVE N,FAC
      DATA N/0/
      IF(N.GT.0) GO TO 100
      FAC = FLAMDA/OMEGAC
      N = 1
  100 ENV = C*DEXP(-FLAMDA*TAU)
      X1 = OMEGAC*TAU
      RXXI = ENV*(DCOS(X1)+FAC*DSIN(X1))
      RETURN
      END
C=========================================================
      SUBROUTINE RCOR(TAU,RXXI,RXXC)
C
C     RCOR CALCULATES THE CORRECTED AUTO-CORRELATION FUNCTION
C     FROM THE IDEAL AUTO-CORRELATION FUNCTION.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/PAR/A1,C,OMEGA0,OMEGAC,FLAMDA
      COMMON/FR/OMEGAL,OMEGAH
      SAVE ERRABS,ERREL,IWEIGH,ZERO
      DATA ERRABS,ERREL,IWEIGH,ZERO/1.D-7,1.D-7,1,0.D0/
      EXTERNAL FINT
      CALL DQDAWO(FINT,ZERO,OMEGAL,IWEIGH,TAU,ERRABS,ERREL,RESULTL,
     1ERREST)
      CALL DQDAWF(FINT,OMEGAH,IWEIGH,TAU,ERRABS,RESULTH,ERREST)
      RXXC = RXXI-A1*(RESULTL+RESULTH)
      RETURN
      END
C=========================================================
      FUNCTION FINT(OMEGA)
C
C     FINT IS THE FUNCTION TO BE INTEGRATED.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/PAR/A1,C,OMEGA0,OMEGAC,FLAMDA
      X1 = OMEGA**2
      X2 = (X1-OMEGA0**2)**2+4.D0*X1*FLAMDA**2
      FINT = 1.D0/X2
      RETURN
      END
C=========================================================
      SUBROUTINE GROUND(BACKGR,SLOPE,PEAKMAX,B0,ALPHA)
C
C     GROUND DETERMINES THE BACKGROUND AMPLITUDE B0 AND THE
C     DECAY CONSTANT ALPHA.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/CONTROL/IBGR
```

```
      COMMON/CONST/DPI
      IF(BACKGR.GT.0.D0) GO TO 100
      IBGR = 0
      BACKGR = 0.D0
      SLOPE = 0.D0
      B0 = 0.D0
      ALPHA = 0.D0
      RETURN
  100 B0 = 1.D-2*BACKGR*PEAKMAX
      IF(SLOPE.NE.0.D0) GO TO 105
      IBGR = 1
      ALPHA = 0.D0
      RETURN
  105 IBGR = 2
      ALPHA = -SLOPE*DLOG(1.D1)/DPI
      RETURN
      END
C===========================================================
      FUNCTION BGR(TAU,ALPHA)
C
C     BGR CALCULATES THE BACKGROUND FUNCTION IN THE TIME DOMAIN.
C
C     IBGR = 0 : NO BACKGROUND,
C     IBGR = 1 : CONSTANT BACKGROUND IN THE FREQUENCYY DOMAIN,
C     IBGR = 2 : BACKGROUND WITH A GIVEN SLOPE IN THE FREQUENCY DOMAIN.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/CONTROL/IBGR
      COMMON/FR/OMEGAL,OMEGAH
      IF(IBGR-1) 50,55,200
   50 BGR = 0.D0
      RETURN
   55 IF(TAU.GT.0.D0) GO TO 100
      BGR = OMEGAH-OMEGAL
      RETURN
  100 BGR = (DSIN(OMEGAH*TAU)-DSIN(OMEGAL*TAU))/TAU
      RETURN
  200 DOMEGA = OMEGAH-OMEGAL
      X1 = DEXP(-ALPHA*DOMEGA)
      IF(TAU.GT.0.D0) GO TO 210
      IF(DABS(ALPHA)*DOMEGA.LT.1.D-20) GO TO 205
      BGR = (1.D0-X1)/ALPHA
      RETURN
  205 BGR = DOMEGA
      RETURN
  210 X2 = ALPHA**2+TAU**2
      XL = OMEGAL*TAU
      XH = OMEGAH*TAU
      BGR = ALPHA*(DCOS(XL)-X1*DCOS(XH))
      BGR = BGR-TAU*(DSIN(XL)-X1*DSIN(XH))
      BGR = BGR/X2
      RETURN
      END
```

# LISTING

## ACFOSC6

```
      PROGRAM ACFOSC6
C
C     THE CODE REFERS TO A SECOND-ORDER OSCILLATOR MODEL EXCITED BY RAND
C     WHITE NOISE. IT CALCULATES THE IDEAL AUTO-CORRELATION FUNCTION
C     (ACF) AND THE ACF WHICH RESULTS FROM PREVIOUSLY BAND-PASS FILTERED
C     SIGNALS. THE BAND-PASS FILTER IS ASSUMED TO BE RECTANGULAR.
C     THE RESONANCE FREQUENCY OF THE PEAK IN THE PSD MUST LIE WITHIN THE
C     FILTER CUTOFF FREQUENCIES.
C
C     THE CORRECTED AUTO-CORRELATION IS CALCULATED FROM THE IDEAL
C     AUTO-CORRELATION FUNCTION. IT ALLOWS THE ADDITION OF BACKGROUND
C     SPECIFIED IN THE FREQUENCY DOMAIN.
C     THE TWO-SIDED CORRECTED ACF INCLUDING BACKGROUND IS READ OUT ON
C     A SEPARATE FILE (FILE4) WITH THE SAME FORMAT AS USED IN THE CODE A
C     IF ONE SETS LAGS = 256, THESE DATA CAN BE USED FOR TESTING THE DIF
C     VERSIONS OF THE CODE ACFIT.
C
C     THE CODE ASSUMES THAT THE DIGITAL ESTIMATION OF THE ACF IS BASED
C     ON THE INDIRECT METHOD BY FAST FOURIER TRANSFORM TECHNIQUES WITH
C     ZERO-PADDING. IT REQUIRES THE INPUT OF THE NYQUIST CUTOFF
C     FREQUENCY CFR. THE TRANSFORM SIZE (SEGMENT LENGTH) NT IS GIVEN
C     IN THE DATA DECLARATION. CFR AND NT DETERMINE THE RANGE TAUMAX
C     OF THE ACF. FOR A HIGH RESOLUTION CALCULATION OF THE ACF THE
C     SAMPLING INTERVAL DTAU IS GIVEN BY TAUMAX/LAGS, WHERE LAGS IS
C     THE NUMBER OF LAGS.
C
C     WITHIN A RUN A LIMITED NUMBER OF ACF'S (CALLED JOBS) WITH
C     DIFFERENT PAIRS OF CUTOFF FREQUENCIES AND DIFFERENT VALUES
C     FOR A TWO-PARAMETRIC BACKGROUND CAN BE CALCULATED.
C
C     THE CODE REQUIRES IMSL ROUTINES.
C
C     CODE WRITTEN BY K.BEHRINGER, OCTOBER 1999, FOR BATCH OPERATION
C     ON THE DEC-ALPHA 2100 COMPUTER.
C
C     PARAMETERS :
C
C          LAGMAX  = MAXIMUM NUMBER OF LAGS
C          FRLHMAX = MAXIMUM NUMBER OF CUTOFF FREQUENCY PAIRS
C
C
C     DECLARATIONS
C
      IMPLICIT REAL*8 (A-H,O-Z)
      INTEGER*4 FRLHMAX
      PARAMETER (LAGMAX=1000,FRLHMAX=10)
      CHARACTER*50 RUN,FILE1,FILE2,FILE3,FILE4
      DIMENSION TAU(2*LAGMAX+1),RXXI(LAGMAX+1),ENV(LAGMAX+1),
     1RXXC(2*LAGMAX+1),RXXF(2*LAGMAX+1),FRL(FRLHMAX),FRH(FRLHMAX),
     2BACKGR(FRLHMAX),SLOPE(FRLHMAX),TMIN(3),RXXMIN(3),
     3TMAX(3),RXXMAX(3)
      COMMON/PAR/A1,C,OMEGA0,OMEGAC,FLAMDA
```

```
      COMMON/FR/OMEGAL,OMEGAH
      COMMON/CONST/DPI
      DATA C,PI,NT/1.D0,3.141592654D0,256/,
     1FILE1/'ACFOSC6.IN                                  '/,
     2FILE2/'ACFOSC6.PRT                                 '/,
     3FILE3/'ACFOSC6_^^^.PLO                             '/,
     4FILE4/'ACFOSC6_^^^.DAT                             '/
C
C     FORMATS
C
 1000 FORMAT(A/D12.5/I4/2(D12.5/)I3/I2)
 1005 FORMAT(4D12.5)
 2000 FORMAT('1',40X,'P R O G R A M     A C F O S C 6'///1X,
     1'FILES : INPUT PARAMETER DATA :  FILE1 = ',A/9X,
     2'PRINT OUTPUT',9X,':  FILE2 = ',A/9X,
     3'PLOT OUTPUT',10X,':  FILE3 = ',A/9X,
     4'DATA OUTPUT',10X,':  FILE4 = ',A//)
 2005 FORMAT(1X,'RUN DENOTATION',36X,'RUN',5X,'=',1X,A//1X,
     1'NYQUIST CUTOFF FREQUENCY (HZ)',21X,'CFR',5X,'=',1PD15.5//1X,
     2'NUMBER OF LAGS',36X,'LAGS',4X,'=',I5//1X,
     3'UNDAMPED RESONANCE FREQUENCY (HZ)',17X,'FR0',5X,'=',D15.5//1X,
     4'DECAY RATIO',39X,'DR',6X,'=',D15.5//1X,
     5'INITIAL NUMBER OF PLOT FILE',23X,'IPLOT',3X,'=',I5//1X,
     6'GIVEN NUMBER OF JOBS',30X,'JOBMAX',2X,'=',I5)
 2010 FORMAT(1X,'ACCEPTED NUMBER OF JOBS',27X,'JOBMAX',2X,'=',I5/)
 2015 FORMAT(1X,'CUTOFF FREQUENCIES (HZ) :',5X,'JOB',6X,'FRL',11X,
     1'FRH',7X,'BACKGR (0/0)',5X,'SLOPE',
     21P/<FRLHMAX>(/32X,I2,4D14.5))
 2020 FORMAT('1DECAY CONSTANT (1/SEC)',28X,'FLAMDA',2X,'=',1PD15.5//1X,
     1'RESONANCE QUALITY FACTOR',26X,'QR',6X,'=',D15.5//1X,
     2'PSD RESONANCE FREQUENCY (HZ) (GIVEN)',14X,'FR0',5X,'=',
     3D15.5/1X,
     4'ACF RESONANCE FREQUENCY (HZ)',22X,'FR0C',4X,'=',D15.5//1X,
     5'AMPLITUDE FACTOR ACF',30X,'C',7X,'=',D15.5//1X,
     6'AMPLITUDE FACTOR PSD',30X,'A',7X,'=',D15.5//1X,
     7'PEAK AMPLITUDE IN THE PSD',25X,'PEAKMAX =',D15.5/)
 2025 FORMAT(1X,'EQUIVALENCES IN THE ESTIMATION PROCEDURE :'//11X,
     1'TRANSFORM SIZE (SEGMENT LENGTH)',9X,'NT',6X,'=',I5/11X,
     2'SAMPLING FREQUENCY (HZ)',17X,'SFR',5X,'=',1PD15.5/11X,
     3'RANGE OF THE ACF (SEC)',18X,'TAUMAX',2X,'=',D15.5//1X,
     4'SAMPLING INTERVAL FOR ACF (SEC)',19X,'DTAU',4X,'=',D15.5/)
 2026 FORMAT(1X,'FIRST THREE MINIMA AND MAXIMA OF THE IDEAL ACF :'
     1//31X,'TAUMIN (SEC)',5X,'RXXMIN',5X,'TAUMAX (SEC)',5X,
     2'RXXMAX',1P/3(/29X,4D14.5))
 2030 FORMAT('1NUMBER OF JOB',37X,'JOB',5X,'=',I5//1X,
     1'PLOT OUTPUT FILE',34X,'FILE3',3X,'=',1X,A//1X,
     2'DATA OUTPUT FILE',34X,'FILE4',3X,'=',1X,A//1X,
     3'LOWER CUTOFF FREQUENCY (HZ)',23X,'FRL',5X,'=',1PD15.5//1X,
     4'UPPER CUTOFF FREQUENCY (HZ)',23X,'FRH',5X,'=',D15.5//1X,
     5'RELATIVE BACKGROUND AMPLITUDE ADMIXTURE (0/0)',5X,
     6'BACKGR',2X,'=',D15.5//1X,
     7'BACKGROUND SLOPE INDEX',28X,'SLOPE',3X,'=',
     8D15.5//1X,
     9'BACKGROUND AMPLITUDE (ABSOLUTE VALUE)',13X,'B',7X,'=',
     AD15.5//1X,
     B'BACKGROUND DECAY CONSTANT (SEC/RAD)',15X,'ALPHA',3X,'=',
     CD15.5)
 2100 FORMAT(////11X,'OPENING ERROR ! FILE = ',I2)
```

```
 2105 FORMAT(////11X,'READ ERROR IN FILE1 !')
 2110 FORMAT(////11X,'ERRANEOUS PARAMETER DATA !')
 2115 FORMAT(////11X,'E N D')
 3000 FORMAT(1X,I4,3X,1P,6E12.5)
 4000 FORMAT(1X,I4,1P,2E12.4)
C
C     INPUT PARAMETER DATA
C
      OPEN(UNIT=2,FILE=FILE2,STATUS='NEW',DEFAULTFILE='DIRINPUT')
      WRITE(2,2000) FILE1,FILE2,FILE3,FILE4
      NF = 1
      OPEN(UNIT=1,FILE=FILE1,STATUS='OLD',DEFAULTFILE='DIRINPUT',
     1ERR=100)
      READ(1,1000,ERR=105,END=105) RUN,CFR,LAGS,FR0,DR,IPLOT,JOBMAX
      IF(CFR.LT.5.D0) CFR=5.D0
      IF(CFR.GT.1.D1) CFR=1.D1
      IF(LAGS.LT.20) LAGS=20
      IF(LAGS.GT.LAGMAX) LAGS=LAGMAX
      IF(FR0.LT.1.D-2*CFR) FR0=1.D-2*CFR
      IF(FR0.GT.5.D-1*CFR) FR0=5.D-1*CFR
      IF(DR.LT.1.D-1) DR=1.D-1
      IF(DR.GT.9.9D-1) DR=9.9D-1
      IF(IPLOT.LT.1) IPLOT=1
      IF(IPLOT.GT.1000-FRLHMAX) IPLOT=1000-FRLHMAX
      IF(JOBMAX.LT.1) JOBMAX=1
      IF(JOBMAX.GT.FRLHMAX) JOBMAX=FRLHMAX
      WRITE(2,2005) RUN,CFR,LAGS,FR0,DR,IPLOT,JOBMAX
      DO 1 JOB=1,JOBMAX
    1 READ(1,1005,ERR=105,END=110) FRL(JOB),FRH(JOB),
     1BACKGR(JOB),SLOPE(JOB)
      GO TO 115
  110 JOBMAX = JOB-1
  115 WRITE(2,2010) JOBMAX
      IF(JOBMAX.LT.1) GO TO 105
      WRITE(2,2015) (JOB,FRL(JOB),FRH(JOB),BACKGR(JOB),
     1SLOPE(JOB),JOB=1,JOBMAX)
      DO 2 JOB=1,JOBMAX
      IF(FRL(JOB).LT.0.D0.OR.FRH(JOB).GT.CFR) GO TO 120
      IF(FRL(JOB).GT.8.D-1*FR0.OR.FRH(JOB).LT.1.2D0*FR0) GO TO 120
      IF(BACKGR(JOB).LT.0.D0.OR.DABS(SLOPE(JOB)).GT.6.D0) GO TO 120
    2 CONTINUE
      CLOSE(UNIT=1,STATUS='KEEP')
      DPI = 2.D0*PI
      OMEGA0 = DPI*FR0
      CALL LAMDA(DPI,OMEGA0,DR,FLAMDA,QR)
      OMEGAC = DSQRT(OMEGA0**2-FLAMDA**2)
      FR0C = OMEGAC/DPI
      A1 = 4.D0*C*FLAMDA
      A = A1/OMEGA0**2
      PEAKMAX = A1*FINT(OMEGA0)
      A1 = A1/PI
      WRITE(2,2020) FLAMDA,QR,FR0,FR0C,C,A,PEAKMAX
      SFR = 2.D0*CFR
      TAUMAX = DFLOTJ(NT)/SFR
      DTAU = TAUMAX/DFLOTJ(LAGS)
      WRITE(2,2025) NT,SFR,TAUMAX,DTAU
      CALL MINMAX(PI,OMEGAC,QR,TMIN,RXXMIN,TMAX,RXXMAX)
```

```
      WRITE(2,2026) (TMIN(N),RXXMIN(N),TMAX(N),RXXMAX(N),N=1,3)
C
C     COMPUTATION OF THE IDEAL ACF
C
      DO 3 N=1,LAGS+1
      K = N+LAGS
      TAU(K) = DTAU*DFLOTJ(N-1)
    3 CALL RIDEAL(TAU(K),RXXI(N),ENV(N))
C
C     COMPUATION OF THE CORRECTED ACF -  DATA OUTPUT
C
      DO 10 JOB=1,JOBMAX
      NF = 3
      ENCODE(3,'(I3)',FILE3(9:11)) IPLOT
      OPEN(UNIT=3,FILE=FILE3,STATUS='NEW',DEFAULTFILE='DIRINPUT',
     1ERR=100)
      NF = 4
      ENCODE(3,'(I3)',FILE4(9:11)) IPLOT
      OPEN(UNIT=4,FILE=FILE4,STATUS='NEW',DEFAULTFILE='DIRINPUT',
     1ERR=100)
      OMEGAL = DPI*FRL(JOB)
      OMEGAH = DPI*FRH(JOB)
      CALL GROUND(BACKGR(JOB),SLOPE(JOB),PEAKMAX,B,ALPHA)
      B1 = B*DEXP(ALPHA*(OMEGA0-OMEGAL))/PI
      WRITE(2,2030) JOB,FILE3,FILE4,FRL(JOB),FRH(JOB),BACKGR(JOB),
     1SLOPE(JOB),B,ALPHA
      DO 11 N=1,LAGS+1
      K = N+LAGS
      CALL RCOR(TAU(K),RXXI(N),RXXC(K))
      RXXF(K) = RXXC(K)+B1*BGR(TAU(K),ALPHA)
   11 WRITE(3,3000) N,SNGL(TAU(K)),SNGL(RXXI(N)),SNGL(ENV(N)),
     1SNGL(-ENV(N)),SNGL(RXXC(K)),SNGL(RXXF(K))
      K = 2*LAGS+2
      DO 12 N=1,LAGS
      TAU(N) = -TAU(K-N)
      RXXC(N) = RXXC(K-N)
   12 RXXF(N) = RXXF(K-N)
      DO 13 N=1,2*LAGS+1
   13 WRITE(4,4000) N,SNGL(TAU(N)),SNGL(RXXF(N))
      IPLOT = IPLOT+1
      CLOSE(UNIT=3,STATUS='KEEP')
   10 CLOSE(UNIT=4,STATUS='KEEP')
      WRITE(2,2115)
      STOP
  100 WRITE(2,2100) NF
      STOP
  105 WRITE(2,2105)
      STOP
  120 WRITE(2,2110)
      STOP
      END
C=============================================================
      SUBROUTINE LAMDA(DPI,OMEGA0,DR,FLAMDA,QR)
C
C     LAMDA CALCULATES FLAMDA AND QR FROM OMEGA0 AND DR.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      QR = 5.D-1*DSQRT(1.D0+DPI**2/(DLOG(DR))**2)
```

```
      FLAMDA = 5.D-1*OMEGA0/QR
      RETURN
      END
C===========================================================
      SUBROUTINE RIDEAL(TAU,RXXI,ENV)
C
C     RIDEAL CALCULATES THE IDEAL AUTO-CORRELATION FUNCTION
C     AND ITS ENVELOPE FOR A VALUE OF TAU.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/PAR/A1,C,OMEGA0,OMEGAC,FLAMDA
      SAVE N,FAC
      DATA N/0/
      IF(N.GT.0) GO TO 100
      FAC = FLAMDA/OMEGAC
      N = 1
  100 ENV = C*DEXP(-FLAMDA*TAU)
      X1 = OMEGAC*TAU
      RXXI = ENV*(DCOS(X1)-FAC*DSIN(X1))
      RETURN
      END
C===========================================================
      SUBROUTINE MINMAX(PI,OMEGAC,QR,TAUMIN,RXXMIN,TAUMAX,RXXMAX)
C
C     MINMAX CALCULATES THE FIRST THREE MINIMA AND MAXIMA OF THE
C     IDEAL ACF.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION TAUMIN(3),RXXMIN(3),TAUMAX(3),RXXMAX(3)
      S = 2.D0*DACOS(5.D-1/QR)
      DO 1 N=1,3
      TAUMIN(N) = (2.D0*PI*DFLOTJ(N-1)+S)/OMEGAC
      TAUMAX(N) = (PI*DFLOTJ(2*N-1)+S)/OMEGAC
      CALL RIDEAL(TAUMIN(N),RXXMIN(N),ENV)
    1 CALL RIDEAL(TAUMAX(N),RXXMAX(N),ENV)
      RETURN
      END
C===========================================================
      SUBROUTINE RCOR(TAU,RXXI,RXXC)
C
C     RCOR CALCULATES THE CORRECTED AUTO-CORRELATION FUNCTION
C     FROM THE IDEAL AUTO-CORRELATION FUNCTION.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/PAR/A1,C,OMEGA0,OMEGAC,FLAMDA
      COMMON/FR/OMEGAL,OMEGAH
      SAVE ERRABS,ERREL,IWEIGH,ZERO
      DATA ERRABS,ERREL,IWEIGH,ZERO/1.D-8,1.D-8,1,0.D0/
      EXTERNAL FINT
      CALL DQDAWO(FINT,ZERO,OMEGAL,IWEIGH,TAU,ERRABS,ERREL,RESULTL,
     1ERREST)
      CALL DQDAWF(FINT,OMEGAH,IWEIGH,TAU,ERRABS,RESULTH,ERREST)
      RXXC = RXXI-A1*(RESULTL+RESULTH)
      RETURN
      END
C===========================================================
      FUNCTION FINT(OMEGA)
```

```
C
C     FINT IS THE FUNCTION TO BE INTEGRATED.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/PAR/A1,C,OMEGA0,OMEGAC,FLAMDA
      X1 = OMEGA**2
      X2 = (X1-OMEGA0**2)**2+4.D0*X1*FLAMDA**2
      FINT = X1/X2
      RETURN
      END
C=========================================================
      SUBROUTINE GROUND(BACKGR,SLOPE,PEAKMAX,B0,ALPHA)
C
C     GROUND DETERMINES THE BACKGROUND AMPLITUDE B0 AND THE
C     DECAY CONSTANT ALPHA.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/CONTROL/IBGR
      COMMON/CONST/DPI
      IF(BACKGR.GT.0.D0) GO TO 100
      IBGR = 0
      BACKGR = 0.D0
      SLOPE = 0.D0
      B0 = 0.D0
      ALPHA = 0.D0
      RETURN
  100 B0 = 1.D-2*BACKGR*PEAKMAX
      IF(SLOPE.NE.0.D0) GO TO 105
      IBGR = 1
      ALPHA = 0.D0
      RETURN
  105 IBGR = 2
      ALPHA = -SLOPE*DLOG(1.D1)/DPI
      RETURN
      END
C=========================================================
      FUNCTION BGR(TAU,ALPHA)
C
C     BGR CALCULATES THE BACKGROUND FUNCTION IN THE TIME DOMAIN.
C
C     IBGR = 0 : NO BACKGROUND,
C     IBGR = 1 : CONSTANT BACKGROUND IN THE FREQUENCYY DOMAIN,
C     IBGR = 2 : BACKGROUND WITH A GIVEN SLOPE IN THE FREQUENCY DOMAIN.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/CONTROL/IBGR
      COMMON/FR/OMEGAL,OMEGAH
      IF(IBGR-1) 50,55,200
   50 BGR = 0.D0
      RETURN
   55 IF(TAU.GT.0.D0) GO TO 100
      BGR = OMEGAH-OMEGAL
      RETURN
  100 BGR = (DSIN(OMEGAH*TAU)-DSIN(OMEGAL*TAU))/TAU
      RETURN
  200 DOMEGA = OMEGAH-OMEGAL
      X1 = DEXP(-ALPHA*DOMEGA)
      IF(TAU.GT.0.D0) GO TO 210
```

```
      IF(DABS(ALPHA)*DOMEGA.LT.1.D-20) GO TO 205
      BGR = (1.D0-X1)/ALPHA
      RETURN
  205 BGR = DOMEGA
      RETURN
  210 X2 = ALPHA**2+TAU**2
      XL = OMEGAL*TAU
      XH = OMEGAH*TAU
      BGR = ALPHA*(DCOS(XL)-X1*DCOS(XH))
      BGR = BGR-TAU*(DSIN(XL)-X1*DSIN(XH))
      BGR = BGR/X2
      RETURN
      END
```

# 10. PROGRAM ACFIT6 AND PROGRAM ACFIT7

Precision : double

Operation : foreground, background

Required auxiliary routines : from IMSL : DQDAWO, DQDAWF

Purpose of the programs :

Least-Squares Fit of the Model Auto-Correlation Function to the Estimated Auto-Correlation Function

Feature Summary :

- The code ACFIT6 refers to model A, the code ACFIT7 to model B. The parameter data input is the same for both codes.
- The lag time range of the fit is kept variable. Within a run of each code a limited number of lag time ranges can be given. Each case is called a job.
- 3 options for the kind of the least-squares fit.
- 10 options for weighting the square errors.
- The least-squares fit procedure requires gradient functions of the fit function.
- There are many branchings, if anomaleous fit conditions appear.

## 10.1 Mathematical Background

The code ACFIT6 performs a least-squares fitting of the oscillator model A to an auto-correlation function (ACF) estimated on unfiltered or previously band-pass filtered signal data. The code ACFIT7 does the same for the oscillator model B. Both models are described in Section 2. The codes are experimental and higly modularly designed. They are specially related to the input of ACF data estimated by the code ACCF1 (Section 7) from the file ACCF0''.PLO, on which two-sided ACF data are written. They take automatically the right-hand sided part (subroutine INPUT). The ACFs cover lag times up to the maximum value $\tau_{max}$. With ordinarily used values for the segment length $N_s$=256 data points and the sampling frequency of 12.5 Hz, $\tau_{max}$ has the value of 20.48 sec. The ACF data point at the lag number $N_s$ is undefined and is set equal zero in ACCF1. For a run of ACFIT6 or ACFIT7, a set of fit range parameters r ($0<r<1$) must be given. The integer value $rN_s$ determines the last ACF data point to be included in a fit, and defines the lag time end point $\tau_{end}$ for a fit. The value of $\tau_{end}$ is taken from the read-in ACF data, so that no sampling frequency value must be given. The r values are ascendently ordered in each code. A fit with a specified r value is called a job within a run. There are two reasons for keeping the fit range variable within $\tau_{max}$ :

a) The statistical accuracy of an ACF estimate decreases with increasing lag time values, simply because the number of signal data products involved in averaging decreases. In order to be on the safe side, one should restrict the use of an ACF estimate to about 50 % of $\tau_{max}$. But values of up to 70 % can sometimes be accepted.

b) Even in the case of strong signal filtering, all ACF estimates obtained from the benchmark records show a more or less significant appearance of 'background' after the decay of the main oscillation. This phenomenon is not yet completely understood and may be due to the incompleteness of the used simple models. The high orders required in ARMA modelling may reflect this behaviour. The aim of the codes consists in the attempt to extract approximately the most relevant information by simple physical models. This suggests an individual adaptation of the maximum fit range to the decay ratio value DR. In particular, cases with small DR values can only be fitted over relatively short lag time ranges. This leads to large uncertainties in the results.

The criteria used for determining the optimum fit range require the application of the gliding segment analysis GLSA (Section 7.5) and will be given in the description of the code ACFIT7SA (Section 11). The present codes treat only one experimental ACF in a run.

Referring to the models (Section 2), natural fit parameters are $C_0, \omega_c, \lambda, B_0$ and $\alpha$. From $\lambda$ and $\omega_c$ one obtains the DR. Three fit options are available with respect to the treatment of the background term in the power spectral density (PSD). The option parameter is denoted by IBGR.

- IBGR=0 : Three-parametric fit with $B_0$=0. The assumed exponentially decaying background under the peak in the PSD is considered to be negligibly small. If not, investigations showed that a significant underestimation of the DR occurs.

- IBGR=1 : Four-parametric fit including $B_0$ as a fit parameter, but with a given fixed input value of $\alpha$. The value of $\alpha$ can approximately be obtained from a logarithmic plot of the estimated PSD via a slope index S which is defined by

$$S = \log \frac{B(\omega + 2\pi)}{B(\omega)} \tag{10.1}$$

E.g. a slope index S=-2 means that the background under the peak decays exponentially over 2 decades within 1 Hz. This option is only applicable to 'average' ACFs (see Section 7.5), but has been found to be problematic by fit sensitivity studies with the codes ACFOSC5 and ACFOSC6 (Section 9). If one inputs to the fit codes exactly the same slope index values as given for the source codes, at small fit ranges one can mostly observe a significant missestimation of the DR. But the fit parameter data tend to approach (often weakly oscillating) the correct values with increasing fit range values. The behaviour at small fit ranges may be explained by the competition between the linearly entering fit parameters $C_0$ and $B_0$. The GLSA, however, would require the input of an individual slope index value for each 'short-time' ACF. If one uses only a common value taken approximately from the PSD estimated over the entire record length, the GLSA fails. Such individual S values are not obtainable. The data of PSD estimations over a small number of segments scatter too much.

- IBGR=2 : Five-parametric fit including $B_0$ and $\alpha$ as fit parameters. Such a fit, however, is dangerous if the PSD background is small. $|B_0|$ and $\alpha$ can run to very large values. In order to have a simple criterion for deciding whether the background is small or high, the fit codes calculate from the fit parameter values (for each job) the peak maximum $P_{max}$ in the model PSD by

$$P_{max} = S_{xx}(\omega_r) \tag{10.2}$$

and the background amplitude $B_{0p}$ at $\omega_r$ by

$$B_{0p} = B_0 e^{-\alpha(\omega_r - \omega_L)} \tag{10.3}$$

One can then compare the values $B_0$ at $\omega_L$, and $B_{0p} + P_{max}$ at $\omega_r$ with those in the PSD plot (if the peak width is not very small; otherwise the peak amplitude is strongly biased).

The fit parameters require an initialization. The values of the main fit parameters $C_0$, $\omega_c$ and $\lambda$ are internally initialized (subroutine INIT). $C_0 = \hat{R}_{xx}(\tau = 0)$, where $\hat{R}_{xx}(\tau)$ is the input ACF estimate, $\omega_c / 2\pi = 0.5$ Hz for the cases of unfiltered or weakly filtered signal data, or $\omega_c = (\omega_L + \omega_H)/2$ otherwise, $\lambda$ from a DR=0.6. The filter cutoff frequencies $\omega_L$ and $\omega_H$ are input parameters. Their possible values must exactly be transferred from the code FFTF2 (Section 5) in filtered signal cases. In unfiltered signal cases one has to set $\omega_L$=0 and $\omega_H$=value of the Nyquist cutoff frequency. For the initialization of the PSD background fit parameters, values must be given, They are bypassed for the fit option IBGR=0, where $B_0$ is internally set equal zero. The value to be given for $B_0$ is not critical, since $B_0$ enters lineary in the model ACFs. It is input by the value $b_0$ (>0), so that $B_0 = b_0 \hat{R}_{xx}(\tau = 0)$. Instead of $\alpha$, the slope index S is used, from where $\alpha$ is then internally calculated. The initialized fit parameter values refer to the start of the first job with the smallest value of $\tau_{end}$. They are stored as a basic set to which a reset can be made. Under normal run conditions each following job starts with the fit parameter values estimated from the preceding job. Counteractions are taken if the following anomalous conditions occur :

a) A job is allowed to repeat once with a complete fit parameter reset, if $C_0$<0 or $\lambda$<-0.1 result, or the fit does not converge. A negative value of $C_0$ can happen in cases with strongly filtered signal data. Such a fit is worthless. A negative value of $\lambda$ violates the models, but a small negative value must be admitted with respect to the ACF data scattering in cases with high DR values (limit cycle cases). The evaluation of $P_{max}$ and $B_{0p}$ is restricted to estimated values of DR<1 and $\omega_r > \omega_L$; otherwise these values are set equal zero. Obviously, a job repetition is only allowed, if the job in question has not started with the basic set of the parameter values. A job with a 'bad' or not converging fit is cancelled and each code continues with the next job under a complete fit parameter reset.

b) A fit is interrupted if $\omega_r$ becomes imaginary (ACFIT6, model A) or $\alpha$ reaches a negative value below a given level. The latter can otherwise leads to overflow conditions. The job in question is cancelled and each code continues with the next job with a complete fit parameter reset.

c) The next job starts also with a complete fit parameter reset, if in the preceding job a DR<0.2 has been obtained.

d) A partial reset is provided for the next job with the starting condition $\lambda$=0, if $-0.1 \le \lambda < 0$ appears in the preceding job, or with the basic values of $B_0$ and $\alpha$ (IBGR=2), if $B_0$<0 or $\alpha \ge 10$ has been estimated in the preceding job. Because of data scattering the appearance of small negative values of $B_0$ must be admitted. Most of the anomalous occurrencies are accompanied with messages on the print file.

There are two option parameters, IWEIGHT and IWEND, for weighting the squares for error in a fit. With the option parameter IWEIGHT one can select one of the six implemented weight functions which are of the type :

$$w(\tau) = w_1(\tau) w_0(\tau) \tag{10.4}$$

If IWEND=1, for the values IWEIGHT=1-3, $w_1(\tau) = 1.w_0(\tau)$ has the form :

IWEIGHT= 1 : $w_0(\tau) = 1$ (uniform weighting) (10.5)

2 : $w_0(\tau) = 1 - \tau / \tau_{end}$ (linearly decreasing weighting) (10.6)

3 : $w_0(\tau) = \cos(\pi\tau / 2\tau_{end})$ (cosine-shaped weighting) (10.7)

$$0 \le \tau \le \tau_{end}$$

For IWEIGHT=4-6, $w_0(\tau)$ corresponds to the functions selected by IWEIGHT=1-3. $w_1(\tau)$ permits an initial reduction of weighting, and has the form :

$$w_1(\tau) = w_a + (1 - w_a)\sin(\pi\tau / 2\tau_a); 0 \le \tau < \tau_a << \tau_{end}; 0 \le w_a < 1$$

$$= 1; 0 \ge \tau_a \tag{10.8}$$

$w_a$ and $\tau_a$ are additional input parameters. There values are bypassed if they are not needed.

If one sets IWEND=2, $w_0(\tau)$ will be stretched-out by replacing $\tau_{end}$ by $\tau_{max}$ in equation (10.6) or (10.7). Obviously, the value IWEND has no influence, if IWEIGHT=1 or 4. The most interesting weight function has been found for IWEIGHT=3, IWEND=1.

All fit parameter data of each job and, in addition, the derived data for $A_0$, DR, $P_{max}$ , and the reduced chi-square $\chi^2_R$ (sum of weighted squares for error divided by the number of degrees of freedom) are written on a file which has a run identification number. The data of cancelled jobs are set equal to $10^{30}$ and the job number in question (and $\tau_{end}$ ) can be recognized there.

In the GLSA, a separate run is required for each ACF estimate. This rather tedious procedure can be avoided by using the code ACFIT7SA (Section 11) where for model B the code ACFIT7 is repetetively applied. The auxiliary code ACFITEV5 (Section 13) handles the data matrices either from the different runs of ACFIT6 or ACFIT7, or directly from ACFIT7SA, and calculates averages and standard deviations of each output data for each job number under elimination of the cancelled jobs.

One might believe that least-squares fitting is today a rather trivial procedure. This is not so in the present application. Originally, the IMSL nonlinear regression routine DRNLIN was used in the code development for model A. It was investigated with both options to obtain the gradient of the fit function either by finite differences (IDERIV=0) or by supplied gradient functions (IDERIV=1). The involved numerical integrations (by the IMSL routines DQDAWO and DQDAWF) required an internal parameter setting for high integration accuracy which, for garanteeing fit convergence, suggested to divide the integration interval $(0,\omega_L)$ into the two subintervals $(0,\omega_r)$ and $(\omega_r,\omega_L)$, if (instantaneous) values of $\omega_r$ lie below $\omega_L$. Correspondingly, this split was also done for the interval $(\omega_H,\infty)$. Even under these provisions, many fits did not converge, indicating numerical instabilities. The IMSL fit routine showed several disadvantages in its use for the present problem :

a) It is not easy to make an adaptation to non-default values in the implemented convergence criteria.

b) Fortran texts of IMSL routines are not obtainable. There are mostly interconnections to other IMSL routines which must not explicitly be called. This did not allow a computational optimation with respect to the gradient functions (Section 2.3).

c) A solution of programming returns from called subroutine functions to the main program was not found.

The IMSL fit routine was therefore replaced by the old routine MARFIT which was developed and documented by Behringer (1970). MARFIT is based on a combination of the Gauss-Newton method with a modified version of the gradient-expansion method by Marquardt (1963). This routine is very flexible in the application. It handles only one iteration in the fit procedure. All convergence criteria must be given separately in the calling routine (here subroutine FITMAR). The used criteria are simply based on minimizing $\chi^2_R$ under the constraint of a prescribed smallness of the gradient-expansion coefficient. If this coefficient goes to zero, the fit procedure approaches to the Gauss-Newton method. One has to give input values for the allowed maximum number of iterations and an accuracy parameter for controlling the convergence of $\chi^2_R$. Furthermore, an initial value for the gradient-expansion coefficient is required. Stronger additional convergence criteria for each single fit parameter have not been included. MARFIT requires the support by gradient functions and allows a computational optimation for avoiding calculation repetitions of parts of the fit function which appear in the gradient functions. MARFIT offers also an unscaled co-cariance matrix. Use of it has presently not been made. All experimental ACFs showed very smooth curves. But their shapes can significantly scatter in the GLSA.

The IMSL routine DQDAWF treats a Fourier integral in a semi-infinite integration interval. The integral is approximated by repeated calls to the IMSL routine DQDAWO followed by extrapolation. It happened, but very seldom, that DQDAWF initiated a dump with the fatal error message of no convergence in the case of a large instantaneous $\lambda$ value appearing during a fit. IMSL offers an error handling system which interacts with IMSL routines. In the present version of the codes no use of it has been made for taking counteractions.

### 10.2 References

Behringer K.(1970). Programmbeschreibung der Subroutine MARFIT für einen eindimensionalen Least-Squares Fit von Messdaten mit einer beliebigen Funktion von unabhängigen Parametern, Internal EIR Report TM-PH-373.

Marquardt D.W.(1963). J.Soc.Indust.Appl.Math. Vol.11, No.2, 431.

### 10.3 Files

There are 7 files :

- ACFIT6.IN, ACFIT7.IN (file unit 1)

These files contain the input parameter data. The number of the variables, order and formats are the same for both codes.

- ACFIT6.PRT, ACFIT7.PRT (file unit 2)

These files are meant for printing. They contain (with text) all input parameter data, the estimated fit parameter data together with additional derived data for each job, and messages from the computation progress including those from the occurrence of an anomalous fit condition.

- 'FINPUT' (file unit 3)

This file contains the lag time values and the two-sided estimated ACF data to be input. The data are assumed to be of the REAL*4 type written as column vectors with ascendently ordered lag time values. If there are only right-hand sided data, they will also be accepted (subroutine INPUT). 'FINPUT' is a formal parameter. The file name and the data format are input parameters.

- ACFIT6_'''.DAT, ACFIT7_'''.DAT (file unit 4)

These files are concerned with ACF data output. For each job a separate file is opened with the extension number given by IDAT+JOB-1 (see parameter data input list). There are 5 columns with the line format (4X,I4,1P,4E14.5) :

column

| | | |
|---|---|---|
| 1 | : | N, current line number |
| 2 | : | XDATA, lag time TAU (sec), TAU=0,TAUMAX |
| 3 | : | YDATA, given right-hand sided ACF data from file 'FINPUT' |
| 4 | : | YFIT, fit function data |
| 5 | : | DIF, YDATA-YFIT |

These files are specially suitable for graphical plots. All originally given right-hand sided ACF data are read out for each job. If the lag time TAU is above the fit range, the values of YFIT and DIF are set equal zero.

- ACFIT6_'''.PAR, ACFIT7_'''.PAR (file unit 5)

These files contain the fit parameter data and additional data of all jobs. They have an extension number of 3 characters for run identification. The extension number is given by FFORM (see parameter data input list). The files are required in uncertainty estimations by the GLSA. The following data are output on two lines for each job in the format (I4,2X,1P.9E14.5/6X,E14.5) :

| | | |
|---|---|---|
| JOB | : | job number |
| TAUEND | : | lag time end value $\tau_{end}$ (sec) of the fit |
| C0 | : | $C_0$, fit value of the amplitude of the ideal ACF |
| A0 | : | $A_0$, derived value of the PSD amplitude; note, that this value has different scaling between the two codes (Section 2.2). |
| B0 | : | $B_0$, fit value of the PSD background amplitude at $\omega_L / 2\pi$ (Hz), (=0., if IBGR=0) |
| ALPHA | : | $\alpha$, fit value of the PSD background decay constant (sec/rad), (=0., if IBGR=0, =constant, if IBGR=1) |
| FROCF | : | $\omega_c / 2\pi$, fit value of the ACF oscillation frequency (Hz) |
| FLAMDA | : | $\lambda$, fit value of the ACF decay constant (1/sec) |
| DR | : | decay ratio, value derived from FROCF and FLAMDA |
| PEAKMAX | : | $P_{max}$, estimated peak amplitude in the PSD |
| CHISQR | : | $\chi^2_R$, reduced chi-square of the fit |

For a cancelled job, the values starting with C0 are set equal 1.E30.

- ACFIT6_'''.FIT, ACFIT7_'''.FIT (file unit 6)

These files are opened for each job separately, but only optionally. They have the extension number given by IDAT+JOB-1. The data on these files allow to follow the fit iterations. They refer to the end of an iteration step. Real values are not converted into the E-specification. There are 9 columns with the line format (I4,2X,1PD14.5,2D17.8,5D14.5) :

column

| | | |
|---|---|---|
| 1 | : | IT, current number of iteration, starting with IT=0 |
| 2 | : | XLAMDA, gradient-expansion coefficient |
| 3 | : | CHISQ, reduced chi-square of the preceding iteration |
| 4 | : | CHISQR, reduced chi-square of the current iteration |
| 5-9 | : | THETA, vector of the fit parameter data |
| | | THETA(1) = C0 |

THETA(2) = FR0CF (rad/sec)

THETA(3) = FLAMDA (1/sec)

THETA(4) = B0 (=0., if IBGR=0)

THETA(5) = ALPHA (sec/rad) (=0., if IBGR=0, =constant, if IBGR=1)

- ACFIT6_'''.PLT, ACFIT7_'''.PLT

These files contain the same data as written on the files ACFIT6_'''.PAR or ACFIT7_'''.PAR with the exception, that lines of cancelled jobs are deleted. They have also the same extension numbers. The files serve for graphical plots with TAUEND as abscissa.

## 10.4 Parameter Data Input (ACFIT6.IN, ACFIT7.IN)

- line 1, format (A), RUN

A string of max. 50 characters for run identification.

- line 2, format (A), FINPUT

A string of max. 50 characters for the file name of the estimated ACF data (including preceding lag time values) to be put in. The codes assume a maximum number NPMAX=1025 of right-hand sided ACF data. The data are read in until the EOF mark is reached, from where the available number NPA of data points is obtained. The value of NPA is not allowed to be below an internally prescribed value given by NPLIMIT; otherwise a stop will occur. NPLIMIT is presently set equal 20. This value can be changed in the data declaration on line 62 in the listing of ACFIT6 or on line 61 in the listing of ACFIT7.

- line 3, format (A), IFORM

A string of max. 20 characters of the line read format (in E- or F-specification) of the data on file FINPUT including the format brackets.

First variable : lag time

Second variable : ACF data

If the data are taken from the file ACCF0'''.PLO (ACF estimation code ACCF1 or ACCF2, Sections 7, 8), the read format can automatically be set by writing :

First character : * , ACF data of channel A

First two characters : ** , ACF data of channel B

- line 4, format (A), FFORM

A string of 3 characters. FFORM is added to the file names ACFIT6_'''.PAR or ACFIT7_'''.PAR as an extension number for run identification. If these files are used in the GLSA as input to the code ACFITEV4 (Section 12) or to the code ACFITEV5 (Section 13), the last two characters must be a current number (with leading zero) of the involved runs.

- line 5, format (D12.5), FRL

Lower cutoff frequency of the signal filter (Hz).

- line 6, format (D12.5), FRH

Upper (higher) cutoff frequency of the signal filter (Hz).

Internal restrictions : IF(FRL.LT.0.) FRL=0.

IF(FRH.LT.1.1*FRL) STOP

There are no further restrictions in the codes. It is assumed that the band-pass filter is sufficiently large and contains well the resonance frequency value of the peak in the PSD. A value for FRH must always be given. In the case of no signal filtering, on has to write FRL=0., FRH=Nyquist cutoff frequency value. If the ACF to be put in, is obtained from signal data which have been filtered by the code FFTF2 (Section 5), one has exactly to transfer the calculated possible values for FRL and FRH.

- line 7, format (I1), IBGR

Option parameter for the kind of fit.

IBGR= 0 : 3-parametric fit neglecting the PSD background term.

1 : 4-parametric fit including the PSD background amplitude B0, but with a given constant value for ALPHA (determined by the slope index SLOPE).

2 : 5-parametric fit including the PSD background amplitude B0 and the background decay constant ALPHA.

Internal restrictions : IF(IBGR.LT.0) IBGR=0

IF(IBGR.GT.2) IBGR=2

- line 8, format (D12.5), B0FIN

The value to be given for B0FIN is a factor for initializing the background amplitude B0. It must be given in %. The initialized value of B0 follows from 0.01*B0FIN*YDATA(1), where YDATA(1) is the estimated ACF data point at zero lag time.

Internal restrictions : IF(B0FIN.LT.0.) B0FIN=0.

IF(B0FIN.GT.1.E3) B0FIN=1.E3

IF(IBGR.EQ.0) B0FIN=0.

- line 9, format (D12.5), SLOPE

PSD background slope index. SLOPE initializes ALPHA.

Internal restrictions : IF(IBGR.EQ.0) SLOPE=0.

IF(IBGR.EQ.2.AND.SLOPE.EQ.0.) STOP

IF(ABS(SLOPE).GT.6) STOP

Note that a zero slope index value for IBGR>0 indicates a constant background under the peak in the PSD. It gives physically no sense. Tentatively, no restriction has been imposed for IBGR=1. Furthermore, the input of a positive value for SLOPE is left open. It may happen, that the selected interesting peak in the PSD is followed by a second right-hand sided peak, whose left-hand sided tail grows in to the first peak, pretending a background with a positive slope index value.

- line 10, format (I1), IWEIGHT

Parameter for selecting the weight function in the fit.

IWEIGHT= 1 : uniform weighting (equation (10.5))

2 : linearly decreasing weighting (equation (10.6))

3 : cosine-shaped weighting (equation (10.7))

The initial part of the weight function selected by IWEIGHT=1-3 can be modified for lower values according to equation (10.8) :

IWEIGHT= 4 : modified case of IWEIGHT=1

5 : modified case of IWEIGHT=2

6 : modified case of IWEIGHT=3

Internal restrictions : IF(IWEIGHT.LT.1) IWEIGHT=1

IF(IWEIGHT.GT.6) IWEIGHT=6

- line 11, format (D12.5), WA

$w_a$ in the modified weight function (equation (10.8)).

Internal restrictions : IF(IWEIGHT.LT.4.OR.WA.LT.0.) WA=0.

IF(WA.GT.1.) WA=1.

- line 12, format (I2), NWE

Last point in the modified weight functions corresponding to $\tau_a$ in equation (10.8).

Internal restrictions : IF(IWEIGHT.LT.4.OR.NWE.LE.0) NWE=0

IF(NWE.EQ.1) NWE=2

IF(NWE.GT.NPLIMIT) NWE=NPLIMIT

- line 13, format (I1), IWEND

Parameter for selecting the end point in the weight function, if the value of IWEIGHT is different from 1 or 4.

IWEND= 1 : the weight is zero at the fit end point TAUEND.

2 : the weight is zero at the end point TAUMAX.

Internal restrictions : IF(IWEND.LT.1) IWEND=1

IF(IWEND.GT.2) IWEND=2

IF(IWEIGHT.EQ.1.OR.IWEIGHT.EQ.4) IWEND=1

- line 14, format (I3), IDAT

Initial extension number for the file names ACFIT6_'''.DAT, ACFIT7_'''.DAT, ACFIT6_'''.FIT and ACFIT7_'''.FIT for job identification. The extension number is given by IDAT+JOB-1.

Internal restrictions : IF(IDAT.LT.1) IDAT=1

IF(IDAT.GT.1000-NRMAX) IDAT=1000-NRMAX

NRMAX is the maximum number of allowed jobs within a run. It has been limited to a value of 20. This value can be changed in the data declaration.

- line 15, format (I2), JOBMAX

JOBMAX specifies the number of jobs in a run with different fit range values.

Internal restrictions : IF(JOBMAX.LT.1) JOBMAX=1

IF(JOBMAX.GT.NRMAX) JOBMAX=NRMAX

- line 16, format (D12.5), XLIN

Initialization value for the gradient-expansion coefficient in the subroutine MARFIT.

Internal restrictions : IF(XLIN.LT.1.E-8) XLIN=1.E-8

IF(XLIN.GT.100.) XLIN=100.

Recommended value : XLIN=1.E-2 or 1.E-3

- line 17, format (D12.5), EPSIL

Fit convergence parameter. Its value must be given in %. The presently used global fit convergence criterion is based on the use of the reduced chi-square CHISQR. If IT (>1) is the current fit iteration number, convergence is reached if

CHISQR(IT-1)/CHISQR(IT).LE.(1.+0.01*EPSIL)

Under the constraint that the gradient-expansion coefficient XLAMDA (subroutine FITMAR) is less or equal 1.E-4.

Internal restrictions : IF(EPSIL.LT.1.E-4) EPSIL=1.E-4

IF(EPSIL.GT.1.) EPSIL=1.

Recommended value : EPSIL=1.E-3 (%)

If in a slowly converging fit, XLAMDA runs to a value below $10^{-10}$, it is reset to $10^{-10}$ for the next iteration. If XLAMDA exceeds the value $10^{6}$, the fit is interrupted.

- line 18, format (I3), ITMAX

Maximum number of allowed iterations in a fit. Note, that the first call to MARFIT does not count as an iteration. The total allowed number of calls is given by ITMAX+1.

Internal restrictions : IF(ITMAX.LT.50) ITMAX=50

IF(ITMAX.GT.999) ITMAX=999

Recommended value : ITMAX=200

- line 19, format (I1), IOUT

Option parameter for opening the files ACFIT6_'''.FIT or ACFIT7_'''.FIT.

IOUT= 0 : no read-out

1 : read-out

Internal restrictions : IF(IOUT.LT.0) IOUT=0

IF(IOUT.GT.1) IOUT=1

- line 20 and following lines, format (F6.2), RANGE, one value on each line.

RANGE is a column vector of length NRMAX and defines the fit range for each job according to the given value for JOBMAX. The values must be given in %. The last ACF data point to be included in a fit is given by XDATA(INTEGER(0.01*RANGE*NPA)). The given range values are ordered ascendently by the codes (subroutine SORT).

Internal restrictions : IF(RANGE.LT.10..OR.RANGE.GT.99.99) STOP

If a range value leads to a number of ACF data which are below the value of NPLIMIT, the next job is taken. If less range values are given than requested by the value attributed to JOBMAX, the value of JOBMAX is corrected and set to the available number of range values.

Recommended values for medium DR cases :

RANGE=15-50 % in steps of 2.5 % (JOBMAX=15)

One can roughly divide the DR cases in low DR cases with a DR<0.5, in medium DR cases with 0.5≤DR≤0.8, and in high DR cases with a DR>0.8. The cases can mostly recognized from the peak width in the plot of a PSD estimate. For low DR cases, the upper limit of the range values should be reduced to about 35-40 %, because higher values lead mostly to useless fits. Eventually, the given values should start at the lower limit of 10 %. The situation has been found to be viceversa for high DR cases, where the use of values from 30 % up to 70 % can be recommended.

# LISTING

## ACFIT6

```
      PROGRAM ACFIT6
C
C     THE CODE FITS FREQUENCY-FILTERED AUTO-CORRELATION FUNCTION (ACF)
C     DATA TO THE DAMPED OSCILLATOR MODEL WHICH IS USED IN BWR
C     BOILING INSTABILITY ANALYSIS. THERE ARE THE OPTIONS FOR A
C     ZERO-PARAMETRIC, A ONE-PARAMETRIC OR A TWO-PARAMETRIC BACKGROUND F
C     TAU-RANGE CAN BE SELECTED AND SUCCESSIVELY INCREASED WITHIN 10 AND
C     OF THE MAXIMUM TAU-RANGE. THE CODE ASSUMES THE FORMAT OF THE
C     ACF DATA OUTPUT FROM THE CODE ACCF1.
C
C     FOR THE LEAST-SQUARES FIT THE ROUTINE MARFIT IS USED, WHICH REQUIR
C     THE CALCULATION OF THE DERIVATIVES OF THE FIT FUNCTION.
C     THERE ARE OPTIONS FOR SELECTING WEIGHTING FUNCTIONS.
C     THE CORRECTED ACF IS OBTAINED FROM THE IDEAL ACF.
C     THE NUMERICAL INTEGRATION PROCEDURE (BY THE IMSL ROUTINES DQDAWO
C     AND DQDAWF) IS DIVIDED INTO INTEGRATION SUBINTERVALS.
C     THE PARAMETERS TO BE DETERMINED BY THE FIT ARE :
C         THETA(1) = C0 (AMPLITUDE OF THE ACF)
C         THETA(2) = OMEGAC
C         THETA(3) = FLAMDA
C         THETA(4) = B0     (OPTION)
C         THETA(5) = ALPHA  (OPTION)
C
C     CODE WRITTEN BY K.BEHRINGER, JUNE 1999, FOR BATCH OPERATION
C     ON THE PSI DEC-ALPHA 2100 COMPUTER.
C     REVISED VERSION OF ACFIT5, SEPTEMBER 1999.
C
C     COMPILATION COMMAND : FORTRAN/G_FLOATING ACFIT6
C
C     LINK COMMAND        : IMPORT IMSL
C                           LINK ACFIT6,IMSLIBG_STATIC/OPT,IMSLPSECT/OPT
C
C     PARAMETERS :
C
C          NPMAX   = MAXIMUM NUMBER OF THE RIGHT-HAND SIDE ACF DATA.
C                    A CHANGEMENT REQUIRES ADAPTATION IN MANY ROUTINES.
C          NPARMAX = MAXIMUM NUMBER OF FIT PARAMETERS. (=5)
C          NRMAX   = MAXIMUM NUMBER OF TAU-RANGES.
C          IWMAX   = MAXIMUM NUMBER OF AVAILABLE WEIGHTING FUNCTIONS.
C
C     NOTE : IF THE NAMES OF THE DATA OUTPUT FILES (FILE4/5/6/7) ARE CHA
C            A CORRESPONDING CHANGEMENT OF THE LATER ENCODE STATEMENTS
C            IS REQUIRED.
C
C
C     DECLARATIONS
C
      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER (NPMAX=1025,NPARMAX=5,NRMAX=20,IWMAX=6)
      CHARACTER*50 RUN,FILE1,FILE2,FILE3,FINPUT,FILE4,FILE5,FILE6,
     1FILE7
      CHARACTER*20 IFORM,IFORM1,IFORM2
      CHARACTER*3 FFORM
```

```
      REAL*4 SC
      DIMENSION XDATA(NPMAX),YDATA(NPMAX),THETA(NPARMAX),
     1RANGE(NRMAX)
      COMMON/XYDATA/XDATA,YDATA
      COMMON/CONTROL/NPEND,IBGR
      COMMON/PAR/OMEGAL,OMEGAH,PI
      COMMON/W/WA,NWE,NPA1,IWEIGHT
      EQUIVALENCE (FILE3,FINPUT)
      DATA NF,NPLIMIT,PI,SC/1,20,3.141592654D0,1.E30/,
     1IFORM1/'(5X,2E12.4)'/,
     2IFORM2/'(5X,E12.4,12X,E12.4)'/,
     3FILE1/'ACFIT6.IN                                   '/,
     4FILE2/'ACFIT6.PRT                                  '/,
     5FILE3/'<FINPUT>                                    '/,
     6FILE4/'ACFIT6_^^^.DAT                              '/,
     7FILE5/'ACFIT6_^^^.PAR                              '/,
     8FILE6/'ACFIT6_^^^.FIT                              '/,
     9FILE7/'ACFIT6_^^^.PLT                              '/
C
C     FORMATS
C
 1000 FORMAT(4(A/),2(D12.5/),I1/2(D12.5/),I1/D12.5/I2/I1/I3/I2)
 1001 FORMAT(2(D12.5/),I3/I1)
 1005 FORMAT(F6.2)
 2000 FORMAT(1H1,40X,'P R O G R A M     A C F I T 6'///1X,
     1'FILES : PARAMETER DATA : FILE1 = ',A/9X,
     2'PRINT OUTPUT    : FILE2 = ',A/9X,
     3'ACF DATA',7X,': FILE3 = ',A/9X,
     4'DATA OUTPUT',4X,': FILE4 = ',A/9X,
     5'FIT PARAMETER',2X,': FILE5 = ',A/9X,
     6'FIT DATA',7X,': FILE6 = ',A/9X,
     7'PARAMETER PLOT',1X,': FILE7 = ',A///)
 2005 FORMAT(1X,'RUN DENOTATION',36X,'RUN',5X,'=',1X,A//1X,
     1'ACF DATA FILE',37X,'FINPUT',2X,'=',1X,A//1X,
     2'ACF DATA FORMAT',35X,'IFORM',3X,'=',1X,A//1X,
     3'ADDITIONAL DENOTATION OF OUTPUT FILES',13X,'FFORM',3X,
     4'=',1X,A//1X,
     5'LOWER CUTOFF FREQUENCY (HZ)',23X,'FRL',5X,'=',1PD14.5/1X,
     6'UPPER CUTOFF FREQUENCY (HZ)',23X,'FRH',5X,'=',D14.5//1X,
     7'BACKGROUND FIT PARAMETER (0:B0=0,1:B0,2:B0,ALPHA)',1X,
     8'IBGR',4X,'=',I4/1X,
     9'INITIAL BACKGROUND AMPLITUDE (0/0 OF YDATA(1))',4X,
     A'B0FIN',3X,'=',D14.5/1X,
     B'INITIAL BACKGROUND SLOPE INDEX',20X,'SLOPE',3X,'=',D14.5//1X,
     C'PARAMETER FOR WEIGHTING (1:U, 2:L.DCR., 3:COS)',4X,
     D'IWEIGHT =',I4/6X,'(MODIFIED INITIAL PART : 4:1,5:2,6:3)'/1X,
     E'INITIAL WEIGHT AMPLITUDE FACTOR',19X,'WA',6X,'=',D14.5/1X,
     F'END POINT OF WEIGHT FUNCTION INCREASE'13X,'NWE',5X,'=',I4//1X,
     G'WEIGHT ENDPOINT (1:NPEND, 2:NPA)',18X,'IWEND',3X,'=',I4//1X,
     H'INITIAL NUMBER OF DATA OUTPUT FILE',16X,'IDAT',4X,'=',I4//1X,
     I'REQUESTED NUMBER OF JOBS',26X,'JOBMAX',2X,'=',I4/)
 2010 FORMAT('1ACCEPTED  NUMBER OF JOBS',26X,'JOBMAX',2X,'=',I5//1X,
     1'RANGE VALUES (0/0)',44X,'N',3X,'RANGE'/)
 2015 FORMAT(61X,I3,2X,F6.2)
 2016 FORMAT(//1X,'DATA FOR CONVERGENCE (SUBROUTINE FITMAR) :'//6X,
     1'INITIAL VALUE FOR XLAMDA (1.D-8 - 1.D2)',6X,'XLIN',4X,'=',
     21PD15.5//6X,
```

```
     3'CONVERGENCE PARAMETER (0/0, 1.D-4 - 1.D0)',4X,'EPSIL',3X,'=',
     4D15.5//6X,
     5'ALLOWED NUMBER OF ITERATIONS (50 - 999)',6X,'ITMAX',3X,'=',
     6I5//6X,
     7'DATA OUTPUT DURING FIT (0:NO, 1:YES)',9X,'IOUT',4X,'=',I5/)
 2020 FORMAT(////11X,'OPENING ERROR ! FILE = ',I1)
 2025 FORMAT(////11X,'READ-ERROR IN PARAMETER DATA !')
 2030 FORMAT(////11X,'ERRANEOUS PARAMETER DATA !')
 2035 FORMAT(1X,'AVAILABLE NUMBER OF ACF DATA',22X,'NPA',5X,'=',I5//1X,
     1'MAXIMUM TAU-RANGE (SEC)',27X,'TAUMAX',2X,'=',1PD15.5)
 2040 FORMAT('1JOB NUMBER',40X,'JOB',5X,'=',I4//1X,
     1'NUMBER OF DATA OUTPUT FILE',24X,'IDAT',4X,'=',I4//1X,
     2'RANGE VALUE (0/0)',33X,'RANGE',3X,'=',F7.2//1X,
     3'LAST DATA POINT NUMBER',28X,'NPEND',3X,'=',I5//1X,
     4'TAU-RANGE (SEC)',35X,'TAUEND',2X,'=',1PD14.5/)
 2045 FORMAT(1X,'ESTIMATED FIT PARAMETERS (RELATIVE VALUES)',8X,1P,
     1'THETA',3X,'=',5D14.5/1X,
     2'REDUCED CHI-SQUARE OF THE FIT',21X,'CHISQR',2X,'=',D14.5/1X,
     3'NUMBER OF ITERATIONS',30X,'IT',6X,'=',I4/)
 2050 FORMAT(/1X,'ESTIMATED FIT PARAMETERS (ABSOLUTE VALUES)',//11X,
     1'PSD FUNCTION AMPLITUDE',18X,'A0',6X,'=',1PD14.5/11X,
     2'BACKGROUND AMPLITUDE',20X,'B0',6X,'=',D14.5/11X,
     3'BACKGROUND DECAY CONSTANT (SEC/RAD)',5X,'ALPHA',3X,'=',D14.5//
     411X,'UNDAMPED RESONANCE FREQUENCY (HZ)',7X,'FR0',5X,'=',D14.5/11X,
     5'DAMPED RESONANCE FREQUENCY (HZ)',9X,'FR0R',4X,'=',D14.5/11X,
     6'ACF FREQUENCY (HZ)',22X,'FR0CF',3X,'=',D14.5//11X,
     7'DECAY CONSTANT (1/SEC)',18X,'FLAMDA',2X,'=',D14.5/11X,
     8'DECAY RATIO',29X,'DR',6X,'=',D14.5//11X,
     9'PEAK AMPLITUDE IN THE PSD',15X,'PEAKMAX =',D14.5/11X,
     A'PEAK BACKGROUND AMPLITUDE',15X,'B0PEAK',2X,'=',D14.5)
 2055 FORMAT(////11X,'NPEND TOO SMALL !')
 2060 FORMAT(////11X,'E N D')
 4000 FORMAT(4X,I4,1P,4E14.5)
 5000 FORMAT(I4,2X,1P,9E14.5/6X,E14.5)
C
C     INPUT PARAMETER DATA
C
      OPEN(UNIT=2,FILE=FILE2,STATUS='NEW',DEFAULTFILE='DIRINPUT')
      WRITE(2,2000) FILE1,FILE2,FILE3,FILE4,FILE5,FILE6,FILE7
      OPEN(UNIT=1,FILE=FILE1,STATUS='OLD',DEFAULTFILE='DIRINPUT',
     1ERR=100)
      READ(1,1000,ERR=105,END=105) RUN,FINPUT,IFORM,FFORM,FRL,FRH,
     1IBGR,B0FIN,SLOPE,IWEIGHT,WA,NWE,IWEND,IDAT,JOBMAX
      IF(IFORM.EQ.'*') IFORM=IFORM1
      IF(IFORM.EQ.'**') IFORM=IFORM2
      IF(FRL.LT.0.D0) FRL=0.D0
      IF(IBGR.LT.0) IBGR=0
      IF(IBGR.GT.2) IBGR=2
      IF(B0FIN.LT.0.D0) B0FIN=0.D0
      IF(B0FIN.GT.1.D3) B0FIN=1.D3
      IF(IBGR.EQ.0) B0FIN=0.D0
      IF(IBGR.EQ.0) SLOPE=0.D0
      IF(IWEIGHT.LT.1) IWEIGHT=1
      IF(IWEIGHT.GT.IWMAX) IWEIGHT=IWMAX
      IF(IWEIGHT.LT.4.OR.NWE.LE.0.OR.WA.LT.0.D0) THEN
      NWE = 0
      WA = 0.D0
      ELSE
```

```
      IF(NWE.EQ.1) NWE=2
      IF(NWE.GT.NPLIMIT) NWE=NPLIMIT
      IF(WA.GT.1.D0) WA=1.D0
      END IF
      IF(IWEND.LT.1.OR.IWEIGHT.EQ.1) IWEND=1
      IF(IWEND.GT.2) IWEND=2
      IF(IDAT.LT.1) IDAT=1
      IF(IDAT.GT.1000-NRMAX) IDAT=1000-NRMAX
      IF(JOBMAX.LT.1) JOBMAX=1
      IF(JOBMAX.GT.NRMAX) JOBMAX=NRMAX
      WRITE(2,2005) RUN,FINPUT,IFORM,FFORM,FRL,FRH,IBGR,
     1B0FIN,SLOPE,IWEIGHT,WA,NWE,IWEND,IDAT,JOBMAX
      IF(FRH.LT.1.1D0*FRL) GO TO 120
      READ(1,1001,ERR=105,END=105) XLIN,EPSIL,ITMAX,IOUT
      IF(XLIN.LT.1.D-8) XLIN=1.D-8
      IF(XLIN.GT.1.D2) XLIN=1.D2
      IF(EPSIL.LT.1.D-4) EPSIL=1.D-4
      IF(EPSIL.GT.1.D0) EPSIL=1.D0
      IF(ITMAX.LT.50) ITMAX=50
      IF(ITMAX.GT.999) ITMAX=999
      IF(IOUT.LT.0) IOUT=0
      IF(IOUT.GT.1) IOUT=1
      DO 1 N=1,JOBMAX
    1 READ(1,1005,ERR=105,END=110) RANGE(N)
      GO TO 115
  110 JOBMAX = N-1
  115 WRITE(2,2010) JOBMAX
      IF(JOBMAX.EQ.0) GO TO 120
      CALL SORT(RANGE,JOBMAX)
      DO 2 N=1,JOBMAX
    2 WRITE(2,2015) N,RANGE(N)
      DO 3 N=1,JOBMAX
      IF(RANGE(N).LT.1.D1.OR.RANGE(N).GT.9.999D1) GO TO 120
    3 RANGE(N) = 1.D-2*RANGE(N)
      WRITE(2,2016) XLIN,EPSIL,ITMAX,IOUT
      EPSIL = 1.D0+1.D-2*EPSIL
      B0FIN = 1.D-2*B0FIN
      IF(IBGR.EQ.2.AND.SLOPE.EQ.0.D0.OR.
     1DABS(SLOPE).GT.6.D0) GO TO 120
      CLOSE(UNIT=1,STATUS='KEEP')
      NF = 5
      ENCODE(3,'(A)',FILE5(8:10)) FFORM
      OPEN(UNIT=5,FILE=FILE5,STATUS='NEW',DEFAULTFILE='DIRINPUT',
     1ERR=100)
      NF = 3
      OPEN(UNIT=3,FILE=FINPUT,STATUS='OLD',DEFAULTFILE='DIRINPUT',
     1ERR=100)
      NF = 7
      ENCODE(3,'(A)',FILE7(8:10)) FFORM
      OPEN(UNIT=7,FILE=FILE7,STATUS='NEW',DEFAULTFILE='DIRINPUT',
     1ERR=100)
C
C     INPUT ACF DATA
C
      CALL INPUT(NPMAX,IFORM,XDATA,YDATA,NPA)
      IF(NPA.LT.NPLIMIT) STOP 105
      TAUMAX = XDATA(NPA)
```

```
      WRITE(2,2035) NPA,TAUMAX
      CLOSE(UNIT=3,STATUS='KEEP')
C
C     COMPUTATION AND DATA READ-OUT
C
      NF = 4
      DPI = 2.D0*PI
      OMEGAL = DPI*FRL
      OMEGAH = DPI*FRH
      NPAR = NPARMAX
      NPA1 = NPA-1
      IF(IBGR.EQ.1) NPAR=NPAR-1
      IF(IBGR.EQ.0) NPAR=NPAR-2
      CALL INIT(YDATA(1),B0FIN,SLOPE,THETA)
      C0S = THETA(1)
      DO 10 JOB=1,JOBMAX
      NPEND = RANGE(JOB)*NPA
      IF(IWEND.EQ.1) NPA1=NPEND-1
      TAUEND = XDATA(NPEND)
      WRITE(2,2040) JOB,IDAT,1.D2*RANGE(JOB),NPEND,TAUEND
      IF(NPEND.LT.NPLIMIT) GO TO 125
      IF(IOUT.EQ.0) GO TO 129
      NF = 6
      ENCODE(3,'(I3)',FILE6(8:10)) IDAT
      OPEN(UNIT=6,FILE=FILE6,STATUS='NEW',DEFAULTFILE='DIRINPUT',
     1ERR=100)
      NF = 4
  129 IREP1 = 0
      IF(THETA(1).EQ.C0S) IREP1=1
  130 IREP2 = 0
      CALL FITMAR(NPAR,THETA,XLIN,EPSIL,ITMAX,IOUT,CHISQR,IT,IREP2,
     1*132)
      WRITE(2,2045) THETA,CHISQR,IT
      C0 = THETA(1)
      FR0CF = THETA(2)
      FLAMDA = THETA(3)
C     REPETITION OF THE FIT
      IF(C0.GT.0.D0.AND.FR0CF.GT.0.D0.AND.
     1FLAMDA.GT.-1.D-1.AND.IREP2.EQ.0) GO TO 131
      IF(IREP1.GE.1) GO TO 132
      CALL INIT(YDATA(1),B0FIN,SLOPE,THETA)
      IREP1 = 1
      GO TO 130
C*
  131 B0 = THETA(4)
      ALPHA = THETA(5)
      FLAMDASQ = FLAMDA**2
      OMEGA0SQ = FR0CF**2+FLAMDASQ
      FR0 = DSQRT(OMEGA0SQ)
      DR = DEXP(-DPI*FLAMDA/FR0CF)
      A0 = 4.D0*C0*FLAMDA*OMEGA0SQ
      FR0R = 0.D0
      IF(FR0CF.GT.FLAMDA) FR0R=DSQRT(OMEGA0SQ-2.D0*FLAMDASQ)
      PEAKMAX = 0.D0
      IF(DR.LT.1.D0.AND.FR0R.GT.0.D0) PEAKMAX=A0*FINT(FR0R)
      B0PEAK = 0.D0
      IF(PEAKMAX.GT.0.D0) B0PEAK=B0*DEXP(-ALPHA*(FR0R-OMEGAL))
      FR0 = FR0/DPI
```

```
      FROCF = FROCF/DPI
      FROR = FROR/DPI
      WRITE(2,2050) A0,B0,ALPHA,FR0,FROR,FROCF,FLAMDA,DR,PEAKMAX,
     1B0PEAK
      WRITE(5,5000) JOB,SNGL(TAUEND),SNGL(C0),SNGL(A0),SNGL(B0),
     1SNGL(ALPHA),SNGL(FROCF),SNGL(FLAMDA),SNGL(DR),SNGL(PEAKMAX),
     2SNGL(CHISQR)
      WRITE(7,5000) JOB,SNGL(TAUEND),SNGL(C0),SNGL(A0),SNGL(B0),
     1SNGL(ALPHA),SNGL(FROCF),SNGL(FLAMDA),SNGL(DR),SNGL(PEAKMAX),
     2SNGL(CHISQR)
      ENCODE(3,'(I3)',FILE4(8:10)) IDAT
      OPEN(UNIT=4,FILE=FILE4,STATUS='NEW',DEFAULTFILE='DIRINPUT',
     1ERR=100)
      DO 11 N=1,NPA
      IF(N.GT.NPEND) GO TO 135
      YFIT = FIT(THETA,N)
      DIF = YDATA(N)-YFIT
      GO TO 11
  135 YFIT = 0.D0
      DIF = 0.D0
   11 WRITE(4,4000) N,SNGL(XDATA(N)),SNGL(YDATA(N)),SNGL(YFIT),
     1SNGL(DIF)
      CLOSE(UNIT=4,STATUS='KEEP')
      IF(IOUT.EQ.1) CLOSE(UNIT=6,STATUS='KEEP')
C     RESETS
      IF(FLAMDA.LT.0.D0) THETA(3)=0.D0
      IF(DR.LT.2.D-1) CALL INIT(YDATA(1),B0FIN,SLOPE,THETA)
      IF(B0.LT.0.D0.OR.ALPHA.GT.1.D1)
     1CALL INIT1(YDATA(1),B0FIN,SLOPE,THETA)
C*
      GO TO 10
  132 WRITE(5,5000) JOB,SNGL(TAUEND),SC,SC,SC,SC,SC,SC,SC,SC,SC
      CALL INIT(YDATA(1),B0FIN,SLOPE,THETA)
      IF(IOUT.EQ.1) CLOSE(UNIT=6,STATUS='DELETE')
      GO TO 10
  125 WRITE(2,2055)
   10 IDAT = IDAT+1
      WRITE(2,2060)
      STOP
  100 WRITE(2,2020) NF
      STOP
  105 WRITE(2,2025)
      STOP
  120 WRITE(2,2030)
      STOP
      END
C=========================================================
      SUBROUTINE INPUT(NPMAX,IFORM,XDATA,YDATA,NPA)
C
C     DATA INPUT. IT SPECIFICALLY REFERS TO THE OUTPUT DATA
C     GIVEN BY THE CODE ACCF.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*4 XS,YS
      CHARACTER*20 IFORM
      DIMENSION XDATA(NPMAX),YDATA(NPMAX)
 2000 FORMAT(1X,'EOF AT ACF DATA POINT NUMBER',22X,'NEOF',4X,'=',I5/)
```

```
 2005 FORMAT(////11X,'READ=ERROR IN ACF DATA ! NERROR =',I5)
      DO 1 N=1,NPMAX
      READ(3,IFORM,ERR=100,END=105) XS,YS
      IF(XS.GE.0.) GO TO 120
    1 CONTINUE
  110 NPA = 0
      RETURN
  100 NERROR = N
  115 WRITE(2,2005) NERROR
      STOP 100
  105 NEOF = N
      WRITE(2,2000) NEOF
      GO TO 110
  120 XDATA(1) = DBLE(XS)
      YDATA(1) = DBLE(YS)
      NS = N-1
      DO 2 N=2,NPMAX
      READ(3,IFORM,ERR=125,END=130) XS,YS
      XDATA(N) = DBLE(XS)
    2 YDATA(N) = DBLE(YS)
      NPA = NPMAX
      RETURN
  125 NERROR = N+NS
      GO TO 115
  130 NEOF = N+NS
      NPA = N-1
      WRITE(2,2000) NEOF
      RETURN
      END
C=========================================================
      SUBROUTINE INIT(YDATA1,B0FIN,SLOPE,THETA)
C
C     INITIALIZATION OF THETA.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION THETA(1)
      COMMON/PAR/OMEGAL,OMEGAH,PI
      COMMON/CONTROL/NPEND,IBGR
C     C0 :
      THETA(1) = YDATA1
C     OMEGAC :
      THETA(2) = 5.D-1*(OMEGAL+OMEGAH)
      IF(OMEGAL.LE.2.D-1*PI.OR.OMEGAH.GT.3.D0*PI) THETA(2)=PI
C     FLAMDA : (FOR A DECAY RATIO OF ABOUT 0.5)
      THETA(3) = 1.103D-1*THETA(2)
      ENTRY INIT1(YDATA1,B0FIN,SLOPE,THETA)
C     B0 AND ALPHA :
      IF(IBGR.EQ.0) THEN
      THETA(4) = 0.D0
      THETA(5) = 0.D0
      ELSE
      THETA(4) = B0FIN*YDATA1
      THETA(5) = -5.D-1*SLOPE*DLOG(1.D1)/PI
      END IF
      RETURN
      END
C=========================================================
      SUBROUTINE FITMAR(NPAR,THETA,XLIN,EPSIL,ITMAX,IOUT,CHISQR1,IT,
```

```
     1IREP,*)
C
C      FITMAR CONTROLS THE LEAST-SQUARES FIT ROUTINE MARFIT.
C
       IMPLICIT REAL*8 (A-H,O-Z)
       PARAMETER (NPMAX=1025,NPARMAX=5)
       DIMENSION WEIGHTS(NPMAX),THETA(NPARMAX)
       SAVE XLMIN,XLMAX
       DATA XLMIN,XLMAX/1.D-10,1.D6/
       COMMON/MARF/WEIGHTS,XLAMDA,CHISQ,CHISQR,MODE
       COMMON/CONTROL/NPEND,IBGR
       EXTERNAL FIT,FITDER
 2000 FORMAT(1X,'NUMBER OF ITERATIONS EXCEEDS ITMAX')
 2005 FORMAT(1X,'ZERO-DETERMINANT')
 2010 FORMAT(1X,'XLAMDA EXCEEDS XLMAX')
 6000 FORMAT(I4,2X,1PD14.5,2D17.8,5D14.5)
       IRET = 0
       DO 1 N=1,NPEND
    1 WEIGHTS(N) = WEIGHT(N)
       MODE = 1
       XLAMDA = XLIN
       IF(XLAMDA.LT.XLMIN) XLAMDA=XLMIN
       IF(XLAMDA.GT.XLMAX) XLAMDA=XLMAX
       DO 2 I=1,ITMAX+1
       IT = I-1
       ITER = IT
       CALL MARFIT(FIT,FITDER,NPAR,THETA,IRET)
       IF(IRET.LT.0) RETURN 1
       CHISQR1 = CHISQR
       IF(IOUT.EQ.1) WRITE(6,6000) IT,XLAMDA,CHISQ,CHISQR,THETA
       IF(CHISQR.LT.0.D0) GO TO 200
       IF(XLAMDA.LT.XLMIN) XLAMDA=XLMIN
       IF(XLAMDA.GT.XLMAX) GO TO 205
       IF(I.GT.1) GO TO 100
       MODE = 2
       GO TO 2
  100 IF(CHISQ/CHISQR.LE.EPSIL.AND.XLAMDA.LE.1.D-4) RETURN
       CHISQR = 1.00000001D0*CHISQR
    2 CONTINUE
       WRITE(2,2000)
       GO TO 210
  200 WRITE(2,2005)
       GO TO 210
  205 WRITE(2,2010)
  210 IREP = IREP+1
       RETURN
       END
C=========================================================
       SUBROUTINE MARFIT(FN,FNDERIV,NTERMS,C,IRET)
C
C      MARFIT IS A LEAST-SQUARES FIT ROUTINE FOR A ONE-DIMENSIONAL
C      NONLINEAR FUNCTION OF AN INDEPENDENT VARIABLE WITH AN ARBITRARY
C      NUMBER OF PARAMETERS TO BE DETERMINED BY THE FIT. THE ROUTINE
C      IS BASED ON A COMBINATION OF THE GAUSS-NEWTON METHOD WITH THE
C      GRADIENT-EXPANSION METHOD BY D.W.MARQUARDT (AN ALGORITHM FOR
C      LEAST-SQUARES ESTIMATION OF NONLINEAR PARAMETERS, J.SOC.INDUST.
C      APPL.MATH., VOL 11, NO.2 (1963) 431).
```

```
C
C     CODE DEVELOPED AND DOCUMENTED BY K.BEHRINGER, INTERNAL EIR REPORT
C     TM-PH-373 (1970).
C     CODE ADAPTED TO THE PRESENT PROBLEM BY K.BEHRINGER, JUNE 1999.
C
C     FN      : FIT FUNCTION AT A SINGLE ARGUMENT POINT,
C     FNDERIV : SUBROUTINE FOR CALCULATING THE DERIVATIVES OF THE
C               FIT FUNCTION AT A SINGLE ARGUMENT POINT,
C     NTERMS  : NUMBER OF PARAMETERS,
C     C       : PARAMETER VECTOR TO BE DETERMINED BY THE FIT.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER (NPMAX=1025,NPARMAX=5)
      DIMENSION YFIT(NPMAX),XDATA(NPMAX),YDATA(NPMAX),WEIGHTS(NPMAX),
     1ALPHA(NPARMAX,NPARMAX),ERROR(NPARMAX,NPARMAX),SIGMAC(NPARMAX),
     2BETA(NPARMAX),DERIV(NPARMAX),C(NPARMAX)
      COMMON/XYDATA/XDATA,YDATA
      COMMON/CONTROL/NPTS,IBGR
      COMMON/MARF/WEIGHTS,XLAMDA,CHISQ,CHISQR,MODE
      IF(MODE-2) 11,12,13
   11 IF(NPTS.GT.NTERMS) GO TO 14
      CHISQR = -2.D0
      RETURN
   14 DO 1 I=1,NPTS
      YFIT(I) = FN(C,I,IRET)
      IF(IRET.LT.0) RETURN
    1 CONTINUE
   13 CHISQ = 0.D0
      DO 2 I=1,NPTS
    2 CHISQ = CHISQ+WEIGHTS(I)*(YDATA(I)-YFIT(I))**2
      CHISQ = CHISQ/DFLOTJ(NPTS-NTERMS)
      GO TO 15
   12 CHISQ = CHISQR
   15 DO 3 J=1,NTERMS
      BETA(J) = 0.D0
      DO 3 K=1,J
    3 ALPHA(J,K) = 0.D0
      DO 4 I=1,NPTS
      CALL FNDERIV(I,DERIV)
      DO 4 J=1,NTERMS
      BETA(J) = BETA(J)+WEIGHTS(I)*(YDATA(I)-YFIT(I))*DERIV(J)
      DO 4 K=1,J
    4 ALPHA(J,K) = ALPHA(J,K)+WEIGHTS(I)*DERIV(J)*DERIV(K)
      L = 0
   16 DO 5 J=1,NTERMS
      DO 5 K=1,J
      IF(K.EQ.J) GO TO 17
      IF(L.GT.0) GO TO 18
      ALPHA(K,J) = DSQRT(ALPHA(J,J)*ALPHA(K,K))
      IF(ALPHA(K,J)) 22,22,23
   23 ALPHA(J,K) = ALPHA(J,K)/ALPHA(K,J)
   18 ERROR(J,K) = ALPHA(J,K)
      ERROR(K,J) = ERROR(J,K)
      GO TO 5
   17 ERROR(J,J) = 1.D0+XLAMDA
    5 CONTINUE
      CALL MATINV(ERROR,NTERMS,DET)
      IF(DET) 19,22,19
```

```
   22 CHISQR = -1.D0
      RETURN
   19 DO 6 J=1,NTERMS
      SIGMAC(J) = C(J)
      DO 6 K=1,NTERMS
      IF(K.GE.J) GO TO 20
      SIGMAC(J) = SIGMAC(J)+ERROR(J,K)*BETA(K)/ALPHA(K,J)
      GO TO 6
   20 SIGMAC(J) = SIGMAC(J)+ERROR(J,K)*BETA(K)/ALPHA(J,K)
    6 CONTINUE
      CHISQR = 0.D0
      DO 7 I=1,NPTS
      YFIT(I) = FN(SIGMAC,I,IRET)
      IF(IRET.LT.0) RETURN
    7 CHISQR = CHISQR+WEIGHTS(I)*(YDATA(I)-YFIT(I))**2
      CHISQR = CHISQR/DFLOTJ(NPTS-NTERMS)
      IF(CHISQ.GE.CHISQR) GO TO 21
      L = L+1
      XLAMDA = 1.D1*XLAMDA
      IF(L-10) 16,16,21
   21 XLAMDA = 1.D-1*XLAMDA
      DO 8 J=1,NTERMS
      C(J) = SIGMAC(J)
    8 SIGMAC(J) = DSQRT(ERROR(J,J)/ALPHA(J,J))
      RETURN
      END
C==========================================================
      SUBROUTINE MATINV(ARRAY,NORDER,DET)
C
C     MATINV INVERTS A NON-SINGULAR QUADRATIC SYMMETRIC MATRIX
C     ACCORDING TO THE GAUSS-JORDAN ELIMINATION PROCEDURE, AND
C     CALCULATES THE DETERMINANT OF THE GIVEN MATRIX.
C     THE SUBROUTINE IS TAKEN FROM PH.R.BEVINGTON, DATA REDUCTION
C     AND ERROR ANALYSIS FOR THE PHYSICAL SCIENCES, MCGRAW-HILL
C     BOOK COMPANY, 1969.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER (NPARMAX=5)
      DIMENSION ARRAY(NPARMAX,NPARMAX),IK(NPARMAX),JK(NPARMAX)
      DET = 1.D0
      DO 1 K=1,NORDER
      AMAX = 0.D0
      DO 2 I=K,NORDER
      DO 2 J=K,NORDER
      IF(DABS(AMAX).GT.DABS(ARRAY(I,J))) GO TO 2
      AMAX = ARRAY(I,J)
      IK(K) = I
      JK(K) = J
    2 CONTINUE
      IF(AMAX.NE.0.D0) GO TO 11
      DET = 0.D0
      RETURN
   11 I = IK(K)
      IF(I.EQ.K) GO TO 12
      DO 3 J=1,NORDER
      SAVE = ARRAY(K,J)
      ARRAY(K,J) = ARRAY(I,J)
```

```
    3 ARRAY(I,J) = -SAVE
   12 J = JK(K)
      IF(J.EQ.K) GO TO 13
      DO 4 I=1,NORDER
      SAVE = ARRAY(I,K)
      ARRAY(I,K) = ARRAY(I,J)
    4 ARRAY(I,J) = -SAVE
   13 DO 5 I=1,NORDER
      IF(I.NE.K) ARRAY(I,K) = -ARRAY(I,K)/AMAX
    5 CONTINUE
      DO 6 I=1,NORDER
      DO 6 J=1,NORDER
      IF(I.NE.K.AND.J.NE.K) ARRAY(I,J)=ARRAY(I,J)+ARRAY(I,K)*ARRAY(K,J)
    6 CONTINUE
      DO 7 J=1,NORDER
      IF(J.NE.K) ARRAY(K,J)=ARRAY(K,J)/AMAX
    7 CONTINUE
      ARRAY(K,K) = 1.D0/AMAX
    1 DET = DET*AMAX
      DO 8 L=1,NORDER
      K = NORDER-L+1
      J = IK(K)
      IF(J.LE.K) GO TO 14
      DO 9 I=1,NORDER
      SAVE = ARRAY(I,K)
      ARRAY(I,K) = -ARRAY(I,J)
    9 ARRAY(I,J) = SAVE
   14 I = JK(K)
      IF(I.LE.K) GO TO 8
      DO 10 J=1,NORDER
      SAVE = ARRAY(K,J)
      ARRAY(K,J) = -ARRAY(I,J)
   10 ARRAY(I,J) = SAVE
    8 CONTINUE
      RETURN
      END
C=========================================================
      FUNCTION WEIGHT(N)
C
C     A WEIGHT FUNCTION IS CALCULATED FOR THE SELECTED OPTION
C     BY THE PARAMETER IWEIGHT (1-6).
C     NOTE : NPA1=NPA-1
C
C     MODIFICATION OF THE INITIAL PART (IWEIGHT=4-6) :
C     WA  = INITIAL WEIGHT AMPLITUDE FACTOR (0.LE.WA.LT.1.),
C     NWE = END POINT OF WEIGHT FUNCTION INCREASE (1.LT.NPLIMIT)
C           (NPLIMIT IS DEFINED IN THE DATA DECLARATION OF THE
C            MAIN PROGRAM),
C     WF  = WEIGHT FUNCTION FACTOR.
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/W/WA,NWE,NPA1,IWEIGHT
      SAVE PIH
      DATA PIH/1.570796327D0/
      WF = 1.D0
      IF(IWEIGHT.LE.3.OR.N.GE.NWE) GO TO 10
      WF = WA+(1.D0-WA)*DSIN(PIH*DFLOTJ(N-1)/DFLOTJ(NWE))
   10 GO TO (101,102,103,101,102,103) IWEIGHT
C     UNIFORM WEIGHTING :
```

```
  101 WEIGHT = WF
      RETURN
C     LINEARLY DECREASING WEIGHTING :
  102 WEIGHT = WF*DFLOTJ(NPA1+1-N)/DFLOTJ(NPA1)
      RETURN
C     COSINE FUNCTION WEIGHTING :
  103 WEIGHT = WF*DCOS(PIH*DFLOTJ(N-1)/DFLOTJ(NPA1))
      RETURN
      END
C=========================================================
      FUNCTION FIT(THETA,NP,IRET)
C
C     THE FIT FUNCTION IS CALCULATED FOR A VALUE OF TAU.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER (NPMAX=1025)
      DIMENSION THETA(1),XDATA(NPMAX),YDATA(NPMAX),FUNC(NPMAX),
     1RXXCS(NPMAX),BGRS(NPMAX)
      COMMON/XYDATA/XDATA,YDATA
      COMMON/PAR/OMEGAL,OMEGAH,PI
      COMMON/CONTROL/NPEND,IBGR
      COMMON/VAR/C0,B0,ALPHA,OMEGA0SQ,OMEGAC,OMEGAR,FLAMDA,FLAMDASQ
      COMMON/DE/FUNC,RXXCS,BGRS
 2000 FORMAT(////11X,'MESSAGE FROM FIT : IMAGINARY OMEGAR !')
      TAU = XDATA(NP)
      IF(NP.GT.1) GO TO 100
      C0 = THETA(1)
      IF(DABS(C0).LT.1.D-10) C0=DSIGN(1.D-10,C0)
      OMEGAC = THETA(2)
      FLAMDA = THETA(3)
      B0 = THETA(4)
      IF(DABS(B0).LT.1.D-10.AND.IBGR.GT.0) B0=DSIGN(1.D-10,B0)
      ALPHA = THETA(5)
      IF(DABS(ALPHA).LT.1.D-10.AND.IBGR.GT.1)
     1ALPHA=DSIGN(1.D-10,ALPHA)
      FLAMDASQ = FLAMDA**2
      OMEGACSQ = OMEGAC**2
      OMEGA0SQ = OMEGACSQ+FLAMDASQ
      IF(OMEGACSQ.LT.FLAMDASQ) GO TO 105
      OMEGAR = DSQRT(OMEGACSQ-FLAMDASQ)
  100 A = RXXC(TAU,NP)
      RXXCS(NP) = A
      B = BGR(TAU,ALPHA,IRET)
      BGRS(NP) = B
      FIT = C0*A+B0*B/PI
      RETURN
  105 FIT = 0.D0
      WRITE(2,2000)
      IRET = -1
      RETURN
      END
C=========================================================
      SUBROUTINE FITDER(NP,DER)
C
C     FITDER CALCULATES THE DERIVATIVES OF THE FIT FUNCTION.
C
      IMPLICIT REAL*8 (A-H,O-Z)
```

```
      PARAMETER (NPMAX=1025)
      DIMENSION DER(1),XDATA(NPMAX),YDATA(NPMAX),FUNC(NPMAX),
     1RXXCS(NPMAX),BGRS(NPMAX)
      COMMON/XYDATA/XDATA,YDATA
      COMMON/PAR/OMEGAL,OMEGAH,PI
      COMMON/VAR/C0,B0,ALPHA,OMEGA0SQ,OMEGAC,OMEGAR,FLAMDA,FLAMDASQ
      COMMON/FC/CTR
      COMMON/DE/FUNC,RXXCS,BGRS
      SAVE ERRABS,ERREL,IWEIGH,ZERO
      DATA ERRABS,ERREL,IWEIGH,ZERO/1.D-7,1.D-7,1,0.D0/
      EXTERNAL FINT,FINTDER
      TAU = XDATA(NP)
C     DER1 = DFIT/DC0
      DER(1) = RXXCS(NP)
C     DER2 = DFIT/DOMEGAC, DER3 = DFIT/DFLAMDA
      CALL RXXIDER(TAU,D2,D3)
      IF(OMEGAL.EQ.0.D0) GO TO 100
      IF(OMEGAR.EQ.0.D0.OR.OMEGAR.GE.OMEGAL) GO TO 105
      CTR = -1.D0
      CALL DQDAWO(FINTDER,ZERO,OMEGAR,IWEIGH,TAU,ERRABS,ERREL,FLA,
     1ERREST)
      CALL DQDAWO(FINTDER,OMEGAR,OMEGAL,IWEIGH,TAU,ERRABS,ERREL,FLB,
     1ERREST)
      FL2 = FLA+FLB
      CTR = 1.D0
      CALL DQDAWO(FINTDER,ZERO,OMEGAR,IWEIGH,TAU,ERRABS,ERREL,FLA,
     1ERREST)
      CALL DQDAWO(FINTDER,OMEGAR,OMEGAL,IWEIGH,TAU,ERRABS,ERREL,FLB,
     1ERREST)
      FL3 = FLA+FLB
      GO TO 110
  105 CTR = -1.D0
      CALL DQDAWO(FINTDER,ZERO,OMEGAL,IWEIGH,TAU,ERRABS,ERREL,FL2,
     1ERREST)
      CTR = 1.D0
      CALL DQDAWO(FINTDER,ZERO,OMEGAL,IWEIGH,TAU,ERRABS,ERREL,FL3,
     1ERREST)
      GO TO 110
  100 FL2 = 0.D0
      FL3 = 0.D0
  110 IF(OMEGAR.LE.OMEGAH) GO TO 115
      CTR = -1.D0
      CALL DQDAWO(FINTDER,OMEGAH,OMEGAR,IWEIGH,TAU,ERRABS,ERREL,FHA,
     1ERREST)
      CALL DQDAWF(FINTDER,OMEGAR,IWEIGH,TAU,ERRABS,FHB,ERREST)
      FH2 = FHA+FHB
      CTR = 1.D0
      CALL DQDAWO(FINTDER,OMEGAH,OMEGAR,IWEIGH,TAU,ERRABS,ERREL,FHA,
     1ERREST)
      CALL DQDAWF(FINTDER,OMEGAR,IWEIGH,TAU,ERRABS,FHB,ERREST)
      FH3 = FHA+FHB
      GO TO 120
  115 CTR = -1.D0
      CALL DQDAWF(FINTDER,OMEGAH,IWEIGH,TAU,ERRABS,FH2,ERREST)
      CTR = 1.D0
      CALL DQDAWF(FINTDER,OMEGAH,IWEIGH,TAU,ERRABS,FH3,ERREST)
  120 F1 = FUNC(NP)
      F2 = FL2+FH2
```

```
      F3 = FL3+FH3
      Q2 = F1+2.D0*OMEGA0SQ*F2
      Q2 = 8.D0*FLAMDA*OMEGAC*Q2/PI
      DER(2) = (D2-Q2)*C0
      Q3 = (OMEGA0SQ+2.D0*FLAMDASQ)*F1-4.D0*FLAMDASQ*OMEGA0SQ*F3
      Q3 = 4.D0*Q3/PI
      DER(3) = (D3-Q3)*C0
C     DER4 = DFIT/DB0
      DER(4) = BGRS(NP)/PI
C     DER5 = DFIT/DALPHA
      DER(5) = B0*BGRDER(TAU,ALPHA)/PI
      RETURN
      END
C==========================================================
      FUNCTION RXXC(TAU,NP)
C
C     RXXC IS THE CORRECTED AUTO-CORRELATION FUNCTION
C     FOR A VALUE OF TAU.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER (NPMAX=1025)
      DIMENSION FUNC(NPMAX),RXXCS(NPMAX),BGRS(NPMAX)
      COMMON/PAR/OMEGAL,OMEGAH,PI
      COMMON/VAR/C0,B0,ALPHA,OMEGA0SQ,OMEGAC,OMEGAR,FLAMDA,FLAMDASQ
      COMMON/DE/FUNC,RXXCS,BGRS
      SAVE ERRABS,ERREL,IWEIGH,ZERO
      DATA ERRABS,ERREL,IWEIGH,ZERO/1.D-8,1.D-8,1,0.D0/
      EXTERNAL FINT
      F1 = RXXI(TAU)
      IF(OMEGAL.EQ.0.D0) GO TO 100
      IF(OMEGAR.EQ.0.D0.OR.OMEGAR.GE.OMEGAL) GO TO 105
      CALL DQDAWO(FINT,ZERO,OMEGAR,IWEIGH,TAU,ERRABS,ERREL,F2A,ERREST)
      CALL DQDAWO(FINT,OMEGAR,OMEGAL,IWEIGH,TAU,ERRABS,ERREL,F2B,ERREST)
      F2 = F2A+F2B
      GO TO 110
  105 CALL DQDAWO(FINT,ZERO,OMEGAL,IWEIGH,TAU,ERRABS,ERREL,F2,ERREST)
      GO TO 110
  100 F2 = 0.D0
  110 IF(OMEGAR.LE.OMEGAH) GO TO 115
      CALL DQDAWO(FINT,OMEGAH,OMEGAR,IWEIGH,TAU,ERRABS,ERREL,F3A,ERREST)
      CALL DQDAWF(FINT,OMEGAR,IWEIGH,TAU,ERRABS,F3B,ERREST)
      F3 = F3A+F3B
      GO TO 120
  115 CALL DQDAWF(FINT,OMEGAH,IWEIGH,TAU,ERRABS,F3,ERREST)
  120 A = 4.D0*FLAMDA*OMEGA0SQ/PI
      F23 = F2+F3
      FUNC(NP) = F23
      RXXC = F1-A*F23
      RETURN
      END
C==========================================================
      FUNCTION RXXI(TAU)
C
C     RXXI IS THE IDEAL AUTO-CORRELATION FUNCTION
C     FOR A VALUE OF TAU.
C
      IMPLICIT REAL*8 (A-H,O-Z)
```

```
      COMMON/VAR/C0,B0,ALPHA,OMEGA0SQ,OMEGAC,OMEGAR,FLAMDA,FLAMDASQ
      FAC = FLAMDA/OMEGAC
      X1 = FLAMDA*TAU
      X2 = OMEGAC*TAU
      RXXI = DEXP(-X1)*(DCOS(X2)+FAC*DSIN(X2))
      RETURN
      END
C========================================================
      SUBROUTINE RXXIDER(TAU,D2,D3)
C
C     RXXIDER CALCULATES THE OMEGAC-DERIVATIVE AND THE
C     FLAMDA-DERIVATIVE OF THE IDEAL ACF.
C
C     D2 = DRXXI/DOMEGAC, D3 = DRXXI/DFLAMDA
C
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/VAR/C0,B0,ALPHA,OMEGA0SQ,OMEGAC,OMEGAR,FLAMDA,FLAMDASQ
      X1 = FLAMDA*TAU
      X2 = OMEGAC*TAU
      XS = DSIN(X2)
      XC = DCOS(X2)
      XE = DEXP(-X1)
      D2 = XE*(X1*XC/OMEGAC-(TAU+FLAMDA/OMEGAC**2)*XS)
      D3 = -XE*(TAU*XC+(X1-1.D0)*XS/OMEGAC)
      RETURN
      END
C========================================================
      FUNCTION FINT(OMEGA)
C
C     FINT IS THE FUNCTION TO BE INTEGRATED.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/VAR/C0,B0,ALPHA,OMEGA0SQ,OMEGAC,OMEGAR,FLAMDA,FLAMDASQ
      X = OMEGA**2
      FINT = (X-OMEGA0SQ)**2+4.D0*X*FLAMDASQ
      FINT = 1.D0/FINT
      RETURN
      END
C========================================================
      FUNCTION FINTDER(OMEGA)
C
C     FINTDER IS THE FUNCTION TO BE INTEGRATED IN THE FIT FUNCTION
C     DERIVATIVES.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/VAR/C0,B0,ALPHA,OMEGA0SQ,OMEGAC,OMEGAR,FLAMDA,FLAMDASQ
      COMMON/FC/CTR
      FINTDER = FINT(OMEGA)**2*(OMEGA**2+CTR*OMEGA0SQ)
      RETURN
      END
C========================================================
      FUNCTION BGR(TAU,ALPHA,IRET)
C
C     BGR CALCULATES THE BACKGROUND FUNCTION IN THE TIME DOMAIN.
C
C     IBGR = 0 : NO BACKGROUND,
C     IBGR = 1 : BACKGROUND WITH CONSTANT SLOPE IN THE FREQUENCY DOMAIN,
C     IBGR = 2 : EXPONENTIAL BACKGROUND FUNCTION
```

```
C                IN THE FREQUENCY DOMAIN.
C
C     REVISED OCTOBER 20, 1999.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/CONTROL/NPEND,IBGR
      COMMON/PAR/OMEGAL,OMEGAH,PI
 2000 FORMAT(////11X,'MESSAGE FROM BGR, ALPHA = ',1PD12.5)
      IF(IBGR-1) 50,55,200
   50 BGR = 0.D0
      RETURN
   55 IF(ALPHA.NE.0.D0) GO TO 200
      IF(TAU.GT.0.D0) GO TO 100
      BGR = OMEGAH-OMEGAL
      RETURN
  100 BGR = (DSIN(OMEGAH*TAU)-DSIN(OMEGAL*TAU))/TAU
      RETURN
  200 DOMEGA = OMEGAH-OMEGAL
      X1 = ALPHA*DOMEGA
      IF(X1.LE.-1.D2) GO TO 300
      X1 = DEXP(-X1)
      IF(TAU.GT.0.D0) GO TO 210
      IF(DABS(ALPHA)*DOMEGA.LT.1.D-20) GO TO 205
      BGR = (1.D0-X1)/ALPHA
      RETURN
  205 BGR = DOMEGA
      RETURN
  210 X2 = ALPHA**2+TAU**2
      XL = OMEGAL*TAU
      XH = OMEGAH*TAU
      BGR = ALPHA*(DCOS(XL)-X1*DCOS(XH))
      BGR = BGR-TAU*(DSIN(XL)-X1*DSIN(XH))
      BGR = BGR/X2
      RETURN
  300 WRITE(2,2000) ALPHA
      BGR = 0.D0
      IRET = -1
      RETURN
      END
C=============================================================
      FUNCTION BGRDER(TAU,ALPHA)
C
C     BGRDER CALCULATES THE ALPHA-DERIVATIVE OF THE NORMALIZED
C     BACKGROUND FUNCTION IN THE TIME DOMAIN.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/CONTROL/NPEND,IBGR
      COMMON/PAR/OMEGAL,OMEGAH,PI
      IF(IBGR.GT.1) GO TO 200
      BGRDER = 0.D0
      RETURN
  200 DOMEGA = OMEGAH-OMEGAL
      X1 = DEXP(-ALPHA*DOMEGA)
      IF(TAU.GT.0.D0) GO TO 210
      IF(DABS(ALPHA)*DOMEGA.LT.1.D-20) GO TO 205
      BGRDER = ((X1-1.D0)/ALPHA+X1*DOMEGA)/ALPHA
      RETURN
```

```
  205 BGRDER = -DOMEGA**2
      RETURN
  210 X2 = ALPHA**2+TAU**2
      XL = OMEGAL*TAU
      XH = OMEGAH*TAU
      XSL = DSIN(XL)
      XSH = DSIN(XH)
      XCL = DCOS(XL)
      XCH = DCOS(XH)
      S = (TAU**2-ALPHA**2)*(XCL-X1*XCH)
      S = S+2.D0*ALPHA*TAU*(XSL-X1*XSH)
      S = S/X2
      S = S+DOMEGA*X1*(ALPHA*XCH-TAU*XSH)
      BGRDER = S/X2
      RETURN
      END
C============================================================
      SUBROUTINE SORT(X,N)
C
C     SORT IS A SUBROUTINE FOR ASSORTING A SET OF DATA IN ORDER OF
C     INCREASING VALUES.
C
C     N = NUMBER OF DATA POINTS
C     X = VECTOR OF DATA TO BE REPLACED BY THE SET
C         OF ASSORTED DATA
C
      REAL*8 X,XS
      DIMENSION X(1)
      IF(N.LT.2) RETURN
      DO 1 K=1,N-1
      DO 1 I=K+1,N
      IF(X(I).GE.X(K)) GO TO 1
      XS = X(K)
      X(K) = X(I)
      X(I) = XS
    1 CONTINUE
      RETURN
      END
```

# LISTING

## ACFIT7

```
      PROGRAM ACFIT7
C
C     THE CODE FITS FREQUENCY-FILTERED AUTO-CORRELATION FUNCTION (ACF)
C     DATA TO A SECOND-ORDER OSCILLATOR MODEL FOR BWR
C     BOILING INSTABILITY ANALYSIS. THERE ARE THE OPTIONS FOR A
C     ZERO-PARAMETRIC, A ONE-PARAMETRIC OR A TWO-PARAMETRIC BACKGROUND F
C     TAU-RANGE CAN BE SELECTED AND SUCCESSIVELY INCREASED WITHIN 10 AND
C     OF THE MAXIMUM TAU-RANGE. THE CODE ASSUMES THE FORMAT OF THE
C     ACF DATA OUTPUT FROM THE CODE ACCF1.
C
C     FOR THE LEAST-SQUARES FIT THE ROUTINE MARFIT IS USED, WHICH REQUIR
C     THE CALCULATION OF THE DERIVATIVES OF THE FIT FUNCTION.
C     THERE ARE OPTIONS FOR SELECTING WEIGHTING FUNCTIONS.
C     THE CORRECTED ACF IS OBTAINED FROM THE IDEAL ACF.
C     THE NUMERICAL INTEGRATION PROCEDURE (BY THE IMSL ROUTINES DQDAWO
C     AND DQDAWF) IS DIVIDED INTO INTEGRATION SUBINTERVALS.
C     THE PARAMETERS TO BE DETERMINED BY THE FIT ARE :
C         THETA(1) = C0 (AMPLITUDE OF THE ACF)
C         THETA(2) = OMEGAC
C         THETA(3) = FLAMDA
C         THETA(4) = B0    (OPTION)
C         THETA(5) = ALPHA (OPTION)
C
C     CODE WRITTEN BY K.BEHRINGER, OCTOBER 1999, FOR BATCH OPERATION
C     ON THE PSI DEC-ALPHA 2100 COMPUTER.
C
C     COMPILATION COMMAND : FORTRAN/G_FLOATING ACFIT7
C
C     LINK COMMAND        : IMPORT IMSL
C                           LINK ACFIT7,IMSLIBG_STATIC/OPT,IMSLPSECT/OPT
C
C     PARAMETERS :
C
C          NPMAX   = MAXIMUM NUMBER OF THE RIGHT-HAND SIDE ACF DATA.
C                    A CHANGEMENT REQUIRES ADAPTATION IN MANY ROUTINES.
C          NPARMAX = MAXIMUM NUMBER OF FIT PARAMETERS. (=5)
C          NRMAX   = MAXIMUM NUMBER OF TAU-RANGES.
C          IWMAX   = MAXIMUM NUMBER OF AVAILABLE WEIGHTING FUNCTIONS.
C
C     NOTE : IF THE NAMES OF THE DATA OUTPUT FILES (FILE4/5/6/7) ARE CHA
C            A CORRESPONDING CHANGEMENT OF THE LATER ENCODE STATEMENTS
C            IS REQUIRED.
C
C
C     DECLARATIONS
C
      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER (NPMAX=1025,NPARMAX=5,NRMAX=20,IWMAX=6)
      CHARACTER*50 RUN,FILE1,FILE2,FILE3,FINPUT,FILE4,FILE5,FILE6,
     1FILE7
      CHARACTER*20 IFORM,IFORM1,IFORM2
      CHARACTER*3 FFORM
```

```
      REAL*4 SC
      DIMENSION XDATA(NPMAX),YDATA(NPMAX),THETA(NPARMAX),
     1RANGE(NRMAX)
      COMMON/XYDATA/XDATA,YDATA
      COMMON/CONTROL/NPEND,IBGR
      COMMON/PAR/OMEGAL,OMEGAH,PI
      COMMON/W/WA,NWE,NPA1,IWEIGHT
      EQUIVALENCE (FILE3,FINPUT)
      DATA NF,NPLIMIT,PI,SC/1,20,3.141592654D0,1.E30/,
     1IFORM1/'(5X,2E12.4)'/,
     2IFORM2/'(5X,E12.4,12X,E12.4)'/,
     3FILE1/'ACFIT7.IN                                    '/,
     4FILE2/'ACFIT7.PRT                                   '/,
     5FILE3/'<FINPUT>                                     '/,
     6FILE4/'ACFIT7_^^^.DAT                               '/,
     7FILE5/'ACFIT7_^^^.PAR                               '/,
     8FILE6/'ACFIT7_^^^.FIT                               '/,
     9FILE7/'ACFIT7_^^^.PLT                               '/
C
C     FORMATS
C
 1000 FORMAT(4(A/),2(D12.5/),I1/2(D12.5/),I1/D12.5/I2/I1/I3/I2)
 1001 FORMAT(2(D12.5/),I3/I1)
 1005 FORMAT(F6.2)
 2000 FORMAT(1H1,40X,'P R O G R A M     A C F I T 7'///1X,
     1'FILES : PARAMETER DATA : FILE1 = ',A/9X,
     2'PRINT OUTPUT    : FILE2 = ',A/9X,
     3'ACF DATA',7X,': FILE3 = ',A/9X,
     4'DATA OUTPUT',4X,': FILE4 = ',A/9X,
     5'FIT PARAMETER',2X,': FILE5 = ',A/9X,
     6'FIT DATA',7X,': FILE6 = ',A/9X,
     7'PARAMETER PLOT',1X,': FILE7 = ',A///)
 2005 FORMAT(1X,'RUN DENOTATION',36X,'RUN',5X,'=',1X,A//1X,
     1'ACF DATA FILE',37X,'FINPUT',2X,'=',1X,A//1X,
     2'ACF DATA FORMAT',35X,'IFORM',3X,'=',1X,A//1X,
     3'ADDITIONAL DENOTATION OF OUTPUT FILES',13X,'FFORM',3X,
     4'=',1X,A//1X,
     5'LOWER CUTOFF FREQUENCY (HZ)',23X,'FRL',5X,'=',1PD14.5/1X,
     6'UPPER CUTOFF FREQUENCY (HZ)',23X,'FRH',5X,'=',D14.5//1X,
     7'BACKGROUND FIT PARAMETER (0:B0=0,1:B0,2:B0,ALPHA)',1X,
     8'IBGR',4X,'=',I4/1X,
     9'INITIAL BACKGROUND AMPLITUDE (0/0 OF YDATA(1))',4X,
     A'B0FIN',3X,'=',D14.5/1X,
     B'INITIAL BACKGROUND SLOPE INDEX',20X,'SLOPE',3X,'=',D14.5//1X,
     C'PARAMETER FOR WEIGHTING (1:U, 2:L.DCR., 3:COS)',4X,
     D'IWEIGHT =',I4/6X,'(MODIFIED INITIAL PART : 4:1,5:2,6:3)'/1X,
     E'INITIAL WEIGHT AMPLITUDE FACTOR',19X,'WA',6X,'=',D14.5/1X,
     F'END POINT OF WEIGHT FUNCTION INCREASE'13X,'NWE',5X,'=',I4//1X,
     G'WEIGHT ENDPOINT (1:NPEND, 2:NPA)',18X,'IWEND',3X,'=',I4//1X,
     H'INITIAL NUMBER OF DATA OUTPUT FILE',16X,'IDAT',4X,'=',I4//1X,
     I'REQUESTED NUMBER OF JOBS',26X,'JOBMAX',2X,'=',I4/)
 2010 FORMAT('1ACCEPTED  NUMBER OF JOBS',26X,'JOBMAX',2X,'=',I5//1X,
     1'RANGE VALUES (0/0)',44X,'N',3X,'RANGE'/)
 2015 FORMAT(61X,I3,2X,F6.2)
 2016 FORMAT(//1X,'DATA FOR CONVERGENCE (SUBROUTINE FITMAR) :'//6X,
     1'INITIAL VALUE FOR XLAMDA (1.D-8 - 1.D2)',6X,'XLIN',4X,'=',
     21PD15.5//6X,
     3'CONVERGENCE PARAMETER (0/0, 1.D-4 - 1.D0)',4X,'EPSIL',3X,'=',
```

```
     4D15.5//6X,
     5'ALLOWED NUMBER OF ITERATIONS (50 - 999)',6X,'ITMAX',3X,'=',
     6I5//6X,
     7'DATA OUTPUT DURING FIT (0:NO, 1:YES)',9X,'IOUT',4X,'=',I5/)
 2020 FORMAT(////11X,'OPENING ERROR ! FILE = ',I1)
 2025 FORMAT(////11X,'READ-ERROR IN PARAMETER DATA !')
 2030 FORMAT(////11X,'ERRANEOUS PARAMETER DATA !')
 2035 FORMAT(1X,'AVAILABLE NUMBER OF ACF DATA',22X,'NPA',5X,'=',I5//1X,
     1'MAXIMUM TAU-RANGE (SEC)',27X,'TAUMAX',2X,'=',1PD15.5)
 2040 FORMAT('1JOB NUMBER',40X,'JOB',5X,'=',I4//1X,
     1'NUMBER OF DATA OUTPUT FILE',24X,'IDAT',4X,'=',I4//1X,
     2'RANGE VALUE (0/0)',33X,'RANGE',3X,'=',F7.2//1X,
     3'LAST DATA POINT NUMBER',28X,'NPEND',3X,'=',I5//1X,
     4'TAU-RANGE (SEC)',35X,'TAUEND',2X,'=',1PD14.5/)
 2045 FORMAT(1X,'ESTIMATED FIT PARAMETERS (RELATIVE VALUES)',8X,1P,
     1'THETA',3X,'=',5D14.5/1X,
     2'REDUCED CHI-SQUARE OF THE FIT',21X,'CHISQR',2X,'=',D14.5/1X,
     3'NUMBER OF ITERATIONS',30X,'IT',6X,'=',I4/)
 2050 FORMAT(/1X,'ESTIMATED FIT PARAMETERS (ABSOLUTE VALUES)',//11X,
     1'PSD FUNCTION AMPLITUDE',18X,'A0',6X,'=',1PD14.5/11X,
     2'BACKGROUND AMPLITUDE',20X,'B0',6X,'=',D14.5/11X,
     3'BACKGROUND DECAY CONSTANT (SEC/RAD)',5X,'ALPHA',3X,'=',D14.5//
     411X,'PSD RESONANCE FREQUENCY (HZ)',12X,'FR0',5X,'=',D14.5/11X,
     5'ACF FREQUENCY (HZ)',22X,'FR0CF',3X,'=',D14.5//11X,
     6'RESONANCE QUALITY FACTOR',16X,'QR',6X,'=',D14.5/11X,
     7'DECAY CONSTANT (1/SEC)',18X,'FLAMDA',2X,'=',D14.5/11X,
     8'DECAY RATIO',29X,'DR',6X,'=',D14.5//11X,
     9'PEAK AMPLITUDE IN THE PSD',15X,'PEAKMAX =',D14.5/11X,
     A'PEAK BACKGROUND AMPLITUDE',15X,'B0PEAK',2X,'=',D14.5)
 2055 FORMAT(////11X,'NPEND TOO SMALL !')
 2060 FORMAT(////11X,'E N D')
 4000 FORMAT(4X,I4,1P,4E14.5)
 5000 FORMAT(I4,2X,1P,9E14.5/6X,E14.5)
C
C     INPUT PARAMETER DATA
C
      OPEN(UNIT=2,FILE=FILE2,STATUS='NEW',DEFAULTFILE='DIRINPUT')
      WRITE(2,2000) FILE1,FILE2,FILE3,FILE4,FILE5,FILE6,FILE7
      OPEN(UNIT=1,FILE=FILE1,STATUS='OLD',DEFAULTFILE='DIRINPUT',
     1ERR=100)
      READ(1,1000,ERR=105,END=105) RUN,FINPUT,IFORM,FFORM,FRL,FRH,
     1IBGR,B0FIN,SLOPE,IWEIGHT,WA,NWE,IWEND,IDAT,JOBMAX
      IF(IFORM.EQ.'*') IFORM=IFORM1
      IF(IFORM.EQ.'**') IFORM=IFORM2
      IF(FRL.LT.0.D0) FRL=0.D0
      IF(IBGR.LT.0) IBGR=0
      IF(IBGR.GT.2) IBGR=2
      IF(B0FIN.LT.0.D0) B0FIN=0.D0
      IF(B0FIN.GT.1.D3) B0FIN=1.D3
      IF(IBGR.EQ.0) B0FIN=0.D0
      IF(IBGR.EQ.0) SLOPE=0.D0
      IF(IWEIGHT.LT.1) IWEIGHT=1
      IF(IWEIGHT.GT.IWMAX) IWEIGHT=IWMAX
      IF(IWEIGHT.LT.4.OR.NWE.LE.0.OR.WA.LT.0.D0) THEN
      NWE = 0
      WA = 0.D0
      ELSE
```

```
      IF(NWE.EQ.1) NWE=2
      IF(NWE.GT.NPLIMIT) NWE=NPLIMIT
      IF(WA.GT.1.D0) WA=1.D0
      END IF
      IF(IWEND.LT.1.OR.IWEIGHT.EQ.1) IWEND=1
      IF(IWEND.GT.2) IWEND=2
      IF(IDAT.LT.1) IDAT=1
      IF(IDAT.GT.1000-NRMAX) IDAT=1000-NRMAX
      IF(JOBMAX.LT.1) JOBMAX=1
      IF(JOBMAX.GT.NRMAX) JOBMAX=NRMAX
      WRITE(2,2005) RUN,FINPUT,IFORM,FFORM,FRL,FRH,IBGR,
     1B0FIN,SLOPE,IWEIGHT,WA,NWE,IWEND,IDAT,JOBMAX
      IF(FRH.LT.1.1D0*FRL) GO TO 120
      READ(1,1001,ERR=105,END=105) XLIN,EPSIL,ITMAX,IOUT
      IF(XLIN.LT.1.D-8) XLIN=1.D-8
      IF(XLIN.GT.1.D2) XLIN=1.D2
      IF(EPSIL.LT.1.D-4) EPSIL=1.D-4
      IF(EPSIL.GT.1.D0) EPSIL=1.D0
      IF(ITMAX.LT.50) ITMAX=50
      IF(ITMAX.GT.999) ITMAX=999
      IF(IOUT.LT.0) IOUT=0
      IF(IOUT.GT.1) IOUT=1
      DO 1 N=1,JOBMAX
    1 READ(1,1005,ERR=105,END=110) RANGE(N)
      GO TO 115
  110 JOBMAX = N-1
  115 WRITE(2,2010) JOBMAX
      IF(JOBMAX.EQ.0) GO TO 120
      CALL SORT(RANGE,JOBMAX)
      DO 2 N=1,JOBMAX
    2 WRITE(2,2015) N,RANGE(N)
      DO 3 N=1,JOBMAX
      IF(RANGE(N).LT.1.D1.OR.RANGE(N).GT.9.999D1) GO TO 120
    3 RANGE(N) = 1.D-2*RANGE(N)
      WRITE(2,2016) XLIN,EPSIL,ITMAX,IOUT
      EPSIL = 1.D0+1.D-2*EPSIL
      B0FIN = 1.D-2*B0FIN
      IF(IBGR.EQ.2.AND.SLOPE.EQ.0.D0.OR.
     1DABS(SLOPE).GT.6.D0) GO TO 120
      CLOSE(UNIT=1,STATUS='KEEP')
      NF = 5
      ENCODE(3,'(A)',FILE5(8:10)) FFORM
      OPEN(UNIT=5,FILE=FILE5,STATUS='NEW',DEFAULTFILE='DIRINPUT',
     1ERR=100)
      NF = 3
      OPEN(UNIT=3,FILE=FINPUT,STATUS='OLD',DEFAULTFILE='DIRINPUT',
     1ERR=100)
      NF = 7
      ENCODE(3,'(A)',FILE7(8:10)) FFORM
      OPEN(UNIT=7,FILE=FILE7,STATUS='NEW',DEFAULTFILE='DIRINPUT',
     1ERR=100)
C
C     INPUT ACF DATA
C
      CALL INPUT(NPMAX,IFORM,XDATA,YDATA,NPA)
      IF(NPA.LT.NPLIMIT) STOP 105
      TAUMAX = XDATA(NPA)
      WRITE(2,2035) NPA,TAUMAX
```

```
      CLOSE(UNIT=3,STATUS='KEEP')
C
C     COMPUTATION AND DATA READ-OUT
C
      NF = 4
      DPI = 2.D0*PI
      OMEGAL = DPI*FRL
      OMEGAH = DPI*FRH
      NPAR = NPARMAX
      NPA1 = NPA-1
      IF(IBGR.EQ.1) NPAR=NPAR-1
      IF(IBGR.EQ.0) NPAR=NPAR-2
      CALL INIT(YDATA(1),B0FIN,SLOPE,THETA)
      C0S = THETA(1)
      DO 10 JOB=1,JOBMAX
      NPEND = RANGE(JOB)*NPA
      IF(IWEND.EQ.1) NPA1=NPEND-1
      TAUEND = XDATA(NPEND)
      WRITE(2,2040) JOB,IDAT,1.D2*RANGE(JOB),NPEND,TAUEND
      IF(NPEND.LT.NPLIMIT) GO TO 125
      IF(IOUT.EQ.0) GO TO 129
      NF = 6
      ENCODE(3,'(I3)',FILE6(8:10)) IDAT
      OPEN(UNIT=6,FILE=FILE6,STATUS='NEW',DEFAULTFILE='DIRINPUT',
     1ERR=100)
      NF = 4
  129 IREP1 = 0
      IF(THETA(1).EQ.C0S) IREP1=1
  130 IREP2 = 0
      CALL FITMAR(NPAR,THETA,XLIN,EPSIL,ITMAX,IOUT,CHISQR,IT,IREP2,
     1*132)
      WRITE(2,2045) THETA,CHISQR,IT
      C0 = THETA(1)
      FR0CF = THETA(2)
      FLAMDA = THETA(3)
C     REPETITION OF THE FIT
      IF(C0.GT.0.D0.AND.FR0CF.GT.0.D0.AND.
     1FLAMDA.GT.-1.D-1.AND.IREP2.EQ.0) GO TO 131
      IF(IREP1.GE.1) GO TO 132
      CALL INIT(YDATA(1),B0FIN,SLOPE,THETA)
      IREP1 = 1
      GO TO 130
C*
  131 B0 = THETA(4)
      ALPHA = THETA(5)
      FLAMDASQ = FLAMDA**2
      OMEGA0SQ = FR0CF**2+FLAMDASQ
      FR0 = DSQRT(OMEGA0SQ)
      DR = DEXP(-DPI*FLAMDA/FR0CF)
      A1 = 4.D0*FLAMDA*C0
      A0 = A1/OMEGA0SQ
      QR = 0.D0
      IF(FLAMDA.GT.0.D0) QR=5.D-1*FR0/FLAMDA
      PEAKMAX = 0.D0
      IF(DR.LT.1.D0) PEAKMAX=A1*FINT(FR0)
      B0PEAK = 0.D0
      IF(PEAKMAX.GT.0.D0) B0PEAK=B0*DEXP(-ALPHA*(FR0-OMEGAL))
```

```
      FR0 = FR0/DPI
      FR0CF = FR0CF/DPI
      WRITE(2,2050) A0,B0,ALPHA,FR0,FR0CF,QR,FLAMDA,DR,PEAKMAX,
     1B0PEAK
      WRITE(5,5000) JOB,SNGL(TAUEND),SNGL(C0),SNGL(A0),SNGL(B0),
     1SNGL(ALPHA),SNGL(FR0CF),SNGL(FLAMDA),SNGL(DR),SNGL(PEAKMAX),
     2SNGL(CHISQR)
      WRITE(7,5000) JOB,SNGL(TAUEND),SNGL(C0),SNGL(A0),SNGL(B0),
     1SNGL(ALPHA),SNGL(FR0CF),SNGL(FLAMDA),SNGL(DR),SNGL(PEAKMAX),
     2SNGL(CHISQR)
      ENCODE(3,'(I3)',FILE4(8:10)) IDAT
      OPEN(UNIT=4,FILE=FILE4,STATUS='NEW',DEFAULTFILE='DIRINPUT',
     1ERR=100)
      DO 11 N=1,NPA
      IF(N.GT.NPEND) GO TO 135
      YFIT = FIT(THETA,N)
      DIF = YDATA(N)-YFIT
      GO TO 11
  135 YFIT = 0.D0
      DIF = 0.D0
   11 WRITE(4,4000) N,SNGL(XDATA(N)),SNGL(YDATA(N)),SNGL(YFIT),
     1SNGL(DIF)
      CLOSE(UNIT=4,STATUS='KEEP')
      IF(IOUT.EQ.1) CLOSE(UNIT=6,STATUS='KEEP')
C     RESETS
      IF(FLAMDA.LT.0.D0) THETA(3)=0.D0
      IF(DR.LT.2.D-1) CALL INIT(YDATA(1),B0FIN,SLOPE,THETA)
      IF(B0.LT.0.D0.OR.ALPHA.GT.1.D1)
     1CALL INIT1(YDATA(1),B0FIN,SLOPE,THETA)
C*
      GO TO 10
  132 WRITE(5,5000) JOB,SNGL(TAUEND),SC,SC,SC,SC,SC,SC,SC,SC,SC
      CALL INIT(YDATA(1),B0FIN,SLOPE,THETA)
      IF(IOUT.EQ.1) CLOSE(UNIT=6,STATUS='DELETE')
      GO TO 10
  125 WRITE(2,2055)
   10 IDAT = IDAT+1
      WRITE(2,2060)
      STOP
  100 WRITE(2,2020) NF
      STOP
  105 WRITE(2,2025)
      STOP
  120 WRITE(2,2030)
      STOP
      END
C=============================================================
      SUBROUTINE INPUT(NPMAX,IFORM,XDATA,YDATA,NPA)
C
C     DATA INPUT. IT SPECIFICALLY REFERS TO THE OUTPUT DATA
C     GIVEN BY THE CODE ACCF.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*4 XS,YS
      CHARACTER*20 IFORM
      DIMENSION XDATA(NPMAX),YDATA(NPMAX)
 2000 FORMAT(1X,'EOF AT ACF DATA POINT NUMBER',22X,'NEOF',4X,'=',I5/)
 2005 FORMAT(////11X,'READ=ERROR IN ACF DATA ! NERROR =',I5)
```

```
      DO 1 N=1,NPMAX
      READ(3,IFORM,ERR=100,END=105) XS,YS
      IF(XS.GE.0.) GO TO 120
    1 CONTINUE
  110 NPA = 0
      RETURN
  100 NERROR = N
  115 WRITE(2,2005) NERROR
      STOP 100
  105 NEOF = N
      WRITE(2,2000) NEOF
      GO TO 110
  120 XDATA(1) = DBLE(XS)
      YDATA(1) = DBLE(YS)
      NS = N-1
      DO 2 N=2,NPMAX
      READ(3,IFORM,ERR=125,END=130) XS,YS
      XDATA(N) = DBLE(XS)
    2 YDATA(N) = DBLE(YS)
      NPA = NPMAX
      RETURN
  125 NERROR = N+NS
      GO TO 115
  130 NEOF = N+NS
      NPA = N-1
      WRITE(2,2000) NEOF
      RETURN
      END
C=============================================================
      SUBROUTINE INIT(YDATA1,B0FIN,SLOPE,THETA)
C
C     INITIALIZATION OF THETA.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION THETA(1)
      COMMON/PAR/OMEGAL,OMEGAH,PI
      COMMON/CONTROL/NPEND,IBGR
C     C0 :
      THETA(1) = YDATA1
C     OMEGAC :
      THETA(2) = 5.D-1*(OMEGAL+OMEGAH)
      IF(OMEGAL.LE.2.D-1*PI.OR.OMEGAH.GT.3.D0*PI) THETA(2)=PI
C     FLAMDA : (FOR A DECAY RATIO OF ABOUT 0.5)
      THETA(3) = 1.103D-1*THETA(2)
      ENTRY INIT1(YDATA1,B0FIN,SLOPE,THETA)
C     B0 AND ALPHA :
      IF(IBGR.EQ.0) THEN
      THETA(4) = 0.D0
      THETA(5) = 0.D0
      ELSE
      THETA(4) = B0FIN*YDATA1
      THETA(5) = -5.D-1*SLOPE*DLOG(1.D1)/PI
      END IF
      RETURN
      END
C=============================================================
      SUBROUTINE FITMAR(NPAR,THETA,XLIN,EPSIL,ITMAX,IOUT,CHISQR1,IT,
```

```
     1IREP,*)
C
C     FITMAR CONTROLS THE LEAST-SQUARES FIT ROUTINE MARFIT.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER (NPMAX=1025,NPARMAX=5)
      DIMENSION WEIGHTS(NPMAX),THETA(NPARMAX)
      SAVE XLMIN,XLMAX
      DATA XLMIN,XLMAX/1.D-10,1.D6/
      COMMON/MARF/WEIGHTS,XLAMDA,CHISQ,CHISQR,MODE
      COMMON/CONTROL/NPEND,IBGR
      EXTERNAL FIT,FITDER
 2000 FORMAT(1X,'NUMBER OF ITERATIONS EXCEEDS ITMAX')
 2005 FORMAT(1X,'ZERO-DETERMINANT')
 2010 FORMAT(1X,'XLAMDA EXCEEDS XLMAX')
 6000 FORMAT(I4,2X,1PD14.5,2D17.8,5D14.5)
      IRET = 0
      DO 1 N=1,NPEND
    1 WEIGHTS(N) = WEIGHT(N)
      MODE = 1
      XLAMDA = XLIN
      IF(XLAMDA.LT.XLMIN) XLAMDA=XLMIN
      IF(XLAMDA.GT.XLMAX) XLAMDA=XLMAX
      DO 2 I=1,ITMAX+1
      IT = I-1
      CALL MARFIT(FIT,FITDER,NPAR,THETA,IRET)
      IF(IRET.LT.0) RETURN 1
      CHISQR1 = CHISQR
      IF(IOUT.EQ.1) WRITE(6,6000) IT,XLAMDA,CHISQ,CHISQR,THETA
      IF(CHISQR.LT.0.D0) GO TO 200
      IF(XLAMDA.LT.XLMIN) XLAMDA=XLMIN
      IF(XLAMDA.GT.XLMAX) GO TO 205
      IF(I.GT.1) GO TO 100
      MODE = 2
      GO TO 2
  100 IF(CHISQ/CHISQR.LE.EPSIL.AND.XLAMDA.LE.1.D-4) RETURN
      CHISQR = 1.00000001D0*CHISQR
    2 CONTINUE
      WRITE(2,2000)
      GO TO 210
  200 WRITE(2,2005)
      GO TO 210
  205 WRITE(2,2010)
  210 IREP = IREP+1
      RETURN
      END
C========================================================
      SUBROUTINE MARFIT(FN,FNDERIV,NTERMS,C,IRET)
C
C     MARFIT IS A LEAST-SQUARES FIT ROUTINE FOR A ONE-DIMENSIONAL
C     NONLINEAR FUNCTION OF AN INDEPENDENT VARIABLE WITH AN ARBITRARY
C     NUMBER OF PARAMETERS TO BE DETERMINED BY THE FIT. THE ROUTINE
C     IS BASED ON A COMBINATION OF THE GAUSS-NEWTON METHOD WITH THE
C     GRADIENT-EXPANSION METHOD BY D.W.MARQUARDT (AN ALGORITHM FOR
C     LEAST-SQUARES ESTIMATION OF NONLINEAR PARAMETERS, J.SOC.INDUST.
C     APPL.MATH., VOL 11, NO.2 (1963) 431).
C
C     CODE DEVELOPED AND DOCUMENTED BY K.BEHRINGER, INTERNAL EIR REPORT
```

```
C     TM-PH-373 (1970).
C     CODE ADAPTED TO THE PRESENT PROBLEM BY K.BEHRINGER, JUNE 1999.
C
C     FN      : FIT FUNCTION AT A SINGLE ARGUMENT POINT,
C     FNDERIV : SUBROUTINE FOR CALCULATING THE DERIVATIVES OF THE
C               FIT FUNCTION AT A SINGLE ARGUMENT POINT,
C     NTERMS  : NUMBER OF PARAMETERS,
C     C       : PARAMETER VECTOR TO BE DETERMINED BY THE FIT.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER (NPMAX=1025,NPARMAX=5)
      DIMENSION YFIT(NPMAX),XDATA(NPMAX),YDATA(NPMAX),WEIGHTS(NPMAX),
     1ALPHA(NPARMAX,NPARMAX),ERROR(NPARMAX,NPARMAX),SIGMAC(NPARMAX),
     2BETA(NPARMAX),DERIV(NPARMAX),C(NPARMAX)
      COMMON/XYDATA/XDATA,YDATA
      COMMON/CONTROL/NPTS,IBGR
      COMMON/MARF/WEIGHTS,XLAMDA,CHISQ,CHISQR,MODE
      IF(MODE-2) 11,12,13
   11 IF(NPTS.GT.NTERMS) GO TO 14
      CHISQR = -2.D0
      RETURN
   14 DO 1 I=1,NPTS
      YFIT(I) = FN(C,I,IRET)
      IF(IRET.LT.0) RETURN
    1 CONTINUE
   13 CHISQ = 0.D0
      DO 2 I=1,NPTS
    2 CHISQ = CHISQ+WEIGHTS(I)*(YDATA(I)-YFIT(I))**2
      CHISQ = CHISQ/DFLOTJ(NPTS-NTERMS)
      GO TO 15
   12 CHISQ = CHISQR
   15 DO 3 J=1,NTERMS
      BETA(J) = 0.D0
      DO 3 K=1,J
    3 ALPHA(J,K) = 0.D0
      DO 4 I=1,NPTS
      CALL FNDERIV(I,DERIV)
      DO 4 J=1,NTERMS
      BETA(J) = BETA(J)+WEIGHTS(I)*(YDATA(I)-YFIT(I))*DERIV(J)
      DO 4 K=1,J
    4 ALPHA(J,K) = ALPHA(J,K)+WEIGHTS(I)*DERIV(J)*DERIV(K)
      L = 0
   16 DO 5 J=1,NTERMS
      DO 5 K=1,J
      IF(K.EQ.J) GO TO 17
      IF(L.GT.0) GO TO 18
      ALPHA(K,J) = DSQRT(ALPHA(J,J)*ALPHA(K,K))
      IF(ALPHA(K,J)) 22,22,23
   23 ALPHA(J,K) = ALPHA(J,K)/ALPHA(K,J)
   18 ERROR(J,K) = ALPHA(J,K)
      ERROR(K,J) = ERROR(J,K)
      GO TO 5
   17 ERROR(J,J) = 1.D0+XLAMDA
    5 CONTINUE
      CALL MATINV(ERROR,NTERMS,DET)
      IF(DET) 19,22,19
   22 CHISQR = -1.D0
```

```
      RETURN
   19 DO 6 J=1,NTERMS
      SIGMAC(J) = C(J)
      DO 6 K=1,NTERMS
      IF(K.GE.J) GO TO 20
      SIGMAC(J) = SIGMAC(J)+ERROR(J,K)*BETA(K)/ALPHA(K,J)
      GO TO 6
   20 SIGMAC(J) = SIGMAC(J)+ERROR(J,K)*BETA(K)/ALPHA(J,K)
    6 CONTINUE
      CHISQR = 0.D0
      DO 7 I=1,NPTS
      YFIT(I) = FN(SIGMAC,I,IRET)
      IF(IRET.LT.0) RETURN
    7 CHISQR = CHISQR+WEIGHTS(I)*(YDATA(I)-YFIT(I))**2
      CHISQR = CHISQR/DFLOTJ(NPTS-NTERMS)
      IF(CHISQ.GE.CHISQR) GO TO 21
      L = L+1
      XLAMDA = 1.D1*XLAMDA
      IF(L-10) 16,16,21
   21 XLAMDA = 1.D-1*XLAMDA
      DO 8 J=1,NTERMS
      C(J) = SIGMAC(J)
    8 SIGMAC(J) = DSQRT(ERROR(J,J)/ALPHA(J,J))
      RETURN
      END
C===========================================================
      SUBROUTINE MATINV(ARRAY,NORDER,DET)
C
C     MATINV INVERTS A NON-SINGULAR QUADRATIC SYMMETRIC MATRIX
C     ACCORDING TO THE GAUSS-JORDAN ELIMINATION PROCEDURE, AND
C     CALCULATES THE DETERMINANT OF THE GIVEN MATRIX.
C     THE SUBROUTINE IS TAKEN FROM PH.R.BEVINGTON, DATA REDUCTION
C     AND ERROR ANALYSIS FOR THE PHYSICAL SCIENCES, MCGRAW-HILL
C     BOOK COMPANY, 1969.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER (NPARMAX=5)
      DIMENSION ARRAY(NPARMAX,NPARMAX),IK(NPARMAX),JK(NPARMAX)
      DET = 1.D0
      DO 1 K=1,NORDER
      AMAX = 0.D0
      DO 2 I=K,NORDER
      DO 2 J=K,NORDER
      IF(DABS(AMAX).GT.DABS(ARRAY(I,J))) GO TO 2
      AMAX = ARRAY(I,J)
      IK(K) = I
      JK(K) = J
    2 CONTINUE
      IF(AMAX.NE.0.D0) GO TO 11
      DET = 0.D0
      RETURN
   11 I = IK(K)
      IF(I.EQ.K) GO TO 12
      DO 3 J=1,NORDER
      SAVE = ARRAY(K,J)
      ARRAY(K,J) = ARRAY(I,J)
    3 ARRAY(I,J) = -SAVE
   12 J = JK(K)
```

```
      IF(J.EQ.K) GO TO 13
      DO 4 I=1,NORDER
      SAVE = ARRAY(I,K)
      ARRAY(I,K) = ARRAY(I,J)
    4 ARRAY(I,J) = -SAVE
   13 DO 5 I=1,NORDER
      IF(I.NE.K) ARRAY(I,K) = -ARRAY(I,K)/AMAX
    5 CONTINUE
      DO 6 I=1,NORDER
      DO 6 J=1,NORDER
      IF(I.NE.K.AND.J.NE.K) ARRAY(I,J)=ARRAY(I,J)+ARRAY(I,K)*ARRAY(K,J)
    6 CONTINUE
      DO 7 J=1,NORDER
      IF(J.NE.K) ARRAY(K,J)=ARRAY(K,J)/AMAX
    7 CONTINUE
      ARRAY(K,K) = 1.D0/AMAX
    1 DET = DET*AMAX
      DO 8 L=1,NORDER
      K = NORDER-L+1
      J = IK(K)
      IF(J.LE.K) GO TO 14
      DO 9 I=1,NORDER
      SAVE = ARRAY(I,K)
      ARRAY(I,K) = -ARRAY(I,J)
    9 ARRAY(I,J) = SAVE
   14 I = JK(K)
      IF(I.LE.K) GO TO 8
      DO 10 J=1,NORDER
      SAVE = ARRAY(K,J)
      ARRAY(K,J) = -ARRAY(I,J)
   10 ARRAY(I,J) = SAVE
    8 CONTINUE
      RETURN
      END
C=========================================================
      FUNCTION WEIGHT(N)
C
C     A WEIGHT FUNCTION IS CALCULATED FOR THE SELECTED OPTION
C     BY THE PARAMETER IWEIGHT (1-6).
C     NOTE : NPA1=NPA-1
C
C     MODIFICATION OF THE INITIAL PART (IWEIGHT=4-6) :
C     WA  = INITIAL WEIGHT AMPLITUDE FACTOR (0.LE.WA.LT.1.),
C     NWE = END POINT OF WEIGHT FUNCTION INCREASE (1.LT.NPLIMIT)
C           (NPLIMIT IS DEFINED IN THE DATA DECLARATION OF THE
C            MAIN PROGRAM),
C     WF  = WEIGHT FUNCTION FACTOR.
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/W/WA,NWE,NPA1,IWEIGHT
      SAVE PIH
      DATA PIH/1.570796327D0/
      WF = 1.D0
      IF(IWEIGHT.LE.3.OR.N.GE.NWE) GO TO 10
      WF = WA+(1.D0-WA)*DSIN(PIH*DFLOTJ(N-1)/DFLOTJ(NWE))
   10 GO TO (101,102,103,101,102,103) IWEIGHT
C     UNIFORM WEIGHTING :
  101 WEIGHT = WF
```

```
      RETURN
C     LINEARLY DECREASING WEIGHTING :
  102 WEIGHT = WF*DFLOTJ(NPA1+1-N)/DFLOTJ(NPA1)
      RETURN
C     COSINE FUNCTION WEIGHTING :
  103 WEIGHT = WF*DCOS(PIH*DFLOTJ(N-1)/DFLOTJ(NPA1))
      RETURN
      END
C=========================================================
      FUNCTION FIT(THETA,NP,IRET)
C
C     THE FIT FUNCTION IS CALCULATED FOR A VALUE OF TAU.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER (NPMAX=1025)
      DIMENSION THETA(1),XDATA(NPMAX),YDATA(NPMAX),FUNC(NPMAX),
     1RXXCS(NPMAX),BGRS(NPMAX)
      COMMON/XYDATA/XDATA,YDATA
      COMMON/PAR/OMEGAL,OMEGAH,PI
      COMMON/CONTROL/NPEND,IBGR
      COMMON/VAR/C0,B0,ALPHA,OMEGA0,OMEGA0SQ,OMEGAC,FLAMDA,FLAMDASQ
      COMMON/DE/FUNC,RXXCS,BGRS
      TAU = XDATA(NP)
      IF(NP.GT.1) GO TO 100
      C0 = THETA(1)
      IF(DABS(C0).LT.1.D-10) C0=DSIGN(1.D-10,C0)
      OMEGAC = THETA(2)
      FLAMDA = THETA(3)
      B0 = THETA(4)
      IF(DABS(B0).LT.1.D-10.AND.IBGR.GT.0) B0=DSIGN(1.D-10,B0)
      ALPHA = THETA(5)
      IF(DABS(ALPHA).LT.1.D-10.AND.IBGR.GT.1)
     1ALPHA=DSIGN(1.D-10,ALPHA)
      FLAMDASQ = FLAMDA**2
      OMEGACSQ = OMEGAC**2
      OMEGA0SQ = OMEGACSQ+FLAMDASQ
      OMEGA0 =DSQRT(OMEGA0SQ)
  100 A = RXXC(TAU,NP)
      RXXCS(NP) = A
      B = BGR(TAU,ALPHA,IRET)
      BGRS(NP) = B
      FIT = C0*A+B0*B/PI
      RETURN
      END
C=========================================================
      SUBROUTINE FITDER(NP,DER)
C
C     FITDER CALCULATES THE DERIVATIVES OF THE FIT FUNCTION.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER (NPMAX=1025)
      DIMENSION DER(1),XDATA(NPMAX),YDATA(NPMAX),FUNC(NPMAX),
     1RXXCS(NPMAX),BGRS(NPMAX)
      COMMON/XYDATA/XDATA,YDATA
      COMMON/PAR/OMEGAL,OMEGAH,PI
      COMMON/VAR/C0,B0,ALPHA,OMEGA0,OMEGA0SQ,OMEGAC,FLAMDA,FLAMDASQ
      COMMON/FC/CTR
      COMMON/DE/FUNC,RXXCS,BGRS
```

```
      SAVE ERRABS,ERREL,IWEIGH,ZERO
      DATA ERRABS,ERREL,IWEIGH,ZERO/1.D-7,1.D-7,1,0.D0/
      EXTERNAL FINT,FINTDER
      TAU = XDATA(NP)
C     DER1 = DFIT/DC0
      DER(1) = RXXCS(NP)
C     DER2 = DFIT/DOMEGAC, DER3 = DFIT/DFLAMDA
      CALL RXXIDER(TAU,D2,D3)
      IF(OMEGAL.EQ.0.D0) GO TO 100
      IF(OMEGA0.GE.OMEGAL) GO TO 105
      CTR = -1.D0
      CALL DQDAWO(FINTDER,ZERO,OMEGA0,IWEIGH,TAU,ERRABS,ERREL,FLA,
     1ERREST)
      CALL DQDAWO(FINTDER,OMEGA0,OMEGAL,IWEIGH,TAU,ERRABS,ERREL,FLB,
     1ERREST)
      FL2 = FLA+FLB
      CTR = 1.D0
      CALL DQDAWO(FINTDER,ZERO,OMEGA0,IWEIGH,TAU,ERRABS,ERREL,FLA,
     1ERREST)
      CALL DQDAWO(FINTDER,OMEGA0,OMEGAL,IWEIGH,TAU,ERRABS,ERREL,FLB,
     1ERREST)
      FL3 = FLA+FLB
      GO TO 110
  105 CTR = -1.D0
      CALL DQDAWO(FINTDER,ZERO,OMEGAL,IWEIGH,TAU,ERRABS,ERREL,FL2,
     1ERREST)
      CTR = 1.D0
      CALL DQDAWO(FINTDER,ZERO,OMEGAL,IWEIGH,TAU,ERRABS,ERREL,FL3,
     1ERREST)
      GO TO 110
  100 FL2 = 0.D0
      FL3 = 0.D0
  110 IF(OMEGA0.LE.OMEGAH) GO TO 115
      CTR = -1.D0
      CALL DQDAWO(FINTDER,OMEGAH,OMEGA0,IWEIGH,TAU,ERRABS,ERREL,FHA,
     1ERREST)
      CALL DQDAWF(FINTDER,OMEGA0,IWEIGH,TAU,ERRABS,FHB,ERREST)
      FH2 = FHA+FHB
      CTR = 1.D0
      CALL DQDAWO(FINTDER,OMEGAH,OMEGA0,IWEIGH,TAU,ERRABS,ERREL,FHA,
     1ERREST)
      CALL DQDAWF(FINTDER,OMEGA0,IWEIGH,TAU,ERRABS,FHB,ERREST)
      FH3 = FHA+FHB
      GO TO 120
  115 CTR = -1.D0
      CALL DQDAWF(FINTDER,OMEGAH,IWEIGH,TAU,ERRABS,FH2,ERREST)
      CTR = 1.D0
      CALL DQDAWF(FINTDER,OMEGAH,IWEIGH,TAU,ERRABS,FH3,ERREST)
  120 F1 = FUNC(NP)
      F2 = FL2+FH2
      F3 = FL3+FH3
      Q2 = 1.6D1*FLAMDA*OMEGAC*F2/PI
      DER(2) = (D2-Q2)*C0
      Q3 = F1-4.D0*FLAMDASQ*F3
      Q3 = 4.D0*Q3/PI
      DER(3) = (D3-Q3)*C0
C     DER4 = DFIT/DB0
```

```
      DER(4) = BGRS(NP)/PI
C     DER5 = DFIT/DALPHA
      DER(5) = B0*BGRDER(TAU,ALPHA)/PI
      RETURN
      END
C============================================================
      FUNCTION RXXC(TAU,NP)
C
C     RXXC IS THE CORRECTED AUTO-CORRELATION FUNCTION
C     FOR A VALUE OF TAU.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER (NPMAX=1025)
      DIMENSION FUNC(NPMAX),RXXCS(NPMAX),BGRS(NPMAX)
      COMMON/PAR/OMEGAL,OMEGAH,PI
      COMMON/VAR/C0,B0,ALPHA,OMEGA0,OMEGA0SQ,OMEGAC,FLAMDA,FLAMDASQ
      COMMON/DE/FUNC,RXXCS,BGRS
      SAVE ERRABS,ERREL,IWEIGH,ZERO
      DATA ERRABS,ERREL,IWEIGH,ZERO/1.D-8,1.D-8,1,0.D0/
      EXTERNAL FINT
      F1 = RXXI(TAU)
      IF(OMEGAL.EQ.0.D0) GO TO 100
      IF(OMEGA0.GE.OMEGAL) GO TO 105
      CALL DQDAWO(FINT,ZERO,OMEGA0,IWEIGH,TAU,ERRABS,ERREL,F2A,ERREST)
      CALL DQDAWO(FINT,OMEGA0,OMEGAL,IWEIGH,TAU,ERRABS,ERREL,F2B,ERREST)
      F2 = F2A+F2B
      GO TO 110
  105 CALL DQDAWO(FINT,ZERO,OMEGAL,IWEIGH,TAU,ERRABS,ERREL,F2,ERREST)
      GO TO 110
  100 F2 = 0.D0
  110 IF(OMEGA0.LE.OMEGAH) GO TO 115
      CALL DQDAWO(FINT,OMEGAH,OMEGA0,IWEIGH,TAU,ERRABS,ERREL,F3A,ERREST)
      CALL DQDAWF(FINT,OMEG0,IWEIGH,TAU,ERRABS,F3B,ERREST)
      F3 = F3A+F3B
      GO TO 120
  115 CALL DQDAWF(FINT,OMEGAH,IWEIGH,TAU,ERRABS,F3,ERREST)
  120 A = 4.D0*FLAMDA/PI
      F23 = F2+F3
      FUNC(NP) = F23
      RXXC = F1-A*F23
      RETURN
      END
C=============================================================
      FUNCTION RXXI(TAU)
C
C     RXXI IS THE IDEAL AUTO-CORRELATION FUNCTION
C     FOR A VALUE OF TAU.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/VAR/C0,B0,ALPHA,OMEGA0,OMEGA0SQ,OMEGAC,FLAMDA,FLAMDASQ
      FAC = FLAMDA/OMEGAC
      X1 = FLAMDA*TAU
      X2 = OMEGAC*TAU
      RXXI = DEXP(-X1)*(DCOS(X2)-FAC*DSIN(X2))
      RETURN
      END
C============================================================
      SUBROUTINE RXXIDER(TAU,D2,D3)
```

```
C
C     RXXIDER CALCULATES THE OMEGAC-DERIVATIVE AND THE
C     FLAMDA-DERIVATIVE OF THE IDEAL ACF.
C
C     D2 = DRXXI/DOMEGAC, D3 = DRXXI/DFLAMDA
C
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/VAR/C0,B0,ALPHA,OMEGA0,OMEGA0SQ,OMEGAC,FLAMDA,FLAMDASQ
      X1 = FLAMDA*TAU
      X2 = OMEGAC*TAU
      XS = DSIN(X2)
      XC = DCOS(X2)
      XE = DEXP(-X1)
      D2 = -XE*(X1*XC/OMEGAC+(TAU-FLAMDA/OMEGAC**2)*XS)
      D3 = -XE*(TAU*XC+(1.D0-X1)*XS/OMEGAC)
      RETURN
      END
C=========================================================
      FUNCTION FINT(OMEGA)
C
C     FINT IS THE FUNCTION TO BE INTEGRATED.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/VAR/C0,B0,ALPHA,OMEGA0,OMEGA0SQ,OMEGAC,FLAMDA,FLAMDASQ
      FINT = FINT1(OMEGA)*OMEGA**2
      RETURN
      END
C=========================================================
      FUNCTION FINT1(OMEGA)
C
C     FINT1 IS A PART OF THE FUNCTION FINT.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/VAR/C0,B0,ALPHA,OMEGA0,OMEGA0SQ,OMEGAC,FLAMDA,FLAMDASQ
      X = OMEGA**2
      FINT1 = (X-OMEGA0SQ)**2+4.D0*X*FLAMDASQ
      FINT1 = 1.D0/FINT1
      RETURN
      END
C=========================================================
      FUNCTION FINTDER(OMEGA)
C
C     FINTDER IS THE FUNCTION TO BE INTEGRATED IN THE FIT FUNCTION
C     DERIVATIVES.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/VAR/C0,B0,ALPHA,OMEGA0,OMEGA0SQ,OMEGAC,FLAMDA,FLAMDASQ
      COMMON/FC/CTR
      X = OMEGA**2
      FINTDER = FINT1(OMEGA)**2*X*(X+CTR*OMEGA0SQ)
      RETURN
      END
C=========================================================
      FUNCTION BGR(TAU,ALPHA,IRET)
C
C     BGR CALCULATES THE BACKGROUND FUNCTION IN THE TIME DOMAIN.
C
```

```
C     IBGR = O : NO BACKGROUND,
C     IBGR = 1 : BACKGROUND WITH CONSTANT SLOPE IN THE FREQUENCY DOMAIN,
C     IBGR = 2 : EXPONENTIAL BACKGROUND FUNCTION
C                IN THE FREQUENCY DOMAIN.
C
C     REVISED OCTOBER 20, 1999.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/CONTROL/NPEND,IBGR
      COMMON/PAR/OMEGAL,OMEGAH,PI
 2000 FORMAT(////11X,'MESSAGE FROM BGR, ALPHA = ',1PD12.5)
      IF(IBGR-1) 50,55,200
   50 BGR = 0.D0
      RETURN
   55 IF(ALPHA.NE.0.D0) GO TO 200
      IF(TAU.GT.0.D0) GO TO 100
      BGR = OMEGAH-OMEGAL
      RETURN
  100 BGR = (DSIN(OMEGAH*TAU)-DSIN(OMEGAL*TAU))/TAU
      RETURN
  200 DOMEGA = OMEGAH-OMEGAL
      X1 = ALPHA*DOMEGA
      IF(X1.LE.-1.D2) GO TO 300
      X1 = DEXP(-X1)
      IF(TAU.GT.0.D0) GO TO 210
      IF(DABS(ALPHA)*DOMEGA.LT.1.D-20) GO TO 205
      BGR = (1.D0-X1)/ALPHA
      RETURN
  205 BGR = DOMEGA
      RETURN
  210 X2 = ALPHA**2+TAU**2
      XL = OMEGAL*TAU
      XH = OMEGAH*TAU
      BGR = ALPHA*(DCOS(XL)-X1*DCOS(XH))
      BGR = BGR-TAU*(DSIN(XL)-X1*DSIN(XH))
      BGR = BGR/X2
      RETURN
  300 WRITE(2,2000) ALPHA
      BGR = 0.D0
      IRET = -1
      RETURN
      END
C=========================================================
      FUNCTION BGRDER(TAU,ALPHA)
C
C     BGRDER CALCULATES THE ALPHA-DERIVATIVE OF THE NORMALIZED
C     BACKGROUND FUNCTION IN THE TIME DOMAIN.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/CONTROL/NPEND,IBGR
      COMMON/PAR/OMEGAL,OMEGAH,PI
      IF(IBGR.GT.1) GO TO 200
      BGRDER = 0.D0
      RETURN
  200 DOMEGA = OMEGAH-OMEGAL
      X1 = DEXP(-ALPHA*DOMEGA)
      IF(TAU.GT.0.D0) GO TO 210
      IF(DABS(ALPHA)*DOMEGA.LT.1.D-20) GO TO 205
```

```
      BGRDER = ((X1-1.D0)/ALPHA+X1*DOMEGA)/ALPHA
      RETURN
  205 BGRDER = -DOMEGA**2
      RETURN
  210 X2 = ALPHA**2+TAU**2
      XL = OMEGAL*TAU
      XH = OMEGAH*TAU
      XSL = DSIN(XL)
      XSH = DSIN(XH)
      XCL = DCOS(XL)
      XCH = DCOS(XH)
      S = (TAU**2-ALPHA**2)*(XCL-X1*XCH)
      S = S+2.D0*ALPHA*TAU*(XSL-X1*XSH)
      S = S/X2
      S = S+DOMEGA*X1*(ALPHA*XCH-TAU*XSH)
      BGRDER = S/X2
      RETURN
      END
C===========================================================
      SUBROUTINE SORT(X,N)
C
C     SORT IS A SUBROUTINE FOR ASSORTING A SET OF DATA IN ORDER OF
C     INCREASING VALUES.
C
C     N = NUMBER OF DATA POINTS
C     X = VECTOR OF DATA TO BE REPLACED BY THE SET
C         OF ASSORTED DATA
C
      REAL*8 X,XS
      DIMENSION X(1)
      IF(N.LT.2) RETURN
      DO 1 K=1,N-1
      DO 1 I=K+1,N
      IF(X(I).GE.X(K)) GO TO 1
      XS = X(K)
      X(K) = X(I)
      X(I) = XS
    1 CONTINUE
      RETURN
      END
```

## 11. PROGRAM ACFIT7SA

Precision : double

Operation : background

Required auxiliary routines : from IMSL : DQDAWO, DQDAWF

Purpose of the program :

Least-Squares Fit of the Auto-Correlation Function of Model B to a Set of Estimated Auto-Correlation Functions in the Gliding Segment Analysis.

Feature Summary :

- Repetetive application of the code ACFIT7
- The set of auto-correlation functions to be input is assumed to be estimated by the code ACCF2. The set size is limited.
- Two files for the parameter data input must be edited for a run.

### 11.1 Comments

The code ACFIT7 (Section 10) performs a least-squares fitting of the theoretical auto-correlation function (ACF) of the oscillator model B to one experimental ACF estimated on unfiltered or previously band-pass filtered signal data. As outlined in Section 7.5, information about the uncertainty of the fit parameter data and other derived data, like the decay ratio DR, can be obtained by considering either fits to instantaneous ACFs estimated for each segment of the signal data, which has been called a segment analysis (SA), or fits to 'short-time' ACFs estimated over a small fixed number of segments. These 'short-time' ACFs move, subsequently shifted by one segment, over the signal record until no further segment is available. This type of analysis has briefly been called a gliding segment analysis (GLSA) and has been found to be superior to the SA. The code ACFIT7SA allows least-squares fitting the ACF of the model B to this set of estimated ACFs within one run. A similar code version for model A has not been considered, since model B has been experienced to be more suitable than model A with respect to its peak resonance frequency problem.

The code is simply built-up by using repetetively the code ACFIT7 as a subroutine. It assumes that the set of ACFs to be put in, have been estimated by the code ACCF2

(Section 8). The treatment of an estimated ACF is called a subrun. Within each subrun there are again the jobs according to the given number of fit range data.

Only two files for the parameter data input are required. On the first one, one has mainly to specify the set size of the ACFs, i.e. the number of subruns. The second file contains the input parameter data as to be given for ACFIT7, but only for the first subrun. They remain valid for the further subruns with the exception of two parameters which concern the serial number of the ACF data files and an extension number in the data output files. The number of subruns is limited. The maximum number amounts 99.

The use of the code ACFIT7 as a subroutine required a few adaptations in the main program. The added listing of ACFIT7SA is represented only up to line 455. The continuation must be taken from the listing of ACFIT7 without any further modifications.

A run of ACFIT7SA can produce a large quantity of data. An allowance for about 100'000 blocks on a data disc is needed. A run normally goes through all requested subruns in the GLSA due to the high assurance provided in ACFIT7 against the appearance of anomalous fit conditions. But as mentioned in Section 10, the IMSL error-handling system is not implemented.

In Section 11.5, the criterion for determining the optimum fit range in the GLSA is given, followed in Section 11.6 with examples of numerical results obtained from 19 analyzed benchmark records. For comparison, results from other benchmark participants using different other methods are listed. The evaluation of the optimum fit range requires the help by the code ACFITEV5 (Section 13).

### 11.2 Files

There are 9 files :

- ACFIT7SA.IN (file unit 10)

This file contains the input parameter data for the run of ACFIT7SA. Only two data must be given.

- ACFIT7SA.PRT (file unit 11)

This file is meant for printing and contains (with text) the input parameter data of ACFIT7SA. Messages can appear which are related to run terminations due to an erroneous subrun handling.

- ACFIT7.IN (file unit 1)

This file contains the same input parameter data which must be specified for a run of ACFIT7. The data refer to all subruns with the exception of the file name for FINPUT and the value to be attributed to the parameter FFORM which concern the first subrun. For more detail, see parameter data input list.

The following 6 files have the same meaning as in the code ACFIT7. Their names differ only by additional extension numbers.

- ACFIT7_''.PRT (file unit 2)

The extension number has two digits with a leading zero, starting always with the value 01. It corresponds to the current subrun number.

- 'FINPUT' (file unit 3)

- ACFIT7_'''_'''.DAT (file unit 4)

There is a double extension number. The first one refers to the current value of FFORM for run and subrun identification. The second one indicates the current number of IDAT for identifying the job within a subrun.

- ACFIT7_'''.PAR (file unit 5)

The extension number refers to the current value of FFORM for run and subrun identification. These files must be saved. The data are required for further processing by the code ACFITEV5 (Section 13).

- ACFIT7_'''_'''.FIT (file unit 6)

The double extension number is the same as in ACFIT7_'''_'''.DAT. These files are optionally opened (by the input parameter IOUT).

ACFIT7_'''.PLT (file unit 7)

The extension number has the same value as in ACFIT7_'''.PAR.

## 11.3 Parameter Data Input on File ACFIT7SA.IN

- line 1, format (A), RUN

A string of max. 50 characters for run identification.

- line 2, format (I4), NSEG

Number of ACF data files to be treated within a run (= number of subruns). It represents the set size of estimated ACFs. In the originally first investigated SA, this number corresponds to the number of segments either used or being available in the signal record.

Internal restrictions : IF(NSEG.LT.1) NSEG=1

IF(NSEG.GT.NSEGMAX) NSEG=NSEGMAX

Where NSEGMAX is an internal parameter set equal 99.

The code does not make a correction of an erroneously given value which exceeds the available ACF set size.

## 11.4 Parameter Data Input on File ACFIT7.IN

- line 1, format (A), RUN

The character string of RUN can be different from that on ACFIT7SA.IN.

- line 2, format (A), FINPUT

The code ACCF2 generates the data output files ACCF0'''.PLO which have serial numbers of exactly 3 digits with leading zeros. They start with an initial number specified there. For FINPUT one has to insert the file name with the lowest number (inclusive leading zeros). This number is then incremented by 1 for each following subrun.

Obviously, if for NSEG a value is given which is larger than the available number of files, a run terminates. The message for the opening error will appear on the next opened print file ACFIT7_''.PRT, if the last ACF file number is less than 999; otherwise a corresponding message is sent to the print file ACFIT7SA.PRT.

- line 3, format (A), IFORM

- line 4, format (A), FFORM

The specification to be given to this 3-character parameter is somewhat different from that in the code ACFIT7. FFORM serves for run and subrun identification in the data output file names. The value attributed to the first character can be a letter or a number (0-9). It is maintained in all subruns. The two other characters are used as a serial number of the subruns. If a leading zero is omitted, the code inserts it. This initial number is incremented by 1 for each following subrun. The allowed, but not assured initial value is from 1 up to 100-NSEG. If the current number exceeds the upper limit, the run terminates with a message on ACFIT7SA.PRT.

- line 5, format (D12.5), FRL

- line 6, format (D12.5), FRH

- line 7, format (I1), IBGR

- line 8, format (D12.5), B0FIN

- line 9, format (D12.5), SLOPE

- line 10, format (I1), IWEIGHT

- line 11, format (D12.5), WA

- line 12, format (I2), NWE

- line 13, format (I1), IWEND

- line 14, format (I3), IDAT

- line 15, format (I2), JOBMAX

- line 16, format (D12.5), XLIN

- line 17, format (D12.5), EPSIL

- line 18, format (I3), ITMAX

- line 19, format (I1), IOUT

- line 20 and following lines, format (F6.2), RANGE

## 11.5 Criterion for the Optimum Fit Range in the GLSA

From the data written on the files ACFIT7_'''.PAR one can calculate average values and standard deviations as a function of the fit range, i.e. of the fit lag time end point $\tau_{end}$ under elimination of cancelled jobs. It was empirically found that the standard deviation of the DR, $s_{DR}(\tau_{end})$ moves through a minimum. This minimum (which is subjected to statistics) is often not very sharp. It mostly lies toward the given lower boundary of $\tau_{end}$ at low DR cases, and viceversa, toward the given upper boundary of $\tau_{end}$ at high DR cases. This criterion has presently been applied for selecting the 'best' fits with the weight function IWEIGHT=3, IWEND=1. The use of this weight function gives in general smoother curves for the average $\overline{DR}(\tau_{end})$ and $s_{DR}(\tau_{end})$ than the use of the weight function IWEIGHT=1. Some studies with the weight functions IWEIGHT>3 did not show any improvements. The average quantity $\overline{\chi_R^2}(\tau_{end})$ is not suitable for establishing a further criterion for the optimum fit range. In general, this quantity increases initially, can run into a plateau or again decrease. But its standard deviation $s_{\chi_R^2}(\tau_{end})$ is of the same order of magnitude as $\overline{\chi_R^2}(\tau_{end})$. It is obvious that the optimum fit range must not coincide between the three-parametric and the five-parametric fit procedure on the same ACF set. Five-parametric fits, where $|B_0|$ and $\alpha$ run to large values, have been accepted.

## 11.6 Numerical Results Obtained from Benchmark Signal Data in the GLSA

Among the 91 benchmark records, the analysis results obtained from 19 records will be represented. The records were measured under nearly stable reactor conditions and belong to low, medium and high DR cases. The data were sampled with an actual sampling frequency of 25 Hz, but decimated to an effective sampling frequency of 12.5 Hz. In the analysis, the most important common parameter data were :

- In FFTF2 : segment length of 256 data points,
- In ACCF2 : short-time ACFs estimated over 5 segments,
- In ACFIT7SA : weight function in the least-squares fits : IWEIGHT=3, IWEND=1,
  fit range r, RANGE=15-50 % in steps of 2.5 % (in low DR cases being visible from PSD plots, the upper boundary was mostly set somewhat less).

The results are listed in Table 11.1. Each record has been treated with the three-parametric (IBGR=0) and the five-parametric (IBGR=2) fit procedure. Column 1 contain the serial record number I. In column 2, the file name of the benchmark record is given. The applied filter bandwidth follows in column 3. It has somewhat subjectively taken from PSD plots with respect to the five-parametric fits. In particular, the lower filter cutoff frequency has been chosen as the smallest value which assures the approximative validity of the assumed PSD background function. The fit option value is given in column 4. In column 5, two values for the number of ACFs involved in the GLSA are listed. The first one denoted by 'g' refers to the given number of ACFs, the second one denoted by 'a' is the accepted number at the optimum fit range. In column 6, the optimum fit range r (in %) and the corresponding value of $\tau_{end}$ are given. Columns 7 and 8 contain the average oscillation frequency $\bar{f}_c = \bar{\omega}_c / 2\pi$ in Hz and the average decay ratio $\overline{DR}$ with their standard deviations at the optimum fit range.

The analysis numbers I=7 and 8 refer to two long records, each containing 54 segments. For a more detailed analysis, the evaluation was at first made at three blocks and then over the whole record length. Block 1 encompasses the segments 1-22 (18 ACFs), block 2 the segments 19-40 (18 ACFs), and block 3 the segments 37-54 (14 ACFs). The auxiliary code ACFITEV4 (Section 12) allows plotting the estimated values of the oscillation frequency and the DR obtained from the individual ACFs at the optimum fit range for the whole record length (see Fig. 12.1 in Section 12). The signal C2_TEST.L1 shows a transient. The DR is time-dependent. The signal C2_TEST.L2 is practically stationary.

For comparison, the results from 15 other methods are summarized in Table 11.2. The first line of each analysis case refers to the value of the oscillation frequency. The second line contains the values of the DR. The data were taken from the benchmark proceedings which is referenced in Section 1. Unfortunately, no uncertainties have been given there.

The fit procedure with the option IBGR=0 gives mostly a significant underestimation of the DR combined with a down-shift of the estimated oscillation frequency. This behaviour demonstrates the importance of taking the PSD background into account. The fit procedure with the option IBGR=2 shows in many cases results which range well within the data obtained by the other methods. However, for high DR cases, one can observe the tendency of a DR overestimation. If one extends the given upper limit of the fit range up to about r=70 %, then normally slightly reduced DR values are obtained which are in better agreement with the data in Table 11.2.

**Table 11.1:** Results for the Oscillation Frequency and the Decay Ratio.

| I | Record | Filter Bandwidth (Hz) | IBGR | Number of ACFs g | Number of ACFs a | Fit Range r (%) | Fit Range tau-end (s) | Frequency f (Hz) | Frequency s | Decay Ratio DR | Decay Ratio s |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | C1_APRM.1 | 0.2441-0.7813 | 0 | 11 | 11 | 15.0 | 2.96 | 0.397 | 0.014 | 0.355 | 0.051 |
| | | | 2 | 11 | 11 | 15.0 | 2.96 | 0.438 | 0.009 | 0.598 | 0.055 |
| 2 | C1_APRM.3 | 0.2930-0.7813 | 0 | 11 | 11 | 15.0 | 2.96 | 0.429 | 0.018 | 0.323 | 0.094 |
| | | | 2 | 11 | 11 | 27.5 | 5.52 | 0.479 | 0.020 | 0.603 | 0.094 |
| 3 | C1_APRM.4 | 0.3418-0.6348 | 0 | 11 | 11 | 15.0 | 2.96 | 0.457 | 0.014 | 0.409 | 0.077 |
| | | | 2 | 11 | 11 | 22.5 | 4.48 | 0.486 | 0.012 | 0.698 | 0.170 |
| 4 | C1_APRM.12 | 0.3418-0.6348 | 0 | 11 | 11 | 17.5 | 3.44 | 0.438 | 0.016 | 0.645 | 0.107 |
| | | | 2 | 11 | 11 | 40.0 | 8.08 | 0.466 | 0.004 | 0.822 | 0.039 |
| 5 | C2_TEST.S11 | 0.1465-0.6836 | 0 | 11 | 11 | 15.0 | 2.96 | 0.309 | 0.023 | 0.101 | 0.072 |
| | | | 2 | 11 | 11 | 15.0 | 2.96 | 0.382 | 0.031 | 0.308 | 0.071 |
| 6 | C2_TEST.S31 | 0.2930-0.6836 | 0 | 11 | 11 | 17.5 | 3.44 | 0.385 | 0.016 | 0.277 | 0.080 |
| | | | 2 | 11 | 11 | 27.5 | 5.52 | 0.433 | 0.016 | 0.490 | 0.061 |
| 7 | C2_TEST.L1 | 0.2441-0.7813 | | | | | | | | | |
| | Block 1 | | 0 | 18 | 18 | 35.0 | 7.04 | 0.346 | 0.037 | 0.189 | 0.104 |
| | | | 2 | 18 | 13 | 25.0 | 5.04 | 0.408 | 0.027 | 0.384 | 0.084 |
| | Block 2 | | 0 | 18 | 18 | 15.0 | 2.96 | 0.374 | 0.022 | 0.286 | 0.086 |
| | | | 2 | 18 | 18 | 17.5 | 3.44 | 0.422 | 0.016 | 0.481 | 0.039 |
| | Block 3 | | 0 | 14 | 14 | 20.0 | 4.00 | 0.364 | 0.022 | 0.237 | 0.039 |
| | | | 2 | 14 | 14 | 15.0 | 2.96 | 0.414 | 0.011 | 0.458 | 0.061 |
| | Total | | 0 | 50 | 50 | 20.0 | 4.00 | 0.366 | 0.024 | 0.254 | 0.086 |
| | | | 2 | 50 | 49 | 17.5 | 3.44 | 0.411 | 0.022 | 0.443 | 0.086 |
| 8 | C2_TEST.L2 | 0.2441-0.7813 | | | | | | | | | |
| | Block 1 | | 0 | 18 | 18 | 17.5 | 3.44 | 0.483 | 0.015 | 0.397 | 0.076 |
| | | | 2 | 18 | 12 | 22.5 | 4.48 | 0.509 | 0.012 | 0.621 | 0.072 |
| | Block 2 | | 0 | 18 | 18 | 15.0 | 2.96 | 0.457 | 0.023 | 0.324 | 0.106 |
| | | | 2 | 18 | 18 | 15.0 | 2.96 | 0.502 | 0.007 | 0.636 | 0.065 |
| | Block 3 | | 0 | 14 | 14 | 27.5 | 5.52 | 0.479 | 0.007 | 0.413 | 0.078 |
| | | | 2 | 14 | 10 | 22.5 | 4.48 | 0.494 | 0.009 | 0.565 | 0.027 |
| | Total | | 0 | 50 | 50 | 15.0 | 2.96 | 0.464 | 0.020 | 0.348 | 0.092 |
| | | | 2 | 50 | 50 | 20.0 | 4.00 | 0.504 | 0.012 | 0.629 | 0.073 |
| 9 | C4_LPRM.8 | 0.3418-0.7324 | 0 | 12 | 12 | 32.5 | 6.56 | 0.502 | 0.005 | 0.716 | 0.035 |
| | | | 2 | 12 | 8 | 15.0 | 2.96 | 0.502 | 0.005 | 0.747 | 0.021 |
| 10 | C4_LPRM.22 | 0.3418-0.7324 | 0 | 12 | 12 | 20.0 | 4.00 | 0.502 | 0.014 | 0.340 | 0.043 |
| | | | 2 | 12 | 12 | 35.0 | 7.04 | 0.517 | 0.015 | 0.609 | 0.066 |
| 11 | C6_LPRM.111 | 0.2441-0.8789 | 0 | 12 | 12 | 15.0 | 2.96 | 0.357 | 0.040 | 0.067 | 0.044 |
| | | | 2 | 12 | 10 | 15.0 | 2.96 | 0.421 | 0.023 | 0.180 | 0.075 |
| 12 | C6_LPRM.112 | 0.2441-0.8301 | 0 | 12 | 9 | 15.0 | 2.96 | 0.262 | 0.099 | 0.020 | 0.014 |
| | | | 2 | 12 | 12 | 15.0 | 2.96 | 0.459 | 0.019 | 0.270 | 0.063 |
| 13 | C6_LPRM.22 | 0.2441-0.7813 | 0 | 12 | 12 | 15.0 | 2.96 | 0.383 | 0.032 | 0.057 | 0.038 |
| | | | 2 | 12 | 12 | 15.0 | 2.96 | 0.461 | 0.025 | 0.226 | 0.078 |
| 14 | C6_LPRM.210 | 0.2441-0.7813 | 0 | 12 | 12 | 15.0 | 2.96 | 0.416 | 0.016 | 0.203 | 0.048 |
| | | | 2 | 12 | 12 | 15.0 | 2.96 | 0.476 | 0.008 | 0.512 | 0.087 |
| 15 | C6_LPRM.211 | 0.4395-0.6836 | 0 | 12 | 12 | 50.0 | 10.16 | 0.523 | 0.001 | 0.947 | 0.016 |
| 16 | C6_LPRM.213 | 0.3448-0.7324 | 0 | 12 | 12 | 50.0 | 10.16 | 0.523 | 0.001 | 0.960 | 0.013 |
| | | | 2 | 12 | 12 | 50.0 | 10.16 | 0.523 | 0.001 | 1.012 | 0.012 |
| 17 | C6_LPRM.214 | 0.4395-0.6836 | 0 | 12 | 12 | 50.0 | 10.16 | 0.524 | 0.001 | 0.954 | 0.012 |
| | | | 2 | 12 | 12 | 50.0 | 10.16 | 0.525 | 0.001 | 1.005 | 0.026 |
| 18 | C6_LPRM.215 | 0.3448-0.7324 | 0 | 12 | 12 | 50.0 | 10.16 | 0.523 | 0.001 | 0.981 | 0.006 |
| | | | 2 | 12 | 12 | 50.0 | 10.16 | 0.523 | 0.001 | 0.999 | 0.007 |
| 19 | C6_LPRM.218 | 0.3906-0.6836 | 0 | 12 | 12 | 32.5 | 6.56 | 0.508 | 0.010 | 0.549 | 0.094 |
| | | | 2 | 12 | 12 | 32.5 | 6.56 | 0.517 | 0.009 | 0.745 | 0.200 |

## Table 11.2: Benchmark Results

First line : Oscillation Frequency (Hz)

Second line: Decay Ratio

| I | Record | M1 | M2 | M3 | M4 | M5 | M6 | M7 | M8 | M9 | M10 | M11 | M12 | M13 | M14 | M15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | C1_APRM.1 | 0.483 | 0.452 | 0.487 | 0.467 | 0.45 | 0.350 | 0.450 | 0.460 | 0.460 | 0.464 | 0.459 | 0.459 | 0.448 | 0.47 | 0.458 |
| | | 0.460 | 0.423 | 0.576 | 0.640 | 0.42 | 0.580 | 0.330 | 0.500 | 0.420 | 0.422 | 0.460 | 0.420 | 0.512 | 0.57 | 0.566 |
| 2 | C1_APRM.3 | 0.483 | 0.482 | 0.481 | 0.480 | 0.46 | 0.270 | 0.450 | 0.480 | 0.490 | 0.497 | 0.483 | 0.483 | 0.482 | 0.51 | 0.476 |
| | | 0.576 | 0.582 | 0.558 | 0.735 | 0.30 | 0.250 | 0.300 | 0.500 | 0.630 | 0.511 | 0.537 | 0.520 | 0.499 | 0.60 | 0.516 |
| 3 | C1_APRM.4 | 0.489 | 0.490 | 0.487 | 0.481 | 0.48 | 0.280 | 0.470 | 0.490 | 0.460 | 0.480 | 0.490 | 0.490 | 0.518 | 0.51 | 0.490 |
| | | 0.515 | 0.514 | 0.525 | 0.634 | 0.39 | 0.260 | 0.230 | 0.530 | 0.420 | 0.549 | 0.528 | 0.510 | 0.558 | 0.78 | 0.516 |
| 4 | C1_APRM.12 | 0.466 | 0.466 | 0.467 | 0.400 | 0.45 | 0.300 | 0.450 | 0.460 | 0.460 | 0.467 | 0.465 | 0.465 | 0.459 | 0.47 | 0.452 |
| | | 0.812 | 0.809 | 0.828 | 0.751 | 0.68 | 0.430 | 0.560 | 0.780 | 0.780 | 0.757 | 0.792 | 0.780 | 0.740 | 0.66 | 0.559 |
| 5 | C2_TEST.S11 | 0.442 | 0.440 | 0.435 | 0.471 | 0.410 | | 0.490 | 0.440 | 0.420 | 0.361 | 0.424 | 0.424 | 0.478 | 0.45 | 0.444 |
| | | 0.287 | 0.268 | 0.312 | 0.355 | 0.360 | | 0.100 | 0.200 | 0.200 | 0.113 | 0.168 | 0.150 | 0.416 | 0.34 | 0.580 |
| 6 | C2_TEST.S31 | 0.443 | 0.440 | 0.437 | 0.427 | 0.430 | | 0.410 | 0.460 | 0.480 | 0.482 | 0.453 | 0.453 | 0.478 | 0.45 | 0.441 |
| | | 0.338 | 0.384 | 0.475 | 0.646 | 0.360 | | 0.190 | 0.400 | 0.470 | 0.323 | 0.359 | 0.270 | 0.416 | 0.27 | 0.243 |
| 7 | C2_TEST.L1 | 0.454 | 0.453 | 0.453 | 0.472 | 0.440 | | 0.430 | 0.450 | 0.460 | 0.457 | 0.444 | 0.444 | 0.441 | 0.45 | 0.442 |
| | | 0.395 | 0.394 | 0.469 | 0.432 | 0.350 | | 0.160 | 0.350 | 0.550 | 0.339 | 0.360 | 0.270 | 0.386 | 0.23 | 0.393 |
| 8 | C2_TEST.L2 | 0.533 | 0.533 | 0.519 | 0.534 | 0.500 | | 0.520 | 0.530 | 0.510 | 0.537 | 0.516 | 0.516 | 0.516 | 0.54 | 0.516 |
| | | 0.640 | 0.640 | 0.634 | 0.620 | 0.570 | | 0.340 | 0.630 | 0.600 | 0.622 | 0.576 | 0.570 | 0.576 | 0.54 | 0.534 |
| 9 | C4_LPRM.8 | 0.509 | 0.510 | 0.507 | 0.478 | 0.508 | 0.400 | 0.495 | 0.500 | 0.510 | 0.504 | 0.507 | 0.507 | 0.507 | | 0.514 |
| | | 0.703 | 0.705 | 0.694 | 0.733 | 0.688 | 0.740 | 0.620 | 0.760 | 0.710 | 0.729 | 0.760 | 0.700 | 0.733 | | 0.403 |
| 10 | C4_LPRM.22 | 0.530 | 0.531 | 0.530 | 0.557 | 0.540 | 0.450 | 0.495 | 0.530 | 0.550 | 0.512 | 0.526 | 0.526 | 0.523 | | 0.548 |
| | | 0.583 | 0.569 | 0.569 | 0.702 | 0.360 | 0.860 | 0.380 | 0.390 | 0.360 | 0.422 | 0.517 | 0.490 | 0.447 | | 0.582 |
| 11 | C6_LPRM.111 | 0.505 | 0.506 | 0.506 | 0.457 | 0.463 | 0.280 | 0.510 | 0.53 | | 0.558 | 0.504 | 0.504 | 0.529 | | 0.529 |
| | | 0.392 | 0.391 | 0.382 | 0.461 | 0.105 | 0.280 | 0.110 | 0.23 | | 0.277 | 0.361 | 0.400 | 0.344 | | 0.907 |
| 12 | C6_LPRM.112 | 0.541 | 0.542 | 0.545 | 0.492 | | 0.250 | 0.520 | 0.53 | | 0.520 | 0.535 | 0.535 | 0.517 | | 0.488 |
| | | 0.413 | 0.413 | 0.439 | 0.362 | | 0.150 | 0.230 | 0.20 | | 0.545 | 0.333 | 0.320 | 0.292 | | 0.071 |
| 13 | C6_LPRM.22 | 0.519 | 0.520 | 0.504 | 0.528 | 0.513 | 0.390 | 0.510 | 0.510 | 0.530 | 0.471 | 0.517 | 0.517 | 0.513 | | 0.519 |
| | | 0.384 | 0.390 | 0.397 | 0.589 | 0.233 | 0.710 | 0.200 | 0.250 | 0.250 | 0.332 | 0.293 | 0.400 | 0.357 | | 0.391 |
| 14 | C6_LPRM.210 | 0.511 | 0.512 | 0.513 | 0.506 | 0.496 | 0.380 | 0.500 | 0.500 | 0.510 | 0.516 | 0.501 | 0.501 | 0.485 | | 0.494 |
| | | 0.593 | 0.594 | 0.601 | 0.672 | 0.302 | 0.680 | 0.320 | 0.510 | 0.570 | 0.709 | 0.513 | 0.160 | 0.535 | | 0.473 |
| 15 | C6_LPRM.211 | 0.521 | 0.521 | 0.521 | 0.518 | 0.525 | 0.310 | 0.510 | 0.520 | 0.520 | 0.523 | 0.521 | 0.521 | 0.519 | | 0.535 |
| | | 0.889 | 0.889 | 0.890 | 0.870 | 0.611 | 0.450 | 0.500 | 0.980 | 0.950 | 0.966 | 0.858 | 0.950 | 0.768 | | 0.456 |
| 16 | C6_LPRM.213 | 0.521 | 0.522 | 0.522 | 0.519 | 0.525 | 0.420 | 0.510 | 0.520 | 0.520 | 0.524 | 0.521 | 0.521 | 0.520 | | 0.530 |
| | | 0.897 | 0.898 | 0.906 | 0.876 | 0.720 | 0.760 | 0.530 | 0.950 | 0.940 | 0.935 | 0.878 | 0.950 | 0.836 | | 0.501 |
| 17 | C6_LPRM.214 | 0.522 | 0.522 | 0.521 | 0.522 | 0.511 | 0.480 | 0.510 | 0.520 | 0.520 | 0.523 | 0.521 | 0.521 | 0.521 | | 0.523 |
| | | 0.894 | 0.896 | 0.895 | 0.919 | 0.766 | 0.870 | 0.680 | 0.820 | 0.930 | 0.965 | 0.877 | 0.950 | 0.832 | | 0.425 |
| 18 | C6_LPRM.215 | 0.521 | 0.521 | 0.522 | 0.520 | 0.521 | 0.480 | 0.510 | 0.520 | | 0.524 | 0.521 | 0.521 | 0.521 | | 0.525 |
| | | 0.963 | 0.973 | 0.964 | 0.966 | 0.877 | 0.870 | 0.780 | 0.950 | | 0.988 | 0.954 | 0.990 | 0.922 | | 0.495 |
| 19 | C6_LPRM.218 | 0.513 | 0.514 | 0.514 | 0.511 | 0.510 | 0.250 | 0.500 | 0.510 | | 0.498 | 0.507 | 0.507 | 0.454 | | 0.504 |
| | | 0.547 | 0.549 | 0.550 | 0.700 | 0.330 | 0.190 | 0.420 | 0.510 | | 0.507 | 0.503 | 0.470 | 0.518 | | 0.590 |

M1: UPV Standard AR,
M2: UPV Full SVD AR,
M3: UPV Truncated SVD,
M4: UPV Dynamics Reconstruction,
M5: Pennsylvania State University,
M6: Pennsylvania State University : LAPUR Code,
M7: University of Tsukuba,
M8: PSI : ARMA Model (Plateau Method),
M9: PSI : AR-AIC,
M10: JAERI,
M11: SIEMENS AR,
M12: SIEMENS RAC,
M13: TOSHIBA,
M14: TU Delft,
M15: CSNNS Mexico

# LISTING

## A C F I T 7 S A

```
      PROGRAM ACFIT7SA
C
C     ACFIT7SA GOVERNS THE REPETETIVE APPLICATION OF ACFIT7
C     FOR SEGMENT ANALYSIS.
C
C     WRITTEN BY K.BEHRINGER, APRIL 2000.
C
C     COMPILER : COMPAQ FORTRAN
C     LINK COMMAND : IMPORT IMSL
C                    LINK ACFIT7SA,IMSLIBG_STATIC/OPT,IMSLPSECT/OPT
C
      PARAMETER (NSEGMAX=99)
      CHARACTER*50 RUN,FILE10,FILE11
      DATA FILE10/'ACFIT7SA.IN                                       '/
      DATA FILE11/'ACFIT7SA.PRT                                      '/
10000 FORMAT(A/I4)
11000 FORMAT('1',40X,'P R O G R A M     A C F I T 7 S A'///1X,
     1'FILES : PARAMETER INPUT FILE10 = ',A/9X,
     2'PRINT OUTPUT',4X,'FILE11 = ',A//)
11001 FORMAT(1X,'RUN DENOTATION',26X,'RUN',5X,'=',1X,A//1X,
     1'GIVEN NUMBER OF SEGMENTS',16X,'NSEG',4X,'=',I5)
11002 FORMAT(////11X,'OPENING ERROR FILE10 !')
11003 FORMAT(////11X,'READ-ERROR FILE10 !')
11004 FORMAT(////11X,'E N D')
      OPEN(UNIT=11,FILE=FILE11,STATUS='NEW',DEFAULTFILE='DIRINPUT')
      WRITE(11,11000) FILE10,FILE11
      OPEN(UNIT=10,FILE=FILE10,STATUS='OLD',DEFAULTFILE='DIRINPUT',
     1ERR=100)
      READ(10,10000,ERR=101,END=101) RUN,NSEG
      CLOSE(UNIT=10,STATUS='KEEP')
      IF(NSEG.LT.1) NSEG=1
      IF(NSEG.GT.NSEGMAX) NSEG=NSEGMAX
      WRITE(11,11001) RUN,NSEG
      DO 1 ISEG=1,NSEG
    1 CALL ACFIT7(ISEG,NSEG)
      WRITE(11,11004)
      STOP
  100 WRITE(11,11002)
      STOP
  101 WRITE(11,11003)
      STOP
      END
C=======================================================
      SUBROUTINE INDEX(FINPUT,FFORM,ISEG)
C
C     INDEX INCREASES THE CURRENT NUMBERS OF FINPUT AND FFORM BY 1 .
C
C     THE FOLLOWING RESTRICTIONS ARE ASSUMED :
C     1) THE ACF DATA ARE OBTAINED FROM THE CODE ACCF1. THE FILE
C        NUMBERS OF FINPUT ARE THEREFORE RESTRICTED TO THE VALUES
C        1 - 999. ALL THE FILES ARE AVAILABLE IN THE SAME SUBDIRECTORY.
C     2) THE FIRST OF THE THREE CHARACTERS OF FFORM CAN BE A LETTER
C        OR A NUMBER. IT IS LEFT UNCHANGED. THE TWO FURTHER CHARACTERS
```

```
C        ARE CONSIDERED AS A CURRENT NUMBER WITH THE RESTRICTION
C        TO THE VALUE 1 - 99 .
C
      CHARACTER*50 FINPUT
      CHARACTER*3 FFORM
      SAVE IFF,IFIN
11000 FORMAT(////11X,'CURRENT NUMBER OF FINPUT EXCEEDS MAXIMUM VALUE !')
11001 FORMAT(////11X,'CURRENT NUMBER OF FFORM EXCEEDS MAXIMUM VALUE !')
      IF(ISEG.GT.1) GO TO 100
      DECODE(2,'(I2)',FFORM(2:3)) IFF
      DECODE(3,'(I3)',FINPUT(6:8)) IFIN
  100 IFF1 = IFF+ISEG
      IF(IFF1.GE.100) THEN
      WRITE(11,11001)
      STOP
      ELSE
      ENCODE(2,'(I2.2)',FFORM(2:3)) IFF1
      END IF
      IFIN1 = IFIN+ISEG
      IF(IFIN1.GE.1000) THEN
      WRITE(11,11000)
      STOP
      ELSE
      ENCODE(3,'(I3.3)',FINPUT(6:8)) IFIN1
      END IF
      RETURN
      END
C=========================================================
      SUBROUTINE ACFIT7(ISEG,NSEG)
C
C     THE CODE FITS FREQUENCY-FILTERED AUTO-CORRELATION FUNCTION (ACF)
C     DATA TO A SECOND-ORDER OSCILLATOR MODEL FOR BWR
C     BOILING INSTABILITY ANALYSIS. THERE ARE THE OPTIONS FOR A
C     ZERO-PARAMETRIC, A ONE-PARAMETRIC OR A TWO-PARAMETRIC BACKGROUND F
C     TAU-RANGE CAN BE SELECTED AND SUCCESSIVELY INCREASED WITHIN 10 AND
C     OF THE MAXIMUM TAU-RANGE. THE CODE ASSUMES THE FORMAT OF THE
C     ACF DATA OUTPUT FROM THE CODE ACCF1.
C
C     FOR THE LEAST-SQUARES FIT THE ROUTINE MARFIT IS USED, WHICH REQUIR
C     THE CALCULATION OF THE DERIVATIVES OF THE FIT FUNCTION.
C     THERE ARE OPTIONS FOR SELECTING WEIGHTING FUNCTIONS.
C     THE CORRECTED ACF IS OBTAINED FROM THE IDEAL ACF.
C     THE NUMERICAL INTEGRATION PROCEDURE (BY THE IMSL ROUTINES DQDAWO
C     AND DQDAWF) IS DIVIDED INTO INTEGRATION SUBINTERVALS.
C     THE PARAMETERS TO BE DETERMINED BY THE FIT ARE :
C         THETA(1) = C0 (AMPLITUDE OF THE ACF)
C         THETA(2) = OMEGAC
C         THETA(3) = FLAMDA
C         THETA(4) = B0    (OPTION)
C         THETA(5) = ALPHA (OPTION)
C
C     CODE WRITTEN BY K.BEHRINGER, OCTOBER 1999, FOR BATCH OPERATION
C     ON THE PSI DEC-ALPHA 2100 COMPUTER.
C
C
C     PARAMETERS :
C
C          NPMAX   = MAXIMUM NUMBER OF THE RIGHT-HAND SIDE ACF DATA.
```

```
C                 A CHANGEMENT REQUIRES ADAPTATION IN MANY ROUTINES.
C         NPARMAX = MAXIMUM NUMBER OF FIT PARAMETERS. (=5)
C         NRMAX   = MAXIMUM NUMBER OF TAU-RANGES.
C         IWMAX   = MAXIMUM NUMBER OF AVAILABLE WEIGHTING FUNCTIONS.
C
C     NOTE : IF THE NAMES OF THE DATA OUTPUT FILES (FILE4/5/6/7) ARE CHA
C            A CORRESPONDING CHANGEMENT OF THE LATER ENCODE STATEMENTS
C            IS REQUIRED.
C
C
C     DECLARATIONS
C
      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER (NPMAX=1025,NPARMAX=5,NRMAX=20,IWMAX=6)
      CHARACTER*50 RUN,FILE1,FILE2,FILE3,FINPUT,FILE4,FILE5,FILE6,
     1FILE7
      CHARACTER*20 IFORM,IFORM1,IFORM2
      CHARACTER*3 FFORM
      REAL*4 SC
      DIMENSION XDATA(NPMAX),YDATA(NPMAX),THETA(NPARMAX),
     1RANGE(NRMAX)
      COMMON/XYDATA/XDATA,YDATA
      COMMON/CONTROL/NPEND,IBGR
      COMMON/PAR/OMEGAL,OMEGAH,PI
      COMMON/W/WA,NWE,NPA1,IWEIGHT
      EQUIVALENCE (FILE3,FINPUT)
      DATA NF,NPLIMIT,PI,SC/2,20,3.141592654D0,1.E30/,
     1IFORM1/'(5X,2E12.4)'/,
     2IFORM2/'(5X,E12.4,12X,E12.4)'/,
     3FILE1/'ACFIT7.IN                                         '/,
     4FILE2/'ACFIT7_00.PRT                                     '/,
     5FILE3/'<FINPUT>                                          '/,
     6FILE4/'ACFIT7_000_000.DAT                                '/,
     7FILE5/'ACFIT7_000.PAR                                    '/,
     8FILE6/'ACFIT7_000_000.FIT                                '/,
     9FILE7/'ACFIT7_000.PLT                                    '/
      SAVE
C
C     FORMATS
C
 1000 FORMAT(4(A/),2(D12.5/),I1/2(D12.5/),I1/D12.5/I2/I1/I3/I2)
 1001 FORMAT(2(D12.5/),I3/I1)
 1005 FORMAT(F6.2)
 2000 FORMAT(1H1,40X,'P R O G R A M     A C F I T 7'///1X,
     1'FILES : PARAMETER DATA : FILE1 = ',A/9X,
     2'PRINT OUTPUT   : FILE2 = ',A/9X,
     3'ACF DATA',7X,': FILE3 = ',A/9X,
     4'DATA OUTPUT',4X,': FILE4 = ',A/9X,
     5'FIT PARAMETER',2X,': FILE5 = ',A/9X,
     6'FIT DATA',7X,': FILE6 = ',A/9X,
     7'PARAMETER PLOT',1X,': FILE7 = ',A///)
 2005 FORMAT(1X,'RUN DENOTATION',36X,'RUN',5X,'=',1X,A//1X,
     1'ACF DATA FILE',37X,'FINPUT',2X,'=',1X,A//1X,
     2'ACF DATA FORMAT',35X,'IFORM',3X,'=',1X,A//1X,
     3'ADDITIONAL DENOTATION OF OUTPUT FILES',13X,'FFORM',3X,
     4'=',1X,A//1X,
     5'LOWER CUTOFF FREQUENCY (HZ)',23X,'FRL',5X,'=',1PD14.5/1X,
```

```
     6'UPPER CUTOFF FREQUENCY (HZ)',23X,'FRH',5X,'=',D14.5//1X,
     7'BACKGROUND FIT PARAMETER (0:B0=0,1:B0,2:B0,ALPHA)',1X,
     8'IBGR',4X,'=',I4/1X,
     9'INITIAL BACKGROUND AMPLITUDE (0/0 OF YDATA(1))',4X,
     A'B0FIN',3X,'=',D14.5/1X,
     B'INITIAL BACKGROUND SLOPE INDEX',20X,'SLOPE',3X,'=',D14.5//1X,
     C'PARAMETER FOR WEIGHTING (1:U, 2:L.DCR., 3:COS)',4X,
     D'IWEIGHT =',I4/6X,'(MODIFIED INITIAL PART : 4:1,5:2,6:3)'/1X,
     E'INITIAL WEIGHT AMPLITUDE FACTOR',19X,'WA',6X,'=',D14.5/1X,
     F'END POINT OF WEIGHT FUNCTION INCREASE'13X,'NWE',5X,'=',I4//1X,
     G'WEIGHT ENDPOINT (1:NPEND, 2:NPA)',18X,'IWEND',3X,'=',I4//1X,
     H'INITIAL NUMBER OF DATA OUTPUT FILE',16X,'IDAT',4X,'=',I4//1X,
     I'REQUESTED NUMBER OF JOBS',26X,'JOBMAX',2X,'=',I4/)
 2010 FORMAT('1ACCEPTED  NUMBER OF JOBS',26X,'JOBMAX',2X,'=',I5//1X,
     1'RANGE VALUES (0/0)',44X,'N',3X,'RANGE'/)
 2015 FORMAT(61X,I3,2X,F6.2)
 2016 FORMAT(//1X,'DATA FOR CONVERGENCE (SUBROUTINE FITMAR) :'//6X,
     1'INITIAL VALUE FOR XLAMDA (1.D-8 - 1.D2)',6X,'XLIN',4X,'=',
     21PD15.5//6X,
     3'CONVERGENCE PARAMETER (0/0, 1.D-4 - 1.D0)',4X,'EPSIL',3X,'=',
     4D15.5//6X,
     5'ALLOWED NUMBER OF ITERATIONS (50 - 999)',6X,'ITMAX',3X,'=',
     6I5//6X,
     7'DATA OUTPUT DURING FIT (0:NO, 1:YES)',9X,'IOUT',4X,'=',I5/)
 2020 FORMAT(////11X,'OPENING ERROR ! FILE = ',I1)
 2025 FORMAT(////11X,'READ-ERROR IN PARAMETER DATA !')
 2030 FORMAT(////11X,'ERRANEOUS PARAMETER DATA !')
 2035 FORMAT(1X,'AVAILABLE NUMBER OF ACF DATA',22X,'NPA',5X,'=',I5//1X,
     1'MAXIMUM TAU-RANGE (SEC)',27X,'TAUMAX',2X,'=',1PD15.5)
 2040 FORMAT('1JOB NUMBER',40X,'JOB',5X,'=',I4//1X,
     1'NUMBER OF DATA OUTPUT FILE',24X,'IDAT',4X,'=',I4//1X,
     2'RANGE VALUE (0/0)',33X,'RANGE',3X,'=',F7.2//1X,
     3'LAST DATA POINT NUMBER',28X,'NPEND',3X,'=',I5//1X,
     4'TAU-RANGE (SEC)',35X,'TAUEND',2X,'=',1PD14.5/)
 2045 FORMAT(1X,'ESTIMATED FIT PARAMETERS (RELATIVE VALUES)',8X,1P,
     1'THETA',3X,'=',5D14.5/1X,
     2'REDUCED CHI-SQUARE OF THE FIT',21X,'CHISQR',2X,'=',D14.5/1X,
     3'NUMBER OF ITERATIONS',30X,'IT',6X,'=',I4/)
 2050 FORMAT(/1X,'ESTIMATED FIT PARAMETERS (ABSOLUTE VALUES)',//11X,
     1'PSD FUNCTION AMPLITUDE',18X,'A0',6X,'=',1PD14.5/11X,
     2'BACKGROUND AMPLITUDE',20X,'B0',6X,'=',D14.5/11X,
     3'BACKGROUND DECAY CONSTANT (SEC/RAD)',5X,'ALPHA',3X,'=',D14.5//
     411X,'PSD RESONANCE FREQUENCY (HZ)',12X,'FR0',5X,'=',D14.5/11X,
     5'ACF FREQUENCY (HZ)',22X,'FR0CF',3X,'=',D14.5//11X,
     6'RESONANCE QUALITY FACTOR',16X,'QR',6X,'=',D14.5/11X,
     7'DECAY CONSTANT (1/SEC)',18X,'FLAMDA',2X,'=',D14.5/11X,
     8'DECAY RATIO',29X,'DR',6X,'=',D14.5//11X,
     9'PEAK AMPLITUDE IN THE PSD',15X,'PEAKMAX =',D14.5/11X,
     A'PEAK BACKGROUND AMPLITUDE',15X,'B0PEAK',2X,'=',D14.5)
 2055 FORMAT(////11X,'NPEND TOO SMALL !')
 2060 FORMAT(////11X,'E N D')
 4000 FORMAT(4X,I4,1P,4E14.5)
 5000 FORMAT(I4,2X,1P,9E14.5/6X,E14.5)
C
C     INPUT PARAMETER DATA
C
      ENCODE(2,'(I2.2)',FILE2(8:9)) ISEG
      OPEN(UNIT=2,FILE=FILE2,STATUS='NEW',DEFAULTFILE='DIRINPUT',
```

```
     1ERR=300)
      WRITE(2,2000) FILE1,FILE2,FILE3,FILE4,FILE5,FILE6,FILE7
      IF(ISEG.GT.1) GO TO 200
      NF = 1
      OPEN(UNIT=1,FILE=FILE1,STATUS='OLD',DEFAULTFILE='DIRINPUT',
     1ERR=100)
      READ(1,1000,ERR=105,END=105) RUN,FINPUT,IFORM,FFORM,FRL,FRH,
     1IBGR,B0FIN,SLOPE,IWEIGHT,WA,NWE,IWEND,IDAT,JOBMAX
      IF(IFORM.EQ.'*') IFORM=IFORM1
      IF(IFORM.EQ.'**') IFORM=IFORM2
      IF(FFORM(3:3).EQ.' ') THEN
      FFORM(3:3) = FFORM(2:2)
      FFORM(2:2) = '0'
      END IF
      IF(FRL.LT.0.D0) FRL=0.D0
      IF(IBGR.LT.0) IBGR=0
      IF(IBGR.GT.2) IBGR=2
      IF(B0FIN.LT.0.D0) B0FIN=0.D0
      IF(B0FIN.GT.1.D3) B0FIN=1.D3
      IF(IBGR.EQ.0) B0FIN=0.D0
      IF(IBGR.EQ.0) SLOPE=0.D0
      IF(IWEIGHT.LT.1) IWEIGHT=1
      IF(IWEIGHT.GT.IWMAX) IWEIGHT=IWMAX
      IF(IWEIGHT.LT.4.OR.NWE.LE.0.OR.WA.LT.0.D0) THEN
      NWE = 0
      WA = 0.D0
      ELSE
      IF(NWE.EQ.1) NWE=2
      IF(NWE.GT.NPLIMIT) NWE=NPLIMIT
      IF(WA.GT.1.D0) WA=1.D0
      END IF
      IF(IWEND.LT.1.OR.IWEIGHT.EQ.1) IWEND=1
      IF(IWEND.GT.2) IWEND=2
      IF(IDAT.LT.1) IDAT=1
      IF(IDAT.GT.1000-NRMAX) IDAT=1000-NRMAX
      IDATS = IDAT
      IF(JOBMAX.LT.1) JOBMAX=1
      IF(JOBMAX.GT.NRMAX) JOBMAX=NRMAX
  200 WRITE(2,2005) RUN,FINPUT,IFORM,FFORM,FRL,FRH,IBGR,
     1B0FIN,SLOPE,IWEIGHT,WA,NWE,IWEND,IDAT,JOBMAX
      IF(ISEG.GT.1) GO TO 205
      IF(FRH.LT.1.1D0*FRL) GO TO 120
      READ(1,1001,ERR=105,END=105) XLIN,EPSIL,ITMAX,IOUT
      IF(XLIN.LT.1.D-8) XLIN=1.D-8
      IF(XLIN.GT.1.D2) XLIN=1.D2
      IF(EPSIL.LT.1.D-4) EPSIL=1.D-4
      IF(EPSIL.GT.1.D0) EPSIL=1.D0
      IF(ITMAX.LT.50) ITMAX=50
      IF(ITMAX.GT.999) ITMAX=999
      IF(IOUT.LT.0) IOUT=0
      IF(IOUT.GT.1) IOUT=1
      DO 1 N=1,JOBMAX
    1 READ(1,1005,ERR=105,END=110) RANGE(N)
      GO TO 115
  110 JOBMAX = N-1
  115 WRITE(2,2010) JOBMAX
      IF(JOBMAX.EQ.0) GO TO 120
```

```
      CALL SORT(RANGE,JOBMAX)
      DO 2 N=1,JOBMAX
    2 WRITE(2,2015) N,RANGE(N)
      DO 3 N=1,JOBMAX
      IF(RANGE(N).LT.1.D1.OR.RANGE(N).GT.9.999D1) GO TO 120
    3 RANGE(N) = 1.D-2*RANGE(N)
      WRITE(2,2016) XLIN,EPSIL,ITMAX,IOUT
      EPSIL = 1.D0+1.D-2*EPSIL
      B0FIN = 1.D-2*B0FIN
      IF(IBGR.EQ.2.AND.SLOPE.EQ.0.D0.OR.
     1DABS(SLOPE).GT.6.D0) GO TO 120
      CLOSE(UNIT=1,STATUS='KEEP')
  205 NF = 5
      ENCODE(3,'(A)',FILE5(8:10)) FFORM
      OPEN(UNIT=5,FILE=FILE5,STATUS='NEW',DEFAULTFILE='DIRINPUT',
     1ERR=100)
      NF = 3
      OPEN(UNIT=3,FILE=FINPUT,STATUS='OLD',DEFAULTFILE='DIRINPUT',
     1ERR=100)
      NF = 7
      ENCODE(3,'(A)',FILE7(8:10)) FFORM
      OPEN(UNIT=7,FILE=FILE7,STATUS='NEW',DEFAULTFILE='DIRINPUT',
     1ERR=100)
      ENCODE(3,'(A)',FILE4(8:10)) FFORM
      ENCODE(3,'(A)',FILE6(8:10)) FFORM
C
C     INPUT ACF DATA
C
      CALL INPUT(NPMAX,IFORM,XDATA,YDATA,NPA)
      IF(NPA.LT.NPLIMIT) STOP 105
      TAUMAX = XDATA(NPA)
      WRITE(2,2035) NPA,TAUMAX
      CLOSE(UNIT=3,STATUS='KEEP')
C
C     COMPUTATION AND DATA READ-OUT
C
      NF = 4
      DPI = 2.D0*PI
      OMEGAL = DPI*FRL
      OMEGAH = DPI*FRH
      NPAR = NPARMAX
      NPA1 = NPA-1
      IF(IBGR.EQ.1) NPAR=NPAR-1
      IF(IBGR.EQ.0) NPAR=NPAR-2
      CALL INIT(YDATA(1),B0FIN,SLOPE,THETA)
      C0S = THETA(1)
      DO 10 JOB=1,JOBMAX
      NPEND = RANGE(JOB)*NPA
      IF(IWEND.EQ.1) NPA1=NPEND-1
      TAUEND = XDATA(NPEND)
      WRITE(2,2040) JOB,IDAT,1.D2*RANGE(JOB),NPEND,TAUEND
      IF(NPEND.LT.NPLIMIT) GO TO 125
      IF(IOUT.EQ.0) GO TO 129
      NF = 6
      ENCODE(3,'(I3.3)',FILE6(12:14)) IDAT
      OPEN(UNIT=6,FILE=FILE6,STATUS='NEW',DEFAULTFILE='DIRINPUT',
     1ERR=100)
      NF = 4
```

```
  129 IREP1 = 0
      IF(THETA(1).EQ.COS) IREP1=1
  130 IREP2 = 0
      CALL FITMAR(NPAR,THETA,XLIN,EPSIL,ITMAX,IOUT,CHISQR,IT,IREP2,
     1*132)
      WRITE(2,2045) THETA,CHISQR,IT
      C0 = THETA(1)
      FR0CF = THETA(2)
      FLAMDA = THETA(3)
C     REPETITION OF THE FIT
      IF(C0.GT.0.D0.AND.FR0CF.GT.0.D0.AND.
     1FLAMDA.GT.-1.D-1.AND.IREP2.EQ.0) GO TO 131
      IF(IREP1.GE.1) GO TO 132
      CALL INIT(YDATA(1),B0FIN,SLOPE,THETA)
      IREP1 = 1
      GO TO 130
C*
  131 B0 = THETA(4)
      ALPHA = THETA(5)
      FLAMDASQ = FLAMDA**2
      OMEGA0SQ = FR0CF**2+FLAMDASQ
      FR0 = DSQRT(OMEGA0SQ)
      DR = DEXP(-DPI*FLAMDA/FR0CF)
      A1 = 4.D0*FLAMDA*C0
      A0 = A1/OMEGA0SQ
      QR = 0.D0
      IF(FLAMDA.GT.0.D0) QR=5.D-1*FR0/FLAMDA
      PEAKMAX = 0.D0
      IF(DR.LT.1.D0) PEAKMAX=A1*FINT(FR0)
      B0PEAK = 0.D0
      IF(PEAKMAX.GT.0.D0) B0PEAK=B0*DEXP(-ALPHA*(FR0-OMEGAL))
      FR0 = FR0/DPI
      FR0CF = FR0CF/DPI
      WRITE(2,2050) A0,B0,ALPHA,FR0,FR0CF,QR,FLAMDA,DR,PEAKMAX,
     1B0PEAK
      WRITE(5,5000) JOB,SNGL(TAUEND),SNGL(C0),SNGL(A0),SNGL(B0),
     1SNGL(ALPHA),SNGL(FR0CF),SNGL(FLAMDA),SNGL(DR),SNGL(PEAKMAX),
     2SNGL(CHISQR)
      WRITE(7,5000) JOB,SNGL(TAUEND),SNGL(C0),SNGL(A0),SNGL(B0),
     1SNGL(ALPHA),SNGL(FR0CF),SNGL(FLAMDA),SNGL(DR),SNGL(PEAKMAX),
     2SNGL(CHISQR)
      ENCODE(3,'(I3.3)',FILE4(12:14)) IDAT
      OPEN(UNIT=4,FILE=FILE4,STATUS='NEW',DEFAULTFILE='DIRINPUT',
     1ERR=100)
      DO 11 N=1,NPA
      IF(N.GT.NPEND) GO TO 135
      YFIT = FIT(THETA,N)
      DIF = YDATA(N)-YFIT
      GO TO 11
  135 YFIT = 0.D0
      DIF = 0.D0
   11 WRITE(4,4000) N,SNGL(XDATA(N)),SNGL(YDATA(N)),SNGL(YFIT),
     1SNGL(DIF)
      CLOSE(UNIT=4,STATUS='KEEP')
      IF(IOUT.EQ.1) CLOSE(UNIT=6,STATUS='KEEP')
C     RESETS
      IF(FLAMDA.LT.0.D0) THETA(3)=0.D0
```

```
      IF(DR.LT.2.D-1) CALL INIT(YDATA(1),B0FIN,SLOPE,THETA)
      IF(B0.LT.0.D0.OR.ALPHA.GT.1.D1)
     1CALL INIT1(YDATA(1),B0FIN,SLOPE,THETA)
C*
      GO TO 10
  132 WRITE(5,5000) JOB,SNGL(TAUEND),SC,SC,SC,SC,SC,SC,SC,SC,SC
      CALL INIT(YDATA(1),B0FIN,SLOPE,THETA)
      IF(IOUT.EQ.1) CLOSE(UNIT=6,STATUS='DELETE')
      GO TO 10
  125 WRITE(2,2055)
   10 IDAT = IDAT+1
      WRITE(2,2060)
      IF(ISEG.EQ.NSEG) RETURN
      CALL INDEX(FINPUT,FFORM,ISEG)
      CLOSE(UNIT=2,STATUS='KEEP')
      CLOSE(UNIT=5,STATUS='KEEP')
      CLOSE(UNIT=7,STATUS='KEEP')
      IDAT = IDATS
      NF = 2
      RETURN
  100 WRITE(2,2020) NF
      STOP
  105 WRITE(2,2025)
      STOP
  120 WRITE(2,2030)
      STOP
  300 WRITE(11,2020) NF
      WRITE(11,'(11X,''ISEG = '',I3)')
      STOP
      END
C=========================================================
      SUBROUTINE INPUT(NPMAX,IFORM,XDATA,YDATA,NPA)
C
C     DATA INPUT. IT SPECIFICALLY REFERS TO THE OUTPUT DATA
C     GIVEN BY THE CODE ACCF.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*4 XS,YS
      CHARACTER*20 IFORM
      DIMENSION XDATA(NPMAX),YDATA(NPMAX)
 2000 FORMAT(1X,'EOF AT ACF DATA POINT NUMBER',22X,'NEOF',4X,'=',I5/)
 2005 FORMAT(////11X,'READ=ERROR IN ACF DATA ! NERROR =',I5)
      DO 1 N=1,NPMAX
      READ(3,IFORM,ERR=100,END=105) XS,YS
      IF(XS.GE.0.) GO TO 120
    1 CONTINUE
  110 NPA = 0
      RETURN
  100 NERROR = N
  115 WRITE(2,2005) NERROR
      STOP 100
  105 NEOF = N
      WRITE(2,2000) NEOF
      GO TO 110
  120 XDATA(1) = DBLE(XS)
      YDATA(1) = DBLE(YS)
      NS = N-1
      ........
```

## 12. PROGRAM ACFITEV4

Precision : single

Operation : foreground, background

Required auxiliary routines : none

Purpose of the program :

Ordering of the Fit Parameter Data Obtained in the Gliding Segment Analysis for a Given Fit Range.

Feature Summary :

- Auxiliary code for ordering the fit parameter data on the files ACFIT7_'''.PAR from a run of the code ACFIT7SA for a given fit range.
- A time axis is introduced for plotting.
- Cancelled jobs in a subrun of ACFIT7SA are eliminated.

### 12.1 Comments

It is often desirable to plot fit parameter data and other derived data obtained in the subruns of ACFIT7SA (Section 11) for a given fit range, e.g. for the optimum fit range. One can then not only observe the often wild scattering of these data between the subruns, but also recognize non-stationary slow transient behaviour, if the signal record is sufficiently long in time. The code ACFITEV4 picks out a specified line of data in each of the matrices written on the files ACFIT7_'''.PAR and transfers it to a new file with an attributed time value. The code allows also to select only a part of successive subruns with an appropriate setting of the parameter values of NSEG and FFORM (see parameter data input list). The given time axis t starts always with zero value. The calculated time values correspond to the signal time points at the end of each subrun. If the 'short-time' auto-correlation functions (ACFs) have been estimated over $N_{av}$ consecutive signal segments, each having the time length $\tau_{max}$, the time values $t_i$ follow from $t_i = (N_{av} + i - 1)\tau_{max}$, i=1,2,... . If the code meets a cancelled job in a subrun, it will be left out which leads to a 'hole' in a graphical point plot.

## 12.2 Files

There are 4 files :

- ACFITEV4.IN (file unit 1)

This file contains the input parameter data. Most of them have to correspond to the conditions in a run of ACFIT7SA.

- ACFITEV4.PRT (file unit 2)

This file is destinated for printing and contains (with rext) all input parameter data and additionally messages of the computation progress.

- ACFIT7_'''.PAR (file unit 3)

Set of fit parameter data files generated in a run of ACFIT7SA with the extension number specified there by FFORM. The data on these files are sequentially input to ACFITEV4.The data format must not be transferred.

- ACFITEV4_'''.PLO (file unit 4)

This file contains the output data for plotting. The extension number is an input parameter for run identification. For each subrun in ACFIT7SA the data of the selected fit range are output on two lines in the format (I4,2X,1P,9E14.5/6X,E14.5). The first two data denote :

ISEC : current subrun number i.

T : time point $t_i$ (sec).

Then 9 data of the selected fit range follow for the subrun in question :

C0 ($C_0$), A0 ($A_0$), B0 ($B_0$), ALPHA ($\alpha$ in sec/rad), FR0CF ($\omega_c / 2\pi$ in Hz), FLAMDA ($\lambda$ in 1/sec), DR, PEAKMAX ($P_{max}$), and CHISQR ($\chi_R^2$) on the seconde line.

## 12.3 Parameter Data Input

- line 1, format (A), RUN

A string of max. 50 characters for run identification.

- line 2, format (I4), NS

Number of data points in a segment of the signal record.

Internal restrictions : The value must be a power of 2 within $2^5$ to $2^{10}$.

- line 3, format (I1), NAV

Number of segments involved in the ACF estimations by the code ACCF2 (Section 8). The code assumes a small number for estimating either instantaneous ACFs or 'short-time' ACFs.

Internal restrictions : IF(NAV.LT.1) NAV=1

IF(NAV.GT.9) NAV=9

- line 4, format (E12.5), SFR

Sampling frequency of the signal record in Hz.

Internal restriction : IF(SFR.LE.0..OR.SFR.GT.100.) SFR=100.

- line 5, format (I2), JOBNR

Job number of the fit range to be selected. It refers to all subruns.

Internal restrictions : IF(JOBNR.LT.1) JOBNR=1

IF(JOBNR.GT.MRMAX) JOBNR=MRMAX

where MRMAX is an internal parameter set equal 20. It corresponds to the allowed maximum number of RANGE values in ACFIT7SA.

- line 6, format (I2), NSEG

Number of successive subruns to be included in the treatment. It must correspond to the value used in ACFIT7SA, if the data of all subruns produced there should be included. It can be smaller if only a part of subruns is considered. But it should not exceed the available residual number of subruns.

Internal restrictions : IF(NSEG.LT.1) NSEG=1

IF(NSEG.GT.NSEGMAX) NSEG=NSEGMAX

where NSEGMAX is an internal parameter set equal 50 which is the allowed maximum number of subruns in ACFIT7SA.

- line 7, format (A), FFORM

A string of 3 characters specifying the extension number of ACFIT7_'''.PAR for the first subrun to be taken. It must not correspond to the smallest number. It can also relate to a later subrun number. Together with an appropriate value attributed to NSEG, any part of the available successive subruns can be selected. The structure of FFORM is explained in the description of the code ACFIT7SA.

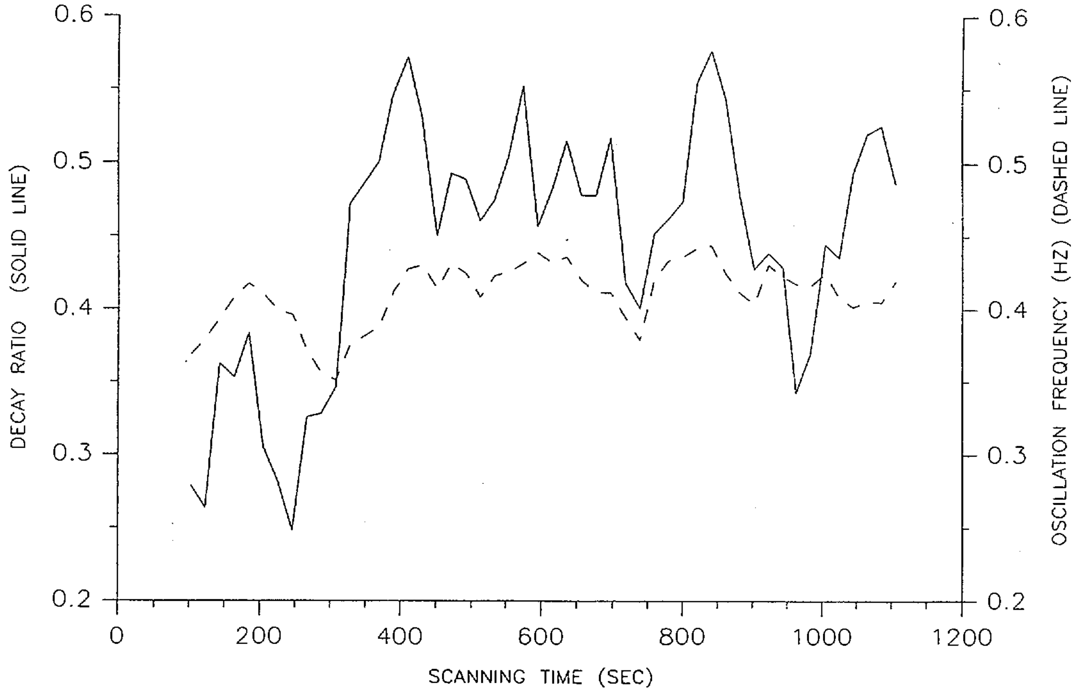
- line 8, format (A), PFORM

A string of 3 characters specifying the extension number of ACFITEV4_'''.PLO for run identification.

### 12.4 Example

As an example, Fig. 12.1 shows a plot of the decay ratio (full line) and the ACF oscillation frequency (dashed line) as functions of the scanning time, estimated by the gliding segment analysis (GLSA) over the entire length of the benchmark record C2_TEST.L1. The fit procedure refers to the option IBGR=2. The plotted data are taken from the job number at the optimum fit range whose determination requires the application of the code ACFITEV5 (Section 13). As already mentioned in Section 11.6, the analyzed record exhibits a well visible transient behaviour of the DR at a time point of about 300 sec.

**Fig. 12.1:** Decay Ratio (full line) and Oscillation Frequency (dashed line) in the GLSA of the Benchmark Record C2_TEST.L1 at the Optimum Fit Range as Functions of the Scanning time.



# LISTING

## ACFITEV4

```
      PROGRAM ACFITEV4
C
C     ACFITEV4 IS AN AUXILIARY CODE FOR THE GLIDING SEGMENT ANALYSIS
C     (GLSA). IT ORDERS THE FIT PARAMETER DATA OBTAINED FROM THE CODE
C     ACFIT7SA SEGMENTWISE FOR A GIVEN FIT RANGE. THE DATA CAN BE
C     PLOTTED AS FUNCTION OF THE ANALYSIS TIME.
C
C     CODE WRITTEN BY K.BEHRINGER, JUNE 2000.
C
C     IMSL ROUTINES ARE NOT REQUIRED.
C
C     PARAMETERS :
C
C         LINE      : NUMBER OF FIT PARAMETER DATA (=9)
C         MRMAX     : MAXIMUM NUMBER OF FIT RANGES
C         NSEGMAX   : MAXIMUM NUMBER OF SEGMENTS
C                     (SEGMENT SHIFTS + 1)
C
C
C     DECLARATIONS
C
      PARAMETER (LINE=9,MRMAX=20,NSEGMAX=50)
      CHARACTER*50 RUN,FILE1,FILE2,FILE3,FILE4
      CHARACTER*3 FFORM,PFORM
      DIMENSION Y(LINE)
      DATA
     1FILE1/'ACFITEV4.IN                                          '/,
     2FILE2/'ACFITEV4.PRT                                         '/,
     3FILE3/'ACFIT7_^^^.PAR                                       '/,
     4FILE4/'ACFITEV4_^^^.PLO                                     '/
C
C     FORMATS
C
 1000 FORMAT(A/I4/I1/E12.5/I2/I2/A/A)
 2000 FORMAT('1',40X,'P R O G R A M     A C F I T E V 4'///1X,
     1'FILES : PARAMETER DATA INPUT : FILE1 = ',A/9X,
     2'PRINT OUTPUT',9X,': FILE2 = ',A/9X,
     3'DATA INPUT',11X,': FILE3 = ',A/9X,
     4'PLOT OUTPUT',10X,': FILE4 = ',A///)
 2005 FORMAT(1X,'RUN DENOTATION',26X,'RUN',5X,'=',1X,A//1X,
     1'SEGMENT LENGTH (NUMBER OF DATA POINTS)',2X,'NS',6X,'=',I5//1X,
     2'NUMBER OF AVERAGES IN THE ACF''S',9X,'NAV',5X,'=',I5//1X,
     3'SAMPLING FREQUENCY (HZ)',17X,'SFR',5X,'=',1PE15.5//1X,
     4'SELECTED FIT RANGE NUMBER',15X,'JOBNR',3X,'=',I5//1X,
     5'GIVEN NUMBER OF SEGMENTS',16X,'NSEG',4X,'=',I5//1X,
     6'INITIAL EXTENSION NUMBER OF FILE3',7X,'FFORM',3X,'=',1X,A//1X,
     7'EXTENSION NUMBER OF FILE4',15X,'PFORM',3X,'=',1X,A//)
 2010 FORMAT(1X,'TIME LENGTH OF A SEGMENT (SEC)',10X,'DT',6X,'=',
     11PE15.5//1X,
     2'START TIME OF THE DATA PLOT',13X,'T0',6X,'=',E15.5)
 2015 FORMAT(////11X,'E N D')
 2020 FORMAT(////11X,'WRONG INPUT PARAMETER DATA !')
 2025 FORMAT('1ISEG',8X,'TIME (SEC)',10X,'DATA FILES'/)
```

```
 2030 FORMAT(2X,I2,7X,1PE12.5,8X,A)
 2035 FORMAT('+',81X,'FILE ABORTED')
 2040 FORMAT(////11X,'OPENING ERROR ! NF = ',I1)
 2045 FORMAT(////11X,'READ-ERROR IN FILE3 !')
 2050 FORMAT(////11X,'SELECTED JOB NUMBER DOES NOT EXIST !')
 3000 FORMAT(20X,8E14.5/6X,E14.5)
 4000 FORMAT(I4,2X,1P,9E14.5/6X,E14.5)
C
C     INPUT PARAMETER DATA
C
      OPEN(UNIT=2,FILE=FILE2,STATUS='NEW',DEFAULTFILE='DIRINPUT')
      WRITE(2,2000) FILE1,FILE2,FILE3,FILE4
      NF = 1
      OPEN(UNIT=1,FILE=FILE1,STATUS='OLD',DEFAULTFILE='DIRINPUT',
     1ERR=100)
      READ(1,1000,ERR=105,END=105) RUN,NS,NAV,SFR,JOBNR,NSEG,
     1FFORM,PFORM
      IF(NAV.LT.1) NAV=1
      IF(NAV.GT.9) NAV=9
      IF(JOBNR.LT.1) JOBNR=1
      IF(JOBNR.GT.MRMAX) JOBNR=MRMAX
      IF(NSEG.LT.1) NSEG=1
      IF(NSEG.GT.NSEGMAX) NSEG=NSEGMAX
      WRITE(2,2005) RUN,NS,NAV,SFR,JOBNR,NSEG,FFORM,PFORM
      DO 1 N=5,10
      IF(NS.EQ.2**N) GO TO 110
    1 CONTINUE
      GO TO 115
  110 IF(SFR.GT.0..AND.SFR.LE.100.) GO TO 120
  115 WRITE(2,2020)
      STOP
  120 DT = FLOAT(NS)/SFR
      T0 = FLOAT(NAV)*DT
      WRITE(2,2010) DT,T0
C
C     DATA INPUT AND READ-OUT
C
      FILE4(10:12) = PFORM
      NF = 4
      OPEN(UNIT=4,FILE=FILE4,STATUS='NEW',DEFAULTFILE='DIRINPUT',
     1ERR=100)
      WRITE(2,2025)
      NF = 3
      DO 2 ISEG=1,NSEG
      CALL INDEX(FILE3,FFORM,ISEG)
      OPEN(UNIT=3,FILE=FILE3,STATUS='OLD',DEFAULTFILE='DIRINPUT',
     1ERR=100)
      DO 3 JOB=1,JOBNR
    3 READ(3,3000,ERR=125,END=130) (Y(L),L=1,LINE)
      T = T0+DT*FLOAT(ISEG-1)
      WRITE(2,2030) ISEG,T,FILE3
      IF(Y(1).GE.1.E30) THEN
      WRITE(2,2035)
      GO TO 2
      ELSE
      WRITE(4,4000) ISEG,T,(Y(L),L=1,LINE)
      END IF
```

```
    2 CLOSE(UNIT=3,STATUS='KEEP')
  135 WRITE(2,2015)
      STOP
  100 WRITE(2,2040) NF
      IF(NF.EQ.3) GO TO 135
      STOP
  105 WRITE(2,2020)
      STOP
  125 WRITE(2,2045)
      STOP
  130 WRITE(2,2050)
      STOP
      END
C========================================================
      SUBROUTINE INDEX(FINPUT,FFORM,ISEG)
C
C     INDEX INCREASES THE CURRENT NUMBER OF FINPUT (ACFIT7_^^^.PAR)
C     BY 1. THE INITIAL NUMBER IS CONTAINED IN FFORM.
C     THE FIRST OF THE THREE CHARACTERS OF FFORM CAN BE A LETTER
C     OR A NUMBER. IT IS LEFT UNCHANGED. THE TWO FURTHER CHARACTERS
C     REPRESENT THE INITIAL NUMBER OF FINPUT WITH THE RESTRICTION
C     TO THE VALUE 1 UNTIL 99-NSEG.
C
      CHARACTER*50 FINPUT
      CHARACTER*3 FFORM
      SAVE IFF
 2000 FORMAT(////11X,'CURRENT NUMBER OF FINPUT EXCEEDS MAXIMUM VALUE !')
      IF(ISEG.GT.1) GO TO 100
      IF(FFORM(3:3).EQ.' ') THEN
      FFORM(3:3) = FFORM(2:2)
      FFORM(2:2) = '0'
      END IF
      FINPUT(8:10) = FFORM
      DECODE(2,'(I2)',FFORM(2:3)) IFF
      IFF = IFF-1
      RETURN
  100 IFIN = IFF+ISEG
      IF(IFIN.GT.99) THEN
      WRITE(2,2000)
      STOP
      ELSE
      ENCODE(2,'(I2.2)',FINPUT(9:10)) IFIN
      END IF
      RETURN
      END
```

# 13 PROGRAM ACFITEV5

Precision : single

Operation : foreground, background

Required auxiliary routines : none

Purpose of the program :

Calculation of Average Values and Standard Deviations of the Fit Parameter Data in the Gliding Segment Analysis

Feature Summary :

- Corresponding values of the fit parameters and other derived quantities obtained from the code ACFIT7SA are averaged over the subruns, for each fit range, i.e for each job, separately.
- Standard deviations are calculated.
- Cancelled jobs are eliminated.

## 13.1 Comments

The application of the code concerns the last step in the gliding segment analysis (GLSA). The code calculates average values and standard deviations over the subruns as functions of the fit range, i.e. of the fit lag time end point $\tau_{end}$ (job number) from the data written on the files ACFIT7_'''.PAR which have been output by the code ACFIT7SA (Section 11). Cancelled jobs in the data of the subruns are eliminated. The average values and the standard deviation values refer always to the number of 'good' subruns for the job in question. The output average values and standard deviation values for $|B_0|$ and $\alpha$ become meaningless if in the five-parametric fit procedure one or a few large subrun values are present. The appearance of such values has been admitted without counteractions in a run of the code ACFIT7SA. Another remark concerns the average value of $P_{max}$. As mentioned in Section 10.1, the evaluation of $P_{max}$ is restricted to estimated values of DR<1; otherwise $P_{max}$ is set equal zero. Such zero-values are not eliminated. If their number is small against the number of 'good' subruns, their influence in the average value of $P_{max}$ may not be dramatic. However, a list of the number of detected zero-values is given on the print file for each job.

The calculation of standard deviations assumes tacitly that the signal record to be analyzed is stationary. Otherwise, higher values are expected to result which are not only due to statistics, but may also reflect transient process behaviour. An investigation of this problem under 'clean' and well defined conditions could be made with noise signals artificially generated by the code RICE3 (Section 14) with a given spectral density (PSD) which simulates model B and includes filtering, i.e. the data of the reference PSD must only be given within the bandwidth of the assumed rectangular filter. The application of the code FFTF2 (Section 5) is then not needed.

The code is designed in such a way that all data from all files ACFIT7_'''.PAR to be considered, are sequentially read in in a 3-dimensional array. The values of $\tau_{end}$ on the first file are, in addition, stored separately. The code checks the agreement of these values correspondingly with those on the following files.

### 13.2 Files

There are essentially 5 files :

- ACFITEV5.IN (file unit 1)

The input parameter data are written on this file.

- ACFITEV5.PRT (file unit 2)

This file is destinated for printing. It contains (with text) all input parameter data and messages from the computation progress. In addition, a list of the series of the files ACFIT7_'''.PAR to be included in a run is given. A second list follows which gives the number of averages (with respect to the 'good' subruns) as a function of the job number (fit range). Optionally, all data on the used files ACFIT7_'''.PAR can be output. This is less tedious than to print the data on these files before separately. Finally, a list is given of the detected zero-values of $P_{max}$ as a function of the job number.

A remark to the text must be made in order to avoid confusion. As in the codes ACFIT7SA (Section 11) and ACFITEV4 (Section 12), the text uses the denotation 'number of segments' with the attributed variable NSEG. It refers more precisely to the number of subruns. In the originally first investigated segment analysis (SA) with instantaneous auto-correlation function (ACF) estimates the number of subruns corresponds to the number of segments.

- ACFITEV5_'''.AV (file unit 3)

The average values are output on this file. The data are written in the same formatted manner as given on the files ACFIT7_'''.PAR. They appear for each job on two lines with the format (I4,2X,1P,9E14.5/6X,E14.5). The extension number is an input parameter for run identification.

- ACFITEV5_'''.STD (file unit 4)

The standard deviation values are output on this file. The data are also written in the same formatted manner as given on the files ACFIT7_'''.PAR with the exception that two additional data are written at the end of the data string of a job. These two additional data are the standard deviation of $\chi^2_R$ divided by the average value $\overline{\chi^2_R}$, and the square of the standard deviation of $\chi^2_R$ divided by the average value $\overline{\chi^2_R}$. The data appear again for each job on two lines with the format (I4,2X,1P,9E14.5/6X,3E14.5). The extension number is the same as in ACFITEV5_'''.AV.

- NFILE = ACFIT7_'''.PAR (file unit variable, starting with 5)

Set of fit parameter data files generated in a run of ACFIT7SA with the extension numbers specified there by FFORM. The file names and the data on these files are sequentially input to ACFITEV5. The file names are stored. The data format must not be transferred.

## 13.3 Parameter Data Input

- line 1, format (A), RUN

A string of max. 50 characters for run identification.

- line 2, format (A), FFORM

A string of 3 characters specifying the extension number of ACFIT7_'''.PAR for the first subrun to be taken. It must not correspond to the smallest number. It can also relate to a later subrun number. Together with an appropriate value attributed to NSEG, any part of the available successive subruns can be selected. The structure of FFORM is explained in the description of the code ACFIT7SA.

- line 3, format (A), PFORM

A string of 3 characters specifying the extension number of ACFITEV5_'''.AV and ACFITEV5_'''.STD for run identification.

- line 4, format (I2), MRANGE

Number of jobs (fit ranges) in each subrun.

Internal restrictions : IF(MRANGE.LT.1) MRANGE=1

IF(MRANGE.GT.MRMAX) MRANGE=MRMAX

where MRMAX is an internal parameter set equal 20, which value corresponds to the allowed maximum number of RANGE values in ACFIT7SA.

If the value given for MRANGE exceed the available number of jobs, it will automatically be reduced. The same will happen in the case of a detected disagreement in the mentioned check on the $\tau_{end}$ values.

- line 5, format (I2), NSEG

Number of successive subruns to be included in the treatment. It must correspond to the value used in ACFIT7SA, if the data of all subruns produced there should be considered. It can be smaller, if only a part of successive subruns should be treated. But it should not exceed the available residual number of subruns. However, an erroneously given value will be corrected by the code.

Internal restrictions : IF(NSEG.LT.2) NSEG=2

IF(NSEG.GT.NSEGMAX) NSEG=NSEGMAX

where NSEGMAX is an internal parameter set equal 50, which is the allowed maximum number of subruns in ACFIT7SA.

- line 6, format (I1), IOUT

Option parameter for the read-out of the data on the involved files ACFIT7_'''.PAR to the print file ACFITEV5.PRT.

IOUT=1 : no read-out

2 : read-out

The data of two subsequent files are given on one page, if the value of MRANGE is less or equal 15. Order and format of the data are transferred unchanged.

Internal restrictions : IF(IOUT.LT.0) IOUT=0

IF(IOUT.GT.1) IOUT=1

# LISTING

## ACFITEV5

```
      PROGRAM ACFITEV5
C
C     ACFITEV5 REFERS TO THE CODE ACFIT6 AND ACFIT7 (FILES ACFIT6_^^^.PA
C     ACFIT7_^^^.PAR) IF IT IS APPLIED TO SEGMENT ACF'S. IT CALCULATES
C     THE SEGMENT AVERAGE VALUES AND THE STANDARD
C     DEVIATIONS OF THE FOLLOWING 9 QUANTITIES WHICH ARE GIVEN AS
C     FUNCTIONS OF TAUEND (RANGE END OF A FIT IN A SEGMENT) :
C
C     C0      : AMPLITUDE OF THE IDEAL ACF
C     A0      : AMPLITUDE OF THE IDEAL PSD
C     B0      : BACKGROUND AMPLITUDE IN THE MEASURED PSD
C     ALPHA   : BACKGROUND DECAY CONSTANT IN THE MEASURED PSD (SEC/RAD)
C     FROCF   : ACF FREQUENCY (HZ)
C     FLAMDA  : DECAY TIME CONSTANT OF THE ACF (1/SEC)
C     DR      : DECAY RATIO
C     PEAKMAX : PEAK AMPLITUDE IN THE IDEAL PSD
C     CHISQR  : REDUCED CHI-SQUARE
C
C     THE AVERAGE VALUES ARE OUTPUT ON FILE ACFITEV5_^^^.AV . THE STANDA
C     DEVIATIONS ARE OUTPUT ON FILE ACFITEV5_^^^.STD . IN ADDITION, THE
C     FOLLOWING TWO QUANTITIES ARE GIVEN ON FILE ACFITEV3_^^^.STD :
C
C     STANDARD DEVIATION OF CHISQR / AVERAGE VALUE OF CHISQR, AND
C     SQUARE OF THE STANDARD DEVIATION OF CHISQR / AVERAGE VALUE OF CHIS
C
C     CODE WRITTEN BY K.BEHRINGER, JUNE 2000.
C     ACFITEV5 IS A MODIFIED VERSION OF ACFITEV3.
C
C     IMSL ROUTINES ARE NOT REQUIRED.
C
C     PARAMETERS :
C
C          LINE    : NUMBER OF AVERAGE VALUES OR STANDARD DEVIATIONS
C                    FOR A FIT RANGE (=9)
C          MRMAX   : MAXIMUM NUMBER OF FIT RANGES
C          NSEGMAX : MAXIMUM NUMBER OF SEGMENTS
C
C     UNIT AND FILE NOTATIONS :
C
C          UNIT NF=1            FILE1
C               NF=2            FILE2
C               NF=3            FILE3
C               NF=4            FILE4
C               NF=5            NFILE(1)
C               NF=6            NFILE(2)
C               ........................
C               NF=NSEGMAX+4    NFILE(NSEGMAX)
C
C     NOTE : IF THE NAMES OF FILE3 AND/OR FILE4 ARE CHANGED,
C            THE CORRESPONDING ENCODE STATEMENTS MUST BE ADAPTED.
C
C
C     DECLARATIONS
```

```
C
      PARAMETER (LINE=9,MRMAX=20,NSEGMAX=50)
      PARAMETER (IZERO=LINE*MRMAX)
      CHARACTER*50 NFILE(NSEGMAX),RUN,FILE1,FILE2,FILE3,FILE4,FINPUT
      CHARACTER*3 FFORM,PFORM
      INTEGER*2 JOBS
      DIMENSION Y(LINE,MRMAX,NSEGMAX),YAV(LINE,MRMAX),YSTD(LINE,MRMAX),
     1TAUEND(MRMAX),TAUENDS(MRMAX),JOBS(MRMAX,NSEGMAX),XSEG(NSEGMAX),
     2MPEAK(MRMAX)
      EQUIVALENCE (TAUEND(1),MPEAK(1))
      DATA L,M/2*0/,YAV/IZERO*0./,
     1YSTD/IZERO*0./,XSEG/NSEGMAX*0./,
     2FILE1/'ACFITEV5.IN                                      '/,
     3FILE2/'ACFITEV5.PRT                                     '/,
     4FILE3/'ACFITEV5_^^^.AV                                  '/,
     5FILE4/'ACFITEV5_^^^.STD                                 '/,
     6FINPUT/'ACFIT7_^^^.PAR                                   '/
C     FOR ACFIT6 : SET FINPUT = ACFIT6_^^^.PAR
C     FOR ACFIT7 : SET FINPUT = ACFIT7_^^^.PAR
C
C     FORMATS
C
 1000 FORMAT(3(A/),I2/I2/I1)
 2000 FORMAT('1',40X,'P R O G R A M     A C F I T E V 5'//1X,
     1'FILES : PARAMETER INPUT',16X,': FILE1 = ',A/9X,
     2'PRINT OUTPUT',19X,': FILE2 = ',A/9X,
     3'AVERAGE DATA OUTPUT',12X,': FILE3 = ',A/9X,
     4'STANDARD DEVIATION DATA OUTPUT',1X,': FILE4 = ',A/9X,
     5'SEGMENT DATA INPUT (FINPUT)',4X,': NFILE(1 TO NSEGMAX)'//)
 2005 FORMAT(1X,'RUN DENOTATION',36X,'RUN',5X,'=',1X,A//1X,
     1'INITIAL EXTENSION NUMBER OF FINPUT',16X,'FFORM',3X,'=',1X,
     2A//1X,
     3'EXTENSION NUMBER OF FILE3 AND FILE4',15X,'PFORM',3X,'=',1X,
     4A//1X,
     5'GIVEN NUMBER OF FIT RANGES',24X,'MRANGE',2X,'=',I3//1X,
     6'GIVEN NUMBER OF SEGMENT FILES',21X,'NSEG',4X,'=',I3//1X,
     7'READ-OUT OF INPUT DATA (0:NO, 1:YES)',14X,'IOUT',4X,'=',I2/)
 2010 FORMAT(1X,'ACCEPTED NUMBER OF SEGMENT FILES',18X,'NSEG',4X,'=',
     1I3/)
 2015 FORMAT('1SEGMENT FILE DENOTATION')
 2020 FORMAT(51X,'NFILE',I2.2,1X,'=',1X,A)
 2025 FORMAT(1X,'REDUCED NUMBER OF FIT RANGES',22X,'MRANGE',2X,'=',I3)
 2030 FORMAT(////11X,'E N D')
 2035 FORMAT(////11X,'OPENING-ERROR ! NF =',I4)
 2040 FORMAT(////11X,'READ-ERROR ! L =',I4,', M=',I4,', NF =',I4)
 2045 FORMAT('1AVAILABLE NUMBER OF SEGMENTS PER FIT RANGE',8X,
     1'JOB',6X,'JSEG'/<MRANGE>(/51X,I3,7X,I3))
 2050 FORMAT(1X,'SEGMENT NUMBER N =',I4/)
 2055 FORMAT('1NUMBER OF ZERO-VALUES OF PEAKMAX',18X,'JOB',6X,
     1'MPEAK'/)
 2060 FORMAT(51X,I3,7X,I3)
 3000 FORMAT(I4,2X,1P,9E14.5/6X,E14.5)
 4000 FORMAT(I4,2X,1P,9E14.5/6X,3E14.5)
 5000 FORMAT(I4,2X,9E14.5/6X,E14.5)
C
C     PARAMETER DATA INPUT
C
```

```
      OPEN(UNIT=2,FILE=FILE2,STATUS='NEW',DEFAULTFILE='DIRINPUT')
      WRITE(2,2000) FILE1,FILE2,FILE3,FILE4
      NF = 1
      OPEN(UNIT=1,FILE=FILE1,STATUS='OLD',DEFAULTFILE='DIRINPUT',
     1ERR=100)
      READ(1,1000,ERR=105,END=105) RUN,FFORM,PFORM,
     1MRANGE,NSEG,IOUT
      IF(MRANGE.LT.1) MRANGE=1
      IF(MRANGE.GT.MRMAX) MRANGE=MRMAX
      IF(NSEG.LT.2) NSEG=2
      IF(NSEG.GT.NSEGMAX) NSEG=NSEGMAX
      IF(IOUT.LT.0) IOUT=0
      IF(IOUT.GT.1) IOUT=1
      WRITE(2,2005) RUN,FFORM,PFORM,MRANGE,NSEG,IOUT
      DO 1 N=1,NSEG
      CALL INDEX(FINPUT,FFORM,N)
    1 NFILE(N) = FINPUT
      WRITE(2,2015)
      DO 2 N=1,NSEG
    2 WRITE(2,2020) N,NFILE(N)
      WRITE(2,'(1X)')
      CLOSE(UNIT=1,STATUS='KEEP')
C
C     SEGMENT DATA INPUT
C
      DO 3 N=1,NSEG
      NF = N+4
    3 OPEN(UNIT=NF,FILE=NFILE(N),STATUS='OLD',DEFAULTFILE='DIRINPUT',
     1ERR=110)
  125 DO 4 N=1,NSEG
      NF = N+4
      DO 4 M=1,MRANGE
      READ(NF,5000,ERR=105,END=120) JOB,TAUEND(M),(Y(L,M,N),L=1,LINE)
      IF(N.EQ.1) TAUENDS(M)=TAUEND(M)
      IF(N.GT.1.AND.TAUEND(M).NE.TAUENDS(M)) GO TO 120
    4 CONTINUE
      DO 5 NF=5,NSEG+4
    5 CLOSE(UNIT=NF,STATUS='KEEP')
C
C     SELECTION OF THE GOOD JOBS
C
      L = 1
      DO 6 N=1,NSEG
      DO 6 M=1,MRANGE
      JOBS(M,N) = 0
      IF(Y(L,M,N).LT.1.E30) JOBS(M,N)=1
      XSEG(M) = XSEG(M)+FLOATI(JOBS(M,N))
    6 CONTINUE
      WRITE(2,2045) (M,IFIX(XSEG(M)),M=1,MRANGE)
      XSEGM = MRMAX+1
      DO 7 M=1,MRANGE
      IF(XSEG(M).GT.1) GO TO 7
      XSEG(M) = XSEGM
      DO 8 N=1,NSEG
    8 JOBS(M,N) = 0
    7 CONTINUE
C
C     COMPUTATION AND READ-OUT
```

```
C
      IF(IOUT.EQ.0) GO TO 130
      IP = 1
      DO 9 N=1,NSEG
      IP = IP+1
      IF(IP.GE.2.OR.MRANGE.GT.15) THEN
      WRITE(2,'(1H1,$)')
      IP = 0
      ELSE
      WRITE(2,'(/)')
      END IF
      WRITE(2,2050) N
      DO 9 M=1,MRANGE
    9 WRITE(2,3000) M,TAUENDS(M),(Y(L,M,N),L=1,LINE)
  130 DO 10 N=1,NSEG
      DO 10 M=1,MRANGE
      DO 10 L=1,LINE
   10 YAV(L,M) = YAV(L,M)+FLOATI(JOBS(M,N))*Y(L,M,N)
      DO 11 M=1,MRANGE
      DO 11 L=1,LINE
   11 YAV(L,M) = YAV(L,M)/XSEG(M)
      NF = 3
      ENCODE(3,'(A)',FILE3(10:12)) PFORM
      OPEN(UNIT=3,FILE=FILE3,STATUS='NEW',DEFAULTFILE='DIRINPUT',
     1ERR=100)
      DO 12 M=1,MRANGE
      IF(XSEG(M).EQ.XSEGM) GO TO 12
      WRITE(3,3000) M,TAUENDS(M),(YAV(L,M),L=1,LINE)
   12 CONTINUE
      DO 13 N=1,NSEG
      DO 13 M=1,MRANGE
      IF(JOBS(M,N).EQ.0) GO TO 13
      DO L=1,LINE
      YSTD(L,M) = YSTD(L,M)+(Y(L,M,N)-YAV(L,M))**2
      END DO
   13 CONTINUE
      DO 14 M=1,MRANGE
      DO 14 L=1,LINE
   14 YSTD(L,M) = SQRT(YSTD(L,M)/(XSEG(M)-1.))
      NF = 4
      ENCODE(3,'(A)',FILE4(10:12)) PFORM
      OPEN(UNIT=4,FILE=FILE4,STATUS='NEW',DEFAULTFILE='DIRINPUT',
     1ERR=100)
      DO 15 M=1,MRANGE
      IF(XSEG(M).EQ.XSEGM) GO TO 15
      Y10 = YSTD(LINE,M)/YAV(LINE,M)
      Y11 = YSTD(LINE,M)**2/YAV(LINE,M)
      WRITE(4,4000) M,TAUENDS(M),(YSTD(L,M),L=1,LINE),Y10,Y11
   15 CONTINUE
      DO 16 M=1,MRANGE
   16 MPEAK(M) = 0
      L = 8
      DO 17 N=1,NSEG
      DO 17 M=1,MRANGE
      IF(Y(L,M,N).EQ.0.) MPEAK(M)=MPEAK(M)+1
   17 CONTINUE
      WRITE(2,2055)
```

```
      DO 18 M=1,MRANGE
      IF(XSEG(M).EQ.XSEGM) GO TO 18
      WRITE(2,2060) M,MPEAK(M)
   18 CONTINUE
      WRITE(2,2030)
      STOP
  100 WRITE(2,2035) NF
      STOP
  110 NSEG = N-1
      WRITE(2,2010) NSEG
      IF(NSEG.GE.2) GO TO 125
      STOP
  105 WRITE(2,2040) L,M,NF
      STOP
  120 MRANGE = M-1
      WRITE(2,2025) MRANGE
      IF(MRANGE.LT.1) STOP
      DO 19 NF=5,NSEG+4
   19 REWIND(UNIT=NF)
      GO TO 125
      END
C============================================================
      SUBROUTINE INDEX(FINPUT,FFORM,ISEG)
C
C     INDEX INCREASES THE CURRENT NUMBER OF FINPUT (ACFIT7_^^^.PAR)
C     BY 1. THE INITIAL NUMBER IS CONTAINED IN FFORM.
C     THE FIRST OF THE THREE CHARACTERS OF FFORM CAN BE A LETTER
C     OR A NUMBER. IT IS LEFT UNCHANGED. THE TWO FURTHER CHARACTERS
C     REPRESENT THE INITIAL NUMBER OF FINPUT WITH THE RESTRICTION
C     TO THE VALUE 1 UNTIL 99-NSEG.
C
      CHARACTER*50 FINPUT
      CHARACTER*3 FFORM
      SAVE IFF
 2000 FORMAT(////11X,'CURRENT NUMBER OF FINPUT EXCEEDS MAXIMUM VALUE !')
      IF(ISEG.GT.1) GO TO 100
      IF(FFORM(3:3).EQ.' ') THEN
      FFORM(3:3) = FFORM(2:2)
      FFORM(2:2) = '0'
      END IF
      FINPUT(8:10) = FFORM
      DECODE(2,'(I2)',FFORM(2:3)) IFF
      IFF = IFF-1
      RETURN
  100 IFIN = IFF+ISEG
      IF(IFIN.GT.99) THEN
      WRITE(2,2000)
      STOP
      ELSE
      ENCODE(2,'(I2.2)',FINPUT(9:10)) IFIN
      END IF
      RETURN
      END
```

# 14. PROGRAM RICE3 AND AUXILIARY PROGRAM PRERICE3

Precision : single

Operation : foreground, background (RICE3)
interactive (PRERICE3)

Required auxiliary routines : from IMSL : RNSET, RNNOA, FFTRI, F2TRB (RICE3)

Purpose of the program :

Digital Generation of a Stationary Gaussian Random Noise Signal with Specified Spectral Properties

Feature Summary :

- The code RICE3 allows the digital generation of a very long stationary Gaussian random noise record with given spectral characteristics.
- The generation procedure is based on the Rice representation of Gaussian random noise and on windowing the Rice sequences for obtaining a continuous noise record of arbitrary length.
- Two window functions are available : the two-sequence cosine window and the two-sequence square root window.
- The spectral characteristics of the noise to be generated are obtained from a power spectral density (PSD) which must be specified as a reference or target PSD.
- The interactive code PRERICE3 facilitates the parameter data input to the code RICE3.

## 14.1 Mathematical Background

The principles of the noise generation procedure realized in the code RICE3 are given in the paper of Behringer et al. (1986). But it is felt that a short recapitulation cannot be avoided. According to Rice (1944, 1945), an ensemble member of stationary Gaussian random noise can be represented by a Fourier series. Each member of the ensemble is defined in a finite time span $T_s$. In the discrete version, a member or sequence containing $N_T$ data points sampled with the Nyquist interval $\Delta t = T_s / N_T$ follows from

$$x_n^{(s)} = \sum_{k=1}^{N_T/2-1} \left[a_k \cos(2\pi kn / N_T) + b_k \sin(2\pi kn / N_T)\right]; n = 0,...N_T - 1 \quad (14.1)$$

The value of $N_T$ is assumed to be a power of 2, and has the meaning of a transform size. The amplitude coefficients $a_k$ and $b_k$, two in each Nyquist co-interval $\Delta f_r = 1/T_S$, are assumed to be independent and normally distributed random numbers. They have the ensemble averages

$$< a_k >=< b_k >= 0; k = 0,...N_T / 2 - 1 \quad (14.2)$$

$$< a_k a_{k'} >=< b_k b_{k'} >= \sigma_k^2 \delta_{kk'} \quad (14.3)$$

$$< a_k b_{k'} >= 0 \quad (14.4)$$

$\delta_{kk'}$ is the Kronecker symbol. $\sigma_k^2$ is the variance of the signal component at the centre frequency $f_r(k)$ within $\Delta f_r$, and is related to the expected (two-sided) power spectral density (PSD), $S_{kk}^{(EX)}$, by

$$\sigma_k^2 = 2\Delta f_r S_{kk}^{(EX)} \quad (14.5)$$

Conversely, equation (14.5) can be used to generate a Rice sequence of normal noise from a given PSD. This reference PSD is denoted by $S_{kk}^{(RF)}$. If one makes the variance of the contribution of a signal component at a given frequency proportional to the amplitude of the PSD $S_{kk}^{(RF)}$ at that frequency, by selecting the coefficients

$$a_k = Cg_k\sqrt{S_{kk}^{(RF)}} \; , \; b_k = Cg_k'\sqrt{S_{kk}^{(RF)}} \quad (14.6)$$

where $g_k$ and $g_k'$ are independent Gaussian random numbers with the deviate (0,1) and C is a signal normalizing constant, a sequence of normal noise data with any specified colour is obtained. For convenience, C is chosen as

$$C = \frac{1}{\sqrt{\sum_k S_{kk}^{(RF)}}} \quad (14.7)$$

to achieve unit variance of the signal. The variance of the noise signal generated by such Rice sequences has always the expected value equal 1.

The IMSL routine RNNOA is used to generate pseudo-random numbers from a normal distribution with the deviate (0,1). The IMSL routine RNSET provides for the initialization of the seed value in the start of RNNOA.

Equation (14.6) defines, at most, $N_T/2-1$ occupied frequency components. A frequency component is taken as being occupied if $S_{kk}^{(RF)}$ is greater than zero at that frequency. Only an occupied frequency component needs the attribute of a pair of independent random numbers.

$S_{kk}^{(RF)}$ is assumed to be zero at the frequency number k=0 and at the Nyquist cutoff frequency number $k = N_T / 2$. The expected DC value of the noise signal is zero. The generation procedure simulates the requirement in digitizing analogue noise signals where the digitizer is preceeded ordinarily by a high-pass filter to remove the signal DC component and always by a sharp antialiazing filter to eliminate all frequency components with frequencies equal to and above the Nyquist cutoff frequency.

The Rice sequence can be directly computed from equation (14.1) or it can quickly be produced by the inverse fast Fourier transform (FFT) algorithm to the random numbers $a_k$ and $b_k$. The code RICE3 contains both possibilities which were retained from the development period in order to study the difference in speed. In the direct computation via calls to the sine and cosine functions, only occupied frequency components are considered. In the inverse FFT procedure all Fourier coefficients are initialized equal zero, and only such coefficients are replaced by values of the random numbers $a_k$ and $b_k$ from occupied frequency components.

The Rice representation is based on ensemble averaging techniques. Equation (14.1) exhibits one sequence of data whose size must correspond to the segment length used in the direct PSD estimation (code CPSDES3). There is a crucial point which concerns the problem of continuation, since a Fourier series repeats periodically. Also a record of desired length is not obtainable simply by joining together Rice sequencies, each subsequently produced with a new set of random numbers, because such a record does not correspond to the digital image from a stochastic signal being continuous in time. If one takes a PSD estimate (by the code CPSDES3) on such a record in the same manner as it has been produced, using the rectangular signal window without the application of segment overlap, one must get back exactly what has been input, i.e. the estimated PSD much approach to the reference PSD within statistical limits. The signal discontinuities between subsequent Rice sequences do not enter in the analysis because the segments in the estimation procedure coincide with the segments in the generation procedure. On the other hand, if one starts scanning such sequences simply joined together, at a data point which is not the first in the first segment, this time-shifted analysis shows significant leakage and side-lobe effects in the estimated PSD which arise from the discontinuities between subsequent Rice sequences. The same effects can alo be observed when in the analysis a signal window different from the rectangular one is used, together with the application of segment overlap, in order to retrieve the loss in the degrees of freedom due to windowing.

If $N_s$ Rice sequences are generated, there are $N_s$-1 discontinuities in the noise record. The method of realizing an appropriate noise record consists in 'windowing' Rice sequences with a two-sequence window operator $W_{op}^{(2)}$. If one writes $x_n^{(s)}$ as a vector $\vec{x}^{(s)}$, then the new windowed vector $\vec{x}$, containing $N_T N_s$ data points, is obtained from

$$\vec{x} = W_{op}^{(2)} \left\{ \vec{x}^{(1)}, \vec{x}^{(2)}, \ldots\ldots\ldots, \vec{x}^{(N_s)}, \vec{x}^{(N_s+1)} \right\} \tag{14.8}$$

The procedure requires the generation of an additional sequence for closure. The method preserves the signal variance and represents the digital image of a continuous stationary Gaussion random noise record. It makes the estimated PSD fairly well independent from the starting point in time-shifted analysis and from the application of segment overlap. It reduces leakage and side-lobe production. $\vec{x}$ is also a stochastic time series and normally distributed, since $W_{op}^{(2)}$ is a linear operator. The explicit form of $W_{op}^{(2)}$ is given in the cited paper. Two window operators are known : the two-sequence cosine window and the two-sequence square root window. Both windows have practically equivalent properties. The use of 'windows' refers here to the noise generation procedure and should not be confused with windowing in the direct PSD estimation procedure where a sequence of noise data is weighted by a one-sequence window function.

The maximum length of the noise record to be generated depends on the disc capacity or the allowance for the maximum number of blocks by the system manager. However, the code contains internal restrictions which can easily be changed.

The code PRERICE3 is an interactive preprogram for the quick and convenient preparation of the file RICE.IN which contains the parameter data for operating the code RICE3. In this preprogram a menu of built-in reference PSD functions is included. Presently, 16 functions are available, 6 of them having different mathematical forms.

Anyone of these functions can optionally be called and the data are output on the file RICE0''.PSD. These reference PSD functions do not require any additional parameter data input. The parameter data are fixed in the code. A change to other values must be done there. Test examples with reference PSDs are given in a report by Behringer (1989). It is easy to add further reference PSDs. On the other hand, if an user prefers to establish a reference PSD in a more flexible way with parameter data which can be read in, it is recommended to write a separate code for this.

The noise generation procedure was recently applied to solve a nonlinear system of first-order differential equations by the Runge-Kutta method (Behringer, 1998).

### 14.2 References

Behringer K.,Nishihara H.and Spiekerman G.(1986). Ann.Nucl.Energy 13,443; EIR Report 601.

Behringer K.(1989). Internal PSI Report TM-41-89-10.

Behringer K.(1998). Ann.Nucl.Energy 25,801.

Rice S.O.(1944) Bell Syst.Tech.J. 23,282

Rice S.O.(1945) Bell Syst.Tech.J. 24,46.

### 14.3 Files

There are 3 files in PRERICE3 and 6 files in RICE3.

- PRERICE.PRT (PRERICE3)

This file is meant for printing. It contains (with text) all parameter data which have been designated as input to RICE3. It serves for completing documentation and as a cross-check, since all input parameter data specified for RICE3 appear again on the first print file of RICE3. Additionally, it contains the parameter values of the selected built-in reference PSD.

- RICE.IN (PRERICE3, RICE3)

This file contains the input parameter data for RICE3 which have interactively been produced by PRERICE3. If these data should be written directly (without the help of PRERICE3), order and formats are given in the parameter data input list of RICE3.

- RICE0''.PSD (PRERICE3, RICE3)

This file is only opened in PRERICE3 if a call for a built-in reference PSD function is made. It is numbered by up to two digits through encoding the number of the selected reference PSD. There are 2 columns with the line format (1X,I4,1PE12.5). For enabling the possibility of plotting the data, the first column contains the current frequency number starting with number zero. The reference PSD data are in the second column. For the input to RICE3 the first column is not needed and must be bypassed in the read format. This is automatically done in the read format specification written on the file RICE.IN.

- 'PSDIN' (RICE3)

This file contains the reference PSD data to be input to RICE3 in the case of an external generation. NT/2+1 data points must be given. The first and the last data points are set equal to zero after the data have been read in by RICE3. The data format is an input parameter (PFORM) to be given on RICE.IN. If there is a preceding current point number or a frequency number, it must be bypassed in the format specification. PSDIN is a formal parameter. The name of the file must be specified on the file RICE.IN. If a reference PSD is drawn over from PRERICE3, the name of the file is RICE0''.PSD with the encoded number of the selected reference PSD. In this case, the corresponding data format appears automatically on RICE.IN.

- RICE.DAT (RICE3)

This file contains the computed noise data. There are 3 possibilities for the data format which are specified by the parameter value IOUT (see parameter data input list of PRERICE3).

The following 3 files opened in RICE3 are meant for printing :

- RICE.PR1

This file contains (with text) all input parameter data and messages from the computation progress. In addition, two signal variance values are given. The first value, VARE, is a quantity estimated from the weighted actual Gaussian random numbers appearing in the noise generation process. It should approach to 1 if a sufficiently large number of Gaussian random numbers is involved in the generated noise data. The second value, VARG, is derived from the normalized reference PSD data and must be (as a cross-check) equal to 1.

- RICE.PR2

This file is concerned with the reference PSD data. There are 6 columns :

column

1 : current number of the reference PSD data points, starting with number 1.

2 : frequency in Hz.

3 : given reference PSD data. The first and the last data points are set equal zero.

4 : reference PSD data normalized relatively to the maximum PSD value.

5 : current number of occupied frequency components. It starts with number 1. The PSD value on a line where this number is missing, is excluded from the noise generation process.

6 : frequency number corresponding to the occupied frequency component.

- RICE.PR3

This file contains estimated PSD data of occupied frequency components. For $N_s$ generated sequences, RICE3 computes an estimated PSD value for each occupied frequency component. The estimated PSD data follow from

$$S_{kk}^{(ES)} = \frac{T_s}{4N_s} \sum_{m=1}^{N_s} \left[ \left(a_k^{(m)}\right)^2 + \left(b_k^{(m)}\right)^2 \right] \tag{14.9}$$

If one of the two-sequence windows is applied to the noise generation process, $N_s+1$ Rice sequences must be produced. The squares of the random numbers of the first Rice sequence and of the $(N_s+1)$th Rice sequence are then weighted by a factor of 0.5 in equation (14.9). The estimated PSD data should ideally approach to the reference PSD data of occupied frequency components, if $N_s$ tends to infinity. There are 6 columns :

column

1 : current number of the occupied frequency components.

2 : frequency number.

3 : frequency in Hz.

4 : estimated PSD data for unit signal variance.

5 : reference PSD data, normalized to give unit signal variance.

6 : percentage deviation of the estimated PSD data from the reference PSD data.

### 14.4 Parameter Data Input to PRERICE3

There is an elaborated interactive procedure for the input of the parameter data with text and format indications on the terminal screen. Protection is provided against typing errors, even against data type violations. Each data typed in is immediately rewritten to the screen. It can be corrected if necessary, and must be verified for final acceptance. In a few cases, erroneous data are rejected by the code and must be corrected. The check on the data is the same as implemented in RICE3.

Sequence of the data input :

- format (A), RUN

A string of max. 40 characters for run identification. The string is transferred to the file RICE.IN.

- format (I2), NPSD

Number of reference PSD function. This is the only data which is not transferred to the file RICE.IN. The present range is 0-16.

a) If NPSD=0, no reference PSD function is called. The code assumes that the data of the reference PSD are external on an user file. The code asks then for the name of this file and the data format.

- format (A), PSDIN

A string of max. 40 characters for the file name of the reference PSD data.

- format (A), PFORM

A string of max. 20 characters specifying the line format of the reference PSD data (including the brackets). If, in addition to these data, there is a preceding column containing the frequency values or frequency numbers, it must be bypassed.

b) If NPSD>0, the file RICE0''.PSD is opened with the encoded number of NPSD. The data of the selected reference PSD are computed and written on this file. The value of PFORM which is written on the file RICE.IN equals (5X,E12.5).

- format (E10.3), PRMIN

This parameter determines a percentage level relative to the maximum value in the reference PSD, below which occupied frequency components are cancelled in RICE3. With the help of this parameter one can keep away unimportant small frequency components in the noise generation process, thereby speeding up the computation. The parameter can also be utilized to modify a given reference PSD.

Internal restriction : IF(PRMIN.LT.1.E-4.OR.PRMIN.GE.100.) PRMIN=1.E-2

- format (I4), NT

Transform size (number of data points in a Rice sequence). Its value must be a power of 2 within the range $2^5$ to $2^{13}$. In the analysis of the generated noise data by the direct PSD estimation method (code CPSDES3, Section 4) the same value must be used. NT determines the spectral resolution.

- format (I5), NS

Number of sequences to be generated in the case of no windowing. If by the following parameter IOPT any option for the generation of a windowed noise record is called, then NS+1 sequences are produced. The allowed range is (1,32000), the upper limit depending on the value of NT. The value NT*NS determines the total number of noise data in the record. This number has been limited to 1'024'000 data points.

- format (I1), IOPT

Parameter specifying the noise generation mode. There is the allowance of 6 values :

| | Rice sequence generation | | Two-sequence windowing |
|---|---|---|---|
| | directly | inverse FFT | |
| IOPT=0 : | X | | no windowing |
| 1 : | | X | no windowing |
| 2 : | X | | cosine window |

| | | | |
|---|---|---|---|
| 3 : | | X | cosine window |
| 4 : | X | | square root window |
| 5 : | | X | square root window |

Internal restrictions : IF(IOPT.LT.0) IOPT=0

IF(IOPT.GT.5) IOPT=5

- format (I1), IOUT

Parameter specifying the noise data output format. RICE3 offers 3 possibilities for the sequential output of the generated noise data, whereby each data point is separate.

IOUT= 0 : Formatted noise data, format (1X,1PE12.5).

1 : Formatted noise data with a preceding current data point number starting with number 1, format (1X,I7,1PE12.5).

2 : Unformatted binary noise data (one data point segment in the structure NOG_FLOATING). This output option reduces the required number of blocks by a factor 2 against the option IOUT=0.

Internal restrictions: IF(IOUT.LT.0) IOUT=0

IF(IOUT.GT.2) IOUT=2

- format (E10.3), SFR

Sampling frequency in Hz. This parameter does not enter into the noise generation process. For the time interval between succeeding noise data points one can assume an arbitrary value. Since in RICE3 estimated PSD data are also computed from the unwindowed Rice sequences, they will be scaled there by the value of this parameter.

Internal restriction : SFR>0.

- format (I9), ISEED

Initial seed number for the Gaussian random number generation.

Internal restrictions : IF(ISEED.LT.0) ISEED=0

IF(ISEED.GT.9.E8) ISEED=9.E8

If ISEED is set equal to zero, a seed value is computed in RICE3 using the system clock. The generated noise data are then not reproducible and will represent another ensemble at a different computation time.

### 14.5 Built-in Reference PSD Functions in PRERICE3

The shape of the built-in reference PSDs depends sometimes on the transform size parameter NT. The maximum PSD value has been normalized to 100 % as far as it has been possible without any additional calculation. k denotes the frequency number over the interval (0,NT/2). The range of k specified for each PSD refers to occupied frequency components. Outside this range the PSD data are zero.

NPSD=1 : Broad-band white noise PSD

$PSD_k$=100 ; k=(1,NT/2-1)

NPSD=2 : Band-limited white noise PSD

$PSD_k$=100 ; k=(NT/8,3NT/8)

NPSD=3 : Single-frequency PSD (representing extremely narrow-band random noise)

$PSD_k$=100 ; k=NT/4

NPSD=4 : Single-frequency PSD

$PSD_k$=100 ; k=NT/4+1

NPSD=5 : Lineary decaying PSD

$PSD_k$=100(1-(k-1)/(NT/2-1)) ; k=(1,NT/2)

NPSD=6 : Exponentially decaying PSD, one decade

$PSD_k = 100e^{-\lambda(k-1)}$ ; k=(1,NT/2-1)

with a $\lambda$ value which provides for $PSD_{NT/2-1} / PSD_1 = 10^{-1}$

NPSD=7 : Exponentially decaying PSD, two decades

as NPSD=6, but with a $\lambda$ value for $PSD_{NT/2-1} / PSD_1 = 10^{-2}$

NPSD=8 : Exponentially decaying PSD, four decades

as NPSD=6, but with a $\lambda$ value for $PSD_{NT/2-1} / PSD_1 = 10^{-4}$

NPSD=9 : Sine-shaped PSD

$PSD_k = 50(1 - \cos(4\pi k / NT))$ ; k=(0,NT/2)

NPSD=10 : Narrow-band resonance PSD

$$PSD_k = 100 \frac{4a^2}{(1 - f_k^2)^2 + 4a^2 f_k^2}$$

with $f_k$=4k/NT; k=(1,NT/2-1)

a =0.05

NPSD=11 : Broad-band resonance PSD

as NPSD=10, but with the value a=0.2

NPSD=12 : PSD simulating two broad peaks

$$PSD_k = a_1 \left[ \frac{1}{a_2 + (f_k - a_4)^2} + \frac{1}{a_3 + (f_k - a_5)^2} \right]$$

with $f_k$=2k/NT ; k=(1,NT/2-1)

$a_1$=2.46842E-1

$a_2$=2.5E-3

$a_3$=4.0E-3

$a_4$=1.5625E-1

$a_5$=5.9375E-1

For a full development of the peaks, NT should be ≥ 64.

The following 4 reference PSDs have numerically fixed values for k and refer to the cases NPSD=2-4. They have been used in special test applications.

NPSD=13 : k=(1,50); NT≥128

NPSD=14 : k=(1,40); NT≥128

NPSD=15 : k=(1,8)

NPSD=16 : k=1

If an user wishes to add a further reference PSD function, he has to do the following:

- To write a subroutine PSD7 which computes the data of the new reference PSD function, and include in this subroutine the COMMON block

```
COMMON A(10),P(4097),NTH,K(10)
```

where

A = array for specifying real parameter data (input)

P = array for the computed PSD data (output),

observe that P(N) refers to the frequency number k=N-1.

NTH = NT/2 (input)

K = array for specifying integer parameter data (input)

- To observe that PSD data must only be computed for occupied frequency components. All elements of P as far as they are required, are initialized to be zero on line 216-217 in the main program.

- In the main program :

  - Increase the parameter value attributed to NPSDS from 16 to 17 on line 27.
  - Add the further label 217 to the computed GO TO statement on line 219. Free label numbers are 217-300 and 304-399.
  - Add after line 263 :

    GO TO 400

    217...... attribute parameter data to the elements of A and K according to the requirements in the new subroutine PSD7. Observe that a few elements of these arrays have already been initialized on line 213-215.

    CALL PSD7

### 14.6 Parameter Data Input to RICE3

If the parameter data are edited directly without the help of PRERICE3, they must be written on the file RICE.IN line by line using the following formats :

| | format |
|---|---|
| - RUN | (A) max. 40 characters |
| - PSDIN | (A) max. 40 characters |
| - PFORM | (A) max. 20 characters |
| - PRMIN | (E10.3) |
| - NT | (I4) |
| - NS | (I5) |

- IOPT (I1)
- IOUT (I1)
- SFR (E10.3)
- ISEED (I9)

### 14.7 Numbered Stops in RICE3

- STOP 100 : The file RICE.IN cannot be opened.
- STOP 101 : The file RICE.PR1 cannot be opened.
- STOP 102 : A read-error has occurred or the EOF mark has been reached on the file RICE.IN.
- STOP 103 : The value specified for SFR or the value specified for NT is incorrect.
- STOP 104 : The file PSDIN cannot be opened.
- STOP 105 : A read-error has occurred or the EOF mark has been reached on the file PSDIN.
- STOP 106 : All reference PSD data have values less than or equal to zero.
- STOP 107 : The file RICE.PR2 cannot be opened.
- STOP 108 : There is at least one negative data point among positive data in the reference PSD.
- STOP 109 : The file RICE.DAT cannot be opened.
- STOP 110 : The file RICE.PR3 cannot be opened.

# LISTING

## PRERICE3

```
      PROGRAM PRERICE3
C
C     THE CODE IS AN INTERACTIVE UTILITY PROGRAMME FOR PREPARING THE
C     PARAMETER DATA INPUT TO THE GAUSSIAN RANDOM NOISE GENERATION
C     CODE RICE2. OPTIONALLY, DATA FROM BUILT-IN REFERENCE PSD FUNCTIONS
C     CAN ADDITIONALLY BE COMPUTED.
C
C     CODE WRITTEN BY K.BEHRINGER, APRIL 1989.
C     REWRITTEN IN MAY 1996.
C     EXTENDED VERSION OF PRERICE2 FOR A TRANSFORM SIZE OF 8092 DATA
C     POINTS, SEPTEMBER 1996.
C
C     INTERACTIVE VERSION FOR THE VAX 11/785.
C
C     LINK COMMAND : NO EXTERNAL ROUTINES ARE REQUIRED.
C
      CHARACTER RUN*40,PSDIN*40,PFORM*20,RY,LY
      COMMON A(10),P(4097),NTH,K(10)
C
C     ADJUSTABLE PARAMETERS :
C
C     NPSDS : NUMBER OF BUILT-IN REFERENCE PSD FUNCTIONS
C             (UP TO TWO DIGITS)
C     IOPTS : NUMBER OF NOISE DATA GENERATION OPTIONS IN RICE2
C              MINUS ONE (ONE DIGIT)
C
      PARAMETER (NPSDS=16,IOPTS=5)
C
      DATA PI,LY/3.1415927,'Y'/,
     1PSDIN/'RICE000.PSD                             '/,
     2PFORM/'(5X,E12.5)          '/
C
C     FORMATS
C
 1000 FORMAT('1',40X,'P R O G R A M     P R E R I C E 3',7X,
     1'(VERSION FOR VAX 11/785)'//21X,
     2'INTERACTIVE UTILITY CODE FOR PREPARING THE PARAMETER DATA',1X,
     3'TO RICE3'///1X,'FILES : PRINT OUTPUT',18X,
     4': FILE PRERICE.PRT'/9X,
     5'PARAMETER DATA OUTPUT',9X,': FILE RICE.IN'/9X,
     6'REFERENCE PSD DATA (OPTION)    : FILE RICE0^^.PSD'///)
 1001 FORMAT(1X,'OPTION FOR REFERENCE PSD DATA NPSD =',I3//1X,
     1'PARAMETER DATA GIVEN AS INPUT TO RICE3 :'//11X,
     2'RUN',5X,'=',1X,A//11X,'PSDIN',3X,'=',1X,A//11X,
     3'PFORM',3X,'=',1X,A//11X,'PRMIN',3X,'=',1PE15.4//11X,
     4'NT',6X,'=',I6//11X,'NS',6X,'=',I6//11X,
     5'IOPT',4X,'=',I6//11X,'IOUT',4X,'=',I6//11X,
     6'SFR',5X,'=',E15.4//11X,'ISEED',3X,'=',I10)
 1002 FORMAT(///1X,'E N D')
 2000 FORMAT(A/A/A/1PE10.3/I4/I5/I1/I1/E10.3/I9)
 5000 FORMAT(A)
 5001 FORMAT(I2)
```

```
 5002 FORMAT(E10.3)
 5003 FORMAT(I4)
 5004 FORMAT(I5)
 5005 FORMAT(I1)
 5006 FORMAT(I9)
 6000 FORMAT('1PROGRAM PRERICE3',9X,'(VERSION FOR VAX 11/785)'//1X,
     1'THE CODE IS AN INTERACTIVE UTILITY PROGRAMME FOR PREPARING THE'
     2/' PARAMETER DATA INPUT TO THE GAUSSIAN RANDOM NOISE GENERATION'
     3/' CODE RICE3. THE GIVEN PARAMETER DATA ARE WRITTEN ON THE FILE'
     4/' RICE.IN ACCORDING TO THE REQUIRED FORMAT SPECIFICATIONS. FOR'
     5/' A CROSS-CHECK THEY ARE ALSO AVAILABLE ON THE PRINT FILE',1X,
     6'PRERICE.PRT.'/1X,
     7'THE CODE CONTAINS THE OPTION OF GENERATING THE DATA OF BUILT-IN'
     8/' REFERENCE PSD FUNCTIONS. THESE DATA ARE WRITTEN ON THE FILE'
     9/' RICE0^^.PSD WHICH IS NUMBERED BY UP TO TWO DIGITS'
     A/' CORREPONDING TO THE SELECTED REFERENCE PSD NUMBER.'/)
 6001 FORMAT(/1X,'RUN DENOTATION (FOR RICE3) ? (MAX.40 CH.)'/1X,
     14('^^^^^^^^^^*'))
 6002 FORMAT(1X,'YOU SPECIFIED :'/1X,A)
 6003 FORMAT('$CORRECT ? (Y/N) ')
 6004 FORMAT(/1X,'REFERENCE PSD NUMBER NPSD ? (I2,(0,',I2,'))'/1X,
     1'(FOR NPSD=0, THE REFERENCE PSD DATA ARE ASSUMED TO BE EXTERNAL.)'
     2/1X,'^^')
 6005 FORMAT(1X,'YOU SPECIFIED :'/1X,I2)
 6006 FORMAT(/1X,'FILE PSDIN',8X,':',1X,A/1X,
     1'DATA FORMAT PFORM :',1X,A)
 6007 FORMAT(/1X,'FILE PSDIN ? (MAX.40 CH.)'/1X,4('^^^^^^^^^^*'))
 6008 FORMAT(/1X,'DATA FORMAT PFORM ? (MAX.20 CH.)'/1X,
     1'(E- OR F-FORMAT SPECIFICATION INCL.(...))'/1X,2('^^^^^^^^^^*'))
 6009 FORMAT(/1X,'REL. MINIMUM PRMIN (0/0) ? (E10.3,(1.E-4.LT.100.))'/
     11X,'^^.^^^E^^^')
 6010 FORMAT(1X,'YOU SPECIFIED :'/1X,1PE10.3)
 6011 FORMAT(/1X,'TRANSFORM SIZE NT ? (I4,(32,8192))'/1X,'^^^^')
 6012 FORMAT(1X,'YOU SPECIFIED :'/1X,I4)
 6013 FORMAT(1X,'INCORRECT VALUE, REPEAT !')
 6014 FORMAT(/1X,'NUMBER OF SEQUENCES NS ?',1X,
     1'(I5,(1,32000) DEPENDING ON THE VALUE OF NT)'/1X,'^^^^^')
 6015 FORMAT(1X,'YOU SPECIFIED :'/1X,I5)
 6016 FORMAT(/1X,'OPTION OF NOISE DATA GENERATION IOPT ? (I1,(0,',I1,
     1'))'/1X,'^')
 6017 FORMAT(1X,'YOU SPECIFIED :'/1X,I1)
 6018 FORMAT(/1X,'OPTION OF NOISE DATA OUTPUT IOUT ? (I1,(0,2))'/1X,'^')
 6019 FORMAT(/1X,'SAMPLING FREQUENCY SFR (HZ) ? (E10.3)'/1X,
     1'^^.^^^E^^^')
 6020 FORMAT(/1X,'INITIAL SEED NUMBER ISEED ? (I9,(0,900000000))'/1X,
     1'^^^^^^^^^')
 6021 FORMAT(1X,'YOU SPECIFIED :'/1X,I9)
 6022 FORMAT(//1X,'COMPUTATION OF THE REFERENCE PSD DATA STARTS !')
C
C     PARAMETER DATA INPUT
C
      OPEN(UNIT=1,FILE='PRERICE.PRT',STATUS='NEW')
      OPEN(UNIT=2,FILE='RICE.IN',STATUS='NEW')
      WRITE(6,6000)
      WRITE(1,1000)
  100 WRITE(6,6001)
      READ(5,5000) RUN
      WRITE(6,6002) RUN
```

```
      WRITE(6,6003)
      READ(5,5000) RY
      IF(RY.NE.LY) GO TO 100
  105 WRITE(6,6004) NPSDS
      READ(5,5001,ERR=105) NPSD
      IF(NPSD.LT.0) NPSD=0
      IF(NPSD.GT.NPSDS) NPSD=NPSDS
      WRITE(6,6005) NPSD
      WRITE(6,6003)
      READ(5,5000) RY
      IF(RY.NE.LY) GO TO 105
      IF(NPSD.EQ.0) GO TO 110
      ENCODE(2,5001,PSDIN(6:7)) NPSD
      WRITE(6,6006) PSDIN,PFORM
      OPEN(UNIT=3,FILE=PSDIN,STATUS='NEW')
      GO TO 120
  110 WRITE(6,6007)
      READ(5,5000) PSDIN
      WRITE(6,6002) PSDIN
      WRITE(6,6003)
      READ(5,5000) RY
      IF(RY.NE.LY) GO TO 110
  115 WRITE(6,6008)
      READ(5,5000) PFORM
      WRITE(6,6002) PFORM
      WRITE(6,6003)
      READ(5,5000) RY
      IF(RY.NE.LY) GO TO 115
  120 WRITE(6,6009)
      READ(5,5002,ERR=120) PRMIN
      IF(PRMIN.LT.1.E-04.OR.PRMIN.GE.100) PRMIN=1.E-02
      WRITE(6,6010) PRMIN
      WRITE(6,6003)
      READ(5,5000) RY
      IF(RY.NE.LY) GO TO 120
  125 WRITE(6,6011)
      READ(5,5003,ERR=125) NT
      DO 1 N=5,13
      IF(NT.EQ.2**N) GO TO 130
    1 CONTINUE
      WRITE(6,6013)
      GO TO 125
  130 WRITE(6,6012) NT
      WRITE(6,6003)
      READ(5,5000) RY
      IF(RY.NE.LY) GO TO 125
  135 WRITE(6,6014)
      READ(5,5004,ERR=135) NS
      IF(NS.LT.1) NS=1
      N = 1024000/NT
      IF(NS.GT.N) NS=N
      WRITE(6,6015) NS
      WRITE(6,6003)
      READ(5,5000) RY
      IF(RY.NE.LY) GO TO 135
  140 WRITE(6,6016) IOPTS
      READ(5,5005,ERR=140) IOPT
```

```
      IF(IOPT.LT.0) IOPT=0
      IF(IOPT.GT.IOPTS) IOPT=IOPTS
      WRITE(6,6017) IOPT
      WRITE(6,6003)
      READ(5,5000) RY
      IF(RY.NE.LY) GO TO 140
  145 WRITE(6,6018)
      READ(5,5005,ERR=145) IOUT
      IF(IOUT.LT.0) IOUT=0
      IF(IOUT.GT.2) IOUT=2
      WRITE(6,6017) IOUT
      WRITE(6,6003)
      READ(5,5000) RY
      IF(RY.NE.LY) GO TO 145
  150 WRITE(6,6019)
      READ(5,5002,ERR=150) SFR
      IF(SFR.LE.0.) GO TO 155
      WRITE(6,6010) SFR
      WRITE(6,6003)
      READ(5,5000) RY
      IF(RY.NE.LY) GO TO 150
      GO TO 160
  155 WRITE(6,6013)
      GO TO 150
  160 WRITE(6,6020)
      READ(5,5006,ERR=160) ISEED
      IF(ISEED.LT.0) ISEED=0
      IF(ISEED.GT.900000000) ISEED=900000000
      WRITE(6,6021) ISEED
      WRITE(6,6003)
      READ(5,5000) RY
      IF(RY.NE.LY) GO TO 160
C
C     PARAMETER DATA OUTPUT
C
      WRITE(1,1001) NPSD,RUN,PSDIN,PFORM,PRMIN,NT,NS,IOPT,IOUT,SFR,ISEED
      WRITE(2,2000) RUN,PSDIN,PFORM,PRMIN,NT,NS,IOPT,IOUT,SFR,ISEED
      IF(NPSD.EQ.0) GO TO 405
C
C     COMPUTATION OF REFERENCE PSD DATA AND OUTPUT
C
      CLOSE(UNIT=2,STATUS='KEEP')
      WRITE(6,6022)
      PFORM = '(1X,I4,1PE12.5)'
      NTH = NT/2
      K(1) = 2
      K(2) = NTH
      A(1) = 100.
      DO 2 N=1,NTH+1
    2 P(N) = 0.
      GO TO (301,202,203,204,205,206,207,208,209,210,211,212,
     1213,214,215,216) NPSD
  202 K(1) = NTH/4+1
      K(2) = K(1)+NTH/2
      GO TO 301
  203 K(1) = NTH/2+1
      K(2) = K(1)
      GO TO 301
```

```
  204 K(1) = NTH/2+2
      K(2) = K(1)
  301 CALL PSD1
      GO TO 400
  205 CALL PSD2
      GO TO 400
  206 A(2) = 10.
      GO TO 302
  207 A(2) = 100.
      GO TO 302
  208 A(2) = 1.E+04
  302 CALL PSD3
      GO TO 400
  209 A(2) = 2.*PI/FLOAT(NTH)
      CALL PSD4
      GO TO 400
  210 K(3) = NTH/2
      A(2) = 0.05
      GO TO 303
  211 K(3) = NTH/2
      A(2) = 0.2
  303 CALL PSD5
      GO TO 400
  212 A(1) = 2.46842E-03*A(1)
      A(2) = 0.0025
      A(3) = 0.004
      A(4) = 0.15625
      A(5) = 0.59375
      CALL PSD6
      GO TO 400
  213 K(2) = 51
      GO TO 301
  214 K(2) = 41
      GO TO 301
  215 K(2) = 9
      GO TO 301
  216 K(2) = K(1)
      GO TO 301
C
  400 WRITE(3,PFORM) (N-1,P(N),N=1,NTH+1)
  405 WRITE(1,1002)
      STOP
      END
C==============================================
      SUBROUTINE PSD1
C
C     RECTANGULAR REFERENCE PSD.
C
      COMMON A(10),P(4097),NTH,K(10)
      A1 = A(1)
      DO 1 N=K(1),K(2)
    1 P(N) = A1
      RETURN
      END
C==============================================
      SUBROUTINE PSD2
C
```

```
C     LINEARLY DECAYING REFERENCE PSD.
C
      COMMON A(10),P(4097),NTH,K(10)
      A1 = A(1)
      A2 = 1./FLOAT(NTH-1)
      DO 1 N=K(1),K(2)
    1 P(N) = A1*(1.-A2*FLOAT(N-2))
      RETURN
      END
C===============================================
      SUBROUTINE PSD3
C
C     EXPONENTIALLY DECAYING REFERENCE PSD.
C
      COMMON A(10),P(4097),NTH,K(10)
      A1 = A(1)
      A2 = ALOG(A(2))/FLOAT(NTH-2)
      DO 1 N=K(1),K(2)
    1 P(N) = A1*EXP(-A2*FLOAT(N-2))
      RETURN
      END
C===============================================
      SUBROUTINE PSD4
C
C     SINE REFERENCE PSD.
C
      COMMON A(10),P(4097),NTH,K(10)
      A1 = 0.5*A(1)
      A2 = A(2)
      DO 1 N=K(1),K(2)
    1 P(N) = A1*(1.-COS(A2*FLOAT(N-1)))
      RETURN
      END
C===============================================
      SUBROUTINE PSD5
C
C     RESONANCE REFERENCE PSD.
C
      COMMON A(10),P(4097),NTH,K(10)
      A2 = 4.*A(2)**2
      A1 = A(1)*A2
      A3 = 1./FLOAT(K(3))
      DO 1 N=K(1),K(2)
      X = (A3*FLOAT(N-1))**2
    1 P(N) = A1/((1.-X)**2+A2*X)
      RETURN
      END
C===============================================
      SUBROUTINE PSD6
C
C     REFERENCE PSD SIMULATING TWO BROAD PEAKS.
C
      COMMON A(10),P(4097),NTH,K(10)
      A1 = A(1)
      A2 = A(2)
      A3 = A(3)
      A4 = A(4)
      A5 = A(5)
```

```
      DO 1 N=K(1),K(2)
      X = FLOAT(N-1)/FLOAT(NTH)
    1 P(N) = A1/(A2+(X-A4)**2)+A1/(A3+(X-A5)**2)
      RETURN
      END
```

# LISTING

## RICE3

```
      PROGRAM RICE3
C
C     THE CODE RICE3 GENERATES A STATIONARY GAUSSIAN RANDOM NOISE
C     RECORD ACCORDING TO A GIVEN REFERENCE PSD. THE DATA OF THE
C     REFERENCE PSD MUST BE STORED ON A SEPARATE FILE. THE NOISE
C     GENERATION PROCEDURE IS BASED ON THE RICE FORMULA AND ITS
C     MODIFICATION DEVELOPED BY K.BEHRINGER,H.NISHIHARA AND
C     G.SPIEKERMAN, ANN.NUCL.ENERGY 13,1986,443 (REPRINT IN EIR-
C     REPORT 601). THE CODE CONTAINS SEVERAL OPTIONAL PROCEDURES
C     FOR THE NOISE GENERATION.
C
C     THE ORIGINAL CODE RICE1 HAS BEEN WRITTEN BY K.BEHRINGER IN
C     JANUARY 1985 FOR THE OPERATION ON THE CDC-6500. THE PRESENT
C     VERSION, RICE2,IS A COMPLETE REVISION OF RICE1 MADE BY
C     K.BEHRINGER IN APRIL 1989. IT HAS BEEN ADAPTED FOR BATCH
C     OPERATION ON THE VAX 11/785.
C     REWRITTEN IN APRIL 1996 FOR OPERATION ON THE
C     DEC-ALPHA-2100 COMPUTER.
C     EXTENDED VERSION OF RICE2 FOR A TRANSFORM SIZE OF 8192 DATA
C     POINTS, SEPTEMBER 1996
C
C     AN INTERACTIVE CODE, PRERICE3, WILL BE AVAILABLE FOR PREPARING
C     THE PARAMETER DATA INPUT.
C
C     LINK COMMAND : IMPORT IMSL
C                    LINK RICE3
C
      CHARACTER RUN*40,PSDIN*40,PFORM*20
      REAL*8 PES
      COMMON PES(4095),WFFTR(16399),X(8192),XS(8192),F(8192),
     1GRNX(4095),GRNY(4095),WEIGHT(4095),NRF(4095)
      EXTERNAL WINDOWA,WINDOWB
      DATA IOPTS,DT,CFR,DF/5,3*0./
C
C     FORMATS
C
 1000 FORMAT(A/A/A/E10.3/I4/I5/I1/I1/E10.3/I9)
 7000 FORMAT('1',40X,'P R O G R A M     R I C E 3',13X,
     1'(VERSION FOR VAX 11/785)'//26X,
     2'GENERATION OF GAUSSIAN RANDOM NOISE BY THE RICE FORMULA'///1X,
     3'FILES : PARAMETER DATA INPUT',9X,': FILE RICE.IN'/9X,
     4'REFERENCE PSD DATA',11X,': FILE ''PSDIN'''/9X,
     5'PRINT OUTPUT - PARAMETER',5X,': FILE RICE.PR1'/9X,
     6'PRINT OUTPUT - REFERENCE PSD : FILE RICE.PR2'/9X,
     7'PRINT OUTPUT - ESTIMATED PSD : FILE RICE.PR3'/9X,
     8'NOISE DATA',19X,': FILE RICE.DAT'//)
 7001 FORMAT(1X,'RUN DENOTATION',26X,'RUN',5X,'=',1X,A//1X,
     1'FILE REFERENCE PSD DATA',17X,'PSDIN',3X,'=',1X,A//1X,
     2'FORMAT REFERENCE PSD DATA',15X,'PFORM',3X,'=',1X,A//1X,
     3'REL. MINIMUM REFERENCE PSD DATA',9X,'PRMIN',3X,'=',1PE15.4/11X,
     4'(0/0 OF MAX.AMPLITUDE)'//1X,
     5'TRANSFORM SIZE',26X,'NT',6X,'=',I6//1X,
     6'NUMBER OF SEQUENCES TO BE GENERATED',5X,'NS',6X,'=',I6//1X,
     7'OPTION OF GENERATION',20X,'IOPT',4X,'=',I6//1X,
     8'OPTION OF NOISE DATA OUTPUT',13X,'IOUT',4X,'=',I6//1X,
     9'SAMPLING FREQUENCY (HZ)',17X,'SFR',5X,'=',E15.4/1X,
     A'SAMPLING INTERVAL (SEC)',17X,'DT',6X,'=',E15.4/1X,
     B'NYQUIST CUTOFF FREQUENCY (HZ)',11X,'CFR',5X,'=',E15.4/1X,
     C'NYQUIST COINTERVAL (HZ)',17X,'DF',6X,'=',E15.4//1X,
     D'INITIAL SEED NUMBER',21X,'ISEED',3X,'=',I10)
 7002 FORMAT(//1X,'SIGNAL VARIANCE : FROM ESTIMATED PSD',4X,
     1'VARE',4X,'=',1PE15.4/17X,': FROM GIVEN PSD',8X,
     2'VARG',4X,'=',E15.4)
 7003 FORMAT(///1X,'E N D')
 7005 FORMAT(//11X,'STOP, READ-ERROR OR EOF MARK AT FILE RICE.IN !')
 7010 FORMAT(//11X,'STOP, INCORRECT INPUT PARAMETER VALUES !')
 7015 FORMAT(//11X,'STOP, FILE ''PSDIN'' CANNOT BE OPENED !')
```

```
 7020 FORMAT(//11X,'STOP, FILE RICE.DAT CANNOT BE OPENED !')
 7025 FORMAT(//11X,'STOP, FILE RICE.PR3 CANNOT BE OPENED !')
 9000 FORMAT(('1 NP0',3X,'NFR',6X,'F (HZ)',5X,'P ESTIMATED',5X,'P GIVEN'
     1,6X,'DEV (0/0)',1P/50(/1X,I4,I6,4E14.5)/))
10000 FORMAT(1X,1PE12.5)
10001 FORMAT(1X,I7,1PE12.5)
C
C     INPUT PARAMETER DATA
C
      OPEN(UNIT=1,FILE='RICE.IN',STATUS='OLD',ERR=100,
     1DEFAULTFILE='DIRINPUT')
      OPEN(UNIT=7,FILE='RICE.PR1',STATUS='NEW',ERR=105,
     1DEFAULTFILE='DIRINPUT')
      WRITE(7,7000)
      READ(1,1000,ERR=110,END=110) RUN,PSDIN,PFORM,PRMIN,NT,NS,IOPT,
     1IOUT,SFR,ISEED
      IF(PRMIN.LT.1.E-04.OR.PRMIN.GE.100.) PRMIN=1.E-02
      IF(NT.LT.32) NT=32
      IF(NT.GT.8192) NT=8192
      IF(NS.LT.1) NS=1
      N = 1024000/NT
      IF(NS.GT.N) NS=N
      IF(IOPT.LT.0) IOPT=0
      IF(IOPT.GT.IOPTS) IOPT=IOPTS
      IF(IOUT.LT.0) IOUT=0
      IF(IOUT.GT.2) IOUT=2
      IF(SFR.LE.0.) GO TO 115
      DT = 1./SFR
      CFR = 0.5*SFR
      DF = SFR/FLOAT(NT)
  115 IF(ISEED.LT.0) ISEED=0
      IF(ISEED.GT.900000000) ISEED=900000000
      WRITE(7,7001) RUN,PSDIN,PFORM,PRMIN,NT,NS,IOPT,IOUT,SFR,DT,CFR,
     1DF,ISEED
      IF(SFR.LE.0.) GO TO 120
      DO 1 N=5,13
      IF(NT.EQ.2**N) GO TO 200
    1 CONTINUE
      GO TO 120
C
C     INPUT REFERENCE PSD
C
  200 OPEN(UNIT=2,FILE=PSDIN,STATUS='OLD',ERR=205,
     1DEFAULTFILE='DIRINPUT')
      PRMIN = 0.01*PRMIN
      NP = NT/2+1
      CALL PRRICE2(NP,DF,PRMIN,PFORM,NP0)
      CLOSE(UNIT=1,STATUS='KEEP')
      CLOSE(UNIT=2,STATUS='KEEP')
C
C     NOISE DATA GENERATION
C
      IF(IOUT.EQ.2) GO TO 300
      OPEN(UNIT=10,FILE='RICE.DAT',STATUS='NEW',ERR=305,
     1DEFAULTFILE='DIRINPUT')
      GO TO 310
  300 OPEN(UNIT=10,FILE='RICE.DAT',STATUS='NEW',FORM='UNFORMATTED',
     1ERR=305,DEFAULTFILE='DIRINPUT')
  310 OPEN(UNIT=9,FILE='RICE.PR3',STATUS='NEW',ERR=315,
     1DEFAULTFILE='DIRINPUT')
      DO 2 N=1,NP0
    2 PES(N) = 0.D0
      IOPT = IOPT+1
      CALL RNSET(ISEED)
      DO 3 I=1,NS
      GO TO (330,331,332,332,333,333) IOPT
  330 CALL OPT0(NT,NP0)
      GO TO 340
  331 CALL OPT1(NT,NP0)
      GO TO 340
  332 CALL OPT2(NT,NP0,IOPT,I,NS,WINDOWA)
      GO TO 340
```

```
  333 CALL OPT2(NT,NP0,IOPT,I,NS,WINDOWB)      .
  340 IF(IOUT-1) 345,350,355
  345 WRITE(10,10000)  (X(N),N=1,NT)
      GO TO 3
  350 N1 = NT*(I-1)
      DO 4 N=1,NT
      N2 = N1+N
    4 WRITE(10,10001) N2,X(N)
      GO TO 3
  355 DO N=1,NT
      WRITE(10) X(N)
      END DO
    3 CONTINUE
      CLOSE(UNIT=10,STATUS='KEEP')
C
C     OUTPUT OF PSD DATA AND SIGNAL VARIANCE
C
      VARG = 0.5*DT*FLOAT(NT)
      VARE = 0.5*VARG/FLOAT(NS)
      DO 5 N=1,NP0
      WEIGHT(N) = VARG*WEIGHT(N)**2
      PES(N) = DBLE(VARE)*PES(N)
      F(N) = DF*FLOAT(NRF(N))
    5 X(N) = 100.*(SNGL(PES(N))/WEIGHT(N)-1.)
      WRITE(9,9000) (N,NRF(N),F(N),SNGL(PES(N)),WEIGHT(N),X(N),N=1,NP0)
      WRITE(9,7003)
      CLOSE(UNIT=9,STATUS='KEEP')
      VARG = 0.
      VARE = 0.
      DO 6 N=1,NP0
      VARG = VARG+WEIGHT(N)
    6 VARE = VARE+SNGL(PES(N))
      DF = 2.*DF
      VARG = DF*VARG
      VARE = DF*VARE
      WRITE(7,7002) VARE,VARG
      WRITE(7,7003)
      STOP
C
C     NUMBERED STOPS
C
  100 STOP 100
  105 STOP 101
  110 WRITE(7,7005)
      STOP 102
  120 WRITE(7,7010)
      STOP 103
  205 WRITE(7,7015)
      STOP 104
  305 WRITE(7,7020)
      STOP 109
  315 WRITE(7,7025)
      STOP 110
      END
C==============================================
      SUBROUTINE PRRICE2(NP,DF,PRMIN,PFORM,NP0)
C
C     INPUT OF THE REFERENCE PSD AND COMPUTATIONS OF THE WEIGHTS.
C
      CHARACTER PFORM*20
      REAL*8 PES
      COMMON PES(4095),WFFTR(16399),X(8192),XS(8192),COEF(8192),
     1GRNX(4095),GRNY(4095),WEIGHT(4095),NRF(4095)
C
C     FORMATS
C
 7030 FORMAT(//11X,'STOP, READ-ERROR AT FILE ''PSDIN'' !')
 7035 FORMAT(//11X,'STOP, EOF MARK AT FILE ''PSDIN'' !')
 7040 FORMAT(//11X,'STOP, REFERENCE PSD DOES NOT CONTAIN POSITIVE',
     1' DATA !')
 7045 FORMAT(//11X,'STOP, FILE RICE.PR2 CANNOT BE OPENED !')
 7050 FORMAT(//11X,'STOP, REFERENCE PSD CONTAINS',I4,
     1' NEGATIVE DATA !')
```

```
 8000 FORMAT('1',2X,'N',7X,'F (HZ)',10X,'P',12X,'PN',10X,'NP0',3X,
     1'NRF'/)
 8001 FORMAT(1X,1P,I4,3E14.5,2X,2I6)
 8005 FORMAT(///1X,'E N D')
C
C     DATA INPUT
C
      READ(2,PFORM,ERR=100,END=105) (X(N),N=1,NP)
C
C     DATA CHECK
C
      X(1) = 0.
      X(NP) = 0.
      PMAX = 0.
      NP0 = NP-1
      L = 0
      DO 1 N=2,NP0
      P = X(N)
      IF(P.LT.0.) L=L+1
    1 PMAX = AMAX1(P,PMAX)
      IF(PMAX.LE.0.) GO TO 115
      OPEN(UNIT=8,FILE='RICE.PR2',STATUS='NEW',ERR=120,
     1DEFAULTFILE='DIRINPUT')
      NP0 = 0
      SP = 0.
      LP = 0
      DO 2 N=1,NP
      IF(N.LE.LP) GO TO 125
      WRITE(8,8000)
      LP = LP+50
  125 NF = N-1
      F = DF*FLOAT(NF)
      P = X(N)
      PN = P/PMAX
      IF(PN.LT.PRMIN) GO TO 130
      NP0 = NP0+1
      NRF(NP0) = NF
      WRITE(8,8001) N,F,P,PN,NP0,NF
      SP = SP+PN
      WEIGHT(NP0) = PN
      GO TO 2
  130 WRITE(8,8001) N,F,P,PN
    2 CONTINUE
      WRITE(8,8005)
      IF(L.GT.0) GO TO 135
      CLOSE(UNIT=8,STATUS='KEEP')
C
C     COMPUTATION OF THE WEIGHTS
C
      SP = 1./SP
      DO 3 N=1,NP0
    3 WEIGHT(N) = SQRT(SP*WEIGHT(N))
      RETURN
C
C     NUMBERED STOPS
C
  100 WRITE(7,7030)
      GO TO 110
  105 WRITE(7,7035)
  110 STOP 105
  115 WRITE(7,7040)
      STOP 106
  120 WRITE(7,7045)
      STOP 107
  135 WRITE(7,7050) L
      STOP 108
      END
C==============================================
      SUBROUTINE OPT0(NT,NP0)
C
C     NOISE GENERATION DIRECTLY BY THE RICE FORMULA.
C
```

```
      REAL*8 PES
      COMMON PES(4095),WFFTR(16399),X(8192),XS(8192),ARG(8192),
     1GRNX(4095),GRNY(4095),WEIGHT(4095),NRF(4095)
      SAVE I
      DATA I,PI/0,3.1415927/
      IF(I.GT.0) GO TO 100
      A = 2.*PI/FLOAT(NT)
      DO 1 K=1,NP0
    1 ARG(K) = A*FLOAT(NRF(K))
      I = 1
  100 CALL RNNOA(NP0,GRNX)
      CALL RNNOA(NP0,GRNY)
      DO 2 K=1,NP0
      W = WEIGHT(K)
      A = W*GRNX(K)
      B = W*GRNY(K)
      GRNX(K) = A
      GRNY(K) = B
    2 PES(K) = PES(K)+DBLE(A**2+B**2)
      DO 3 N=1,NT
      A = 0.
      B = FLOAT(N-1)
      DO 4 K=1,NP0
      W = B*ARG(K)
    4 A = A+GRNX(K)*COS(W)+GRNY(K)*SIN(W)
    3 X(N) = A
      RETURN
      END
C===============================================
      SUBROUTINE OPT1(NT,NP0)
C
C     NOISE GENERATION VIA FFT TECHNIQUES.
C
      REAL*8 PES
      COMMON PES(4095),WFFTR(16399),X(8192),XS(8192),COEF(8192),
     1GRNX(4095),GRNY(4095),WEIGHT(4095),NRF(4095)
      SAVE I
      DATA I/0/
      IF(I.GT.0) GO TO 100
      DO 1 K=1,NT
    1 COEF(K) = 0.
      CALL FFTRI(NT,WFFTR)
      I = 1
  100 CALL RNNOA(NP0,GRNX)
      CALL RNNOA(NP0,GRNY)
      DO 2 K=1,NP0
      W = WEIGHT(K)
      A = W*GRNX(K)
      B = W*GRNY(K)
      GRNX(K) = A
      GRNY(K) = B
      PES(K) = PES(K)+DBLE(A**2+B**2)
      N = NRF(K)+1
      COEF(2*(N-1)) = 0.5*A
    2 COEF(2*N-1) = -0.5*B
      CALL F2TRB(NT,COEF,X,WFFTR)
      RETURN
      END
C===============================================
      SUBROUTINE OPT2(NT,NP0,IOPT,IS,NS,WSUB)
C
C     A TWO-SEQUENCE WINDOW IS APPLIED TO THE GENERATED NOISE DATA.
C
      REAL*8 PES
      COMMON PES(4095),WFFTR(16399),X(8192),XS(8192),COEF(8192),
     1GRNX(4095),GRNY(4095),WEIGHT(4095),NRF(4095)
      SAVE I
      DATA I/0/
  100 IF(I) 105,106,107
  106 IF(2*(IOPT/2).NE.IOPT) GO TO 120
      I = 1
      CALL OPT1(NT,NP0)
      GO TO 110
```

```
  120 I = -1
      CALL OPT0(NT,NP0)
  110 DO 1, N=1,NT
    1 XS(N) = X(N)
      DO 2 N=1,NP0
    2 PES(N) = 0.5D0*PES(N)
      GO TO 100
  105 CALL OPT0(NT,NP0)
      GO TO 115
  107 CALL OPT1(NT,NP0)
  115 DO 3 N=1,NT
      CALL WSUB(N,NT,W1,W2)
      X1 = XS(N)
      X2 = X(N)
      XS(N) = X2
    3 X(N) = X1*W1+X2*W2
      IF(IS.LT.NS) RETURN
      DO 4 N=1,NP0
    4 PES(N) = PES(N)-DBLE(0.5*(GRNX(N)**2+GRNY(N)**2))
      RETURN
      END
C=================================================
      SUBROUTINE WINDOWA(N,NT,W1,W2)
C
C     TWO-SEQUENCE COSINE WINDOW.
C
      SAVE PIH
      DATA PIH/1.5707963/
      W1 = PIH*FLOAT(N-1)/FLOAT(NT)
      W2 = SIN(W1)
      W1 = COS(W1)
      RETURN
      END
C=================================================
      SUBROUTINE WINDOWB(N,NT,W1,W2)
C
C     TWO-SEQUENCE SQUARE ROOT WINDOW.
C
      W1 = FLOAT(N-1)/FLOAT(NT)
      W2 = SQRT(W1)
      W1 = SQRT(1.-W1)
      RETURN
      END
```

## 15. FINAL REMARKS

The method and the codes presented for determining the oscillation frequency and the decay ratio in BWR stability analyses are based on a transparent physical procedure with simple phenomenological models. The oscillator model B has been preferred and found to be superior to the oscillator model A. The analysis method is off-line. It is presently still far away from on-line application.

The investigations will be a contribution to a problem. The benchmark results reveal that presently, in the field of signal analysis, many different methodologies are used and the uncertainties of the various approaches are in some cases very different. The trivial question what is the best or more reliable method for measuring the decay ratio has not been answered by the benchmark project. The documentation of the codes is thought for further development.

The criterion used for finding the optimum fit range is empirical and may exhibit one possibility. Further investigations and ideas are desirable.

The FORTRAN codes described in this report are available for interested users. Requests should be directed by e-mail to DIETER.HENNIG∂PSI.CH, or by a letter to the author. The most important codes (CPSDES3, FFT2, ACCF2, ACFIT7A and ACFITEV5) have been brought as a package into the MATLAB environment (UNIX operating system).

**Acknowledgements**

The author would like to thank Dr.A.Pritzker, PSI, for permission to use the PSI DEC computer system. He also thanks Dr.D.Hennig, PSI, for many interesting discussions and for making available the benchmark signal data. Thanks are also to Mr.B.Askari (summer student at PSI) for transferring codes into the MATLAB environment. Finally, the author is indebted to Dr.T.Dury, PSI, for proof-reading the English text.