96-80-7294
MOTC-IOT-95-H2DA003

訊號分析法比較研究暨其於水波應用探討 — **Asyst** 仔波程式集

交 通 部 運 輸 研 究 所

中華民國 96 年 5 月

96-80-7294
MOTC-IOT-95-H2DA003

訊號分析法比較研究暨其於水波應用探討 — **Asyst** 仔波程式集

著 者：李勇榮

交 通 部 運 輸 研 究 所
中華民國 96 年 5 月

訊號分析法比較研究暨其於水波應用探討- Asyst 仔波程式集

著　　者：李勇榮
出版機關：交通部運輸研究所
地　　址：台北市敦化北路 240 號
網　　址：www.ihmt.gov.tw (中文版>中心出版品)
電　　話：(04) 26587176
出版年月：中華民國 96 年 5 月
印 刷 者：
版(刷)次冊數：初版一刷 110 冊
本書同時登載於交通部運輸研究所網站
定　　價：600 元
展 售 處：
交通部運輸研究所運輸資訊組• 電話：(02)23496880
國家書坊台視總店：台北市八德路 3 段 10 號 B1• 電話：(02)25781515
五南文化廣場：臺中市中山路 2 號 B1•電話：(04)22260330

GPN：1009601248　　　ISBN：978-986-00-9801-3 (平裝)

96 訊號分析法比較研究暨其於水波應用探討 - Asyst 仔波程式集

交通部運輸研究

GPN： 1009601248
定價 600 元

交通部運輸研究所出版品摘要表

出版品名稱：**訊號分析法比較研究暨其於水波應用探討 —Asyst 仔波程式集**

國際標準書號（或叢刊號）	政府出版品統一編號	運輸研究所出版品編號	計畫編號
ISBN:978-986-00-9801-3 (平裝)	1009601248	96-80-7294	95-H2DA003

主辦單位：港灣技術研究中心
主管：邱永芳
計劃主持人：李勇榮
研究人員：李勇榮
聯絡電話：04-26587183
傳真號碼：04-26587183、 04-26571329

研究期間
自 95 年 01 月
至 95 年 12 月

關鍵詞：廣泛之仔波分析程式、Asyst 程式語言及補助軟件集、多類屬之離散仔波、連續仔波、正反轉換、爆展、熵值、常標值、最適函基、最適階函基、特性函數相位、共關協振、調適化時頻窗連續仔波轉換、等。

摘要：

此處吾人以 Asyst 程式語言，由根基起建，開發一套含蓋範圍相當廣泛之仔波分析程式。另亦特別著重資料之處理、檢視、呈現等功能。此外尚從事工作平台之開發、圖表文書管理系統建置，此乃藉由整合一些外加程式及補助功能，撰寫各類程式碼，如： Postfix 語言、 Lotus 報表介面、 Mathematica 程式語言、 WinEDT 巨集程式語言、LaTeX 程式及軟件集、等。所開發仔波相關程式功能包含多類屬之離散仔波與連續仔波，暨其諸多特性探討，如：正反轉換、爆展、熵值、常標值、最適函基、最適階函基、特性函數相位、共關協振、調適化時頻窗連續仔波轉換、及一新型類仔波轉換暨其特性探討、等。

出版日期	頁數	定價	本 出 版 品 取 得 方 式
96 年 5 月	422	600	凡屬機密性出版品均不對外公開。普通性出版品，公營、公益機關團體及學校可函洽本所免費贈閱；私人及私營機關團體可按定價價購。

機密等級：
□密□機密 □極機密 □絕對機密
（解密【限】條件：□ 年 月 日解密，□公布後解密，□附件抽存後解密，
□工作完成或會議終了時解密，□另行檢討後辦理解密）
■普通

備註：本研究之結論與建議不代表交通部之意見。

PUBLICATION ABSTRACTS OF RESEARCH PROJECTS
INSTITUTE OF TRANSPORTATION
MINISTRY OF TRANSPORTATION AND COMMUNICATIONS

TITLE: **Time-Frequency Analyses and Their Applications - Wavelet Programming in Asyst**

ISBN(OR ISSN)	GOVERNMENT PUBLICATIONS NUMBER	IOT SERIAL NUMBER	PROJECT NUMBER
978-986-00-9801-3 (pbk)	1009601248	96-80-7294	95-H2DA003

DIVISION: HARBOR & MARINE TECHNOLOGY CENTER
DIVISION DIRECTOR: Yung-Fang Chiu
PRINCIPAL INVESTIGATOR: Yueon-Ron Lee
PROJECT STAFF: Yueon-Ron Lee
PHONE: 886-4-26587183
FAX: 886-4-26587183, 886-4-26571329

PROJECT PERIOD
FROM Jan. 2006
TO Dec. 2006

KEY WORDS: Comprehensive wavelet programs in Asyst, Programming add-ins and application auxiliaries, Discrete and the continuous wavelet transforms, Forward and inverse transforms, Wavelet and wavelet packet blowups, Entropy statistics, Norm distributions, Best basis and best level identifications, Phase of the characteristic function, Wavelet and spectral coherences, Adapted time-frequency windows, Wavelet variant.

ABSTRACT:

Comprehensive wavelet programs, as well as codes for proficient data processing and rendering, were developed from the ground up using the Asyst programming language.Programming add-ins and application auxiliaries are also integrated, notably, the Postfix language, spreadsheet interface, the Mathematica language, the WinEdt macro language, LaTeX codes and packages, workbench development, and figure and document management system, etc. The programs cover both the discrete and the continuous wavelet transforms and include many characterizations, such as forward and inverse transforms, wavelet and wavelet packet blowups, entropy statistics, norm distributions, best basis and best level identifications, phase of a characteristic function, wavelet and spectral coherences, continuous wavelet transform associated with adapted time-frequency windows, and a new wavelet variant and its characterizations, etc.

DATE OF PUBLICATION	NUMBER OF PAGES	PRICE	CLASSIFICATION
May 2007	422	600	☐RESTRICTED ☐CONFIDENTIAL ☐SECRET ☐TOP SECRET ▒UNCLASSIFIED

訊號分析法比較研究
暨其於水波應用探討 —
Asyst 仔波程式集

Time-Frequency Analyses and
Their Applications —
Wavelet Programming in Asyst

李 勇 榮

交通部運輸研究所港灣技術研究中心

Email: ronlee@ms4.hinet.net

Revision: 1.1, March 29, 2007, 12:51:36
Printed: March 29, 2007

LIST OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

摘要

此處吾人以Asyst程式語言，由根基起建，開發一套含蓋範圍相當廣泛之仔波分析程式。另亦特別著重資料之處理、檢視、呈現等功能。此外尚從事工作平台之開發、圖表文書管理系統建置，此乃藉由整合一些外加程式及補助功能，撰寫各類程式碼，如：Postfix語言、Lotus報表介面、Mathematica程式語言、WinEDT巨集程式語言、LaTeX程式及軟件集、等。所開發仔波相關程式功能包含多類屬之離散仔波與連續仔波，暨其諸多特性探討，如：正反轉換、爆展、熵值、常標值、最適函基、最適階函基、特性函數相位、共關協振、調適化時頻窗連續仔波轉換、及一新型類仔波轉換暨其特性探討、等。

ABSTRACT

Comprehensive wavelet programs, as well as codes for proficient data processing and rendering, were developed from the ground up using the Asyst programming language. Programming add-ins and application auxiliaries are also integrated, notably, the Postfix language, spreadsheet interface, the Mathematica language, the WinEdt macro language, LaTeX codes and packages, workbench development, and figure and document management system, etc. The programs cover both the discrete and the continuous wavelet transforms and include many characterizations, such as forward and inverse transforms, wavelet and wavelet packet blowups, entropy statistics, norm distributions, best basis and best level identifications, phase of a characteristic function, wavelet and spectral coherences, continuous wavelet transform associated with adapted time-frequency windows, and a new wavelet variant and its characterizations, etc.

Chapter 1

Introduction

Data analysis involves two main tasks: one concerns the theoretical aspect that focuses on the study of various analyzing function bases and their analytical properties; the other concerns the figurative display of physical data either from simulations or from observations. The latter is generally impossible without computer programming. For engineers they mostly are interested only in applications. And with the wide availability of bundled packages of numerical programs, fewer and fewer researchers and scholars are willing to take up the jobs of writing intrinsic codes that touch the analytical prospects of theories. In reality, and being my personal experience, it is much more painstaking and time-consuming to put theories into computer codes than to master the analytical aspects of theories. One simple and definitive explanation for this is that code writing is a higher order process that can only be achieved when its lower order process of theoretical mastering has been completed.

It is the author's wish that the programs here should provide a very comprehensive coverage of various wavelet basis categories and should also be capable of exploring many related characterizations of wavelet functions. Much attention is pay to execution automation and flexibility. And user-friendliness is also one of the concerns.

Basically all the wavelet programs and most of the relevant data processing were developed from the ground up using the Asyst programming language. It is our belief that any keyboard input or manual interruption be minimized. A principle is that, except inputting parameters, there should be absolutely no intermittent interferences from raw data inputs to final outcome renditions.

To achieve such goal, several programming add-ins or application auxiliaries are integrated; notably, they include:

- The Postfix language — The Postfix add-in for the Asyst programming enables the generation of high quality Encapsulated Postscript figures directly form the core programs. With proper labeling and legends, excellent quality figures as many as one wishes can be generated with mistakes avoided and correctness guaranteed. More importantly, especially during testings or parameter alterations or case variations, one is able to eliminate the painful task of plotting seemly countless figures using graphic application software. One more point is that the statements or analyses deriving from the figures are confidently error free.

- The on-screen real time display of PCX format figures — The Encapsulated Postscript figures is mainly for quality printing, and it forms in the background and dose not display in real time during a running process. Therefore, the on-screen real time display of figures is an essential part and should greatly enhance the debugging efficiency and make possible the writing of a huge and complex program that is also user-friendly, easy to maintain, as well as interactive and flexible.

- The data spreadsheet interface — The input or output of data from or to the Excel or Lotus-123 compatible worksheet is integrated. In cases that articulated or complex figures are desired such a function is readily convenient. Asyst programs for such postprocessing are also developed.

- The Mathematica programming language — The language is used both in the characterization of a wavelet variant and the post generation of various graphical renditions of two-dimensional or three-dimensional modulus and phase planes. Cares are also taken to eliminate any manual intervention of data commuting between the outcomes of Asyst analyses and the inputs of the Mathematica programs.

- The WinEdt macro programming language — The language is specifically used to develop a working platform or shell environment for writing the Asyst codes. With this all the code pieces, subroutines, and component files are managed in a scientific, efficient, and non-fallible way. Codes are displayed in much organized and eye-pleasant manners. Missing such an integration the editing and debugging must be quite painful and exhausting.

- The LaTeX packages — A vast amount of LaTeX codes were also written for the following functions: Displays and printouts of various figures for analyzing purposes; Eye-pleasant listings of Asyst and WinEDT programs for code development and debugging; Auto generations of printouts, papers, and reports; Facilities for easy management of numerous case studies. ❖

Chapter 2

Discrete Wavelet Bases

2.1 Introduction

Our numerical programs cover both the discrete and continuous wavelet transforms. In this chapter we give a brief description, as well as the relevant notations, of the discrete wavelet bases that were programmed. To the author's knowledge, the discrete wavelet basis categories covered here should be quite comprehensive since they have essentially included wavelet designs that span the full spectrum between extreme and opposite analytical properties in wavelet characterizations. And it is the author's belief that we have all the confidence on the results and statements we made in our previous studies.

2.2 Wavelet basis categories and their notations

The Riesz wavelet bases programmed here can be divided into three categories: orthonormal (ON), semi-orthogonal (SO), bi-orthogonal (BO). Wavelet packets, which are subgroups of the orthonormal bases, will also be covered later.

For the orthonormal bases they are further divided into several different subgroups: Daubechies wavelets (both the most and least asymmetric), Coiflets, Meyer wavelet, and Battle-Lemarié wavelets.

No detailing of these wavelets will be given; only the main criteria and core features, as well as relevant notations, of each categories will be briefed. With this the notations can be traced in our programs.

Let a function or a signal be denoted by $f(t)$; the two-scale scaling function of a Riesz basis be $\phi(t)$; the associate mother wavelet be $\psi(t)$ and its dyadic wavelets be $\psi_{j,k}(t) = \sqrt{2^j}\,\psi(2^j t - k)$, where $j, k \in \mathbf{Z}$ and k stands for translation and j for dilation. The concept of translations and dilations are illustrated in Figures 2.1 through 2.6.

The shell program to yield these figures is given as "`R_TnD.shl`". Note that a shell program appears first and it generally provides parameters for the relevant core programs that appear later. A shell program is a good reminder for functional, as well as proper, execution of a target purpose.

The space V_j (formed by $\psi_{j,k}$, $k \in \mathbf{Z}$ for a given j) in the multiresolution ladder are nested in $\cdots \subset V_{-1} \subset V_0 \subset V_1 \cdots$, and the finest and the coarsest scale space, say, for a 1024-point signal, are V_{10} and V_0, respectively; the number of filter coefficients or the number of convolution weights be N if the associated wavelet is finitely supported (support length equals to $N-1$); the dual wavelet and dual scaling function, if exist, be $\widetilde{\psi}(t)$ and $\widetilde{\phi}(t)$; the inner product be $\langle \cdot, \cdot \rangle$; and the Kronecker delta be $\delta_{j,k}$, $j, k \in \mathbf{Z}$, which is equal to 0 for $j \neq k$ and 1 for $j = k$.

Up until now, all practical wavelets of discrete transform are associated with the algorithm of multiresolution analysis (MRA) [5, 12]. For Riesz wavelets there always exist dual wavelets (post notations of "O" and "D") except for orthonormal wavelets, which are self-dual. Any discrete wavelet transform involves a set of two convolution operations: one yields detail information (notation "D" in figure 3.1); another yields smooth information (notation "S" in figure 3.1) [14]. Convolutions can either be implemented in an exact and direct way in time domain for compactly supported wavelets (support lengthes are represented as digital numbers) or, conversely, in an approximate and indirect way in frequency domain.

The symbols of various basis categories and related subgroups are given in the following sections.

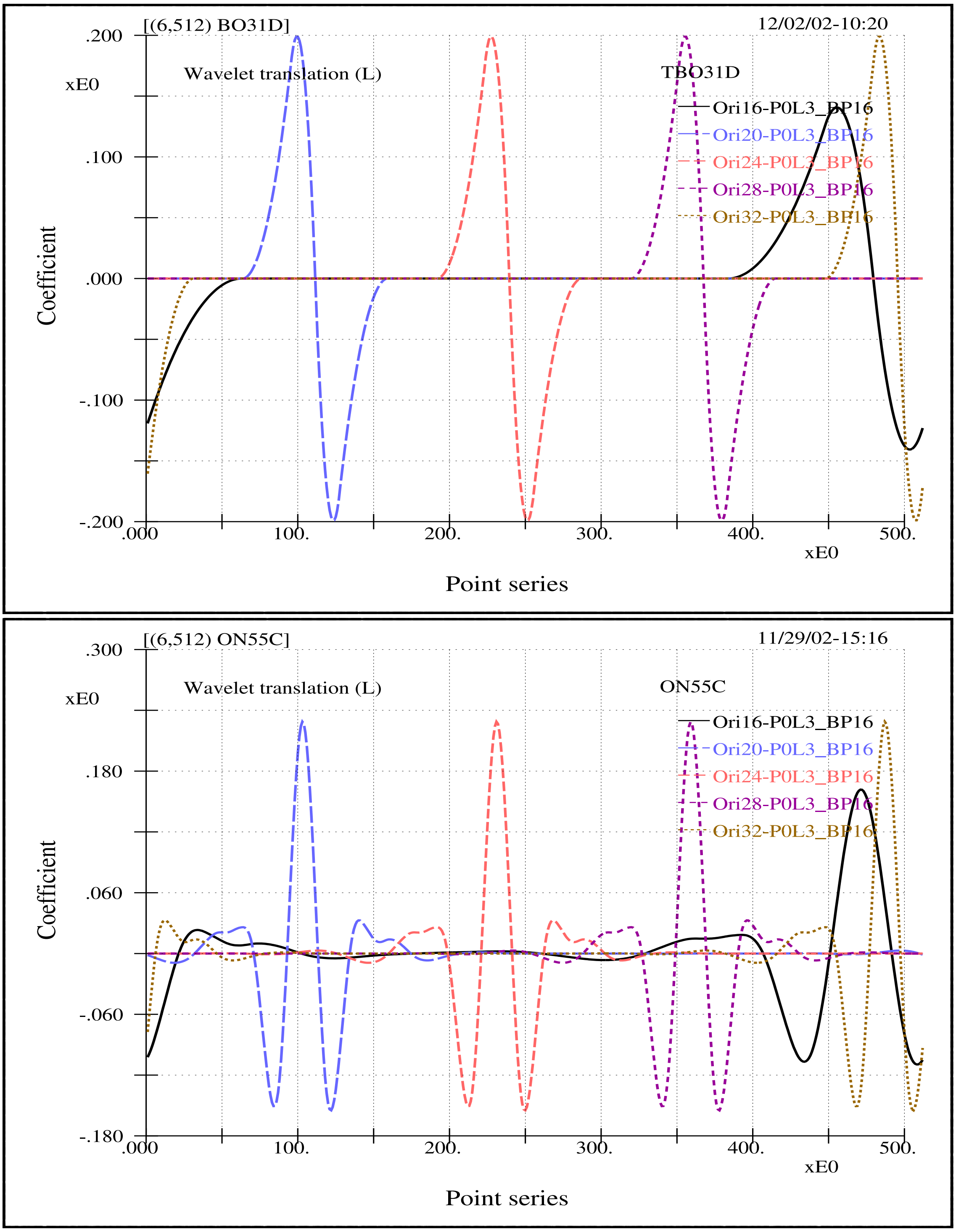


Figure 2.1: The wavelet translation concept within the scale range of level 3.

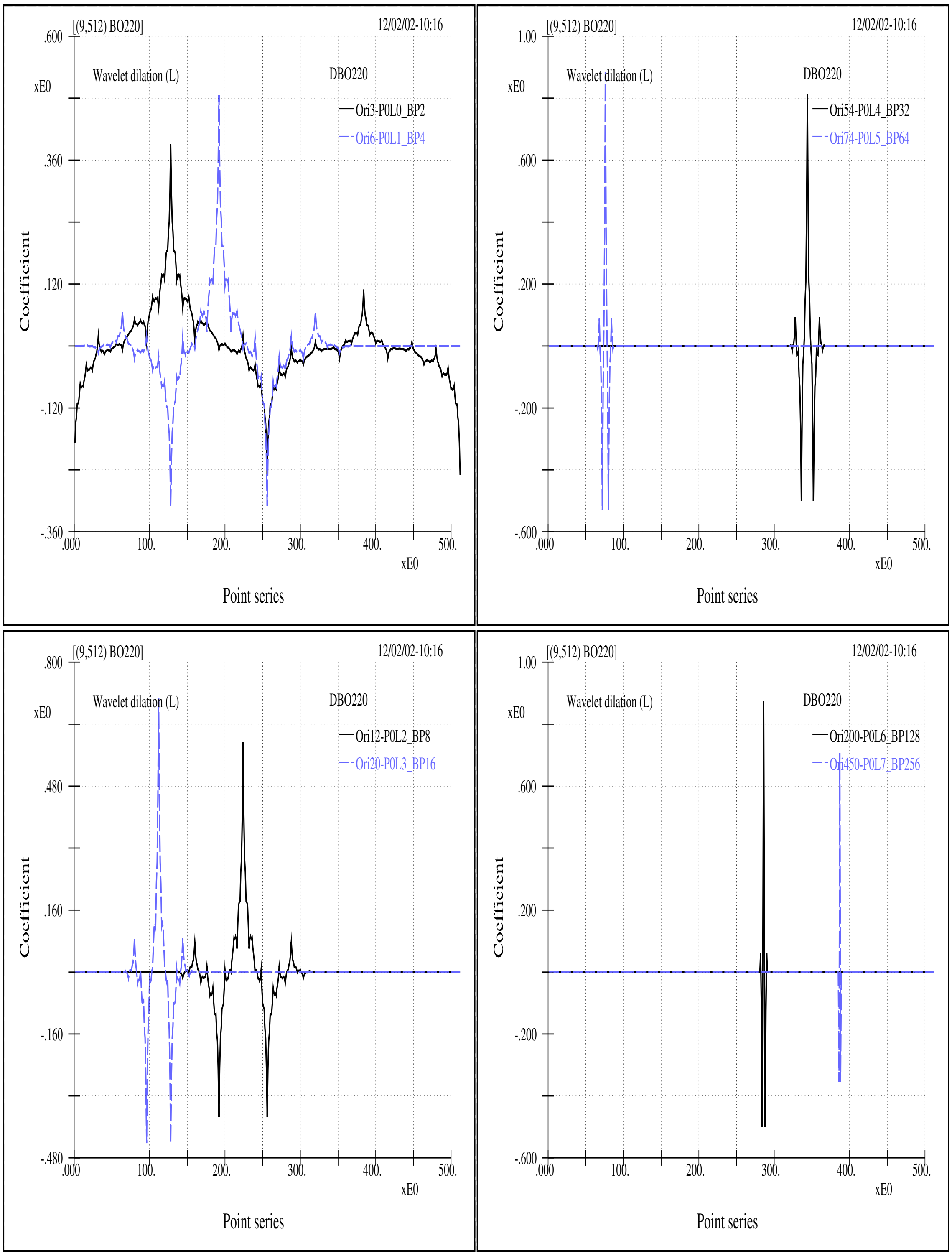


Figure 2.2: The wavelet dilation concept from scale level 0 to level 7 for the BO22O wavelet. Each wavelet curve corresponds to an individual translation location.

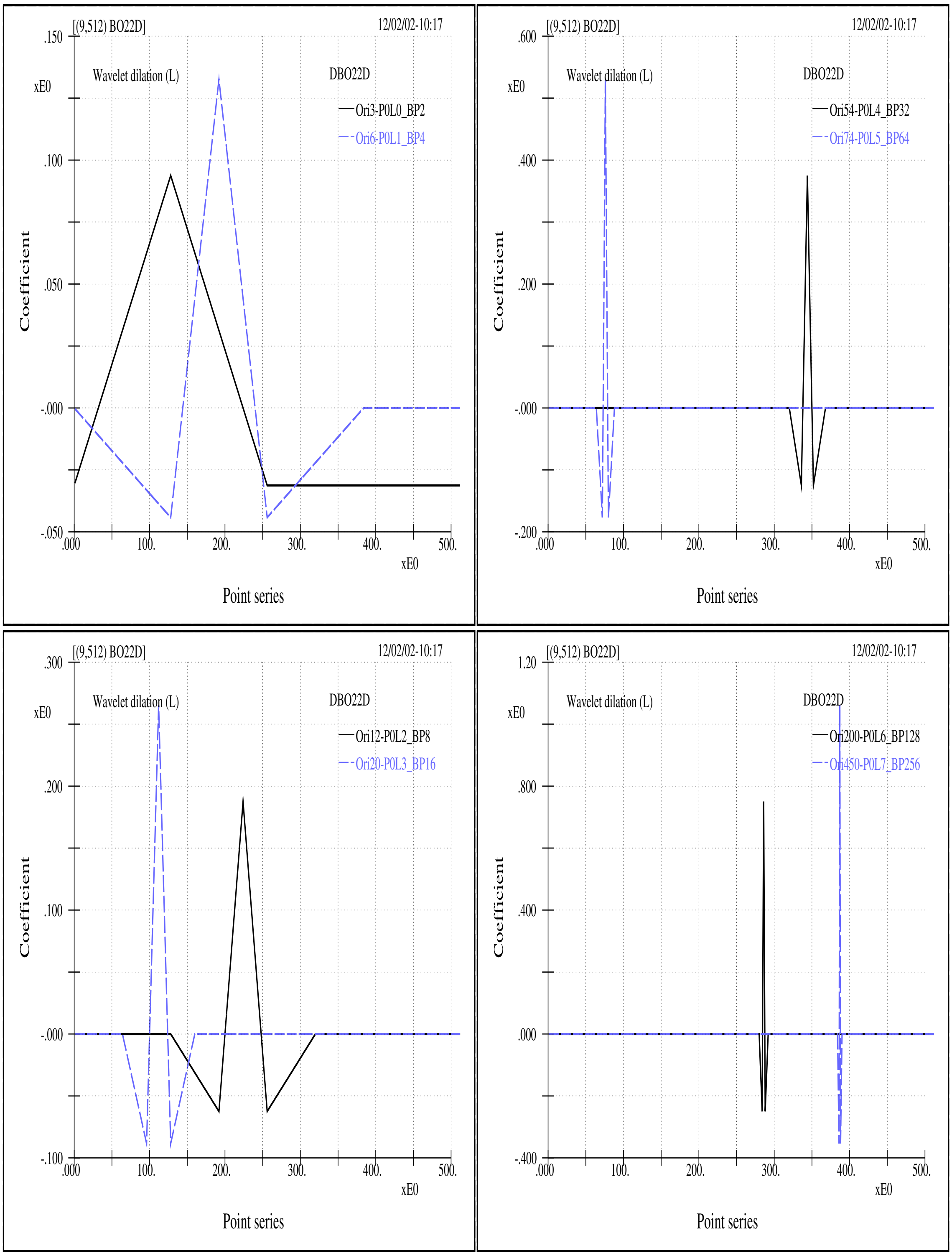


Figure 2.3: The wavelet dilation concept from scale level 0 to level 7 for the BO22D wavelet. Each wavelet curve corresponds to an individual translation location.

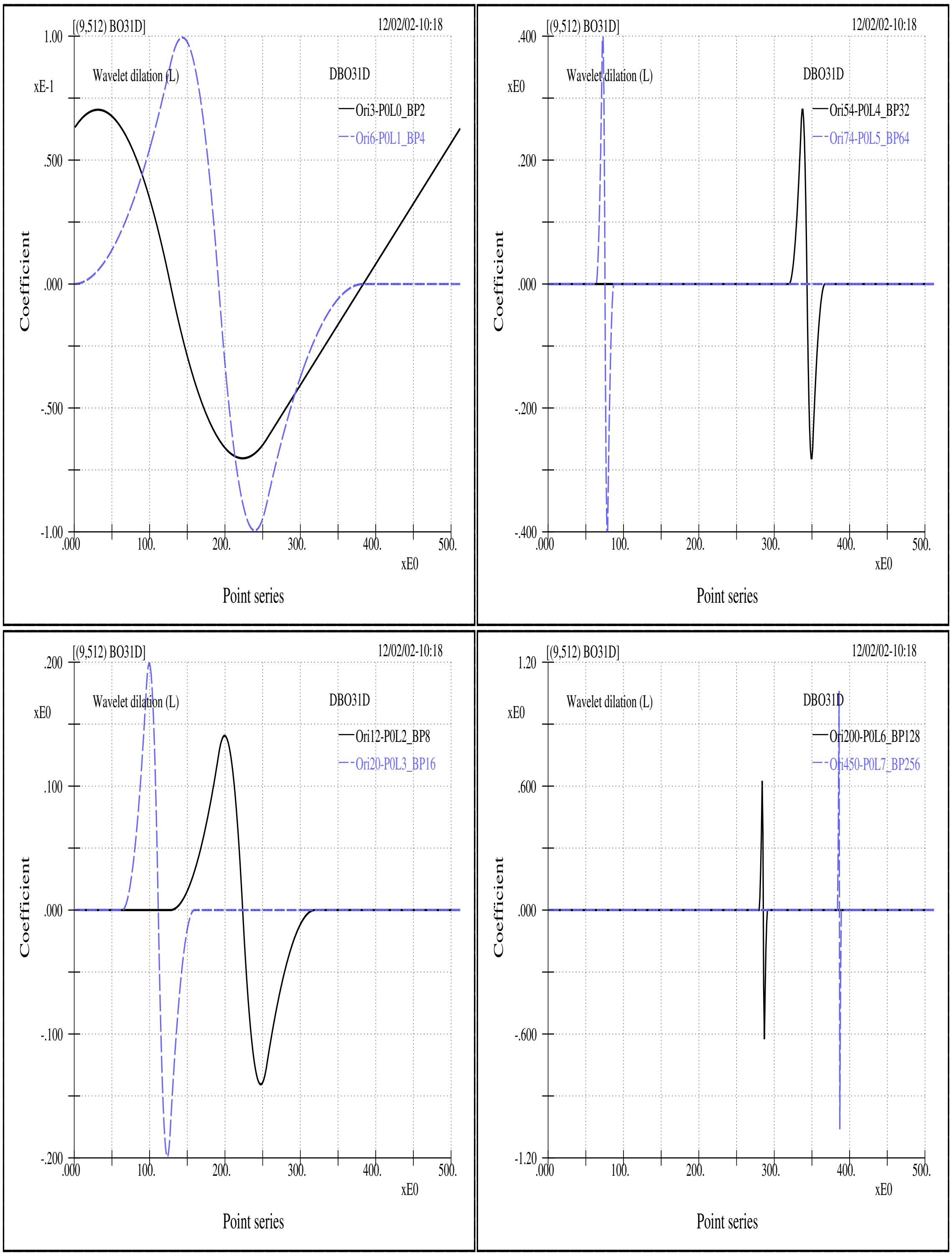


Figure 2.4: The wavelet dilation concept from scale level 0 to level 7 for the BO31D wavelet. Each wavelet curve corresponds to an individual translation location.

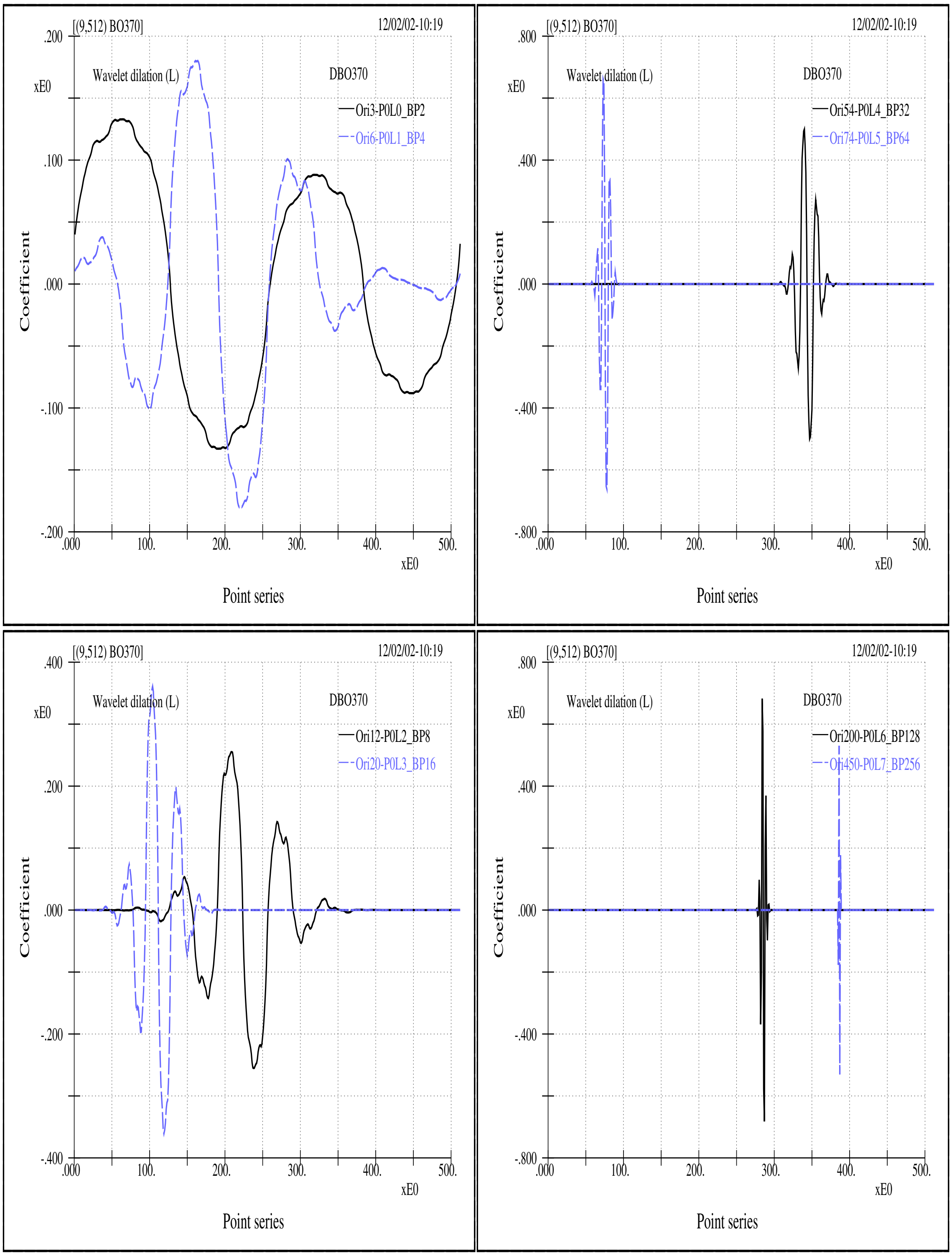


Figure 2.5: The wavelet dilation concept from scale level 0 to level 7 for the BO370 wavelet. Each wavelet curve corresponds to an individual translation location.

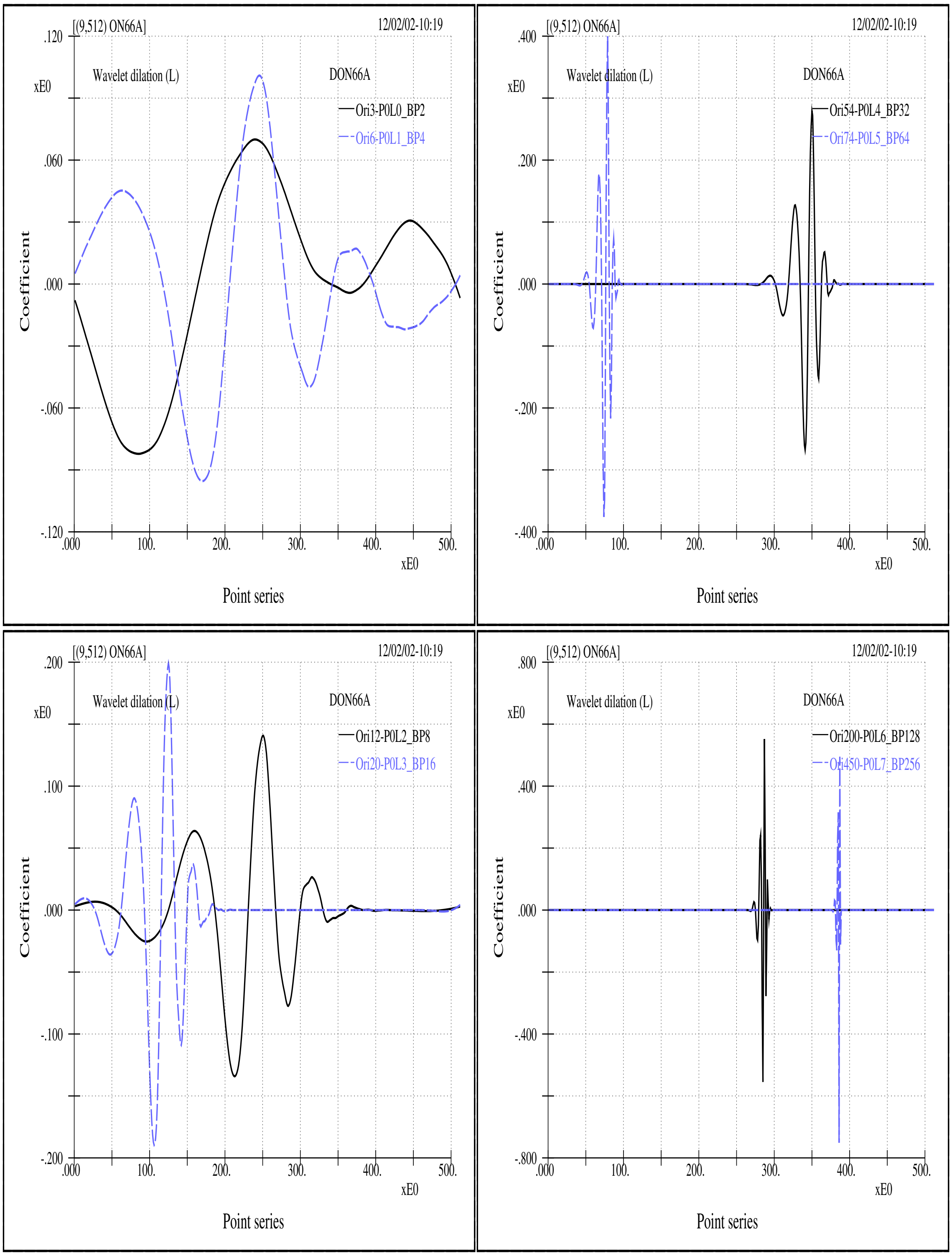


Figure 2.6: The wavelet dilation concept from scale level 0 to level 7 for the ON66A wavelet. Each wavelet curve corresponds to an individual translation location.

2.3 Orthonormal wavelets (ON)

The orthonormal wavelets covered here include the following categories: Daubechies most compactly supported wavelets (denoted as ONxxA); Daubechies least asymmetric wavelets (ONxxS); Coiflets (ONxxC); Meyer wavelet (Meyer); Battle and Lemarié wavelet (B&L). As mention before, the notation x is an integer related to the support length (physically, an indication of the span of a function's independent variable) of wavelet.

$$\psi = \widetilde{\psi}, \tag{2.1}$$

$$\phi = \widetilde{\phi}, \tag{2.2}$$

$$\langle \psi_{j,k}, \widetilde{\psi}_{l,m} \rangle = \delta_{j,l}\delta_{k,m}, \tag{2.3}$$

$$f(t) = \sum_{j,k} \langle f, \psi_{j,k} \rangle \psi_{j,k}, \tag{2.4}$$

One MRA ladder (single set of frame bounds),

One filter pair (one smooth and one detail).

2.3.1 Daubechies most compactly supported wavelets (ONxxA)

Given the same span length of support (from different basis categories), the wavelet in this subgroup provides the maximum number of vanishing moment. Stated otherwise, wavelet in this subgroup are the most compactly supported wavelet among wavelets with compatible number of vanishing moment. The famed most compactly supported wavelet of all has only four convolution filter coefficients (ON22A). These wavelets are quite asymmetry (so, the "A" in ONxxA). The vanishing moments and the number of filter coefficients are, respectively,

$$\int_{-\infty}^{\infty} t^l \psi(t) dt = 0, \quad l = 0, 1, \cdots, x, \tag{2.5}$$

$$N = 2x, \tag{2.6}$$

where x is the integer number in ONxxA. The minimum number of x is 2.

The mother and farther wavelets for the group corresponding to the originating points of 12 (boundary point based on level 2) and 6 (boundary point based on level 3), respectively, for this group are shown in Figures 2.7 and 2.8. The shell program to yield these is "`R_DnR.shl`".

2.3.2 Daubechies least asymmetric wavelets (ONxxS)

For a given support width, these wavelets, in contrast to those of the ONxxA subgroup, are the most symmetric ones (so, the "S" in ONxxS, but still not symmetric). They have the same representations of vanishing moments and number of filter coefficients as those of ONxxA. But the known minimum number of x is 4. The mother and farther wavelets for this group corresponding to the same originating points as the previous ones are shown in Figures 2.9 and 2.10.

2.3.3 Coiflets (ONxxC)

The Coiflets have vanishing moments for both ψ and ϕ. Judging from the theory of Taylor expansion, they therefore have high compressibility for fine detail information, that is to say, a great portion of wavelet transform coefficients corresponding to the finer scales are relatively small. Henceforth, there exists simple quadrature rule to calculate the fine smooth information (i.e., the calculation of inner product of a function and the fine-scale scaling function is more efficient)[5]. Since any discrete wavelet transform involves both smoothing and detailing operations, some advantages from these two properties do exist in certain applications that do not stress lossless of signal contents or perfect reconstructions [4, 15]. Their vanishing moments and number of convolution filter coefficients for the Coiflets are

$$\int_{-\infty}^{\infty} t^l \psi(t) dt = 0, \qquad l = 0, 1, \cdots, x, \tag{2.7}$$

$$\int_{-\infty}^{\infty} \phi(t)dt = 1, \tag{2.8}$$

$$\int_{-\infty}^{\infty} t^l \phi(t)dt = 0, \quad l = 1, \cdots, x, \tag{2.9}$$

$$N = 6x. \tag{2.10}$$

For this group the mother and farther wavelets are shown in Figures 2.11 and 2.12.

2.3.4 Meyer wavelet (Meyer or ME)

The Meyer wavelet (denoted as Meyer or ME) is the wavelet that has the most compact support of all in the frequency domain. Note here, if without any specific assignment, "support length" refers to the time domain. Therefore, due to the significant contrasting properties of support lengthes between the two opposing Fourier domains, the Meyer wavelet is infinitely differentiable in time domain and its support length $N \to \infty$. More precisely, the time domain wavelet has an infinite Lipschitz regularity C^{∞} and does not possess an exponential decay. The associated mother and farther wavelets corresponding to the same originating points are shown in Figure 2.13.

2.3.5 Battle and Lemarié wavelet (B&L)

The Battle and Lemarié wavelet (denoted as B&L or LE) of m^{th} order is constructed from the orthonormal scaling function that is derived by applying the standard orthonormalization trick to the m^{th} order cardinal B-spline function N_m [1, 2].

For $m = 1$, it is exactly the Haar wavelet. And the Haar wavelet is the only finitely supported wavelet in this group. The Haar wavelet is also a discontinuous wavelet with the most compact support and it is also the case of BO11O=BO11D in the bi-orthogonal category to be mentioned below. All other wavelets in this group are infinitely supported. These wavelets have an exponential decay and possess C^{m-2} regularity.

The mother and farther wavelets for the Battle-Lemarié wavelet ($m = 4$) are shown in Figure 2.14. Compared to those curves of Meyer wavelet (Figure 2.13), they look quite identical even though their constructions or derivations or formulations, are completely different. In addition, their features of Lipschitz regularity and decay property are analytically quite different.

2.4 Semi-orthogonal wavelets (SOxO and SOxD)

The basis functions of the dual wavelets for a semi-orthogonal wavelet are inter-scale, but not inner-scale, orthogonal. Their scaling functions are cardinal B-spline N_m and have finite two-scale relations. In this category although there are two distinctive (and independent) filter pairs (one for the decomposition and the other for the reconstruction), there is only one MRA V_j-ladder. It was shown by Chui [2, 3] that semi-orthogonal wavelets associated with cardinal B-spline of order higher than $m = 3$ are almost all identical to a modulated Gaussian function (but a modulated Gaussian is not a wavelet). It is therefore sufficient to take the semi-orthogonal wavelets associated with cubic B-spline of fourth order ($m = 4$). They have the following characterizations.

$$\psi \neq \widetilde{\psi}, \tag{2.11}$$

$$\phi = \widetilde{\phi}, \tag{2.12}$$

$$\langle \psi_{j,k}, \psi_{l,m} \rangle = \langle \widetilde{\psi}_{j,k}, \widetilde{\psi}_{l,m} \rangle = \delta_{j,l}, \tag{2.13}$$

$$f(t) = \sum_{j,k} \langle f, \psi_{j,k} \rangle \widetilde{\psi}_{j,k} = \sum_{j,k} \langle f, \widetilde{\psi}_{j,k} \rangle \psi_{j,k}, \tag{2.14}$$

$$N = 3x - 1 \quad \text{for SO}x\text{D}, \tag{2.15}$$

$$N \to \infty \quad \text{for SO}x\text{O}. \tag{2.16}$$

One MRA ladder ,

Two filter pairs ,

The mother and farther wavelets of the fourth order and the associated dual wavelets are shown in Figure 2.15.

2.5 Bi-orthogonal wavelets (BOxyO and BOxyD)

The wavelets in this category are constructed also by Daubechies. They sometimes are called non-orthogonal wavelets. As is well known all real-valued compactly supported orthonormal wavelets, except the Haar wavelet, are not symmetrical. However, from the point of view of reconstructing a signal from its partially truncated wavelet coefficients, the symmetry is a desired property for the convolution filter when more natural perceptions are desired and smoother variations are important in reconstruction rendering.

A very practical and also very important implication can be stated here. If non-symmetrical function bases are used, then a small change in the wave form causes dramatic variations of scale information of transform coefficients. That is to say the two sets of transform coefficient are strikingly different and almost incomparable. In other words, to have minor impacts to the data analysis, it is desirable to have basis that is as symmetrical as possible. Moreover, when considering that acquired signals may contain random errors or noise and that uncontrolled factors may be present in experiments, one is easy to comprehend the significance of this property. In fact numerous results and figures from our previous studies have provided conclusive evidences of this.

For a Bi-orthogonal wavelet the symmetry of wavelet curve is achieved by sacrificing orthogonality among its basis functions. And by doing so one has dual wavelet and scaling function pairs, i.e., one wavelet and one scaling function together with one dual wavelet and one dual scaling function, to comprise the same reconstruction of a signal.

It is obvious that the analytical requirements or conditions for semi-orthogonal cases are more general, i.e., less restricted, than those of orthogonal ones, and that the bi-orthogonal cases are even more general. This situation is indicated by the existence of the additional freedom of dual scaling function, as is reflected by the two parameters x and

y in the notations of BOxyO and BOxyD. Nevertheless, the wavelets in this category involve only one pair of independent filters for both decomposition and reconstruction even though there involve two different MRA ladders that are associated with their own individual sets of Riesz bounds. This feature is quite opposite to the case of semi-orthogonal wavelets where they involve one MRA ladder but with two filter pairs.

$$\psi \neq \widetilde{\psi}, \tag{2.17}$$

$$\phi \neq \widetilde{\phi}, \tag{2.18}$$

$$\langle \psi_{j,k}, \widetilde{\psi}_{l,m} \rangle = \langle \phi_{j,k}, \widetilde{\phi}_{l,m} \rangle = \delta_{j,l}\delta_{k,m}, \tag{2.19}$$

$$f(t) = \sum_{j,k} \langle f, \psi_{j,k} \rangle \widetilde{\psi}_{j,k} = \sum_{j,k} \langle f, \widetilde{\psi}_{j,k} \rangle \psi_{j,k}, \tag{2.20}$$

$$N = 2y + x - 1 \quad \text{for BO}xy\text{O and } x \text{ odd}, \tag{2.21}$$

$$N = 2y + x - 2 \quad \text{for BO}xy\text{O and } x \text{ even}, \tag{2.22}$$

$$N = 2y + x - 1 \quad \text{for BO}xy\text{D and } y \text{ odd}, \tag{2.23}$$

$$N = 2y + x - 2 \quad \text{for BO}xy\text{D and } y \text{ even}. \tag{2.24}$$

Two MRA ladders,

One filter pair.

The mother and farther wavelets for this group and their associated duals are shown in Figures 2.16 through 2.19. ❖

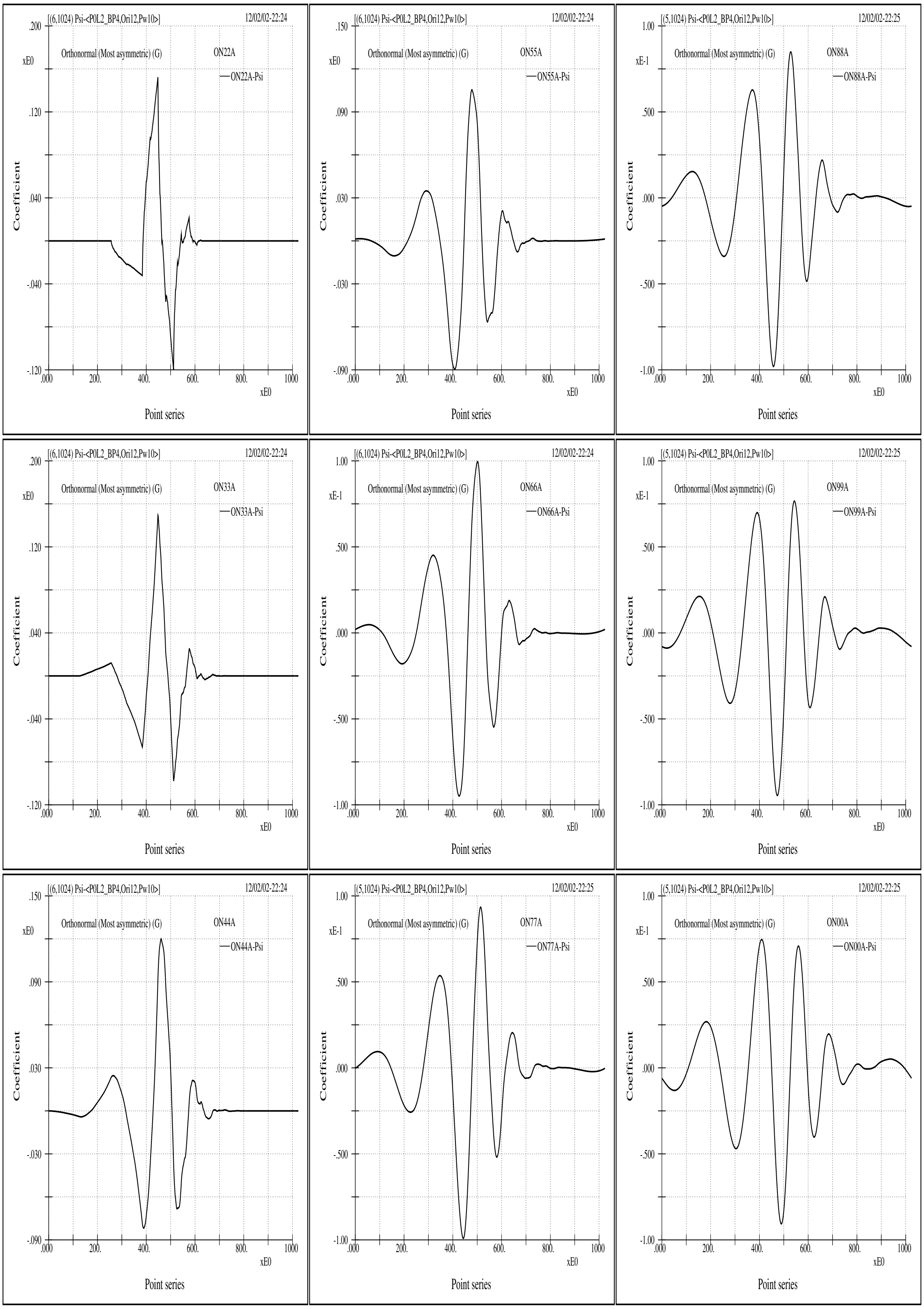


Figure 2.7: The mother wavelets of the ONxxA group originating from the point location of 12. Here the boundary point should be based on a level less or equal to 3.

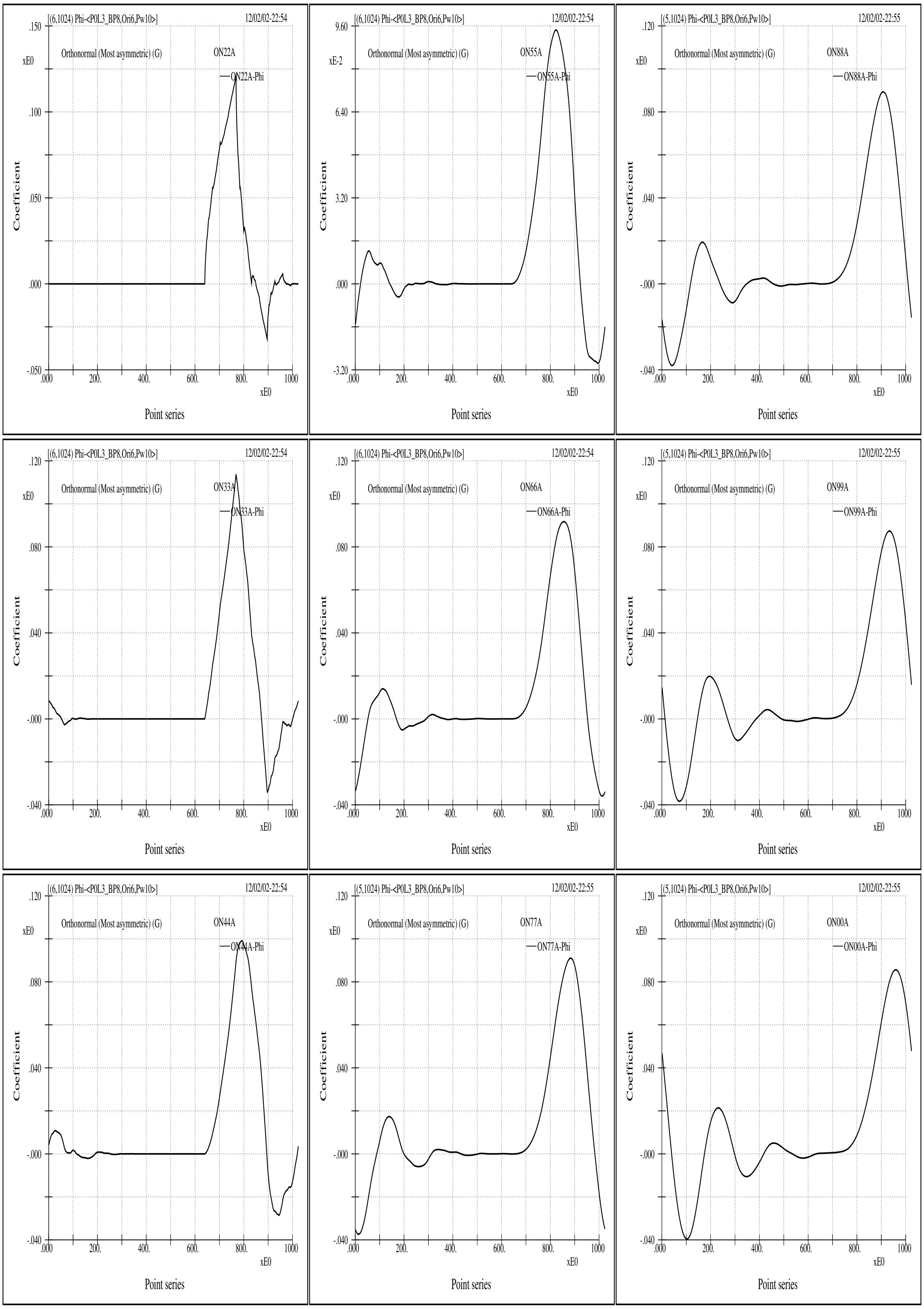


Figure 2.8: The farther wavelets of the ONxxA group originating from the point location of 6. Here the boundary point should be based on a level higher or equal to 3.

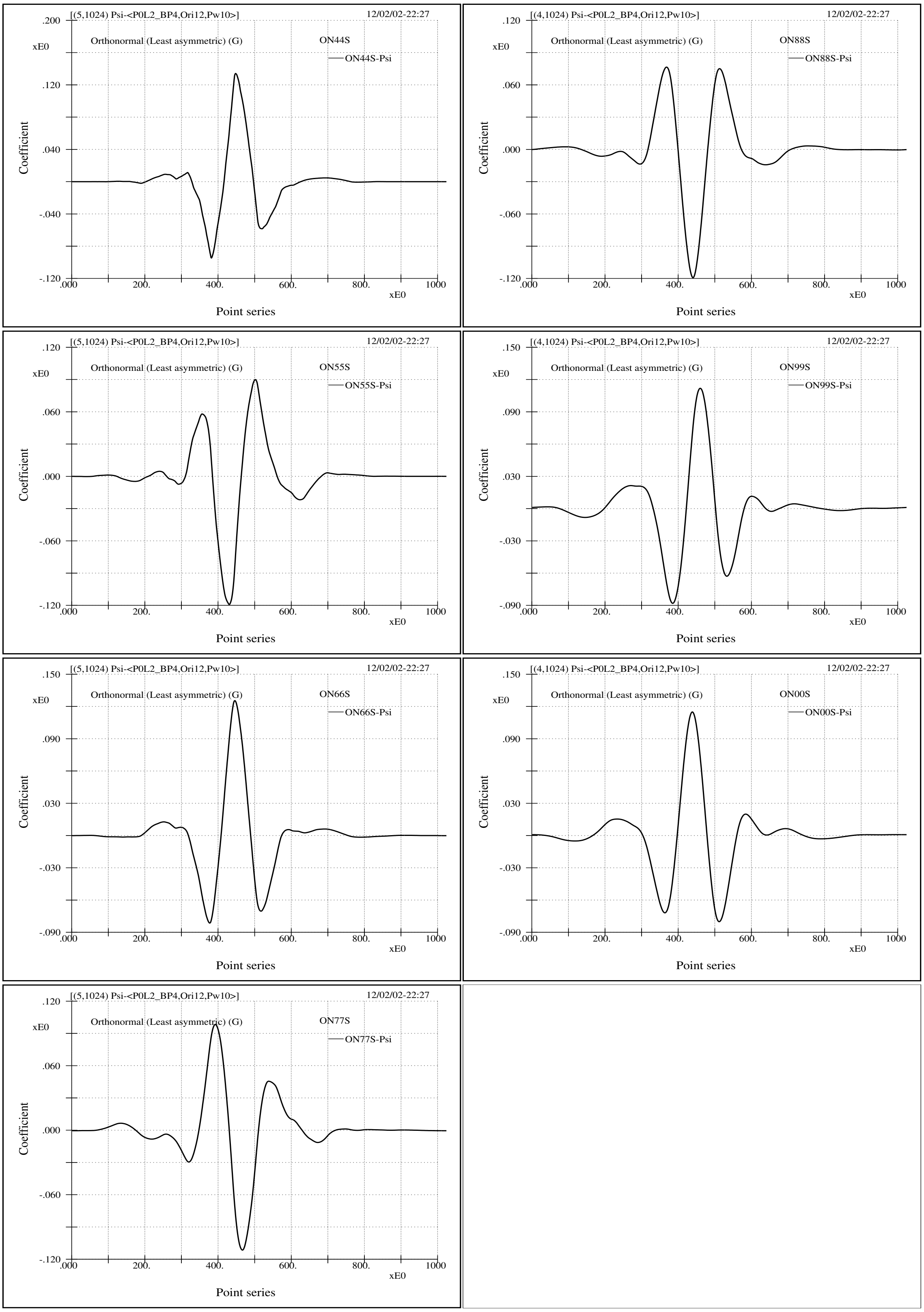


Figure 2.9: The mother wavelets of the ONxxS group originating from the point location of 12. Here the boundary point should be based on a level less or equal to 3.

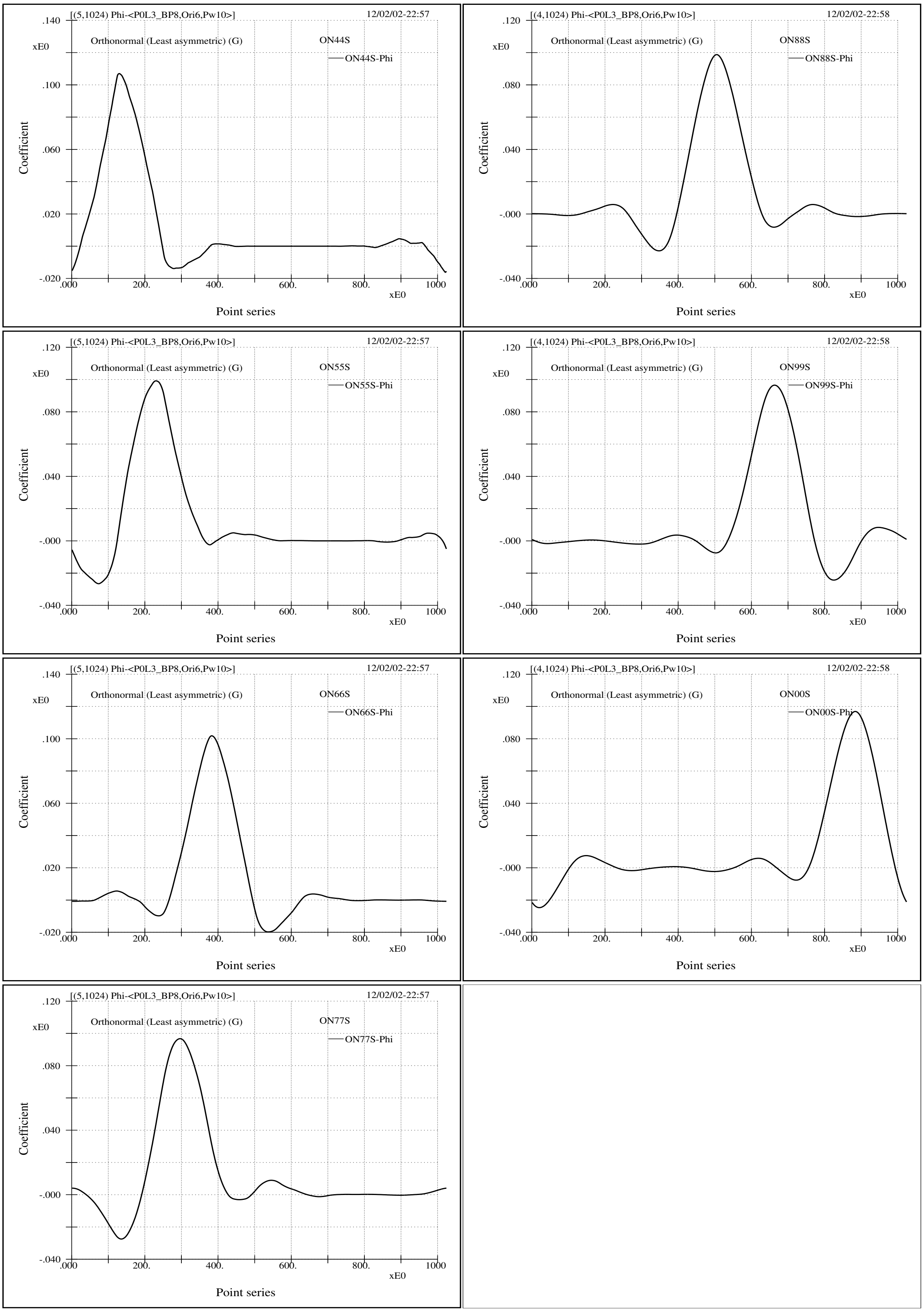


Figure 2.10: The farther wavelets of the ONxxS group originating from the point location of 6. Here the boundary point should be based on a level higher or equal to 3.

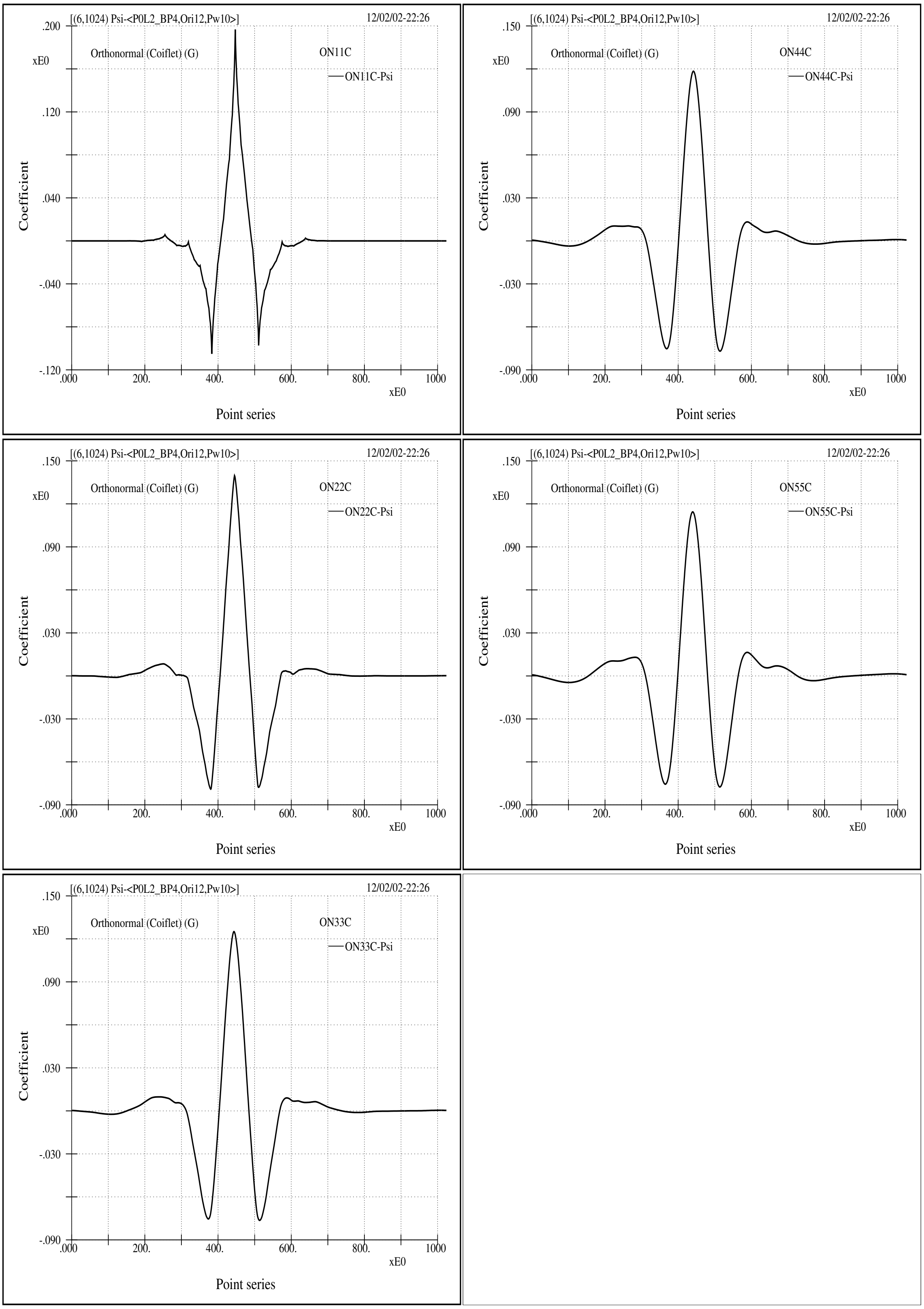


Figure 2.11: The mother wavelets of the ONxxC group originating from the point location of 12. Here the boundary point should be based on a level less or equal to 3.

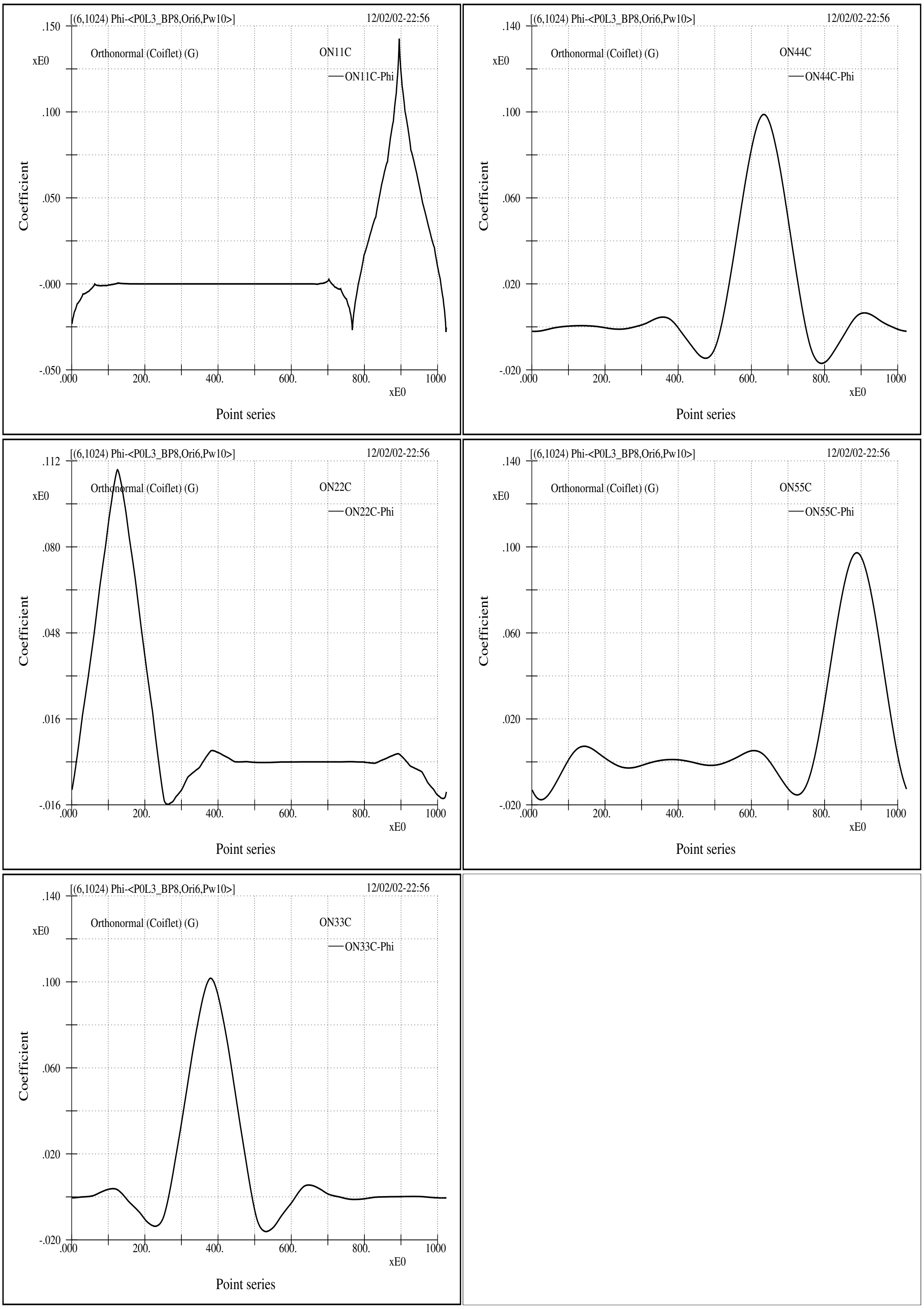


Figure 2.12: The farther wavelets of the ONxxC group originating from the point location of 6. Here the boundary point should be based on a level higher or equal to 3.

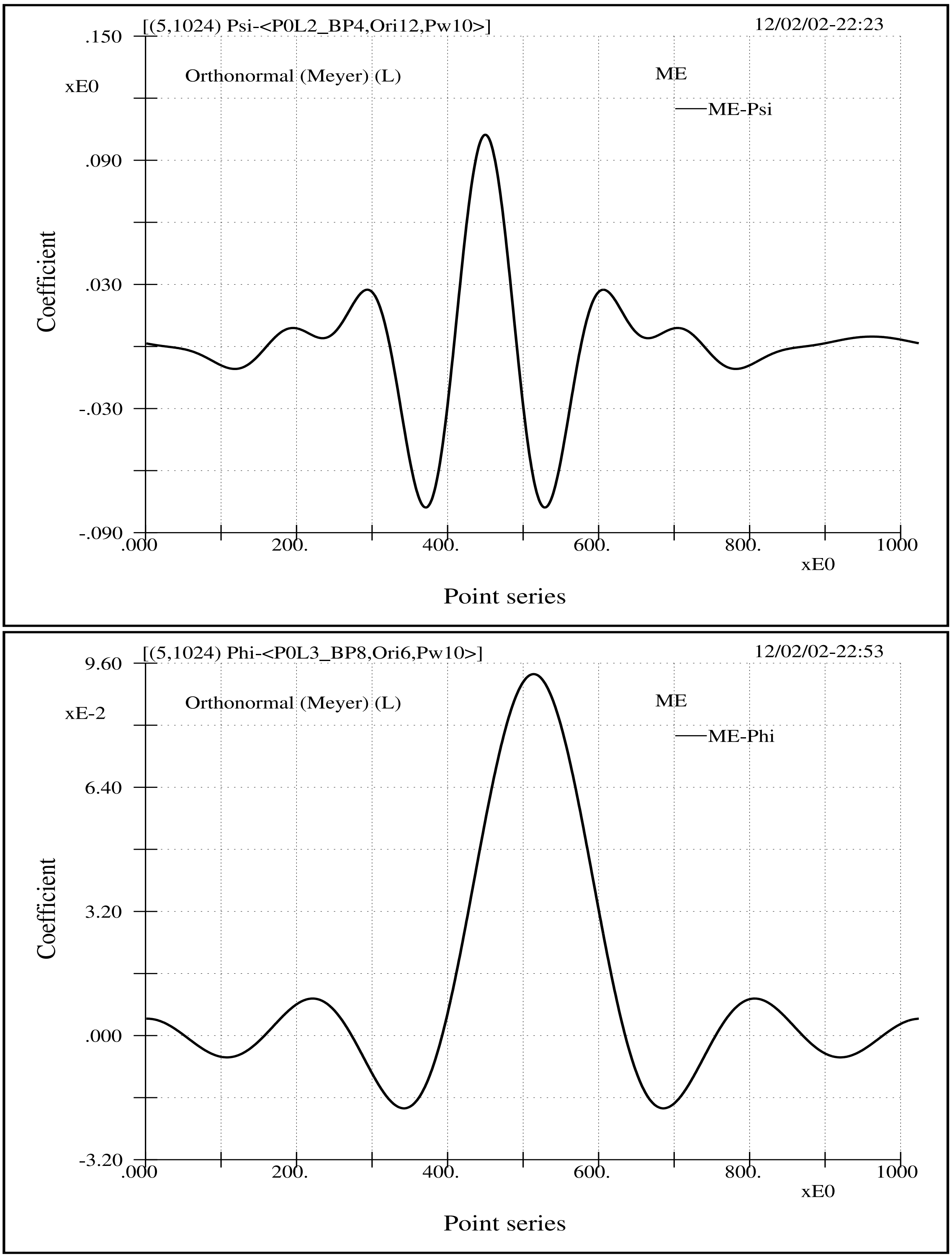
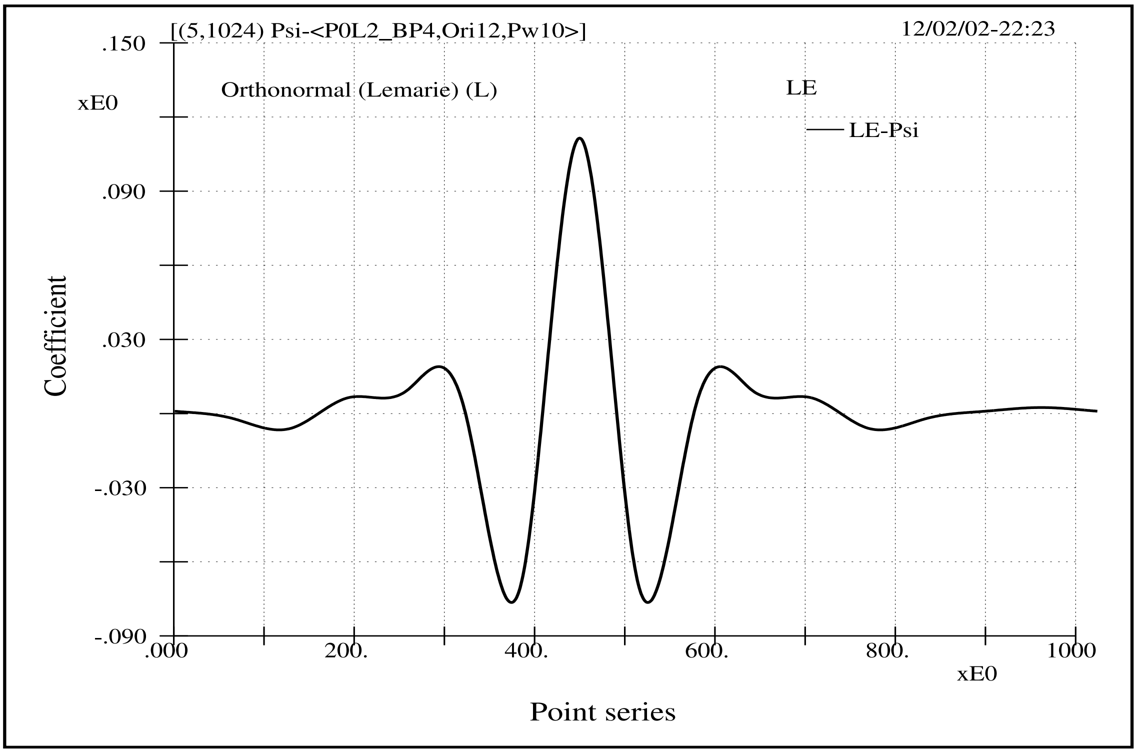


Figure 2.13: The mother `(top)` and farther `(bottom)` wavelets of the Meyer wavelet originating from the point location of 12 and 6, respectively, for the boundary point based on level 3. This figure is to be compared to the next one.



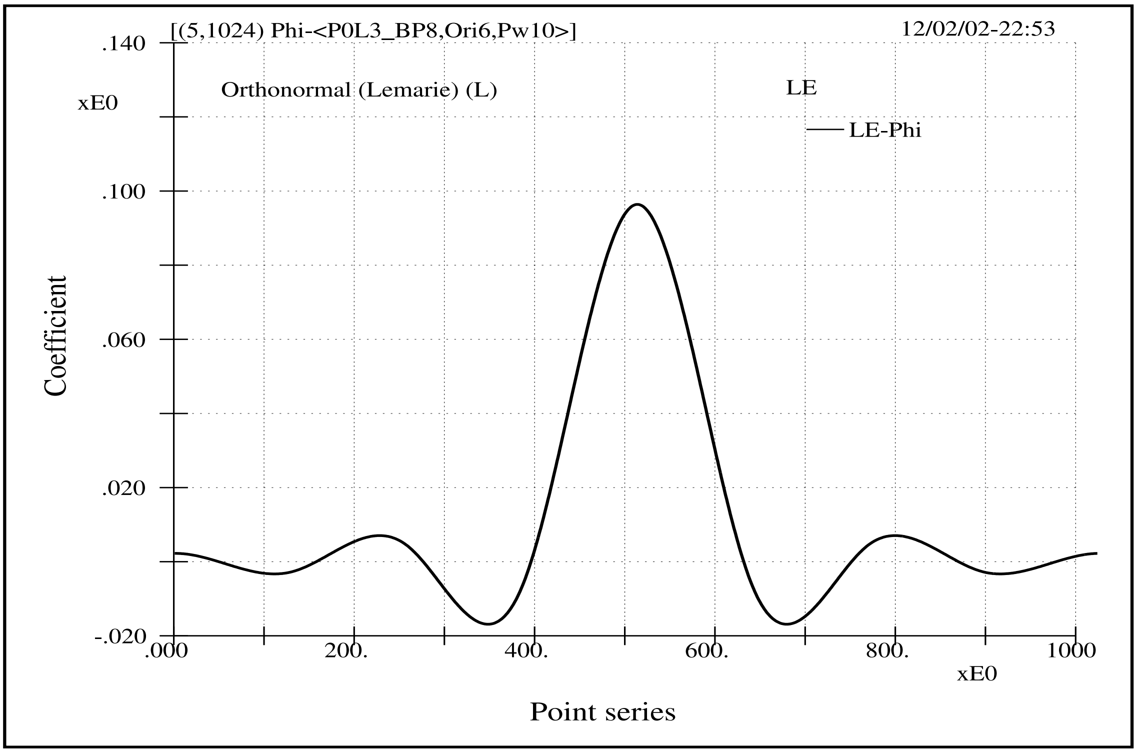


Figure 2.14: The mother `(top)` and farther `(bottom)` wavelets of the Battle and Lemarié wavelet originating from the point location of 12 and 6, respectively, for the boundary point based on level 3. Comparing the wavelet functions shown here with those shown in last figure (Figure 2.13), we see that two wavelets of similar looks but with quite distinctive constructions and analytic properties (such as regularity, differentiability, rate of decay, support length, etc.) It therefore gives rise the concerns that many complicated aspects of discrete Riesz wavelet seem not to reflect their associations with practical concerns.

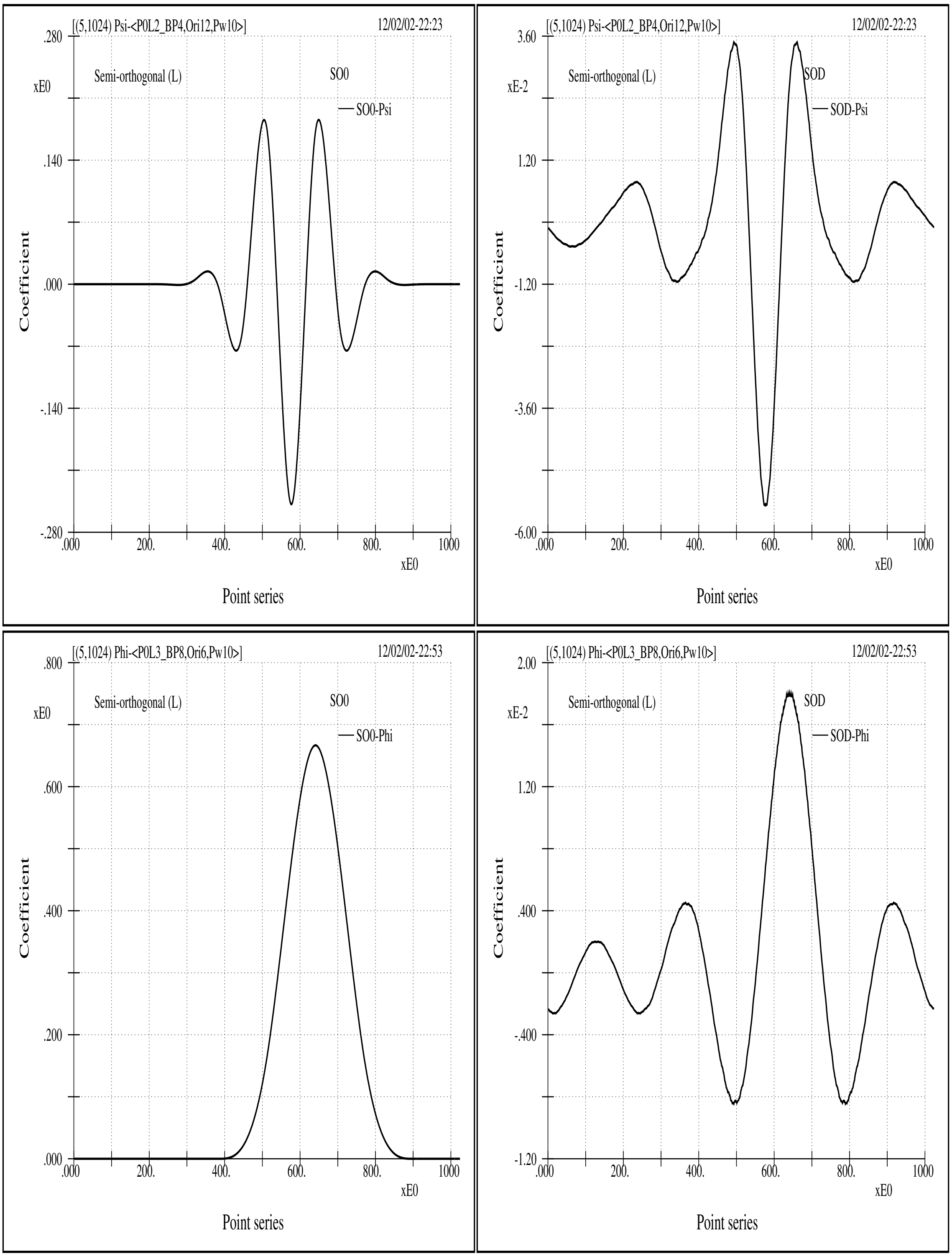


Figure 2.15: The mother `(top left)` and farther `(bottom left)` wavelets, as well as their duals `(right)`, of Chui's semi-orthogonal wavelet [2, 3] originating from the point location of 12 and 6, respectively, for the boundary point based on level 3.

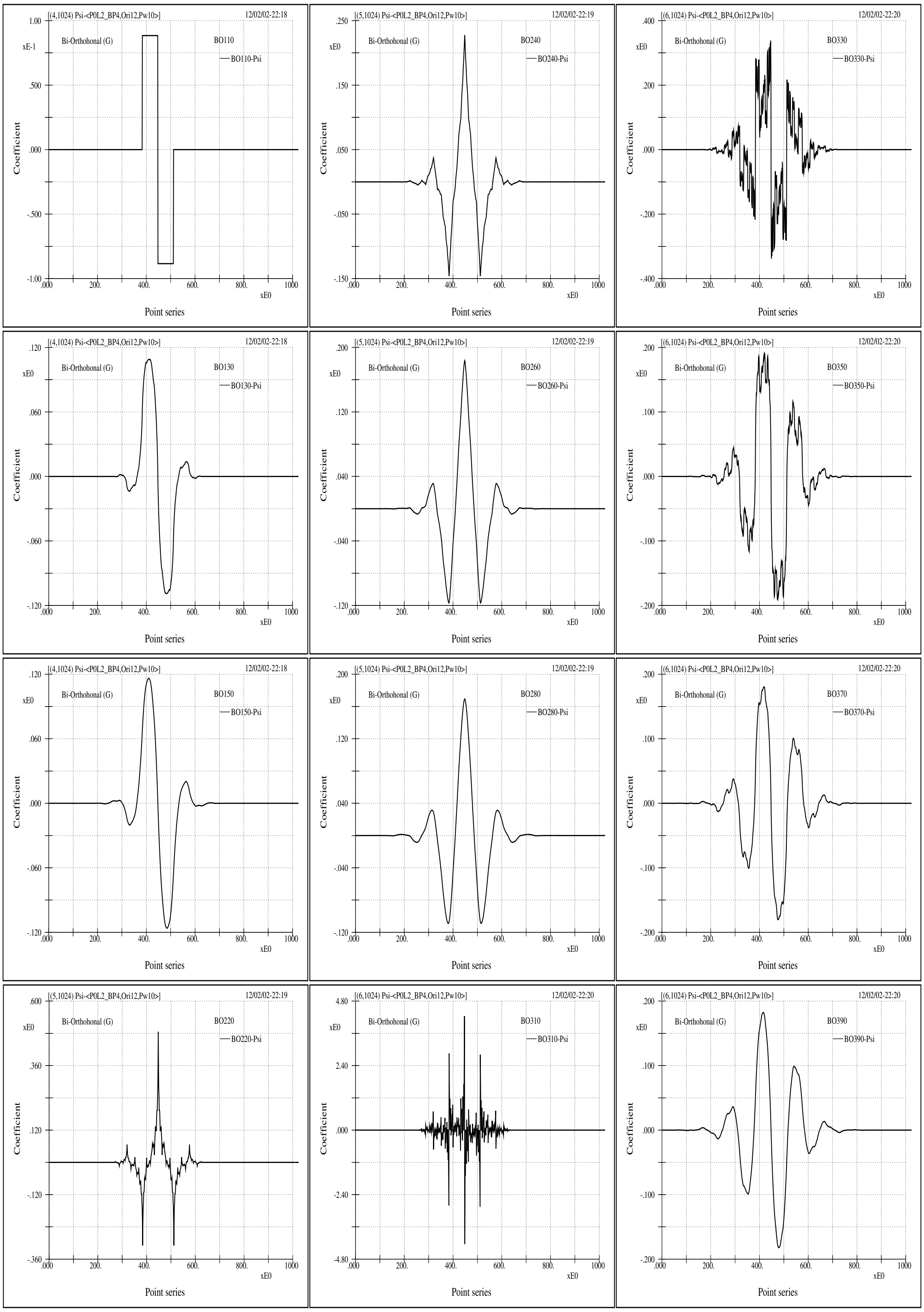


Figure 2.16: The mother wavelets of the BOxxO group originating from the point location of 12. Here the boundary point should be based on a level less or equal to 3.

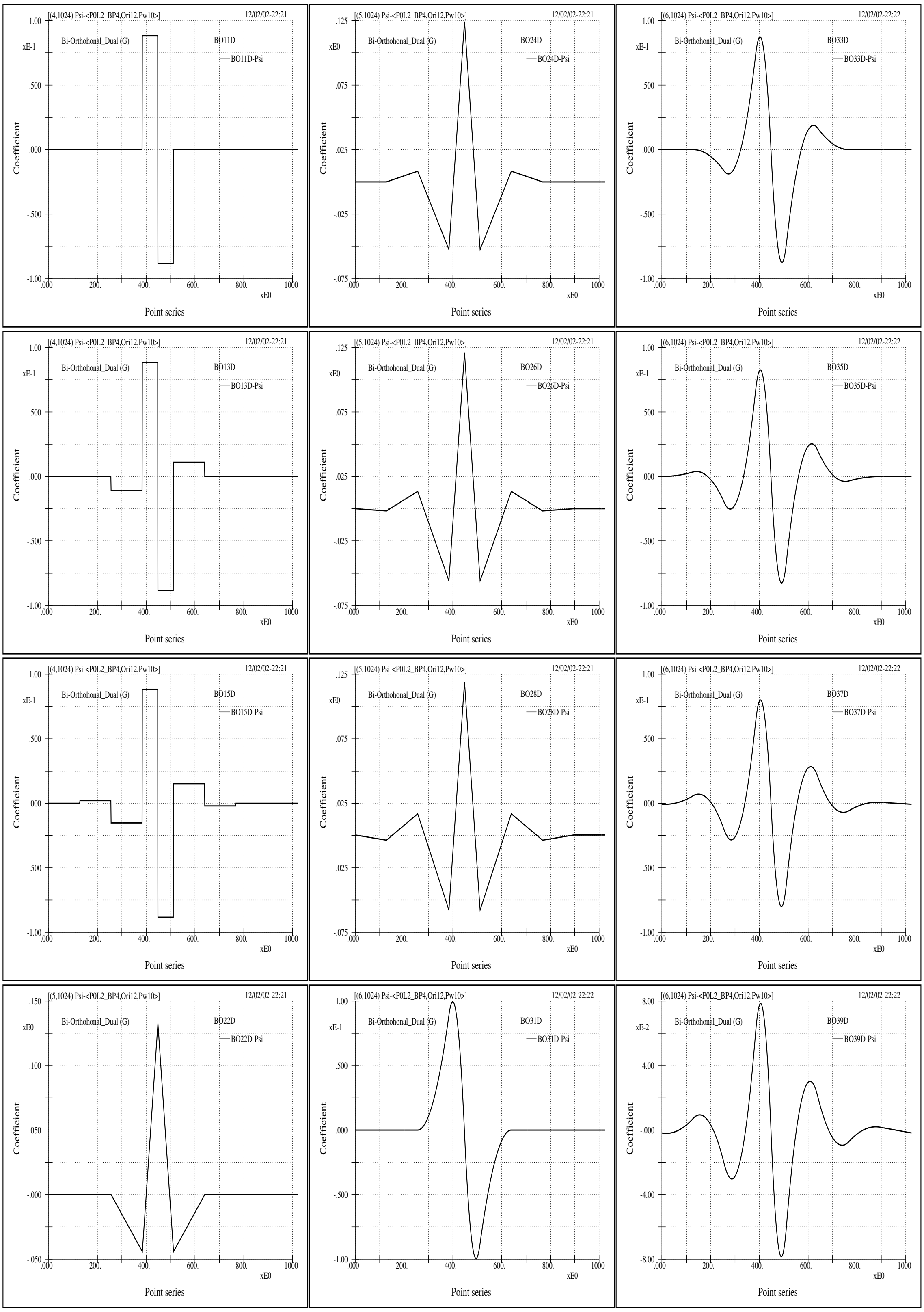


Figure 2.17: The mother wavelets of the BOxxD group originating from the point location of 12. Here the boundary point should be based on a level less or equal to 3.

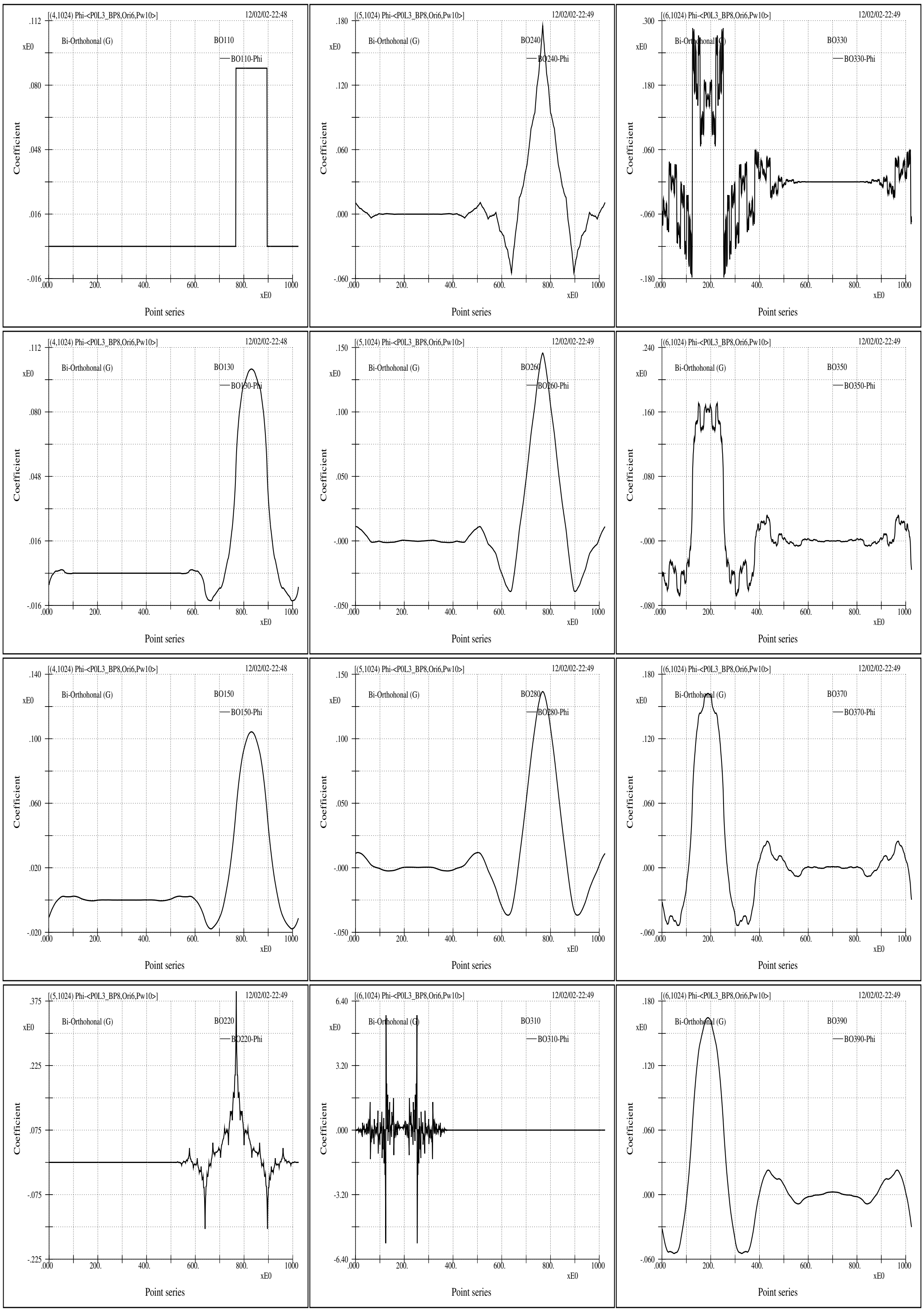


Figure 2.18: The farther wavelets of the BOxxO group originating from the point location of 6. Here the boundary point should be based on a level higher or equal to 3.

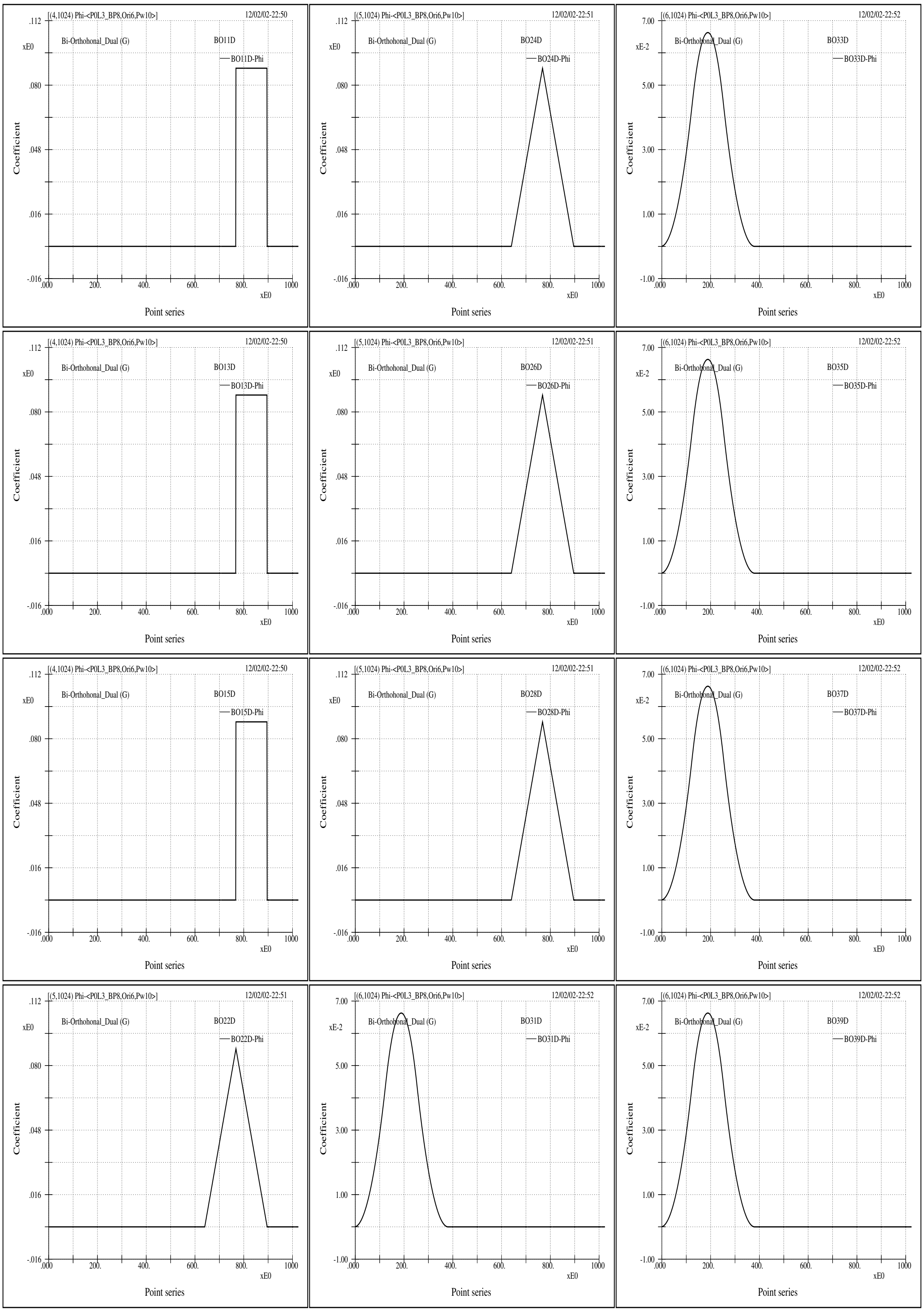


Figure 2.19: The farther wavelets of the BOxxD group originating from the point location of 6. Here the boundary point should be based on a level higher or equal to 3.

Chapter 3

Characterizations of Discrete Wavelet Bases

3.1 Wavelet packets

The wavelet transform coefficients derived from an orthonormal wavelet decomposition can be decomposed further by using either the same set of filter coefficients (called two-scale sequence in Chui [2]) associated with the original wavelet, or different sets of filter coefficients associated with any other orthonormal wavelets. These further decompositions are of a tree-like refinement process and are called the wavelet packet transforms. There can basically be infinitely many wavelet packet decompositions.

The wavelet packet coefficients give better frequency resolutions with the reverse effect of longer time supports. There are no simple formula to describe the tree-like decompositions, but a schematic plot easily elucidates the mechanism and is shown in Figure 3.1.

To form the set of transform coefficients, the branch patterns and the number of branches can be chosen in any way so long as there is no repeated occurrence within any column for the whole stretch of the coefficients. That is to say, any column, wide or narrow, must have one and only one contribution from all levels (rows). But different columns may have contributions from the same row. Due to the tree-like process the computation is more time-consuming.

For this category we have two criteria for selecting a best basis. One of the criteria is generally simply referred as the “best basis”; another the “best level basis”. Take for example, for a 1024-point signal, the finest level occurs at $j = \log_2 1024 = 10$ and there are 2^{10} different choices of bases. And within these 2^{10} choices the one which yields the minimum entropy is called the “best basis”. But if we enforce the restriction that all the wavelet packets be at the same level j, then we have 10 levels (0 to 9) to choose from, and the level that yields minimum entropy is called best level basis.

The indexes of a wavelet packet coefficient, i.e., the subscript and superscript of U in the figure determine the time of occurrence of that coefficient and also indicate the associated support length and frequency resolution.

Mathematica programs “`TFR-WP_Shell.nb`” and “`TFR-WP_Stand-alone.nb`” are used to render the shape and location of the coefficient’s time-frequency window within the phase plane. The first Mathematica program is a shell program for lean display and easy screen navigation, as well as avoiding any mishaps of unintentional keyboard touch. The second Mathematica program is the core program used by the first. Figure 3.2 illustrates this.

Basically, any time-frequency window shown in the figure is only of a representative locale regarding the support lengthes in time and in frequency for the corresponding wavelet packet transform coefficient. The figure also shows the effects of non-symmetrical filtering. One specific feature is that the areas of all individual windows are all equal.

Relevant Asyst shell programs for best basis identifications and renditions are “`R_Lev.shl`” and “`S_WPM.shl`”.

3.2 Wavelet blowups

Wavelets are fractal in nature, that is to say, no matter how deep we zoom into the wavelet curve the zoom-ins all show the the same characterization. If a zoom-in is call a blowup then all the blowups appear to have similar shape with the same regularity, differentiabil-

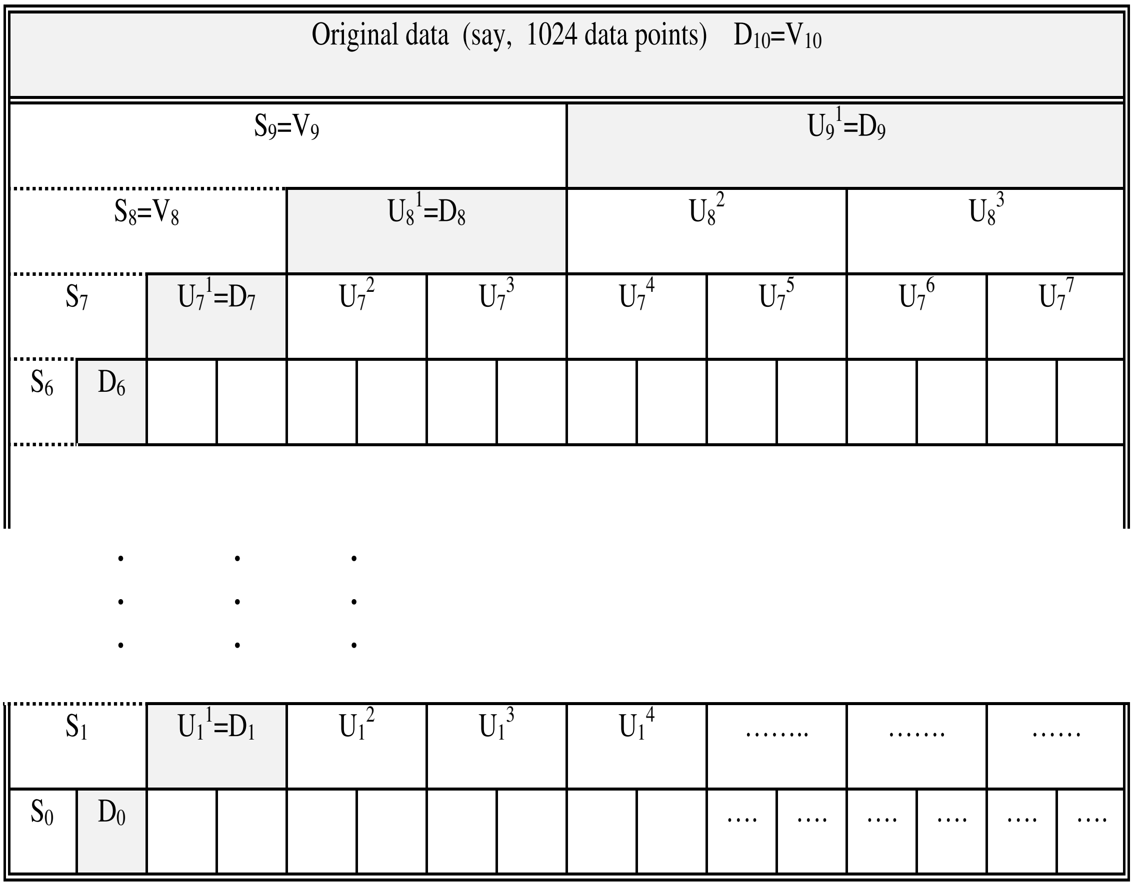
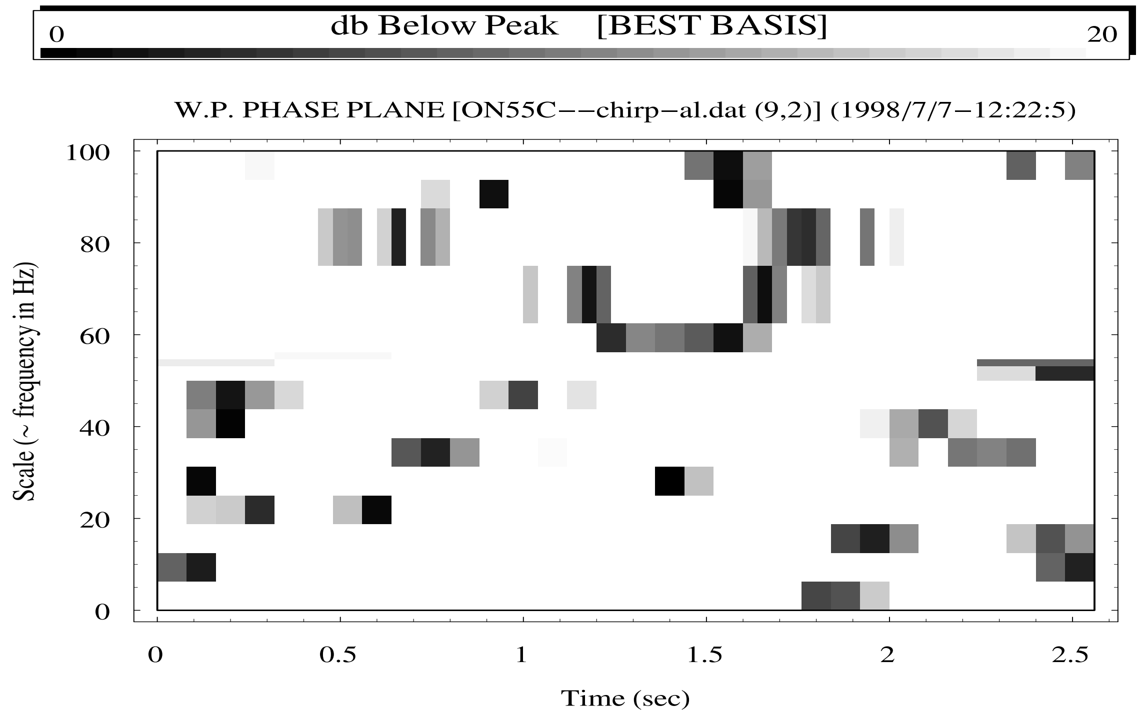


Figure 3.1: Schematic representation of the tree-like structure of the wavelet packet decomposition. S(=V in the text) and D stand for smooth and detail information, respectively. U with superscript larger than 1 stands for further decomposition of D by wavelet packets. All subscripts mean scale levels. All superscripts mean relative locations of the frequency bands for compatible subscripts.



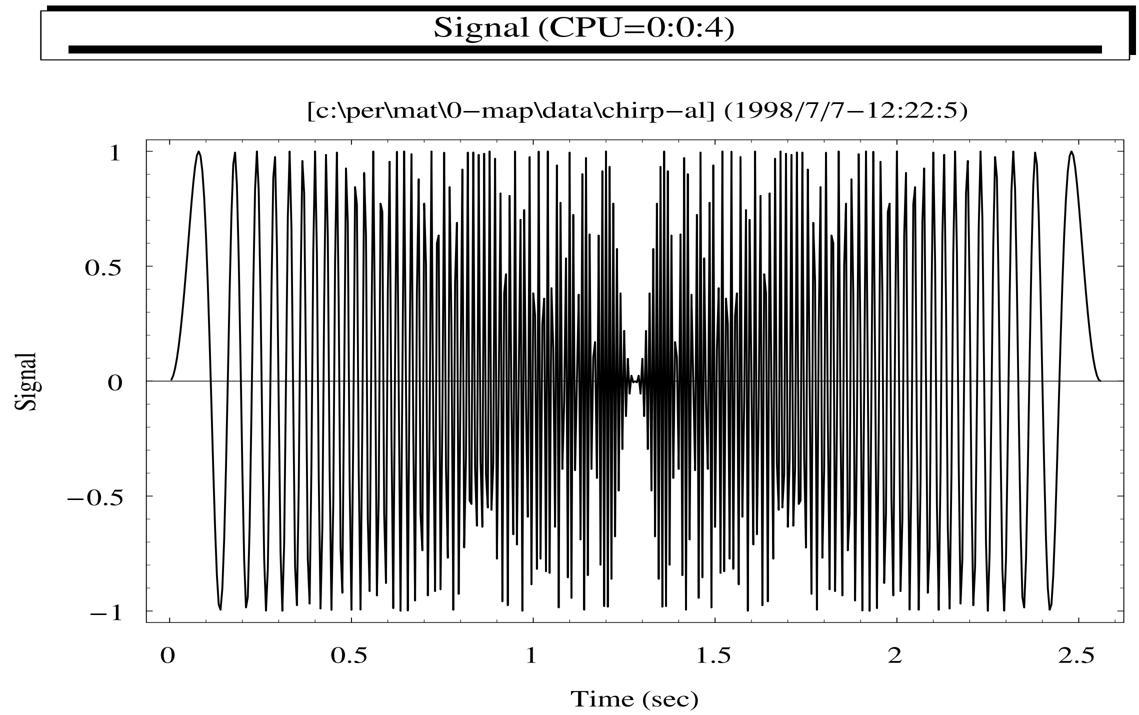


Figure 3.2: Phase plane of a wavelet packet's best basis time-frequency windows `(top)` for a linear chirp signal that is sampled under aliasing condition `(bottom)`. Here wavelet packets associated with coiflet of 30 convolution weights is used. The original signal, if not under-sampled, has linear instantaneous frequency distribution form 0 to 100 Hz. Note the non-symmetric effects and the scattering of windows due to the composite frequency bands that form the wavelet.

ity, and decaying property.

The Asyst program is written to be able to blow any wavelet constructions, such as mother and father wavelets, wavelet bases and wavelet packet bases for any point at any level. A few examples are shown in Figures 3.3 to 3.10. The Asyst shell program for this is "`R_Blo.shl`".

Here we note that wavelets with fancy analytical properties are often of bizarre wave forms and not of our choice for studying water wave related physics — either judging from they entropy values (to be discussed neat) or form their stability conditions.

Moreover, the blowup exercises here are able to hint the behaviors of several numerical and theoretical aspects of wavelet analysis, such as, edge effects, effects of finite resolution, the convergency, and the error propagation property.

Figures 3.9 and 3.10 show the blow-ups of bi-orthogonal wavelet BO31O and BO35O, respectively. Relevant data for BO31O is: Origin of wavelet curve: level 2, position 12 (i.e., element U_2^{12} in figure 3.1); Blow-up point: 150; data length: 512. Each sub-figure shows successive blow-up scale of 2^6. Here the blow-ups diverge rapidly, i.e., the wavelet fails to identify itself numerically in the refinement cascade. Relevant data for BO35O is: Origin of wavelet curve: level 2, position 12 (i.e., element U_2^{12} in figure 3.1); Blow-up point: 225; data length: 512. Each sub-figure shows successive blow-up scale of 2^6. Here the blow-ups converge but go with peculiar inclinations.

Figure 3.8 also exhibits the grouping tendency of wavelet packets. ❖

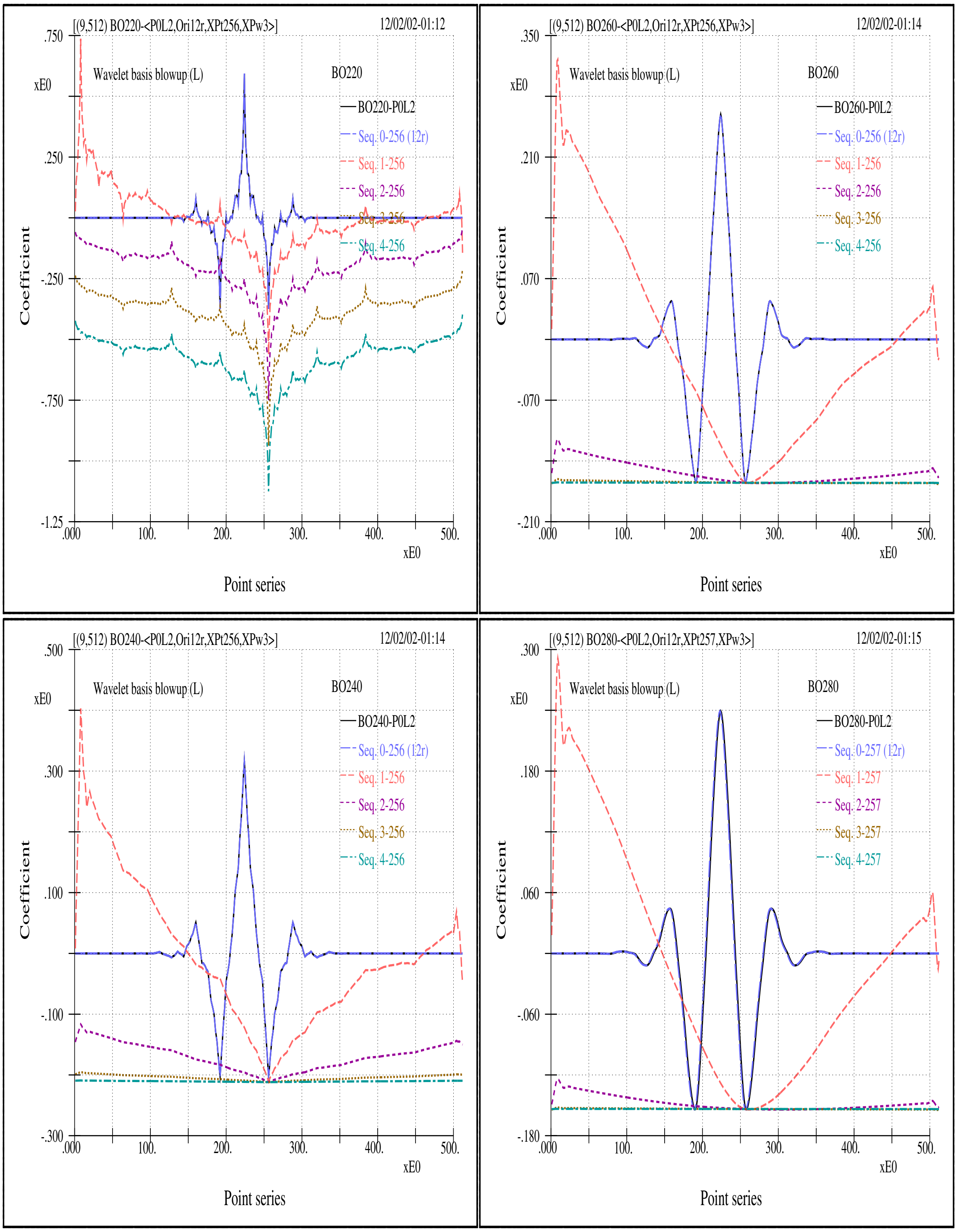


Figure 3.3: The blowups of a few wavelets of the BO2xO group. Each successive blowup scale is 2^3. The originating point of the wavelet function and the blowup location point are labeled in individual sub-figure.

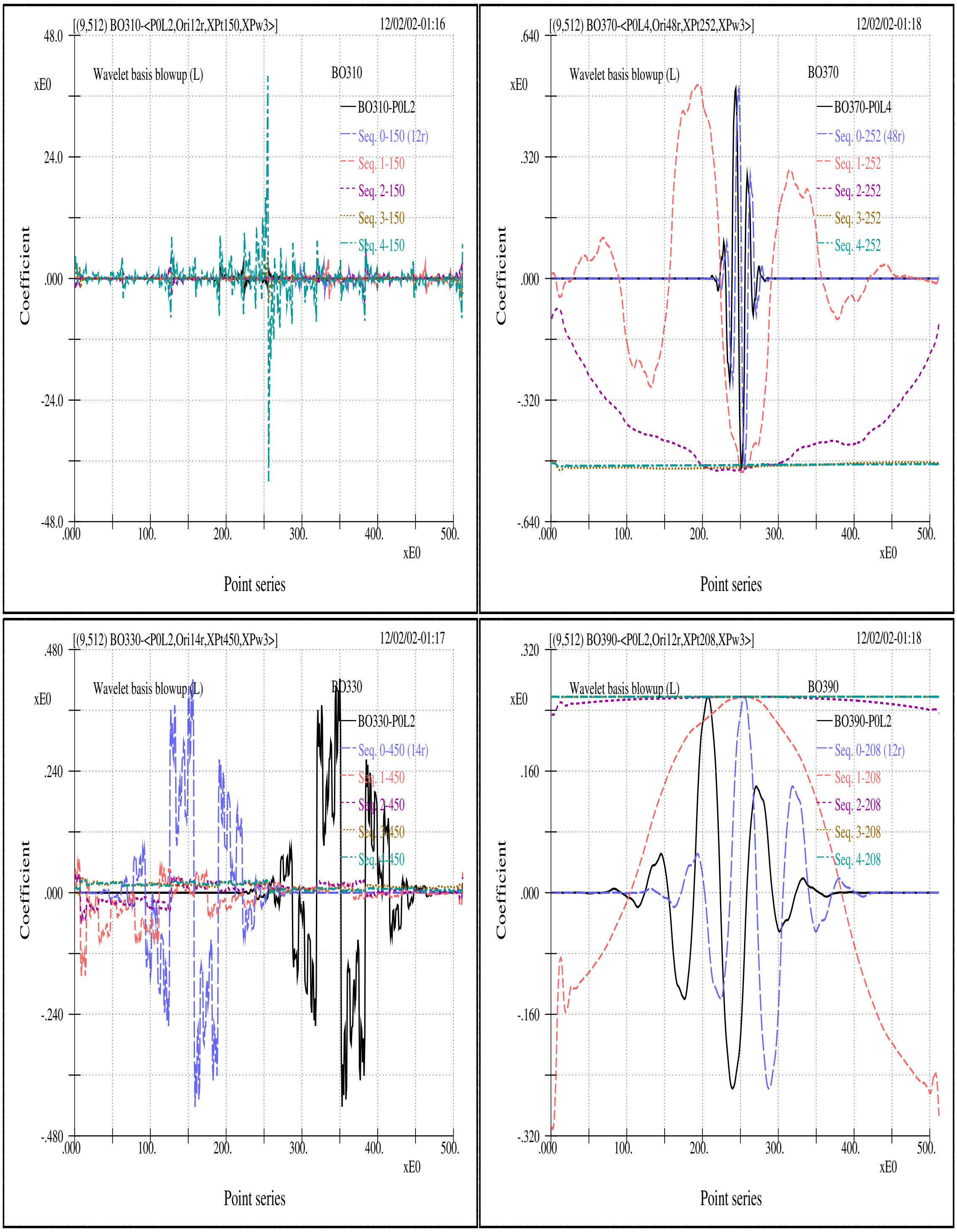


Figure 3.4: The blowups of a few wavelets of the BO3xO group.

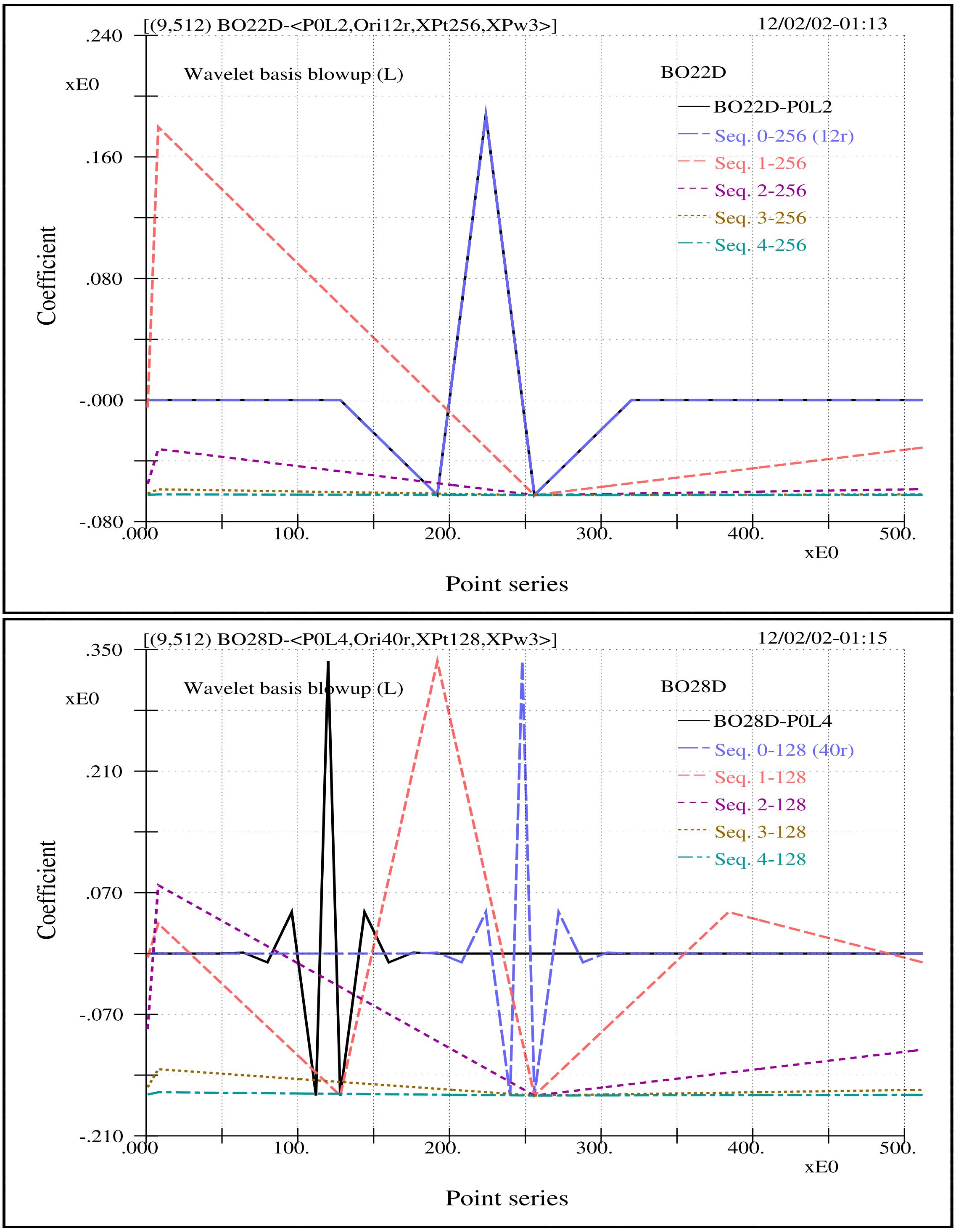


Figure 3.5: The blowups of a few wavelets of the BO2xD group.

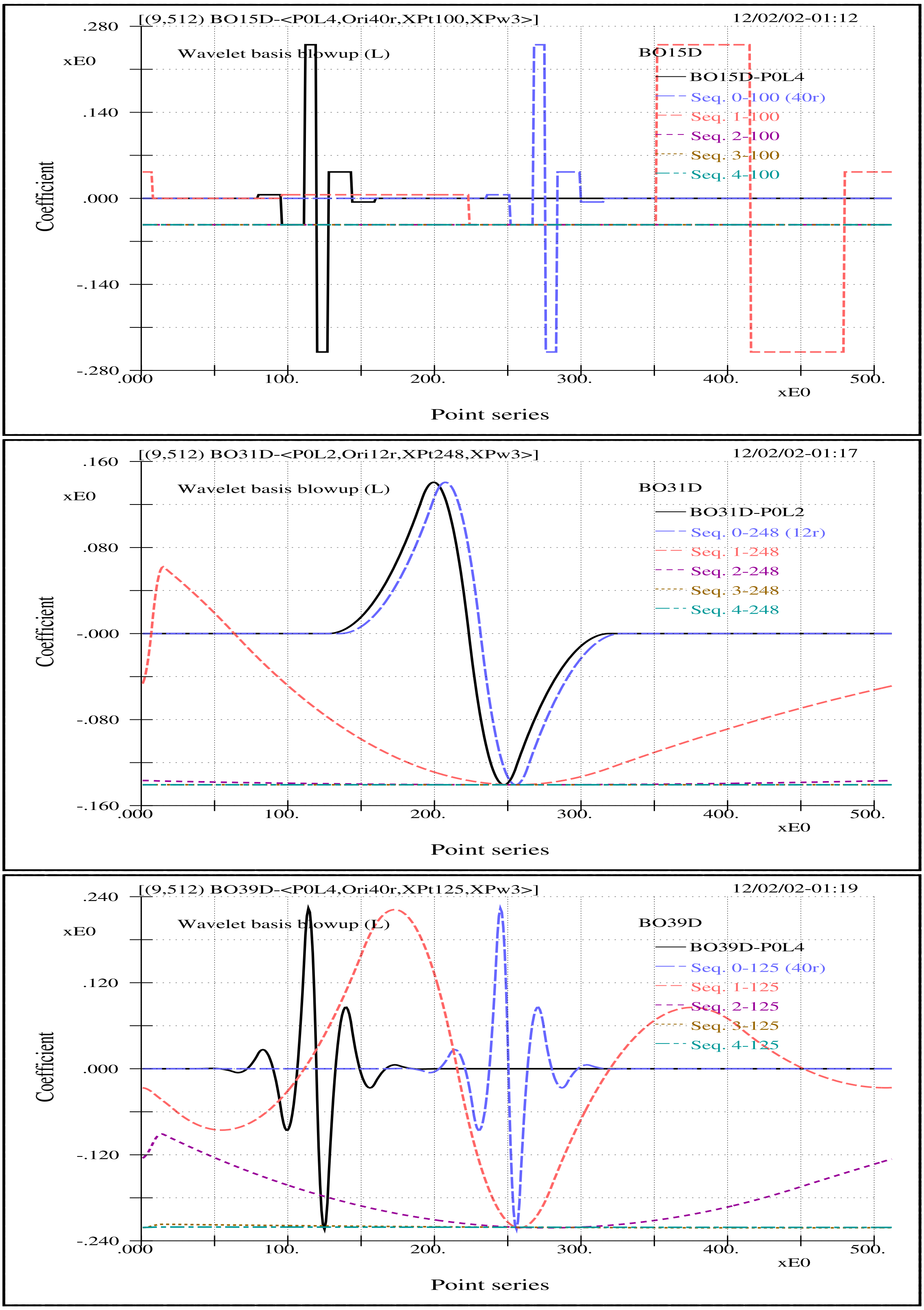


Figure 3.6: The blowups of a few wavelets of the BOxyD group.

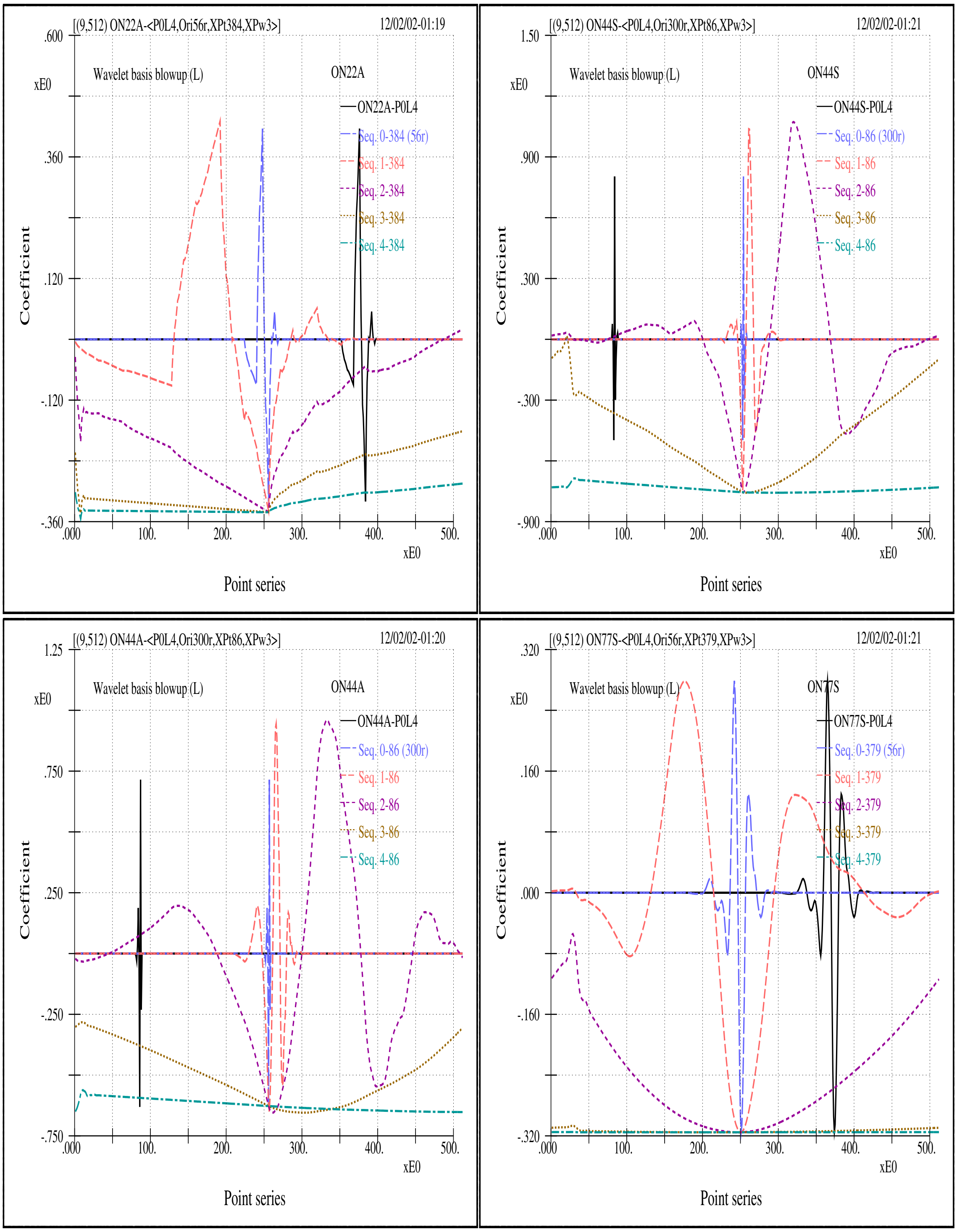


Figure 3.7: The blowups of a few wavelets of the ONxxA and ONxxS groups.

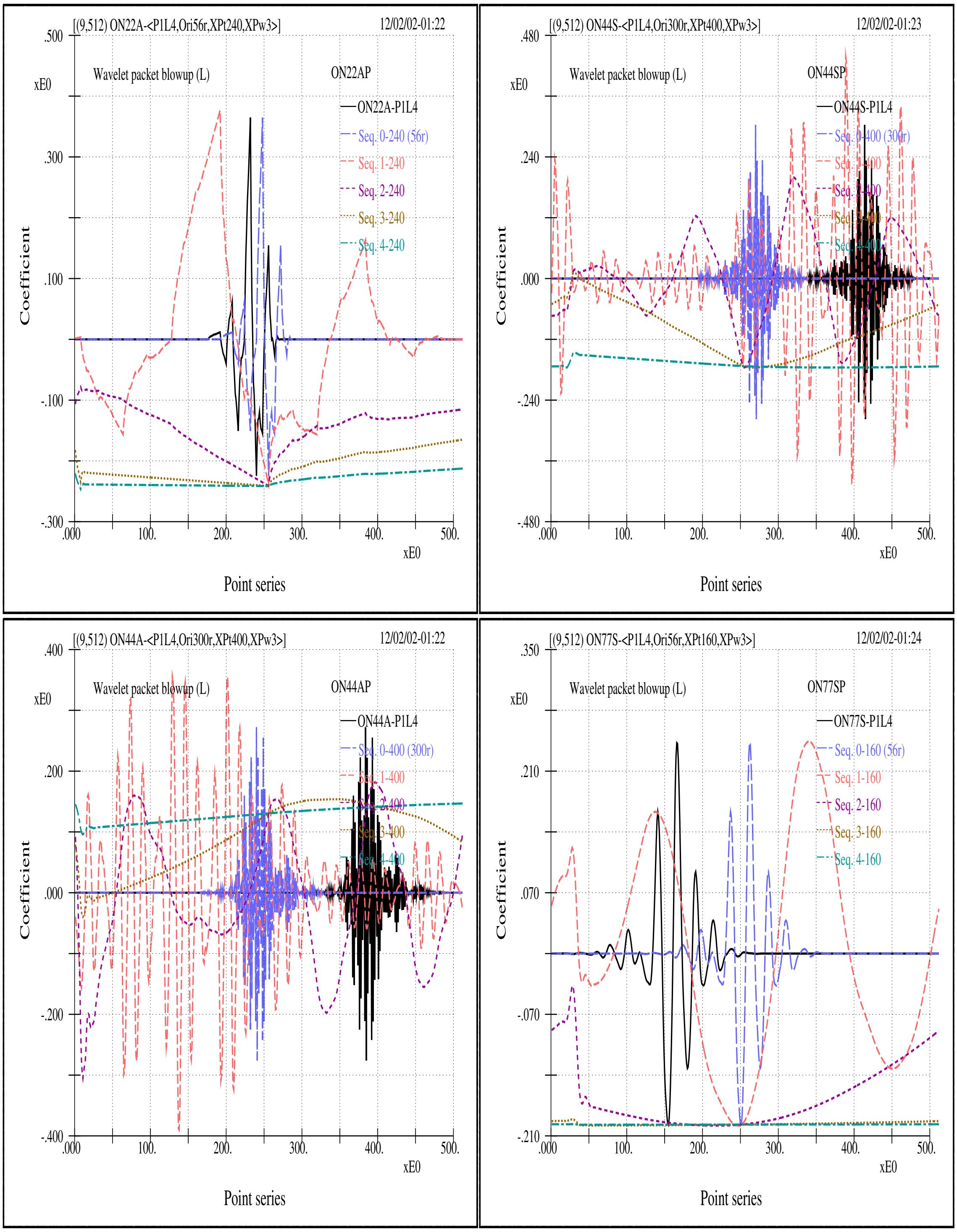


Figure 3.8: The blowups of a few wavelet packets of the ONxxA and ONxxS groups. Note the grouping tendency of the wavelet packets.

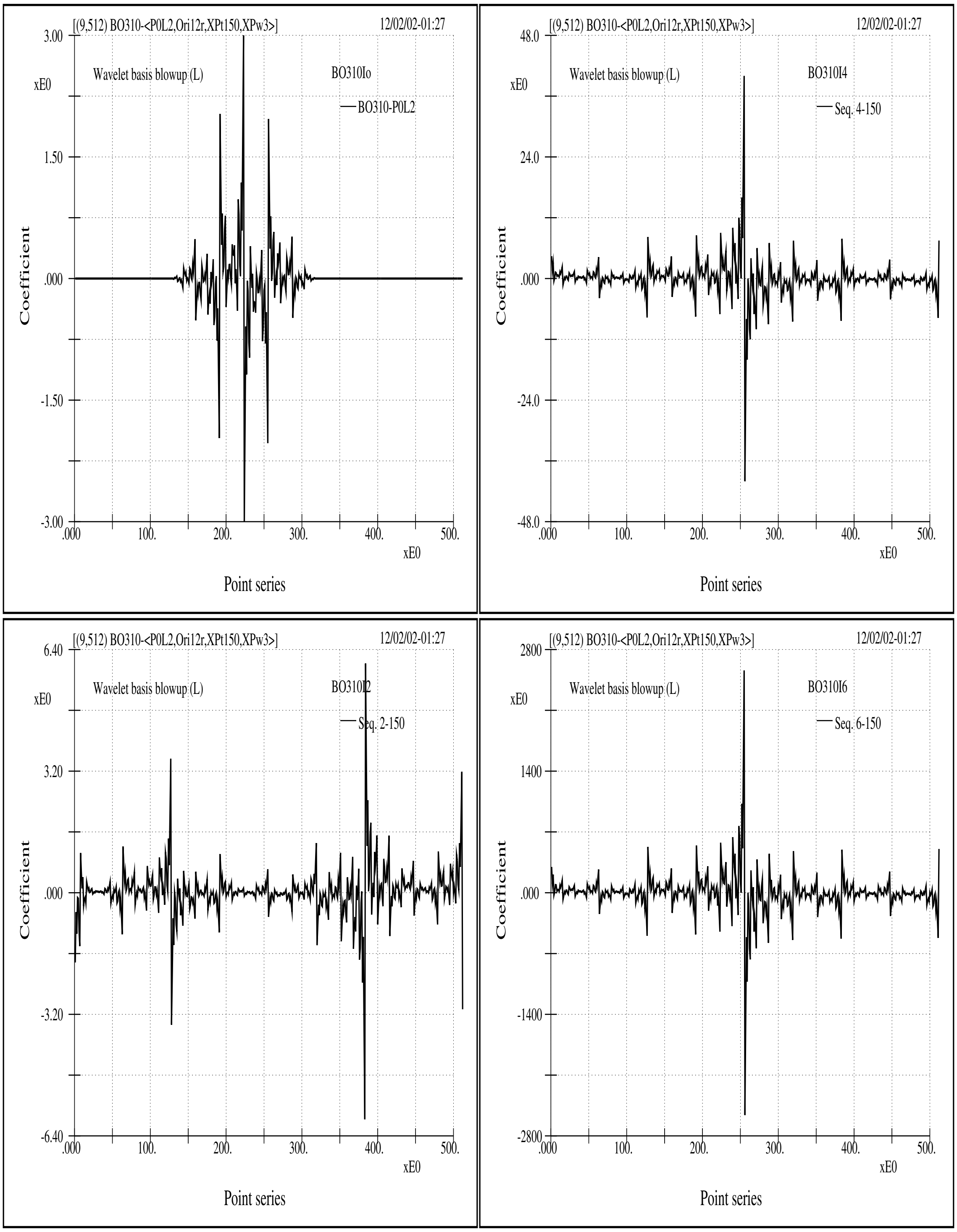


Figure 3.9: The blowups of the BO31O wavelet, noting the vast difference in the ordinate. Here successive blowup scale is 2^6.

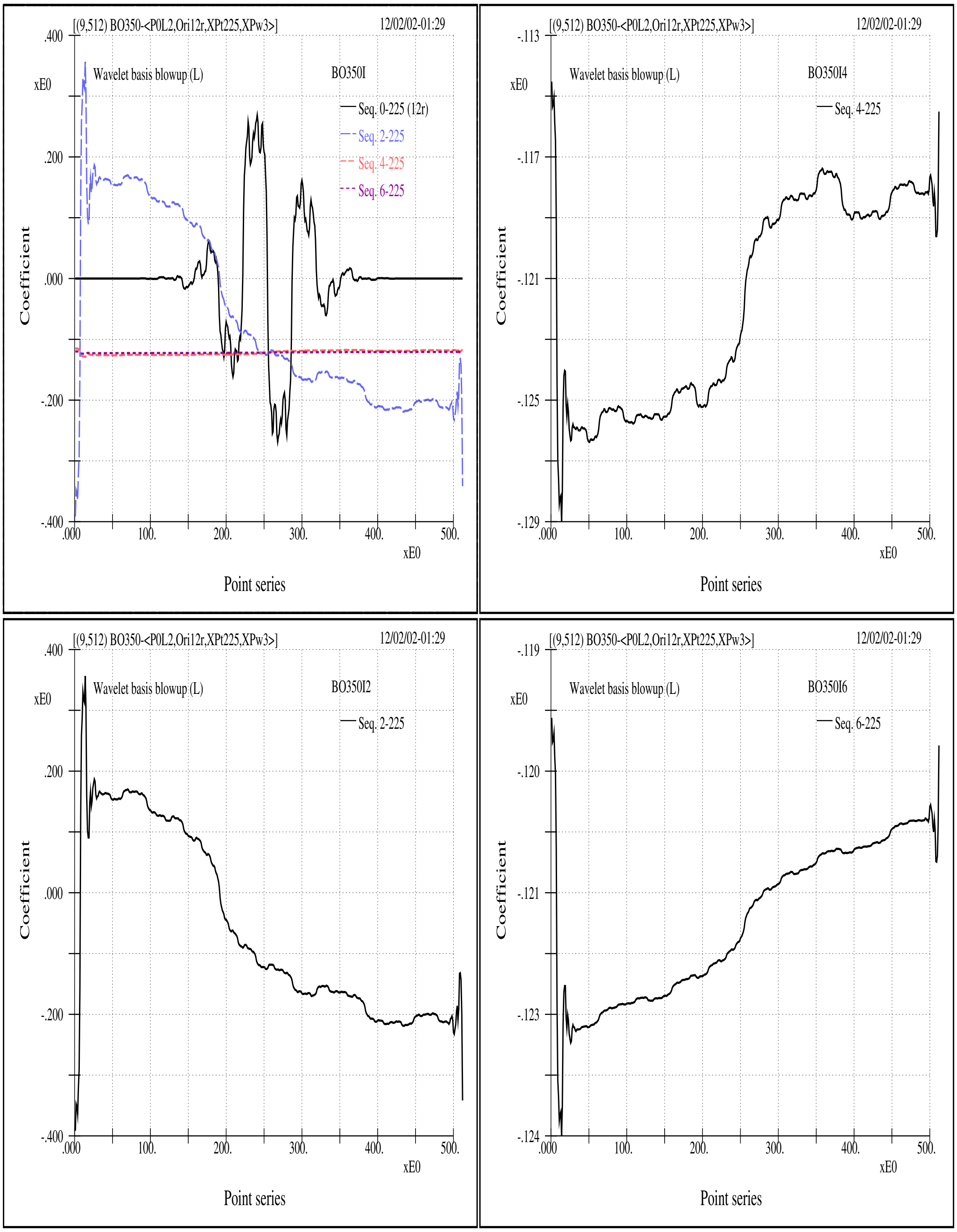


Figure 3.10: The blowups of the BO35O wavelet, noting the difference of the inclinations of the zoom-in curves. Here successive blowup scale is 2^6.

3.3 The entropy criteria

Entropy is a terminology in the statistical physics; statistics provides indication without absolute assurance. For the present context the entropy can be viewed as a measure of the "distance" between a signal and its reconstructed signal using partially truncated transform coefficients.

To avoid the somewhat mystified notions as one might get from some of the readings, it might be better, as well as easier, to give straightforward descriptions by stepping through the actual numerical process first and returning to its statistical implication later.

Let suppose that we have a 1024-point sampled data, then there is a set of 1024 wavelet coefficients (C=$\{c_i\}$). Take the absolute or squared value of these coefficients, sort them, and then divide the sequence into M (say, 100 or 200 or 300) divisions which are equally spaced from 0 to the maximum value of the coefficients. Then we have the statistics of occurrence for each division, and the distribution of these normalized occurrences is the probability density distribution or probability density function (denoted by pdf), say $\{p_1, p_2, \cdots, p_{M-1}, p_M\}$. The entropy is

$$H(p) = -\sum_i p_i \log p_i . \tag{3.1}$$

Where, when $p_i = 0$, it is assumed that $0 \log 0 = 0$, since in reality one can assumed that there exists an almost zero probability in that interval without affecting the total sum of probability; after all, what we are pursuing is only a value of statistics and the modification virtually has no influence on the norm value. If absolute values of c_i are taken, $H(p)$ is the L^1-norm entropy; if squared values are taken, it is squared L^2-norm entropy. Of course, other values of power can be used, but the squared L^2-norm, being related to the energy, is physically the most significant.

Now comes the practical aspect of this definition of entropy: let suppose two probability distribution functions sorted in a decreasing order are p and q, if p decreases faster than q, then $H(p) \leq H(q)$ [15]. The above inequality of entropy is only one-way correct; the reverse is not always true. But smaller entropy implies that more energy is gathered for a fixed number of transform coefficients at the large value end; or stated otherwise, a smaller number of wavelet coefficients can consolidate more energy. Therefore, if only a fixed percentage of coefficients is kept for reconstructing a signal, the truncated error, i.e., the distance from the total sum, thus incurred is likely to be smaller for the set of coefficients that has smaller entropy.

Other notions of entropy exist. One is sometimes referred as the geometric notion [15]. Again, the notion can better be described by giving its procedures first and the simple physical interpretation next.

By setting the number of divisions to be the same as the number of coefficients and by defining probability density to be the normalized (with respect to the total power) value of the squared wavelet coefficient, that is to say, now the total energy is $\|C\|^2 = \sum_i |c_i|^2$ and the probability density is $p_i = |c_i|^2/\|C\|^2$, one gets the alternative form of entropy by simply substituting P_i into Equation 3.1:

$$H(p) = \log \|C\|^2 - \frac{\sum_i |c_i|^2 \log |c_i|^2}{\|C\|^2}. \tag{3.2}$$

The notion here is simple: if one just put more weight on coefficients of small energy and less weight on coefficients of large energy (all coefficients being normalized), then the weighted energy is an indication of entropy. And since taking the log of a value is sort of a weighting operation and since the total energy is finite, small entropy therefore means that the number of those significant coefficients is small, or stated otherwise, more energy is concentrated in fewer coefficients.

One equivalent indicator of entropy of a pdf is the theoretical dimension $D(p)$ and is defined as [15]

$$D(p) = e^{H(p)} = \prod_i \left(p_i^{-p_i} \right). \tag{3.3}$$

As was stated, entropy does not tell how conclusive the result is. Nevertheless, our numerical results yield little ambiguity regarding the judgements or statements that we can make for our water wave related signals.

Tables 3.1 and 3.2) shows entropy values based on several setups: direct coefficient entropy related to L^2-norm based on Equation 3.3, pdf entropy related to L^2-norm with 300 (column 2) and 200 (column 4) divisions, and pdf entropy related to L^1-norm based on Equation 3.1 (column 3). Theoretical dimension for one of the setups is also given (column 5). The tables use raw data of a wind-wave signal from a wave tank experiment.

The relevant shell programs for this is "`S_Ent.shl`", and "`S_Nor.shl`".

3.4 The phase of the wavelet Characteristic function m_0

The entropy results are of a statistical approach, and they by no means touches any analytical insight of the various function bases. Herein, by studying the phase distribution of a wavelet characterizing function for each basis, the analytical essence that gives rise the practical usefulness of a function basis is shown to be the fulfillment of a linear phase of the characterizing function.

Following the convention used by Daubechies [5], the wavelet characterizing function is termed as the $m_0(\xi)$ function, which is the kernel of individual wavelet and has the following mathematical content:

A multiresolution analysis consists of a sequence of the closed subspaces V_j of the nested ladder,

$$\cdots V_2 \subset V_1 \subset V_0 \subset V_{-1} \subset V_{-2} \subset \cdots, \tag{3.4}$$

and satisfies the requirement

$$f \in V_j \Longleftrightarrow f(2^j \cdot) \in V_0. \tag{3.5}$$

The invariance of V_0 under integer translations states that

$$f \in V_0 \Longrightarrow f(\cdot - n) \in V_0 \text{ for all } n \in \mathbb{Z}. \tag{3.6}$$

Now comes the main statement that there exists $\phi \in V_0$ so that

$$\{\phi_{0,n};\, n \in \mathbb{Z}\} \text{ is an orthonormal or Riesz basis in } V_0, \tag{3.7}$$

where, for all $j, n \in \mathbb{Z}$, $\phi_{j,n}(x) = \sqrt{2^{-j}}\phi(2^{-j}x - n)$. Here the ϕ is often called the scaling function of the multiresolution analysis. Furthermore, for the $\{\phi_{j,n};\, j, n \in \mathbb{Z}\}$ there exists its counterpart wavelet basis $\{\psi_{j,k};\, j, k \in \mathbb{Z}\}$, $\psi_{j,k}(x) = \sqrt{2^{-j}}\psi(2^{-j}x - k)$, such that

$$P_{j-1}f = P_j f + \sum_{k\in\mathbb{Z}} \langle f, \psi_{j,k}\rangle \psi_{j,k}. \tag{3.8}$$

Since $\phi \in V_0 \subset V_{-1}$ and $\phi_{-1,n}$ are basis in V_{-1}, we have

$$\phi = \sum_n h_n \phi_{-1,n}, \tag{3.9}$$

with

$$h_n = \langle \phi, \phi_{-1,n}\rangle. \tag{3.10}$$

We therefore have

$$\phi(x) = \sqrt{2}\sum_n h_n \phi(2x-n) \tag{3.11}$$

or

$$\widehat{\phi}(\xi) = \frac{1}{\sqrt{2}}\sum_n h_n e^{-in\xi/2}\widehat{\phi}(\xi/2). \tag{3.12}$$

In an alternative form

$$\widehat{\phi}(\xi) = m_0(\xi/2)\widehat{\phi}(\xi/2), \tag{3.13}$$

where

$$m_0(\xi) = \frac{1}{\sqrt{2}}\sum_n h_n e^{-in\xi}. \tag{3.14}$$

Suffice it to say that the $m_0(\xi)$ function is comprised of the summation of the wavelet construction convolution filters (or filter weights corresponding to the support length of the wavelet) multiplied by the complex exponential functions of their individual scales, and it is intrinsic to the transcendental formulations of the mother wavelet and the two-scale equation.

Figures 3.11 to 3.18 show the phase distributions of all the covered wavelet categories. The relevant shell program here is “`R_Pha.shl`”. ❖

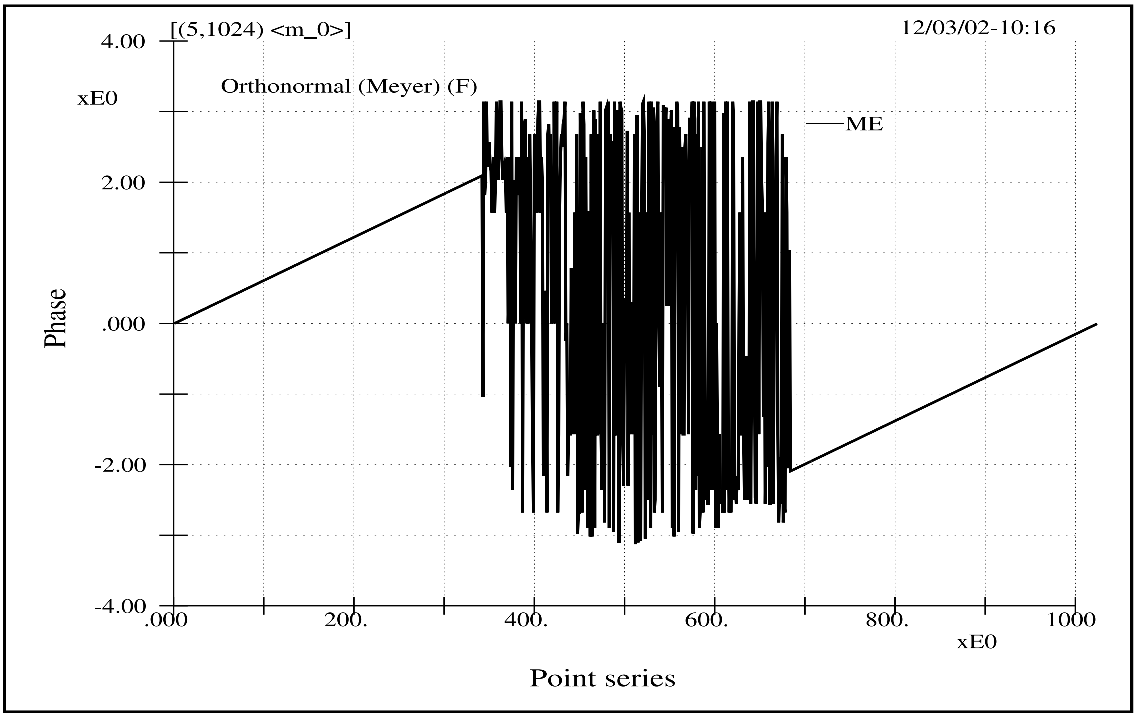


Figure 3.11: The phase distribution of the m_0 function of the Meyer wavelet.

Table 3.1: Entropy of orthonormal and semi-orthogonal wavelet coefficients as well as spectral coefficients under various statistic criteria.

Wavelet	L**2 coefficient entropy (0 division)	L**2 probability entropy (300 divisions)	L**1 probability entropy (300 divisions)	L**2 probability entropy (200 divisions)	Theotetical dimension (L**2 300 divisions)
B&L	4.691	1.330	3.417	1.179	3.782
Meyer	4.647	1.294	3.365	1.132	3.646
SO3O	4.833	1.669	3.756	1.488	5.307
SO3D	1.823	0.219	1.306	0.172	1.245
Spectrum	2.809	0.270	3.044	0.244	1.310
ON22A	4.993	1.761	3.891	1.516	5.815
ON33A	4.773	1.384	3.499	1.225	3.975
ON44A	4.790	1.517	3.596	1.363	4.559
ON55A	4.819	1.553	3.631	1.367	4.727
ON66A	4.790	1.373	3.456	1.203	3.946
ON77A	4.675	1.355	3.461	1.203	3.877
ON88A	4.645	1.229	3.283	1.082	3.418
ON99A	4.719	1.412	3.501	1.252	4.106
ON00A	4.787	1.423	3.511	1.244	4.149
ON44S	4.835	1.461	3.557	1.281	4.311
ON55S	4.758	1.492	3.576	1.298	4.426
ON66S	4.754	1.402	3.501	1.225	4.065
ON77S	4.751	1.336	3.331	1.188	3.804
ON88S	4.714	1.366	3.481	1.224	3.918
ON99S	4.755	1.469	3.570	1.288	4.345
ON00S	4.635	1.278	3.378	1.134	3.591
ON11C	4.938	1.696	3.832	1.457	5.452
ON22C	4.827	1.468	3.520	1.284	4.342
ON33C	4.756	1.488	3.573	1.333	4.427
ON44C	4.690	1.297	3.337	1.157	3.658
ON55C	4.644	1.309	3.405	1.154	3.703

Table 3.2: Entropy of bi-orthogonal wavelet coefficients under various statistic criteria.

Wavelet	L**2 coefficient entropy (0 division)	L**2 probability entropy (300 divisions)	L**1 probability entropy (300 divisions)	L**2 probability entropy (200 divisions)	Theoretical dimension (L**2 300 divisions)
BO11O	5.395	2.623	4.502	2.299	13.777
BO11D	5.395	2.623	4.502	2.299	13.777
BO13O	4.943	1.806	3.883	1.627	6.084
BO13D	5.266	2.371	4.373	2.053	10.708
BO15O	4.866	1.678	3.755	1.495	5.357
BO15D	5.227	2.291	4.327	1.987	9.882
BO22O	5.282	2.362	4.363	2.083	10.609
BO22D	4.434	1.181	3.284	1.034	3.257
BO24O	4.963	1.862	3.985	1.634	6.438
BO24D	4.359	1.090	3.220	0.962	2.975
BO26O	4.881	1.703	3.835	1.492	5.490
BO26D	4.332	1.064	3.174	0.940	2.899
BO28O	4.857	1.624	3.782	1.452	5.073
BO28D	4.318	1.069	3.157	0.941	2.914
BO31O	5.824	3.174	4.741	2.835	23.894
BO31D	4.377	1.058	2.655	0.936	2.880
BO33O	5.084	2.001	4.062	1.756	7.393
BO33D	4.205	1.102	2.827	0.965	3.011
BO35O	4.850	1.697	3.847	1.506	5.457
BO35D	4.125	1.026	2.776	0.908	2.789
BO37O	4.790	1.658	3.821	1.442	5.247
BO37D	4.106	0.986	2.737	0.873	2.679
BO39O	4.776	1.660	3.835	1.432	5.258
BO39D	4.098	0.967	2.713	0.866	2.629

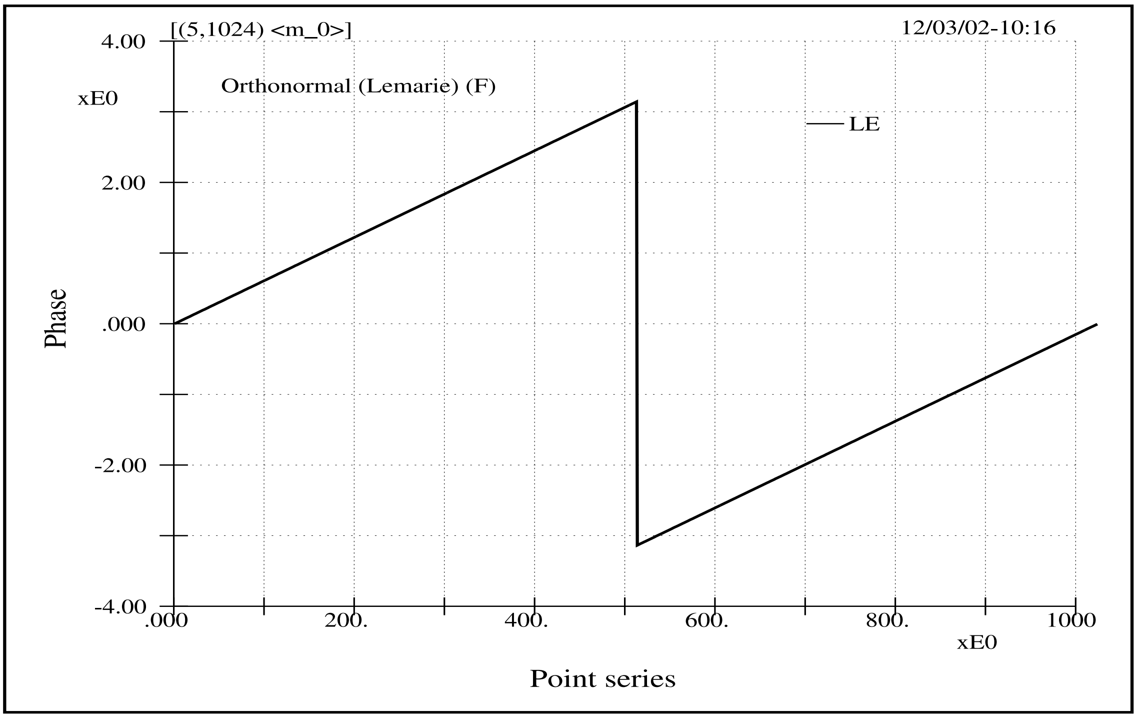


Figure 3.12: The phase distribution of the m_0 function of the Battle and Lemarié wavelet, noting the difference from that of Meyer wavelet.

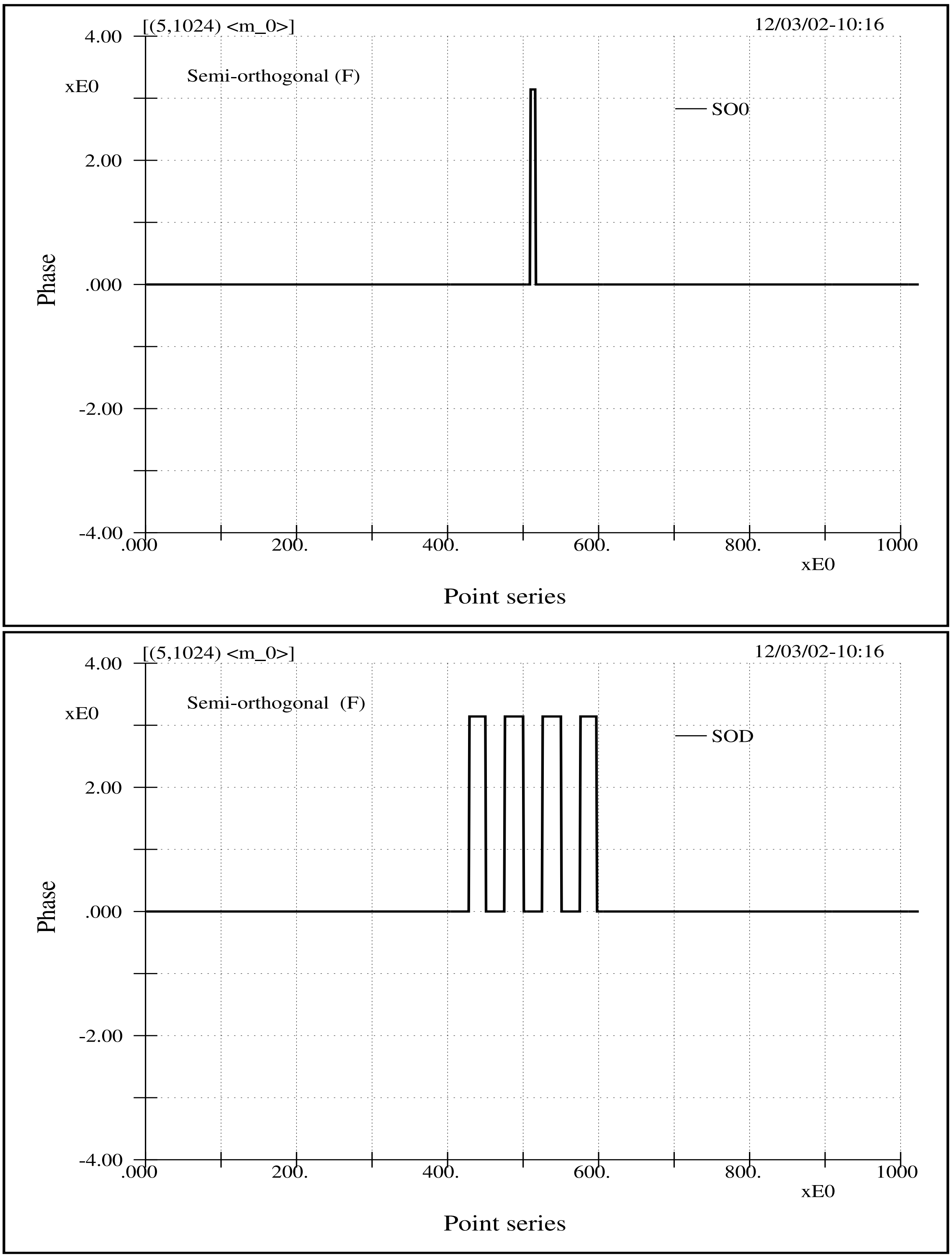


Figure 3.13: The phase distributions of the m_0 functions of the semi-orthogonal wavelet and its dual.

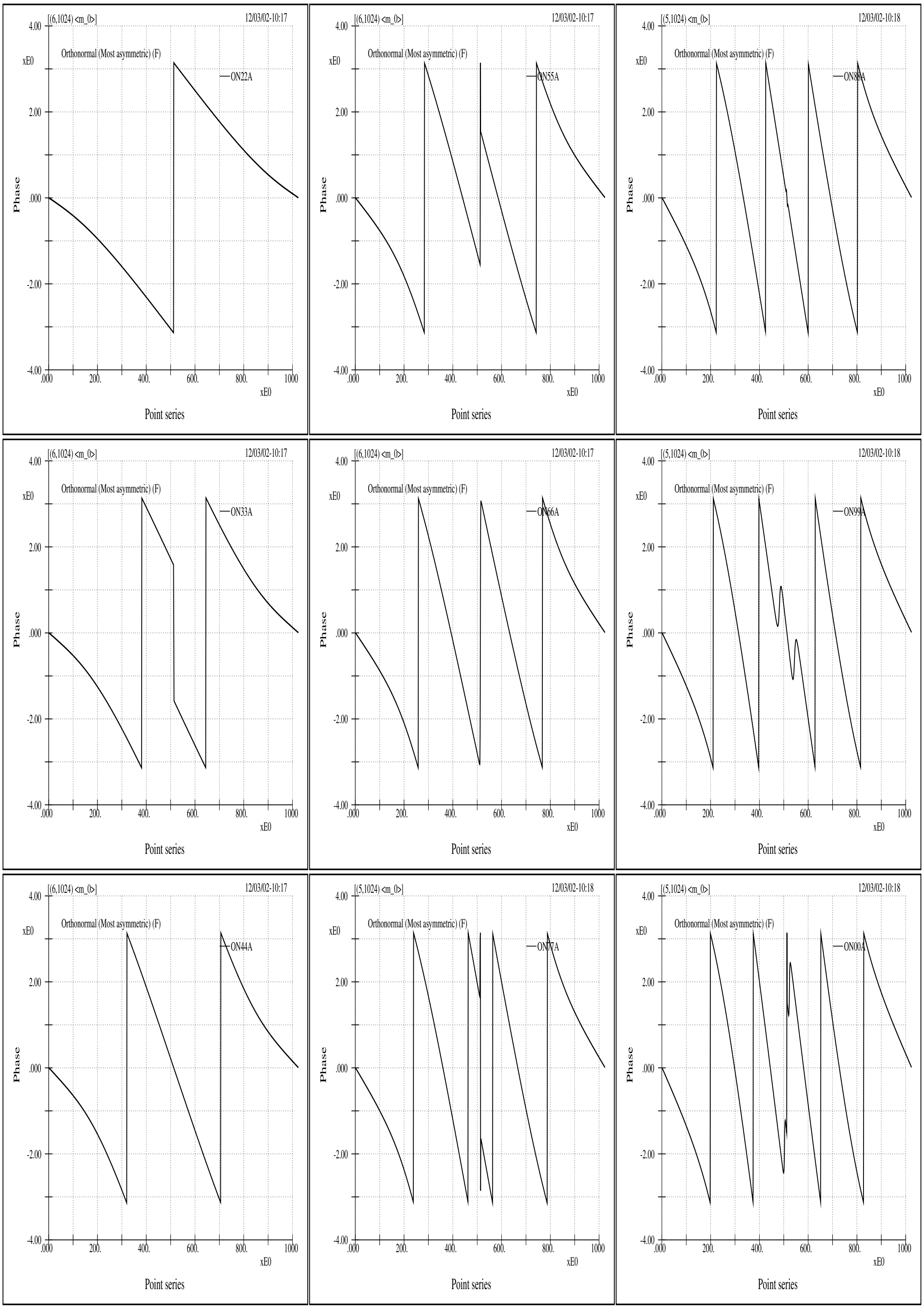


Figure 3.14: The phase distributions of the m_0 functions of the wavelets of the most asymmetric group.

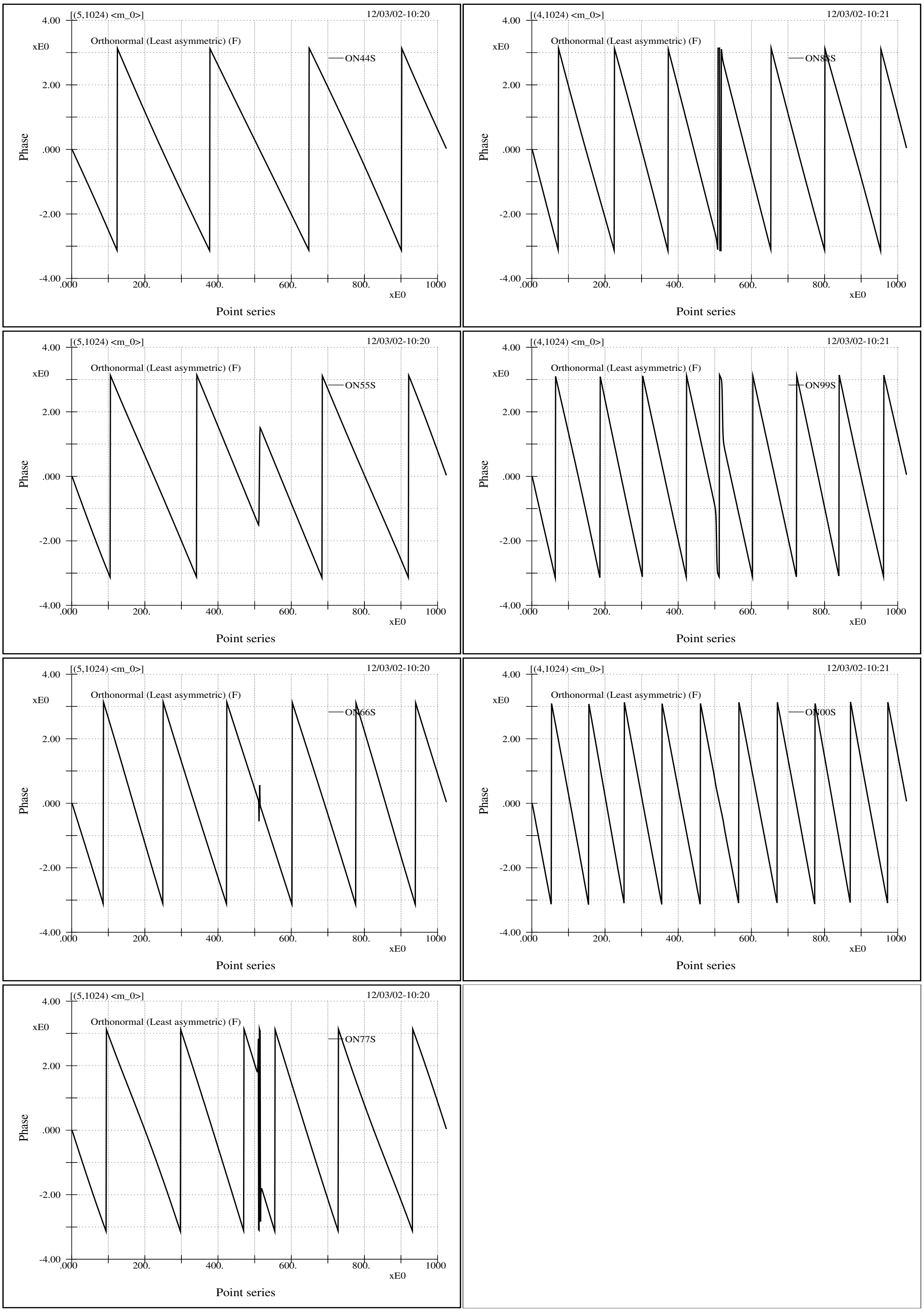


Figure 3.15: The phase distributions of the m_0 functions of the wavelets of the least asymmetric group.

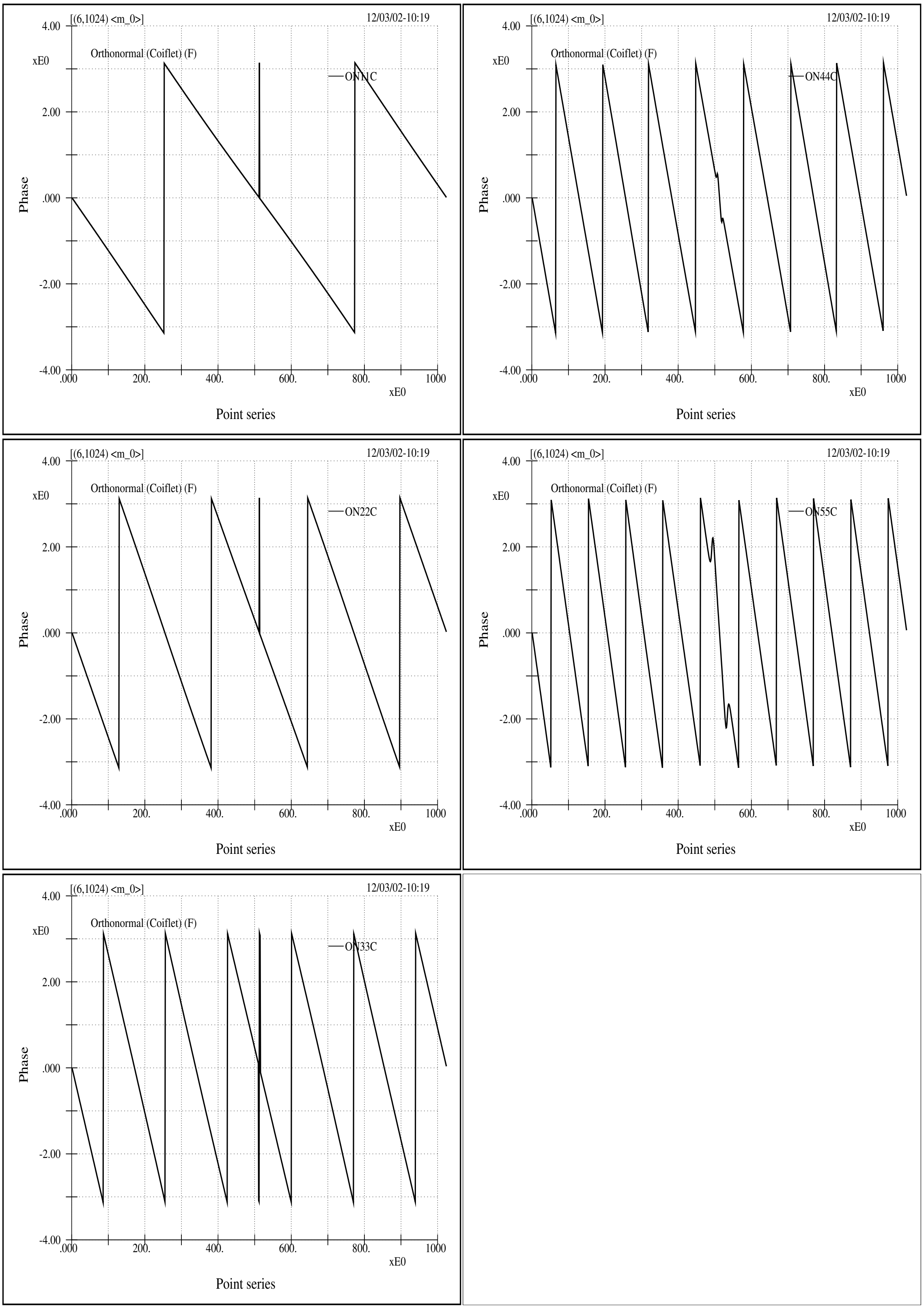


Figure 3.16: The phase distributions of the m_0 functions of the coiflets.

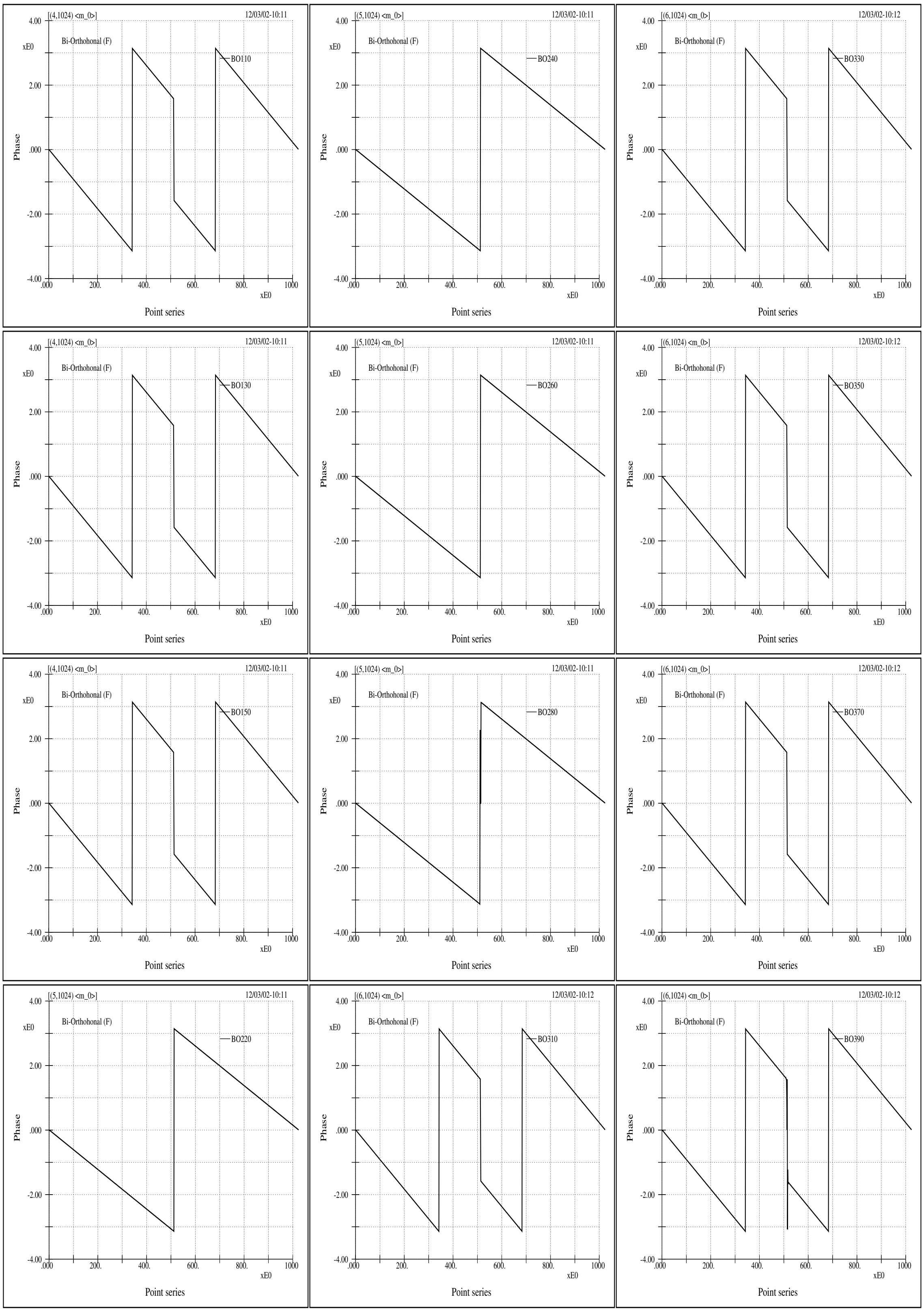


Figure 3.17: The phase distributions of the m_0 functions of the bi-orthogonal wavelets.

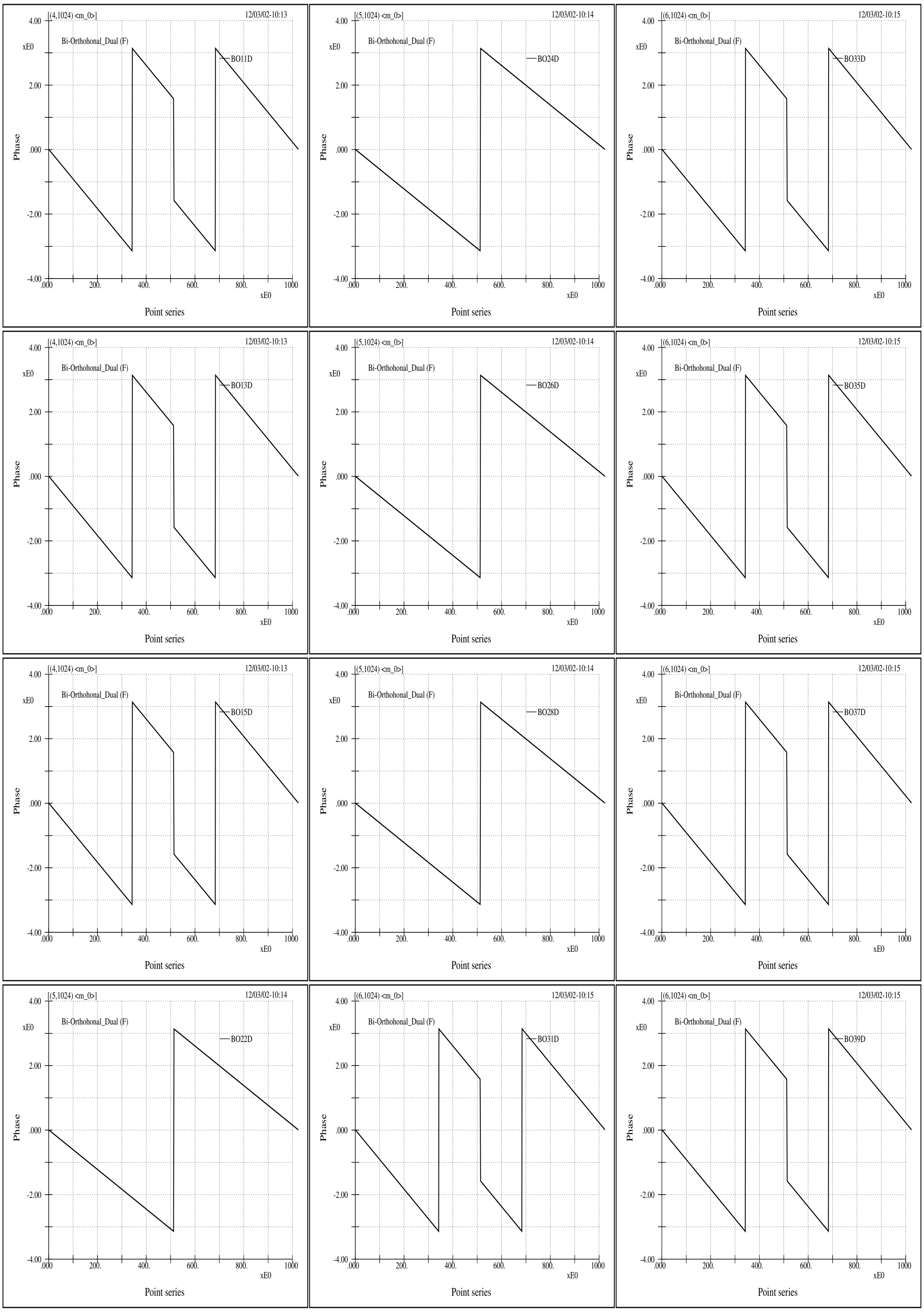


Figure 3.18: The phase distributions of the m_0 functions of the bi-orthogonal wavelets.

Chapter 4

Continuous Wavelet Transforms

4.1 Introduction

It should be emphasized that the discrete and the continuous wavelet transforms (DWT and CWT) be treated as two different entities. Since, unlike the Fourier cases where the discrete and the continuous transforms not only are dealing with the same basis but also are deploying basically the same formulations, while DWT and CWT generally refer to two quite different methodologies which not only focus on their individual function bases but also involve different data treatment schemes.

For the CWT the programming mainly covers the following topics: The continuous wavelet perspective of the optimum discrete wavelet basis, The demand for better physics — the continuous wavelet transform using adapted time-frequency windows, and, A proposed wavelet variant with superiority over the Morlet wavelet in ridge extractions.

4.2 The continuous counterpart of the optimum discrete basis

Let the Gaussian function be

$$g_\alpha(t) = \frac{1}{2\sqrt{\pi\alpha}} e^{-\frac{t^2}{4\alpha}}, \tag{4.1}$$

where α is a representative value of the second moment of the Gaussian function and the constants is for the purpose of normalization, the modulated Gaussian is

$$G_{b,\omega}^{\alpha}(t) = e^{i\omega t} g_{\alpha}(t - b). \tag{4.2}$$

And the Gabor transform of a function f is

$$\left(\mathcal{G}_b^{\alpha} f\right)(\omega) = \langle f, G_{b,\omega}^{\alpha} \rangle = \int_{-\infty}^{\infty} f(t) e^{-i\omega t} g_{\alpha}(t - b) \mathrm{d}t. \tag{4.3}$$

As is stated by Daubechies ([5]) that the Morlet wavelet is almost identical to a modulated Gaussian, and as is given by Chui ([2]) that a modulated Gaussian matches almost exactly with cardinal B-spline wavelet of order greater than or equal to three, we therefore have an extremely natural transition from the identified discrete best wavelet basis to the Morlet wavelet associated with CWT.

Note that Chui's results ([2]) are, for $m \geq 3$, the even order ψ_m's (such as the cubic spline wavelet ψ_4) match almost exactly with

$$\mathrm{Re} G_{b,\omega}^{\alpha}(t) = (\cos \omega t) g_{\alpha}(t - b), \tag{4.4}$$

and the odd order ones match with

$$\mathrm{Im} G_{b,\omega}^{\alpha}(t) = (\sin \omega t) . g_{\alpha}(t - b) \tag{4.5}$$

4.3 The continuous wavelet transform using adapted time-frequency windows

The Morlet wavelet is the following complex function:

$$\psi(t) = \pi^{-1/4} (e^{-i\omega_0 t} - e^{-\omega_0^2/2}) e^{-t^2/2}, \tag{4.6}$$

in which ω_0 is a constant related to carrier frequency and the term $e^{-\omega_0^2/2}$ justifies the wavelet admissability condition. The Fourier transform of the Morlet wavelet is almost a shifted Gaussian and is given by

$$\widehat{\psi}(\omega) = \pi^{-1/4}[e^{-(\omega-\omega_0)^2/2} - e^{-\omega^2/2}e^{-\omega_0^2/2}]. \tag{4.7}$$

In addition to the general meaning of the modulation frequency, the ω_0 has the physical implication of the amplitude ratio r — the ratio between the second highest peak and the first highest peak of $\psi(t)$ — i.e.,

$$r = \psi(t_2)/\psi(0), \tag{4.8}$$

in which t_2 is the abscissa of the second highest peak. The exact value of t_2 can be obtained by solving numerically the transcendental equation derived from the derivative of the ψ function, but a fairly good estimate is obtained by simply dropping the second term of the above complex function since the second term is generally five order of magnitude less than the maximum value of the first term, i.e.,

$$\omega_0 \approx \frac{2\pi}{t_2} \approx \pi\left(-\frac{2}{\ln r}\right)^{1/2}. \tag{4.9}$$

The higher the ω_0 is, the smaller the ratio r becomes. If ω_0 is constant, then the ratio r for different wavelet dilations or scales keeps constant too. Here comes the core question: whether constituent wave components of different scales and time spans all possess this fixed decay feature? To show that this is not true, let us examine the composite water wave system that is with viscous damping.

For deep water waves with a clean surface the energy losses due to viscous dissipation arise almost entirely from the straining of the irrotational motion in the water column, and the part of contribution from viscous stresses in the surface layer is negligible. It was shown [6, 13] that the time rate of change of the energy density is

$$\dot{E} = -2\mu\sigma^2 {a_w}^2 k, \tag{4.10}$$

where μ, σ, a_w, and k are the dynamic viscosity of the water, the wave frequency, wave amplitude, and wave number, respectively. Since in deep water $E = (2k)^{-1}\rho\sigma^2 {a_w}^2$, where ρ is the water density, the attenuation coefficient

$$\gamma_\nu = -\frac{\dot{E}}{2E} = 2\nu k^2, \tag{4.11}$$

where ν is the kinematic viscosity of the water. Therefore the energy density of the wave evolves as

$$E = C_1 e^{-2\gamma_\nu t}, \tag{4.12}$$

where C_1 is a constant, and the amplitude decreases with time in accordance with

$$a_w = \sqrt{\frac{C_1 2k}{\rho\sigma^2}} e^{-\gamma_\nu t} = C_2 e^{-2\nu k^2 t}, \tag{4.13}$$

where C_2 is a constant if σ does not vary. Comparing the decay of wave amplitude of Morlet wavelet with the decay of the physical model, one sees both similarity and dissimilarity. The similarity is that the attenuation coefficients in both models have inverse square dependence on scales — the former in $(1/a)^2$ and the latter in k^2. The dissimilarity is in the time dependence of the exponent in the exponential — in Morlet wavelet it is in t^2 dependence, while in the physical model it is in linear dependence. It is therefore anticipated that Morlet wavelets based on a fixed modulation shape are not good representations of water waves of different scales. Or stated otherwise, basis functions originating form a single mother Morlet wavelet do not form a good basis.

Now the situation is clear: the constant ω_0 either overestimates the viscous decay of water waves at the low-frequency end or, otherwise, under-estimates those at the high-frequency end. Form a practical judgement of the modulation curves, it is quite reasonable to argue that the deviation is probably more significant for waves with a longer life span when a standard r value of Morlet wavelet, i.e., $r = 0.5$, is assumed. The perceptions here provide the footing for the present adaptation — with different values of amplitude ratio r for different wave scales we are really attemptimg to simulate the evolution process with a more realistic condition. The expansion or contraction of wavelet support length for a specific scale just reflects the devising of flexible constructions of time-frequency windows, and adjusting r is in turn using a variable ω_0. The general guideline is to use a comparatively larger ω_0 (associated with a narrower frequency band) for waves of a longer time support; and vice versa, a comparatively smaller ω_0 (a wider frequency band) for a shorter life span. Here it naturally comes to assume the ω_0 to be a function of scale, i.e., $\omega_0 = \omega_0(a)$. And the varying shapes and sizes of the time-frequency windows are now determined by

$$\psi_a\left(\frac{t-b}{a}\right) = \pi^{-1/4}\left[e^{-i\frac{\omega_0(a)}{a}(t-b)} - e^{-\omega_0(a)^2/2}\right]e^{-\frac{(t-b)^2}{2a^2}}. \tag{4.14}$$

For our experiments in which water waves were measured in a wind blowing oval tank the reasonable frequencies should lie between 1.5 and 10 Hz. We mentioned that the Morlet wavelet is likely to overestimate the decay of longer waves in the long run; therefore, relative to higher frequency waves, we should reduce the decay parameter ω_0 for low frequency ones. Based on this understanding we heuristically assume

$$\mathrm{Erfc}\left[\frac{4}{10}\left(\frac{\omega_0}{a} + 2.5\right) - 2\right]3 + 5 = a\omega \tag{4.15}$$

where Erfc is the complimentary error function and ω is the carrier frequency. This equation may be modified according to the type of signal studied or according to the frequency range of one's interest. Figure 4.1 shows the curve of the function. The logic for the choice of its constants is self explained in the attached program piece. Figure 4.2 shows a numerical experiment with and without the adaptation.

The Asyst shell program is “`S_WVT.shl`”. And two relevant Mathematica programs are “`TFR-WV_Shell.nb`” and “`TFR-WV_Stand-alone.nb`”. Note that these two Mathematica files are also used for the wavelet variant to be given later. ❖

```
obeg=11.;  oend=5.;
fcenter=2.5;  fdilation=10/4;   fshift=2. ;

perfc=Plot[ Erfc[(1/fdilation)* (freq +fcenter)-fshift]*
                             (obeg-oend)/2+ oend, {freq, -2.5, 8.5} ]
```

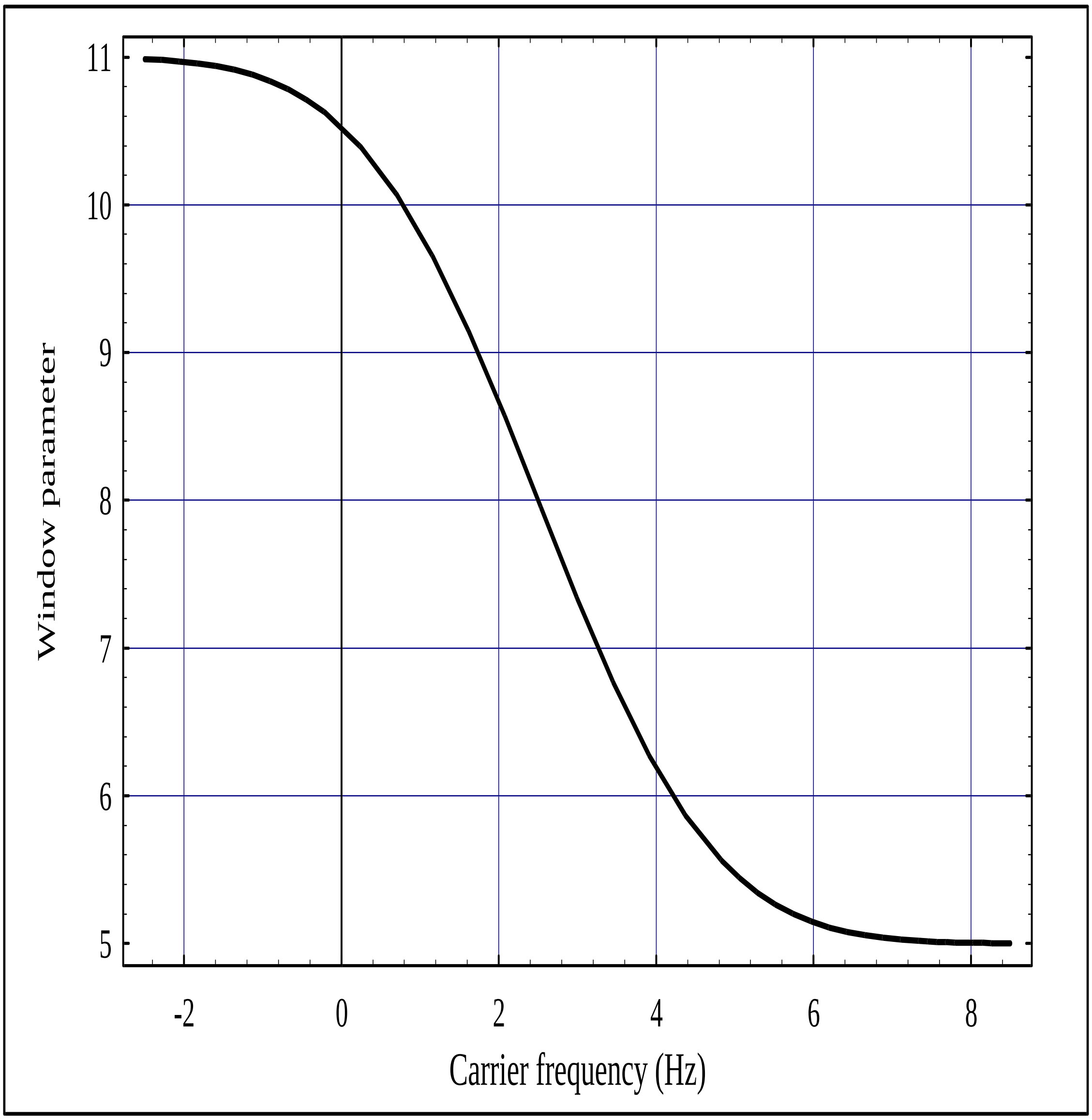


Figure 4.1: The assumed wave decay parameter ω_0 as a function of carrier frequency. The curve can be adjusted according to several parameters: approximate peak frequency, significant range of frequency, range of decay parameter, as well as a shift adjustment parameter; as are indicated in the attached program piece.

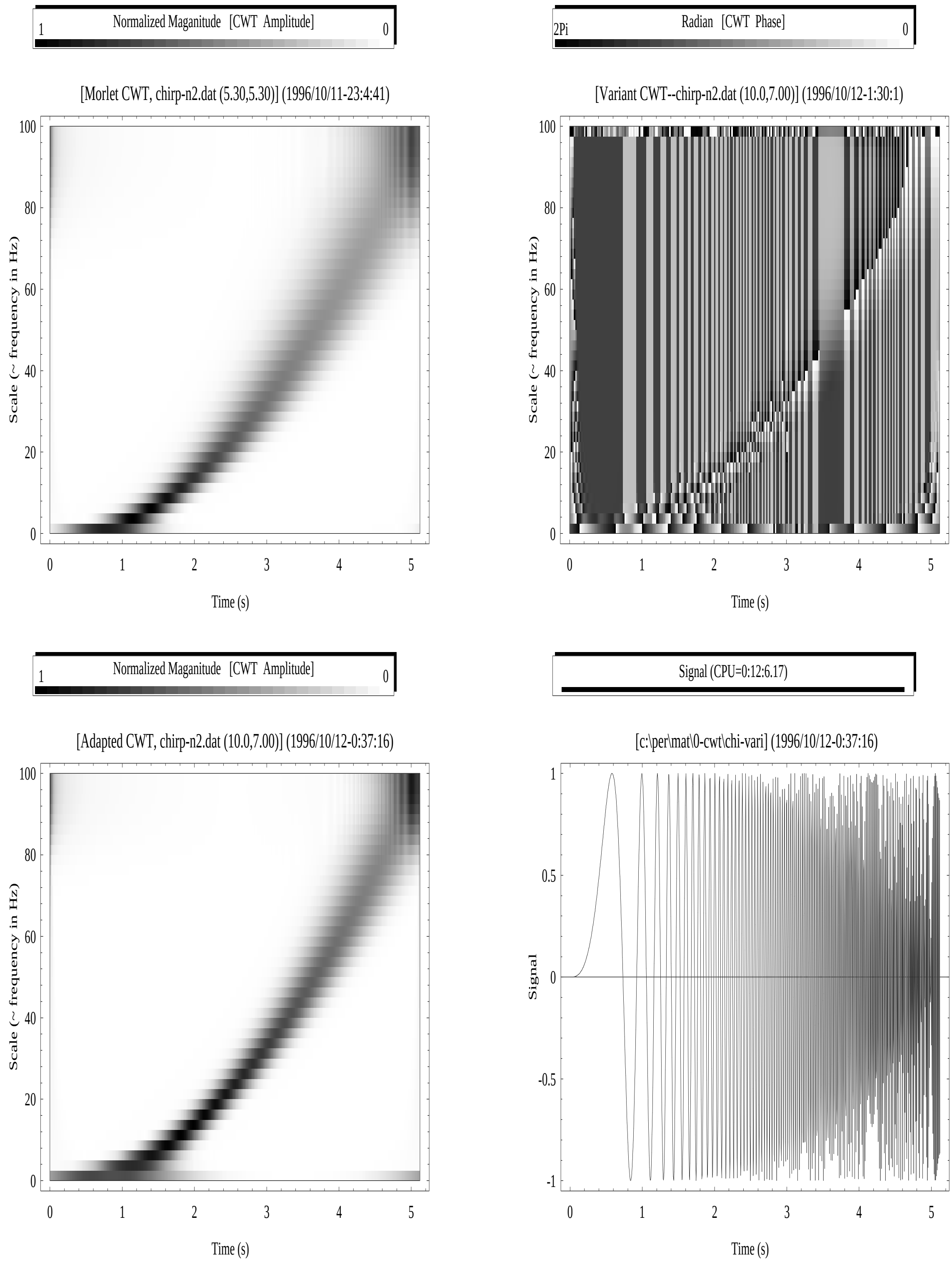


Figure 4.2: Phase plane characters for a parabolic chirp `(bottom right)` with `(top left)` and without `(bottom left)` adapting time-frequency windows. Top right shows a map of the phase that is obtained from using a newly devised wavelet variant by Lee and Wu [10]. The wavelet variant has properties quite in contrast to those of Morlet wavelet and has refined ridge extraction capability.

Chapter 5

Wavelet Coherences and Spectral Coherences

5.1 Introduction

Coherences among different target quantities manifest the degree of mutual interactions in a multi-scale coupling system. By studying their variations for different experimental setups or for various parameters it is possible to identify the evolutions of different scales and to isolate key influencing factors and the associated effects.

Programs for both spectral and wavelet coherences were developed and used to study the scale features in wind, wave, and rain coupling system. A few energy phenomena and related cascade mechanism had also been provided in previous studies.

It was beyond expectations that our analyses had been able to provide clear and definite proofs (as well as explanation for causes) of wavelet coherences' superiority over the Fourier coherences. On the one hand the wavelet tool for coherences yield much better information that is both lean and unambiguous; on the other hand the tool is much economical in the amount of data needed for analysis.

5.2 Wavelet coherence and spectral coherence

The wavelet coherence and the spectral coherence as formulated in the program are given below.

The cross correlation function of two functions $g(t)$ and $h(t)$ is the following inner product $c(t)$

$$c(t) = \langle g(t+\tau), h(\tau) \rangle, \tag{5.1}$$

where τ is a dummy variable. The correlation coefficient function of $c(t)$ is $r_s(t)$,

$$r_s(t) = \frac{c(t)}{\|g(t)\| \|h(t)\|}. \tag{5.2}$$

For real $g(t)$ and $h(t)$, its Fourier transform is

$$\frac{\widehat{c(t)}}{\|g(t)\| \|h(t)\|} = \frac{G(\omega)\overline{H(\omega)}}{\|G(\omega)\| \|\overline{H(\omega)}\|}. \tag{5.3}$$

The artifacts to be introduced in the spectral coherence are associated with the form of expected values as well as the introduction of a normalization as given by

$$R_s^2(\omega) = \frac{|\mathbf{E}[G(\omega)\overline{H(\omega)}]|^2}{\left(\mathbf{E}[|G(\omega)|^2]\mathbf{E}[|H(\omega)|^2]\right)^{1/2}}, \tag{5.4}$$

where the symbol $\mathbf{E}$ stands for taking expected value. Since expected values take no action without introducing one more dimension, this equation is identically unity for all frequencies if each data sequence is not segmented and arranged in an array with one additional dimension. The process of this segmentation is just like that commonly implemented in calculating the power spectrum, and its purpose is to reduce the uncertainty or standard deviation of the spectrum. There is no doubt that inherent properties of the discrete Fourier analysis impose similar limitations to the conclusiveness of spectral coherences.

As to the wavelet coherence, the derivation is even simpler along with fewer artifacts. The wavelet resolution of identity of two functions is

$$\langle g, h\rangle = \frac{1}{c_\psi}\int_0^\infty \frac{1}{a^2}\int_{-\infty}^\infty \langle g, \psi_{a,b}\rangle\overline{\langle h, \psi_{a,b}\rangle}dbda. \tag{5.5}$$

For a fixed scale a

$$\langle g_a, h_a\rangle = \frac{1}{c_\psi}\frac{1}{a^2}\int_{-\infty}^\infty \langle g, \psi_{a,b}\rangle\overline{\langle h, \psi_{a,b}\rangle}db. \tag{5.6}$$

Here the integration with respect to the translation parameter b is physically, as well as intuitively, similar to the operation of taking an expected value by summing up the elements in the population space. It is therefore quite straightforward to define the wavelet coherence as the natural extension of the normalized equation of resolution of identity:

$$R_w^2(a) = \frac{|\mathbf{E}_b[\langle g, \psi_{a,b}\rangle\overline{\langle h, \psi_{a,b}\rangle}]|^2}{\left(\mathbf{E}_b[|\langle g, \psi_{a,b}\rangle|^2]\mathbf{E}_b[|\langle h, \psi_{a,b}\rangle|^2]\right)^{1/2}}, \tag{5.7}$$

where the subscript b in $\mathbf{E}_b$ stands for taking average with respect to the translation parameter.

Based upon the above formulations the Asyst shell programs for wavelet coherence and spectral coherence are "`S_Coh_WC.Shl`" and "`S_Coh_SC.Shl`", respectively.

Relevant files for figure manipulations in a batch process are "`P-WCoh-A.shl`", "`P-SCoh-A.shl`", "`FigA-Ini.asy`", "`FigAPlot.shl`", "`FigAPlot.asy`", "`FigABP.bpf`", and "`C-CC-A.BPF`", etc.

Figures 5.1 and 5.2 show some of the comparisons between wavelet coherence and spectral coherence. ❖

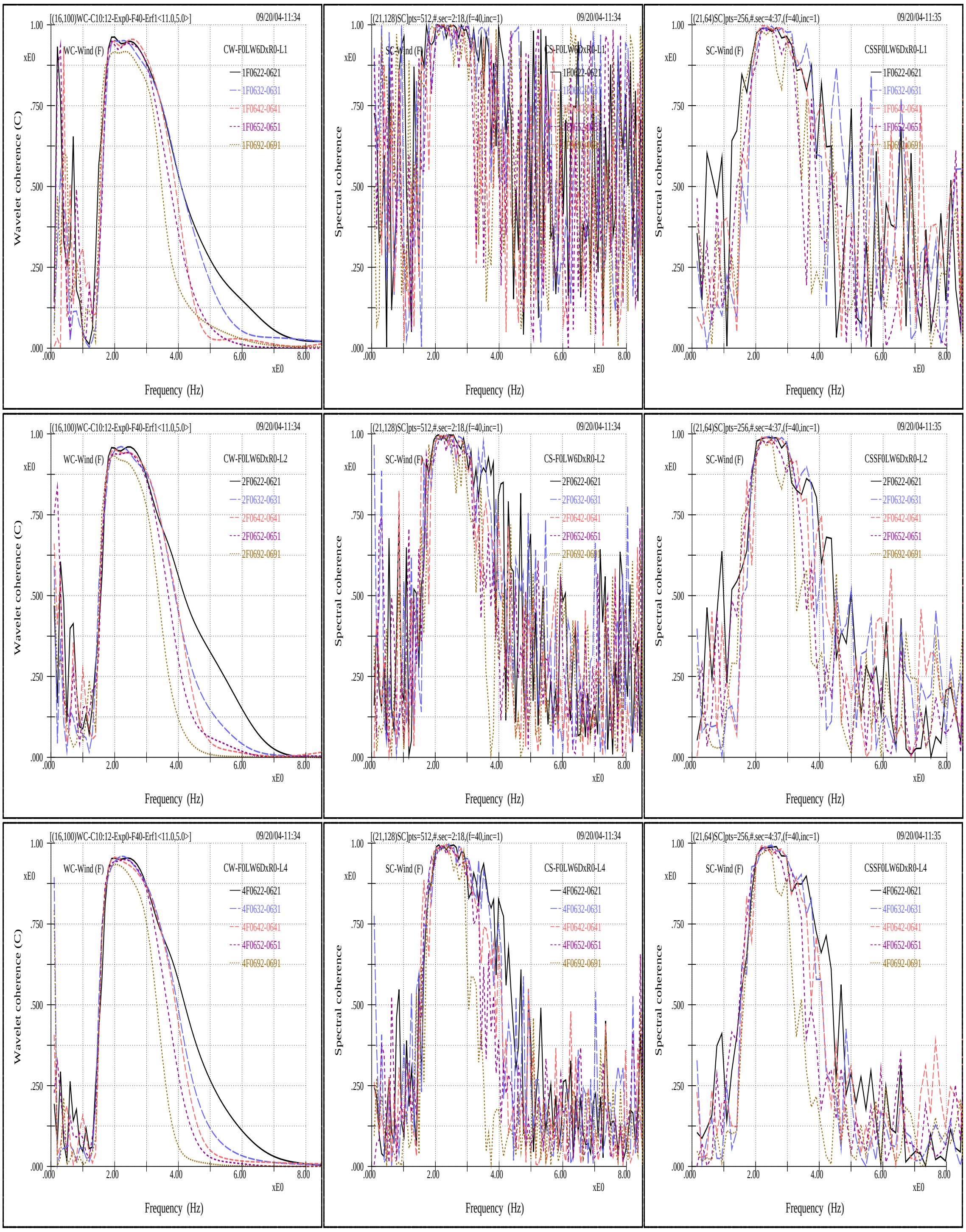


Figure 5.1: Wavelet and spectral coherences between surface wave and aqueous flow associated with three different data lengths. The data lengths are 1024、2048、4096 points, respectively form top to bottom. Depths of flow measurement are 2、3、4、 5、9 cm, respectively, below the still water surface. The left sub-figures are for the wavelet; the right for the spectra associated with two sets of FFT parameters.

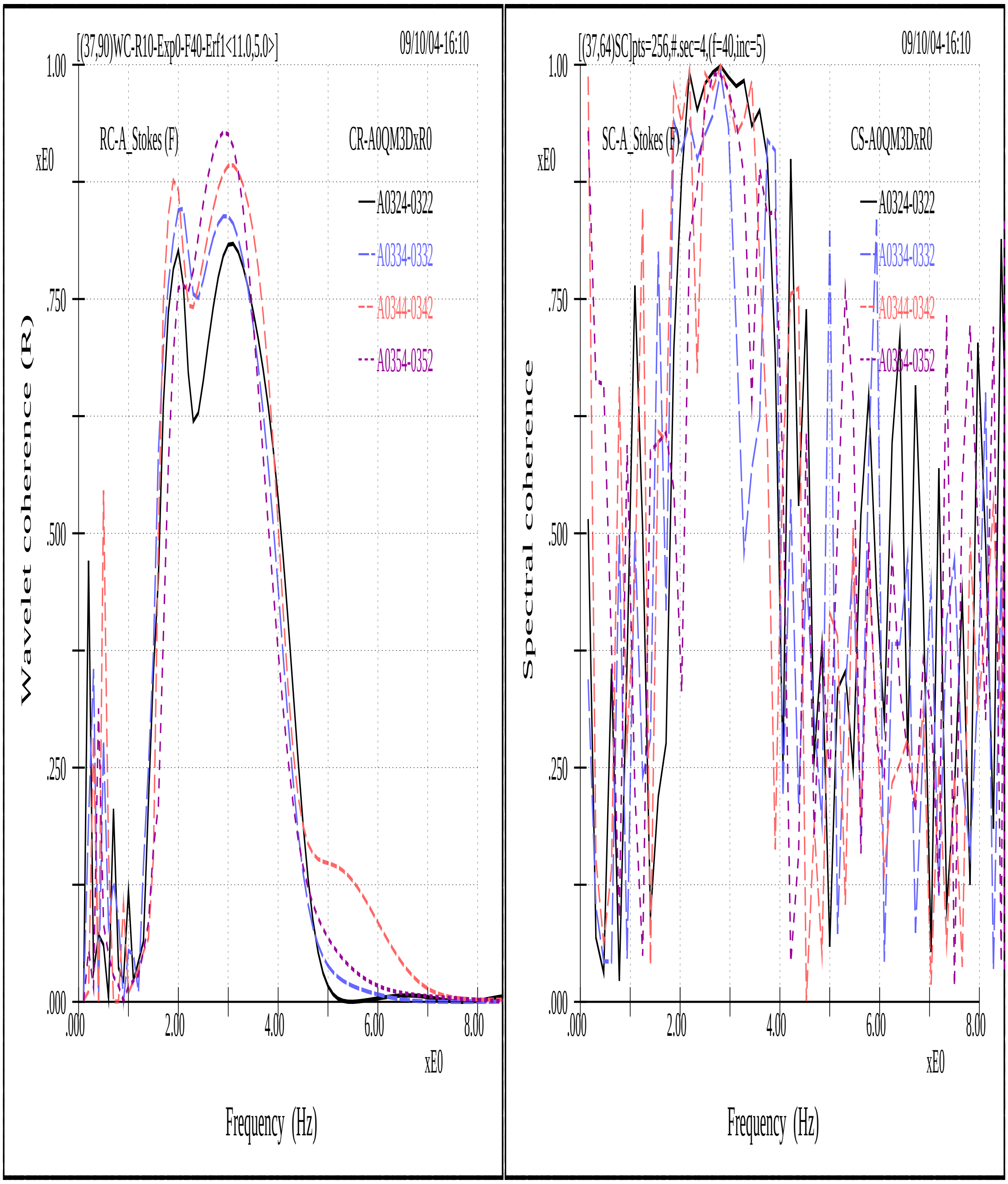


Figure 5.2: The comparison between the wavelet real part coherence (Left) and the spectral coherence (right). Here a Stokes wave with high wave steepness is used. It can be seen from the wavelet results that there is a highest coherent level at the deepest flow measurement point.

Chapter 6

Wavelet Variant

6.1 Introduction

For the studies of water wave related signals using wavelet approach, our results had shown that a very important factor that contributes to the usefulness of a wavelet function basis in revealing the most intimate and intricate physical aspects of water wave related signals is that the function basis should possess the following property: the associated mother wavelet should have "complete oscillation" and the associated scaling function should have "total positivity". Physically speaking, this property means that the basis functions, in comparison with other basis functions, are relatively quite regular. And practically, this means that a transform associated with "complete oscillation" and "total positivity" provides information that is far more tangible than otherwise provided. This further implies, in ever plainer language, that if there is a slight change in signal content than a basis with such a property will yield transform coefficients that are more or less "reasonably expected" or "mildly altered"; otherwise, the variation of transform coefficients arising from such a slight change may be completely ad hoc [7, 9, 11].

Based on the cognition above, as well as from our previous studies on the entropy performances for the comprehensive wavelet basis categories, the cardinal spline wavelet had been shown to be exactly the optimum basis for modeling water wave related signals from the point of view of discrete transforms (including the discrete Fourier transform)

[8, 11, 9]. And a natural extension of such an optimum discrete basis to the continuous transform thus implies the Morlet wavelet to be a very appropriate candidate.

With these understandings, we had carried out various comparisons on the performances of our wavelet variant and those of the Morlet wavelet. In brief, our programs had delivered the conclusions that the present methodology not only possesses the same easiness in numerical implementation but also holds a few advantages, such as, an improved capability in extracting constituent power ridges of signals from both the modulus and phase plane renditions, and more informative and intricate characterizations.

6.2 The devise of the wavelet variant

The wavelet variant's "basis function" $\psi(t)$ is defined as:

$$\psi(t) = \frac{1}{\pi^{\frac{1}{4}}} \left[\operatorname{sgn}(t) \sin \omega_0 t - i \cos \omega_0 t \right] e^{\frac{-t^2}{2}}. \tag{6.1}$$

The quote implies that the function serves as the seed of a function basis in a way similar to what a mother wavelet does. In the equation, ω_0 is relevant to the modulation frequency of the counterpart Gabor transform (or windowed Fourier transform); $\operatorname{sgn}(t)$ is the sign function; the exponential stands for a Gaussian envelope; and, the constant is somewhat related to a unit norm and serves for matching the counterpart constant of the Morlet wavelet. Here the basis function is entitled with a "quasi" term due to its lack of wavelet admissability. But, judging the nature of the present methodology, one may well regard the basis function to be a "wavelet variant". In fact, compared with the simplified form of the Morlet wavelet, the major difference is the presence of the sign function.

The scaled and translated versions of the wavelet variant is :

$$\psi_{a,b}(t) = \frac{1}{\sqrt{a}\pi^{\frac{1}{4}}} \left[\operatorname{sgn}(t) \sin \omega_0 \left(\frac{t-b}{a} \right) - i \cos \omega_0 \left(\frac{t-b}{a} \right) \right] e^{\frac{-\frac{t-b}{a}^2}{2}}, \tag{6.2}$$

where a is the scale parameter and b is the translation parameter. The $\frac{\omega_0}{a}$ physically means a carrier frequency and is the core target of the transform information. Note that the "scale" or "frequency" ordinate shown in all modulus and phase renditions to be presented in later chapters represents exactly the values of this variable. And this provides easily perceivable physics, as opposite to many studies that adopted the imperceivable scalar "a".

For our methodology we define the modulus and phase in the following ways.

Let $f(t)$ be a signal function, the modulus of the transform coefficient is defined either as

$$|\langle f(t), \mathbf{I_m}\psi(t)\rangle + i\mathcal{H}\left[\langle f(t), \mathbf{I_m}\psi(t)\rangle\right]|, \tag{6.3}$$

or

$$|\langle f(t), \mathbf{R_e}\psi(t)\rangle + i\mathcal{H}\left[\langle f(t), \mathbf{R_e}\psi(t)\rangle\right]|, \tag{6.4}$$

where $\mathbf{R_e}$ and $\mathbf{I_m}$ represent real and imaginary part, respectively, $\langle , \rangle$ means the inner product, and $\mathcal{H}$ stands for the Hilbert transform. Note that the implementation of transform in either definition is based on only real or imaginary part alone. In this sense, the modulus may lack the mathematical formalism of a "basis", but here we first point out that the first definition gives basically the same result as Morlet wavelet's, while the second definition yields information that is especially useful in easy extraction of signal power ridges and that is also superior to what provided by the Morlet wavelet.

As to the phase it is defined as

$$\tan^{-1}\frac{\mathbf{R_e}\langle f(t),\, \psi(t)\rangle}{\mathbf{I_m}\langle f(t),\, \psi(t)\rangle} + \left(\frac{\pi}{2}\ \text{ or }\ 0\right), \tag{6.5}$$

or

$$\tan^{-1}\frac{\mathbf{I_e}\langle f(t),\, \psi(t)\rangle}{\mathbf{R_m}\langle f(t),\, \psi(t)\rangle} + \left(\frac{\pi}{2}\ \text{ or }\ 0\right). \tag{6.6}$$

Basically the added constant reflects a phase rotation, and they can be used to switch the pattern of significant time-frequency features or to show easy visualizations in accord with either the power ridges of component signals or the time-frequency spreads of constituent components of a signal (or the spreads of basis functions).

Here the Asyst and the Mathematica programs used are the same as those of the adaptive continuous wavelet transform, i.e., "`S_WVT.shl`", "`TFR-WV_Shell.nb`" and "`TFR-WV_Stand-alone.nb`". With a specified parameter the programs automatically select the right parts of codes.

In addition, the Mathematica program "`Wavelet-Variant-Characterizations.nb`" provides feature characterizations and comparisons between the wavelet variant and the Morlet wavelet.

A few of the outcomes are shown in figures 6.1, 6.2, 6.3, and 6.4, and their explanations are provided in the figures. ❖

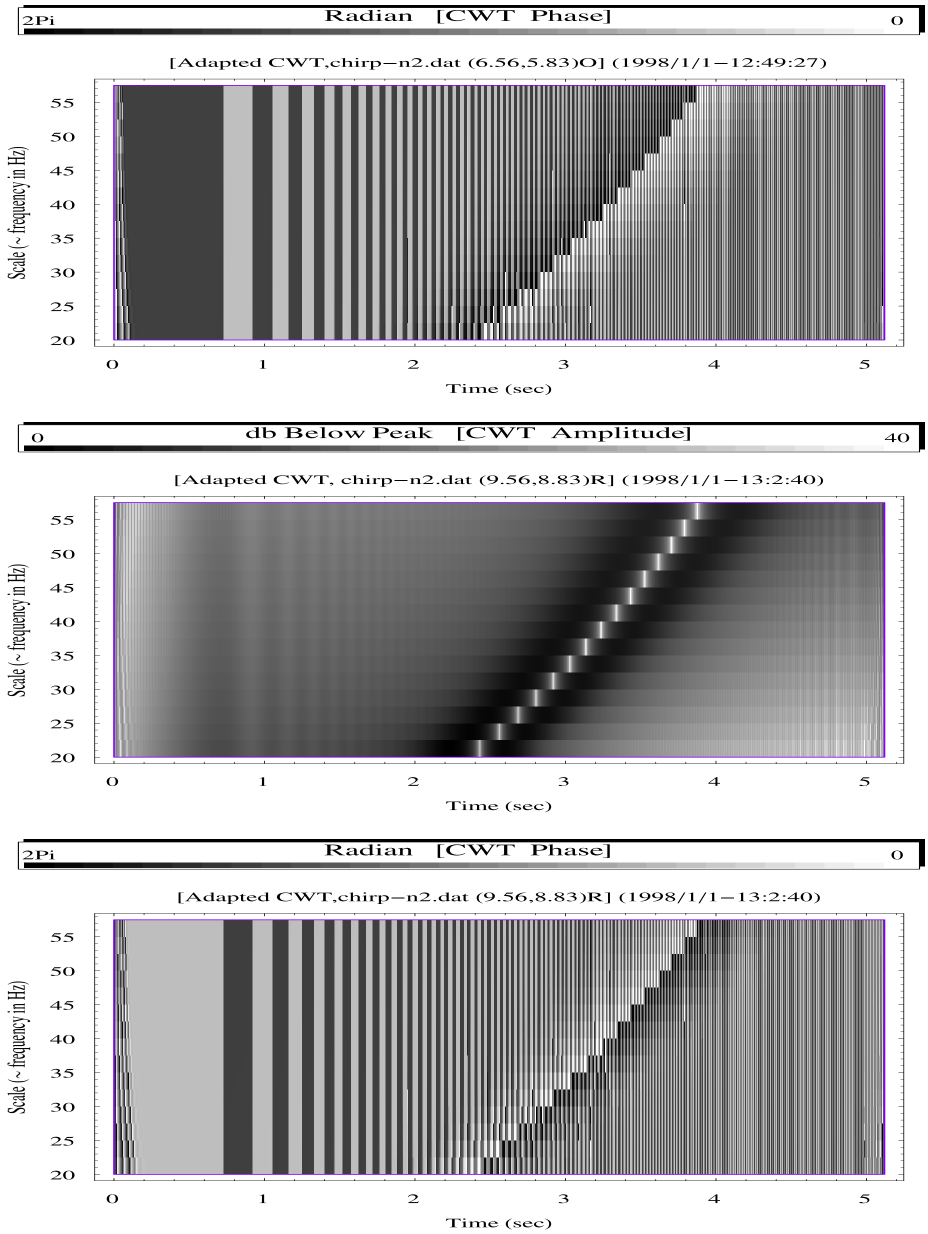


Figure 6.1: The modulus (mid sub-figure) and phase (top and bottom sub-figures) planes for the same time-Frequency zoom-in section using the proposed quasi wavelet basis function, but here the sub-figures are associated with different numerical resolutions in scale and also with different adaptations in each scale's time-frequency windows (i.e., different ranges of the parameter ω_0). Even though the scale resolution here is coarser when compared to that of the previous figure, both the modulus and phase planes still show clear features of the instantaneous frequency. Moreover, the phase plane renditions provides yet the same, as well as very sharp, interfacial features at all the time translation steps, even for those intermediate time translation points that do not locate at the scale resolution points.

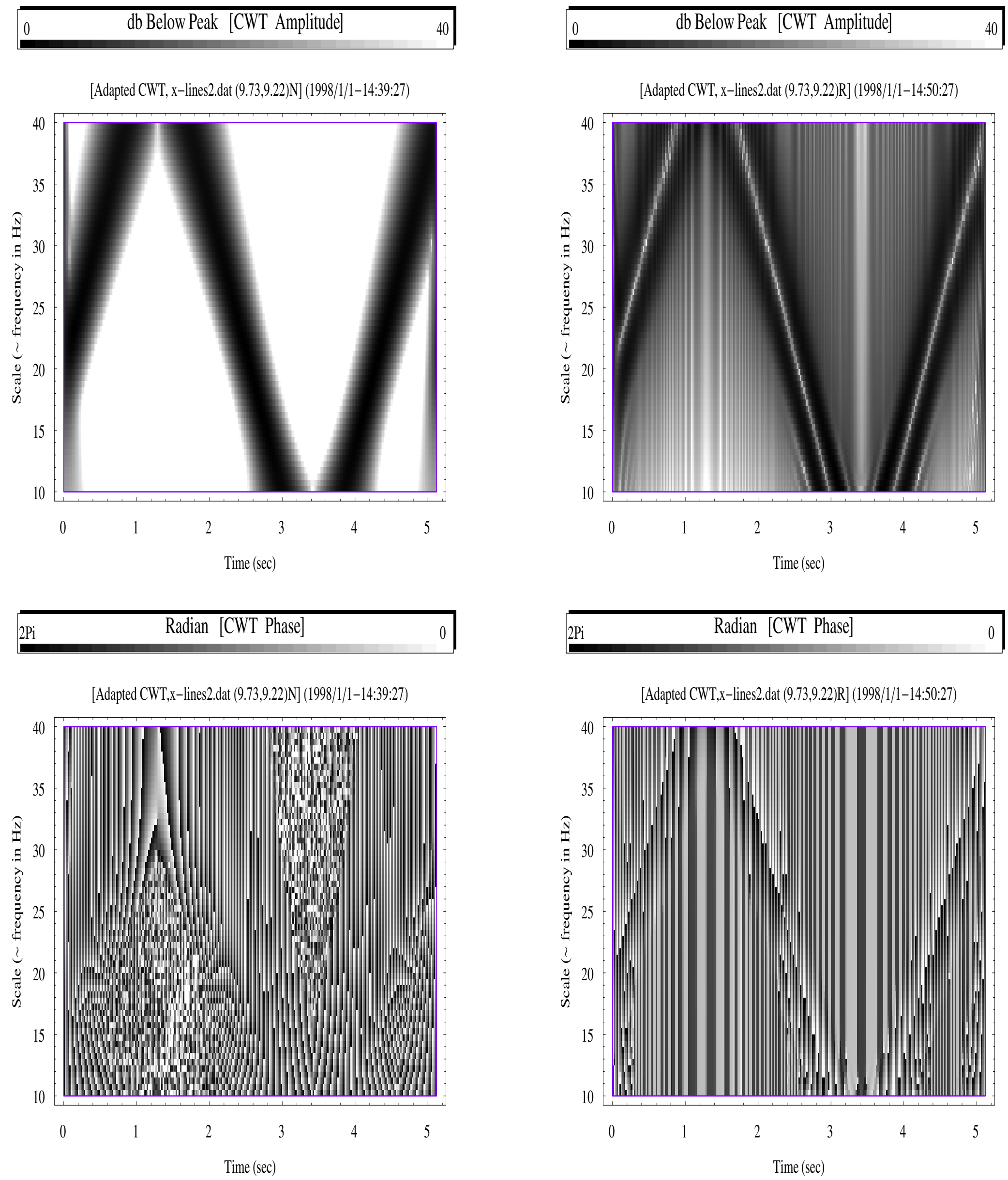


Figure 6.2: This figure shows a zoom-in section of an X-signal composed of two crossing linear chirps with equal power contents. The left sub-figures are for the simplified Morlet wavelet, and the right sub-figures are for the quasi wavelet basis function. They manifest the same depictions as given by the parabolic chirp.

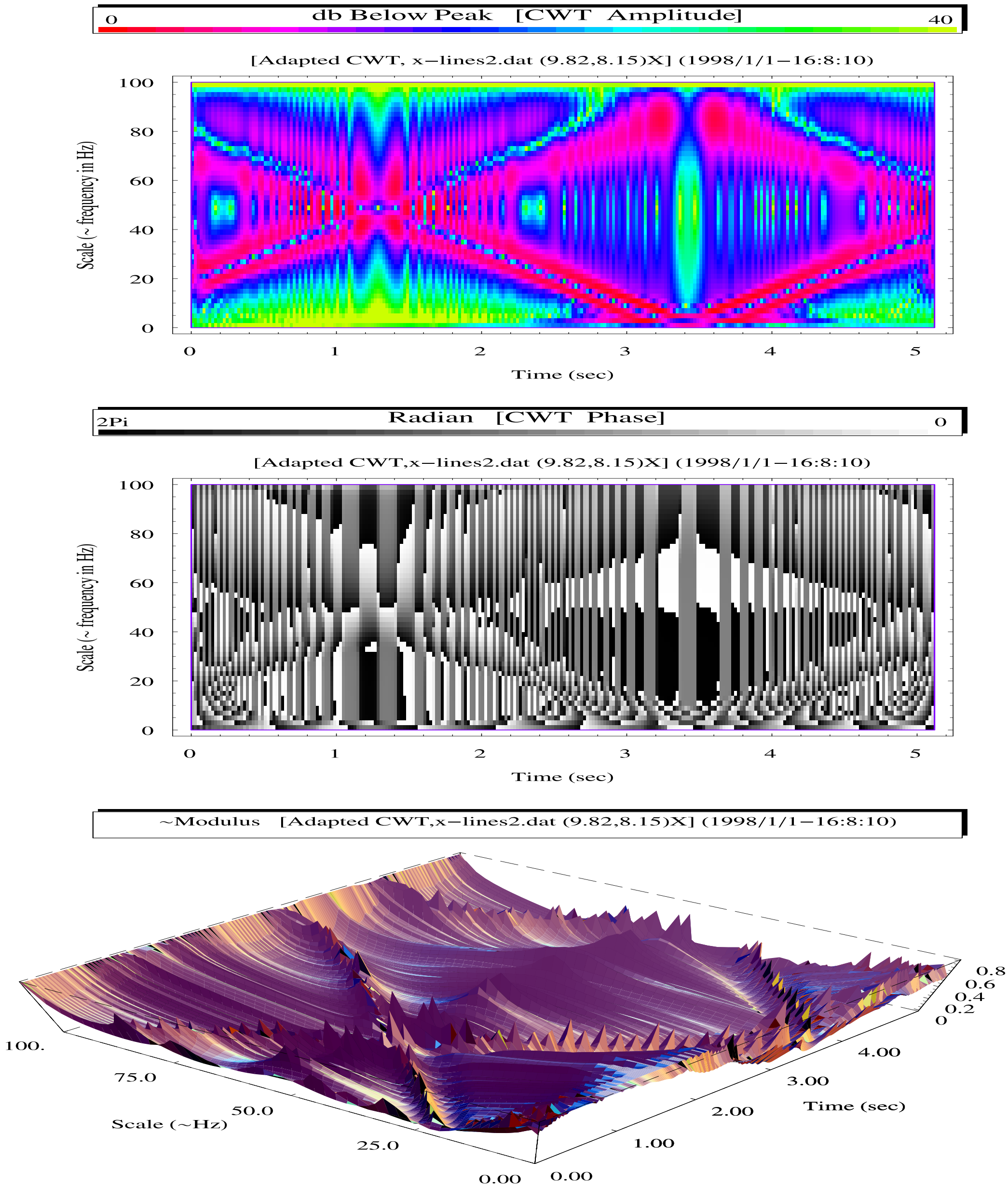


Figure 6.3: This figure illustrates the effects of phase rotation on the same X-signal based on the quasi wavelet basis function. Again, various interfacial lines in the mid sub-figure serve as indicators of the extend of frequency leakage and phase noise. It is noted that the top sub-figure is in color but may be printed in black and white.

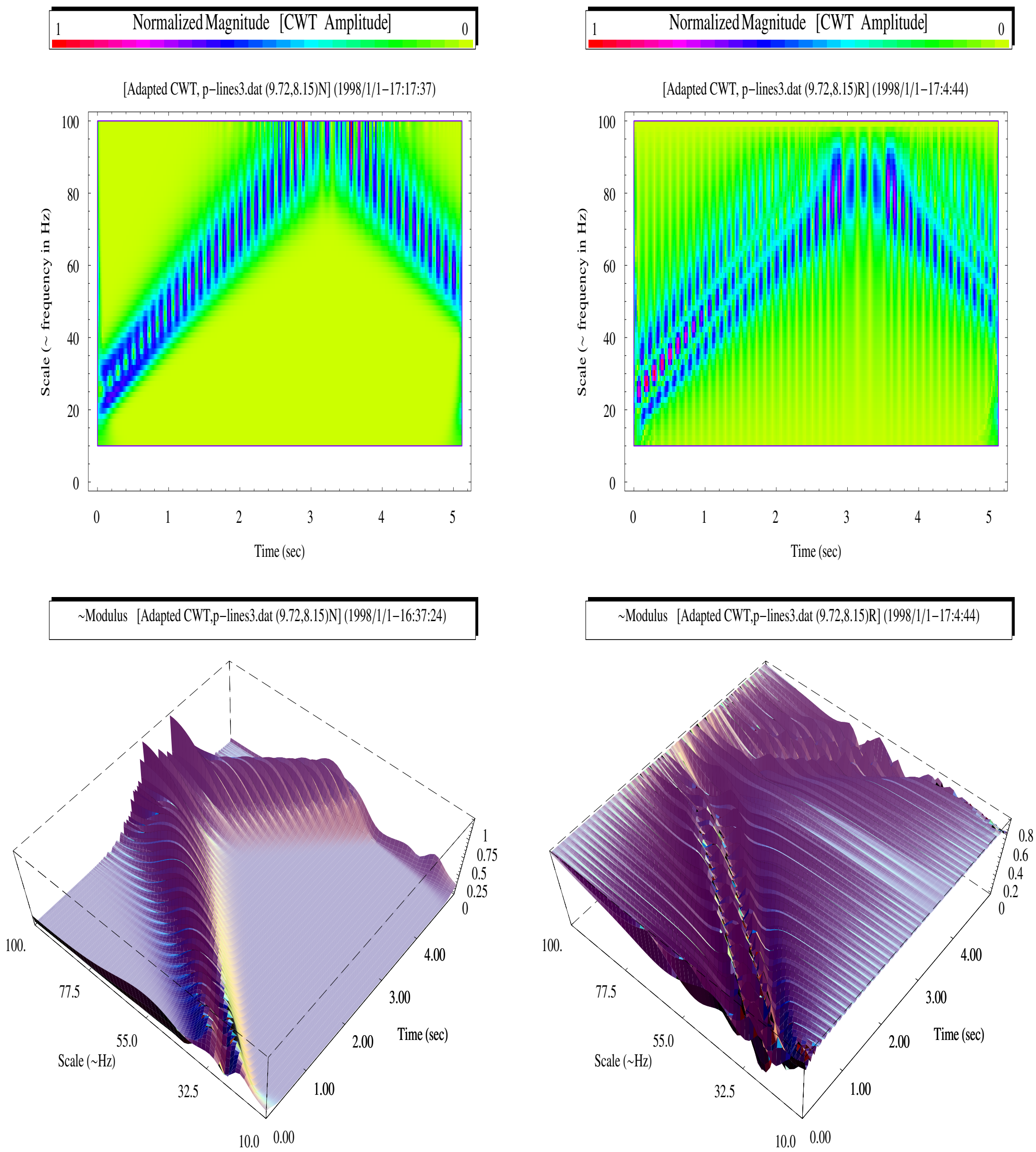


Figure 6.4: This figure shows the ridge extraction of a signal composed of a pair of parallel chirps that are with equal power contents. Here the frequency separation between the two chirps is one tenth Nyquist rate. The left sub-figures are for the simplified Morlet wavelet, and the right sub-figures are for the quasi wavelet basis function. It is clear that he power ridge given by the simplified Morlet wavelet is entirely misleading. But, for the quasi wavelet basis function, the two interfacial lines are clearly identifiable except at region near the Nyquist frequency. In the bottom right sub-figure, we have intentionally inverted the rendering, i.e., a trough in the 2-D plane (top right sub-figure) turns to a peak in the 3-D figure; and what are seen are the two clear spike lines.

Chapter 7

Program-related Issues

7.1 Introduction

Here we cover a few program-related issues concerning the completeness, logicalness, and integrity of the numerical processing system and the adjunct elements. These include: the memory management and optimization for the Asyst codes, auxiliary programs, add-ins, the design of working environment (or workbench development), and the management and publishing system for figures, programs, and documents.

7.2 Memory management and overlay requirement

The Asyst is a DOS-based programming language. Memory management is a crucial part of the working of all the programs. Each type of variable and data (such as, Symbol, String, Array, Unnamed Array, Token, Expended Memory Token, System Buffer, etc.) needs its special memory allocation, and any type of run out of memory space will cause the program to cease. Back then the Asyst version 3.0 it needed special attentions for the two DOS boot-up files for batch and system configuration, i.e., "`autoexec.bat`" and "`config.sys`". And it sometimes took many weeks to figure out proper parameters and assignments to be put into these files for the functioning of the programs.

Figure 7.1 is a screen snap of the settings of the Memory management and overlay requirement for the working of all the present programs.

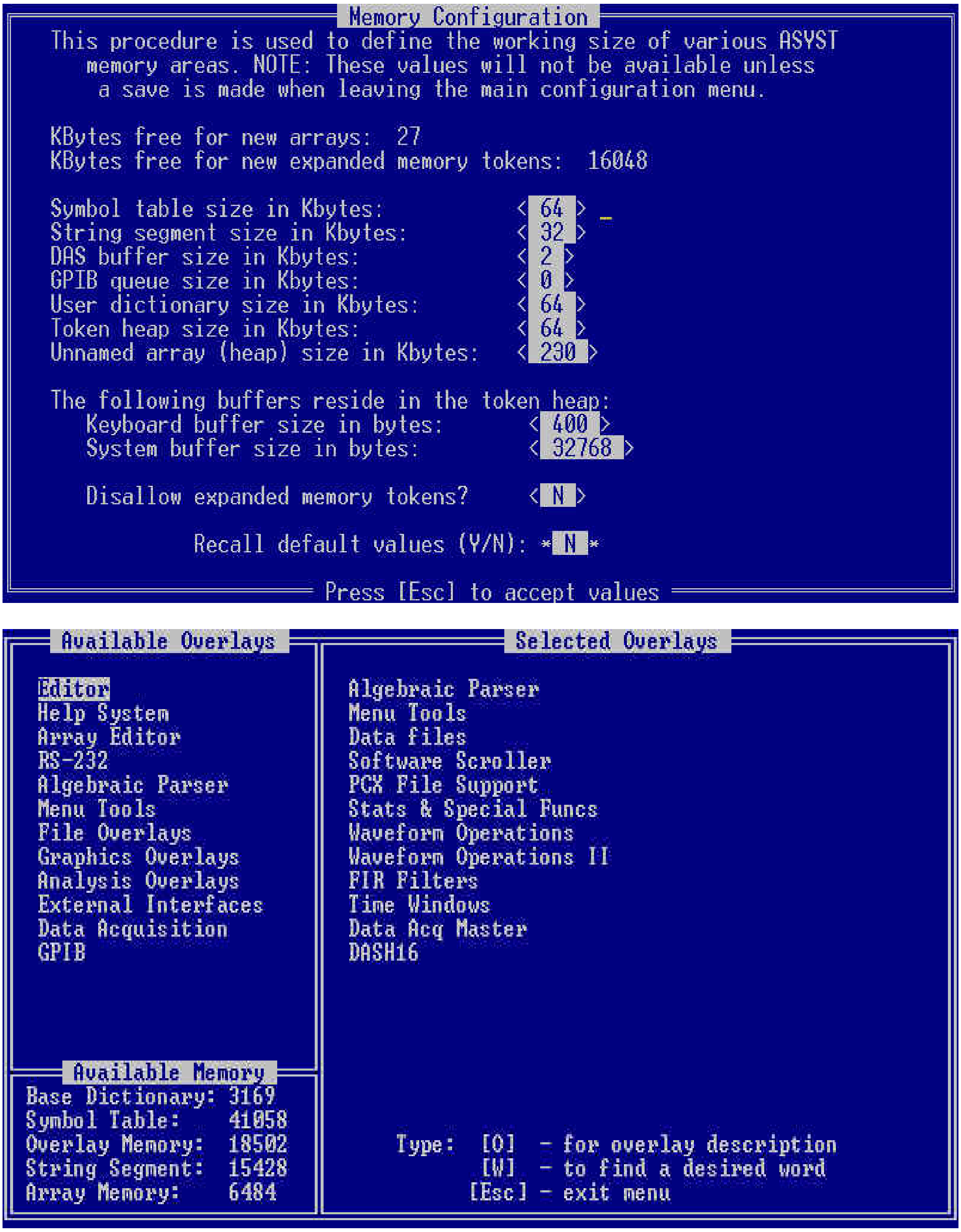


Figure 7.1: The Memory management and overlay requirements for the present programs.

7.3 Auxiliary and add-in programs

The Asyst language operates in stack principal; different types of variable and data occupy their individual memory spaces and may put their contents in their individual stacks. An Asyst "word" is equivalent to a Fortran "subroutine". In that sense, to understand the primary programs one needs to know, in addition to the language that Asyst specifies, all the "words" the author speaks. So, there are a vast amount of words (subroutines) that are not specified in the core programs, and these words, as well as a few add-in programs, are given as auxiliary and add-in programs. It is the author's wish that no critical part is missing in our list of programs. Generally speaking, these programs or words can be categorized in accordance with their purpose to serve as a run-time executable or as a general utility.

Words in a run-time executable always reside in memory ever since the launch of the Asyst executable. These words generally concern hardware specific issues, real-time screen display, line and color schemes, string and number manipulations, file and array commuting, words for interactive purpose, etc.

General utility programs are for generating spreadsheet (Lotus 123) files and encapsulated Postscript (EPS) figures; they need to be loaded when needed.

Representative files for these are "`Lee-Word.asy`", "`123-A2W.asy`", "`FigA-Ini.asy`", "`FigAPlot.shl`", "`FigAPlot.asy`".

A few Mathematica add-in programs serve as postprocessing of Asyst outcomes and are mainly used to render 2-D and 3-D graphics. They are "`TFR-WP_Shell.nb`" and "`TFR-WP_Stand-alone.nb`", "`TFR-WV_Shell.nb`", and "`TFR-WV_Stand-alone.nb`". Note again that the shell programs are mainly for the purpose of easy navigation within program, convenient parameter setup, reminder of proper sequence of execution, paper-wasting consciousness, and, quite importantly, the safeguard against unintentional keyboard touches.

7.4 The workbench

The WinEdt program and its macro programming language are use to devise a workbench for the Asyst code writing. There is too much detail to be given here for its setting. Only a few snapshots are shown in Figures 7.2 and 7.3.

7.5 System for documentations

The workbench is for the pleasant work through computer screen; working on paper need a different approach. And it is vitally important to have a management and publishing system for figures, programs, and documents. For example, during the numerical experiments or case studies with numerous setups and parameters, there can be seemly countless figures to be not only generated (this part is done by our auxiliary programming) but also read and compared; and when debugging the twined (and also intertwined) program(s), the limited scope of screen display quite sometimes insufficient and inconvenient for the eyes and the mind. At these times one can really appreciate the existence of a systematic and automated documentation system. For this purpose quite a few codes were also written. They mainly utilize the WinEDT system and its macro language, and the LaTeX codes together with many freeware packages. In fact all the Asyst programs listed here are generated in a single batch with just a few mouse clicks alone — no typing, no sorting, and no intervention, but a nice tea break. ❖

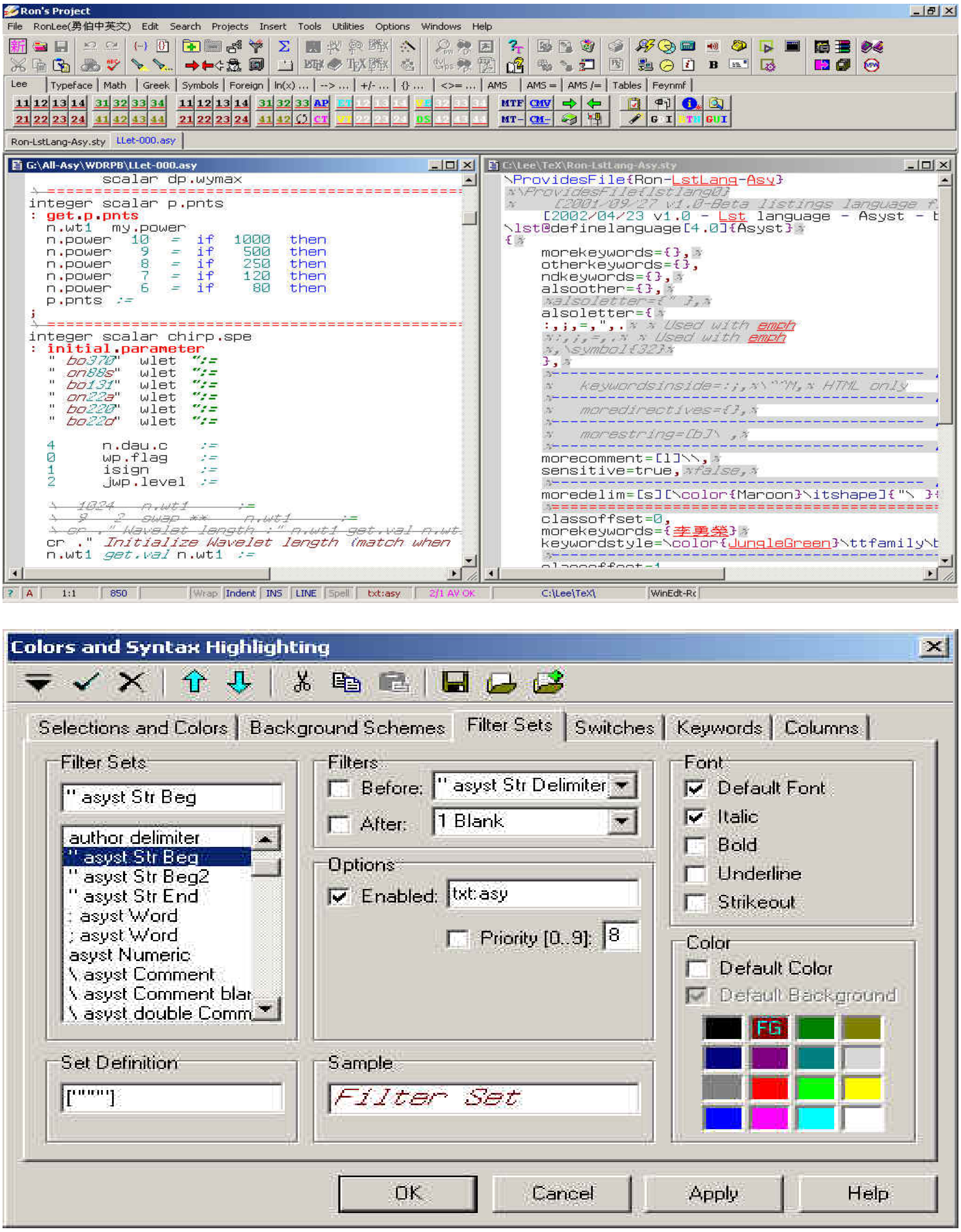


Figure 7.2: Some aspects of the workbench for the Asyst programs.

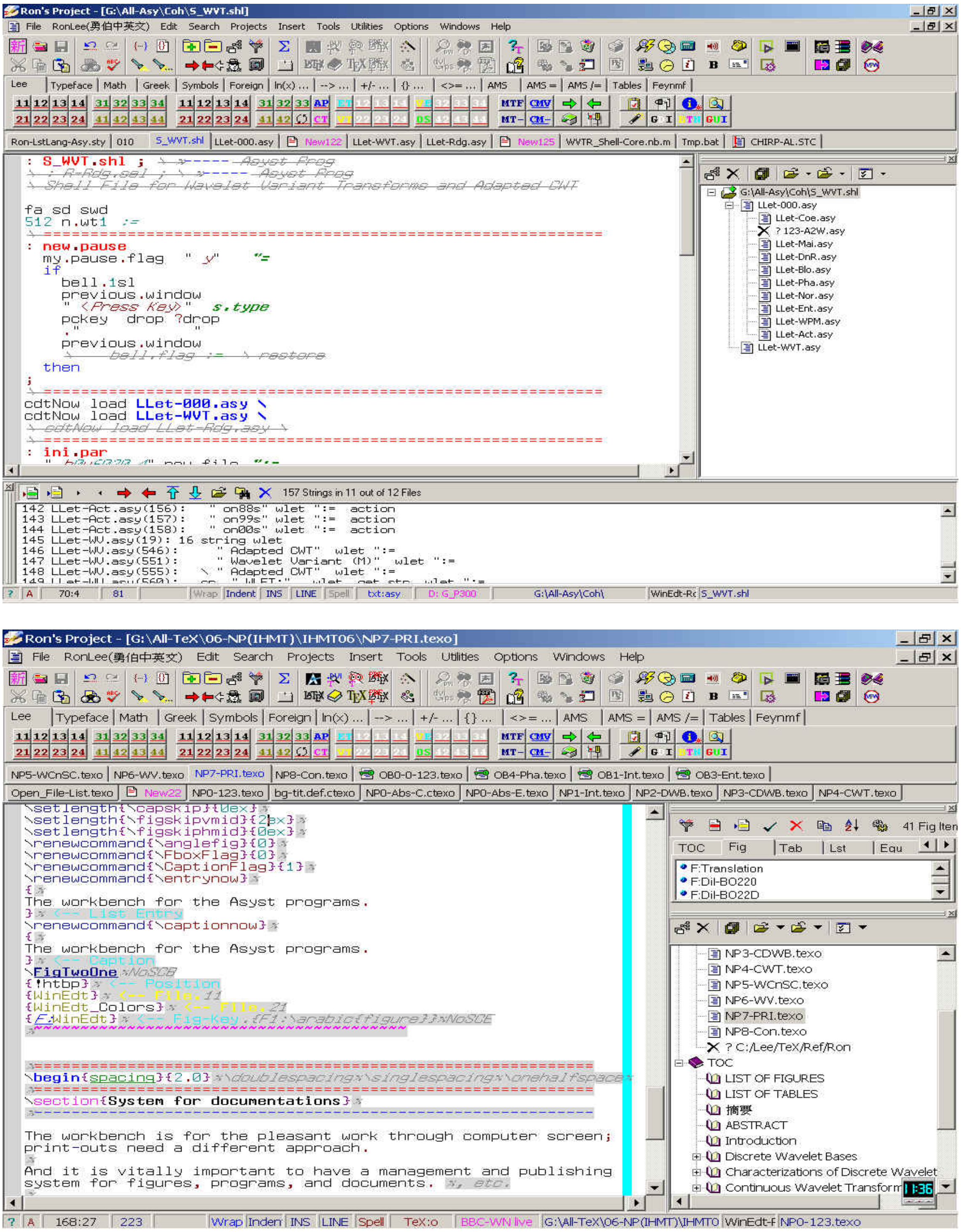


Figure 7.3: The workbench for the document, file, and figure manipulations.

Chapter 8

Conclusions

Using the Asyst programming language, a comprehensive set of wavelet program was developed from the ground up specifically for the analyses of water wave related signals. Various codes for proficient data processing and rendering were also written and integrated in the main programs. Numerous programming add-ins and application auxiliaries were also incorporated, notably, the Postfix language for auto generation of encapsulated Postscript figures, spreadsheet interface for data outputs, the Mathematica language for graphic representations of various 2D and 3D plots, as well as the WinEdt macro language, LaTeX codes and packages, etc. for the development of the workbench and the figure and document management system. The wavelet scope of the programs includes:

- both the discrete and the continuous wavelet transforms,
- forward and inverse discrete transforms,
- wavelet and wavelet packet blowups,
- norm distributions, best basis and best level identifications,
- entropy statistics and the identification of the optimum basis for water wave related signals,

- phase distribution of the wavelet characteristic function and its implication,
- wavelet and spectral coherences and their comparisons,
- continuous wavelet transform associated with a scheme using adapted time-frequency windows,
- a newly devised wavelet variant and its characterizations, as well as its applications.

Finally, hope it warrant to say a few more.

Life is limited, but code never ends. When reviewing a program the author is quite often trapped in an overwhelming feeling of shortage of memory and wreathing of striving. Setting out to write codes using the specific approach of wavelet, the author surely had to confess that he had certain purposes in mind and expectations in heart. Nevertheless, an expectation is simply a wish amid uncertainty and anxiety and can never be deemed a promise. But it is to the author's relief that the programs did work and in a few respects even leaded to some totally unexpected. One last point to note is that these programs are surely far from elegant, not even tidy at all, but the touch sensation is strong for the programmer. ❖

APPENDIX — Asyst Programs

R_DnR.shl

勇伯程式庫 A01492-20070125-A043 ❖ January 25, 2007 ❖ [Page count: 5]

```
: R_DnR.shl ; \ Wavelet Decomposition and Reconstruction \ -*- txt:asy -*-

fa sd \ swd
1024 n.wt1 :=
cdtNow load LLet-000.asy \

\ ================================================================
\ ----------------------------------------------------------------
: ini.par.chirp.al
  " Chirp-al.dat" now.file ":=
  512  n.wt1    :=
  200  s.freq.keep :=          \ Sampling freq -- original rate
  1    i.inc    :=             \ array index Inc for sub-sampling
;
\ ----------------------------------------------------------------
: ini.par.f
  " f0w6020.2" now.file ":=
  1024 n.wt1    :=
  40   s.freq.keep :=          \ Sampling freq -- original rate
  1    i.inc    :=             \ array index Inc for sub-sampling
;
: ini.par.b
  " b0w6020.4" now.file ":=
  1024 n.wt1    :=
  200  s.freq.keep :=          \ Sampling freq -- original rate
  4    i.inc    :=             \ array index Inc for sub-sampling
\   0    abs.flag   :=           \ absolute value norm
\   0    filter.flag :=          \ percentage of norm energy kept
\   0.975 norm.keep.p.best :=    \ for initial filtering
\   6.0  p.b.e  :=               \ Pass band end (Hz)
\   6.0  t.w    :=               \ Transition width (Hz)
\   0    best.flag   :=          \ unclear, possibly not necessary (WPM)
\   100  #.div  :=               \ foe entropy calculation
;
\ ================================================================
-1   isign := \ ." <- or -> :"
" D" act.flag.str ":= \ 3    act.flag :=
ini.par.b
\ ini.par.chirp.al
\ ================================================================
set.act
set.let
set.123
\ for P N E -- set.let and set.123 are Moved into word within set.act
lee
\ ----------------------------------------------------------------
\ \ Keep all default setings for all the above inputs
\ ================================================================
\ \ ---------------------------------------- ■ \ \ For Blowup
\ \ \ act.all
\ \
\ \ \ act.all.1 \ LMS
```

```
\ \ \ act.all.2 \ ONA
\ \ \ act.all.3 \ ons
\ \ \ act.all.4 \ onc
\ act.all.5 \ bo1
\ \ \ act.all.6 \ bo2
\ \ \ act.all.7 \ bo3
\ \ ================================================================
\ \ \ Use inverse transform (i.e., <--> -1)
\ \ \ The first i.ele  and jwp.level is for generating wavelet curve
\ \ \ The second i.ele is for blowup
\ \ \ The second jwp.level is for blowup scale (i.e., Eq. to n.power - jwp.level)
\ set.let
\ set.123
\ blow
\ ================================================================
\ ----------------------------------------------------------------
\ ================================================================
\ ---------------------------------------- - \ \ For DnR
\ act.all
\
\ act.all.1 \ LMS
\ act.all.2 \ ONA
\ act.all.3 \ ons
\ act.all.4 \ onc
\ act.all.5 \ bo1
\ act.all.6 \ bo2
\ act.all.7 \ bo3
\ ================================================================
: ColRowIni
  1  gp.flag  :=
  1  file.no  :=
  \ 1  plot.gp  :=
  plot.gp 1 +  plot.gp  :=
  \
  i.ele  2  jwp.level **  <=
  if
     " Phi-"  PhiPsi.str ":=
  else
     " Psi-"  PhiPsi.str ":=
  then
;
: app.ramp.col
  wlet tmp.s.u ":=
  tmp.s.u "upper "drop
  " $$[("  file.no 0 n>s "cat " ," "cat
  123.ary.n []size swap drop 0 n>s "cat
  \ " ) " "cat tmp.s.u "cat " _" "cat PhiPsi.str "cat " <" "cat
  " ) " "cat PhiPsi.str "cat " <" "cat
  \ ---------------------------------------- -
  \ \ $$[(5,1034)<1024>]
  \ 123.ary.n []size swap drop 0 n>s
  \ ---------------------------------------- -
  \ \ $$[(5,1034) Psi-<POL2_BP4,Element:12,Power:10)>]
  " P" "cat  wp.flag  0 n>s "cat
  " L" jwp.level 0 n>s "cat
```

```
  " _BP" "cat 2 jwp.level ** 0 n>s "cat
  \ " ,Power:" "cat n.power 0 n>s "cat
  " ,Ori" "cat i.ele 0 n>s "cat
  " ,Pw" "cat n.power 0 n>s "cat
  \ ---------------------------------------- -
  "cat
  " >]" "cat
  123.ary.s "[ 1 ] ":=
  \
  123.ary.n []size swap drop real ramp 1.0 * becomes> 123.ary.n
  \
  to.123
  file.no 1 + file.no := \ post inc of column
;

\ ---------------------------------------- -
\ \ Use act.all...  (Large wks array)
\ 0 plot.gp  :=
\ ColRowIni  " psi-LMS.wks"   " 000.wks"  set.123.BP   act.all.1 \ LMS
\ ColRowIni  " psi-ONA.wks"   " 000.wks"  set.123.BP   act.all.2 \ ONA
\ ColRowIni  " psi-ONS.wks"   " 000.wks"  set.123.BP   act.all.3 \ ONS
\ ColRowIni  " psi-ONC.wks"   " 000.wks"  set.123.BP   act.all.4 \ ONC
\ ColRowIni  " psi-BO1.wks"   " 000.wks"  set.123.BP   act.all.5 \ BO1
\ ColRowIni  " psi-BO2.wks"   " 000.wks"  set.123.BP   act.all.6 \ BO2
\ ColRowIni  " psi-BO3.wks"   " 000.wks"  set.123.BP   act.all.7 \ BO3
\ -----------------------------------------------------------------
\ \ Use act.Gp...  (small and refined wks array for figure plotting)
ACT.Gp.Ini
ColRowIni PhiPsi.str " LMS.wks"  "cat " 0.0" set.123.BP act.Gp.LMS  app.ramp.col
ColRowIni PhiPsi.str " ONC.wks"  "cat " 0.0" set.123.BP act.Gp.onC  app.ramp.col
ColRowIni PhiPsi.str " ONA1.wks" "cat " 0.0" set.123.BP act.Gp.onA1 app.ramp.col
ColRowIni PhiPsi.str " ONA2.wks" "cat " 0.0" set.123.BP act.Gp.onA2 app.ramp.col
ColRowIni PhiPsi.str " ONS1.wks" "cat " 0.0" set.123.BP act.Gp.onS1 app.ramp.col
ColRowIni PhiPsi.str " ONS2.wks" "cat " 0.0" set.123.BP act.Gp.onS2 app.ramp.col
ColRowIni PhiPsi.str " BO10.wks" "cat " 0.0" set.123.BP act.Gp.bo10 app.ramp.col
ColRowIni PhiPsi.str " BO1D.wks" "cat " 0.0" set.123.BP act.Gp.bo1D app.ramp.col
ColRowIni PhiPsi.str " BO20.wks" "cat " 0.0" set.123.BP act.Gp.bo20 app.ramp.col
ColRowIni PhiPsi.str " BO2D.wks" "cat " 0.0" set.123.BP act.Gp.bo2D app.ramp.col
ColRowIni PhiPsi.str " BO30.wks" "cat " 0.0" set.123.BP act.Gp.bo30 app.ramp.col
ColRowIni PhiPsi.str " BO3D.wks" "cat " 0.0" set.123.BP act.Gp.bo3D app.ramp.col
\ \ -----------------------------------------------------------------
\ \ : Plt.shl
\ \   PhiPsi.str "upper " PHI" "=
\ \   if
\ \     cdtNow load Plt_FW.shl \
\ \   then
\ \   PhiPsi.str "upper " PSI" "=
\ \   if
\ \     cdtNow load Plt_MW.shl \
\ \   then
\ \ ;
\ \ \ Plt.shl \ \ load execution depth exceeded
\ \ =================================================================
\ \ -----------------------------------------------------------------
\ \ =================================================================
```

```
\ PhiPsi.str tmp.s ":=
\ fa
\ 4 string PhiPsi.str
\ tmp.s PhiPsi.str ":=
\ \ ==============================================================
\ cdt load FigA-Ini.asy \
\ 2.0 ps.linewidth.now :=
\ \ ==============================================================
\ : file.name
\   stack.data tmp.s " LMS.WKS" "cat
\   stack.load " MnF-LMS.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\
\ : file.name
\   stack.data PhiPsi.str " LMS.WKS" "cat
\   stack.load " MnF-LMS.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\
\ : file.name
\   stack.data PhiPsi.str " ONC.WKS" "cat
\   stack.load " MnF-ONC.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\
\ : file.name
\   stack.data PhiPsi.str " ONA1.WKS" "cat
\   stack.load " MnF-ONA1.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\
\ : file.name
\   stack.data PhiPsi.str " ONA2.WKS" "cat
\   stack.load " MnF-ONA2.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\
\ : file.name
\   stack.data PhiPsi.str " ONS1.WKS" "cat
\   stack.load " MnF-ONS1.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\
\ : file.name
\   stack.data PhiPsi.str " ONS2.WKS" "cat
\   stack.load " MnF-ONS2.BPF"
\ ;
\ cdt load FigACore.asy \
```

```
\ forget file.name
\
\ : file.name
\   stack.data PhiPsi.str " B010.WKS" "cat
\   stack.load " MnF-B010.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\
\ : file.name
\   stack.data PhiPsi.str " B01D.WKS" "cat
\   stack.load " MnF-B01D.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\
\ : file.name
\   stack.data PhiPsi.str " B020.WKS" "cat
\   stack.load " MnF-B020.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\
\ : file.name
\   stack.data PhiPsi.str " B02D.WKS" "cat
\   stack.load " MnF-B02D.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\
\ : file.name
\   stack.data PhiPsi.str " B030.WKS" "cat
\   stack.load " MnF-B030.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\
\ : file.name
\   stack.data PhiPsi.str " B03D.WKS" "cat
\   stack.load " MnF-B03D.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\ \ ================================================================
```

R_TnD.shl

勇伯程式庫 A01493-20070125-A044 ❖ January 25, 2007 ❖ [Page count: 9]

```
: R_TnD.shl ; \ Wavelet translation and Dilation \ -*- txt:asy -*-

fa sd \ swd
\ 1024 n.wt1 := \ \ Use ini.par.f
512 n.wt1 :=  \ \ Use ini.par.chirp.al
cdtNow load LLet-000.asy \

\ ================================================================
\ ----------------------------------------------------------------
: ini.par.chirp.al
  " Chirp-al.dat" now.file ":=
  512  n.wt1    :=
  200  s.freq.keep :=          \ Sampling freq -- original rate
  1    i.inc    :=             \ array index Inc for sub-sampling
;
\ ----------------------------------------------------------------
: ini.par.f
  " f0w6020.2" now.file ":=
  1024 n.wt1    :=
  40   s.freq.keep :=          \ Sampling freq -- original rate
  1    i.inc    :=             \ array index Inc for sub-sampling
;
: ini.par.b
  " b0w6020.4" now.file ":=
  1024 n.wt1    :=
  200  s.freq.keep :=          \ Sampling freq -- original rate
  4    i.inc    :=             \ array index Inc for sub-sampling
\   0     abs.flag    :=         \ absolute value norm
\   0     filter.flag :=         \ percentage of norm energy kept
\   0.975 norm.keep.p.best :=    \ for initial filtering
\   6.0   p.b.e :=               \ Pass band end (Hz)
\   6.0   t.w    :=              \ Transition width (Hz)
\   0     best.flag    :=        \ unclear, possibly not necessary (WPM)
\   100   #.div :=               \ foe entropy calculation
;
\ ================================================================
-1   isign := \ ." <- or -> :"
" D" act.flag.str ":= \ 3    act.flag :=
\ ini.par.b
ini.par.chirp.al
\ ================================================================
set.act
set.let
set.123
\ for P N E -- set.let and set.123 are Moved into word within set.act
lee
\ ----------------------------------------------------------------
\ \ Keep all default setings for all the above inputs
\ ================================================================
\ ---------------------------------------- ■ \ \ For Blowup
\ act.all
\
```

```
\ act.all.1 \ LMS
\ act.all.2 \ ONA
\ act.all.3 \ ons
\ act.all.4 \ onc
\ act.all.5 \ bo1
\ act.all.6 \ bo2
\ act.all.7 \ bo3
\ =================================================================
\ \ Use inverse transform (i.e., <--> -1)
\ \ The first i.ele  and jwp.level is for generating wavelet curve
\ \ The second i.ele is for blowup
\ \ The second jwp.level is for blowup scale (i.e., Eq. to n.power - jwp.level)
\ set.let
\ set.123
\ blow
\ =================================================================
\ -----------------------------------------------------------------
\ =================================================================

\ =================================================================
: ColRowIni
  1  gp.flag  :=
  1  file.no  :=
  \ 1  plot.gp  :=
  plot.gp 1 +  plot.gp  :=
  \
  i.ele  2  jwp.level **  <=
  if
     " Phi-"  PhiPsi.str ":=
  else
     " Psi-"  PhiPsi.str ":=
  then
;
\ : rename.wks \ [ "str - ]
\   cdanow
\   123.file.name defer> rename \ copy
\   tmp.s.u  "cat " .wks" "cat
\   "dup
\   defer> to
\   defer> delete
\ ;
: rename.wks \ [ "str - ]
  cdanow
  123.file.name defer> copy
  tmp.s.u  "cat " .wks" "cat
  defer> to
  123.file.name defer> delete
;
: copy.wks \ [ "str - ]
  cdanow
  123.file.name defer> copy
  tmp.s.u  "cat " .wks" "cat
  defer> to
\ cdanow
\ 123.file.name defer> copy
```

```
\ " 000.wks" defer> to
;
8 string copy.type
: app.wks.col
  " $$[(" file.no 0 n>s "cat " ," "cat
  123.ary.n []size swap drop 0 n>s "cat
  " ) " "cat
  tmp.s.u  "cat
\   " -P" "cat
\   wp.flag  0 n>s "cat
\   " L" "cat jwp.level 0 n>s "cat
  " ]" "cat
  123.ary.s "[ 1 ] ":=
  123.ary.n []size swap drop real ramp 1.0 * becomes> 123.ary.n
  \
  to.123
  file.no 1 + file.no := \ post inc of column
  \ ----------------------------------------
  copy.type
  rename.wks \ copy.wks
;

: to.wks
  " Ori" i.ele 0 n>s "cat
  " -P" "cat wp.flag  0 n>s "cat
  " L" "cat jwp.level 0 n>s "cat
  " _BP"  "cat 2 jwp.level 1 + ** 0 n>s "cat
  123.ary.s "[ 1 ] ":=
  swt.t becomes> 123.ary.n
  to.123
  cr cr ." Column.#"  file.no n.type
  file.no 1 + file.no :=
;
\ =================================================================
: lee.auto
  isign  -1  =
  if
    n.wt1  dp.real ramp  becomes> b
    1  0 fix.format
    0.  b  :=
    \ cr ." JWP Level(" wlet s.type ." ):"  jwp.level  get.val   jwp.level :=
\ -----------------------------------------------------------------
    cr ." Element "
    wp.flag 1 =
    if
      " (WP: located at level "  jwp.level 0 n>s "cat " )" "cat s.type.3
    else
      " (WB: min. level at "  jwp.level 0 n>s "cat " )" "cat s.type.3
    then
    ."  (Phi<-"
    1  0 fix.format
    jwp.level  9 >=
    if                          4
    else
      jwp.level  6 >=
```

```
   if           3
   else
    jwp.level  3 >=
    if          2
    else        1
    then
   then
  then
  dup  num.dig :=   0  fix.format
  2 jwp.level    ** n.type ." |->Psi:[WB"
  \ ----------------------------------------
  wp.flag 1 =
  if
   ." <-"
   1  0 fix.format
   jwp.level  9 >=
   if            4
   else
    jwp.level  6 >=
    if           3
    else
     jwp.level  3 >=
     if          2
     else        1
     then
    then
   then
   dup  num.dig :=   0  fix.format
   2 jwp.level 1 + ** n.type ." |->WP]): "
  else
   ." ]): "
  then
\ ----------------------------------------------------------------
\  cr ." Element (WB<-"
\  1  0 fix.format
\  jwp.level  9 >=
\  if            4
\  else
\   jwp.level  6 >=
\   if           3
\   else
\    jwp.level  3 >=
\    if          2
\    else        1
\    then
\   then
\  then
\  dup  num.dig :=   0  fix.format
\  2 jwp.level 1 + ** n.type ." |->WP):"
  3 0 fix.format
  \ i.ele get.val i.ele :=
  1 0 fix.format
  \ cr ." Clear (1) or Overlap:"    i.lin     get.val   i.lin     :=
  1.  b [ i.ele ] :=
  b  inverse.wt
```

```
    wp.flag 1 =
    if
      drop  swp.t
    else
      drop  swt.t
    then
  else
    {hp.wnd}  " Have you set B (in --> mode)?" s.type   \ my.pause
    cr ." Choose array for Forward Transform "
    " 0:default B, 1:WB.t, 2:WP.t" s.type.3
    0 get.val
    case
      1  of   swt.t  endof
      2  of   swp.t  endof
      0  of   b      endof
    endcase
    previous.window
    forward.wt      \ always d.t for WB and WP -- set Wp.flag
  then
  n.power  10 = if  sub[ 1 , 1000 ] then
  n.power   9 = if  sub[ 1 ,  500 ] then
  n.power   8 = if  sub[ 1 ,  250 ] then
  n.power   7 = if  sub[ 1 ,  120 ] then
  n.power   6 = if  sub[ 1 ,   70 ] then
  i.lin
  i.lin     1 =
  if  yap   lab.spe   else  ydp  then
  10 5 fix.format
;
\ -----------------------------------------------------------------
\ : lee.auto \ no pause for input
\   isign  -1  =
\   if
\     n.wt1  dp.real ramp  becomes> b
\     1  0 fix.format
\     0.  b  :=
\     \ \ Auto \  cr ." JWP Level(" wlet s.type ." ):"  jwp.level  get.val
jwp.level :=
\     cr ." Element (WB--"
\     1  0 fix.format
\     jwp.level  9 >=
\     if                        4
\     else
\       jwp.level  6 >=
\       if                      3
\       else
\         jwp.level  3 >=
\         if                    2
\         else                  1
\         then
\       then
\     then
\     dup  num.dig :=    0  fix.format
\     2 jwp.level 1 + ** n.type ." --WP):"
\     3 0 fix.format
```

```
\      \ \ Auto \  i.ele get.val i.ele :=    1 0 fix.format
\      \ \ Auto \  cr ." Clear (1) or Overlap:"     i.lin      get.val   i.lin
:=
\      1.  b [ i.ele ] :=
\      b  inverse.wt
\      wp.flag 1 =
\      if
\        drop  swp.t
\      else
\        drop  swt.t
\      then
\    else
\      {hp.wnd}  " Have you set B (in --> mode)?" s.type  my.pause
\      \ \ Auto \  cr ." Choose  1:WB.t,  2:WP.t,  0:default B"  0 get.val
\      0 \ \ Auto added\
\      case
\        1  of   swt.t  endof
\        2  of   swp.t  endof
\        0  of   b      endof
\      endcase
\      previous.window
\      forward.wt      \ always d.t for WB and WP -- set Wp.flag
\    then
\    n.power  10 = if  sub[ 1 , 1000 ] then
\    n.power   9 = if  sub[ 1 ,  500 ] then
\    n.power   8 = if  sub[ 1 ,  250 ] then
\    n.power   7 = if  sub[ 1 ,  120 ] then
\    n.power   6 = if  sub[ 1 ,   70 ] then
\    i.lin
\    i.lin     1 =
\    if  yap   lab.spe   else  ydp  then
\    10 5 fix.format
\ ;
\ =================================================================
\ =================================================================
: scale.012
" Dil" copy.type ":=
0   jwp.level :=
3   i.ele     :=
lee.auto
to.wks

1   jwp.level :=
6   i.ele     :=
lee.auto
to.wks

2   jwp.level :=
12  i.ele     :=
lee.auto
to.wks

3   jwp.level :=
20  i.ele     :=
lee.auto
```

```
to.wks

4   jwp.level  :=
54  i.ele      :=
lee.auto
to.wks

5   jwp.level  :=
74  i.ele      :=
lee.auto
to.wks

6   jwp.level  :=
200 i.ele      :=
lee.auto
to.wks

7   jwp.level  :=
450 i.ele      :=
lee.auto
to.wks
;
\ -----------------------------------------------------------------
: position.012
" Tra" copy.type  ":=
3   jwp.level  :=
16  i.ele      :=
lee.auto
to.wks

3   jwp.level  :=
20  i.ele      :=
lee.auto
to.wks

3   jwp.level  :=
24  i.ele      :=
lee.auto
to.wks

3   jwp.level  :=
28  i.ele      :=
lee.auto
to.wks

3   jwp.level  :=
32  i.ele      :=
lee.auto
to.wks
;
\ =================================================================
\ =================================================================
" bo220" wlet  ":=
0 jwp.level  :=
ColRowIni
```

```
1 file.no :=
set.let
set.123
\ ---------------------------------------- -
scale.012
wlet tmp.s.u ":=
tmp.s.u "upper "drop
app.wks.col
\ =================================================================
" bo22d" wlet ":=
0 jwp.level :=
ColRowIni
1 file.no :=
set.let
set.123
\ ---------------------------------------- -
scale.012
wlet tmp.s.u ":=
tmp.s.u "upper "drop
app.wks.col
\ =================================================================
" bo31d" wlet ":=
0 jwp.level :=
ColRowIni
1 file.no :=
set.let
set.123
\ ---------------------------------------- -
scale.012
wlet tmp.s.u ":=
tmp.s.u "upper "drop
app.wks.col
\ =================================================================
" bo370" wlet ":=
0 jwp.level :=
ColRowIni
1 file.no :=
set.let
set.123
\ ---------------------------------------- -
scale.012
wlet tmp.s.u ":=
tmp.s.u "upper "drop
app.wks.col
\ =================================================================
" on66a" wlet ":=
0 jwp.level :=
ColRowIni
1 file.no :=
set.let
set.123
\ ---------------------------------------- -
scale.012
wlet tmp.s.u ":=
tmp.s.u "upper "drop
```

```
app.wks.col
\ =================================================================
\ -----------------------------------------------------------------
\ =================================================================
" bo31d" wlet ":=
0   jwp.level :=
ColRowIni
1 file.no :=
set.let
set.123
\ ---------------------------------------- -
position.012
wlet tmp.s.u ":=
tmp.s.u "upper "drop
app.wks.col
\ =================================================================
" on55c" wlet ":=
0   jwp.level :=
ColRowIni
1 file.no :=
set.let
set.123
\ ---------------------------------------- -
position.012
wlet tmp.s.u ":=
tmp.s.u "upper "drop
app.wks.col
\ =================================================================
```

R_Blo.shl

勇伯程式庫 A01494-20070125-A045 ❖ January 25, 2007 ❖ [Page count: 5]

```
: R-Blo.shl ; \ Blowups of wavelet bases and packets  \ -*- txt:asy -*-

fa sd \ swd

\ 1024 n.wt1 := \ \ Use ini.par.f
512 n.wt1 :=  \ \ Use ini.par.chirp.al

\ =================================================================
\ -----------------------------------------------------------------
cdtNow load LLet-000.asy \
\ =================================================================
\ -----------------------------------------------------------------
: ini.par.chirp.al
  " Chirp-al.dat" now.file ":=
  512  n.wt1    :=
  200  s.freq.keep :=          \ Sampling freq -- original rate
  1    i.inc    :=             \ array index Inc for sub-sampling
;
\ -----------------------------------------------------------------
: ini.par.f
  " f0w6020.2" now.file ":=
  1024 n.wt1    :=
  40   s.freq.keep :=          \ Sampling freq -- original rate
  1    i.inc    :=             \ array index Inc for sub-sampling
;
: ini.par.b
  " b0w6020.4" now.file ":=
  1024 n.wt1    :=
  200  s.freq.keep :=          \ Sampling freq -- original rate
  4    i.inc    :=             \ array index Inc for sub-sampling
\   0     abs.flag    :=         \ absolute value norm
\   0     filter.flag  :=        \ percentage of norm energy kept
\   0.975 norm.keep.p.best  :=   \ for initial filtering
\   6.0   p.b.e  :=              \ Pass band end (Hz)
\   6.0   t.w    :=              \ Transition width (Hz)
\   0     best.flag    :=        \ unclear, possibly not necessary (WPM)
\   100   #.div  :=              \ foe entropy calculation
;
\ =================================================================
-1   isign := \ ." <- or -> :"
" D" act.flag.str ":= \ 3    act.flag :=
\ ini.par.b \ \ For 1024
ini.par.chirp.al \ \ For 512
\ \ =================================================================
\ set.act
\ set.let
\ set.123
\ \ for P N E -- set.let and set.123 are Moved into word within set.act
\ lee
\ \ \ act.all
\ \ \ =================================================================
\ \
```

```
\ \ \ act.all.1 \ LMS
\ \ \ act.all.2 \ ONA
\ \ \ act.all.3 \ ons
\ \ \ act.all.4 \ onc
\ act.all.5 \ bo1
\ \ \ act.all.6 \ bo2
\ \ \ act.all.7 \ bo3
\ \ \ ===============================================================
\ ===============================================================
set.act
set.let
set.123
\ for P N E -- set.let and set.123 are Moved into word within set.act
lee
\ ---------------------------------------------------------------
blow
\ ---------------------------------------------------------------
\ \ Keep all default setings for all the above inputs
\ \ ===============================================================
\ ---------------------------------------- ■ \ \ For Blowup
\ \ act.all
\
\ \ act.all.1 \ LMS
\ \ act.all.2 \ ONA
\ \ act.all.3 \ ons
\ \ act.all.4 \ onc
\ \ act.all.5 \ bo1
\ \ act.all.6 \ bo2
\ \ act.all.7 \ bo3
\ \ ===============================================================
\ \ Use inverse transform (i.e., <--> -1)
\ \ The first i.ele  and jwp.level is for generating wavelet curve
\ \ The second i.ele is for blowup
\ \ The second jwp.level is for blowup scale (i.e., Eq. to n.power - jwp.level)
\ set.let
\ set.123
\ blow
\ ===============================================================
\ ---------------------------------------------------------------
\ ===============================================================
\ ---------------------------------------- ■ \ \ For DnR
\ : ColRowIni
\   1  gp.flag   :=
\   1  file.no   :=
\   \ 1  plot.gp   :=
\   plot.gp 1 +  plot.gp   :=
\   \
\   i.ele  2  jwp.level **  <=
\   if
\      " PHI-"  PhiPsi.str ":=
\   else
\      " PSI-"  PhiPsi.str ":=
\   then
\ ;
\ : app.ramp.col
```

```
\   " $$[("  file.no 0 n>s "cat " ," "cat
\   123.ary.n []size swap drop 0 n>s "cat
\   " ) " "cat PhiPsi.str "cat " <" "cat
\   \ ----------------------------------------
\   \ \ $$[(5,1034)<1024>]
\   \ 123.ary.n []size swap drop 0 n>s
\   \ ----------------------------------------
\   \ \ $$[(5,1034) Psi-<Level:2_Pt4,Power:10,Element:12)>]
\   " Level:" jwp.level 0 n>s "cat
\   " _Pt" "cat 2 jwp.level ** 0 n>s "cat
\   \ " ,Power:" "cat n.power 0 n>s "cat
\   " ,Pw" "cat n.power 0 n>s "cat
\   " ,Element:" "cat i.ele 0 n>s "cat
\   \ ----------------------------------------
\   "cat
\   " >]" "cat
\   123.ary.s "[ 1 ] ":=
\   \
\   123.ary.n []size swap drop real ramp 1.0 * becomes> 123.ary.n
\   \
\   to.123
\   file.no 1 + file.no := \ post inc of column
\ ;
\
\ \ ----------------------------------------
\ \ 0 plot.gp  :=
\ \ ColRowIni  " psi-LMS.wks"   " 000.wks"  set.123.BP   act.all.1 \ LMS
\ \ ColRowIni  " psi-ONA.wks"   " 000.wks"  set.123.BP   act.all.2 \ ONA
\ \ ColRowIni  " psi-ONS.wks"   " 000.wks"  set.123.BP   act.all.3 \ ONS
\ \ ColRowIni  " psi-ONC.wks"   " 000.wks"  set.123.BP   act.all.4 \ ONC
\ \ ColRowIni  " psi-BO1.wks"   " 000.wks"  set.123.BP   act.all.5 \ BO1
\ \ ColRowIni  " psi-BO2.wks"   " 000.wks"  set.123.BP   act.all.6 \ BO2
\ \ ColRowIni  " psi-BO3.wks"   " 000.wks"  set.123.BP   act.all.7 \ BO3
\ \ -----------------------------------------------------------------
\ ACT.Gp.Ini
\ ColRowIni PhiPsi.str " LMS.wks"  "cat " 0.0" set.123.BP act.Gp.LMS  app.ramp.col
\ ColRowIni PhiPsi.str " ONC.wks"  "cat " 0.0" set.123.BP act.Gp.onC  app.ramp.col
\ ColRowIni PhiPsi.str " ONA1.wks" "cat " 0.0" set.123.BP act.Gp.onA1 app.ramp.col
\ ColRowIni PhiPsi.str " ONA2.wks" "cat " 0.0" set.123.BP act.Gp.onA2 app.ramp.col
\ ColRowIni PhiPsi.str " ONS1.wks" "cat " 0.0" set.123.BP act.Gp.onS1 app.ramp.col
\ ColRowIni PhiPsi.str " ONS2.wks" "cat " 0.0" set.123.BP act.Gp.onS2 app.ramp.col
\ ColRowIni PhiPsi.str " BO10.wks" "cat " 0.0" set.123.BP act.Gp.bo10 app.ramp.col
\ ColRowIni PhiPsi.str " BO1D.wks" "cat " 0.0" set.123.BP act.Gp.bo1D app.ramp.col
\ ColRowIni PhiPsi.str " BO20.wks" "cat " 0.0" set.123.BP act.Gp.bo20 app.ramp.col
\ ColRowIni PhiPsi.str " BO2D.wks" "cat " 0.0" set.123.BP act.Gp.bo2D app.ramp.col
\ ColRowIni PhiPsi.str " BO30.wks" "cat " 0.0" set.123.BP act.Gp.bo30 app.ramp.col
\ ColRowIni PhiPsi.str " BO3D.wks" "cat " 0.0" set.123.BP act.Gp.bo3D app.ramp.col
\
\ \ : Plt.shl
\ \   PhiPsi.str "upper " PHI" "=
\ \   if
\ \     cdtNow load Plt_FW.shl \
\ \   then
\ \   PhiPsi.str "upper " PSI" "=
\ \   if
```

```
\ \      cdtNow load Plt_MW.shl \
\ \    then
\ \ ;
\ \ \ Plt.shl \ \ load execution depth exceeded
\
\ \ =================================================================
\ \ -----------------------------------------------------------------
\ \ =================================================================
\ PhiPsi.str tmp.s ":=
\ fa
\ 4 string PhiPsi.str
\ tmp.s PhiPsi.str ":=
\ \ =================================================================
\ cdt load FigA-Ini.asy \
\ 2.0 ps.linewidth.now :=
\ \ =================================================================
\ : file.name
\   stack.data tmp.s " LMS.WKS" "cat
\   stack.load " MnF-LMS.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\
\ : file.name
\   stack.data PhiPsi.str " LMS.WKS" "cat
\   stack.load " MnF-LMS.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\
\ : file.name
\   stack.data PhiPsi.str " ONC.WKS" "cat
\   stack.load " MnF-ONC.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\
\ : file.name
\   stack.data PhiPsi.str " ONA1.WKS" "cat
\   stack.load " MnF-ONA1.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\
\ : file.name
\   stack.data PhiPsi.str " ONA2.WKS" "cat
\   stack.load " MnF-ONA2.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\
\ : file.name
\   stack.data PhiPsi.str " ONS1.WKS" "cat
\   stack.load " MnF-ONS1.BPF"
\ ;
```

```
\ cdt load FigACore.asy \
\ forget file.name
\
\ : file.name
\   stack.data PhiPsi.str " ONS2.WKS" "cat
\   stack.load " MnF-ONS2.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\
\ : file.name
\   stack.data PhiPsi.str " BO10.WKS" "cat
\   stack.load " MnF-BO10.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\
\ : file.name
\   stack.data PhiPsi.str " BO1D.WKS" "cat
\   stack.load " MnF-BO1D.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\
\ : file.name
\   stack.data PhiPsi.str " BO20.WKS" "cat
\   stack.load " MnF-BO20.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\
\ : file.name
\   stack.data PhiPsi.str " BO2D.WKS" "cat
\   stack.load " MnF-BO2D.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\
\ : file.name
\   stack.data PhiPsi.str " BO30.WKS" "cat
\   stack.load " MnF-BO30.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\
\ : file.name
\   stack.data PhiPsi.str " BO3D.WKS" "cat
\   stack.load " MnF-BO3D.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\ \ ================================================================
```

R_Lev.shl

勇伯程式庫 A01495-20070125-A046 ❖ January 25, 2007 ❖ [Page count: 7]

```
: R-Lev.shl ; \ Show WB and WP at various levels/points \ -*- txt:asy -*-

fa sd \ swd

\ 1024 n.wt1 := \ \ Use ini.par.f
512 n.wt1 :=  \ \ Use ini.par.chirp.al
cdtNow load LLet-000.asy \

\ ==============================================================
\ --------------------------------------------------------------
: ini.par.chirp.al
  " Chirp-al.dat" now.file ":=
  512  n.wt1    :=
  200  s.freq.keep :=          \ Sampling freq -- original rate
  1    i.inc    :=             \ array index Inc for sub-sampling
;
\ --------------------------------------------------------------
: ini.par.f
  " f0w6020.2" now.file ":=
  1024 n.wt1    :=
  40   s.freq.keep :=          \ Sampling freq -- original rate
  1    i.inc    :=             \ array index Inc for sub-sampling
;
: ini.par.b
  " b0w6020.4" now.file ":=
  1024 n.wt1    :=
  200  s.freq.keep :=          \ Sampling freq -- original rate
  4    i.inc    :=             \ array index Inc for sub-sampling
\   0     abs.flag    :=         \ absolute value norm
\   0     filter.flag :=         \ percentage of norm energy kept
\   0.975 norm.keep.p.best  :=   \ for initial filtering
\   6.0   p.b.e  :=              \ Pass band end (Hz)
\   6.0   t.w    :=              \ Transition width (Hz)
\   0     best.flag    :=        \ unclear, possibly not necessary (WPM)
\   100   #.div  :=              \ foe entropy calculation
;
\ ==============================================================
-1   isign  := \ ." <- or -> :"
" D" act.flag.str ":= \ 3    act.flag :=
\ ini.par.b
ini.par.chirp.al
\ ==============================================================
cdtNow load LLet-Lev.asy \ lee.auto
\ ==============================================================
set.act
set.let.auto
set.123.auto
\ for P N E -- set.let and set.123 are Moved into word within set.act
lee.auto
\ --------------------------------------------------------------
\ \ Keep all default setings for all the above inputs
\ ==============================================================
```

```
\ ---------------------------------------- ■ \ \ For Blowup
\ act.all
\
\ act.all.1 \ LMS
\ act.all.2 \ ONA
\ act.all.3 \ ons
\ act.all.4 \ onc
\ act.all.5 \ bo1
\ act.all.6 \ bo2
\ act.all.7 \ bo3
\ =================================================================
\ \ Use inverse transform (i.e., <--> -1)
\ \ The first i.ele  and jwp.level is for generating wavelet curve
\ \ The second i.ele is for blowup
\ \ The second jwp.level is for blowup scale (i.e., Eq. to n.power - jwp.level)
\ set.let
\ set.123
\ blow
\ =================================================================
\ -----------------------------------------------------------------
\ =================================================================

\ =================================================================
: ColRowIni
  1  gp.flag  :=
  1  file.no  :=
  \ 1  plot.gp  :=
  plot.gp 1 +  plot.gp  :=
  \
  i.ele  2  jwp.level **  <=
  if
     " Phi-"  PhiPsi.str ":=
  else
     " Psi-"  PhiPsi.str ":=
  then
;
\ : rename.wks \ [ "str - ]
\   cdanow
\   123.file.name defer> rename \ copy
\   tmp.s.u  "cat " .wks" "cat
\   "dup
\   defer> to
\   defer> delete
\ ;
: rename.wks \ [ "str - ]
  cdanow
  123.file.name defer> copy
  tmp.s.u  "cat " .wks" "cat
  defer> to
  123.file.name defer> delete
;
: copy.wks \ [ "str - ]
  cdanow
  123.file.name defer> copy
  tmp.s.u  "cat " .wks" "cat
```

```
  defer> to
\ cdanow
\ 123.file.name defer> copy
\ " 000.wks" defer> to
;
8 string copy.type
: app.wks.col
  " $$[(" file.no 0 n>s "cat " ," "cat
  123.ary.n []size swap drop 0 n>s "cat
  " ) " "cat
  tmp.s.u  "cat
  " -P" "cat
  wp.flag  0 n>s "cat
\   " L" "cat jwp.level 0 n>s "cat
  " ]" "cat
  123.ary.s "[ 1 ] ":=
  123.ary.n []size swap drop real ramp 1.0 * becomes> 123.ary.n
  \
  to.123
  file.no 1 + file.no := \ post inc of column
  \ -----------------------------------------
  copy.type
  rename.wks \ copy.wks
;

: to.wks
  " P" wp.flag  0 n>s "cat
  " L" "cat jwp.level 0 n>s "cat
  " _BP"  "cat 2 jwp.level 1 + ** 0 n>s "cat
  "  Ori" "cat i.ele 0 n>s "cat
  123.ary.s "[ 1 ] ":=
  \ swt.t becomes> 123.ary.n
  swp.t becomes> 123.ary.n
  to.123
  cr cr ." Column.#"  file.no n.type
  file.no 1 + file.no :=
;
\ ==================================================================
\ ==================================================================
: wbp.set.ini
\ 1 wp.flag :=
1 jwp.level :=
ColRowIni
1 file.no :=
set.let.auto
set.123.auto
;
\ -----------------------------------------
\ ==================================================================
: wbp.scale.4
wbp.set.ini

1   jwp.level :=
48  i.ele     :=
lee.auto
```

```
to.wks

2   jwp.level  :=
48  i.ele      :=
lee.auto
to.wks

3   jwp.level  :=
48  i.ele      :=
lee.auto
to.wks

4   jwp.level  :=
48  i.ele      :=
lee.auto
to.wks
;
\ -----------------------------------------------------------------
: wbp.scale.6.70
wbp.set.ini

0   jwp.level  :=
70  i.ele      :=
lee.auto
to.wks

1   jwp.level  :=
70  i.ele      :=
lee.auto
to.wks

2   jwp.level  :=
70  i.ele      :=
lee.auto
to.wks

3   jwp.level  :=
70  i.ele      :=
lee.auto
to.wks

4   jwp.level  :=
70  i.ele      :=
lee.auto
to.wks

5   jwp.level  :=
70  i.ele      :=
lee.auto
to.wks
;
\ -----------------------------------------------------------------
: wbp.scale.6.100
wbp.set.ini
```

```
0   jwp.level :=
100  i.ele     :=
lee.auto
to.wks

1   jwp.level :=
100  i.ele     :=
lee.auto
to.wks

2   jwp.level :=
100  i.ele     :=
lee.auto
to.wks

3   jwp.level :=
100  i.ele     :=
lee.auto
to.wks

4   jwp.level :=
100  i.ele     :=
lee.auto
to.wks

5   jwp.level :=
100  i.ele     :=
lee.auto
to.wks
;
\ -----------------------------------------------------------------
: del.and.rento
  "dup
  defer> delete
  "swap
  defer> rename
  defer> to
;
\ " WB3ON11C.WKS" " WBON11C3.WKS" del.and.rento
\ =================================================================
\ =================================================================
" on22a" wlet ":=
\ -----------------------------------------------------------------
wlet tmp.s.u ":=
tmp.s.u "upper "drop
\ -----------------------------------------------------------------
1 wp.flag :=
\ ----------------------------------------
" WP1" copy.type ":=
wbp.scale.4
app.wks.col
\ ----------------------------------------
" WP2" copy.type ":=
wbp.scale.6.70
app.wks.col
```

```
\ ---------------------------------------- -
" WP3" copy.type ":=
wbp.scale.6.100
app.wks.col
\ -----------------------------------------------------------------
0 wp.flag :=
\ ---------------------------------------- -
" WB1" copy.type ":=
wbp.scale.4
app.wks.col
\ ---------------------------------------- -
" WB2" copy.type ":=
wbp.scale.6.70
app.wks.col
\ ---------------------------------------- -
" WB3" copy.type ":=
wbp.scale.6.100
app.wks.col
\ =================================================================
cdanow
" WP1" tmp.s.u "cat " .wks" "cat
" WP" tmp.s.u "cat " 1.wks" "cat del.and.rento
" WP2" tmp.s.u "cat " .wks" "cat
" WP" tmp.s.u "cat " 2.wks" "cat del.and.rento
" WP3" tmp.s.u "cat " .wks" "cat
" WP" tmp.s.u "cat " 3.wks" "cat del.and.rento

" WB1" tmp.s.u "cat " .wks" "cat
" WB" tmp.s.u "cat " 1.wks" "cat del.and.rento
" WB2" tmp.s.u "cat " .wks" "cat
" WB" tmp.s.u "cat " 2.wks" "cat del.and.rento
" WB3" tmp.s.u "cat " .wks" "cat
" WB" tmp.s.u "cat " 3.wks" "cat del.and.rento

\ =================================================================
" on11c" wlet ":=
\ -----------------------------------------------------------------
wlet tmp.s.u ":=
tmp.s.u "upper "drop
\ -----------------------------------------------------------------
1 wp.flag :=
\ ---------------------------------------- -
" WP1" copy.type ":=
wbp.scale.4
app.wks.col
\ ---------------------------------------- -
" WP2" copy.type ":=
wbp.scale.6.70
app.wks.col
\ ---------------------------------------- -
" WP3" copy.type ":=
wbp.scale.6.100
app.wks.col
\ -----------------------------------------------------------------
0 wp.flag :=
```

```
\ ---------------------------------------- -
" WB1" copy.type ":=
wbp.scale.4
app.wks.col
\ ---------------------------------------- -
" WB2" copy.type ":=
wbp.scale.6.70
app.wks.col
\ ---------------------------------------- -
" WB3" copy.type ":=
wbp.scale.6.100
app.wks.col
\ =================================================================
cdanow
" WP1" tmp.s.u "cat " .wks" "cat
" WP" tmp.s.u "cat " 1.wks" "cat del.and.rento
" WP2" tmp.s.u "cat " .wks" "cat
" WP" tmp.s.u "cat " 2.wks" "cat del.and.rento
" WP3" tmp.s.u "cat " .wks" "cat
" WP" tmp.s.u "cat " 3.wks" "cat del.and.rento

" WB1" tmp.s.u "cat " .wks" "cat
" WB" tmp.s.u "cat " 1.wks" "cat del.and.rento
" WB2" tmp.s.u "cat " .wks" "cat
" WB" tmp.s.u "cat " 2.wks" "cat del.and.rento
" WB3" tmp.s.u "cat " .wks" "cat
" WB" tmp.s.u "cat " 3.wks" "cat del.and.rento
```

R_Pha.shl

勇伯程式庫 A01496-20070125-A047 ❖ January 25, 2007 ❖ [Page count: 4]

```
: R_Pha.shl ; \ Wavelet $m_{0}$ phase \ -*- txt:asy -*-

\ ================================================================
fa sd
1024 n.wt1 :=
cdtNow load LLet-000.asy \

\ ================================================================
\ ----------------------------------------------------------------
: ini.par.chirp.al
  " Chirp-al.dat" now.file ":=
  512  n.wt1    :=
  200  s.freq.keep :=          \ Sampling freq -- original rate
  1    i.inc    :=             \ array index Inc for sub-sampling
  0    abs.flag    :=          \ absolute value norm
\   0.975 norm.keep.p.best :=    \ percentage of norm energy kept
\   0    filter.flag :=          \ for initial filtering
\   60   p.b.e :=                \ Pass band end (Hz)
\   30   t.w    :=               \ Transition width (Hz)
\   0    best.flag    :=         \ unclear, possibly not necessary (WPM)
\   100  #.div :=
;
\ \ ----------------------------------------------------------------
\ : ini.par.f
\   " f0w6020.2" now.file ":=
\   512  n.wt1    :=
\   40   s.freq.keep :=          \ Sampling freq -- original rate
\   1    i.inc    :=             \ array index Inc for sub-sampling
\   0    abs.flag    :=          \ absolute value norm
\   0    filter.flag :=          \ percentage of norm energy kept
\   0.975 norm.keep.p.best :=    \ for initial filtering
\   6.0  p.b.e :=                \ Pass band end (Hz)
\   6.0  t.w    :=               \ Transition width (Hz)
\   0    best.flag    :=         \ unclear, possibly not necessary (WPM)
\   100  #.div :=                \ foe entropy calculation
\ ;
: ini.par.b
  " b0w6020.4" now.file ":=
  1024 n.wt1    :=
  200  s.freq.keep :=          \ Sampling freq -- original rate
  4    i.inc    :=             \ array index Inc for sub-sampling
  0    abs.flag    :=          \ absolute value norm
;
\ ----------------------------------------------------------------
-1   isign := \ ." <- or -> :"
" P" act.flag.str ":= \ 3    act.flag :=
ini.par.b
\ ini.par.chirp.al

\ ================================================================
set.act cr
set.let
```

```
set.123
lee
\ \ set.let and set.123 are Moved into word within set.act
\ act.all

: ColRowIni
  1  gp.flag  :=
  1  file.no  :=
  \ 1  plot.gp   :=
  plot.gp 1 +  plot.gp  :=
  \
  i.ele  2  jwp.level **  <=
  if
     " PHI-"  PhiPsi.str ":="
  else
     " PSI-"  PhiPsi.str ":="
  then
;
: app.ramp.col
  wlet tmp.s.u ":="
  tmp.s.u "upper "drop
  " $$[("  file.no 0 n>s "cat " ," "cat
  123.ary.n []size swap drop 0 n>s "cat
  \ " ) " "cat tmp.s.u "cat " -" "cat " <" "cat
  " ) " "cat " <" "cat
  \ ---------------------------------------- ■
  " m_0" "cat
  \ ---------------------------------------- ■
  " >]" "cat
  123.ary.s "[ 1 ] ":=
  \
  123.ary.n []size swap drop real ramp 1.0 * becomes> 123.ary.n
  \
  to.123
  file.no 1 + file.no := \ post inc of column
;

\ ---------------------------------------- ■
\ 0 plot.gp  :=
\ ColRowIni  " psi-LMS.wks"   " 000.wks"  set.123.BP   act.all.1 \ LMS
\ ColRowIni  " psi-ONA.wks"   " 000.wks"  set.123.BP   act.all.2 \ ONA
\ ColRowIni  " psi-ONS.wks"   " 000.wks"  set.123.BP   act.all.3 \ ONS
\ ColRowIni  " psi-ONC.wks"   " 000.wks"  set.123.BP   act.all.4 \ ONC
\ ColRowIni  " psi-BO1.wks"   " 000.wks"  set.123.BP   act.all.5 \ BO1
\ ColRowIni  " psi-BO2.wks"   " 000.wks"  set.123.BP   act.all.6 \ BO2
\ ColRowIni  " psi-BO3.wks"   " 000.wks"  set.123.BP   act.all.7 \ BO3
\ -----------------------------------------------------------------
ACT.Gp.Ini
ColRowIni " Pha-" " LMS.wks"  "cat " 0.0" set.123.BP act.Gp.LMS  app.ramp.col
ColRowIni " Pha-" " ONC.wks"  "cat " 0.0" set.123.BP act.Gp.onC  app.ramp.col
ColRowIni " Pha-" " ONA1.wks" "cat " 0.0" set.123.BP act.Gp.onA1 app.ramp.col
ColRowIni " Pha-" " ONA2.wks" "cat " 0.0" set.123.BP act.Gp.onA2 app.ramp.col
ColRowIni " Pha-" " ONS1.wks" "cat " 0.0" set.123.BP act.Gp.onS1 app.ramp.col
ColRowIni " Pha-" " ONS2.wks" "cat " 0.0" set.123.BP act.Gp.onS2 app.ramp.col
ColRowIni " Pha-" " BO10.wks" "cat " 0.0" set.123.BP act.Gp.bo10 app.ramp.col
```

```
ColRowIni " Pha-" " BO1D.wks" "cat " 0.0" set.123.BP act.Gp.bo1D app.ramp.col
ColRowIni " Pha-" " BO20.wks" "cat " 0.0" set.123.BP act.Gp.bo20 app.ramp.col
ColRowIni " Pha-" " BO2D.wks" "cat " 0.0" set.123.BP act.Gp.bo2D app.ramp.col
ColRowIni " Pha-" " BO30.wks" "cat " 0.0" set.123.BP act.Gp.bo30 app.ramp.col
ColRowIni " Pha-" " BO3D.wks" "cat " 0.0" set.123.BP act.Gp.bo3D app.ramp.col

\ \ : Plt.shl
\ \   cdtNow load Plt_Pha.shl \
\ \ ;
\ \ \ Plt.shl \ \ load execution depth exceeded
\
\ \ =================================================================
\ \ -----------------------------------------------------------------
\ \ =================================================================
\ \ PhiPsi.str tmp.s ":=
\ fa
\ \ 4 string PhiPsi.str
\ \ tmp.s PhiPsi.str ":=
\ \ =================================================================
\ cdt load FigA-Ini.asy \
\ 2.0 ps.linewidth.now :=
\ \ =================================================================
\ : file.name
\   stack.data " Pha-LMS.WKS"
\   stack.load " Pha-LMS.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\
\ : file.name
\   stack.data " Pha-ONC.WKS"
\   stack.load " Pha-ONC.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\
\ : file.name
\   stack.data " Pha-ONA1.WKS"
\   stack.load " Pha-ONA1.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\
\ : file.name
\   stack.data " Pha-ONA2.WKS"
\   stack.load " Pha-ONA2.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\
\ : file.name
\   stack.data " Pha-ONS1.WKS"
\   stack.load " Pha-ONS1.BPF"
\ ;
\ cdt load FigACore.asy \
```

```
\ forget file.name
\
\ : file.name
\   stack.data " Pha-0NS2.WKS"
\   stack.load " Pha-0NS2.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\
\ : file.name
\   stack.data " Pha-B010.WKS"
\   stack.load " Pha-B010.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\
\ : file.name
\   stack.data " Pha-B01D.WKS"
\   stack.load " Pha-B01D.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\
\ : file.name
\   stack.data " Pha-B020.WKS"
\   stack.load " Pha-B020.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\
\ : file.name
\   stack.data " Pha-B02D.WKS"
\   stack.load " Pha-B02D.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\
\ : file.name
\   stack.data " Pha-B030.WKS"
\   stack.load " Pha-B030.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\
\ : file.name
\   stack.data " Pha-B03D.WKS"
\   stack.load " Pha-B03D.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name
\ \ ================================================================
```

R_PsiC.shl

勇伯程式庫 A01497-20070125-A048 ❖ January 25, 2007 ❖ [Page count: 2]

```
: R-PsiC.shl ; \ Plot Psi Curves \ -*- txt:asy -*-
\ : PsiC-Run.asy ;

fa sd \ swd

\ %=================================================================
\ ASY: Psi (Morlet or Variant) Curve data WKS [Friday, 00/09/29 at 14:39:04]
\  %-------------------------------------------------------------
\  %---------------------------------------- Generate psi function data
\  Use shell file psiC-run.asy to Generate psi function data wks file
\     Steps of implementation
\     0.  edit psiCwks.asy
\     1.  load LLet-0.asy
\     2.  load LLet-cwt.asy
\     3.  set.123
\         \   create wks file if not exist
\     4.  set.act
\         \   5
\         \   1024
\         \   start frequency:  0.6
\         \   # of frequency:1
\     5.  edit and load psiCwks.asy
\         \   to generate wks file
\         \   and then prepare, edit, or combine wks files for use in plotting
\         \   (see psiCplot.asy)
\ %=================================================================

\  %---------------------------------------- Generate psi function data wks

: new.pause
  my.pause.flag  " y"   "=
  if
    bell.1sl
    previous.window
    " <Press Key>"  s.type
    pckey  drop ?drop
    ."            "
    previous.window
  then
;

\ cdtNow load WD.asy
cdtNow load LLet-000.asy \
\ cdtNow load LLet-Rdg.asy \
cdtNow load LLet-WVT.asy \

\ =================================================================
: ini.par
  " b0w6020.4" now.file ":=
  1024 n.wt1    :=
  200  s.freq.keep :=
  5    i.inc    :=
```

```
  0.6  fre.beg :=
  1    #.fre   :=
  0.1  fre.inc :=
  \ 10 5 fix.format
;
\ ==================================================================
1    erf.flag :=
11.0 sai.beg :=
5.00 sai.end :=
256  mat.pts :=
" N" psi.flag ":=
" C" act.flag.str ":= \ 5    act.flag :=
\ ----------------------------------------
ini.par
\ ==================================================================

set.123
set.act
\ cdtNow load PsiC-Wks.asy
cdtNow load LLet-Psi.asy \
```

S_Ent.shl

勇伯程式庫 A01498-20070125-A049 ❖ January 25, 2007 ❖ [Page count: 2]

```
: S_Ent.shl ; \ :: R-Ent.sel ; \ %----- Asyst Prog

fa sd swd

\ cdtNow load WD.asy \
cdtNow load WDtmp.asy \ \ 星期五, 2007/01/12 at 10:39:41
                      \ \ g:\exp-asc\WT\  ->  g:\all-asy\Coh\
1024 n.wt1 :=
cdtNow load LLet-000.asy \
    \ ^^ Set proper length of fft.pts

\ =================================================================
\ -----------------------------------------------------------------
: ini.par.chirp.al
  " Chirp-al.dat" now.file ":=
  512  n.wt1    :=
  200  s.freq.keep :=         \ Sampling freq -- original rate
  1    i.inc    :=            \ array index Inc for sub-sampling
  0    abs.flag   :=          \ absolute value norm
  0.975 norm.keep.p.best :=   \ percentage of norm energy kept
  0    filter.flag :=         \ for initial filtering
  60   p.b.e :=               \ Pass band end (Hz)
  30   t.w   :=               \ Transition width (Hz)
  0    best.flag    :=        \ unclear, possibly not necessary (WPM)
  100  #.div :=
;
\ -----------------------------------------------------------------
: ini.par.f
  " f0w6020.2" now.file ":=
  1024 n.wt1    :=
  40   s.freq.keep :=         \ Sampling freq -- original rate
  1    i.inc    :=            \ array index Inc for sub-sampling
  0    abs.flag   :=          \ absolute value norm
  0    filter.flag :=         \ percentage of norm energy kept
  0.975 norm.keep.p.best :=   \ for initial filtering
  6.0  p.b.e :=               \ Pass band end (Hz)
  6.0  t.w   :=               \ Transition width (Hz)
  0    best.flag    :=        \ unclear, possibly not necessary (WPM)
  100  #.div :=               \ foe entropy calculation
;
: ini.par.b
  " b0w6020.4" now.file ":=
  1024 n.wt1    :=
  200  s.freq.keep :=         \ Sampling freq -- original rate
  4    i.inc    :=            \ array index Inc for sub-sampling
  0    abs.flag   :=          \ absolute value norm
  0    filter.flag :=         \ percentage of norm energy kept
  0.975 norm.keep.p.best :=   \ for initial filtering
  6.0  p.b.e :=               \ Pass band end (Hz)
  6.0  t.w   :=               \ Transition width (Hz)
  0    best.flag    :=        \ unclear, possibly not necessary (WPM)
  100  #.div :=               \ foe entropy calculation
```

```
;
\ ================================================================
-1   isign := \ ." <- or -> :"
" E" act.flag.str ":= \ 3     act.flag :=
ini.par.b
\ ini.par.chirp.al
\ ================================================================
set.act
\ set.let and set.123 are Moved into word within set.act
act.all
```

....

S_Nor.shl

勇伯程式庫 A01499-20070125-A050 ❖ January 25, 2007 ❖ [Page count: 2]

```
: S_Nor.shl ; \ : R-Nor.sel ; \ %----- Asyst Prog

fa sd swd

\ cdtNow load WD.asy \
cdtNow load WDtmp.asy \ \ 星期五, 2007/01/12 at 10:39:41
                      \ \ g:\exp-asc\WT\  ->  g:\all-asy\Coh\
1024 n.wt1 :=
cdtNow load LLet-000.asy \
    \ ^^ Set proper length of fft.pts

\ =================================================================
\ -----------------------------------------------------------------
: ini.par.chirp.al
  " Chirp-al.dat" now.file ":=
  512  n.wt1    :=
  200  s.freq.keep :=        \ Sampling freq -- original rate
  1    i.inc    :=           \ array index Inc for sub-sampling
  0    abs.flag    :=        \ absolute value norm
  0.975 norm.keep.p.best :=  \ percentage of norm energy kept
  0    filter.flag :=        \ for initial filtering
  60   p.b.e :=              \ Pass band end (Hz)
  30   t.w   :=              \ Transition width (Hz)
  0    best.flag    :=       \ unclear, possibly not necessary (WPM)
  100  #.div :=
;
\ -----------------------------------------------------------------
: ini.par.f
  " f0w6020.2" now.file ":=
  1024 n.wt1    :=
  40   s.freq.keep :=        \ Sampling freq -- original rate
  1    i.inc    :=           \ array index Inc for sub-sampling
  0    abs.flag    :=        \ absolute value norm
  0    filter.flag :=        \ percentage of norm energy kept
  0.975 norm.keep.p.best :=  \ for initial filtering
  6.0  p.b.e :=              \ Pass band end (Hz)
  6.0  t.w   :=              \ Transition width (Hz)
  0    best.flag    :=       \ unclear, possibly not necessary (WPM)
  100  #.div :=              \ foe entropy calculation
;
: ini.par.b
  " b0w6020.4" now.file ":=
  1024 n.wt1    :=
  200  s.freq.keep :=        \ Sampling freq -- original rate
  4    i.inc    :=           \ array index Inc for sub-sampling
  0    abs.flag    :=        \ absolute value norm
  0    filter.flag :=        \ percentage of norm energy kept
  0.975 norm.keep.p.best :=  \ for initial filtering
  6.0  p.b.e :=              \ Pass band end (Hz)
  6.0  t.w   :=              \ Transition width (Hz)
  0    best.flag    :=       \ unclear, possibly not necessary (WPM)
  100  #.div :=              \ foe entropy calculation
```

```
;
\ -----------------------------------------------------------------
-1   isign := \ ." <- or -> :"
" N" act.flag.str ":= \ 3 act.flag :=
ini.par.b
\ ini.par.chirp.al

\ =================================================================
set.act
\ set.let and set.123 are Moved into word within set.act
act.all
```

S_WPM.shl

勇伯程式庫 A01500-20070125-A051 ❖ January 25, 2007 ❖ [Page count: 2]

```
: S_WPM.sel ; \ -*- txt:asy -*-

fa sd swd

: new.pause
  my.pause.flag  " y"   "=
  if
    bell.1sl
    previous.window
    " <Press Key>"  s.type
    pckey  drop ?drop
    ."           "
    previous.window
  then
;

\ cdtNow load WDtmp.asy \

512 n.wt1 :=
cdtNow load LLet-000.asy \
    \ ^^ Set proper length of fft.pts
\
\ ==================================================================
: ini.par
  " b0w6020.4" now.file ":=
  1024 n.wt1    :=
  200  s.freq.keep :=
  5    i.inc    :=
  1    best.flag    :=
;
\ ------------------------------------------------------------------
: ini.par.chirp.al
  " Chirp-al.dat" now.file ":=
  512  n.wt1    :=
  200  s.freq.keep :=
  1    i.inc    :=
  1    best.flag    :=
;
\ ==================================================================
" N" act.flag.str ":= \ 3    act.flag :=
\ ini.par
ini.par.chirp.al
\ ----------------------------------------
1        isign := \ ." <- or -> :"
1        wp.flag := \ WP(1) or WB(0)
" ON55C" wlet ":=
\ ==================================================================
set.act \ Chose 3
action
my.pause sd
get.map
```

S_WVT.shl

勇伯程式庫 A01501-20070125-A052 ❖ January 25, 2007 ❖ [Page count: 2]

```
: S_WVT.shl ; \ %----- Asyst Prog
\ : R-Rdg.sel ; \ %----- Asyst Prog
\ Shell File for Wavelet Variant Transforms and Adapted CWT

fa sd swd
512 n.wt1 :=
\ ==================================================================
: new.pause
  my.pause.flag  " y"   "=
  if
    bell.1sl
    previous.window
    " <Press Key>"  s.type
    pckey  drop ?drop
    ."         "
    previous.window
    \     bell.flag :=   \ restore
  then
;
\ ==================================================================
cdtNow load LLet-000.asy \
cdtNow load LLet-WVT.asy \
\ cdtNow load LLet-Rdg.asy \
\ ==================================================================
: ini.par
  " b0w6020.4" now.file ":=
  1024 n.wt1    :=
  200  s.freq.keep :=
  5    i.inc    :=
  1.6  fre.beg  :=
  40   #.fre    :=
  0.1  fre.inc  :=
;
: ini.par.f
  5    act.flag :=
  " f0w6020.2" now.file ":=
  1024 n.wt1    :=
  40   s.freq.keep :=
  1    i.inc    :=
  1.6  fre.beg  :=
  40   #.fre    :=
  0.1  fre.inc  :=
;
: ini.par.chirp
  5    act.flag :=
  " Chirp-al.dat" now.file ":=
  512  n.wt1    :=
  200  s.freq.keep :=
  1    i.inc    :=
  5    fre.beg  :=
  20   #.fre    :=
  5    fre.inc  :=
```

```
;
\ ============================================================
\ ------------------------------------------------------------
" C" act.flag.str ":= \ 5    act.flag :=
1    erf.flag :=
11.0 sai.beg :=
5.00 sai.end :=
256  mat.pts :=
\ " N" psi.flag ":=
" O" psi.flag ":=
10 5 fix.format
\ ----------------------------------------
\ ini.par
\ ini.par.f
ini.par.chirp
\ ============================================================
set.act \ Choose 5
sd echo.off
: choose.e.m
  cr
  ." Choose data type to send to MMA (E: Hilbert envelop; M: Modulus): "
  " E" "dup get.str "=
  if
    to.mat.e.p
  else
    to.mat.m.p
  then
;
choose.e.m
```

S_Coh_WC.Shl

勇伯程式庫 A01502-20070125-A053 ❖ January 25, 2007 ❖ [Page count: 3]

```
: S_Coh_WC.Shl ; \ : WC_S.sel ; \ -*- txt:asy -*-
\ \ 星期五, 2004/09/10 at 14:47:02
\ \  123-A2W.asy  -> 123-A2Wn.asy
\ ----------------------------------------------------------------

fa sd \ swd

integer scalar #.fft.pts scalar #.pts
2 string SC.or.WC

" WC" SC.or.WC ":=
: spe.coh
  \ dummy; used in Coh_BP.bp
;
512 n.wt1 :=
\ ================================================================
: new.pause
  my.pause.flag  " y"   "=
  if
    bell.1sl
    previous.window
    " <Press Key>"  s.type
    pckey  drop ?drop
    ."            "
    previous.window
  then
;

\  %---------------------------------------- Generate Coherence Data for Plots
\  Use the shell file "WC-run.asy" to run wavelet coherence:
\     The steps to implement
\     0.  edit  WC-BP.bp              Define the coherence pairs (get.bat)
\     1.  load  LLet-0.asy
\     2.  load  LLet-cwt.asy
\     3.  Load  WC-0.asy
\     4.  set.act                     Initialize a few parameters
\         % 5.  % set.123                     Preparing wks files
\         % 6.  % set.coh                     Initializing some parameters
\         % 7.  % run.coh   OK!

\ ================================================================
cdtNow load LLet-000.asy \   \  星期五, 2007/01/12 at 10:03:51
\ cdtNow load LLet-00n.asy \ \  星期五, 2004/09/10 at 14:03:02
\                            \  123-A2W.asy   -> 123-A2Wn.asy
\                            \  Replace All [dir.lee.asc] to [cda.now.str]
\                            \  dir.lee.asc   -> cda.now.dir
\                            \  g:\Exp-Asc\WT -> g:\Exp-Asc\Coh
cdtNow load LLet-WVT.asy \   \  星期五, 2007/01/12 at 10:03:51
\ cdtNow load LLet-Rdg.asy \ \  星期五, 2007/01/12 at 10:03:51
cdtNow load LLet-Coh.asy \
\ cdtNow load LLet-WC.asy \
\ \ cdtNow load LLet-WCo.asy \ LLet-WCo-200010161021.asy & WC-BP.bp
```

```
\ ==================================================================

: ini.par.a.b.p.q.i5
  \ " b0w6020.4" now.file ":=
  1024 n.wt1    :=
  200  s.freq.keep :=
  5    i.inc    :=
  0.1  fre.beg  :=
  90   #.fre    :=
  0.1  fre.inc  :=
  256  mat.pts  :=
  \ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
  \ " 0-WC-A.wks" cda.now.str "swap "cat 123.file.name ":=
  \ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
;
: ini.par.f
  \ " f0w6020.2" now.file ":=
  1024 n.wt1    :=
  4096 n.wt1    :=
  40   s.freq.keep :=
  1    i.inc    :=
  0.1  fre.beg  :=
  100  #.fre    :=
  0.1  fre.inc  :=
  256  mat.pts  :=
  \ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
\   1    fre.beg  :=
\   10   #.fre    :=
\   1    fre.inc  :=
  \ " 0-WC-FR.wks" cda.now.str "swap "cat 123.file.name ":=
  \ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
;
\ : ini.par
\   now.file 1 "left "upper " F" "=
\   if
\     ini.par.f
\   else
\     ini.par.a.b.p.q.i5
\   then
\   256  mat.pts  :=
\ ;
\ -----------------------------------------------------------------
: Ini.input.data.type
\   cr ." Batch Processing Data Type" " Coh_BP.bp ::" s.type.2
\   cr ."                                               F: F series"
\   cr ."                                               B: A(P), B(Q), ... series"
\   cr ."                                               " input.data.type.flag get.str
\   "dup s.type
\   "upper input.data.type.flag ":=
  input.data.type.flag " F" "=
  if
    ini.par.f
  else
    ini.par.a.b.p.q.i5
  then
```

```
  \    256   mat.pts  :=
;
\ -----------------------------------------------------------------
\ =================================================================
" N" psi.flag  ":=
" W" act.flag.str  ":= \ 6     act.flag  :=
" C" rea.or.com  ":=
1    erf.flag  :=
11.0 sai.beg  :=
5.00 sai.end  :=
\ ----------------------------------------- ■
Ini.input.data.type
specify.now.file \ Redefine ini data
\ ini.par
\     1024 n.wt1    :=
\ =================================================================
set.act
```

SWD.asy

勇伯程式庫 A01504-20070125-A055 ❖ January 25, 2007 ❖ [Page count: 4]

```
: SWD.asy ; \ %----- Asyst Prog -*- txt:asy -*-

echo.off
\ fa
\ integer scalar cdtnow.flag
\   1 string cdtnow.flag.str
\ 32 string cdt.now.str
\ 32 string cda.now.str
\ 32 string cdd.now.str
32 string cdt.str
32 string cda.str
32 string cdd.str
32 string cdf.str

0 set.file.parse
\ ================================================================
: cd.common \ [ Txt Asc Fig Dat - ]
  cdd.str ":=
  cdd.now.str ":=
  cdf.str ":=
  cdf.now.str ":=
  cda.str ":=
  cda.now.str ":=
  cdt.str ":=
  cdt.now.str ":=
\   " c:\per\asy\dat\" cdd.now.str ":=
\   " c:\per\asy\fig\" cdf.now.str ":=
\   " c:\per\asy\asc\" cda.now.str ":=
\   " c:\per\asy\txt\" cdt.now.str ":=
  \ ----------------------------------------
  cda.str dir.lee.asc ":=
  cdd.str dir.lee.dat ":=
\   cdf.now.str dir.lee.fig ":=
;
\ ================================================================
: cdOri
  " c:\per\asy\txt\"
  " c:\per\asy\txt\"
  " c:\per\asy\asc\"
  " c:\per\asy\asc\"
  " G:\Fig\Asy\Fig\"
  " G:\Fig\Asy\Fig\"
  " c:\per\asy\dat\"
  " c:\per\asy\dat\"
  cd.common \ [ Txt Asc Fig Dat - ]
;
: cdSta
  " c:\per\asy\txt\"
  " c:\per\asy\txt\"
  " G:\Exp-Asc\tmp\"
  " G:\Exp-Asc\tmp\"
  " G:\Fig\Asy\Fig\"
```

```
  " G:\Fig\Asy\Fig\"
  " G:\Exp-Dat\WC\"
  " G:\Exp-Dat\WC\"
  cd.common
;
: cdWT
  " G:\All-Asy\WT\"
  " G:\All-Asy\WT\"
  " G:\Exp-Asc\WT\"
  " G:\Exp-Asc\WT\"
\   " G:\Fig\Asy\Fig\"
\   " G:\Fig\Asy\Fig\"
   " G:\Fig\Asy\WT\"
   " G:\Fig\Asy\WT\"
  " G:\Exp-Dat\WC\"
  " G:\Exp-Dat\WC\"
  cd.common
;
: cdDnR
  " G:\All-Asy\WDRPB\"
  " G:\All-Asy\WDRPB\"
  " G:\Exp-Asc\WDRPB\"
  " G:\Exp-Asc\WDRPB\"
  " G:\Fig\Asy\WDRPB\"
  " G:\Fig\Asy\WDRPB\"
  " G:\Exp-Dat\WC\"
  " G:\Exp-Dat\WC\"
  cd.common
;
\ : cdCoh
\   " G:\All-Asy\WC\"
\   " G:\All-Asy\WC\"
\   " G:\Exp-Asc\WC\"
\   " G:\Exp-Asc\WC\"
\   " G:\Fig\Asy\Fig\"
\   " G:\Fig\Asy\Fig\"
\   " G:\Exp-Dat\WC\"
\   " G:\Exp-Dat\WC\"
\   cd.common
\ ;
: cdCoh
  " G:\All-Asy\Coh\" \ \ cdt.now.str ":=
  " G:\All-Asy\Coh\" \ \ cdt.str ":=
  " G:\Exp-Asc\Coh\"  \ \ cda.now.str ":=
  " G:\Exp-Asc\WT\"  \ \ cda.str ":=
  " G:\Fig\Asy\Coh\" \ \ cdf.now.str ":=
  " G:\Fig\Asy\Coh\" \ \ cdf.str ":=
  " G:\Exp-Dat\WC\"  \ \ cdd.now.str ":=
  " G:\Exp-Dat\WC\"  \ \ cdd.str ":=
  cd.common
;
: cdHT
  " c:\per\asy\txt\"
  " c:\per\asy\txt\"
  " G:\Exp-Asc\WC\"
```

```
  " G:\Exp-Asc\WC\"
  " G:\Fig\Asy\Fig\"
  " G:\Fig\Asy\Fig\"
  " G:\Exp-Dat\WC\"
  " G:\Exp-Dat\WC\"
  cd.common
;
: cdMod
  " G:\All-Asy\Mod\"
  " C:\per\asy\txt\"
\  " G:\Exp-Asc\mod\"
  " G:\Exp-Asc\Asc\" \ cda.now - output
  " G:\Exp-Asc\WC\"  \ cda - input
  " G:\Fig\Asy\Fig\"
  " G:\Fig\Asy\Fig\"
  " G:\Exp-Dat\WC\"
  " G:\Exp-Dat\WC\"
  cd.common
;
: cdTmp
  " G:\All-Asy\tmp\"
  " G:\All-Asy\tmp\"
  " G:\Exp-Asc\tmp\"
  " G:\Exp-Asc\tmp\"
  " G:\Fig\Asy\Fig\"
  " G:\Fig\Asy\Fig\"
  " G:\Exp-Dat\WC\"
  " G:\Exp-Dat\WC\"
  cd.common
;
: cdPrj
  " G:\All-Asy\prj\"
  " G:\All-Asy\prj\"
  " G:\Exp-Asc\prj\"
  " G:\Exp-Asc\prj\"
  " G:\Fig\Asy\Fig\"
  " G:\Fig\Asy\Fig\"
  " G:\Exp-Dat\WC\"
  " G:\Exp-Dat\WC\"
  cd.common
;
: cdEnt
  " c:\per\asy\txt\"
  " c:\per\asy\txt\"
  " G:\Exp-Asc\WC\"
  " G:\Exp-Asc\WC\"
  " G:\Fig\Asy\Fig\"
  " G:\Fig\Asy\Fig\"
  " G:\Exp-Dat\WC\"
  " G:\Exp-Dat\WC\"
  cd.common
;
\ ================================================================
: Choose.dir.WP.str
  cdtnow.flag.str "upper
```

```
  "dup " O" "= if cdOri else cdPrj then
  "dup " S" "= if cdSta then
  "dup " W" "= if cdWT  then
  "dup " D" "= if cdDnR then
  "dup " C" "= if cdCoh then
  "dup " E" "= if cdEnt then
  "dup " H" "= if cdHT  then
  "dup " M" "= if cdMod then
  "dup " T" "= if cdTmp then
       " J" "= if cdPrj then
;
: Choose.dir.str
  cr ." Set Working Directories:"
  cr ."             "    " O: Ori (txt)" s.type
  cr ."             "    " S: Statistics (Sta)" s.type
  cr ."             "    " W: Wavelet Transform ()" s.type
  cr ."             "
  " D: Decomposition, Reconstruction, Phase , and Blowup (WDRPB)" s.type
  cr ."             "    " C: Coherence (Wavelet and Fourier Coherence) (WC)" s.type
  cr ."             "    " E: Entropy" s.type
  cr ."             "    " H: Hilbert Transform (HT)" s.type
  cr ."             "    " M: Modualtion (Mod)" s.type
  cr ."             "    " T: Temporary (Tmp)" s.type
  cr ."             "    " J: Project ()" s.type
  cr ."    Set to?"
  cdtnow.flag.str get.str cdtnow.flag.str ":=
  Choose.dir.WP.str
;
: show.dir
  cr ." txt     dir:" dir.lee.txt "len 1 - "left s.type
  cr ." cdtNow dir:" cdt.now.str "len 1 - "left s.type.3
  cr ." cdt     dir:" cdt.str     "len 1 - "left s.type
  cr
  cr ." cdaNow dir:" cda.now.str "len 1 - "left s.type.3
  cr ." cda     dir:" cda.str     "len 1 - "left s.type
  cr
  cr ." cdfNow dir:" cdf.now.str "len 1 - "left s.type.3
  cr ." cdf     dir:" cdf.str     "len 1 - "left s.type
  cr
  cr ." cddNow dir:" cdd.now.str "len 1 - "left s.type.3
  cr ." cdd     dir:" cdd.str     "len 1 - "left s.type
;
" M" cdtnow.flag.str ":=
" D" cdtnow.flag.str ":=
" W" cdtnow.flag.str ":=

" C" cdtnow.flag.str ":=
sc
Choose.dir.str
show.dir  cdtNow
```

LLet-000.asy

勇伯程式庫 A01505-20070125-A056 ❖ January 25, 2007 ❖ [Page count: 16]

```
: LLet-000.asy ; \    -*- txt:asy -*-
\ 1.  The Core file for all Wavelet programs.
\ 2.
\
\ ----------------------------------------------------------------
\ {29.line}
\ dir.lee.txt        " nr-dwt.txt" "cat  defer>  load
echo.on

real     scalar s.freq
integer scalar n.fft.pts  scalar i.inc
26  0   29  79 window {hp.wnd}
nd
: iand \ [ n1 , n2  --  n3 ]
  #>mask   #>mask   and    mask>#
;

cdtNow load LLet-Coe.asy \
cdt load 123-A2W.asy \
cdtNow load LLet-Mai.asy \

: dp.real.flag \ To see the difference,
  real         \ keep this
  \ dp.real    \ you can exercise  a []sum for e{17}=1
;
integer scalar i.ini
        scalar i.int
        scalar i.end
        scalar i.no
        scalar up.ax.ini
        scalar up.ax.int
        scalar up.ax.end
        scalar up.ax.no
        scalar up.axmin
        scalar up.axmax
        scalar dp.ax.ini
        scalar dp.ax.int
        scalar dp.ax.end
        scalar dp.ax.no
        scalar dp.axmin
        scalar dp.axmax
        scalar ax.shf
        scalar dual.flag
        scalar i.chirp
        scalar i.envelop
integer scalar i.cou  scalar n.blow
3  string file.ext
1  string yadp.flag
64 string tmp.s.u

real     scalar t.dur
         scalar up.wxmin
```

```
        scalar up.wxmax
        scalar up.wymin
        scalar up.wymax
        scalar dp.wxmin
        scalar dp.wxmax
        scalar dp.wymin
        scalar dp.wymax
\ ==============================================================
integer scalar p.pnts
: get.p.pnts
  n.wt1  my.power
  n.power  10  =  if  1000  then
  n.power   9  =  if   500  then
  n.power   8  =  if   250  then
  n.power   7  =  if   120  then
  n.power   6  =  if    80  then
  p.pnts :=
;
\ ==============================================================
integer scalar chirp.spe
: initial.parameter
  " bo370"  wlet  ":=
  " on88s"  wlet  ":=
  " bo131"  wlet  ":=
  " on22a"  wlet  ":=
  " bo220"  wlet  ":=
  " bo22d"  wlet  ":=

  4     n.dau.c   :=
  0     wp.flag   :=
  1     isign     :=
  2     jwp.level :=

  \  1024   n.wt1     :=
  \  9   2  swap **    n.wt1     :=
  \ cr ." Wavelet length :" n.wt1 get.val n.wt1 :=
  cr ." Initialize Wavelet length (match when [DnR, Phase, Norm-WP, Ent]):"
  n.wt1 get.val n.wt1 :=

  0     i.int     :=

  n.wt1  my.power

  1   isign   =
  if    \ ------------------------ \ i : freq ;
    cr
    " 0:General;  1:Envelop;  2:Chirp ::  "
    s.type   0     get.val
    case  0  of                  endof
          1  of  1 i.envelop :=  endof
          2  of  1 i.chirp   :=
                 cr ." Chirp.spe:(11 Nyquist/Linear, 12, 21, 22)"
                 11 get.val chirp.spe :=
                                 endof
    endcase
```

```
    i.chirp  1  =
    if
      50      i.ini      :=   \ first executed;  small index and low line type
      1       i.no       :=
    else
      30      i.ini      :=   \ first executed;  small index and low line type
      1       i.no       :=
    then
    1         up.ax.ini  :=
    1         up.ax.int  :=
    1         up.axmin   :=
\     2  n.power **
     get.p.pnts  p.pnts
              up.axmax   :=
    1         dp.ax.ini  :=
    1         dp.ax.int  :=
    1         dp.axmin   :=
    200       dp.axmax   :=
  \ i.ini   i.int  i.no 1 -    *  +      i.end :=
  \ i.end  1 +                           i.end :=
  else  \ ------------------------ \ i : unit base function ;
        \ 0   2   4   8   16   32   64   128   512   1024   2048
    6         i.ini      :=   \ first executed;  small index and low line type
    5         i.end      :=   \ last executed;  small index and high line type
    1         i.no       :=
    1         up.ax.ini  :=
    1         up.ax.int  :=
    1         up.axmin   :=
\     500       up.axmax   :=
    1         dp.ax.ini  :=
    1         dp.ax.int  :=
    1         dp.axmin   :=
    \ n.wt1   dp.axmax   :=
\     n.wt1     2048   =   if   2000    then
\     n.wt1     1024   =   if   1000    then
\     n.wt1      512   =   if    500    then
\     n.wt1      256   =   if    250    then
\     n.wt1      128   =   if    120    then
\     n.wt1       64   =   if     80    then
    2  n.power **
    dup       up.axmax   :=
              dp.axmax   :=
  \ i.end   i.int  i.no 1 -    *  -      i.ini :=
  then
  1     ax.shf :=
;
initial.parameter
\ =================================================================
: parameter.flag ; \ dummy
: parameter.change ;
: options.change
  14 foreground
  12 foreground
   37 1  goto.xy  ."      i.ini/end=  " \ 54 1 goto.xy ."   "
```

```
   37 2  goto.xy  ."      i.interv =  " \ 54 2 goto.xy ."  "
   37 3  goto.xy  ."      i.no     =  " \ 54 3 goto.xy ."  "
\  37 4  goto.xy  ."               =  " \ 54 3 goto.xy ."  "
\  37 5  goto.xy  ."       dig.mid = "  \ 54 3 goto.xy ."  "

   52 1  goto.xy    i.ini     get.val  i.ini      :=
   52 2  goto.xy    i.int     get.val  i.int      :=
   52 3  goto.xy    i.no      get.val  i.no       :=
   52 4  goto.xy    UP.aXmax  get.val  UP.aXmax   :=
   52 5  goto.xy    DP.aXmax  \ 5  0     fix.format
                              get.val  DP.aXmax   :=

\  7  7  goto.xy  ."           Reset options again : "   " <F>"  int.text
   bell.2sl
\  pckey  drop if myself then
;
0 tmp.v.i  :=
: parameter.change
  14 foreground
  12 foreground
   7  0  goto.xy  ."                             "
   7  1  goto.xy  ."     Wlet      =  " \ 24 1 goto.xy ."  "
   7  2  goto.xy  ."     n.wt1     =  " \ 24 2 goto.xy ."  "
   7  3  goto.xy  ."     -> <-     = "  \ 24 3 goto.xy ."  "
   7  4  goto.xy  ."     chirp     = "  \ 24 4 goto.xy ."  "
\  7  5  goto.xy  ."       OPTIONS = "  \ 24 5 goto.xy ."  "

   22 1  goto.xy    wlet       get.str  wlet       ":=
   22 2  goto.xy    n.wt1      get.val  n.wt1       :=
   22 3  goto.xy    isign      get.val  isign       :=
   22 4  goto.xy    i.chirp    get.val  i.chirp     :=
   22 5  goto.xy

  tmp.v.i 999 <>
  if
    999    tmp.v.i   :=
    isign  1  =
    if
      30     i.ini     :=
      1      i.no      :=
    else
      4      i.ini     :=
      5      i.no      :=
      1000      dp.axmax  :=
    then
    i.chirp  1  =
    if
      50     i.ini     :=   \ first executed;  small index and low line type
      1      i.no      :=
    then
  then
   37 1  goto.xy  ."     i.ini/end = "   i.ini    n.type
   37 2  goto.xy  ."     i.interv  = "   i.int    .
   37 3  goto.xy  ."     i.no      = "   i.no     .
```

```
\ 22 5  goto.xy     " OK?"      s.type
\ bell.2sl
\ pckey  drop
\ if
    options.change
\ then
  6  foreground
  7  7  goto.xy   ."              Reset whole again : "    " <F>"  int.text
  bell.2sl
  my.def.colors
  pckey  drop if myself then
;
: parameter.flag.p1
\  nd
  1  1  =       ?drop     \  <-- pcx
  1  1  <>  \  ?drop     \  <-- text
  if
    graphics.display   up
    dir.lee.txt
    xeq.flag  " w"  "= if  " def-anaw.pcx" "cat then
    xeq.flag  " r"  "= if  " def-anar.pcx" "cat then
    xeq.flag  " d"  "= if  " def-anad.pcx" "cat then
    defer>  pcx>vup
\ \    {show.wnd.1}
    fg.color   foreground
    bg.color   background
    sc
  else
\    sd
    ."                    ----------------------------------------"   cr
    ."                    | DWT                                   |"   cr
    ."                    |                                       |"   cr
    ."                    |                                       |"   cr
    ."                    |                                       |"   cr
    ."                    ----------------------------------------"  \ cr cr
\    {show.wnd.1}
    bg.color   background
    fg.color   foreground
    sc  \  <-- be aware of the sequence !
  then

;
: parameter.flag
  \ 2 1       fix.format
  4 0       fix.format
  parameter.flag.p1
  \ {show.wnd.s} {border}
  sc
\ \  {show.wnd.s} {border}
  bg.color   background
  14         foreground
  sc

   7  1  goto.xy   ."    Wlet       = "    wlet       s.type
   7  2  goto.xy   ."    n.wt1      = "    n.wt1      .
```

```
   7  3  goto.xy  ."    -> <-      = "   isign     .
   7  4  goto.xy  ."    chirp      = "   i.chirp    .

   37 1  goto.xy  ."    i.ini/end  = "   i.ini    n.type
   37 2  goto.xy  ."    i.interv   = "   i.int      .
   37 3  goto.xy  ."    i.no       = "   i.no       .
   37 4  goto.xy  ."    UP.Xmax    = "   up.axmax .
\                                        5  0  fix.format
   37 5  goto.xy  ."    DP.Xmax    = "   dp.axmax .
\                                        2  1  fix.format

\  7  5  goto.xy  ."      OPTIONS = "  "  -->"    s.type
   6  foreground
   7  7  goto.xy  ."               Change DEFAULT  : "   " <F>"  int.text
   bell.2sl
   my.def.colors
  pckey  drop
  if
    parameter.change
  then
  bg.color   background
  fg.color   foreground
\  {29.line}
;
: parameter.reset
  isign  1  =
  if
    i.int  0  =
    if
      i.ini    my.power
      2        n.power  i.no  1 -  +  **     i.end :=
      i.end  1 +  1 +                        i.end :=
    else
      i.ini    i.int  i.no 1 -    *  +       i.end :=
      i.end  1 +                             i.end :=
    then
  else  \ ------------------------ \ i : unit base function ;
    i.ini   i.end :=
    i.int   0  =
    if
      i.end    my.power
      2        n.power  i.no 1 -  +  **   1 +      i.ini :=
    else
      i.end    i.int  i.no 1 -    *  -       i.ini :=
    then
  then
;
parameter.flag
parameter.reset
\ ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
\ dir.lee.txt        " my-wt.txt" "cat  defer>  load
\ ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
10.      t.dur      :=

token  a          exp.mem>  a
```

```
token  b          exp.mem>  b
token  b.p        exp.mem>  b.p
integer scalar i.run
: run
  1  i.run :=
  i.end   i.ini              \  do : ini  end  int
  do
    1   isign   =
    if    \ forward
      2.   Pi *   i  0.9 *  *
      n.wt1 dp.real.flag ramp *    n.wt1 /  sin
      2.   Pi *   i  1.0 *  *
      n.wt1 dp.real.flag ramp *    n.wt1 /  sin
      2.   Pi *   i  1.1 *  *
      n.wt1 dp.real.flag ramp *    n.wt1 /  sin
      +  +   becomes>  B

      i.envelop  0  =
      if
        2.   Pi *   i *  \ t.dur *        \  for sin wave (phase = 2 Pi f t)
        n.wt1 dp.real.flag ramp *    n.wt1 /      \
        sin     becomes>  B
      then

      n.wt1 1 +  ramp   becomes>  B.p
      b [ n.wt1 ]    b.p [ n.wt1 1 + ]  :=
      n.wt1  1 +    1
      do   b [ i ]   b.p [ i ] :=      loop

      i.chirp  1  =
      if
      \   2.   Pi *   i *   t.dur *                       \   For chirp
      \   n.wt1     dp.real.flag ramp  /                  \ decreasing

          \ theta = 2 pi k t^2                \ f=2 k t
          \ Nyquist or Aliasing
          chirp.spe  11 =
          chirp.spe  12 =  or
          if
            2.   Pi *   1. *                   \ i=1 nyqust    \   For chirp
          \ 2.   Pi *  0.02  *                 \ i=1 nyqust    \   For chirp
          else
            2.   Pi *   2. *                   \ i>1 aliasing  \   For chirp
          then

          \ Linear or Quadratic
          chirp.spe  11 =
          chirp.spe  21 =  or
          if
            4. /   n.wt1 float /               \ k linear    \  Nyqust f -> /(4n)
            n.wt1 dp.real.flag ramp dup *  *   \ t^2
          else
            6. /   n.wt1 float dup * /                 \ k quadratic
            n.wt1 dp.real.flag ramp dup dup * *  *     \ t^3
          then
```

```
        sin    becomes>  B
        2.   Pi *   i *   t.dur *
        n.wt1 1 + dp.real.flag ramp  /
        sin    becomes>  B.p
    then

  else    \ backward
    n.wt1  dp.real.flag  ramp  becomes>  B
    n.wt1 1 +  dp.real.flag  ramp  becomes>  B.p
    0.d    b         :=
    1.d    b [ i ]   :=
    0.d    b.p [ up.axmax 1 + ] :=
    up.axmax  1 +  up.axmin
    do
      b [ i ]     b.p [ i ] :=
    loop
  then
  alg.flag  1 =
  if
\      nd
    " c:\per\asy\asc\algorith.dat"  defer>  delete
    " c:\per\asy\asc\algorith.dat"  defer>  out>file
  then
  i ."       "  n.type
  "time

\     isign    \    get.h&g.dau
  assign.wlet
  get.h&g

  0  domain.flag  :=
  b   isign  1  =  if    forward.wt   else   inverse.wt   then
    becomes> a
  \ b   n.wt1  isign  wlet      wt      \ NR
  alg.flag  1 =
  if
    out>file.close   gd
  then
  i   i.ini  =
  if
    up
    horizontal
      axis.fit.off
    vertical
      axis.fit.on
    b.p  sub[ up.axmin , up.axmax ax.shf + ]
    []size  ramp  ax.shf - swap  xy.data.fit  drop  drop
    \  b  []size  ramp  swap  xy.data.fit   drop  drop
    up.axmax
    dup             up.ax.end :=
    up.ax.int /     up.ax.no  :=
    \ horizontal    0   up.ax.end      world.set  \ reset up wxmin&max
    wxmin     up.wxmin :=
    wymin     up.wymin :=
```

```
    wxmax     up.wxmax :=
    wymax     up.wymax :=
    xy.axis.plot
    n.wt1  ramp  sub[ up.ax.ini , up.ax.no , up.ax.int ]
    b            sub[ up.ax.ini , up.ax.no , up.ax.int ]   i.run      xydp
    dp
    horizontal
      axis.fit.on
    vertical
      axis.fit.on
    \ a sub[ dp.axmin , dp.axmax ax.shf + ]
    \ []size  ramp  ax.shf - swap  xy.data.fit  drop drop
    a sub[ dp.axmin , dp.axmax          ]
    []size  ramp             swap  xy.data.fit  drop drop
    dp.axmax
    dup        dp.ax.end :=
    dp.ax.int /   dp.ax.no  :=
    horizontal    0   dp.ax.end      world.set  \ reset dp wxmin&max
    wxmin     dp.wxmin :=
    wymin     dp.wymin :=
    wxmax     dp.wxmax :=
    wymax     dp.wymax :=
    xy.axis.plot
    n.wt1  ramp  sub[ dp.ax.ini , dp.ax.no , dp.ax.int ]
    a            sub[ dp.ax.ini , dp.ax.no , dp.ax.int ]   i.run      xydp
  else
    up
    up.wxmin  wxmin    :=       \ restore up wx&wy min&max
    up.wymin  wymin    :=
    up.wxmax  wxmax    :=
    up.wymax  wymax    :=
    n.wt1  ramp  sub[ up.ax.ini , up.ax.no , up.ax.int ]
    b            sub[ up.ax.ini , up.ax.no , up.ax.int ]   i.run      xydp
    dp
    dp.wxmin  wxmin    :=       \ restore dp wx&wy min&max
    dp.wymin  wymin    :=
    dp.wxmax  wxmax    :=
    dp.wymax  wymax    :=
    n.wt1  ramp  sub[ dp.ax.ini , dp.ax.no , dp.ax.int ]
    a            sub[ dp.ax.ini , dp.ax.no , dp.ax.int ]   i.run      xydp
  then
  "time    "swap
  cr  s.type   cr s.type    ."   "    i     n.type
  i.int  0  =
  if
    isign  1  =
    if
      i.ini  my.power
      2  n.power i.run +  **
      2  n.power i.run 1 - +  **  -
    else
      i.ini  my.power
      2  n.power i.run 1 -  -   **
      2  n.power i.run      -   **  -  neg
    then
```

```
      i.int  :=
      i.int
      0  i.int  :=
    else
      i.int
    then
    i.run 1 +  i.run :=
  +loop
;
\ gd  sc  "time  s.type
gd  sc  "time  s.type

\ ------------------------------------------------------------------------
run
\ ------------------------------------------------------------------------
: inv1
  0.  b :=   1.  b [ 5 ]  :=
  b  inverse.wt
     gd hp
     up 1 yap
     swt.t  becomes> b
  b  forward.wt
     sub[ 1 , p.pnts ]
     dp 1 yap
\  MY.PAUSE
  0.  b :=   1.  b [ 16  2 n.power 1 - ** + ]  :=
  b  inverse.wt
     sub[ 1 , p.pnts ]
     gd hp 1 yap
;
: inv2
  0.  b :=   1.  b [ 8   2 n.power 2 - ** + ]  :=
  b  inverse.wt
     sub[ 1 , p.pnts ]
     2 ydp

  0.  b :=   1.  b [ 4   2 n.power 3 - ** + ]  :=
  b  inverse.wt
     sub[ 1 , p.pnts ]
     3 ydp
;
: inv3
  0.  b :=   1.  b [ 2   2 n.power 4 - ** + ]  :=
  b  inverse.wt
     sub[ 1 , p.pnts ]
     5 ydp
;

: inv4
  0.  b :=   1.    b [ 24 ]  :=
  \ b [ 5 ]  :=
  b  inverse.wt
  gd hp
     up 1 yap
     swt.t  becomes> b
```

```
  b  forward.wt
     sub[ 1 , p.pnts ]
     dp 1 yap
  \ MY.PAUSE
  0.  b :=   1.  b [ 16  2 n.power 1 - ** + ]  :=
  b  inverse.wt
     sub[ 1 , p.pnts ]
     gd hp 1 yap
;
: inv5
  0.  b :=   1.  b [ 8   2 n.power 2 - ** + ]  :=
  b  inverse.wt
     sub[ 1 , p.pnts ]
     2 ydp
  0.  b :=   1.  b [ 4   2 n.power 3 - ** + ]  :=
  b  inverse.wt
     sub[ 1 , p.pnts ]
     3 ydp
;
: inv6
  0.  b :=   1.  b [ 2   2 n.power 4 - ** + ]  :=
  b  inverse.wt
     sub[ 1 , p.pnts ]
     5 ydp
;
integer scalar i.ele  scalar i.lin  scalar num.dig
  12  i.ele  :=
  1   i.lin  :=
: lab.spe
  " Spec:"   wlet "cat  " ,"   "cat
  "  JWP Level "  "cat  jwp.level 0 n>s  "cat  "  at location " "cat
  i.ele  0 n>s  "cat  "  [WB<-("  "cat
  \ 2 jwp.level 1 + **  num.dig  n>s  "cat  " )->WP]" "cat
  2 jwp.level 1 + **  0  n>s  "cat  " )->WP]" "cat
  normal.coords  0.5  0.03  position  centered.label
  1.  1.  position   world.coords
;
: lab.x
  normal.coords     position  centered.label
  1.  1.  position   world.coords
;

dir.asy.sys " 123io.sov" "cat defer> load.overlay
26  0   29  79  window  hp.menu.wnd
-1   isign := \ ." <- or -> :"
: set.let.p1
  hp.menu.wnd \  {border}
  2   0  goto.xy
      ." Wavelet (" " on22a/s, bo220/d, so0/d , le, me" s.type ." ):"
      wlet  s.type
  60  0  goto.xy    2  0 fix.format
      ." <- or -> :"   isign  n.type
      1 0 fix.format
  2   1  goto.xy
      ." WP(1) / WB(0):"   wp.flag 1 = if " WP(1)" else " WB(0)"  then
```

```
    s.type   wp.flag  n.type
  30  1  goto.xy
    ." JWP Level:"   jwp.level   n.type
  60  1  goto.xy     4 0 fix.format
    ." #.pnts:"           n.wt1  n.type
  2   2  goto.xy     2 0 fix.format
    ." .Pcx: "                file.ext  s.type
  30  2  goto.xy
    ." 123->"           wlet " .wks" "cat  s.type
  60  2  goto.xy
    ." File.#:"               file.no   n.type
;
: set.let
  hp.menu.wnd \  {border}
  sc
  set.let.p1
  2   0  goto.xy
    ." Wavelet (" " on22a/s, bo220/d, so0/d , le, me" s.type ." ):"
    b.t
    wlet  get.str  wlet ":=
    b.t
  60  0  goto.xy    2  0 fix.format
    \ ." <- or -> :"                -1  get.val isign :=
    ." <- or -> :"     isign get.val isign :=
    1 0 fix.format
  2   1  goto.xy
    ." WP(1) / WB(0):"   wp.flag 1 = if " WP(1)" else " WB(0)"  then
    s.type   wp.flag  get.val   wp.flag :=
  30  1  goto.xy
    ." JWP Level:"   jwp.level  get.val  jwp.level :=
  60  1  goto.xy     4 0 fix.format
    ." #.pnts:"           n.wt1  get.val   n.wt1  :=
  2   2  goto.xy     2 0 fix.format
    ." .Pcx: "                file.ext  get.str file.ext ":=
  30  2  goto.xy
    ." 123->"           wlet " .wks" "cat  get.str tmp.s   ":=
  60  2  goto.xy
    ." File.#:"               file.no   get.val file.no   :=
  dir.lee.asc   tmp.s  "cat   123.file.name  ":=
  isign 1 =
  if
    \ n.wt1  dp.real ramp  becomes> b
    \ 0    b  :=
    cr
    " Has B been set?!"  s.type
  then
\ 123.file.name     defer>  123file.create
\ release.overlay
\ previous.window
;

: set.let.res
  {hp.wnd}
  sc
  cr  ." Wavelet:(" " on22a/s, bo220/d, so0/d , le, me" s.type ." )"
```

```
      wlet  get.str  wlet ":=
  1 0 fix.format
  cr  ." WP (1) or WB (0) :"  wp.flag 1 = if " WP(1)" else " WB(0)"  then
      s.type   wp.flag  get.val  wp.flag :=
  cr  ." JWP Level :"  jwp.level  get.val  jwp.level :=
  4 0 fix.format
  cr  ." Number of Points:" n.wt1  get.val  n.wt1  :=
  2 0 fix.format
  cr  ." Inverse or Forward(1):"  -1  get.val isign :=
  cr  ." File Extention for PCX: " file.ext  get.str file.ext ":=
  cr  ." 123 File name: " wlet " .wks" "cat  get.str tmp.s  ":=
  cr  ." File.no: "                file.no   get.val file.no   :=
  dir.lee.asc   tmp.s  "cat   123.file.name  ":=
  isign 1 =
  if
    \ n.wt1  dp.real ramp  becomes> b
    \ 0   b  :=
    " Has B been set?!"  s.type
  then
  123.file.name     defer>  123file.create
  release.overlay
  previous.window
;
release.overlay

: lee
  isign  -1  =
  if
    n.wt1  dp.real ramp  becomes> b
    1  0 fix.format
    0.  b  :=
    cr ." JWP Level(" wlet s.type ." ):"  jwp.level  get.val   jwp.level :=
\ ---------------------------------------------------------------------------
    cr ." Element "
    wp.flag 1 =
    if
      " (WP: located at level "  jwp.level 0 n>s "cat " )" "cat s.type.3
    else
      " (WB: min. level at "  jwp.level 0 n>s "cat " )" "cat s.type.3
    then
    ."  (Phi<-"
    1  0 fix.format
    jwp.level  9 >=
    if                      4
    else
      jwp.level  6 >=
      if                    3
      else
        jwp.level  3 >=
        if                  2
        else                1
        then
      then
    then
    dup  num.dig :=    0  fix.format
```

```
    2 jwp.level     ** n.type ." |->Psi:[WB"
    \ --------------------------------------- -
    wp.flag 1 =
    if
      ." <-"
      1  0 fix.format
      jwp.level  9 >=
      if                        4
      else
        jwp.level  6 >=
        if                      3
        else
          jwp.level  3 >=
          if                    2
          else                  1
          then
        then
      then
      dup  num.dig :=    0  fix.format
      2 jwp.level 1 + ** n.type ." |->WP]): "
    else
      ." ]): "
    then
\ ----------------------------------------------------------------------------
\     cr ." Element (WB<-"
\     1  0 fix.format
\     jwp.level  9 >=
\     if                        4
\     else
\       jwp.level  6 >=
\       if                      3
\       else
\         jwp.level  3 >=
\         if                    2
\         else                  1
\         then
\       then
\     then
\     dup  num.dig :=    0  fix.format
\     2 jwp.level 1 + ** n.type ." |->WP):"
    3 0 fix.format
    i.ele get.val i.ele :=    1 0 fix.format
    cr ." Clear (1) or Overlap:"     i.lin      get.val   i.lin     :=
    1.  b [ i.ele ] :=
    b  inverse.wt
    wp.flag 1 =
    if
      drop  swp.t
    else
      drop  swt.t
    then
  else
    {hp.wnd}  " Have you set B (in --> mode)?" s.type  \ my.pause
    cr ." Choose array for Forward Transform "
    " 0:default B, 1:WB.t, 2:WP.t" s.type.3
```

```
    0 get.val
    case
      1  of   swt.t  endof
      2  of   swp.t  endof
      0  of   b      endof
    endcase
    previous.window
    forward.wt     \ always d.t for WB and WP -- set Wp.flag
  then
  n.power  10 = if  sub[ 1 , 1000 ] then
  n.power   9 = if  sub[ 1 ,  500 ] then
  n.power   8 = if  sub[ 1 ,  250 ] then
  n.power   7 = if  sub[ 1 ,  120 ] then
  n.power   6 = if  sub[ 1 ,   70 ] then
  i.lin
  i.lin     1 =
  if  yap   lab.spe   else  ydp  then
  10 5 fix.format
;
10 5 fix.format
: NOR.TEST
  hp
  b  forward.wt  1 yap
  norm.wp.best
  norm.l 1 yap  my.pause
  norm.v 2 yap
;
: plot
\  jwp.level 0 =
\  if
\    awp.t xsect[ n.power 1 + , ! ]  tmp.v.i yap
\  else
  n.wt1  my.power
  \ jwp.level tmp.v.i :=
  jwp.level 1 - tmp.v.i :=
  hp
  begin
    tmp.v.i 1 +  tmp.v.i :=
\ \\     tmp.v.i    1 =
    tmp.v.i   0 =
    if
      awp.t xsect[ n.power 1 + , ! ]
    else
      awp.t xsect[ tmp.v.i , ! ]
    then
    tmp.v.i
    tmp.v.i  jwp.level =
    if
      yap
    else
      ydp
    then
    hp   \   previous.window
    \    tmp.v.i  n.type
    normal.coords
```

```
    tmp.v.i   0   n>s    tmp.v.i  20. /   0.05  position
    tmp.v.i line.type label
    tmp.v.i 1 +  20. /   0.05  position
    world.coords
    my.pause
    \ previous.window
    tmp.v.i  n.power  =
  until
;
: get.lab
  wlet tmp.s.u ":=
  tmp.s.u "upper "drop
  normal.coords  .5  .05  position
  tmp.s.u
  centered.label
  world.coords
;

sd
cdtNow load LLet-DnR.asy \
cdtNow load LLet-Blo.asy \
cdtNow load LLet-Pha.asy \
cdtNow load LLet-Nor.asy \
cdtNow load LLet-Ent.asy \
  \ \ cdtNow   load LLet-Lev.txt \
cdtNow load LLet-WPM.asy \
cdtNow load LLet-Act.asy \
```

LLet-Act.asy

勇伯程式庫 A01506-20070125-A057 ❖ January 25, 2007 ❖ [Page count: 14]

```
: LLet-Act.asy ; \ %----- Asyst Prog
\ : LLet-Ba2.asy ; \ : LLet-Bat.asy ;

1 file.no :=
1 gp.flag :=
\ " abc" file.ext ":=
" pcx" file.ext ":= \ for imaging files

3 string act.flag.str
integer scalar act.flag   scalar plot.gp

\ : action
\   act.flag
\   case
\   0  of  dec&rec     endof
\   1  of  dec&rec     endof
\   2  of  m0.s.phase  endof
\   3  of  norm.bat    endof
\   4  of  ent.bat    endof
\ \ 4  of  \ cr  ." #.weight "   #.wei   get.val  #.wei   :=
\ \         cr  ." array?"
\ \         cr  ." s.freq    "   s.freq  get.val  s.freq  :=
\ \         cr  ." filter    "   tmp.v.i get.val  tmp.v.i :=
\ \         s.freq  tmp.v.i  fre.mod
\ \                    endof
\   endcase
\ ;
: action
  act.flag.str " D" "=
  if
    dec&rec
  then
  act.flag.str " P" "=
  if
    m0.s.phase
  then
  act.flag.str " N" "=
  if
    norm.bat
  then
  act.flag.str " E" "=
  if
    ent.bat
  then
;

: ACT.me.p1
  " me"     wlet ":=  action
;
: ACT.me
  dir.lee.asc  " me"    "cat  " .wks" "cat   123.file.name  ":=
  1  file.no  :=
```

```
  ACT.me.p1
;
: ACT.le.p1
  " le"    wlet ":=  action
;
: ACT.le
  dir.lee.asc  " le"    "cat  " .wks" "cat   123.file.name  ":=
  1  file.no  :=
  ACT.le.p1
;
: ACT.so.p1
  " so0"   wlet ":=  action
  " sod"   wlet ":=  action
;
: ACT.so
  dir.lee.asc  " so"     "cat  " .wks" "cat   123.file.name  ":=
  ACT.so.p1
;
: ACT.tmp
  dir.lee.asc  " 123"  "cat  " .wks" "cat   123.file.name  ":=
\  " so0"   wlet ":=  5   i.ele  :=  action
\  " so0"   wlet ":=  384 i.ele  := action
\ " bo33d"   wlet ":=  5   i.ele  :=  action
\ " bo33d"   wlet ":=  384 i.ele  :=  action
  " on22a"   wlet ":= action
;
: ACT.bo1.p1
\   " bo110" wlet ":=  action
\   " bo11d" wlet ":=  action
\   " bo130" wlet ":=  action
\   " bo13d" wlet ":=  action
\   " bo150" wlet ":=  action
\   " bo15d" wlet ":=  action
  " bo110" wlet ":=  action
  " bo130" wlet ":=  action
  " bo150" wlet ":=  action
  " bo11d" wlet ":=  action
  " bo13d" wlet ":=  action
  " bo15d" wlet ":=  action
;
: ACT.bo1
  dir.lee.asc  " bo1" "cat  " .wks" "cat   123.file.name  ":=
  1  file.no  :=
  ACT.bo1.p1
;
: ACT.bo2.p1
\   " bo220" wlet ":=  action
\   " bo22d" wlet ":=  action
\   " bo240" wlet ":=  action
\   " bo24d" wlet ":=  action
\   " bo260" wlet ":=  action
\   " bo26d" wlet ":=  action
\   " bo280" wlet ":=  action
\   " bo28d" wlet ":=  action
  " bo220" wlet ":=  action
```

```
  " bo240" wlet ":=  action
  " bo260" wlet ":=  action
  " bo280" wlet ":=  action
  " bo22d" wlet ":=  action
  " bo24d" wlet ":=  action
  " bo26d" wlet ":=  action
  " bo28d" wlet ":=  action
;
: ACT.bo2
  dir.lee.asc  " bo2" "cat  " .wks" "cat   123.file.name  ":=
  1  file.no  :=
  ACT.bo2.p1
;
: ACT.bo3.p1
\    " bo310" wlet ":=  action
\    " bo31d" wlet ":=  action
\    " bo330" wlet ":=  action
\    " bo33d" wlet ":=  action
\    " bo350" wlet ":=  action
\    " bo35d" wlet ":=  action
\    " bo370" wlet ":=  action
\    " bo37d" wlet ":=  action
\    " bo390" wlet ":=  action
\    " bo39d" wlet ":=  action
  " bo310" wlet ":=  action
  " bo330" wlet ":=  action
  " bo350" wlet ":=  action
  " bo370" wlet ":=  action
  " bo390" wlet ":=  action
  " bo31d" wlet ":=  action
  " bo33d" wlet ":=  action
  " bo35d" wlet ":=  action
  " bo37d" wlet ":=  action
  " bo39d" wlet ":=  action
;
: ACT.bo3
  dir.lee.asc  " bo3" "cat  " .wks" "cat   123.file.name  ":=
  1  file.no  :=
  ACT.bo3.p1
;
: ACT.onc.p1
  " on11c" wlet ":=  action
  " on22c" wlet ":=  action
  " on33c" wlet ":=  action
  " on44c" wlet ":=  action
  " on55c" wlet ":=  action
;
: ACT.onc
  dir.lee.asc  " onc" "cat  " .wks" "cat   123.file.name  ":=
  1  file.no  :=
  ACT.onc.p1
;
: ACT.ona.p1
  " on22a" wlet ":=  action
  " on33a" wlet ":=  action
```

```
  " on44a" wlet ":=  action
  " on55a" wlet ":=  action
  " on66a" wlet ":=  action
  " on77a" wlet ":=  action
  " on88a" wlet ":=  action
  " on99a" wlet ":=  action
  " on00a" wlet ":=  action
;
: ACT.ona
  dir.lee.asc  " ona" "cat  " .wks" "cat   123.file.name  ":=
  1  file.no   :=
  ACT.ona.p1
;
: ACT.ons.p1
  " on44s" wlet ":=  action
  " on55s" wlet ":=  action
  " on66s" wlet ":=  action
  " on77s" wlet ":=  action
  " on88s" wlet ":=  action
  " on99s" wlet ":=  action
  " on00s" wlet ":=  action
;
: ACT.ons
  dir.lee.asc  " ons" "cat  " .wks" "cat   123.file.name  ":=
  1  file.no   :=
  ACT.ons.p1
;

3  0   24  79   window gd.txt.wnd
6.0    p.b.e :=
6.0    t.w :=
: set.cwt ;
\ : for.norm&ent
\   5   4  goto.xy
\             \ ." Now.file (name.ext) ?"  now.file  get.str  now.file ":=
\              ." Now.file (name.ext):"  now.file  get.str  now.file ":=
\   5   5  goto.xy    8 4 fix.format
\ \            ." Sampling freq KEEP :" s.freq.keep  get.val  s.freq.keep :=
\             ." Sampling freq -- original rate (s.freq.keep):"
\             s.freq.keep  get.val  s.freq.keep :=
\          2 0 fix.format
\   5   6  goto.xy
\             ." array index Inc :" i.inc  get.val  i.inc :=
\   5   7  goto.xy    8 4 fix.format
\             ." Norm.keep.p.BEST :"
\          7 4 fix.format
\          norm.keep.p.best get.val norm.keep.p.best :=
\   5   8  goto.xy
\             ." abs.flag for L**1 :"  abs.flag  get.val abs.flag :=
\          2 0 fix.format
\   5   9  goto.xy
\             ." filter.flag :"  0  get.val  filter.flag :=
\          1 filter.flag =
\          if
\   5   11 goto.xy
```

```
\                 ." Pass band end (Hz) ?"  p.b.e get.val     p.b.e :=
\   5   12 goto.xy
\                 ." Transition width (Hz) ?" t.w get.val  t.w   :=
\           then
\   5   13 goto.xy
\               ." Section level :" n.wt1 my.power n.power  get.val  sec.level :=
\           set.let
\   5   15 goto.xy
\           set.123
\ ;
: for.norm&ent \ batch display of parameters
  5  4  goto.xy
            \ ." Now.file (name.ext) ?"  now.file s.type \ get.str  now.file ":=
             ." Now.file (name.ext):"  now.file s.type \ get.str  now.file ":=
  5  5  goto.xy    8 4 fix.format
\            ." Sampling freq KEEP :" s.freq.keep  n.type \ get.val  s.freq.keep :=
            ." Sampling freq -- original rate (s.freq.keep):"
            s.freq.keep  n.type \ get.val  s.freq.keep :=
        2 0 fix.format
  5  6  goto.xy
            ." array index Inc :" i.inc  n.type \ get.val  i.inc :=
  5  7  goto.xy    8 4 fix.format
            ." Norm.keep.p.BEST :"
        7 4 fix.format
        norm.keep.p.best n.type \ get.val norm.keep.p.best :=
  5  8  goto.xy
            ." abs.flag for L**1 :"  abs.flag n.type \ get.val abs.flag :=
        2 0 fix.format
  5  9  goto.xy
            ." filter.flag :" filter.flag n.type \ get.val  filter.flag :=
        1 filter.flag =
        if
  5  11 goto.xy
              ." Pass band end (Hz) ?"  p.b.e n.type \ get.val     p.b.e :=
  5  12 goto.xy
              ." Transition width (Hz) ?" t.w n.type \ get.val  t.w   :=
        then
  5  13 goto.xy
             ." Section level :" n.wt1 my.power n.power
             sec.level :=
             sec.level n.type
  5  14 goto.xy 4 0 fix.format
  act.flag.str "upper " E" "=
  if
    ." #.div:" #.div n.type \ get.val  #.div :=
  then
  set.let
  5  15 goto.xy
  set.123
;
: for.norm&ent.get \ interactive input
  5  4  goto.xy
            \ ." Now.file (name.ext) ?"  now.file  get.str  now.file ":=
             ." Now.file (name.ext):"  now.file  get.str  now.file ":=
  5  5  goto.xy    8 4 fix.format
```

```
\           ." Sampling freq KEEP :" s.freq.keep  get.val  s.freq.keep :=
          ." Sampling freq -- original rate (s.freq.keep):"
          s.freq.keep  get.val  s.freq.keep :=
       2 0 fix.format
  5  6  goto.xy
          ." array index Inc :" i.inc  get.val  i.inc :=
  5  7  goto.xy    8 4 fix.format
          ." Norm.keep.p.BEST :"
       7 4 fix.format
       norm.keep.p.best get.val norm.keep.p.best :=
  5  8  goto.xy
          ." abs.flag for L**1 :"  abs.flag  get.val abs.flag :=
       2 0 fix.format
  5  9  goto.xy
          ." filter.flag :"  0  get.val  filter.flag :=
       1 filter.flag =
       if
  5  11 goto.xy
            ." Pass band end (Hz) ?"  p.b.e get.val    p.b.e :=
  5  12 goto.xy
            ." Transition width (Hz) ?" t.w get.val  t.w   :=
       then
  5  13 goto.xy
           ." Section level :" n.wt1 my.power n.power  get.val  sec.level :=
  5  14 goto.xy 4 0 fix.format
  act.flag.str "upper " E" "=
  if
    ." #.div:" #.div get.val  #.div :=
  then
  set.let
  5  15 goto.xy
  set.123
;
\ : for.norm&ent.res
\         cr ." Now.file (name.ext):"  now.file  get.str  now.file ":=
\            8 4 fix.format
\         cr ." Sampling freq -- original rate (s.freq.keep):"
\            s.freq.keep  get.val  s.freq.keep :=
\         cr ." Norm.keep.p.BEST?"
\         7 4 fix.format
\         norm.keep.p.best get.val norm.keep.p.best :=
\         cr ." abs.flag for L**1"  abs.flag  get.val abs.flag :=
\         2 0 fix.format
\         cr ." Index Interval for Sub-sampling:" 4  get.val  i.inc :=
\         cr ." filter.flag?"  0  get.val  filter.flag :=
\         1 filter.flag =
\         if
\           cr ." Pass band end (Hz)?" 6 get.val     p.b.e :=
\           cr ." Transition width (Hz)?" 6 get.val  t.w   :=
\         then
\         cr ." Section level:" n.wt1 my.power n.power  get.val  sec.level :=
\         set.let
\         set.123
\ ;
1  i.beg :=
```

```
4  i.inc  :=
\ : set.act
\   gd
\   gd.txt.wnd    {border}
\   3   3   goto.xy
\      ." 1:Decomp&Recon, 2:MO.phase, 3:Norm.wt, 4:Entropy, 5:CWT --"
\   act.flag  get.val  act.flag :=
\   act.flag
\   case
\     3  of  for.norm&ent  endof
\     4  of  for.norm&ent
\             \        5   14 goto.xy
\           4 0 fix.format
\           cr ." #.div:"  #.div  get.val  #.div :=
\                              endof
\     5  of
\           5   4  goto.xy
\              ." Now.file (name.ext):"  now.file  get.str  now.file ":=
\           5   5  goto.xy
\           4 0 fix.format
\           n.wt1  my.power
\              ." Number of points of wavelet (n.wt1):"
\              2 n.power **  get.val  n.wt1 :=
\           5   6  goto.xy    8 4 fix.format
\              ." Sampling freq -- original rate (s.freq.keep):"
\              s.freq.keep  get.val  s.freq.keep :=
\           5   7  goto.xy
\           2 0 fix.format
\              ." Index Interval for Sub-sampling:" i.inc  get.val  i.inc :=
\           5   8  goto.xy
\              ." Index of first point of array" i.beg  get.val  i.beg :=
\              \ ." Index of first point of array" i.beg  get.val  i.beg :=
\
\           cr set.cwt
\           endof
\   endcase
\ ;
: set.act
  gd
  gd.txt.wnd    {border}
  3   2   goto.xy
  ." D:Dec&Rec, P:Phase.MOs, N:Norm.wt, E:Entropy, C:CWT, W:WCoh :: "
  act.flag.str s.type \ get.str  act.flag.str ":=
  \ act.flag.str get.str  act.flag.str ":=
  cr
  act.flag.str "upper " N" "=
  if
    for.norm&ent
  then
  act.flag.str "upper " E" "=
  if
    for.norm&ent
\     5  14 goto.xy 4 0 fix.format
\     ." #.div:"  #.div get.val  #.div :=
  then
```

```
\   act.flag.str "upper " C" "=
\   if
\     cwt.56
\     for.case5&6
\     cr set.cwt
\     cr new.pause
\     cwt
\   then
\   act.flag.str "upper " W" "=
\   if
\     cwt.56
\     for.case5&6
\     cr set.cwt
\     5  19 goto.xy
\     \ cr
\     ." Real or Imag or Comp:" rea.or.com s.type \ get.str "upper rea.or.com ":=
\     cr new.pause
\     cr set.123
\     10 5 fix.format
\     run.coh
\   then
;
: set.act.res
  cr  ." 1:Decomp&Recon, 2:MO.phase, 3:Norm.wt, 4:Entropy, 5:CWT --"
  5  get.val  act.flag :=
  act.flag
  case
    3  of  for.norm&ent  endof
    4  of  for.norm&ent
            4 0 fix.format
        cr ." #.div:"  #.div  get.val  #.div :=
                          endof
    5  of
          cr ." Now.file (name.ext):"  now.file  get.str  now.file ":=
          4 0 fix.format
          n.wt1  my.power
          cr ." Number of points of wavelet (n.wt1):"
          2 n.power **  get.val  n.wt1 :=
          8 4 fix.format
          cr ." Sampling freq -- original rate (s.freq.keep):"
            s.freq.keep  get.val  s.freq.keep :=
          2 0 fix.format
          cr ." Index Interval for Sub-sampling:" i.inc  get.val  i.inc :=
          cr ." Index of first point of array" i.beg  get.val  i.beg :=
     \    cr ." Number of freq resolution #.fre"  1  get.val  #.fre :=
     \    cr ." Section level:" n.wt1 my.power n.power  get.val  sec.level :=
     \    set.let
     \    set.123
     \  cr ." #.div:"  #.div  get.val  #.div :=
                          endof

  endcase
  \ LEE
;
```

```
: app.p.s
  dir.lee.asc  now.file  "cat  ftoa.bat
  sub[ 1 , 2 n.power ** , i.inc ]
  2 n.power **  999  " x"  my.p.s
  dup  catenate
  norm.wt&wp
  \  8 file.no  :=
  " power.s"  123.ary.s "[ 1 ]  ":=
  to.123
;

integer scalar gp.fac

: copy.pcx
  0  tmp.v.i :=
  begin
    tmp.v.i   1 + tmp.v.i :=
    cdaNow  tmp.v.i 0 n>s  " ."  "cat file.ext "cat       defer>  copy
            tmp.v.i 0 n>s  " .g" "cat plot.gp  0 n>s "cat defer>  to
    tmp.v.i  file.no 1 -  =
  until
    cdaNow  " *."  file.ext "cat                          defer>  delete
;
\ : act.all.1
\ \   cr ." Now file (b0w6020.2)"  " b0w6020.4"  get.str  now.file ":=
\   cr ." Now file (b0w6020.2)"  now.file  get.str  now.file ":=
\   cr ." 123 file name : "  " 123"  get.str
\   dir.lee.asc  "swap  "cat  " .wks" "cat   123.file.name  ":=
\   cr ." Group factor (10 cols --> 1) : "  1  get.val  gp.fac  :=
\   cr ." Section level for norm:"  n.power  get.val  sec.level :=
\
\   1              file.no  :=
\   10 gp.fac *  col.inc  :=
\
\   1  gp.fac *  gp.flag  :=
\   1  plot.gp  :=
\   1  file.no  :=
\      act.le.p1
\      act.me.p1
\      wp.flag
\      0  wp.flag :=
\      act.so.p1
\      act.flag  4  <>
\      if
\        to.123.1  \ additional col
\        to.123.2  \ additional col
\        copy.pcx
\        8 file.no :=
\        app.p.s
\      else
\        copy.pcx
\      then
\        wp.flag :=
\ ;
\ =================================================================
```

```
: ACT.Gp.Ini \  <-- Part of act.all.1
  \ cr ." Now file (b0w6020.2)"  now.file  get.str  now.file ":=
                 " b0w6020.2"                     now.file ":=
  \ cr ." Group factor (10 cols --> 1) : "  1  get.val  gp.fac  :=
                                            1           gp.fac  :=
  \ cr ." Section level for norm:" n.wt1 my.power n.power get.val  sec.level :=
                                   n.wt1 my.power n.power          sec.level :=
  1             file.no  :=
  10 gp.fac *   col.inc  :=
  \
  1  gp.fac *   gp.flag  :=
  1  plot.gp  :=
  1  file.no  :=
  \   wp.flag
  \   0  wp.flag :=
  \   wp.flag :=
;
: ACT.Gp.LMS
  " le"     wlet ":=  action
  " me"     wlet ":=  action
  " so0"    wlet ":=  action
  " sod"    wlet ":=  action
  copy.pcx
;
: ACT.Gp.onC
  " on11c" wlet ":=  action
  " on22c" wlet ":=  action
  " on33c" wlet ":=  action
  " on44c" wlet ":=  action
  " on55c" wlet ":=  action
  copy.pcx
;
: ACT.Gp.onA1
  " on22a" wlet ":=  action
  " on33a" wlet ":=  action
  " on44a" wlet ":=  action
  " on55a" wlet ":=  action
  " on66a" wlet ":=  action
  copy.pcx
;
: ACT.Gp.onA2
  " on77a" wlet ":=  action
  " on88a" wlet ":=  action
  " on99a" wlet ":=  action
  " on00a" wlet ":=  action
  copy.pcx
;
: ACT.Gp.onS1
  " on44s" wlet ":=  action
  " on55s" wlet ":=  action
  " on66s" wlet ":=  action
  " on77s" wlet ":=  action
  copy.pcx
;
: ACT.Gp.onS2
```

```
  " on88s" wlet ":=  action
  " on99s" wlet ":=  action
  " on00s" wlet ":=  action
  copy.pcx
;
: ACT.Gp.bo10
  " bo110" wlet ":=  action
  " bo130" wlet ":=  action
  " bo150" wlet ":=  action
  copy.pcx
;
: ACT.Gp.bo1D
  " bo11d" wlet ":=  action
  " bo13d" wlet ":=  action
  " bo15d" wlet ":=  action
  copy.pcx
;
: ACT.Gp.bo20
  " bo220" wlet ":=  action
  " bo240" wlet ":=  action
  " bo260" wlet ":=  action
  " bo280" wlet ":=  action
  copy.pcx
;
: ACT.Gp.bo2D
  " bo22d" wlet ":=  action
  " bo24d" wlet ":=  action
  " bo26d" wlet ":=  action
  " bo28d" wlet ":=  action
  copy.pcx
;
: ACT.Gp.bo30
  " bo310" wlet ":=  action
  " bo330" wlet ":=  action
  " bo350" wlet ":=  action
  " bo370" wlet ":=  action
  " bo390" wlet ":=  action
  copy.pcx
;
: ACT.Gp.bo3D
  " bo31d" wlet ":=  action
  " bo33d" wlet ":=  action
  " bo35d" wlet ":=  action
  " bo37d" wlet ":=  action
  " bo39d" wlet ":=  action
  copy.pcx
;

10 5 fix.format
\ ==========================================================
: act.all.1
\ \   cr ." Now file (b0w6020.2)"  " b0w6020.4"  get.str  now.file ":=
\   cr ." Now file (b0w6020.2)"  now.file  get.str  now.file ":=
\   cr ." 123 file name : (123.wks -- No Path)"  " 123.wks"  get.str
\   dir.lee.asc  "swap  "cat  123.file.name  ":=
```

```
  cr ." Now file (b0w6020.2)"  now.file  get.str  now.file ":=
  cr ." 123 file name (G:\path\123.wks -- With path): "  123.file.name  get.str
  123.file.name  ":=

  cr ." Group factor (10 cols --> 1) : "  1  get.val  gp.fac  :=
  \ cr ." Section level for norm:"  n.power  get.val  sec.level :=
  cr ." Section level for norm:" n.wt1 my.power n.power get.val  sec.level :=

  1              file.no  :=
  10 gp.fac *  col.inc  :=

  1  gp.fac *  gp.flag  :=
  1  plot.gp  :=
  1  file.no  :=
     act.le.p1
     act.me.p1
     wp.flag
     0  wp.flag :=
     act.so.p1
  \ ----------------------------------------
\       act.flag  4  <>
\       if
\         to.123.1  \ additional col
\         to.123.2  \ additional col
\         copy.pcx
\         8 file.no :=
\         app.p.s
\       else
\         copy.pcx
\       then
\         wp.flag :=
  \  act.flag  4  <>
     act.flag.str " E"  "=
     if
       copy.pcx
     else
       to.123.1  \ additional col
       to.123.2  \ additional col
       copy.pcx
       8 file.no :=
       app.p.s
     then
       wp.flag :=
;
: act.all.2
  2  gp.fac *  gp.flag  :=
  2  plot.gp  :=

  1  file.no  :=
     act.ona.p1
     copy.pcx
;
: act.all.3
  3  gp.fac *  gp.flag  :=
  3  plot.gp  :=
```

```
  1  file.no  :=
     act.ons.p1
     copy.pcx
;
: act.all.4
  4  gp.fac *  gp.flag  :=
  4  plot.gp  :=
  1  file.no  :=
     act.onc.p1
     copy.pcx
;
: act.all.5
  wp.flag
  5  gp.fac *  gp.flag  :=
  5  plot.gp  :=
  1  file.no  :=   0 wp.flag :=
     act.bo1.p1
     wp.flag :=
     copy.pcx
;
: act.all.6
  wp.flag
  6  gp.fac *  gp.flag  :=
  6  plot.gp  :=
  1  file.no  :=   0 wp.flag :=
     act.bo2.p1
     wp.flag :=
     copy.pcx
;
: act.all.7
  wp.flag
  7  gp.fac *  gp.flag  :=
  7  plot.gp  :=
  1  file.no  :=   0 wp.flag :=
     act.bo3.p1
     wp.flag :=
     copy.pcx
;

: act.all
  act.all.1
  act.all.2
  act.all.3
  act.all.4
  act.all.5
  act.all.6
  act.all.7
;
10 5 fix.format

\
\ " b0w6020.4"     now.file  ":= \ 200Hz  4Ch
\ " f0w6020.2"     now.file  ":= \ 40Hz   2Ch
\ " chirp-al.dat"  now.file  ":= \ --Hz   1Ch
```

LLet-Blo.asy

勇伯程式庫 A01507-20070125-A058 ❖ January 25, 2007 ❖ [Page count: 5]

```
: LLet-Blo.asy ; \ %----- Asyst Prog
\ Be sure to use PSI to blow up.
\ Since it is already in the psi region

\ 1       file.no    :=
\ " pcx"  file.ext  ":=
" d" yadp.flag ":=
dp.real scalar blow.const
: get.3.pnts
  swt.t [ 2 n.power 1 - **  3 -                          ]  -5  n>s "    " "cat
  swt.t [ 2 n.power 1 - **  2 -                          ]  -5  n>s "    " "cat
  "cat
  swt.t [ 2 n.power 1 - **  1 -                          ]  -5  n>s "    " "cat
  "cat
  swt.t [ 2 n.power 1 - **      ]  -5  n>s "cat "    " "cat
  swt.t [ 2 n.power 1 - **  1 +                          ]  -5  n>s "cat
  "    " "cat
  swt.t [ 2 n.power 1 - **  2 +                          ]  -5  n>s "cat
  "    " "cat
  swt.t [ 2 n.power 1 - **  3 +                          ]  -5  n>s "cat
\ swt.t [ 2 n.power 1 - **  2 n.power jwp.level - ** - ]  -5  n>s "    " "cat
\ swt.t [ 2 n.power 1 - **      ]  -5  n>s "cat "    " "cat
\ swt.t [ 2 n.power 1 - **  2 n.power jwp.level - ** + ]  -5  n>s "cat
  normal.coords 0.5 0.8 .025 i.cou * - position centered.label world.coords
;
: yadp
  yadp.flag  " d"  "=
  \ if  ydp   get.3.pnts
  if  ydp   get.3.pnts  file.ext make.abc
  else
      yap   get.3.pnts  file.ext make.abc
  then
;
4 string i.ele.ori.str
4 string jwp.ori.str
: app.ramp.col.blow
  wlet tmp.s.u ":=
  tmp.s.u "upper "drop
  \ ---------------------------------------- ■
  \ \ $$[(8,512) B031D-<POL2,Ori12r,XPt248,XPw3>]
  \ ---------------------------------------- ■
  " $$[("  file.no 0 n>s "cat " ," "cat
  123.ary.n []size swap drop 0 n>s "cat
  " ) " "cat
  tmp.s.u  "cat
  \    " -p" "cat
  \    wp.flag  0 n>s "cat
  \    " L" "cat jwp.level 0 n>s "cat
  " -" "cat " <" "cat
  jwp.ori.str "cat
  " ,Ori" "cat
  i.ele.ori.str "cat
```

```
  " r" "cat
  " ,XPt" "cat
  i.ele  0 n>s "cat
  " ,XPw" "cat
  n.power jwp.level -     0 n>s "cat
  \ ----------------------------------------
  " >]" "cat
  123.ary.s "[ 1 ] ":=
  \
  123.ary.n []size swap drop real ramp 1.0 * becomes> 123.ary.n
  \
  to.123
  file.no 1 + file.no := \ post inc of column
;
: blow
  wlet tmp.s.u ":=
  tmp.s.u "upper "drop
  1 file.no :=
  sc
  " Make the BASIS curve first -- Lee" s.type   \  my.pause
  cr ." Wlet:"  wlet get.str  wlet ":=
  lee

  tmp.s.u " -P" "cat  wp.flag  0 n>s "cat   " L" "cat
  jwp.level 0 n>s "cat    123.ary.s "[ 1 ] ":=
  \ ----------------------------------------
  " P" wp.flag  0 n>s "cat
  " L" "cat jwp.level 0 n>s "cat
  jwp.ori.str ":=
  \ ----------------------------------------
  wp.flag 1 =
  if
    swp.t becomes> 123.ary.n
  else
    swt.t becomes> 123.ary.n
  then
  to.123
  file.ext make.abc
  i.ele  0 n>s   \ For Rotated
  \ ----------------------------------------
  "dup i.ele.ori.str ":=
  \ ----------------------------------------

  wp.flag  \ for recovery

  wp.flag 1 =
  if
    swp.t  becomes> swt.t
  then
  0    wp.flag :=
  2  0 fix.format
  wlet  3 1 "sub   tmp.s ":=
  tmp.s          " 2"   "=
  tmp.s          " 3"   "=  or
    if
```

```
                      4  \ 16phi + 16psi
    else
     tmp.s          " 4"   "=
     tmp.s          " 5"   "=  or
     tmp.s          " 6"   "=  or
     tmp.s          " 7"   "=  or
     tmp.s          " 8"   "=  or
     if
                       5  \ 32phi + 32psi
     else
                       6  \ 64phi + 64psi
     then
    then

\ ------
drop  6
\ ------
       jwp.level  :=
    1  n.blow     :=
    6  n.blow     :=

    \  \\ " h"  phi.or.psi  ":=
    " s"  phi.or.psi  ":=
    \ cr ." pHi(H) or pSi(S):"  phi.or.psi  get.str  phi.or.psi  ":=
    cr ." pSi(s) only:"  phi.or.psi  get.str  phi.or.psi  ":=
    cr ." Default JWP Level(" wlet s.type ." ):( >4)"
          jwp.level  get.val  jwp.level :=
    cr ." Number of Blow-Up:"              n.blow     get.val   n.blow
:=
        " Blow Factor 2^"  S.TYPE  n.power jwp.level -  n.blow *  n.type
    cr ." Blow-up Center (WB--"
    1  0 fix.format
    jwp.level  9 >=
    if                        4
    else
      jwp.level  6 >=
      if                      3
      else
        jwp.level  3 >=
        if                    2
        else                  1
        then
      then
    then
    dup  num.dig :=    0  fix.format
    2 jwp.level 1 + ** n.type ." --WP):"
    3 0 fix.format
    \ i.ele
    2 n.power 1 - **  get.val i.ele :=    1 0 fix.format
    \ cr ." Clear (1) or Overlap:"    i.lin     get.val   i.lin     :=
      0.    b :=
      swt.t sub[ i.ele 2 jwp.level 1 - **  1  -  - , 2 jwp.level ** ]
      b   sub[ 1 , 2 jwp.level ** ] :=
      \ ---> Always is now the "smooth information" for both psi and phi
```

```
\ swt.t
\ n.wt1 2 /  i.ele -  rotate
\ 5 yap

  wlet 1 2 "sub " on"  "=
  if
    2 sqrt  n.power jwp.level - **
  else
  \   1.D
    2 sqrt  n.power jwp.level - **
  then
  blow.const  :=
  gd hp

  swt.t
  i.ele  n.wt1 2 /  -   neg rotate
  dup
  " r"
  "cat  \ from the first i.ele
  \ ----------------------------------------
  0  i.cou :=
  i.cou 0 n>s  " -" "cat  i.ele 0 n>s  "cat
  "  (" "cat
  "swap "cat " )" "cat
  " Seq.  " "swap "cat \ \ added
  \ ----------------------------------------
  123.ary.s "[ 1 ] ":=
  becomes> 123.ary.n  to.123
  5 yap
  file.ext make.abc

\ swt.t [ i.ele 1 - ]  -5  n>s  "   " "cat
\ swt.t [ i.ele     ]  -5  n>s  "cat "   " "cat
\ swt.t [ i.ele 1 + ]  -5  n>s  "cat
  swt.t [ i.ele            3 -                        ]  -5  n>s "   " "cat
  swt.t [ i.ele            2 -                        ]  -5  n>s "   " "cat
  "cat
  swt.t [ i.ele            1 -                        ]  -5  n>s "   " "cat
  "cat
  swt.t [ i.ele                ]  -5  n>s "cat "   " "cat
  swt.t [ i.ele            1 +                        ]  -5  n>s "cat
  "   "  "cat
  swt.t [ i.ele            2 +                        ]  -5  n>s "cat
  "   "  "cat
  swt.t [ i.ele            3 +                        ]  -5  n>s "cat
  "dup
\ normal.coords   0.5  0.8      position  centered.label  world.coords
  normal.coords   0.5  0.8 .025 + position  centered.label  world.coords
  \ \\ " h"  phi.or.psi  ":=
  " s"  phi.or.psi  ":=
  1  i.cou :=
  b inverse.wt  drop
  swt.t  blow.const *  becomes> swt.t
  swt.t
  i.cou 0 n>s  " -" "cat  i.ele 0 n>s  "cat
```

```
  " Seq.  " "swap "cat

  123.ary.s "[ 1 ]  ":=
  swt.t  becomes> 123.ary.n  to.123
  get.p.pnts   sub[ 1 , p.pnts ]   1  yadp  lab.spe
  normal.coords   0.5  0.8      position  centered.label  world.coords
  get.3.pnts
\  file.ext make.abc

  n.blow 1 >
  if
    begin
      i.cou  1 +  i.cou :=
      0  b :=
      swt.t sub[ 2 n.power 1 - **
                   2 jwp.level 1 - **  1  -  - , 2 jwp.level ** ]
      b   sub[ 1 , 2 jwp.level ** ] :=
      b  inverse.wt  drop
      swt.t  blow.const *  becomes> swt.t
      swt.t
      \ ----------------------------------------
      i.cou 0 n>s  " -" "cat i.ele 0 n>s "cat
      " Seq.  " "swap "cat
      123.ary.s "[ 1 ]  ":=
      swt.t  becomes> 123.ary.n  to.123
      get.p.pnts  sub[ 1 , p.pnts ]  i.cou  yadp   \ ydp
      i.cou  n.blow =
    until
  then
  \ ----------------------------------------
  app.ramp.col.blow
  \ ----------------------------------------
  normal.coords 0.5 0.8 .025 i.cou 1 + * - position
  " BLOW-UP AT POINT "  i.ele  0 n>s "cat  "  centered at point " "cat
  2 n.power 1 - **  0 n>s "cat
  centered.label world.coords
  normal.coords 0.5 0.8 .025 i.cou 2 + * - position
  " EACH BLOW-UP SCALE : 2 ** "  n.power jwp.level -     0 n>s "cat
  centered.label
  normal.coords 0.5 0.8 .025 i.cou 3 + * - position
  " Wavelet: "  wlet "cat
  centered.label
  1. 1. position
  world.coords
  file.ext  make.abc
  10 5 fix.format

  wp.flag :=  \ recover
;
```

LLet-Coe.asy

勇伯程式庫 A01508-20070125-A059 ❖ January 25, 2007 ❖ [Page count: 9]

```
: LLet-Coe.asy ; \ %----- Asyst Prog
\  Discrete Wavelet Transform
\  1.    define  PWTset
\  2.    define  PWT
\  3.    run     DWT ( <- pwt <- pwtset ) by giving n.dau.c
\ -----------------------------------------------------------------
nd
\ : iand   \ [ n1 , n2  --  n3 ]
\    #>mask   #>mask   and    mask>#
\ ;

integer scalar n.dau.c
        scalar ncmax
        scalar alg.flag
        scalar isign
        \ scalar n.wt1 \ moved into Lee-Word.asy
        scalar sp.o
        scalar a.size
        scalar b.size
        scalar c.size
        scalar p.size
        scalar q.size
5       string wlet
\ n.fft.pts          n.wt1 :=
  50         ncmax     :=
  4          sp.o      :=
\ real  dim[  ncmax     ]  array cc
dp.real  dim[  ncmax     ]  array cc
         dim[  ncmax     ]  array cr.nr
         dim[  2         ]  array dc1
         dim[  4         ]  array dc2
         dim[  6         ]  array dc3
         dim[  8         ]  array dc4
         dim[  10        ]  array dc5
         dim[  12        ]  array dc6
         dim[  14        ]  array dc7
         dim[  16        ]  array dc8
         dim[  18        ]  array dc9
         dim[  20        ]  array dc10
         dim[  8         ]  array dc4las
         dim[  10        ]  array dc5las
         dim[  12        ]  array dc6las
         dim[  14        ]  array dc7las
         dim[  16        ]  array dc8las
         dim[  18        ]  array dc9las
         dim[  20        ]  array dc10las
dc4las  @[ 1 ]  enter[
     -.107148901418D , -.041910965125D , 0.703739068656D ,
     1.136658243408D , 0.421234534204D , -.140317624179D ,
     -.017824701442D , 0.045570345896D
     ]  drop
dc5las  @[ 1 ]  enter[
```

```
      0.038654795955D , 0.041746864422D , -.055344186117D ,
      0.281990696854D , 1.023052966894D , 0.896581648380D ,
      0.023478923136D , -.247951362613D , -.029842499869D ,
      0.027632152958D
      ]  drop
dc6las  @[ 1 ]  enter[
      0.021784700327D , 0.004936612372D , -.166863215412D ,
      -.068323121587D , 0.694457972958D , 1.113892783926D ,
      0.477904371333D , -.102724969862D , -.029783751299D ,
      0.063250562660D , 0.002499922093D , -.011031867509D
      ]  drop
dc7las  @[ 1 ]  enter[
      0.003792658534D , -.001481225915D , -.017870431651D ,
      0.043155452582D , 0.096014767936D , -.070078291222D ,
      0.024665659489D , 0.758162601964D , 1.085782709814D ,
      0.408183939725D , -.198056706807D , -.152463871896D ,
      0.005671342686D , 0.014521394762D
      ]  drop
dc8las  @[ 1 ]  enter[
      0.002672793393D , -.000428394300D , -.021145686528D ,
      0.005386388754D , 0.069490465911D , -.038493521263D ,
      -.073462508761D , 0.515398670374D , 1.099106630537D ,
      0.680745347190D , -.086653615406D , -.202648655286D ,
      0.010758611751D , 0.044823623042D , -.000766690896D ,
      -.004783458512D
      ]  drop
dc9las  @[ 1 ]  enter[
      0.001512487309D , -.000669141509D , -.014515578553D ,
      0.012528896242D , 0.087791251554D , -.025786445930D ,
      -.270893783503D , 0.049882830959D , 0.873048407349D ,
      1.015259790832D , 0.337658923602D , -.077172161097D ,
      0.000825140929D , 0.042744433602D , -.016303351226D ,
      -.018769396836D , 0.000876502539D , 0.001981193736D
      ]  drop
dc10las @[ 1 ]  enter[
      0.001089170447D , 0.000135245020D , -.012220642630D ,
      -.002072363923D , 0.064950924579D , 0.016418869426D ,
      -.225558972234D , -.100240215031D , 0.667071338154D ,
      1.088251530500D , 0.542813011213D , -.050256540092D ,
      -.045240772218D , 0.070703567550D , 0.008152816799D ,
      -.028786231926D , -.001137535314D , 0.006495728375D ,
      0.000080661204D , -.000649589896D
      ]  drop
dc1   @[ 1 ]  enter[
      1.D  2.D sqrt / , 1.D  2.D sqrt /         ]  drop
dc2   @[ 1 ]  enter[
      0.4829629131445341D ,  0.8365163037378079D ,
      0.2241438680420134D , -0.1294095225512604D ]  drop
dc3   @[ 1 ]  enter[
      0.3326705529500826D ,  0.8068915093110925D ,
      0.4598775021184915D , -0.1350110200102545D ,
     -0.0854412738820266D ,  0.0352262918857095D ] drop
      \ ALREADY IN DOUBLE PRECISION REAL
      \ But with calculation error
\     1.  1. 10. sqrt * + 1.  5. 2. 10. sqrt * + sqrt * + 16. 2. sqrt * / ,
```

```
\    5.  1. 10. sqrt * + 3.  5. 2. 10. sqrt * + sqrt * + 16. 2. sqrt * / ,
\    10. 2. 10. sqrt * - 2.  5. 2. 10. sqrt * + sqrt * + 16. 2. sqrt * / ,
\    10. 2. 10. sqrt * - 2.  5. 2. 10. sqrt * + sqrt * - 16. 2. sqrt * / ,
\    5.  1. 10. sqrt * + 3.  5. 2. 10. sqrt * + sqrt * - 16. 2. sqrt * / ,
\    1.  1. 10. sqrt * + 1.  5. 2. 10. sqrt * + sqrt * - 16. 2. sqrt * / ] drop
dc4  @[ 1 ]  enter[
     0.2303778133088964D , 0.7148465705529154D , 0.6308807679398587D ,
     -.0279837694168599D , -.1870348117190931D , 0.0308413818355607D ,
     0.0328830116668852D , -.0105974017850690D
     ] drop
dc5  @[ 1 ]  enter[
     0.1601023979741929D , 0.6038292697971895D , 0.7243085284377726D ,
     0.1384281459013203D , -.2422948870663823D , -.0322448695846381D ,
     0.0775714938400459D , -.0062414902127983D , -.0125807519990820D ,
     0.0033357252854738D
     ] drop
dc6  @[ 1 ]  enter[
     0.111540743350D ,  0.494623890398D ,  0.751133908021D ,
     0.315250351709D , -0.226264693965D , -0.129766867567D ,
     0.097501605587D ,  0.027522865530D , -0.031582039318D ,
     0.000553842201D ,  0.004777257511D , -0.001077301085D ]  drop
dc7  @[ 1 ]  enter[
     0.0778520540850037D , 0.3965393194818912D , 0.7291320908461957D ,
     0.4697822874051889D , -.1439060039285212D , -.2240361849938412D ,
     0.0713092192668272D , 0.0806126091510774D , -.0380299369350104D ,
     -.0165745416306655D , 0.0125509985560986D , 0.0004295779729214D ,
     -.0018016407040473D , 0.0003537137999745D
     ] drop
dc8  @[ 1 ]  enter[
     0.0544158422431072D , 0.3128715909143166D , 0.6756307362973195D ,
     0.5853546836542159D , -.0158291052563823D , -.2840155429615824D ,
     0.0004724845739124D , 0.1287474266204893D , -.0173693010018090D ,
     -.0440882539307971D , 0.0139810279174001D , 0.0087460940474065D ,
     -.0048703529934520D , -.0003917403733770D , 0.0006754494064506D ,
     -.0001174767841284D
     ] drop
dc9  @[ 1 ]  enter[
     0.0380779473638778D , 0.2438346746125858D , 0.6048231236900955D ,
     0.6572880780512736D , 0.1331973858249883D , -.2932737832791663D ,
     -.0968407832229492D , 0.1485407493381256D , 0.0307258147933385D ,
     -.0676328290613279D , 0.0002509471148340D , 0.0223616621236798D ,
     -.0047232047577518D , -.0042815036824635D , 0.0018476468830563D ,
     0.0002303857635232D , -.0002519631889427D , 0.0000393473203163D
     ] drop
dc10 @[ 1 ]  enter[
     0.026670057901D ,  0.188176800078D ,  0.527201188932D ,
     0.688459039454D ,  0.281172343661D , -0.249846424327D ,
    -0.195946274377D ,  0.127369340336D ,  0.093057364604D ,
    -0.071394147166D , -0.029457536822D ,  0.033212674059D ,
     0.003606553567D , -0.010733175483D ,  0.001395351747D ,
     0.001992405295D , -0.000685856695D , -0.000116466855D ,
     0.000093588670D , -0.000013264203D ]  drop
token  tmp.t  exp.mem> tmp.t
token  csa    exp.mem> csa
token  csb    exp.mem> csb
```

```
token  csp    exp.mem> csp
token  csq    exp.mem> csq
integer scalar sp.order
: make.csabpq.1
   tmp.t  dup  sub[ 1 , tmp.v.i ]  rev[ 1 ]  swap catenate  becomes> csp
  2              dp.real ramp   becomes> tmp.t
   tmp.t  @[ 1 ]  enter[
   0.5D , 0.5D
   ] drop
   tmp.t   becomes> csa
\   csa  becomes> csb
\   csp  becomes> csq
;
: bo11
  2  tmp.v.i  :=
  tmp.v.i        dp.real ramp   becomes> tmp.t
   tmp.t  @[ 1 ]  enter[
   0.5D , 0.5D
   ] drop
   tmp.t                                                 becomes> csp
   csp                                                   becomes> csa
;
: bo13
  3  tmp.v.i  :=
  tmp.v.i        dp.real ramp   becomes> tmp.t
   tmp.t  @[ 1 ]  enter[
   0.5D ,   1.D 16.D / ,  -1.D 16.D /
   ]  drop
  make.csabpq.1
;
: bo15
  5  tmp.v.i  :=
  tmp.v.i        dp.real ramp   becomes> tmp.t
   tmp.t  @[ 1 ]  enter[
   0.5D , 11.D 128.D / , -11.D 128.D / ,
   -3.D 256.D / , 3.D 256.D /
   ]  drop
  make.csabpq.1
;
: make.csabpq.2
   tmp.t  dup  sub[ 2 , tmp.v.i 1 - ]  rev[ 1 ]  swap catenate becomes> csp
  3              dp.real ramp   becomes> tmp.t
   tmp.t  @[ 1 ]  enter[
   0.25D ,  0.50D ,  0.25D
   ] drop
   tmp.t                                            becomes> csa
   csa  becomes> csb
   csp  becomes> csq
;
: bo22
  3  tmp.v.i  :=
  tmp.v.i        dp.real ramp   becomes> tmp.t
   tmp.t  @[ 1 ]  enter[
   0.75D , 0.25D , -0.125D
   ]  drop
```

```
  make.csabpq.2
;
: bo24
  5  tmp.v.i  :=
  tmp.v.i         dp.real ramp   becomes> tmp.t
   tmp.t   @[ 1 ]  enter[
   45.D 64.D / , 19.D 64.D / ,  -1.D 8.D / ,  -3.D 64.D / , 3.D 128.D /
   ]  drop
  make.csabpq.2
;
: bo26
  7  tmp.v.i  :=
  tmp.v.i         dp.real ramp   becomes> tmp.t
   tmp.t   @[ 1 ]  enter[
   175.D 256.D / , 81.D 256.D / , -123.D 1024.D / , -39.D 512.D / ,
   17.D 512.D / , 5.D 512.D / , -5.D 1024.D /
   ]  drop
   make.csabpq.2
;
: bo28
  9  tmp.v.i  :=
  tmp.v.i         dp.real ramp   becomes> tmp.t
   tmp.t   @[ 1 ]  enter[
   22050.D , 10718.D , -3796.D , -3126.D ,
   1228.D   ,  670.D   ,   -300.D  ,   -70.D  ,  35.D
   ]  drop
   make.csabpq.2
   csp  2.D -15.D ** *   becomes> csp
;
: make.csabpq.3
   tmp.t  dup  sub[ 1 , tmp.v.i ]  rev[ 1 ]  swap catenate becomes> csp
  4               dp.real ramp   becomes> tmp.t
   tmp.t   @[ 1 ]  enter[
   0.125D , 0.375D ,  0.375D , 0.125D
   ] drop
   tmp.t   becomes> csa
;
: bo31
  2  tmp.v.i  :=
  tmp.v.i         dp.real ramp   becomes> tmp.t
   tmp.t   @[ 1 ]  enter[
   .75D  , -.25D
   ]  drop
   make.csabpq.3
;
: bo33
  4  tmp.v.i  :=
  tmp.v.i         dp.real ramp   becomes> tmp.t
   tmp.t   @[ 1 ]  enter[
   45.D 64.D / , -7.D 64.D / , -9.D 64.D / , 3.D 64.D /
   ]  drop
   make.csabpq.3
;
: bo35
  6  tmp.v.i  :=
```

```
  tmp.v.i        dp.real ramp   becomes> tmp.t
   tmp.t  @[ 1 ]  enter[
   175.D 256.D / , -13.D 256.D / , -97.D 512.D / , 19.D 512.D / ,
   15.D 512.D / , -5.D 512.D /
   ]  drop
   make.csabpq.3
;
: bo37
  8  tmp.v.i  :=
  tmp.v.i        dp.real ramp   becomes> tmp.t
   tmp.t  @[ 1 ]  enter[
   11025.D , -307.D  , -3489.D ,   336.D ,
   865.D   , -195.D  ,  -105.D ,    35.D
   ]  drop
   make.csabpq.3
   csp  2.D -14.D ** *   becomes> csp
;
: bo39
  10  tmp.v.i  :=
  tmp.v.i        dp.real ramp   becomes> tmp.t
   tmp.t  @[ 1 ]  enter[
   87318.D , 190.D , -29676.D , 1140.D , 9188.D , -1308.D ,
   -1911.D , 469.D , 189.D , -63.D
   ]  drop
   make.csabpq.3
   csp  2.D -17.D ** *   becomes> csp
;
: so3 \ use odd number of coeff.
  4  sp.order  :=
  \ csa []size   2 /  tmp.v.i :=  drop
  20  tmp.v.i  :=
   tmp.v.i  1 +  dp.real ramp   becomes> tmp.t
   tmp.t  @[ 1 ]  enter[
   0.6315615124035756 ,
   0.2833241287791393 ,
  -0.1995539275818 ,
  -0.1647025826446927 ,
   0.0912758685480245 ,
   0.0894189178498617 ,
  -0.04696662484663471 ,
  -0.0480151020366 7939 ,
   0.02493689963969966 ,
   0.02572001001600068 ,
  -0.01330469960239009 ,
  -0.0137696808134998 ,
   0.007118265435884861 ,
   0.007370917988870163 ,
  -0.003809841711572108 ,
  -0.003945542141907374 ,
   0.002039281445696639 ,
   0.002111975662281219 ,
  -0.001091581045517322 ,
  -0.001130499753033917 ,
   0.0005843011868825511
   ]  drop
```

```
  tmp.t  dup  sub[ 2 , tmp.v.i ]  rev[ 1 ]  swap catenate becomes> csa

  20  tmp.v.i :=
   tmp.v.i  1 +  dp.real ramp   becomes> tmp.t
   tmp.t  @[ 1 ]  enter[
  -1.043261469941104 ,
   0.3312247945402051 ,
   0.5247423148960167 ,
  -0.2444969416039156 ,
  -0.2755917431211694 ,
   0.1391545679803643 ,
   0.1468596002088173 ,
  -0.07550189463168975 ,
  -0.07853017653475226 ,
   0.04053910511068559 ,
   0.04202575195946283 ,
  -0.02171503773409453 ,
  -0.02249434750478476 ,
   0.01162550347729064 ,
   0.01204063510565295 ,
  -0.006223133530293623 ,
  -0.006445098095866915 ,
   0.003331149664619865 ,
   0.003449932648653844 ,
  -0.001783102667159073 ,
  -0.001846681160105444
  ]  drop
  tmp.t  dup  sub[ 2 , tmp.v.i ]  rev[ 1 ]  swap  catenate becomes> csb

  2  tmp.v.i :=
   tmp.v.i  1 +  dp.real ramp   becomes> tmp.t
   tmp.t  @[ 1 ]  enter[
   0.5303300858899106 4 ,
   0.3535533905932737 62 ,
   0.0883883476483184 41
  ]  drop
  tmp.t  dup  sub[ 2 , tmp.v.i ]  rev[ 1 ]  swap  catenate becomes> csp

  5  tmp.v.i :=
   tmp.v.i  1 +  dp.real ramp   becomes> tmp.t
   tmp.t  @[ 1 ]  enter[
  -0.42552675939261878 ,
   0.32412568278496457 ,
  -0.138615376946886697 ,
   0.029410170437744053 ,
  -0.0021746339500776759 ,
   0.0000175373705651425 48
  ]  drop
  tmp.t  dup  sub[ 2 , tmp.v.i ]  rev[ 1 ]  swap  catenate becomes> csq
;
: cc1
  6  tmp.v.i :=
  tmp.v.i        dp.real ramp   becomes> tmp.t
   tmp.t  @[ 1 ]  enter[
   -.051429728471D ,  .238929728471D ,  .602859456942D ,  .272140543058D ,
```

```
   -.051429972847D , -.011070271529D
   ] drop
   tmp.t    becomes> csp
;
: cc2
  12 tmp.v.i :=
  tmp.v.i         dp.real ramp   becomes> tmp.t
   tmp.t  @[ 1 ]  enter[
    .011587596739D , -.029320137980D , -.047639590310D ,  .273021046535D ,
    .574682393857D ,  .294867193696D , -.054085607092D , -.042026480461D ,
    .016744410163D ,  .003967883613D , -.001289203356D , -.000509505399D
   ] drop
   tmp.t    becomes> csp
;
: cc3
  18 tmp.v.i :=
  tmp.v.i         dp.real ramp   becomes> tmp.t
   tmp.t  @[ 1 ]  enter[
   -.002682418671D ,  .005503126709D ,  .016583560479D , -.046507764479D ,
   -.043220763560D ,  .286503335274D ,  .561285256870D ,  .302983571773D ,
   -.050770140755D , -.058196250762D ,  .024434094321D ,  .011229240962D ,
   -.006369601011D , -.001820458916D ,  .000790205101D ,  .000329665174D ,
   -.000050192775D , -.000024465734D
   ] drop
   tmp.t    becomes> csp
;
: cc4
  24 tmp.v.i :=
  tmp.v.i         dp.real ramp   becomes> tmp.t
   tmp.t  @[ 1 ]  enter[
    .000630961046D , -.001152224852D , -.005194524026D ,  .011362459244D ,
    .018867235378D , -.057464234429D , -.039652648517D ,  .293667390895D ,
    .553126452562D ,  .307157326198D , -.047112738865D , -.068038127051D ,
    .027813640153D ,  .017735837438D , -.010756318517D , -.004001012886D ,
    .002652665946D ,  .000895594529D , -.000416500571D , -.000183829769D ,
    .000044080354D ,  .000022082857D , -.000002304942D , -.000001262175D
   ] drop
   tmp.t    becomes> csp
;
: cc5
  30 tmp.v.i :=
  tmp.v.i         dp.real ramp   becomes> tmp.t
   tmp.t  @[ 1 ]  enter[
   -.0001499638D ,    .0002535612D ,     .0015402457D ,    -.0029411108D ,
   -.0071637819D ,    .0165520664D ,     .0199178043D ,    -.0649972628D ,
   -.0368000736D ,    .2980923235D ,     .5475054294D ,     .3097068490D ,
   -.0438660508D ,   -.0746522389D ,     .0291958795D ,     .0231107770D ,
   -.0139736879D ,   -.0064800900D ,     .0047830014D ,     .0017206547D ,
   -.0011758222D ,   -.0004512270D ,     .0002137298D ,     .0000993776D ,
   -.0000292321D ,   -.0000150720D ,     .0000026408D ,     .0000014593D ,
   -.0000001184D ,   -.0000000673D
   ] drop
   tmp.t    becomes> csp
;
```

LLet-DnR.asy

勇伯程式庫 A01509-20070125-A060 ❖ January 25, 2007 ❖ [Page count: 3]

```
: LLet-DnR.asy ; \ %----- Asyst Prog

4 string PhiPsi.str
\ 0  to.stack.flag :=
\ 1  to.123.flag   :=
\ 8 string  tmp.s.u \ don't change the case of wlet -- don't use wlet "upper

: dec&rec
  \   512   n.wt1         :=
  \   2     jwp.level     :=
  \   8     i.ele         :=
  \   382   i.ele         :=       \ bo:  2^jwp.level --> 1 ,  i --> i+1 for B.O

  i.ele  2  jwp.level **  <=
  wlet tmp.s.u ":=
  tmp.s.u "upper "drop
  if
     tmp.s.u " -Phi"  "cat  123.ary.s  "[ 1 ] ":=
     " Phi-"  PhiPsi.str ":=
  else
     tmp.s.u " -Psi"  "cat  123.ary.s  "[ 1 ] ":=
     " Psi-"  PhiPsi.str ":=
  then
\   if
\      wlet " -Phi"  "cat  123.ary.s  "[ 1 ] ":=
\      " Phi-"  PhiPsi.str ":=
\   else
\      wlet " -Psi"  "cat  123.ary.s  "[ 1 ] ":=
\      " Psi-"  PhiPsi.str ":=
\   then
  \ ---------------------------------------- ■
  " Element: "  i.ele 0 n>s "cat
  \
  " , JWP level: "  jwp.level 0 n>s "cat
  "  (pt"  "cat 2 jwp.level ** 0 n>s  "cat " )"  "cat
  "cat
  " , Scale "  jwp.level 0 n>s "cat  " ->" "cat
  n.power 0 n>s "cat \ " )" "cat
  "cat
  123.ary.s  "[ 2 ]  ":=
  \ ---------------------------------------- ■

  2  gp.flag    :=
  \ 123.ary.s  "[ 1 ]  " -Reconstructed" "cat 123.ary.s  "[ 1 ]  ":=
  123.ary.tmp  becomes> 123.ary.n
  i.ele 2 <=
  if
    123.ary.n  @[ 1 ]  enter[
      jwp.level
    , n.power
    , i.ele
    , i.ele
```

```
    , swt.t [ i.ele     ]
    , swt.t [ i.ele 1 + ]
    , swt.t [ i.ele 2 + ]
    , swt.t [ i.ele 3 + ]
    ]  drop
  else
    123.ary.n  @[ 1 ]  enter[
      jwp.level
    , n.power
    , i.ele
    , i.ele 2 -
    , swt.t [ i.ele 2 - ]
    , swt.t [ i.ele 1 - ]
    , swt.t [ i.ele     ]
    , swt.t [ i.ele 1 + ]
    , swt.t [ i.ele 2 + ]
    , swt.t [ i.ele 3 + ]
    ]  drop
  then
  to.123
  1  gp.flag    :=

  get.p.pnts
  0.  b :=
  1. b [ i.ele ] :=
  b inverse.wt  drop
  1 gp.flag :=
  \ " n" tmp.s  ":=  swt.t  to.123
  swt.t  becomes>  123.ary.n  to.123
  swt.t    sub[ 1 , p.pnts ]
  gd up
  1 yap
  normal.coords
  i.ele  2  jwp.level **  <=
  if
     " h"     phi.or.psi  ":=
     " -PHI"
  else
     " s"     phi.or.psi  ":=
     " -PSI"
  then
  "  (From scale " "cat  jwp.level 0 n>s "cat  " -->" "cat
  n.power 0 n>s "cat  " )" "cat
\   \ ----------------------------------------
\   " Element: "  i.ele 0 n>s "cat
\   \
\   " , JWP level: "  jwp.level 0 n>s "cat
\   "  (pt"  "cat 2 jwp.level ** 0 n>s  "cat " )"  "cat
\   "cat
\   " , Scale "  jwp.level 0 n>s "cat  " ->" "cat
\   n.power 0 n>s "cat \ " )" "cat
\   "cat
\   123.ary.s  "[ 2 ]  ":=
\   \ ----------------------------------------
\   " From scale "  jwp.level 0 n>s "cat  " -->" "cat
```

```
\   n.power 0 n>s "cat \ " )" "cat
\   123.ary.s  "[ 2 ]  ":=
\   " Element: "  i.ele 0 n>s "cat
\   123.ary.s  "[ 3 ]  ":=
\   " JWP level: "  jwp.level 0 n>s "cat
\   "  ("  "cat 2 jwp.level ** 0 n>s  "cat " )"  "cat
\   123.ary.s  "[ 4 ]  ":=
  \ ----------------------------------------
\  wlet  1 4 "sub
  wlet  1 5 "sub
  wlet  5 1 "sub   " d"  "=
  wlet  3 1 "sub   " d"  "=  or
  if   "  (Dual)"  "cat  then
  "swap  "cat
  0.5  0.9   position  centered.label
  world.coords
  swt.t forward.wt  drop
  dp swt.t sub[ 1 , p.pnts ] 2 yap
  swt.t [ i.ele 1 - ]  -4  n>s  "    "  "cat
  swt.t [ i.ele     ]  -4  n>s  "cat   "   "  "cat
  swt.t [ i.ele 1 + ]  -4  n>s  "cat
  normal.coords
  0.5  0.9   position  centered.label
  " with center at position "
  i.ele
  0 n>s  "cat "  (" "cat wlet "cat " )" "cat
  0.5  0.85  position  centered.label
  1.   1.    position  world.coords

\   2  gp.flag    :=
\   123.ary.s  "[ 1 ]  " -m"  "cat   "upper 123.ary.s  "[ 1 ]  ":=
\   123.ary.tmp  becomes> 123.ary.n
\   123.ary.n  @[ 1 ]  enter[
\     jwp.level
\   , n.power
\   , swt.t [ i.ele 1 - ]
\   , swt.t [ i.ele     ]
\   , swt.t [ i.ele 1 + ]
\   , i.ele
\   ]  drop
\   to.123

  {hp.wnd} sc
  hp  file.ext  make.abc
  \   8 5 fix.format swt.t sub[ 11 , 3 ] . wlet s.type
  \   my.pause
;
```

LLet-Ent.asy


```
: LLet-Ent.asy ; \ %----- Asyst Prog

\  Entropy of Distribution
\  e.g.   array abs 300 entropy  \ L**1
\         array     300 entropy  \ relative value
\         array    -300 entropy  \ L**2
\
\

dp.real  scalar t1.max  scalar t1.min  scalar t1.div
integer scalar vi1  scalar vi2  scalar vi3 scalar ic
        scalar #.div.p   scalar #.ele   scalar #.div  scalar abs.flag

: entropy.prob \ [ array  -- entropy ]
  abs.flag 1 =
  if
    abs
  then
  becomes> t1
  #.div.p 0 <
  if
    #.div.p abs #.div.p :=
    t1 abs
    dup *
    becomes> t1
  then
  t1   []size     #.ele :=
       sort
  becomes> t1
  t1  [ #.ele ]  t1.max :=
  t1  [ 1     ]  t1.min :=
  t1.min 0.   >=
  if
    0.          t1.min :=  \ forced for absolute magnitude
  then

  real #.div.p ramp
  becomes> t2
  0.00000000001D  t2  :=   \ make it tiny for 0 provbability

  0   vi1 :=
  0   vi2 :=
  Begin
    vi2 1 +  vi2 :=
    t1.min   t1.max t1.min - #.div.p float / vi2 * +     t1.div :=
    0  ic :=
    begin
      vi1 1 +  vi1 :=
      ic   1 +  ic  :=

      t1   [ vi1 ]  t1.div  >=
      vi1  #.ele =  or
```

```
      ?dup
      if
        vi2  #.div.p <
        t1  [ vi1 ]  t1.max  =  and
        if
          #.div.p  vi2 :=
          1  ic :=
        then
        vi2  #.div.p =
        t1  [ vi1 ]  t1.max  =  or
        if
          begin
            vi1  #.ele  <
            if
              vi1 1 +  vi1 :=
              ic  1 +  ic  :=
            then
            vi1  #.ele  =
          until
          ic 1 +  ic :=
        then
        vi2 1 =
        if
          ic 1 -  t2  [ vi2 ]  :=
        else
          ic      t2  [ vi2 ]  :=
        then
      then
    until
    \ \ \ cr  vi2 n.type  ic .  vi1 .
    vi1  #.ele =
    vi1  #.ele float  #.div.p float /  vi2 float *  >=  or
    vi2  #.div.p = and
  Until

  t2 float  #.ele float /  becomes> t2  \ pdf
  t2 LN t2 * neg
  []sum
;

: entropy.coef \ [ array --- entropy ]
  abs.flag 1 =
  if
    abs
  then
  becomes> t1
  t1 []size #.ele := drop
  0  vi1  :=
  begin
    vi1 1 +  vi1 :=
    t1 [  vi1 ]  abs  0.  =
    if
      0.000000000001D  t1 [ vi1 ] :=
    then
    vi1 #.ele  =
```

```
  until
  t1 dup *  dup LN  *   neg  []sum
  t1 dup *  []sum      /
  t1 dup *  []sum  LN  +
  \  becomes> t3
;
: entropy \ [ array , #.div.p  -- ]
  #.div.p :=
  #.div.p  0  =
  if
    entropy.coef
  else
    entropy.prob
  then
;
: awp.t.ent
  #.div  :=
  jwp.level 0 =
  if
    \ awp.t xsect[ n.power 1 + , ! ] abs dup *    #.div.p  entropy
    awp.t xsect[ n.power 1 + , ! ]                #.div entropy
    0  tmp.v.i :=
    1  tmp.v.i.2 :=
  else
    jwp.level  1 - tmp.v.i :=
    0  tmp.v.i.2 :=
  then

  begin
    tmp.v.i 1 +  tmp.v.i :=
    \ awp.t xsect[ tmp.v.i , ! ] abs dup *    #.div.p  entropy
    awp.t xsect[ tmp.v.i , ! ]                #.div  entropy
    tmp.v.i.2 1 =
    if
      catenate
    then
    1 tmp.v.i.2 :=
    tmp.v.i  n.power =
  until
  norm.v                                  #.div  entropy
  catenate
  norm.v.s                                #.div  entropy
  catenate
  \ dup exp  \ theoretical dimension
;
: ent.bat
  dir.lee.asc  now.file "cat  ftoa.bat
  get.sub.a
  forward.wt
  dup  1 yap
  #.div       entropy
  becomes>   123.ary.n
  wlet  " -" "cat   #.div.p 0 n>s  "cat  123.ary.s  "[ 1 ]  ":=
  123.ary.s  "[ 1 ]  0.5  0.05   lab.x
  to.123
```

```
  hp file.ext make.abc
;
```

LLet-Lev.asy

勇伯程式庫 A01511-20070125-A062 ❖ January 25, 2007 ❖ [Page count: 4]

```
: LLet-Lev.asy ; \ For showing WB and WP at various levels/points \ -*- txt:asy -*-

: lee.auto
  isign  -1  =
  if
    n.wt1  dp.real ramp  becomes> b
    1  0 fix.format
    0.  b  :=
    \ cr ." JWP Level(" wlet s.type ." ):"  jwp.level  get.val   jwp.level :=
\ -----------------------------------------------------------------
    cr ." Element "
    wp.flag 1 =
    if
      " (WP: located at level "  jwp.level 0 n>s "cat " )" "cat s.type.3
    else
      " (WB: min. level at "  jwp.level 0 n>s "cat " )" "cat s.type.3
    then
    ."  (Phi<-"
    1  0 fix.format
    jwp.level  9 >=
    if                         4
    else
      jwp.level  6 >=
      if                       3
      else
        jwp.level  3 >=
        if                     2
        else                   1
        then
      then
    then
    dup  num.dig :=    0  fix.format
    2 jwp.level     ** n.type ." |->Psi:[WB"
    \ ---------------------------------------- ■
    wp.flag 1 =
    if
      ." <-"
      1  0 fix.format
      jwp.level  9 >=
      if                         4
      else
        jwp.level  6 >=
        if                       3
        else
          jwp.level  3 >=
          if                     2
          else                   1
          then
        then
      then
      dup  num.dig :=    0  fix.format
      2 jwp.level 1 + ** n.type ." |->WP]): "
```

```
    else
      ." ]): "
    then
\ ---------------------------------------------------------------
    3 0 fix.format
    \ i.ele get.val i.ele :=
    1 0 fix.format
    \ cr ." Clear (1) or Overlap:"     i.lin      get.val   i.lin      :=
    1.  b [ i.ele ] :=
    b  inverse.wt
    wp.flag 1 =
    if
      drop  swp.t
    else
      drop  swt.t
    then
  else
    {hp.wnd}  " Have you set B (in --> mode)?" s.type  \ my.pause
    cr ." Choose array for Forward Transform "
    " 0:default B, 1:WB.t, 2:WP.t" s.type.3
    0 get.val
    case
      1  of   swt.t  endof
      2  of   swp.t  endof
      0  of   b      endof
    endcase
    previous.window
    forward.wt      \ always d.t for WB and WP -- set Wp.flag
  then
  n.power  10 = if  sub[ 1 , 1000 ] then
  n.power   9 = if  sub[ 1 ,  500 ] then
  n.power   8 = if  sub[ 1 ,  250 ] then
  n.power   7 = if  sub[ 1 ,  120 ] then
  n.power   6 = if  sub[ 1 ,   70 ] then
  i.lin
  i.lin     1 =
  if  yap   lab.spe   else  ydp  then
  10 5 fix.format
;

dir.asy.sys       " 123io.sov" "cat       defer>  load.overlay
: set.123.auto
  {29.line}
  123.file.name
  dir.lee.asc "len  1 +   "drop
  "dup "len "drop  dir.lee.asc "len "drop -  "sub  123.file.name.1 ":=
  dir.asy.sys       " 123io.sov" "cat       defer>  load.overlay
  dir.lee.asc
  \ cr  ." 123 file 1 name:"  " 123.wks"    get.str
  cr  ." 123 file 1 name (close all wks files first!):"
  123.file.name.1
  \ \ for auto \ get.str
  "cat   123.file.name.1  ":=
  dir.lee.asc
  cr  ." 123 file 2 name:"  " 1234.wks"
```

```
  \ \ for auto \ get.str
  "cat   123.file.name.2  ":=
  cr  ." Group flag:"  gp.flag
  \ \ for auto \ get.val
  gp.flag :=
  cr  ." file.no:"     file.no
  \ \ for auto \ get.val
  file.no :=
  123.file.name.1      defer>  123file.create
  123.file.name.2      defer>  123file.create
  cr  ." Initial all WKS files (Yes:999)!!:" 0
  \ \ for auto \ get.val
  999 =
  if
    dir.lee.asc  " le"      "cat  " .wks" "cat  defer> 123file.create
    dir.lee.asc  " me"      "cat  " .wks" "cat  defer> 123file.create
    dir.lee.asc  " so"      "cat  " .wks" "cat  defer> 123file.create
    dir.lee.asc  " ona"     "cat  " .wks" "cat  defer> 123file.create
    dir.lee.asc  " ons"     "cat  " .wks" "cat  defer> 123file.create
    dir.lee.asc  " bo1"     "cat  " .wks" "cat  defer> 123file.create
    dir.lee.asc  " bo2"     "cat  " .wks" "cat  defer> 123file.create
    dir.lee.asc  " bo3"     "cat  " .wks" "cat  defer> 123file.create
    dir.lee.asc  " 1234"    "cat  " .wks" "cat  defer> 123file.create
    dir.lee.asc  " 123999"  "cat  " .wks" "cat  defer> 123file.create
  then
  123.file.name.1  123.file.name ":=
  release.overlay
;
release.overlay
: set.let.auto
  hp.menu.wnd \  {border}
  sc
  set.let.p1
\   2   0  goto.xy
\       ." Wavelet (" " on22a/s, bo220/d, so0/d , le, me" s.type ." ):"
\       b.t
\       wlet
\       \ \ for auto get.str
\       wlet ":=
\       b.t
\   60  0  goto.xy     2  0 fix.format
\       \ ." <- or -> :"              -1  get.val isign :=
\       ." <- or -> :"      isign get.val isign :=
\       1 0 fix.format
\   2   1  goto.xy
\       ." WP(1) / WB(0):"   wp.flag 1 = if " WP(1)" else " WB(0)"  then
\       s.type   wp.flag  get.val  wp.flag :=
\   30  1  goto.xy
\       ." JWP Level:"   jwp.level  get.val  jwp.level :=
\   60  1  goto.xy      4 0 fix.format
\       ." #.pnts:"           n.wt1  get.val  n.wt1 :=
\   2   2  goto.xy      2 0 fix.format
\       ." .Pcx: "                  file.ext  get.str file.ext ":=
\   30  2  goto.xy
\       ." 123->"            wlet " .wks" "cat  get.str tmp.s  ":=
```

```
\   60  2  goto.xy
\       ." File.#:"                  file.no   get.val file.no   :=
  2   0  goto.xy
      ." Wavelet (" " on22a/s, bo220/d, so0/d , le, me" s.type ." ):"
      b.t
      wlet         wlet ":=
      b.t
  60  0  goto.xy    2  0 fix.format
      \ ." <- or -> :"               -1  get.val isign :=
      ." <- or -> :"     isign         isign :=
      1 0 fix.format
  2   1  goto.xy
      ." WP(1) / WB(0):"   wp.flag 1 = if " WP(1)" else " WB(0)"  then
      s.type   wp.flag            wp.flag :=
  30  1  goto.xy
      ." JWP Level:"   jwp.level            jwp.level :=
  60  1  goto.xy     4  0 fix.format
      ." #.pnts:"          n.wt1            n.wt1 :=
  2   2  goto.xy     2  0 fix.format
      ." .Pcx: "                   file.ext           file.ext ":=
  30  2  goto.xy
      ." 123->"            wlet " .wks" "cat          tmp.s  ":=
  60  2  goto.xy
      ." File.#:"                  file.no            file.no   :=
  \ ---------------------------------------- -
  \ wlet " .wks" "cat  get.str tmp.s  ":=
  \ ---------------------------------------- -
  dir.lee.asc   tmp.s  "cat  123.file.name  ":=
  isign 1 =
  if
    \ n.wt1  dp.real ramp  becomes> b
    \ 0   b   :=
    cr
    " Has B been set?!"  s.type
  then
\ 123.file.name     defer>  123file.create
\ release.overlay
\ previous.window
;
```

LLet-Mai.asy

勇伯程式庫 A01512-20070125-A063 ❖ January 25, 2007 ❖ [Page count: 24]

```
: LLet-Mai.asy ; \ %----- Asyst Prog
\ -----------------------------------------------------------------
\  1    Wavelet transform (s.o, o.n)
\  2    Wavelet packet transform (o.n)
\  3.   Both forward and backward
\  4.   Implementing both periodic and aperiodic WT
\  5.   The periodic WT is reversable, i.e., allows perfect reconstruction
\  6.   The aperiodic WT is optimized to reduce edge effects, its reconstruction
\       may show differences from original data
\  7.   keywords:  indexing,  wrapping,  domains
\  8.   The time used is much less than NR book
\
\  Written : Y.R. Lee
\ -----------------------------------------------------------------

\ integer scalar n.fft.pts
\         scalar  n.wt1
\         scalar  isign
integer scalar n.power
        scalar aper.flag
        scalar abs.flag
        scalar icd
        scalar icu
        scalar jwp   scalar kwp   scalar mwp
        scalar jwp.level
        scalar wp.flag
        scalar jwp_m
        scalar i.ele
        scalar a0.i scalar b0.i scalar p0.i scalar q0.i
integer scalar  in
        scalar  im
        scalar  ik
        scalar  no2
        scalar  nn
        scalar  domain.flag
\ integer scalar  c.size

real    scalar var.frac
real scalar norm.keep.p.best
12 string now.file
token awp.t  exp.mem> awp.t
token uwp.t  exp.mem> uwp.t
token vwp.t  exp.mem> vwp.t
token swp.t  exp.mem> swp.t
token swt.t  exp.mem> swt.t
token bwp.t  exp.mem> bwp.t \ <--> b
26 0   26 79  window {tmp.wnd}
previous.window
: my.power
  trunc
  0  n.power  :=
  begin
```

```
    2  /
    dup
    dup  1  >=  if   n.power   1 +   n.power :=  then
    1  =
  until
  drop
;
token  a1.t  exp.mem>  a1.t
token  a2.t  exp.mem>  a2.t
token  a3.t  exp.mem>  a3.t
token  at.t  exp.mem>  at.t
token  rt.t  exp.mem>  rt.t
token  wt.t  exp.mem>  wt.t
token  ct.t  exp.mem>  ct.t
token  s.t   exp.mem>  s.t
token  ht.t  exp.mem>  ht.t
token  hf.t  exp.mem>  hf.t
token  gt.t  exp.mem>  gt.t
token  gf.t  exp.mem>  gf.t
token   u.t  exp.mem>  u.t
token   v.t  exp.mem>  v.t
token   d.t  exp.mem>  d.t
token  a.coef  exp.mem> a.coef
token  b.coef  exp.mem> b.coef
token  p.coef  exp.mem> p.coef
token  q.coef  exp.mem> q.coef
token p.token exp.mem> p.token    token q.token exp.mem> q.token
token a.token exp.mem> a.token    token b.token exp.mem> b.token
1  string phi.or.psi
" s" phi.or.psi  ":=

.975 norm.keep.p.best  :=

\ =================================================================
: get.0.i.phi.or.psi
\ i.ele  2 jwp.level ** <=
\ if
\   phi.or.psi  " h" ":=
\ else
\   phi.or.psi  " s" ":=
\ then
  phi.or.psi  " h"   "=
  \ i.ele  2 jwp.level ** <=
  if
    wlet 1 2 "sub  " on"  "=
    if
    \       p0.i 4 +   p0.i :=
    \       q0.i 4 +   q0.i :=
    \       a0.i 4 -   a0.i :=
    \       b0.i 4 -   b0.i :=

    \  p0.i 3 +   p0.i :=
    \  q0.i 3 +   q0.i :=
    \  a0.i 3 -   a0.i :=
    \  b0.i 3 -   b0.i :=
```

```
    else
    wlet 1 3 "sub  " bo3"  "=
    if
       p0.i 3 +   p0.i :=
       q0.i 3 +   q0.i :=
       a0.i 3 -   a0.i :=
       b0.i 3 -   b0.i :=
    then
    then
  then
;
\ -----------------------------------------------------------------
: get.0.i.bo
  wlet  3 1 "sub  " 2"  "=     \ odd number of coef.
  if
     1  p0.i :=
     0  q0.i :=
    -1  a0.i :=
     0  b0.i :=
  else
     \ (bo13 ---> 2 0 -1 1 )  (0 -2 1 3 ) ( 2 0 -1 3 ) (1 -1 0 2)
     2  p0.i :=
    -0  q0.i :=
    -1  a0.i :=
     1  b0.i :=
  then
  get.0.i.phi.or.psi
;
\ -----------------------------------------------------------------
: get.0.i.on.c \ Coiflet
   c.size 2 /   2  modulo       0 =
   c.size 2 /   2  modulo       1 = or
   if
     p.size 2  /    tmp.v.i   :=     \ For O.N.  (all equal)
       0            neg  tmp.v.i +    p0.i  :=
       q.size 2 -   neg  tmp.v.i +    q0.i  :=    \ q(n)=-1**n * p(-n+1)
       a.size 1 -   neg  tmp.v.i +    a0.i  :=
       1            neg  tmp.v.i +    b0.i  :=
   \  (       4 --->  2 0 -1 1)  (6 ---> 3 -1 -2 2) ........
   \  (  n.dau.c -->  p q  a b)
   then
   get.0.i.phi.or.psi
;
\ -----------------------------------------------------------------
: get.0.i.on
   c.size 2 /   2  modulo       0 =
   c.size 2 /   2  modulo       1 = or
   if
     p.size 2  /    tmp.v.i   :=     \ For O.N.  (all equal)
       0            neg  tmp.v.i +    p0.i  :=
       q.size 2 -   neg  tmp.v.i +    q0.i  :=    \ q(n)=-1**n * p(-n+1)
       a.size 1 -   neg  tmp.v.i +    a0.i  :=
       1            neg  tmp.v.i +    b0.i  :=
   \  (       4 --->  2 0 -1 1)  (6 ---> 3 -1 -2 2) ........
   \  (  n.dau.c -->  p q  a b)
```

```
   then
   get.0.i.phi.or.psi

;
\ -----------------------------------------------------------------
: get.0.i.so \ so3
     0  p0.i  :=
     3  q0.i  :=
     0  a0.i  :=
    -3  b0.i  :=
  get.0.i.phi.or.psi
;
\ -----------------------------------------------------------------
: get.0.i.le&me
  get.0.i.on
;
\ =================================================================
dp.real scalar abpq.con
real
: get.abpq.size
  csp  []size   p.size :=  drop
  csq  []size   q.size :=  drop
  csa  []size   a.size :=  drop
  csb  []size   b.size :=  drop
;
\ -----------------------------------------------------------------
: get.abpq
  wlet  1 2 "sub  " on"  "=
  wlet  1 2 "sub  " le"  "=  or
  wlet  1 2 "sub  " me"  "=  or
  if
    wlet  1 2 "sub  " le"  "=
    wlet  1 2 "sub  " me"  "=  or
    wlet  5 1 "sub  " a"   "=  or    \ asymmetric  (on22--00a)
    if
      1.D             abpq.con  :=
    else
      2.D  sqrt  inv  abpq.con  :=   \ lease asymmetric (on44--00s)
    then
    wlet  5 1 "sub  " c"   "=        \ Coiflet     (on11--55c)
    if
      2.D  sqrt       abpq.con  :=
    then
  else
      2.D  sqrt       abpq.con  :=
  then

  csp  abpq.con  *  \ reconstrution
    []size  c.size :=     becomes>  p.coef
    real n.wt1 c.size  -  ramp  becomes> u.t
    0.  u.t  :=
    p.coef  u.t  catenate
      c.size  2. /  floor neg  rotate
      p0.i                     rotate
      becomes> p.coef
```

```
  csq  abpq.con  *
    []size  c.size :=      becomes>  q.coef
    real n.wt1 c.size  -  ramp  becomes> u.t
    0.  u.t  :=
    q.coef  u.t catenate
      c.size  2. /  floor neg  rotate
      q0.i                     rotate
      becomes> q.coef
  csa  abpq.con  *
    []size  c.size :=      becomes>  a.coef
    real n.wt1 c.size  -  ramp  becomes> u.t
    0.  u.t  :=
    a.coef  u.t  catenate
      c.size  2. /  floor neg  rotate
      a0.i                     rotate
      becomes> a.coef
  csb  abpq.con  *
    []size  c.size :=      becomes>  b.coef
    real n.wt1 c.size  -  ramp  becomes> u.t
    0.  u.t  :=
    b.coef  u.t catenate
      c.size  2. /  floor neg  rotate
      b0.i                     rotate
      becomes> b.coef
;
\ -----------------------------------------------
: get.h&g
  isign  -1  =
  if
    p.coef  q.coef
  else
    a.coef  b.coef
  then
  becomes> gt.t      becomes> ht.t
;
\ =================================================================
: get.h&g.bo
  wlet   3 1  "sub   wlet  4 1  "sub  "cat   tmp.s ":=
  tmp.s  " 11"  "= if       bo11   then
  tmp.s  " 13"  "= if       bo13  then
  tmp.s  " 15"  "= if       bo15   then
  tmp.s  " 22"  "= if       bo22   then
  tmp.s  " 24"  "= if       bo24   then
  tmp.s  " 26"  "= if       bo26   then
  tmp.s  " 28"  "= if       bo28   then
  tmp.s  " 31"  "= if       bo31   then
  tmp.s  " 33"  "= if       bo33   then
  tmp.s  " 35"  "= if       bo35   then
  tmp.s  " 37"  "= if       bo37   then
  tmp.s  " 39"  "= if       bo39   then
  csp       rev[ 1 ]      becomes>  csb    \ b.o
      csb    []size c.size := drop
      c.size 1 +  1
      do
        csb    [ i ]    -1 i 1 - **   *    csb    [ i ] :=
```

```
    loop
  csa       rev[ 1 ]      becomes>  csq    \ b.o
    csq    []size c.size := drop
    c.size 1 +  1
    do
      csq    [ i ]    -1 i 1 - **    *    csq    [ i ] :=
    loop
  csp      becomes>   p.token     \  B.O. switch
  csq      becomes>   q.token
  csa      becomes>   a.token
  csb      becomes>   b.token
  wlet  5 1  "sub  " d"  "=
  if
    p.token  becomes>   csa         \ dual
    q.token  becomes>   csb
    a.token  becomes>   csp
    b.token  becomes>   csq
  else
    p.token  becomes>   csp
    q.token  becomes>   csq
    a.token  becomes>   csa
    b.token  becomes>   csb
  then
  wlet 3 1 "sub   " 1"  "=
  if
    csp      becomes> csp
    csq      becomes> csq
    csa      becomes> csa
    csb neg  becomes> csb
  then
  wlet 3 1 "sub   " 3"  "=
  if
    csp      becomes> csp
    csq neg  becomes> csq
    csa      becomes> csa
    csb      becomes> csb
  then
  wlet 3 1 "sub   " 2"  "=
  if
    wlet 5 1 "sub   " d"  "=
    if
      csq      becomes> csq
      csb neg  becomes> csb
    else
      csq neg  becomes> csq
      csb      becomes> csb
    then
  then
  get.abpq.size
  get.0.i.bo
  get.abpq
;
\ ----------------------------------------------
: get.h&g.so
  so3
```

```
  csp  []size   p.size :=  drop
  csq  []size   q.size :=  drop
  csa  []size   a.size :=  drop
  csb  []size   b.size :=  drop

  csp      becomes>   p.token     \  B.O. switch
  csq      becomes>   q.token
  csa      becomes>   a.token
  csb      becomes>   b.token
  wlet  3 1  "sub  " d"   "=
  if
    p.token  becomes>   csa          \ dual
    q.token  becomes>   csb
    a.token  becomes>   csp
    b.token  becomes>   csq
  else
    p.token  becomes>   csp
    q.token  becomes>   csq
    a.token  becomes>   csa
    b.token  becomes>   csb
  then
  get.abpq.size
  get.0.i.so
  get.abpq
;
\ -----------------------------------------------
: get.h&g.on.c \ Coiflet
  wlet   3 1  "sub   wlet  5 1  "sub  "cat   tmp.s ":=
  tmp.s  " 1c"  "= if        cc1     then
  tmp.s  " 2c"  "= if        cc2     then
  tmp.s  " 3c"  "= if        cc3     then
  tmp.s  " 4c"  "= if        cc4     then
  tmp.s  " 5c"  "= if        cc5     then
  csp
  []size  c.size :=     becomes>  csp
  c.size  p.size :=
  c.size  q.size :=
  c.size  a.size :=
  c.size  b.size :=

    csp     rev[ 1 ]      becomes>  csq
    c.size 1 +  1
    do
      csq    [ i ]    -1 i 1 - **   *    csq    [ i ] :=
    loop
    csp     rev[ 1 ] conj becomes>  csa
    csq     rev[ 1 ] conj becomes>  csb

    csp      becomes> csp
    csq neg  becomes> csq
    csa      becomes> csa
    csb neg  becomes> csb

  csp      becomes>   p.token
  csq      becomes>   q.token
```

```
  csa       becomes>   a.token
  csb       becomes>   b.token
  wlet  5 1  "sub  " s"   "=   \
  if
    p.token             becomes>   csa         \ dual
    q.token             becomes>   csb
    a.token             becomes>   csp
    b.token             becomes>   csq
  then
  get.abpq.size
  get.0.i.on.c
  get.abpq
;
\ =================================================================
\ ------------------------------------------
: get.h&g.on
  wlet   3 1  "sub   wlet  5 1  "sub  "cat   tmp.s ":=
  tmp.s  " 1a"  "= if       dc1    then
  tmp.s  " 2a"  "= if       dc2    then
  tmp.s  " 3a"  "= if       dc3    then
  tmp.s  " 4a"  "= if       dc4    then
  tmp.s  " 5a"  "= if       dc5    then
  tmp.s  " 6a"  "= if       dc6    then
  tmp.s  " 7a"  "= if       dc7    then
  tmp.s  " 8a"  "= if       dc8    then
  tmp.s  " 9a"  "= if       dc9    then
  tmp.s  " 0a"  "= if       dc10   then
  tmp.s  " 4s"  "= if       dc4las    then
  tmp.s  " 5s"  "= if       dc5las    then
  tmp.s  " 6s"  "= if       dc6las    then
  tmp.s  " 7s"  "= if       dc7las    then
  tmp.s  " 8s"  "= if       dc8las    then
  tmp.s  " 9s"  "= if       dc9las    then
  tmp.s  " 0s"  "= if       dc10las   then

  []size  c.size :=     becomes>  csp
  c.size  p.size :=
  c.size  q.size :=
  c.size  a.size :=
  c.size  b.size :=

    csp      rev[ 1 ]       becomes>  csq
    c.size 1 +  1
    do
      csq    [ i ]    -1 i 1 - **   *    csq    [ i ] :=
    loop
    csp      rev[ 1 ] conj becomes>  csa
    csq      rev[ 1 ] conj becomes>  csb

  csp       becomes>   p.token      \  B.O. switch
  csq       becomes>   q.token
  csa       becomes>   a.token
  csb       becomes>   b.token
  wlet  5 1  "sub  " s"   "=   \ least asy (las)
  if
```

```
    p.token             becomes>   csa          \ dual
    q.token             becomes>   csb
    a.token             becomes>   csp
    b.token             becomes>   csq
  then
  get.abpq.size
  get.0.i.on
  get.abpq
;
\ =================================================================
real scalar var
     scalar mea
token x.t  exp.mem> x.t
token t.t  exp.mem> t.t
: get.h&g.gaussian
  isign  :=
\  n.wt1 2. /  0.5 +  mea :=
\  n.wt1 var.frac /  var :=    \ var.frac ~ 10.
\  dp.real  n.wt1  ramp
  100   2. /  0.5 +  mea :=
  100   var.frac /  var :=    \ var.frac ~ 10.
  dp.real  100  ramp

  becomes> x.t

  x.t mea -  var /            \ Normalizing here
\  Don't do normalizing
\  mea  1 -  /                \ Normalizing x
  1 *  becomes> t.t
  t.t dup *  neg
  2.  /
  exp
  2.  pi *  sqrt  /
  becomes> x.t

\   x.t []sum  tmp.v.9 :=
\   x.t tmp.v.9  /
\   becomes> x.t

  x.t
\ hanning.window.apply
       dup []sum   /
  becomes> x.t

  x.t
  []size 2 /     rotate
  becomes> ht.t      \
  x.t  t.t *  neg
\  tmp.v.9 /
  []size 2 /     rotate
  becomes> gt.t      \

\  gt.t
\  arrange.h&g
  isign  -1  =
```

```
  if
    ht.t
    c.size 2 /   2 -  neg  rotate
    c.size 2 /   rotate
    becomes> ht.t
    gt.t
    c.size 2 /   2 -  neg  rotate
    c.size 2 /   rotate
    becomes> gt.t
  then
;
\ =================================================================
: get.abpq.le&me
    p.coef \  u.t  catenate
      c.size  2 /         neg  rotate
      p0.i                     rotate
      becomes> p.coef
    q.coef \  u.t catenate
      c.size  2 /         neg  rotate
      q0.i                     rotate
      becomes> q.coef
    a.coef \  u.t  catenate
      c.size  2 /         neg  rotate
      a0.i                     rotate
      becomes> a.coef
    b.coef \  u.t catenate
      c.size  2 /         neg  rotate
      b0.i                     rotate
      becomes> b.coef
;
\ ----------------------------------------------
: abpq.from.freq
  hf.t  ifft  zreal
  rev[ 1 ]              \ important
  becomes>  p.coef
    p.coef  []size  c.size :=  drop
    p.coef  rev[ 1 ]      becomes>  q.coef
    c.size 1 +  1
    do
      q.coef [ i ]    -1 i 1 - **   *    q.coef [ i ] :=
    loop
    p.coef  rev[ 1 ] conj becomes>  a.coef
    q.coef  rev[ 1 ] conj becomes>  b.coef

  p.coef                becomes>   csa         \ dual
  q.coef                becomes>   csb
  a.coef                becomes>   csp
  b.coef                becomes>   csq
  get.abpq.size
  get.0.i.le&me
  get.abpq.le&me
;
\ =================================================================
dp.real scalar tmp.v
dp.real scalar tmp.v.2
```

```
: nu.v
    tmp.v  abs   3.D *  pi / 2.D /  1.D -
    tmp.v.2  :=
    tmp.v.2  0  <=
    if
      0.D
    else
      tmp.v.2   0.  >
      tmp.v.2   1  <=   and
      if
        tmp.v.2  4 **
        35.D   84.D  tmp.v.2 * -
        70.D tmp.v.2  2 ** * +  20.D tmp.v.2 3 ** * -
        *
      else
        1.D
      then
    then
;
: phi.cap.v
  tmp.v :=
    tmp.v  abs   2.D  pi * 3.D /   <=
    if
      2.D  pi *  sqrt  inv
    else
      tmp.v  abs   2.D  pi * 3.D /   >
      tmp.v  abs   4.D  pi * 3.D /   <=   and
      if
        2.D  pi *  sqrt  inv
        pi   2.d /  nu.v  *
        cos  *
      else
        0.D
      then
    then
;
\ -----------------------------------------------
: get.h&g.me
  rad
  dp.real  n.wt1   ramp
  n.wt1  c.size :=
  2.  Pi  *   *  n.wt1  /       \  2pi/N , 2pi*2/N,   , 2pi*N/n.fft.pts

  becomes> u.t
  0  tmp.v.i :=
  begin
    tmp.v.i 1 +  tmp.v.i :=
    u.t [ tmp.v.i ]  0.D               +     2.D *   phi.cap.v
    u.t [ tmp.v.i ]  -1  2.D  pi * *   +     2.D *   phi.cap.v
    +  2  pi *  sqrt *
    2.D   sqrt  *
    u.t [ tmp.v.i ] :=
    tmp.v.i  n.wt1 =
  until
  u.t
```

```
  becomes> hf.t
  abpq.from.freq
;
\ =================================================================
: compu.le
  dup
  2.  /  sin   2. **            becomes> u.t
         sin   2. **            becomes> v.t
  u.t     1  *                  becomes> u.t
  v.t     1  *                  becomes> v.t
  315.  420. u.t * -    126. u.t 2. **  *  +  4.  u.t 3. **  * -
  315.  420. v.t * -    126. v.t 2. **  *  +  4.  v.t 3. **  * -
  /
  2.  1. u.t -  4. **   *
  *
  z=x+i0   sqrt
;
: get.h&g.Le
  rad
  dp.real  n.wt1   ramp
  n.wt1  c.size :=
  2.  Pi  *   *  n.wt1  /       \  2pi/N , 2pi*2/N,   , 2pi*N/n.fft.pts
  compu.le
  becomes> hf.t

  \  real  n.wt1   ramp
  \  2.  Pi  *   *  n.wt1  /          \  2pi/N , 2pi*2/N,   , 2pi*N/n.fft.pts
  \  Pi  +
  \  compu.le
  \  conj
  \  real  n.wt1   ramp
  \  2.  Pi  *   *  n.wt1  /   neg
  \  dup   cos    swap    sin    z=x+iy
  \  *
  \  becomes> gf.t
  abpq.from.freq
;
\ =================================================================
: per.conv \  [ s , r --  (s*r <-> ) ]  \ Implementing periodic convolution
           \    n   m                   \
    fft
    swap
    fft
    *
    ifft zreal
;
: aper.conv \ [ s , r --  (s*r <-> ) ]  \ Implementing periodic convolution
            \   n   m                   \
  []size  im  :=    becomes>  a2.t        \ response function in length of m

  []size  in  :=    becomes>  a1.t
  domain.flag  0 =
  if                                      \ From Time domain
    in  im   >
    if
```

```
      im   2  modulo   0  =
      if                                    \ Make it into odd
        a2.t   0.   catenate   becomes>  a2.t
      then
    then
    a2.t  []size  im  :=  drop
\    a1.t              becomes>  a3.t        \ Response function in length of n
\
\    0.   a3.t  :=                           \ Padding and wrapping
\    a2.t   a3.t  sub[ 1 , im ]  :=

\    isign 1 =                               \ No need to wrap for inverse WT
\    in   im >  and                          \ Need to wrap for forward WT
\    if
\           a3.t  im 1 -  2 /  neg  rotate  becomes> a3.t
\    then
\    a3.t  1 *  becomes> a3.t
  then
  a1.t  a2.t  rev[ 1 ]   conv.aper
\  sub[   1    ,  in  ]
\  a1.t  a2.t      conv.aper
  sub[ im  2 /  1 +  ,  in  ]
;
\ =================================================================
: assign.wlet
  wlet  2  "left  " me"    "=    if   get.h&g.me  then  \ o.n.
  wlet  2  "left  " le"    "=    if   get.h&g.le  then  \ o.n.
\ wlet  2  "left  " on"    "=    if   get.h&g.on  then
  wlet  2     "left   " on"    "=
  wlet  5  1  "sub    " a"     "=
  wlet  5  1  "sub    " s"     "=  or   and
                                 if   get.h&g.on  then
  wlet  2     "left   " on"    "=
  wlet  5  1  "sub    " c"     "=  and
                                 if   get.h&g.on.c  then
  wlet  2  "left  " bo"    "=    if   get.h&g.bo  then
  wlet  2  "left  " so"    "=    if   get.h&g.so  then
;
\ =================================================================
: make.awp.t \ [ at.t -- ]
  n.wt1  my.power
  10  n.power >=
  if
    real
  else
    dp.real
  then
  n.wt1  ramp
  dup
  1  tmp.v.i  :=
  begin
    tmp.v.i 1 +  tmp.v.i  :=
    dup
    rot
    laminate
```

```
    swap
    \  tmp.v.i  n.power =
    tmp.v.i  n.power 1 + =
  until
  drop
  becomes> awp.t
;
\ -----------------------------------------------
: make.at.t.INV \ [ u.t -- ]
  jwp  n.power  <
  if
    0  tmp.v.i  :=
    begin
      tmp.v.i  1 +   tmp.v.i  :=
      dup  catenate
      tmp.v.i  kwp 1 +  =
    until
    becomes> at.t
  else
    becomes> at.t
  then
;
\ -----------------------------------------------
: get.jwp_m
  jwp        0  =
  if     n.power 1 +    jwp_m :=
  else   jwp            jwp_m :=
  then
;
\ -----------------------------------------------
: get.swp.t \ inverse wavelet packet transform
  0  at.t :=            \  Combine the top level of each WP block
  n.power  jwp :=       \    into a {S D1 D2 ....Dj} array
  begin
    jwp 1 -   jwp :=
    get.jwp_m
    awp.t  xsect[ jwp_m , ! ] sub[ 2  jwp ** 1 +  , 2 jwp ** ]
    at.t                     sub[ 2  jwp ** 1 +  , 2 jwp ** ] :=
    jwp  jwp.level 1 +  =
  until

  jwp.level  0  =
  if
    awp.t  xsect[ jwp_m     , ! ] sub[ 1 , 2  jwp.level 1 + ** ]
  else
    awp.t  xsect[ jwp.level , ! ] sub[ 1 , 2  jwp.level 1 + ** ]
  then
  at.t                              sub[ 1 , 2  jwp.level 1 + ** ] :=
  at.t  becomes>  bwp.t

  jwp.level          jwp :=
  jwp 1 -            nn  :=
  n.power 1 - jwp -  kwp :=
  at.t   sub[ 1              , 2  jwp **    ]    becomes>  s.t
  at.t   sub[ 2  jwp ** 1 +                 ]    becomes>  d.t
```

```
  begin
    jwp 1 +  jwp :=
    nn  1 +  nn  :=
    kwp 1 -  kwp :=
    get.jwp_m
    {tmp.wnd}
    2 0 fix.format
    cr ."        ===> INV WP Scale Form (" jwp.level . ."  --->"  n.power
       ." ) "  ." J = "  jwp .
    4 0 fix.format

    s.t  dup    catenate    dup  becomes>  u.t    \ set proper dim
                                 becomes>  s.t

    0.  u.t   :=    \ upsampling
    s.t  sub[  1 ,  2 nn ** ]    u.t  sub[ 1 , 2 nn ** , 2 ]  :=
         u.t   make.at.t.inv
         at.t
         ht.t  per.conv              \   for inverse WT
         sub[  1     ,   2 nn  1 + **  ]

    0.  u.t   :=
    d.t  sub[  1 ,  2 nn ** ]    u.t  sub[ 1 , 2 nn ** , 2 ]  :=
         u.t   make.at.t.inv
         at.t
         gt.t  per.conv
         sub[  1     ,   2 nn  1 + **  ]

    +   becomes> s.t
    \ --------------------------------------------
    jwp  n.power  <
    if
      d.t  sub[ 2  nn **  1 +  ]    becomes>   d.t
    then
    jwp  n.power   =
  until
  s.t
  \ dup
  awp.t sub[ n.power , 1 ; 1 , 2 n.power ** ]  :=
;
\ ================================================================
token norm.l  exp.mem> norm.l
token norm.v  exp.mem> norm.v
real    scalar  norm.keep.p
dp.real scalar  norm.keep.sum.v
token norm.v.s  exp.mem> norm.v.s  \ best level (single level)
integer scalar ic.least
integer scalar best.level

\ 80.  100. /   norm.keep.p :=
: norm.wp.best
  norm.keep.p.best  norm.keep.p :=
  n.wt1  my.power
  \ ----------------------- best level -------------------------
```

```
jwp.level  best.level  :=
jwp.level    1 -   jwp :=
n.power 1 - jwp -  kwp :=
get.jwp_m
begin
  jwp 1 +  jwp :=
  kwp 1 -  kwp :=
  get.jwp_m
     awp.t  xsect[ jwp_m  , ! ]
       dup *  sort   rev[  1 ]    ^sum
       dup [ 2 n.power ** ] norm.keep.p *    norm.keep.sum.v :=
       0  tmp.v.i :=
       begin
         tmp.v.i 1 +  tmp.v.i :=
         dup  [ tmp.v.i ]           norm.keep.sum.v  >
         tmp.v.i                    2 n.power **     =
         or
      until
      drop
      tmp.v.i       icd :=
      jwp  jwp.level =
      if
        icd ic.least :=
      then

    awp.t  xsect[ jwp 1 +     , ! ]
      dup *  sort   rev[  1 ]    ^sum
      0  tmp.v.i :=
      begin
        tmp.v.i 1 +  tmp.v.i :=
        dup  [ tmp.v.i ]           norm.keep.sum.v  >
        tmp.v.i                    2 n.power **     =
        or
      until
      drop
      tmp.v.i       icu :=

    icu  icd <
    icu  ic.least < and
    if
      icu   ic.least  :=
      cr " Find New Level at "  jwp 1 +  0 n>s "cat
      "type
      "    " icu 0 n>s  "cat  " <" "cat  icd 0 n>s "cat  "type
      jwp 1 +   best.level  :=
    else
    then
  jwp  n.power 1 - =
until
best.level 0 =
if
  awp.t  xsect[ n.power 1 + , ! ]   becomes> norm.v.s
else
  awp.t  xsect[ best.level , ! ]   becomes> norm.v.s
then
```

```
\ ----------------------- best basis -------------------------
n.wt1  integer ramp  becomes> norm.l
n.wt1  dp.real ramp  becomes> norm.v
jwp.level          jwp :=
n.power 1 - jwp -  kwp :=
get.jwp_m
jwp.level                                  norm.l :=
awp.t  xsect[ jwp_m  , ! ]   becomes>  norm.v
\ awp.t  xsect[ jwp_m  , ! ]   norm.v :=

1 to.123.999.flag =
if
  wlet  n.power 0 n>s  "cat " -->" "cat  jwp.level 0 n>s "cat  "  / p=" "cat
  norm.keep.p -3 n>s "cat
  1 row.no :=
  to.123.999
then
begin
  jwp 1 +  jwp :=
  kwp 1 -  kwp :=
  get.jwp_m
  -1        mwp :=
  begin
    mwp 1 +   mwp :=
    norm.v
      sub[   mwp 2 jwp     ** * 1 + , 2 jwp ** ]
      dup * sort   rev[  1 ]    ^sum
      dup [ 2 jwp ** ] norm.keep.p *    norm.keep.sum.v :=
      0  tmp.v.i :=
      begin
        tmp.v.i 1 +  tmp.v.i :=
        dup  [ tmp.v.i ]          norm.keep.sum.v  >
        tmp.v.i                   2 jwp **         =
        or
      until
      drop
      tmp.v.i  2 jwp ** =
      if
        tmp.v.i       icd :=
      else
        tmp.v.i       icd :=
      \ tmp.v.i 1 -   icd :=
      then

    awp.t  xsect[ jwp_m       , ! ]
      sub[   mwp 2 jwp     ** * 1 + , 2 jwp ** ]
      dup * sort   rev[  1 ]    ^sum
      \ dup [ 1 ]  norm.keep.p *    norm.keep.sum.v :=
      0  tmp.v.i :=
      begin
        tmp.v.i 1 +  tmp.v.i :=
        dup  [ tmp.v.i ]          norm.keep.sum.v  >
        tmp.v.i                   2 jwp **         =
        or
```

```
        until
        drop
        tmp.v.i  2 jwp ** =
        if
          tmp.v.i       icu :=
        else
          tmp.v.i       icu :=
        \ tmp.v.i 1 -   icu :=
        then
      icu  icd <
      if
        \ cr ." Find at (J,M)=("  jwp  .  mwp .  ." )"
        cr " Find at (J,M)=("  jwp 0 n>s "cat
        " , " "cat  mwp 0 n>s "cat  " )" "cat
        "type
        1 to.123.999.flag =
        if
          "     " icu 0 n>s  "cat  " <" "cat  icd 0 n>s "cat  "type
          " ("  jwp 0 n>s "cat
          " ," "cat  mwp 0 n>s "cat  " )" "cat
          to.123.999
        then
        jwp
        norm.l sub[  mwp 2 jwp     ** * 1 + , 2 jwp ** ] :=
        awp.t  xsect[ jwp_m        , ! ]
               sub[  mwp 2 jwp     ** * 1 + , 2 jwp ** ]
        norm.v sub[  mwp 2 jwp     ** * 1 + , 2 jwp ** ] :=
      else
      then
      mwp    2 kwp 1 + ** 1 -  =
    until
    jwp  n.power 1 - =
  until
;
\ ================================================================
: inverse.wt \ [ at.t   h(summing)   g(differencin) -- swt.t ]
\ 2  jwp.level :=    \ range from 1 to (n.power-1)
  sub[ 1 , n.wt1 ]
  1 *
  becomes> at.t
  assign.wlet
  -1  isign   :=
  get.h&g      \ get correct sign

  n.wt1  my.power

  make.awp.t
  jwp.level          jwp :=
  jwp 1 -            nn  :=
  n.power 1 - jwp -  kwp :=
  get.jwp_m

  at.t   sub[ 1 , n.wt1 ]  awp.t  sub[ jwp_m ; 1 , 2 n.power ** ]  :=
  at.t   sub[ 1              , 2  jwp **     ]    becomes>  s.t
  at.t   sub[ 2  jwp ** 1 +                  ]    becomes>  d.t
```

```
dp.real.fft

begin
  jwp 1 +  jwp :=
  nn  1 +  nn  :=
  kwp 1 -  kwp :=
  get.jwp_m
  {tmp.wnd}
  2 0 fix.format
  cr ."         ===> INV Scale Form (" jwp.level . ."  --->"  n.power .
     ." ) "  ." J = "  jwp .
  4 0 fix.format

  s.t  dup    catenate    dup  becomes>  u.t    \ set proper dim
                          dup  becomes>  uwp.t
                          dup  becomes>  vwp.t
                          dup  becomes>  swp.t
                               becomes>  s.t

  0.  u.t   :=   \ upsampling
  s.t  sub[  1 ,  2 nn ** ]    u.t  sub[ 1 , 2 nn ** , 2 ]  :=
       u.t   make.at.t.inv
       at.t
       ht.t  per.conv              \   for inverse WT
       sub[  1    ,   2 nn   1 + **   ]

  0.  u.t   :=
  d.t  sub[  1 ,  2 nn ** ]    u.t  sub[ 1 , 2 nn ** , 2 ]  :=
       u.t   make.at.t.inv
       at.t
       gt.t  per.conv
       sub[  1    ,   2 nn   1 + **   ]

  +   becomes> s.t
  \ --------------------------------------------
  wp.flag  1 =
  if
  \ --------------------------------------------
  s.t                 awp.t sub[ jwp_m , 1  ; 1 , 2  jwp      ** ] :=
  \ jwp from (jwp.level + 1 --> n.power )

  jwp  n.power <
  if
    0  mwp :=
    ."  WP(" 2 kwp 1 + ** 1 -  n.type ." trees)"
    {29.line}  sc
    begin
      mwp 1 +  mwp :=
      mwp  .

      0.  uwp.t :=
      0.  vwp.t :=
      jwp  1  >
      if
```

```
          jwp_m 1 -    tmp.v.i.2  :=
        else
          jwp_m        tmp.v.i.2  :=
        then
        awp.t  xsect[ tmp.v.i.2   , ! ]
               sub[   mwp 2 jwp      ** * 1 + , 2 jwp 1 - ** ]
               uwp.t  sub[ 1 , 2 nn ** , 2 ]  :=
        awp.t  xsect[ tmp.v.i.2 , ! ]
               sub[   mwp 2 jwp      ** * 1 +
                          2 jwp 1 - **      + , 2 jwp 1 - ** ]
               vwp.t  sub[ 1 , 2 nn ** , 2 ]  :=

        uwp.t  make.at.t.inv
               at.t
               ht.t  per.conv              \   for inverse WT
               sub[  1     ,   2 nn 1 +   **   ]
        vwp.t  make.at.t.inv
               at.t
               gt.t  per.conv              \   for inverse WT
               sub[  1     ,   2 nn 1 +   **   ]
        +      becomes>  swp.t
        swp.t
        \    awp.t sub[ jwp , 1   ; mwp 2 jwp      ** * 1 + , 2 jwp      ** ] :=
             awp.t sub[ jwp_m , 1  ; mwp 2 jwp      ** * 1 + , 2 jwp      ** ] :=
        mwp  2  kwp 1 + **  1 -   =
      until
    then

    then
    \ --------------------------------------------

    jwp  n.power  <
    if
      d.t  sub[ 2  nn **  1 +  ]    becomes>   d.t
    then
    jwp  n.power  =
  until
  s.t  becomes> swt.t
  wp.flag  1 =
  if
    get.swp.t
  then
  s.t  becomes> swp.t
  swt.t
;
\ =================================================================
: forward.wt \ [ at.t , ht.t , gt.t  -- ]
  sub[ 1 , n.wt1 ]
  1 *
  becomes> at.t
  assign.wlet
  +1  isign  :=
  get.h&g     \ get correct sign
  n.wt1  my.power
```

```
make.awp.t
n.power  jwp  :=
-1       kwp  :=
0        nn   :=
at.t  sub[ 1 , n.wt1 ]  awp.t  sub[ jwp ; 1 , 2 jwp ** ]  :=
begin
  nn  1 +  nn  :=
  kwp 1 +  kwp :=
  jwp 1 -  jwp :=
  get.jwp_m
  {tmp.wnd}
  2 0 fix.format
  cr ."        ===> FWR Scale From (" n.power . ." -->" jwp.level .
     ." ) "  ." J = "  jwp .
  4 0 fix.format
  at.t   ht.t
  aper.flag  1  =
  if  "  >a" s.type  aper.conv
  else  per.conv     then
  c.size 2 /  2  modulo  0 =
  if
    sub[            1    , n.wt1  2 nn ** / ,  2 ]  becomes> u.t
  else
    sub[            1    , n.wt1  2 nn ** / ,  2 ]  becomes> u.t
  then
  at.t   gt.t
  aper.flag  1  =
  if   " >a" s.type  aper.conv
  else  per.conv     then
  c.size 2 /  2  modulo  0 =
  if
    sub[            1    , n.wt1  2 nn ** / ,  2 ]  becomes> v.t
  else
    sub[            1    , n.wt1  2 nn ** / ,  2 ]  becomes> v.t
  then
  \ ---------------------------------
  wp.flag  1 =
  if
  \ ---------------------------------
  u.t  v.t  catenate  awp.t sub[ jwp_m , 1  ; 1 , 2  jwp 1 + ** ] :=
  jwp  n.power 1 - <
  if
    0  mwp  :=
    ."  WP(" 2 kwp ** 1 -  .  ." )"
    {29.line}  sc
    begin
      mwp 1 +  mwp :=
      mwp  .
      awp.t sub[ jwp   1 + , 1 ; mwp 2 jwp 1 + ** * 1 + , 2 jwp 1 + ** ]
      \ jwp from (n.power-1 --> jwp.level)
      0 tmp.v.i :=
      begin
        dup catenate
        tmp.v.i 1 +  tmp.v.i :=
        tmp.v.i  kwp =
```

```
        until
        dup
        ht.t  per.conv
        sub[   1    , n.wt1  2 nn ** / ,  2 ]  becomes> uwp.t
        gt.t  per.conv
        sub[   1    , n.wt1  2 nn ** / ,  2 ]  becomes> vwp.t
        uwp.t  vwp.t  catenate
        awp.t sub[ jwp_m , 1 ; mwp 2 jwp 1 + ** * 1 + , 2 jwp 1 + ** ]  :=
        mwp  2  kwp **  1 -   =
      until
    then

    then
    \ ---------------------------------
    jwp  n.power 1 -  =
    if
      v.t                becomes> d.t
    else
      v.t  d.t  catenate  becomes> d.t
    then
    0  tmp.v.i  :=
    u.t
    begin
      tmp.v.i  1 +   tmp.v.i  :=
      dup  catenate
      tmp.v.i  nn  =
    until
    becomes> at.t
    jwp  jwp.level =
  until
  u.t  d.t  catenate      becomes> d.t
  d.t          becomes> swt.t
  wp.flag  1 =
  if
    jwp.level  0  =
    if
      awp.t xsect[ n.power 1 +  , ! ]
    else
      awp.t xsect[ jwp.level , ! ]
    then
    dup          becomes> swp.t
  else
    d.t
  then
;
\ ================================================================
: aper.wt.convo \ [ at.t , ht.t , gt.t  -- ] use "a"
  becomes> gt.t
  becomes> ht.t
  becomes> at.t
  n.wt1  my.power
  1  nn  :=
  begin
    cr ." --- " nn  .
    \   at.t   ht.t        conv.aper(fft)
```

```
    at.t   ht.t             conv.aper
    sub[ im 1 - 2 / 2 +  , n.wt1  2 nn ** / ,  2 ]  becomes> u.t
    \   at.t   gt.t          conv.aper(fft)
    at.t   gt.t             conv.aper
    sub[ im 1 - 2 / 2 +  , n.wt1  2 nn ** / ,  2 ]  becomes> v.t
    nn  1  =
    if
      v.t                  becomes> d.t
    else
      v.t  d.t  catenate   becomes> d.t
    then
    1  tmp.v.i  :=
    u.t
    begin
      dup  catenate
      tmp.v.i  1 +   tmp.v.i  :=
      tmp.v.i  nn   >
    until
    becomes> at.t
    nn  1 +  nn :=
    nn  n.power 1 -    >
    \    1  1 =
  until
  u.t  d.t  catenate      becomes> at.t
  at.t
;
: my.wt
  isign   1 =
  if   forward.wt
  else inverse.wt
  then
;
\ ==================================================================
: my.wt.multi.bat
  tmp.s        ":=  \ window.type
  becomes> gt.t
  becomes> ht.t
  tmp.v.i.9    :=  \ #.sec
  tmp.v.i.1    :=  \ #.fft.pts

  tmp.v.i.9   1   >
  if
    []size    tmp.v.i :=
    tmp.v.i   tmp.v.i.1  /  2 *    1 -     tmp.v.i.2  :=
    tmp.v.i   tmp.v.i  tmp.v.i.1  /  tmp.v.i.1 *    -
    tmp.v.i.1  2 /
    >
    if
       tmp.v.i.2 1 +   tmp.v.i.2 :=
    then
    tmp.v.i.2  tmp.v.i.9 >
    if
       tmp.v.i.9  tmp.v.i.2 :=
    then
  then
```

```
  tmp.v.i.2   1  >
  if
    2  tmp.v.i.3    :=                                        \  with overlap
\    tmp.v.i.3  tmp.v.i.2  *    tmp.v.i.2   :=
    1  abs.flag :=
  else
    1  tmp.v.i.3    :=                                        \  no   overlap
  then

  0                    \ initial sum
  swap                 \ --> 0 , array

  tmp.v.i.2  1 +   1
  do
    cr " F----M.Sec:" s.type i .
    dup
    sub[ i 1 -    tmp.v.i.1  tmp.v.i.3 /   *    1 +   , tmp.v.i.1 ]
    tmp.s " x" "=  tmp.s  " X"  "=  or if                       then
    tmp.s " k" "=  tmp.s  " K"  "=  or if  blackman.window.apply then
    tmp.s " b" "=  tmp.s  " b"  "=  or if  bartlett.window.apply then
    tmp.s " n" "=  tmp.s  " N"  "=  or if  hanning.window.apply  then
    tmp.s " m" "=  tmp.s  " M"  "=  or if  hamming.window.apply  then
    ht.t  gt.t      my.wt

    abs.flag  1 =
    if
      abs
    then
    rot  \ --> array , p_s , 0
    +
    swap
  loop
  drop
  tmp.v.i.2  /
;
: my.wt.multi \ [ array , sec_size , sections , window_type -- p.s.array ]
  cr " [ array , fft.pts , #.sec , ht.t , gt.t -- abs_w_c ]" s.type
  my.wt.multi.bat
;
\ -----------------------------------------------------------------------------
\ ==================================================================
```

....

LLet-Nor.asy

勇伯程式庫 A01513-20070125-A064 ❖ January 25, 2007 ❖ [Page count: 5]

```
: LLet-Nor.asy ; \ %----- Asyst Prog

real dim[ 29 ]  array  n.k.p.a
integer scalar icp  scalar sec.level  scalar #.ele  scalar filter.flag
        scalar exp.flag               scalar i.beg
token sig.t  exp.mem> sig.t

n.k.p.a  @[ 1 ] enter[
.995 , .990 , .985 , .980 , .975 ,
.965 , .950 , .925 , .900 , .875 ,
.850 , .825 , .800 , .775 , .750 ,
.700 , .650 , .600 , .550 , .500 ,
.450 , .400 , .350 , .300 , .250 ,
.200 , .150 , .100 , .050        ] drop

real scalar s.freq.keep  scalar p.b.e  scalar t.w
: get.sub.a
  exp.flag 2 =
  if
    sub[ i.beg 2 n.power 1 - ** i.inc * - , 2  n.power 1 + ** , i.inc ]
  else
    sub[ i.beg                            , 2  n.power     ** , i.inc ]
  then
  s.freq.keep  float  i.inc float  /  s.freq :=
  1 filter.flag =
  if
    use.blackman.window
    s.freq  filter.sample.rate
    lowpass[ p.b.e , t.w ]
    filter
  then

  \  s.freq  float  i.inc float  /  s.freq :=
  \  200  float  i.inc float  /  s.freq :=
;

: norm.wt&wp \ statistics of percentage needed to have power beyond
  becomes> tmp.t
  n.k.p.a  becomes>  123.ary.n
  tmp.t []size  swap drop  my.power
  \ n.power      sec.level  :=
  sec.level           jwp :=
  \  cr ." sec.level:"  n.power get.val  sec.level :=
  n.power 1 - jwp -  kwp :=
  get.jwp_m
    \    jwp 1 +  jwp :=
    \    kwp 1 -  kwp :=
    get.jwp_m

    0     icp  :=
    begin
      icp  1 +  icp :=
```

```
      3 0 fix.format
      icp  n.type
      n.k.p.a [ icp ]    norm.keep.p :=
        jwp  n.power =
        if
          2 jwp     **  #.ele :=
          0             mwp   :=
        else
          2 jwp 1 - **  #.ele :=
          1             mwp   :=
        then
     tmp.t   sub[   mwp   2 jwp     ** * 1 + , #.ele ]
        dup *  sort   rev[  1 ]     ^sum
        dup becomes> t8
        dup [  #.ele   ]  norm.keep.p *    norm.keep.sum.v :=
        0  tmp.v.i :=
        begin
          tmp.v.i 1 +  tmp.v.i :=
          dup  [ tmp.v.i ]          norm.keep.sum.v  >
          tmp.v.i                   #.ele            =
          or
        until
        drop
        tmp.v.i  #.ele    =
        if
          tmp.v.i       icd :=
        else
        \ tmp.v.i 1 -   icd :=
          tmp.v.i       icd :=
        then
        icd       float
        #.ele     float /
        100. *    \ in percentage
        123.ary.n  [ icp ] :=

      icp   n.k.p.a []size  swap drop  =
    until
    10 5 fix.format
;
: get.lab
  normal.coords  .5  .05  position
  wlet       "upper
  centered.label
  world.coords
;
: get.lab.wp
  normal.coords  .5  .05  position
  wlet
  wp.flag 1 = if  " --- WP Best Basis"  "cat then
  centered.label
  world.coords
;
: norm.bat
  \  wlet " -" "cat  sec.level 0 n>s  "cat
  now.file " [" "cat
```

```
  sec.level 0 n>s "cat " ," "cat  jwp.level 0 n>s "cat  " ]" "cat
  123.ary.s "[ 1 ] ":=
  dir.lee.asc  now.file      "cat   ftoa.bat
  get.sub.a

  dup becomes> sig.t
  dup
  forward.wt
  hp
  up 1 yap
  get.lab
  dp 1 yap
  get.lab

  \ norm.keep.p.best  norm.keep.p :=
  norm.wp.best  \ get norm.v and norm.l
  swt.t
  norm.wt&wp
  to.123.1

  wp.flag 1 =
  wlet  " le"  "=
  wlet  " me"  "=  or
  wlet  1 2 "sub  " on"  "=  or
  and
  if
    norm.v
    norm.wt&wp
    gd hp
    up norm.l 1 yap  get.lab.wp
    dp norm.v 1 yap  get.lab.wp
    123.ary.s "[ 1 ]  " -" "cat  best.level 0 n>s  "cat  123.ary.s "[ 1 ] ":=
    to.123.2
  then
  file.ext make.abc
  2 sec.level **   0 n>s  123.ary.s "[ 1 ] ":=
  n.k.p.a  becomes>  123.ary.n \ for the additional col ( n.k.p.a )
;
\ =================================================================

: test.total.energy
  b  2 n.power **  999 " x" my.p.s
  \  2 n.power **  *  \ error <- from density to total energy
  []sum
  " on22a" wlet ":=
  10 5 fix.format   .
  b  forward.wt
  dup *
  []sum
  10 5 fix.format .
;

: lab.line.seq.res
  normal.coords
  tmp.v.i 0 n>s  tmp.v.i 1 +  20. /   0.05 position
```

```
  tmp.v.i line.type label
                 tmp.v.i 2 +  20. /   0.05 position
  world.coords
;

: lab.line.seq \ [ tmp.v.i -- ]   [ "str -- ]
  "dup "null  "=
  if
    dup  0 n>s  "swap  "drop
  else
  then
  dup    dup
  normal.coords
                          1 +  20. /   0.05 position
          line.type label
                          2 +  20. /   0.05 position
  world.coords
;
: plot.per.res
  n.wt1  my.power
  norm.v                dup  * sort ^sum  12  yap
  12  " Best Basis"  lab.line.seq
  jwp.level 0 =
  if
    0 tmp.v.i  :=
    1 tmp.v.i.2 :=
    awp.t xsect[ n.power 1 + , ! ] dup  * sort ^sum  tmp.v.i  ydp
    tmp.v.i "null lab.line.seq
  else
    jwp.level 1 -  tmp.v.i  :=
  then
  begin
    tmp.v.i 1 +  tmp.v.i  :=
    awp.t xsect[ tmp.v.i , ! ] dup  * sort ^sum
    tmp.v.i
    tmp.v.i.2  1 =
    if
      ydp
    else
      yap
    then
    tmp.v.i "null lab.line.seq
    tmp.v.i  n.power =
  until
;

: 123.tmp
  {29.line}
  dup [ 2 n.power ** ] /
  dup  becomes> 123.ary.n
  tmp.v.i  0 n>s  " -" "cat  wlet "cat  now.file "cat 123.ary.s "[ 1 ]
":=
  to.123
  file.no 1 + file.no :=
  previous.window
```

```
;
: plot.per
  1 file.no :=
  1 gp.flag :=
  \ make sure this .wks exist
  dir.lee.asc  " norm-dis.wks" "cat   123.file.name  ":=
  n.wt1  my.power
  jwp.level 0 =
  if
    0 tmp.v.i  :=
    1 tmp.v.i.2 :=
    awp.t xsect[ n.power 1 + , ! ] dup  * sort ^sum
    123.tmp
    tmp.v.i  yap
    tmp.v.i "null lab.line.seq
  else
    0 tmp.v.i.2 :=
    jwp.level 1 -  tmp.v.i  :=
  then
  begin
    tmp.v.i 1 +  tmp.v.i  :=
    awp.t xsect[ tmp.v.i , ! ] dup  * sort ^sum
    123.tmp
    tmp.v.i
    tmp.v.i.2  1 =
    if
      ydp
      1 tmp.v.i.2 :=
    else
      yap
    then
    tmp.v.i "null lab.line.seq
    tmp.v.i  n.power =
  until
  norm.v                dup  * sort ^sum
  123.tmp
  13  ydp
  13  " Best Basis"  lab.line.seq
  2 n.power **  real ramp   2 n.power ** /
  123.tmp
  swt.t                 dup  * sort ^sum
  123.tmp
  16  ydp
  16  " swt.t"        lab.line.seq
;
```

LLet-Pha.asy

勇伯程式庫 A01514-20070125-A065 ❖ January 25, 2007 ❖ [Page count: 2]

```
: LLet-Pha.asy ; \ %----- Asyst Prog

\ input llet-0.txt first

integer scalar #.wei
real scalar band.start  scalar band.stop

1.1    band.start  :=
25     #.wei       :=
: fre.mod
  " [ array , freq , Hilbert.filter.flag  --  array ]" s.type
  tmp.v.i    :=
  tmp.v      :=
  becomes> t8

  tmp.v   2. /   band.start -   band.stop :=
  t8
  dup
  tmp.v.i  1  =
  if
    tmp.v   filter.sample.rate
    hilbert[ #.wei ; band.start , band.stop ]
    ." epsilon=" n.type
    filter
  else
    hilbert
  then
  \  neg
  swap  z=x+iy
  \ becomes> t7  t7 becomes> t8
  becomes> t8

  \ Calcute dp/dt  --- use polar z=r*exp(i p)
  t8   differentiate.data                          \ dz
  t8   zmag   differentiate.data                   \ dr
  t8   zarg   dup   cos   swap   sin   z=x+iy   *  \ exp(i p) * dr
  -                                                \ --> numerator
  t8     0 1 z=x+iy   *                            \ i * z --> denomenator
  /

  \ zmag                                           \ Wrong
  zreal                                            \ Yes
  tmp.v   0   =
  if
  else
    2. pi *   /     tmp.v   *        \ change unit from radian to frequency
  then
  becomes> t8   \ fre.modu
  t8
;

: m0.s.phase
```

```
  b inverse.wt 1 yap
  wlet tmp.s.u ":=
  tmp.s.u "upper  123.ary.s "[ 1 ] ":=
  p.coef fft zarg  becomes> 123.ary.n
\ q.coef fft zarg  becomes> 123.ary.n
  123.ary.n 1 yap
  get.lab
  to.123
  file.ext make.abc
  \ sd
;
```

LLet-Psi.asy

勇伯程式庫 A01515-20070125-A066 ❖ January 25, 2007 ❖ [Page count: 2]

```
: LLet-Psi.asy ; \ For plotting Psi Curves \ %----- Asyst Prog
\ : PsiC-Wks.asy ;
\ %-------------------------------------------------------------------
\     Output Real and Imag Parts of psi to wks file
\ %-------------------------------------------------------------------

\ \ remove path
123.file.name.1
dir.lee.asc "len  1 +   "drop
"dup "len "drop  dir.lee.asc "len "drop -  "sub  123.file.now ":=

\ \ %=================================================================
\ %------------------------------------------    Morlet (new)
123.file.now  v.t zreal  " Re-N"  1 a2wks
123.file.now  v.t zimag  " Im-N"  2 a2wks
123.file.now  v.t zimag []size swap drop real ramp
" $$[(3,1024)]"  3 a2wks

\ \ %=================================================================
\ %------------------------------------------    ridge
\ " RIWV.wka"  v.t zreal  " Re-R"  1 a2wks
\ " RIWV.wka"  v.t zimag  " Im-R"  2 a2wks
\ " RIWV.wka"  v.t zimag []size swap drop real ramp
\ " $$[(3,1024)]"  3 a2wks
\ \ \ %=================================================================
\ \ %------------------------------------------    Morlet (new)
\ " RImorlet.wka"  v.t zreal  " Re-N"  1 a2wks
\ " RImorlet.wka"  v.t zimag  " Im-N"  2 a2wks
\ " RImorlet.wka"  v.t zimag []size swap drop real ramp
\ " $$[(3,1024)]"  3 a2wks
\ %=================================================================

\ \ %=================================================================
\ %------------------------------------------    template for a2wks
: model
" morletRI.wka"  v.t zreal  " Re-ori+d" 15 a2wks
" morletRI.wka"  v.t zimag  " Im-ori+d" 16 a2wks

" morletRI.wka"  v.t zreal  " Re-ori"    1  a2wks
" morletRI.wka"  v.t zimag  " Im-ori"    2  a2wks

" morletRI.wka"  v.t zreal  " Re-old"    3  a2wks
" morletRI.wka"  v.t zimag  " Im-old"    4  a2wks

" morletRI.wka"  v.t zreal  " Re-old-s"  5  a2wks
" morletRI.wka"  v.t zimag  " Im-old-s"  6  a2wks

" morletRI.wka"  v.t zreal  " Re-old+s"  7  a2wks
" morletRI.wka"  v.t zimag  " Im-old+s"  8  a2wks

" morletRI.wka"  v.t zreal  " Re-ori-s"  9  a2wks
" morletRI.wka"  v.t zimag  " Im-ori-s" 10  a2wks
```

```
" morletRI.wka"  v.t zreal  " Re-ori+s" 11  a2wks
" morletRI.wka"  v.t zimag  " Im-ori+s" 12  a2wks

" morletRI.wka"  v.t zreal  " Re-ori-d" 13 a2wks
" morletRI.wka"  v.t zimag  " Im-ori-d" 14 a2wks

" morletRI.wka"  v.t zreal  " Re-ori+d" 15 a2wks
" morletRI.wka"  v.t zimag  " Im-ori+d" 16 a2wks

" morletRI.wka"
v.t zimag []size swap drop
dup
real ramp  s.freq float /
swap 10 +
" $$[(13"
" ,"         "cat
0 n>s        "cat
" )]-ori"  "cat
\ %-%------------------------------------------
                                      13  a2wks
;

\ \

.............................................................................................
```

LLet-WPM.asy

勇伯程式庫 A01516-20070125-A067 ❖ January 25, 2007 ❖ [Page count: 3]

```
: LLet-WPM.asy ; \ %----- Asyst Prog
\ \  0: Load Llet-0.txt
\ \  1: Set.cwt
\ \  2: Set.act
\ \  3: action
\ \  4: get.map

token wp.a      exp.mem> wp.a      token wp.p      exp.mem> wp.p
token wp.s      exp.mem> wp.s      token wp.o      exp.mem> wp.o
token x.corn1   exp.mem> x.corn1   token x.corn2   exp.mem> x.corn2
token y.corn1   exp.mem> y.corn1   token y.corn2   exp.mem> y.corn2
8  string no.ext
64 string fidir
" c:\per\mat\0-map\data\"  fidir  ":=

integer scalar best.flag
: get.no.ext
  now.file
  " ."   now.file "within 1  -  ?drop  1 swap "sub
\ " -"                                          "cat
  now.file
  " ."   now.file "within  1  +  ?drop  1       "sub   "cat
  no.ext  ":=
;
: get.map
  cr ." BEST BASIS (1) or BEST LEVEL (2) [at "  best.level 0 n>s s.type
  ."  ] : "
  1
  get.val   best.flag   :=
  best.flag 2 =
  if
\     norm.v.s   becomes> norm.v
\     \ awp.t xsect[ best.level , ! ] becomes> norm.v
\     best.level  norm.l   :=
    norm.v.s   becomes> t1
    norm.l     becomes> t2
    best.level      t2   :=
  else
    norm.v     becomes> t1
    norm.l     becomes> t2
  then

  get.no.ext
  -20   6   sci.format
  cr ." Mat file name for No.Ext:"   no.ext  get.str  no.ext ":=
  fidir   no.ext   "cat  " .sip"  "cat
  "dup  defer>  delete
        defer>  out>file
  0 i.ele :=
  Begin
    i.ele 1 +  i.ele :=
    sig.t    [ i.ele ]   .                            cr
```

```
    i.ele  2 n.power ** =
  Until
  out>file.close

  2 n.power **  real ramp  becomes> wp.a
  wp.a becomes> wp.p
  wp.a becomes> wp.s
  wp.a becomes> wp.o
  wp.a becomes> t8
  wp.a becomes> x.corn1
  wp.a becomes> x.corn2
  wp.a becomes> y.corn1
  wp.a becomes> y.corn2
  0 i.ele :=
  Begin
    i.ele 1 +  i.ele :=
    cr ." i.ele:" i.ele  .
  \ -1 mwp  :=  \ mwp=0 & wp.o=0 --> no oscillation --> scaling function
                \ mwp=1          --> WB
                \ mwp>1          --> WP

        i.ele  1 -  float
               2    float  t2  [ i.ele ] ** /  floor  mwp            :=
        t1 [ i.ele ]                           wp.a [ i.ele ] := \ amplitude
        i.ele  mwp 2    t2 [ i.ele ] ** *  -   wp.p [ i.ele ] := \ position
        n.power         t2 [ i.ele ]   -       wp.s [ i.ele ] := \ timesupport
        mwp                                    wp.o [ i.ele ] := \ oscillation
    ."    mwp:"  mwp   .
    i.ele  2 n.power ** =
  Until

  n.power   t8 :=    \ n.power array
  \ wp.p      2 wp.s ** *            becomes>   x.corn1
  \ wp.p 1 +  2 wp.s ** *            becomes>   x.corn2
  wp.p 1 -  2 wp.s ** *            becomes>   x.corn1
  wp.p      2 wp.s ** *            becomes>   x.corn2
  wp.o      2  t8  wp.s - ** *     becomes>   y.corn1
  wp.o 1 +  2  t8  wp.s - ** *     becomes>   y.corn2

  -20   6  sci.format
  fidir   no.ext  "cat  " .mip"  "cat
  "dup  defer>  delete
        defer>  out>file
  0 i.ele :=
  Begin
    i.ele 1 +  i.ele :=
\   x.corn1  [ i.ele ]  .          y.corn1 [ i.ele ] .  cr
    x.corn1  [ i.ele ]  s.freq float /    .
    y.corn1  [ i.ele ]  s.freq float *  2.  /  2 n.power ** float /  .  cr
    i.ele  2 n.power ** =
  Until
  out>file.close
  fidir   no.ext  "cat  " .map"  "cat
  "dup  defer>  delete
        defer>  out>file
```

```
  0 i.ele :=
  Begin
    i.ele 1 +  i.ele :=
\   x.corn2  [ i.ele ]  .           y.corn2 [ i.ele ] .  cr
    x.corn2  [ i.ele ]  s.freq float /    .
    y.corn2  [ i.ele ]  s.freq float *  2.  /  2 n.power ** float /  .  cr
    i.ele  2 n.power ** =
  Until
  out>file.close
  fidir   no.ext  "cat  " .amp"  "cat
  "dup  defer>  delete
        defer>  out>file
  0 i.ele :=
  Begin
    i.ele 1 +  i.ele :=
    wp.a     [ i.ele ]  .                             cr
    i.ele  2 n.power ** =
  Until
  out>file.close
  fidir   no.ext  "cat  " .spp"  "cat
  "dup  defer>  delete
        defer>  out>file
  n.power .    jwp.level .   best.flag .   best.level .   s.freq .
  t2  [ 1 ] .  norm.keep.p.best .
  out>file.close
  cr
  fidir   no.ext  "cat  " .stp"  "cat
  "dup  defer>  delete
        defer>  out>file
  wlet s.type  cr  now.file "type   cr best.level 0 n>s s.type
  out>file.close
  \  " c:\per\mat\currentp.stp"
  fidir " cur-map.fip"  "cat
  "dup  defer>  delete
        defer>  out>file
  no.ext "type
  out>file.close
  cdtNow
  10 5 fix.format
;
\ get.map
```

LLet-WVT.asy

勇伯程式庫 A01517-20070125-A068 ❖ January 25, 2007 ❖ [Page count: 20]

```
: LLet-Rdg.asy ; \ %----- Asyst Prog
\ : LLet-CW2.asy ;

\ Real.FFT

integer scalar #.fre                    scalar mat.inc
integer scalar mat.beg   scalar mat.pts  scalar mat.power
        scalar erf.flag
real scalar f.cen    scalar f.dia    scalar f.shi
real scalar sai.beg  scalar sai.end  scalar sai.now  scalar a.sca
real scalar fre.beg  scalar fre.end  scalar fre.now  scalar fre.inc

1  string rea.or.com
1  string psi.flag
1  string pha.switch.flag
12 string now.file.2
token t3  exp.mem> t3
token t4  exp.mem> t4
32 string wlet
64 string fidir
" c:\per\mat\0-cwt\data\"  fidir  ":=

: get.sai \ sai.beg ~ large value ~ large scale ~ low frequency
          \ sai.end ~ small value ~ small scale ~ high frequency
  1 erf.flag =
  if
    \ Erfc[ .4 (f+2.5) - 2 ] * 3 + 5
    2.  2. *      7.5 -2.5  -  /
    fre.now  -2.5  -  *
    2   -
    erf
    1  swap -
    sai.beg  sai.end -
    2. /
    *
    sai.end  +
  then
  2 erf.flag =
  if
    s.freq  float 2. /  2. /  f.cen   :=   \ % Nyquist first then divided by 2
    s.freq  4.       /        f.dia   :=   \ % Spread width of Erfc[x] is about 4;
    0.                        f.shi   :=   \ %    while Erfc[+2]=0 and Erfc[-2]=>2.
    \  % Erfc[ (f - f.cen )/f.dia - f.shi ] /2 * (sai.beg - sai.end)
    fre.now  f.cen -    f.dia  /
    f.shi -
    erf
    1  swap  -
    2. /
    sai.beg  sai.end -    *
    sai.end  +
  then
  \  0 erf.flag =
```

```
  1  erf.flag  <>
  2  erf.flag  <>  and
  if
    #.fre  1 >
    if
      sai.end  sai.beg -
      fre.end  fre.beg - /
      fre.now  fre.beg - *
      sai.beg  +
    else
      sai.beg
    then
  then
  sai.now  :=
;
\ %==============================================================
\ %1 If not shifted, Re-part of psi is symmetrical and Im-part anti-symmetrical.
\ %2 Whether shifted or not, both Re- and Im-part are sinusoidal.
\ %3 Analytical Modulus =~= Real part Envelope
\ %4 Ridge extraction from envelope.
\ %5 Poor phase information
\ %==============================================================
: get.psi.sca.new \ Original -- The same as Morlet's
  \ rev[ 1 ] is placed in ramp
  \ % ========================================  Carrier
  exp.flag 1 =
  exp.flag 2 = or
  if
    2  n.power **
  else
    2  n.power 1 - **
  then
  real ramp  1. -
  \ %!%-------------------------
    dup 1. +
    rev[ 1 ]
    neg
    catenate
  \    dup  -1 rotate
  \    0.   over [ tmp.v.i    ] :=
  \    rev[ 1 ]
  \  neg
  \    catenate
  \ %^%-------------------------
  s.freq  /   \ absolute time
  becomes> t2

  \ % ========================================  Carrier
  T2
  \ %!%----------------------------------------
  \ %  Choose one !
     sai.now *  a.sca /            \ %%       use to.mat.e.p --> good mod, poor pha
  \  sai.now *  a.sca / pi 2. / +  \ %%       use to.mat.e.p -->
  \  sai.now *  a.sca / pi 2. / -  \ %% The wavelet is positively shifted
    "  (Phase adjusted: 0)"  tmp.s.9  ":=
```

```
\ %!%-----------------------------------------
\ % %----  neg  dup  cos  swap  sin  z=x+iy
dup  cos  swap  sin  z=x+iy       \ The negative sign is removed for
                                  \    consistent definition of FT pairs.
becomes> u.t

\ % =========================================  Modulator
t2
\ %--%                            \ %%  ? Don't shift here (the index
\ %                                     has been in good match). --- ?
\ %!%-----------------------------------------
\ %  Choose one!
   1. *                           \ %% No shift
\  pi 2. /  a.sca *  sai.now / + \ %% The envelope is negatively shifted
\  pi 2. /  a.sca *  sai.now / - \ %% The envelope is positively shifted.
\ %^%-----------------------------------------
dup *
2. /  a.sca /  a.sca /  neg
becomes> v.t

0 tmp.v.i :=                             \  Treat those values that cause
begin                                    \    numerical underflow when taken
  tmp.v.i 1 +  tmp.v.i :=                \    to exponential.
  -50.   v.t [ tmp.v.i ]  >=
  if
    -50.   v.t [ tmp.v.i ]  :=
  then
  exp.flag 1 =
  exp.flag 2 =  or
  if
    tmp.v.i   2 n.power 1 +    **   =
  else
    tmp.v.i   2 n.power        **   =
  then
until
v.t  exp    becomes> v.t

u.t  v.t  *  pi  -0.25  **  *
a.sca  sqrt /
becomes> v.t

\ % =========================================  render analytic signal of psi
\ % Theoretically, there is no effect except a factor 2 difference in modulus.
\   \ % v.t   dup   hilbert   0  1 z=x+iy   *  +
\   v.t   dup   hilbert   0 +1 z=x+iy   *  +   \ \ %  H[cos(x)]= - sin(x)
\   2. /                                     \      render ~0 v.t
\   becomes> v.t
\   v.t   dup   hilbert   0 -1 z=x+iy   *  +   \ \ %  H[cos(x)]= + sin(x)
\   2. /                                     \      compatible with mine
\   becomes> v.t
\  \ %^% ----------- Original
\ % =========================================  difference : A[psi]-psi
\ % How analytic is the Morlet wavelet?
\  v.t
\  v.t   dup   hilbert   0 -1 z=x+iy   *  +   \ \ %  H[cos(x)]= + sin(x)
```

```
  \  2. /
  \  swap -
  \  becomes> v.t
;
\ %================================================================================
\ %  Both Re & Im parts of psi are symmetrical.
\ %  If not shifted, Im part is not sinusoidal ---> Ridge extration from phase.
\ %  If shifted, Re part is not sinusoidal ---> Ridge extraction from envelope.
\ %================================================================================
\ : get.psi.sca.old  \ Ridge detection from phase plot --- Use to.mat.e.p
: get.psi.sca.old \ Ridge detection from phase plot --- Use to.mat.e.p
  \ rev[ 1 ] is placed after ramp
  \ % =========================================  Carrier
  exp.flag 1 =
  exp.flag 2 = or
  if
    2  n.power **
  else
    2  n.power 1 - **
  then
  \ real ramp  s.freq  /   \ absolute time
  real ramp  1. -  s.freq  /   \ absolute time
  \ %!%-----------------------------------------
  \ %  Choose one!
  \                                \ %% Old - Detection of ridge in phase plane
  psi.flag "upper " R" "=
  if
  \  sai.now *  a.sca / pi 2. / +  \ %%  -shift  to.mat.e.p --> good ridge, bad pha
     sai.now *  a.sca / pi 2. / -  \ %%  +shift
    "  (Phase adjusted: -pi/2)"  tmp.s.9  ":=
  else
     sai.now *  a.sca /            \ %%       use to.mat.e.p --> good mod & pha
    "  (Phase adjusted: 0)"  tmp.s.9  ":=
  then
  \ %^%-----------------------------------------
  \  dup  cos  swap  neg  sin  z=x+iy
  neg  dup  cos  swap  sin  z=x+iy
  becomes> u.t

  \ % =========================================  Modulator
  exp.flag 1 =
  exp.flag 2 = or
  if
    2  n.power **
  else
    2  n.power 1 - **
  then
  real ramp  1. -  s.freq  /   \ absolute time
  \ %!%-----------------------------------------
  \ %  Choose one!
  psi.flag "upper " R" "=
  if
  \  pi 2. /  a.sca *  sai.now / + \ %% The envelope is negatively shifted.
     pi 2. /  a.sca *  sai.now / - \ %% The envelope is positively shifted.
  else
```

```
    1. *                          \ %% No shift
  then
  \ %^%----------------------------------------
  dup *
  2. /  a.sca /  a.sca /  neg
  becomes> v.t
  0 tmp.v.i  :=
  begin
    tmp.v.i 1 +  tmp.v.i  :=
    -50.   v.t [ tmp.v.i ]  >=
    if
      -50.   v.t [ tmp.v.i ]  :=
    then
    exp.flag 1 =
    exp.flag 2 =  or
    if
      tmp.v.i   2 n.power     **   =
    else
      tmp.v.i   2 n.power 1 - **   =
    then
  until
  v.t  exp    becomes> v.t

    u.t  v.t  *  pi  -0.25  **  *
    dup  -1 rotate
    \  %    0.   over [ tmp.v.i    ] :=
    dup [ tmp.v.i 1 - ]    over [ tmp.v.i    ] :=
    rev[ 1 ]
  catenate

  a.sca  sqrt /

  becomes> v.t

  v.t  conj                 \ Match the +- sign of the definition of FT pair
  becomes> v.t

  \ \% v.t   dup    hilbert      z=x+iy      \ render analytic signal
  \ v.t   dup    hilbert   0 1 z=x+iy    *  +      \ render analytic signal of psi
  \ becomes> v.t

  \  \\ v.t []size  2 / rotate    \ peak at center
  \  \\ becomes> v.t
;

: get.psi.sca
  psi.flag "upper " O" "=
  psi.flag "upper " R" "=  or
  if
    get.psi.sca.old
  else
    get.psi.sca.new
  then
;
: test.hilbert.sign.compatibility
```

```
  gd
  up
  v.t  zreal
  2. *
  1 yap
  v.t  zreal   hilbert
  2. *
  5 ydp
  v.t   dup    hilbert   0  +1  z=x+iy   *  +
  zreal
  \ 1 yap
  2 ydp
  dp
  v.t  zimag
  2. *
  1 yap
  v.t   dup    hilbert   0  +1  z=x+iy   *  +
  zimag
  2 ydp
  hp
;
: ini.re.im.ary
  1 tmp.v.i :=
  2 mat.power **   real ramp
    begin
      tmp.v.i 1 +  tmp.v.i :=
      2 mat.power **  real ramp
      laminate
      tmp.v.i  #.fre  >=
    until
  becomes> rt.t
  rt.t  trans[ 1 , 2 ]  becomes> rt.t
  0.  rt.t :=
  rt.t becomes> at.t
  rt.t becomes> wp.a
;

: get.v.t.sub \ show half
  exp.flag 1 =
  exp.flag 2 = or
  if
    sub[ 1 , 2 n.power     ** ]
  else
    sub[ 1 , 2 n.power 1 - ** ]
  then
;
: get.mat.sub
  sub[ 1 , mat.pts , mat.inc ]
;
: cwt
  #.fre real ramp becomes> a.coef
  a.coef  becomes> b.coef
  a.coef  becomes> p.coef

  fre.beg  fre.inc   #.fre 1 - *  +  fre.end :=
```

```
n.wt1  my.power
dir.lee.asc  now.file "cat  ftoa.bat
get.sub.a
becomes> sig.t            \ data
ini.re.im.ary
fre.beg   fre.now :=
gd hp
1  vi1  :=
Begin
  get.sai
  sai.now   2 pi fre.now * *  /   a.sca :=
  get.psi.sca             \ get v.t : scaled psi

  3 0 fix.format  cr  vi1 .
  5 2 fix.format  ."  f="  fre.now n.type
      ."  s=" sai.now .  7 4 fix.format  ."  A=" a.sca n.type
  fre.now   b.coef [ vi1 ] :=
  a.sca     a.coef [ vi1 ] :=
  sai.now   p.coef [ vi1 ] :=
  v.t zreal  get.v.t.sub  up 1 yap
  v.t zimag  get.v.t.sub  dp 1 yap
      now.file  \ " --" "cat now.file.2  "cat
      \ "   [File No. " "cat    file.no 1 +  0 n>s  "cat      "  ]" "cat
      normal.coords .5  .05 position
      "  [" "cat psi.flag "cat " ]" "cat
      tmp.s.9 "cat
      "  f=" "cat  fre.now 0.00001 + -2 n>s  "cat
      "  s=" "cat  sai.now  -2 n>s  "cat
      "  A=" "cat  a.sca    -3 n>s  "cat
      centered.label world.coords
  exp.flag 1 =
  if
    sig.t  sub[ 1 , 2 n.power 1 - ** ] rev[ 1 ]
    sig.t
    catenate
    sig.t  sub[ 2 n.power 1 - **  1 + , 2 n.power 1 - ** ] rev[ 1 ]
    catenate
  else  \ exp.flag=0 , 2
    sig.t
  then
  \   dup    hilbert      z=x+iy      \ render analytic signal
                fft             \ F^(w)
  v.t    conj    fft rev[ 1 ]   \ G^(-w)   , psi=conj(g)
  \     v.t            fft rev[ 1 ]        \ G^(-w)
  \     *  a.sca  sqrt /                   \ move to get.psi.sca
  *                                     \ for move to get.psi.sca
  ifft
  dup zreal envelope
  exp.flag 1 =
  exp.flag 2 = or
  if
    sub[ 2 n.power 1 - **  1 + , 2 n.power ** ]
  else
  then
  get.mat.sub
```

```
    wp.a  xsect[ ! , vi1 ] :=
    exp.flag 1 =
    exp.flag 2 = or
    if
      sub[ 2 n.power 1 - **  1 + , 2 n.power ** ]
    else
    then
    get.mat.sub
    dup  zreal  rt.t  xsect[ ! , vi1 ] :=
    zimag  at.t  xsect[ ! , vi1 ] :=
    fre.now  fre.inc +   fre.now :=
    vi1 1 +  vi1 :=
    \  vi1     #.fre 1 +    >
    vi1    #.fre 1 +    =
 Until
 exp.flag 2 =
 if
   sig.t  sub[ 2 n.power 1 - **  1 + , 2 n.power ** ]  becomes> sig.t
 then
;

: to.mat.p
  becomes> awp.t
  \ awp.t    []size tmp.v.i := drop
  -12  6 sci.format
  0 vi2 :=
  begin
    vi2 1 +  vi2 :=
    0 vi1 :=
    begin
      vi1 1 +  vi1 :=
      awp.t [ vi2 , vi1 ]  .
      vi1  #.fre  =
    until
    cr
    vi2 2 mat.power ** =
  until
;
: compu.pha.res \ -pi <--> pi   + pi

  0 tmp.v.i :=  \ reassign rt.t when rt.t=0 in denominator
  begin
    tmp.v.i 1 +  tmp.v.i :=
    rt.t xsect[ ! , tmp.v.i ]
    dup abs  []max  tmp.v :=                \ X
    becomes> t1
    0 vi1 :=
    begin
      vi1 1 +  vi1 :=
      t1 [ vi1 ]  0.  =
      if
        tmp.v  10. -10. ** *
        rt.t xsect[ ! , tmp.v.i ]  [ vi1 ]   :=
      else
      then
```

```
    vi1  2 mat.power ** =
   until
   tmp.v.i  #.fre =
 until

 0 tmp.v.i :=
 begin
   tmp.v.i 1 +  tmp.v.i :=
   \    at.t xsect[ ! , tmp.v.i ]                   \ Y
   \    rt.t xsect[ ! , tmp.v.i ]   /               \ X
   \    atan
   \    becomes> t1

   rt.t xsect[ ! , tmp.v.i ]                   \ X
   at.t xsect[ ! , tmp.v.i ]                   \ Y
   z=x+iy
   zarg  pi + becomes> t1

   t1  bwp.t xsect[ ! , tmp.v.i ]  :=
   tmp.v.i  #.fre =
 until
;

: compu.pha \ 0 -- 2pi

 \  pha.switch.flag  "upper " Y"  "=
 pha.switch.flag  "upper " R"  "=
 if   \ % switch real and imaginary parts
   rt.t   becomes>  t4
   at.t   becomes>  t3
 else
   rt.t   becomes>  t3
   at.t   becomes>  t4
 then

 0 tmp.v.i :=  \ reassign rt.t when rt.t=0 in denominator
 begin
   tmp.v.i 1 +  tmp.v.i :=
   t3   xsect[ ! , tmp.v.i ]
   \ rt.t xsect[ ! , tmp.v.i ]
   dup abs  []max  tmp.v :=               \ X
   becomes> t1
   0 vi1 :=
   begin
     vi1 1 +  vi1 :=
     t1 [ vi1 ]  0.  =
     if
       tmp.v  10. -10. ** *
       t3   xsect[ ! , tmp.v.i ]  [ vi1 ]   :=
       \ rt.t xsect[ ! , tmp.v.i ]  [ vi1 ]   :=
     else
     then
     vi1  2 mat.power ** =
   until
   tmp.v.i  #.fre =
```

```
  until

  0 tmp.v.i :=
  begin
    tmp.v.i 1 +  tmp.v.i :=
    t4 xsect[ ! , tmp.v.i ]                    \ Y
    t3 xsect[ ! , tmp.v.i ]   /                \ X
    \ rt.t xsect[ ! , tmp.v.i ]   /                \ X
    atan
    becomes> t1

    0 vi1 :=
    begin
      vi1 1 +  vi1 :=
      t1 [ vi1 ]  0  >=
      if
        t4 xsect[ ! , tmp.v.i ]  [ vi1 ] 0 >=
        \ at.t xsect[ ! , tmp.v.i ]  [ vi1 ] 0 >=
        if
        else
          t1 [ vi1 ]      pi     +  t1 [ vi1 ] :=
        then
      else
        t4 xsect[ ! , tmp.v.i ]  [ vi1 ] 0 >
        \ at.t xsect[ ! , tmp.v.i ]  [ vi1 ] 0 >
        if
          t1 [ vi1 ]      pi     +  t1 [ vi1 ] :=
        else
          t1 [ vi1 ]      pi 2 * +  t1 [ vi1 ] :=
        then
      then
      vi1  2 mat.power ** =
    until

    t1  bwp.t xsect[ ! , tmp.v.i ]  :=
    tmp.v.i  #.fre =
  until
;

: to.mat.p.2
  0 tmp.v.i :=
  begin
    1  tmp.v.i +  tmp.v.i :=
    dup  [ tmp.v.i ] .
    tmp.v.i  #.fre =
  until
  drop
;

: to.mat.m.p \ modulus  and phase
  psi.flag  " N"  "=
  if
    " Adapted CWT"  wlet ":=
  " M1" s.type
  then
```

```
psi.flag  " 0"  "=
if
  " Wavelet Variant (M)"  wlet ":=
" M2" s.type
then

\ " Adapted CWT"  wlet ":=
get.no.ext
-20   6  sci.format
cr ." Mat file name for No.Ext:"   no.ext  get.str  no.ext ":=
  fidir   no.ext  "cat  " .sic"  "cat
cr ." WLET:"   wlet  get.str  wlet ":=
  "dup  defer>  delete
        defer>  out>file
  0 i.ele :=
  Begin
    i.ele 1 +  i.ele :=
    sig.t    [ i.ele ]  .                        cr
    \ i.ele  2 mat.power ** =
    i.ele  2 n.power ** =   \ not mat.power
  Until
  out>file.close

    rt.t  dup *
    at.t  dup * +
    sqrt
    becomes> awp.t

\  \   Envelope of modulus
\  0 tmp.v.i :=
\  begin
\    1  tmp.v.i +  tmp.v.i :=
\    awp.t xsect[ ! , tmp.v.i ]   envelope
\    awp.t xsect[ ! , tmp.v.i ]   :=
\    tmp.v.i  #.fre =
\  until

  fidir no.ext "cat  " .amc" "cat
  "dup  defer> delete
        defer> out>file
  awp.t   to.mat.p
  out>file.close

awp.t  becomes> bwp.t
compu.pha
  fidir no.ext "cat  " .phc" "cat
  "dup  defer> delete
        defer> out>file
  bwp.t   to.mat.p
  out>file.close
cr fidir   no.ext  "cat  " .scc"  "cat
  "dup  defer>  delete
        defer>  out>file
  b.coef  to.mat.p.2  cr
  a.coef  to.mat.p.2  cr
```

```
    p.coef  to.mat.p.2
    out>file.close
  cr fidir   no.ext  "cat  " .spc"  "cat
    "dup  defer>  delete
          defer>  out>file
    n.power .    jwp.level .   best.flag .   best.level .   s.freq .
    mat.power .  mat.inc    .
    \ t2  [ 1 ] .

    out>file.close
  cr fidir   no.ext  "cat  " .stc"  "cat
    "dup  defer>  delete
          defer>  out>file
    wlet s.type  cr  now.file "type   cr best.level 0 n>s s.type
    cr    pha.switch.flag   s.type
    out>file.close
  cr fidir  " cur-cwt.fip"  "cat
    "dup  defer>  delete
          defer>  out>file
    no.ext "type
    out>file.close
  cdtNow
  10 5 fix.format
;
: to.mat.e.p \ envelope  and phase
  psi.flag  " N"  "=
  if
    " Adapted CWT"  wlet ":=
  " E1" s.type
  then
  psi.flag  " 0"  "=
  if
    " Wavelet Variant (H)"  wlet ":=
  " E2" s.type
  then

  \ " Adapted CWT"  wlet ":=
  get.no.ext
  -20   6  sci.format
  cr ." Mat file name for No.Ext:"   no.ext  get.str  no.ext ":=
    fidir   no.ext  "cat  " .sic"  "cat
  cr ." WLET:"   wlet  get.str  wlet ":=
    "dup  defer>  delete
          defer>  out>file
    0 i.ele :=
    Begin
      i.ele 1 +  i.ele :=
      sig.t    [ i.ele ]  .                        cr
      i.ele  2 n.power ** =    \ not mat.power
    Until
    out>file.close

    \     rt.t  dup *
    \     at.t  dup * +
    \     sqrt
```

```
    \     becomes> awp.t

    wp.a            \ envelope
    becomes> awp.t

    fidir no.ext "cat  " .amc" "cat
    "dup  defer> delete
          defer> out>file
    awp.t   to.mat.p
    out>file.close

  awp.t  becomes> bwp.t
  compu.pha
    fidir no.ext "cat  " .phc" "cat
    "dup  defer> delete
          defer> out>file
    bwp.t   to.mat.p
    out>file.close
  cr fidir   no.ext  "cat  " .scc"  "cat
    "dup  defer>  delete
          defer>  out>file
    b.coef  to.mat.p.2  cr
    a.coef  to.mat.p.2  cr
    p.coef  to.mat.p.2
    out>file.close
  cr fidir   no.ext  "cat  " .spc"  "cat
    "dup  defer>  delete
          defer>  out>file
    n.power .    jwp.level .   best.flag .   best.level .   s.freq .
    mat.power .  mat.inc    .
    \ t2   [ 1 ] .

    out>file.close
  cr fidir   no.ext  "cat  " .stc"  "cat
    "dup  defer>  delete
          defer>  out>file
    wlet s.type  cr  now.file "type   cr best.level 0 n>s s.type
    cr    pha.switch.flag   s.type
    out>file.close
  cr fidir  " cur-cwt.fip"  "cat
    "dup  defer>  delete
          defer>  out>file
    no.ext "type
    out>file.close
  cdtNow
  10 5 fix.format
;
: to.mat
  psi.flag "upper " O" "=
  psi.flag "upper " R" "=  or
  cr
  if
    " to.mat.E.p" s.type.3  cr
    " Phase switch flag : "  s.type  pha.switch.flag s.type.3
    to.mat.e.p
```

```
  else
    " to.mat.M.p" s.type.3
    " Phase switch flag : "  s.type  pha.switch.flag s.type.3
    to.mat.m.p
  then
  cr
;
: cwt.56
\ -----------------------------------------------------
  5  4  goto.xy
     ." Now.file (name.ext):"  now.file  s.type \ get.str  now.file ":=
  5  5  goto.xy 4 0 fix.format
  n.wt1  my.power
     ." Number of points of wavelet (n.wt1):"
     2 n.power **  n.type \ get.val  n.wt1  :=
  5  6  goto.xy
     ." Sampling freq -- original rate (s.freq.keep):"
     s.freq.keep  n.type \ get.val  s.freq.keep :=
  5  7  goto.xy 2 0 fix.format
     ." Index Interval for Sub-sampling:" i.inc  n.type \ get.val  i.inc  :=
     s.freq.keep  float  i.inc float /   6  2  fix.format
     ."    (Sub-Sampling rate: "   1 n>s s.type  ." )"
  5  8  goto.xy 4 0 fix.format
     ." Index of first point of array" i.beg  n.type \ get.val  i.beg :=
  5  9  goto.xy
     ." PSI flag (New or Old [ or old's Ridge in envelope]): "
     psi.flag s.type \ get.str psi.flag ":=
\ -----------------------------------------------------
  n.wt1  my.power
  5  10 goto.xy 7 3 fix.format
     ." Fre.beg:"
     fre.beg  n.type \ get.val  fre.beg  :=
  5  11 goto.xy 3 0 fix.format
     ." Number of freq resolution (#.fre):"  #.fre  n.type \ get.val #.fre :=
  5  12 goto.xy 7 3 fix.format
     ." Fre.inc:"  fre.inc  n.type \ get.val  fre.inc  :=
           fre.beg #.fre 1 - fre.inc * + ." (Freq. range:"
           fre.beg n.type ." --" n.type ." )"
  5  13 goto.xy
     ." ERF flag [1:fixed relation,  2:varing (Nyquist), Else:linear)] :"
                 erf.flag n.type \ get.val  erf.flag :=
  5  14 goto.xy 6 3 fix.format
     ." sai.beg [ Low frequency ] :"  sai.beg  n.type \ get.val  sai.beg
:=
  5  15 goto.xy
     ." sai.end [ High frequency ] :"  sai.end  n.type \ get.val  sai.end
:=
  5  16 goto.xy 4 0 fix.format
     ." Expand Flag(0, 1, 2):"  exp.flag  n.type \ get.val  exp.flag :=
  5  17 goto.xy
     ." MAT.pts (number of points to be used in MMA plot):"
     Mat.pts n.type \ get.val mat.pts  :=
     2 n.power ** mat.pts / mat.inc :=
     n.power \ for recover
     mat.pts my.power n.power mat.power :=
```

```
    n.power :=  \ recover
  5  18 goto.xy  1 0 fix.format
    ." MAT.beg (From 1 to " mat.inc . ." ):" 1  n.type \ get.val  mat.beg
:=
     10 5 fix.format
\ ============================================================
;
: set.cwt
\   n.wt1  my.power
\
\   5  10     goto.xy    6 2 fix.format
\      ." Fre.beg:"    7 3 fix.format
\      \ ." Fre.beg (original rate at " s.freq.keep n.type ." ):"    7 3 fix.format
\      fre.beg  get.val  fre.beg  :=
\      3 0 fix.format
\   5  11     goto.xy
\      ." Number of freq resolution (#.fre):"  #.fre  get.val #.fre :=
\   5  12     goto.xy                                    7 3 fix.format
\      ." Fre.inc:"  fre.inc  get.val  fre.inc  :=
\             fre.beg #.fre 1 - fre.inc * + ." (Freq. range:"
\             fre.beg n.type ." --" n.type ." )"
\   5  13     goto.xy
\      ." ERF flag [1:fixed relation,  2:varing (Nyquist), Else:linear)] :"
\                    erf.flag get.val  erf.flag :=      6 3 fix.format
\   5  14     goto.xy
\      ." sai.beg [ Low frequency ] :"  sai.beg  get.val  sai.beg  :=
\   5  15     goto.xy
\      ." sai.end [ High frequency ] :"  sai.end  get.val  sai.end  :=
\   4 0 fix.format
\
\   5  16     goto.xy
\      ." Expand Flag(0, 1, 2):"  exp.flag  get.val  exp.flag :=
\   5  17     goto.xy
\   \  ." MAT.pts:"  256                   get.val  mat.pts  :=
\      ." MAT.pts (number of points to be used in MMA plot):"
\      Mat.pts              get.val  mat.pts  :=
\   2 n.power **  mat.pts /         mat.inc :=
\   n.power     \ for recover
\   mat.pts  my.power n.power  mat.power :=
\   n.power  :=  \ recover
\   1 0 fix.format
\   5  18     goto.xy
\      ." MAT.beg (From 1 to " mat.inc . ." ):" 1  get.val  mat.beg  :=
\   10 5 fix.format
;

\ : ini.par

: plot.cwt.mag
  0 tmp.v.i  :=
  begin
    tmp.v.i 1 +  tmp.v.i  :=
    rt.t xsect[ ! , tmp.v.i ] dup *
    at.t xsect[ ! , tmp.v.i ] dup * +
    sqrt
```

```
    tmp.v.i  1 =
    if
      tmp.v.i yap
    else
      yadp.flag " a" "=
        if  tmp.v.i yap
        else
            tmp.v.i ydp
        then
    then
    my.pause
    tmp.v.i  #.fre =
  until
;
: plot.cwt.root \ [ 2D-array - ]
  0 tmp.v.i :=
  begin
    tmp.v.i 1 +  tmp.v.i :=
    dup
    xsect[ ! , tmp.v.i ]
    tmp.v.i  1 =
    if
      tmp.v.i yap
    else
      yadp.flag " a" "=
        if  tmp.v.i yap
        else
            tmp.v.i ydp
        then
    then
    my.pause
    tmp.v.i  #.fre =
  until
  drop
;

: plot.cwt.arg
  bwp.t
  plot.cwt.root
;
: plot.cwt.rea
  rt.t
  plot.cwt.root
;
: plot.cwt.img
  at.t
  plot.cwt.root
;
: plot.cwt.env
  wp.a
  plot.cwt.root
;
: plot.cwt.arg.res
  0 tmp.v.i :=
```

```
  begin
    tmp.v.i 1 +  tmp.v.i  :=
    at.t xsect[ ! , tmp.v.i ]                    \ Y
    rt.t xsect[ ! , tmp.v.i ]   /                \ X
    atan
    becomes> t1

    0 vi1 :=
    begin
      vi1 1 +  vi1 :=
      t1 [ vi1 ]  0  >=
      if
        at.t xsect[ ! , tmp.v.i ]   [ vi1 ] 0 >=
        if
        else
          t1 [ vi1 ]      pi      +  t1 [ vi1 ] :=
        then
      else
        at.t xsect[ ! , tmp.v.i ]   [ vi1 ] 0 >
        if
          t1 [ vi1 ]      pi      +  t1 [ vi1 ] :=
        else
          t1 [ vi1 ]      pi 2. * +  t1 [ vi1 ] :=
        then
      then
      vi1  2 mat.power ** =
    until

    t1
    tmp.v.i  1 =
    if
      tmp.v.i yap
    else
      yadp.flag " a" "=
        if  tmp.v.i yap
        else
            tmp.v.i ydp
        then
    then
    my.pause
    tmp.v.i  #.fre =
  until
;
: run.coh
  cr  " You must input LCoh-related file (eg. LCoh-0.asy)!"  s.type
;
: for.case5&6
\           5   4  goto.xy
\               ." Now.file (name.ext):"  now.file  get.str  now.file ":=
\           5   5  goto.xy
\           4 0 fix.format
\           n.wt1  my.power
\               ." Number of points of wavelet (n.wt1):"
\               2 n.power **  get.val  n.wt1 :=
\           5   6  goto.xy
```

```
\               ." Sampling freq -- original rate (s.freq.keep):"
\               s.freq.keep  get.val  s.freq.keep :=
\           5   7  goto.xy
\           2 0 fix.format
\               ." Index Interval for Sub-sampling:" i.inc  get.val  i.inc :=
\               s.freq.keep  float  i.inc float /   6  2  fix.format
\               \             ."    (Sub-Sampling rate: "   n.type  ." )"
\               ."    (Sub-Sampling rate: "   1 n>s s.type  ." )"
\           5   8  goto.xy
\           4 0 fix.format
\               ." Index of first point of array" i.beg  get.val  i.beg :=
\           5   9  goto.xy
\               ." PSI flag (New or Old [ or old's Ridge in envelope]): "
\               psi.flag  get.str  psi.flag ":=
             psi.flag "upper " R"   "=
             if
               \  " Y"    pha.switch.flag  ":=
               psi.flag   pha.switch.flag  ":=
               cr  ."                                ->PhaseSwitch (R:yes; X:No):"
               pha.switch.flag  get.str   pha.switch.flag  ":=
             else
               \  " N"    pha.switch.flag  ":=
               psi.flag   pha.switch.flag  ":=
             then
;
\ : set.act
\   gd
\   gd.txt.wnd  sc   {border}
\   3   2   goto.xy
\       ." 1:Dec&Rec, 2:M0.phase, 3:Norm.wt, 4:Entropy, 5:CWT, 6:Coh --"
\   act.flag  get.val  act.flag :=
\   act.flag
\   case
\     3  of  for.norm&ent  endof
\     4  of  for.norm&ent
\            \         5   14 goto.xy
\           cr ." #.div:"  #.div  get.val  #.div :=
\                             endof
\ \     5  of  for.case5&6
\ \             cr set.cwt
\ \             cwt             endof
\ \     6  of  for.case5&6
\ \             cr set.cwt
\ \             cr set.123
\ \             \ 5  19     goto.xy
\ \             cr
\ \             ." Real or Imag or Comp:" rea.or.com get.str "upper rea.or.com ":=
\ \             10 5 fix.format
\ \                               endof
\ \             run.coh
\     5  of  cwt.56
\            for.case5&6
\            cr set.cwt
\            cr new.pause
\            cwt             endof
```

```
\      6   of   cwt.56
\               for.case5&6
\               cr set.cwt
\               5  19 goto.xy
\               \ cr
\               ." Real or Imag or Comp:" rea.or.com s.type \ get.str "upper rea.or.com "
\               cr new.pause
\               cr set.123
\               10 5 fix.format
\               run.coh
\                                   endof
\   endcase
\ ;
: set.act
  gd
  gd.txt.wnd  sc   {border}
  3   2   goto.xy
  ." D:Dec&Rec, P:Phase.MOs, N:Norm.wt, E:Entropy, C:CWT, W:WCoh :: "
  act.flag.str s.type \ get.str  act.flag.str ":=
  act.flag.str "upper " N" "=
  if
    for.norm&ent
  then
  act.flag.str "upper " E" "=
  if
    for.norm&ent
    5  14 goto.xy
    ." #.div:"  #.div n.type \ get.val  #.div :=
  then
  act.flag.str "upper " C" "=
  if
    cwt.56
    for.case5&6
    cr set.cwt
    cr new.pause
    cwt
  then
  act.flag.str "upper " W" "=
  if
    cwt.56
    for.case5&6
    cr set.cwt
    5  19 goto.xy
    \ cr
    ." Real or Imag or Comp:" rea.or.com s.type \ get.str "upper rea.or.com ":=
    cr new.pause
    cr set.123
    10 5 fix.format
    run.coh
  then
;
: ini.par
  1024 n.wt1   :=
  200  s.freq.keep :=
  \ 5    act.flag :=
```

```
  " C" act.flag.str ":=
  5    i.inc    :=
  1.6  fre.beg :=
  40   #.fre    :=
  0.1  fre.inc :=
  1    erf.flag :=
  11.0 sai.beg :=
  5.00 sai.end :=
  256  mat.pts :=
  " N" psi.flag ":=
  " b0w6020.4" now.file ":=
  10 5 fix.format
;
ini.par
```

Lee-Word.asy

勇伯程式庫 A01518-20070125-A069 ❖ January 25, 2007 ❖ [Page count: 23]

```
: Lee-Word.asy ; \ %----- Asyst Prog
\ -----------------------------------------------------------------
\ WORD-w.Lee - 4/30/90,  03/24/91, 10/20/93, 02/20/94
\
\ 1 --> Set my def file dir
\ 2 --> For printer: 1. Set color scheme, 2. Select printer
\ 3 --> Make com file
\
\ Written by Ron Lee
\ -----------------------------------------------------------------
normal.display
  26    0  29  79  window  {c.text}   \ for clearing text
  26   19  29  79  window  {b.text}
  26    0  29  39  window  {l.text}
  26   41  29  79  window  {r.text}
  0     0  29  18  window  {f.text}
  29    0  29  79  window  {29.line.p1}  \ for BD, etc. operations
  0     0  12  79  window  {tmp.wnd}
  13    0  23  79  window  {show.wnd.l}
  15    9  22  69  window  {show.wnd.s}
\ 16    9  23  69  window  {show.wnd.s}

integer scalar 29.flag    scalar prn.flag
        scalar lc.flag
        scalar n.wt1 \ for Wavelet transform

: {29.line}
  29.flag  1  =
  if
    {29.line.p1}
  then
;
: tedious
  cr  ." prn.flag (1:epson-b ; 2:epson-c ; 3:hp-laser) = "
  #input  prn.flag :=
; tedious
1 string  my.pause.flag

integer scalar bg.color
        scalar bell.flag
        scalar pr.bg.color
        scalar fg.color
        scalar gr.color
        scalar em.color
        scalar wa.color
        scalar t1.color
        scalar t2.color
        scalar outline.color
        scalar my.axis.color
        scalar def.val.flag
        scalar s.freq
        scalar prn.no
```

```
        scalar color.sft
        scalar tmp.c            \ current color for foreground and curvereal
        scalar prn.scale
        scalar 2.wnd.flag
        scalar lc1   scalar lc2   scalar lc3   scalar lc4
        scalar lc5   scalar lc6   scalar lc7   scalar line.c
        scalar fft.sec   scalar n.power
        scalar menu.flag  scalar grid.flag
1 string pcx.h.c.str
: my.power
  trunc
  0  n.power  :=
  begin
    float  2.  /   trunc
    dup
    dup  1  >=  if   n.power   1 +   n.power :=  then
    1  =
  until
  drop
;

0 menu.flag :=   \ default
1 string wnd.str
1 string y.lab.str
1 string offset.str

screen.clear
cr  ." prn.flag"  prn.flag  .   pause

64 string file.ext
: hp.laser
  0              color.sft           :=
  1000           prn.scale           :=
  145            prn.no              :=
  140            prn.no              :=   \ HP
  0              pr.bg.color         :=   \          background color
  11 lc1  :=   15 lc2  :=   10 lc3  :=
  13 lc4  :=   12 lc5  :=   14 lc6  :=   15 lc7  :=
  pr.bg.color    vuport.color
  7              my.axis.color       :=
  my.axis.color axis.color
  12             t1.color            :=
  14             t2.color            :=
  0  bg.color          :=  \          background color
  3  fg.color          :=  \          foreground color
  14 gr.color          :=  \
  10 em.color          :=  \          emphasized color
  10 outline.color     :=
  13 wa.color          :=
;
: epson.black
   0              color.sft           :=
   2000           prn.scale           :=
   105            prn.no              :=
   0              pr.bg.color         :=   \          background color
```

```
    11 lc1 :=   15 lc2 :=   10 lc3 :=
    13 lc4 :=   12 lc5 :=   14 lc6 :=   15 lc7 :=
    pr.bg.color   vuport.color
    7             my.axis.color      :=
    my.axis.color axis.color
    12            t1.color           :=
    14            t2.color           :=
    0  bg.color         :=  \        background color
    3  fg.color         :=  \        foreground color
    14 gr.color         :=  \
    10 em.color         :=  \        emphasized color
    10 outline.color    :=
    13 wa.color         :=
;
: epson.color
    1             color.sft          :=
    2000          prn.scale          :=
 \  105           prn.no             :=
    140           prn.no             :=  \ HP
    15            pr.bg.color        :=  \        background color
 \  11 lc1 :=    0 lc2 :=   10 lc3 :=   13 lc4 :=   12 lc5 :=    9 lc6 :=
     9 lc1 :=   12 lc2 :=   10 lc3 :=
    13 lc4 :=    0 lc5 :=   11 lc6 :=    0 lc7 :=
    pr.bg.color   vuport.color
    0             my.axis.color      :=
    my.axis.color axis.color
    10            t1.color           :=
    0             t2.color           :=
    0  bg.color         :=  \        background color
    3  fg.color         :=  \        foreground color
    14 gr.color         :=  \
    10 em.color         :=  \        emphasized color
    0  outline.color    :=
    13 wa.color         :=
;
: select.color
  prn.flag
  case
    1 of  epson.black  endof
    2 of  epson.color  endof
    3 of  hp.laser     endof
  endcase
; select.color

: my.def.colors
  color.on
  15 color
  bg.color background
  fg.color foreground
; my.def.colors

: l.pro
  105  prn.no  =
  if
    dir.lee.txt " prompt-e.txt"  "cat  defer> load   \ for epson
```

```
  else
    dir.lee.txt " prompt.txt"  "cat  defer> load
  then
  dir.lee.txt " bell.txt"  "cat  defer> load
; l.pro

: go00or11
  outline.color  cursor.color
  \ normal.coords   1.   1.   position
  normal.coords                 position
  world.coords
;
: go00  0  0   go00or11 ;
: go11  1  1   go00or11  "  " label ;

integer scalar tmp.v.i
        scalar tmp.v.i.1
        scalar tmp.v.i.2
        scalar tmp.v.i.3
        scalar tmp.v.i.4
        scalar tmp.v.i.9

real    scalar y.lab.pos
dp.real scalar tmp.v
        scalar tmp.v.2
        scalar tmp.v.3
        scalar tmp.v.4
        scalar tmp.v.9

60 string tmp.s
60 string tmp.s.9
60 string tmp.s.8
1  string xeq.flag

: pro.sign.old  \   see prompt.txt
  \  "dir   " >>" "cat     "type
  15  foreground
  ." >>"
  \ \ \  my.def.colors
;
\ dir.lee.txt " prompt.txt"  "cat  defer> load
\ dir.lee.txt " prompt-e.txt"  "cat  defer> load   \ for epson

: b.t         blink.toggle ;
: inv.of      inverse.off ;
: inv.on      inverse.on ;
: int.of      inten.off ;
: int.on      inten.on ;
: inv.text    inverse.on  "type  inverse.off            ;
: int.text    inten.on    "type  inten.off              ;
: wan.text    wa.color  foreground  b.t    "type  b.t my.def.colors ;
: n.input     1  background  11  foreground  #input  my.def.colors ;
: s.input     200  msec.delay
              bell.1sl  4  background  11  foreground  "input  my.def.colors ;
: n.type      1  background  15  foreground  .       my.def.colors ;
```

```
: s.type      4   background  15  foreground  "type   my.def.colors ;
: s.type.2    12   background  15  foreground  "type   my.def.colors ;
: s.type.3    5    background  14  foreground  "type   my.def.colors ;
: s.type.4    13   background  14  foreground  "type   my.def.colors ;
: cdt         dir.lee.txt "len 1 - "left  defer> chdir ;
: cdd         dir.lee.dat "len 1 - "left  defer> chdir ;
: cda         dir.lee.asc "len 1 - "left  defer> chdir ;
: sc          screen.clear ;
: vc          vuport.clear ;
: ftop        {29.line} screen.clear
              ." ---> :::: "        cda  s.input  defer>   pcx>vup
              cdt screen.clear previous.window ;
: ptof        {29.line} screen.clear
              ." ---> file : !! "  cda  s.input  defer>   vup>pcx
              cdt screen.clear previous.window ;
: ptof.bat    pcx.h.c.str  " y"  "=  if
              {29.line} screen.clear
              tmp.s  ":=   dir.lee.asc  tmp.s  "cat  tmp.s  ":=
              ." ---> file : ! "  tmp.s s.type   tmp.s  defer>   vup>pcx
              500 msec.delay
              cdt screen.clear previous.window  then ;
\ -----------------------------------------------------------------
integer scalar n>s.d1  scalar n>s.d2  scalar n>s.d3
real    scalar n>s1    scalar n>s2
1 string  n>s.s

: n>s
  n>s.d3          :=  \ #.digits
  n>s1            :=
  "null  n>s.s   ":=
  n>s1  0.  <  if   " -"  n>s.s  ":=  then
  n>s1  abs  n>s1  :=
  7 n>s.d1    :=
  begin
    \ 11 5 fix.format
    n>s1    10. n>s.d1    **   /  0.00001 +  trunc    n>s.d2    :=
    \ cr ." --1--" n>s1   n.type  n>s.d1    .   n>s.d2     n.type
    n>s.d1    1 -   n>s.d1    :=
    n>s.d2     0  <>
    n>s.d1     0  <    or
  until
  n>s.d2      48 +   ascii"
  n>s.d1     -1  =    n>s.d3       1 =   and
  if   exit   then
  n>s1       n>s.d2      10.  n>s.d1     1 + **    *  -  n>s1  :=
  n>s.d3        0 <>
  n>s.d1     -1 =   and
  if
    " ."   "cat   \ " ."  s.type
  then

  begin
    n>s1    10.  n>s.d1    **  /  0.00001 +  trunc     n>s.d2    :=
    \ cr  ." --2--" n>s1   n.type  n>s.d1    .   n>s.d2     n.type
    n>s.d1    1 -   n>s.d1    :=
```

```
    n>s.d1    -1 =   n>s.d3      1 =  and
    if  " 0"  "cat   exit  then
    n>s.d3        0          >
    n>s.d1      n>s.d3       <  and
    if
      n>s.d2      48 +   ascii"
      "cat
      begin
        " 0"  "cat
        \ cr  ." --3--"  n>s1   n.type  n>s.d1    .   n>s.d2     n.type
        n>s.d1    1 -   n>s.d1    :=
        n>s.d1      0  <
        if
          exit
        then
        1  1 <>
      until
    then
    n>s1   n>s.d2      10.  n>s.d1    1 + **   *  -   n>s1  :=
    n>s.d1      -2  =
    n>s.d3          0  =  and
    if
    else
      n>s.d2      48 +   ascii"   "cat
    then
  n>s.d1     -1 =
  if
    n>s.d3        0  <
    if
      " ."   "cat
    then
    \  " ."   s.type
  then
    n>s.d1     n>s.d3        <
  until
  n>s.s  " -"  "=  if  " -"  "swap  "cat  then
;
\ ----------------------------------------------------------------

: dashed1                                    solid  ;
: dashed2    0.019  0.004  0.019  0.004  dashed ;
: dashed3    0.011  0.004  0.011  0.004  dashed ;
: dashed4    0.006  0.005  0.006  0.005  dashed ;
: dashed5    0.003  0.003  0.003  0.003  dashed ;
: dashed6    0.014  0.004  0.005  0.004  dashed ;
: line.type  \ "  1--6"  s.type
             6    modulo
             case
        \    1  of  11  solid    endof  \ blue
        \    2  of  14  dashed2  endof  \ yellow
        \    3  of  10  dashed3  endof  \ green
        \    4  of  13  dashed4  endof  \ purple
        \    5  of  12  dashed5  endof  \ red
        \    0  of  15  dashed6  endof  \ white
             1  of  lc1  dup    solid    endof
```

```
             2  of  lc2  dup    dashed2  endof
             3  of  lc3  dup    dashed3  endof
             4  of  lc4  dup    dashed4  endof
             5  of  lc5  dup    dashed5  endof
             0  of  lc6  dup    dashed6  endof
             endcase
             line.c  :=
             color     ;
\ : my.outline outline.color color outline ;
: my.outline outline.color color  solid  outline ;
: yap          line.type  y.auto.plot   my.outline ;
: ydp          line.type  y.data.plot   my.outline ;
: xyap         line.type  xy.auto.plot  my.outline ;
: xydp         line.type  xy.data.plot  my.outline ;
: fa           def.vuport normal.display  forget.all ;
: cd           cr ." Dir: "   s.input defer>   chdir ;
: nd           normal.display   install pro.sign.2   in prompt.xeq ;
: sd           stack.display    install pro.sign.2   in prompt.xeq ;
: sfp          cr ." File.parse: "  n.input  set.file.parse ;
: l            cr ." LOAD: " dir.lee.txt   s.input  "cat  defer> load ;
: l.o          cr ." Overlay name:  "  dir.asy.sys
             s.input  "cat  " .sov"  "cat  defer> load.overlay   cdt ;

1  string  x.axis.flag
1  string  y.axis.flag
\ -----------------------------------------------------------------
: xlyl
  horizontal    linear          " l"    x.axis.flag    ":=
  vertical      linear          " l"    y.axis.flag    ":=
;
: xlyo
  horizontal    linear          " l"    x.axis.flag    ":=
  vertical      logarithmic     " o"    y.axis.flag    ":=
;
: xoyl
  horizontal    logarithmic     " o"    x.axis.flag    ":=
  vertical      linear          " l"    y.axis.flag    ":=
;
: xoyo
  horizontal    logarithmic     " o"    x.axis.flag    ":=
  vertical      logarithmic     " o"    y.axis.flag    ":=
;
\ -----------------------------------------------------------------
: xnyn
  horizontal    axis.fit.on
  vertical      axis.fit.on
;
: xnyf
  horizontal    axis.fit.on
  vertical      axis.fit.off
;
: xfyn
  horizontal    axis.fit.off
  vertical      axis.fit.on
;
```

```
: xfyf
  horizontal    axis.fit.off
  vertical      axis.fit.off
;
\ -----------------------------------------------------------------

: get.val
              dup       n.type  ."  "
              250 msec.delay
              bell.1sl
              n.input
              if
                swap drop
              then ;
: get.str
              "dup      s.type
              s.input
              "len  0  =
              if      "drop
              else    "swap  "drop
              then
;
\ -----------------------------------------------------------------
: for.xyadp.p
  -1 4  fix.format
  {29.line}  sc
  unrot    \  [ x  y  1  -  1  x  y ]
  " (bg.pt):"  s.type  1       get.val  tmp.v.i    :=
  " (end.pt):" s.type  []size  get.val  tmp.v.i.2  :=
  " (inc):"    s.type  1       get.val  tmp.v.i.3  :=
  sub[ tmp.v.i , tmp.v.i.2  tmp.v.i - 1 + tmp.v.i.3 / , tmp.v.i.3 ]  swap
  sub[ tmp.v.i , tmp.v.i.2  tmp.v.i - 1 + tmp.v.i.3 / , tmp.v.i.3 ]  swap
  rot      \  [ 1  x  y  -  x  y  1 ]
  screen.clear  previous.window
;
: xyap.p
  for.xyadp.p
  xyap
;
: xydp.p
  for.xyadp.p
  xydp
;

: mkl.p
              swap
              case
               1  of lc1 swap solid    1 -  2 *  endof
               2  of lc2 swap dashed2  1 -  2 *  endof
               3  of lc3 swap dashed3  1 -  2 *  endof
               4  of lc4 swap dashed4  1 -  2 *  endof
               5  of lc5 swap dashed5  1 -  2 *  endof
               6  of lc6 swap dashed6  1 -  2 *  endof
              endcase
              swap  color  key.line     ;
```

```
: mkl         \  interactive
              \  60  60  key.orig
              60  12  key.size
              " (key.orig/key.size) Line_#"      s.type  1    get.val
              "  Line_pos"  s.type  dup  get.val
              ."  [...]"    s.input
              mkl.p  ;
: mkl.bat     \  [ line.# , line.pos , " text"  -- ]
              mkl.p  ;
\ -----------------------------------------------------------------
: for.a.c.c&c.c.c
  []size integer  ramp          \ r-array size ramp    : (N + M - 1)
  rot
  []size swap  drop             \ input-array size     : (N)
  -                             \ shifted array index : -(N-1) to (M-1)
  s.freq float  /               \ convert to time
  swap
  " index_a , r_a created"  s.type
;
: a.c.c  \  [  array -  index_array  r_array  ]
  dup
  auto.correlation.coeff        \ give r-array
  for.a.c.c&c.c.c
;
: c.c.c  \  [ x_array  y_array  -  index_array  r_array  ]
  swap dup  rot  \  -> y x -> y x x  -> x x y
  \  dup  unrot  \  -> x y y -> y x y
  cross.correlation.coeff       \ give r-array
  for.a.c.c&c.c.c
;
\ -----------------------------------------------------------------
\ actual # of sec's = 2 * #.sec
\ -----------------------------------------------------------------

integer scalar  tmp.v.i.1  scalar #.fft.pts

: my.p.s.bat  \ [ array , sec_size , #.sec , window_type -- p.s.array ]
  tmp.s       ":=  \ window.type
  tmp.v.i.9    :=  \ #.sec
  tmp.v.i.1    :=  \ #.fft.pts
  tmp.v.i.1  #.fft.pts :=
  tmp.v.i.9   1   >
  if
    []size    tmp.v.i  :=
    tmp.v.i  float   tmp.v.i.1 float 2. /   /
    floor 1 -      tmp.v.i.2  :=
\   tmp.v.i    tmp.v.i.1  /  2 *    1 -     tmp.v.i.2  :=
\   tmp.v.i    tmp.v.i  tmp.v.i.1  /  tmp.v.i.1 *    -
\   tmp.v.i.1  2 /
\   >
\   if
\      tmp.v.i.2 1 +   tmp.v.i.2 :=
\   then
    tmp.v.i.2  tmp.v.i.9 >
    if
```

```
      tmp.v.i.9  tmp.v.i.2 :=
    then
  else
      tmp.v.i.9  tmp.v.i.2 :=
  then

  tmp.v.i.2   1  >
  if
    2  tmp.v.i.3    :=                                      \  with overlap
  else
    1  tmp.v.i.3    :=                                      \  no   overlap
  then

  0                   \ initial sum
  swap                \ --> 0 , array

  tmp.v.i.2  1 +   1
  do
    \ cr i .
    dup
    sub[ i 1 -    tmp.v.i.1  tmp.v.i.3 /   *    1 +   , tmp.v.i.1 ]
    \  dir.asy.sys  " timewins.sov"  "cat  defer>  load.overlay
    tmp.s "upper " X"  "=   if                          then
    tmp.s "upper " K"  "=   if   blackman.window.apply  then
    tmp.s "upper " B"  "=   if   bartlett.window.apply  then
    tmp.s "upper " N"  "=   if   hanning.window.apply   then
    tmp.s "upper " M"  "=   if   hamming.window.apply   then
    power.spectrum
    rot  \ --> array , p_s , 0
    +
    swap
  loop
  drop

  2. *          \ Be sure to use only half length ( symmetry)
  []copy
  \ dup  [ 1              ]   2. /   over [ 1              ]  :=
  \ dup  [ tmp.v.i.1 2 / ]   2. /   over [ tmp.v.i.1 2 / ]  :=
  tmp.s  "upper " X"  "=
  if                      \ Correct the weight of data window --> make larger
    \ no operation
  else
    tmp.v.i.1
    tmp.s "upper " K" "= if   blackman.window  then
    tmp.s "upper " B" "= if   bartlett.window  then
    tmp.s "upper " N" "= if   hanning.window   then
    tmp.s "upper " M" "= if   hamming.window   then
    2. ** []sum tmp.v.i.1 / \ The weight is not normalized --> use division
    /
  then
  tmp.v.i.2  float  /     \ averaged by actual #.sec
\ tmp.v.i.1  2 **   /     \
  tmp.v.i.1  1 **   /     \ normalized by #.fft.pts  -->  actual p.s density
\ release.overlay
  sub[ 1 , tmp.v.i.1  2  / ]
```

```
;

: get.x.time   []size  ramp  s.freq float /   swap  ;
\ : get.x.freq   []size  ramp  s.freq float *  #.fft.pts float /  swap ;
: get.x.freq   []size  dup  ramp  s.freq float *  swap float 2. *  /  swap ;
\ release.overlay
: my.p.s      \ [ array , sec_size , sections , window_type -- p.s.array ]
  cr " [ A , fft.pts , #.sec , wnd -- p.s.a ]" s.type
  my.p.s.bat
  tmp.v.i.2  dup fft.sec :=   n.type   tmp.s  "dup wnd.str ":=   s.type
;
\ ----------------------------------------------------------------
: c.2    \ [ x y   -  x y x y ]
  swap   \   y x
  dup    \   y x x
  rot    \   x x y
  dup    \   x x y y
  rot    \   x y y x
  swap   \   x y x y
;
\ ----------------------------------------------------------------

\ ----------------------------------------------------------------
: out.file
  cr
  ." ---> File : "  s.type.2  s.input  cr
  ." Be sure to   "  b.t ." Out>file.close " b.t cr
  dir.lee.txt  "swap  "cat
  "dup "type   ."   <--- Created"
  defer>  out>file
  \  out>file.close
;
\ ----------------------------------------------------------------

graphics.display

vuport  h.port  0.0000 0.1500 vuport.orig  1.0000 0.8500 vuport.size  outline
vuport  v.port  0.1800 0.1500 vuport.orig  0.6400 0.8500 vuport.size  outline
vuport  l.port  0.0000 0.1500 vuport.orig  0.4800 0.8500 vuport.size  outline
vuport  r.port  0.5200 0.1500 vuport.orig  0.4800 0.8500 vuport.size  outline
vuport  u.port  0.0000 0.5800 vuport.orig  1.0000 0.4200 vuport.size  outline
vuport  d.port  0.0000 0.1500 vuport.orig  1.0000 0.4200 vuport.size  outline
vuport  f.port  0.2750 0.0334 vuport.orig  0.7250 0.9666 vuport.size  \ outline
                \ (0 to 28 rows)
: port.axis.1
          horizontal  normal.coords  0.85  0.10  label.scale.point  world.coords
                                     0     2     label.points
          vertical    normal.coords  0.07  0.87  label.scale.point  world.coords
                                     0     2     label.points ;
: port.axis.2
          horizontal  normal.coords  0.85  0.10  label.scale.point  world.coords
                                     0     2     label.points
          vertical    normal.coords  0.04  0.87  label.scale.point  world.coords
                                     0     2     label.points ;
: port.color
```

```
          tmp.c foreground  tmp.c color
          pr.bg.color vuport.color  my.axis.color  dup  axis.color label.color
          go11 ;
: hp      h.port  {b.text}  t1.color tmp.c :=  port.axis.1  0.02  y.lab.pos
:=
          port.color    0  2.wnd.flag := ;
: vp      v.port  {b.text}  t2.color tmp.c :=  port.axis.2  0.02  y.lab.pos
:=
          port.color    0  2.wnd.flag := ;
: lp      l.port  {l.text}  t1.color tmp.c :=  port.axis.2  0.95  y.lab.pos
:=
          port.color    1  2.wnd.flag := ;
: rp      r.port  {r.text}  t2.color tmp.c :=  port.axis.2  0.95  y.lab.pos
:=
          port.color         1  2.wnd.flag := ;
: up      u.port  {l.text}  t1.color tmp.c :=  port.axis.1  0.02  y.lab.pos
:=
          port.color    1  2.wnd.flag := ;
: dp      d.port  {r.text}  t2.color tmp.c :=  port.axis.1  0.02  y.lab.pos
:=
          port.color    1  2.wnd.flag := ;
: fp      f.port  {f.text}  t1.color tmp.c :=  port.axis.1  0.02  y.lab.pos
:=
          port.color    0  2.wnd.flag := ;
: tc      {c.text}  screen.clear ;

\  lp       outline
\  rp       outline
\  up       outline
\  dp       outline
\  hp       outline
\  vp       outline
  {c.text}  echo.off  screen.clear      {b.text}  echo.on

: lab
  dup  999  <>
  if
  -1  2  fix.format
  {29.line} screen.clear
  tmp.c          color
  tmp.c  foreground  ." X_pos:"    0   get.val  tmp.v    :=
  tmp.c  foreground  ."  Y_pos:"   0   get.val  tmp.v.2  :=
  tmp.c  foreground  ."  Center:"  0   get.val  tmp.v.3  :=
  tmp.c  foreground  ."  [...]"        s.input  tmp.s   ":=
  tmp.c  foreground
  tmp.s
  tmp.v    tmp.v.2          position
  tmp.v.3  0  <>
  if   centered.label
  else label
  then
  normal.coords
  1.  1.  position
  world.coords
  screen.clear  previous.window
```

```
  bell.1ll
  else
    drop  "drop "drop
  then
;

: labxy.stack  \ [ x.lab  y.lab  y.pos -- ]
  dup  999  <>
  if
    "swap
    {29.line} screen.clear
    -1  2  fix.format
\   tmp.c            color
    0                color
    normal.coords
    0                            char.dir
    0                            label.dir
    0.55    0.05                 position
                                 centered.label   \ [ "X ]
    90                           label.dir
    90                           char.dir
    \ tmp.v  0.55                position
           0.55                  position
                                 centered.label   \ [ "Y ]
    0                            char.dir
    0                            label.dir
    1.  1.                       position
    world.coords
    screen.clear  previous.window
    bell.1sl
  else
    drop  "drop "drop
  then
;

: labxy
  tmp.c   color.sft +     foreground
  ." [X_s]"                s.input \ tmp.s   ":=
  tmp.c   color.sft +     foreground
  ."  [Y_s]"               s.input \ tmp.s   ":=
  tmp.c   color.sft +     foreground
  ."  X_pos (0.02/0.95):"  0.02    get.val   \  tmp.v    :=
  tmp.c   color.sft +     foreground
  labxy.stack
;

\ : prompt.gd   ." >>" ;

: gd
  install pro.sign in prompt.xeq
  graphics.display
\  lp       outline
\  rp       outline
\  up       outline
\  dp       outline
```

```
\  hp       outline
\  vp       outline
  hp
  vuport.clear
  {29.line}
  screen.clear
  \ ." H-V  L-R  U-D   F    port"
;

dir.asy.sys " genbd.sov"  "cat  defer>  load.overlay
: hp.printer
  hp.laser
  145           bd.printer
  \ pr_clr_b      bd.pat
  pr_usr_bg     pr.bg.color   bd.uclr
;
: epson.printer.b
  epson.black
  105           bd.printer
  \ pr_clr_b      bd.pat
  pr_usr_bg     pr.bg.color   bd.uclr
;
: epson.printer.c
  epson.color
  prn.no        bd.printer
  pr_clr_w      bd.pat
  pr_usr_bg     pr.bg.color   bd.uclr
;
: select.printer
  prn.flag
  case
    1 of  epson.printer.b  endof
    2 of  epson.printer.c  endof
    3 of  hp.printer       endof
  endcase
;
: dump.p
  {29.line} screen.clear
  tmp.c            foreground
  tmp.c            color
  cr
  " ---> File : "   s.type.2
  s.input
  tmp.c            foreground
  "dup
  dir.asy.sys  " genbd.sov"  "cat  defer>  load.overlay
  cdt sc
  select.printer
  landscape     bd.page    \ <-- Compatable
  1500          1500 bd.scale
  dir.lee.txt  "swap  "cat  "type   ."   <--- Creating.  Please wait"
  bd.vuport     ?drop
  release.overlay
  screen.clear  previous.window
  bell.111
```

```
;

: dump.p.o.stack
  {29.line} screen.clear
  tmp.c          foreground
  \ "dup
  dir.asy.sys  " genbd.sov"  "cat  defer>  load.overlay
  cda sc
  portrait  bd.page
  prn.scale   dup   bd.scale
  dir.lee.asc  "swap  "cat  "type   ."   <--- Creating.  Please wait"
  bd.vuport   ?drop
  release.overlay
  cdt screen.clear  previous.window
  bell.1ll
;
: dump.p.o.f
  {29.line} screen.clear
  tmp.c          foreground
  tmp.c          color
  cr
  " ---> File : "   s.type.2
  s.input
  "dup
  dump.p.o.stack
;
: dump.p.o.p   \ to printer   \ [ file.ext - ]
  {29.line} \ screen.clear
  dir.asy.sys  " genbd.sov"  "cat  defer>  load.overlay
  \ " lpt1" "dup
  "dup  tmp.v.i  0 n>s  " ."  "cat  "swap  "cat
  " lpt1"
  prn.no         bd.printer
  dump.p.o.stack
;
: pbmp      \ to printer
  {29.line} \ screen.clear
  dir.asy.sys  " genbd.sov"  "cat  defer>  load.overlay
  " lpt1"
  prn.no         bd.printer
  gd hp outline
  ftop   " ----- Please Wait ------" s.type
  bd.vuport
  my.def.colors
  release.overlay
;

release.overlay
\ ------------------------------------------------------------------
my.def.colors

dir.asy.sys  " PCX.sov"  "cat  defer> load.overlay
: date.time
  graphics.display
  vp
```

```
  my.def.colors
  dir.asy.sys   " PCX.sov"    "cat   defer> load.overlay
  dir.lee.txt   " boot.pcx"   "cat   defer> pcx>vup
  vp
  \  "     -----------    WELCOME    ------------"  s.type bell.1ll \ pckey ?drop drop
  cr   "          ---------    WELCOME    --------"  s.type
  \ ." ---------  <"  "date "type ." > "
  cr  ."           <"  "date "type ." > "
  "time  "type     bell.1ll
  cr
  2500 msec.delay
  release.overlay
  \  my.def.colors
  sd cdt
;
release.overlay
\ ------------------------------------------------------------------
integer dim[ 6 ] array    acq.input
64 string mycase
32 string mydate
32 string mytime
64 string specifics

INTEGER SCALAR T.RATE
        SCALAR s.length
     \  SCALAR s.freq
        SCALAR s.chnls
        SCALAR p.chnl
        SCALAR p.points
        scalar eps.flag

: atof.p1
  dup  \  for .p3
  dup          []dims  tmp.v.i    :=  drop
  tmp.v.i  1  =  if  dup              []size   tmp.v.i.1 :=   drop  then
  tmp.v.i  2  =  if  dup           1  []dim    tmp.v.i.1 :=   drop
                                   2  []dim    tmp.v.i.2 :=   drop  then
;
: atof.p2  \ for interaction
  cr " ARRAY ---> FILE !! : "
  s.type.2
  bell.2sl
  s.input  \ cr
  dir.lee.asc  "swap  "cat   tmp.s  ":=
;
regular.datafile  \ dummy
file.template
   8 comments
   integer dim[ 6 ] subfile
   real    dim[ 9 ] subfile   \ dummy
end
dir.lee.asc  " trash"  "cat  tmp.s  ":=
: atof.p3  \ c.d.tmp file
  cr
```

```
  tmp.s             defer>  file.create
  tmp.s             defer>  file.open
                                                        mycase           1 >comment
                                                        mydate           2 >comment
                                                        mytime           3 >comment
                                                        specifics        4 >comment
                                                        datafile.dat.p23 5 >comment
  1 subfile  acq.input     array>file
  2 subfile                array>file
  file.close
  " OK" s.type   tmp.s  "type
;
: atof.p4
  regular.datafile
  file.template
  8 comments
  integer dim[ 6 ] subfile
     tmp.v.i  1  =
     if
       real    dim[ tmp.v.i.1 ] subfile
     else
       real    dim[ tmp.v.i.1 , tmp.v.i.2 ] subfile
     then
  end
;
: atof  \ [ array -- ]
  atof.P1
  atof.P2
  atof.p4
  atof.p3
;
: atof.bat  \ [ array , "name.ext" ---  ]
  atof.P1
  dir.lee.asc  "swap   "cat  tmp.s  ":=
  atof.p4
  atof.p3
;

\ ----------------------------------------------------------------
: ftoa.p1
  cr  " File( CDA/CDD? ) --> Unnamed ary : "
  s.type  bell.2sl
  s.input   tmp.s  ":=   cr
;
: ftoa.p2
  tmp.s  defer> file.open
  1 comment>   mycase           ":=
  2 comment>   mydate           ":=
  3 comment>   mytime           ":=
  4 comment>   specifics        ":=
  5 comment>   datafile.dat.p23 ":=
  1 subfile    acq.input  file>array
               acq.input  [ 1  ] t.rate       :=
               acq.input  [ 2  ] s.length     :=
               acq.input  [ 3  ] s.freq       :=
```

```
               acq.input  [ 4  ] s.chnls      :=
               acq.input  [ 5  ] p.chnl       :=
               acq.input  [ 6  ] p.points     :=
  2 subfile   file>unnamed.array
  file.close
  tmp.s   "type
  " OK" s.type  ."    "
  cdt
;
: ftoa.bat  \  [ "name.ext"  -- array ]
  tmp.s   ":=
  ftoa.p2
;
: ftoa  \ for general directory
  ftoa.p1
  ftoa.p2
;
\ ------------------------------------------------------------
\ array(s) to ASCII file
\ ------------------------------------------------------------
integer scalar console.flag
        scalar format.flag
        scalar atoa.flag
1              console.flag  :=
1              format.flag   :=       \ 1:fix.format1 ;     2:sci.format

: choose.format
  12   5   fix.format
;
: atoa.p
  dir.lee.asc   "swap   "cat   cr
  "dup
  "dup    defer>  delete  cr

  sc  0  0  goto.xy      "dup  s.type   ."   <--- Creating"
  defer>  out>file
  console.flag  0  =  if  console.off  then
  4   0  fix.format
  []size  dup  .    cr
  1 +  1
  do
    4   0  fix.format
    i  .   ."   "
    tmp.v.i 1 +  1
    do
    \      format.flag  1 = if     10  4  fix.format
    \                       else   11  3  sci.format
    \                       then
      choose.format
      tmp.v.i  i - 1 +     pick  [ j ]  .
    loop  cr
  loop
  tmp.v.i 1 +  1
  do
    drop
```

```
  loop
  out>file.close
  console
  "type   ."   <--- Created"
;
: atoa.bat   \ [ array,   ... , file_name , #_array --  ]
  tmp.v.i   :=
  atoa.p
;
: atoa
  cr
  ." Set  Choose.format  first"    cr
  " #col:" s.type   1  get.val   tmp.v.i   :=
  " ---> File : "  s.type.2    " tmp.dat"  get.str
  atoa.p
;
\ -----------------------------------------------------------------
1024  stack  my.stack
      scalar #.xy

0     #.xy   :=
integer scalar file.no.g1  scalar file.no.g2  scalar file.no.g3
        scalar file.no.g4  scalar file.no.g5  scalar file.no
1    file.no.g1   :=  1    file.no.g2   :=
1    file.no.g3   :=  1    file.no.g4   :=   1    file.no.g5   :=

: call.f.n.g1    file.no.g1  96 +   file.no.g1  1 +   file.no.g1  :=   ;
: call.f.n.g2    file.no.g2  96 +   file.no.g2  1 +   file.no.g2  :=   ;
: call.f.n.g3    file.no.g3  96 +   file.no.g3  1 +   file.no.g3  :=   ;
: call.f.n.g4    file.no.g4  96 +   file.no.g4  1 +   file.no.g4  :=   ;
: call.f.n.g5    file.no.g5  96 +   file.no.g5  1 +   file.no.g5  :=   ;
: call.f.n       file.no     96 +   file.no     1 +   file.no     :=   ;
: make.abc.all   \  [  file.no --  ]   ( "extention - )
  {29.line}   0   0  goto.xy
  \ ascii"              \   p   a
  0  n>s
  atoa.flag  1  =
  if
    "dup  "over  "swap       \   p   a   a  ->  p a a p ->  p a p a
  then
  dir.lee.asc    "swap "cat     " ." "cat    "swap "cat
  0  0  goto.xy  ." --->"    "dup  s.type
  hp          defer>    vup>pcx
  atoa.flag  1  =
  if
    {29.line}
    " -" "cat    "swap "cat     " .dat" "cat
    #.xy  2  *
    atoa.bat
    0  #.xy  :=
  then
  cdt         sc        previous.window
;
: make.abc.g1      call.f.n.g1     make.abc.all   ;   \  [  - ]  ( "ext - )
: make.abc.g2      call.f.n.g2     make.abc.all   ;
```

```
: make.abc.g3       call.f.n.g3      make.abc.all    ;
: make.abc.g4       call.f.n.g4      make.abc.all    ;
: make.abc.g5       call.f.n.g5      make.abc.all    ;
\ : make.abc          call.f.n         make.abc.all    ;
: make.abc          file.no           make.abc.all
              file.no 1 +   file.no :=    ;

: ftop.bat   {29.line} screen.clear
             " ---> :::: "  s.type.2   cda  "dup  s.type  defer> pcx>vup
             cdt               previous.window
;
\ -----------------------------------------------------------------

\ -----------------------------------------------------------------
\ DO NOT USED IN WINDOWS
\ dir.asy.sys  " runtime.aov"  "cat  defer> load.overlay
\ banner: my.banner
\  graphics.display
\  vp
\  \ dir.asy.sys  " PCX.sov"   "cat  defer> load.overlay
\  dir.lee.txt  " boot.pcx"   "cat  defer> pcx>vup
\  \ release.overlay
\  bell.1LL
\  \ cr ." ---------  <"  "date "type ." > "
\  \  "time  5 "left  "type  ."   LEE  ---------"   pckey ?drop  drop
\  cr ." -----------  <"  "date "type ." > "
\  "time  5 "left  "type  ."   -----------"   pckey ?drop  drop
\  \ 4000 msec.delay
\  nd
\ ;banner
\ release.overlay
\ ------------------------------------------------------------------------------

: run.menu
  dir.lee.txt  " lee.men"      "cat  defer> load
  cr  prompt.xeq space
;
: run.acq.spe
  dir.lee.txt  " lacq-spe.men"      "cat  defer> load
  cr  prompt.xeq space
;
\ : lref-0 ;  \ dummy
\ : lsta-0 ;
: run.lref-0
  dir.lee.txt  " lref-0.txt"      "cat  defer> load
  \ forget lref-0
  cr  prompt.xeq space
;
\ F1 shift  function.key.does  run.menu

01 control.key.does  run.menu
17 control.key.does  run.acq.spe

install nop        in bell
install pro.sign   in prompt.xeq
```

```
" y"  my.pause.flag   ":=
: my.pause
  my.pause.flag  " y"   "=
  if
    \    bell.flag
    \    1  bell.flag :=
    bell.1sl
    {29.line}  sc
    0  1  goto.xy    " <Press Key>"  s.type
    pckey  drop ?drop
    0  1  goto.xy   ."              "
    previous.window
    \    bell.flag :=  \ restore
  then
;

 1 string cdtnow.flag.str
32 string cdt.now.str
32 string cda.now.str
32 string cdd.now.str
32 string cdf.now.str
\ my.pause
: cdtNow
  cdt.now.str "len 1 - "left defer> chdir
;
: cdaNow
  cda.now.str "len 1 - "left defer> chdir
;
: cddNow
  cdd.now.str "len 1 - "left defer> chdir
;
: cdfNow
  cdf.now.str "len 1 - "left defer> chdir
;

: SWD \ set working directories
  cdt load SWD.asy
;
\ 12 string tmp.s.defer
\ : defer.load.get \ [ " - ]
\   get.str tmp.s.defer ":=
\   tmp.s.defer defer> load
\ ;
: stack.data ; \ dummy for input gathering of asy data, wks files
: stack.load ; \ dummy for input gathering of asy txt files

token   t1   exp.mem>   t1
token   t2   exp.mem>   t2
token   t3   exp.mem>   t3
token   t4   exp.mem>   t4
token   t5   exp.mem>   t5
token   t6   exp.mem>   t6
token   t7   exp.mem>   t7
token   t8   exp.mem>   t8
```

```
my.def.colors

dir.lee.txt " Eps-AOv.asy"  "cat  defer> load

0 set.file.parse
\ sd SWD
my.pause

\ : WC.all.sel
\ bell fa cr stack.load " SC-Run.sel" "dup s.type.2 swd cr
\ cdtNow defer> load \ Spectral Coherence
\ \ -----------------------------------------------------------
\ bell fa cr stack.load " CWT-Run.sel" "dup s.type.2 swd cr
\ cdtNow defer> load \ CWT
\ \ -----------------------------------------------------------
\ bell fa cr stack.load " WPM-Run.sel" "dup s.type.2 swd cr
\ cdtNow defer> load \ Wavelet Package Map
\ \ -----------------------------------------------------------
\ bell fa cr stack.load " WC-Run.sel" "dup s.type.2 swd cr
\ cdtNow defer> load \ Wavelet Coherence
\ bell fa cr stack.load " WC-A-Plt.sel" "dup s.type.2 swd cr
\ cdtNow defer> load \ Plot All WC Curves
\ \ -----------------------------------------------------------
\ bell fa cr stack.load " PsiC-Run.sel" "dup s.type.2 swd cr
\ cdtNow defer> load \ Generate Psi Curve WKS file
\ bell fa cr stack.load " PsiC-Plt.sel" "dup s.type.2 swd cr
\ cdtNow defer> load \ Plot Psi Curve
\ cr "   CONGRATULATION!" S.type.2
\ ;
\ \ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
: WC.all.sel
bell fa cr stack.load " R-SCoh.sel" "dup s.type.2 swd cr
cdtNow defer> load \ Spectral Coherence
\ -----------------------------------------------------------
bell fa cr stack.load " R-WCoh.sel" "dup s.type.2 swd cr
cdtNow defer> load \ Wavelet Coherence
bell fa cr stack.load " P-WCoh-A.sel" "dup s.type.2 swd cr
cdtNow defer> load \ Plot All WC Curves
\ -----------------------------------------------------------
bell fa cr stack.load " R-Rdg.sel" "dup s.type.2 swd cr
cdtNow defer> load \ CWT
\ -----------------------------------------------------------
bell fa cr stack.load " R-WPM.sel" "dup s.type.2 swd cr
cdtNow defer> load \ Wavelet Package Map
\ -----------------------------------------------------------
bell fa cr stack.load " R-PsiC.sel" "dup s.type.2 swd cr
cdtNow defer> load \ Generate Psi Curve WKS file
bell fa cr stack.load " P-PsiC.sel" "dup s.type.2 swd cr
cdtNow defer> load \ Plot Psi Curve
\ -----------------------------------------------------------
bell fa cr stack.load " R-Ent.sel" "dup s.type.2 swd cr
cdtNow defer> load \ Spectral Coherence
\ -----------------------------------------------------------
bell fa cr stack.load " R-Nor.sel" "dup s.type.2 swd cr
cdtNow defer> load \ Spectral Coherence
```

```
\ ------------------------------------------------------------
bell fa cr stack.load " R-Pha.sel" "dup s.type.2 swd cr
cdtNow defer> load \ Spectral Coherence
\ ------------------------------------------------------------
bell fa cr stack.load " R-DnR.sel" "dup s.type.2 swd cr
cdtNow defer> load \ Spectral Coherence
\ ------------------------------------------------------------
cr "   CONGRATULATION!" S.type.2
;
\ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

\ =================================================================
: Sta.all.sel
bell fa cr stack.load " LSta-0.asy" "dup s.type.2 cr swd
cdt defer> load \ Statistics
bell fa cr stack.load " LWHP-0.asy" "dup s.type.2 cr swd
cdt defer> load \ Probability Distribution of Zero-up wave heights
\ ------------------------------------------------------------
bell fa cr stack.load " LRef-0.asy" "dup s.type.2 cr swd
cdt defer> load \ Reflection Coefficients
\ ------------------------------------------------------------
bell fa cr stack.load " LPlt-0.asy" "dup s.type.2 cr swd
cdt defer> load \ Plot Wave Signal
\ ------------------------------------------------------------
bell fa cr stack.load " ACC-Exp.asy" "dup s.type.2 cr swd
cdt defer> load \ Auto Correlation Coefficients
\ ------------------------------------------------------------
cr "   CONGRATULATION!" S.type.2
;

sd cdt
\ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

123-A2W.asy

勇伯程式庫 A01519-20070125-A070 ❖ January 25, 2007 ❖ [Page count: 6]

```
: 123-A2W.asy ; \ %----- Asyst Prog
\ : To123.asy ; \ %----- Asyst Prog

integer scalar to.stack.flag  scalar to.123.flag  scalar gp.flag
        scalar col.inc        scalar col.no       scalar row.no
        scalar to.123.999.flag

32 string 123.file.name
32 string 123.file.name.1
32 string 123.file.name.2
32 string 123.file.now
3  string file.ext
\ 14 string d&t.string

real  dim[ 20 ] array 123.ary.tmp
token 123.ary.n   exp.mem> 123.ary.n
token 123.ary.n2  exp.mem> 123.ary.n2
dim[ 4 , 64 ]  string.array 123.ary.s
dim[ 4 , 32 ]  string.array 123.ary.s2

0  to.stack.flag :=
1  to.123.flag   :=
1  gp.flag       :=
1  file.no       :=
10 col.inc       :=

dir.lee.asc " 000.wks" "cat 123.file.name ":=

dir.asy.sys       " 123io.sov" "cat       defer>  load.overlay

: set.123.BP \ [" Str1"  " Str2" - ]
  \ {29.line}
  123.file.name.2  ":=
  123.file.name.1  ":=
  dir.lee.asc 123.file.name.1 "cat 123.file.name.1 ":=
  dir.lee.asc 123.file.name.2 "cat 123.file.name.2 ":=
  dir.asy.sys       " 123io.sov" "cat       defer>  load.overlay
  123.file.name.1    defer>  123file.create
  123.file.name.2    defer>  123file.create
  123.file.name.1  123.file.name ":=
  release.overlay
;
: set.123
  {29.line}
  123.file.name
  dir.lee.asc "len  1 +   "drop
  "dup "len "drop  dir.lee.asc "len "drop -  "sub  123.file.name.1 ":=
  dir.asy.sys       " 123io.sov" "cat       defer>  load.overlay
  dir.lee.asc
  \ cr  ." 123 file 1 name:"  " 123.wks"    get.str
  cr  ." 123 file 1 name (close all wks files first!):"
  123.file.name.1  get.str
```

```
  "cat   123.file.name.1  ":=
  dir.lee.asc
  cr  ." 123 file 2 name:"  " 1234.wks"  get.str
  "cat   123.file.name.2  ":=
  cr  ." Group flag:"  gp.flag  get.val  gp.flag :=
  cr  ." file.no:"     file.no  get.val  file.no :=
  123.file.name.1      defer>  123file.create
  123.file.name.2      defer>  123file.create
  cr  ." Initial all WKS files (Yes:999)!!:" 0  get.val
  999 =
  if
    dir.lee.asc  " le"     "cat  " .wks" "cat  defer> 123file.create
    dir.lee.asc  " me"     "cat  " .wks" "cat  defer> 123file.create
    dir.lee.asc  " so"     "cat  " .wks" "cat  defer> 123file.create
    dir.lee.asc  " ona"    "cat  " .wks" "cat  defer> 123file.create
    dir.lee.asc  " ons"    "cat  " .wks" "cat  defer> 123file.create
    dir.lee.asc  " bo1"    "cat  " .wks" "cat  defer> 123file.create
    dir.lee.asc  " bo2"    "cat  " .wks" "cat  defer> 123file.create
    dir.lee.asc  " bo3"    "cat  " .wks" "cat  defer> 123file.create
    dir.lee.asc  " 1234"   "cat  " .wks" "cat  defer> 123file.create
    dir.lee.asc  " 123999" "cat  " .wks" "cat  defer> 123file.create
  then
  123.file.name.1  123.file.name ":=
  release.overlay
;
: to.stack
  to.stack.flag 1 =
  if
    \ dup
  else
    drop
  then
;
: to.123 \ [ - ]
  \ Use 123.file.name
  \ Put data   in 123.ary.n
  \ Put String in 123.ary.s
  \ Specify  fine.no,  gp.flag,  and  col.inc
  to.123.flag 1 =
  if
    dir.asy.sys       " 123io.sov" "cat  defer>  load.overlay
    123.file.name      defer>  123file.open
    1    file.no   gp.flag  1 -  col.inc *   +  123write.down
    \ 123.ary.s  "[ 1 ] ">123file
    \ ----------------------------------------
    "date " -" "cat "time "cat 123.ary.s "[ 4 ] ":=
    \ ----------------------------------------
    123.ary.s           ">123file \ Modified for DnR
    \    2    file.no   gp.flag  1 -  col.inc *   +  123write.down
    11    file.no   gp.flag  1 -  col.inc *   +  123write.down
    123.ary.n       array>123file
    defer>  123file.close
  then
  cr 123.file.name  s.type
  release.overlay
```

```
;
: to.123.1
  123.file.name.1  123.file.name ":=
  to.123
;
: to.123.2
  123.file.name.2  123.file.name ":=
  to.123
;
: to.123.999
  to.123.999.flag 1 =
  if
    dir.asy.sys        " 123io.sov" "cat  defer>  load.overlay
    dir.lee.asc " 123999.wks" "cat  defer>  123file.open
    row.no    file.no   gp.flag  1 -  col.inc *   +  123write.down
    ">123file
    defer>  123file.close
    row.no 1 +  row.no :=
    release.overlay
  then
\ cr 123.file.name  s.type
;

: ato123.int
  cr ." Array in stack to 123 (Interactive):"
  dir.asy.sys        " 123io.sov" "cat  defer>  load.overlay
  dir.lee.asc
  cr ." WKS file name:"  " 123TMP.wks"  get.str  "cat
  cr "dup  \ "type
  "dup  defer>   delete
  "dup  defer>   123file.create
        defer>   123file.open
  cr ." Row:"  1  get.val
  cr ." col:"  1  get.val
  cr ." Across(a)|Down(d):" " D"  get.str

  \   [Array, Row, Col, Across(1)|Down(2) -]"
  \  1  =
  "upper  " A"  "=
  if
    123write.across
  else
    123write.down
  then
  array>123file
  defer>  123file.close
  release.overlay
  cr  "type  "  OK" s.type
;

: ato123 \ [Array, Row, Col, Across(a)|Down(d) -]
  cr ." Array to 123: [Array, Row, Col, Across(a)|Down(d) -]"
  dir.asy.sys        " 123io.sov" "cat  defer>  load.overlay
  dir.lee.asc
```

```
  cr ." WKS file name:" " 123TMP.wks"  get.str  "cat
  cr "dup \ "type
  "dup  defer>   delete
  "dup  defer>   123file.create
        defer>   123file.open
  \  1  =
  "swap
  "upper  " A"  "=
  if
    123write.across
  else
    123write.down
  then
  array>123file
  defer>  123file.close
  release.overlay
  cr  "type  "  OK" s.type
;

integer scalar row.beg
integer scalar col.beg
1 string a.or.d

: ato123.bat.ini
  dir.asy.sys       " 123io.sov" "cat  defer>  load.overlay
  cr  ." Array to 123: [File, String, Array, Row, Col, Across( a)|Down( d)]"
  \  " c:\per\asy\asc\tmp2.wks"   tmp.s.9  ":=
  dir.lee.asc  "dup  s.type  cr  " 123tmp.wks"  get.str   "cat  tmp.s.9
":=
  tmp.s.9   "dup s.type    defer>   delete
  tmp.s.9                  defer>   123file.create
  release.overlay
;
: ato123.bat
  dir.asy.sys       " 123io.sov" "cat  defer>  load.overlay
  col.beg  :=
  row.beg  :=
  a.or.d  ":=
  123.ary.s  "[ 1 ]  ":=
  tmp.s.9  ":=
  tmp.s.9  defer>   123file.open
  a.or.d
  " a"  "=
  if
    123.ary.s
    row.beg       col.beg
    123write.across
    ">123file
    row.beg       col.beg  1 +
    123write.across
    array>123file
  else
    123.ary.s
    row.beg       col.beg
    123write.down
```

```
    ">123file
    \     row.beg 1 +    col.beg
    row.beg 10 +    col.beg
    123write.down
    array>123file
  then
  tmp.s.9  defer>  123file.close
  release.overlay
;

: a2wks
  dir.asy.sys       " 123io.sov" "cat  defer>  load.overlay
  \  dir.lee.asc
  cr ." Array to WKS --- Does the file exist?"
  cr ." ---> " " [file.name, String, Array, Col -- ]"  s.type.3
  "swap
  dir.lee.asc   "swap  "cat
           defer>   123file.open
  dup
  1           swap
  123write.down
  ">123file
  1 10 +      swap
  123write.down
  array>123file
  \  defer>  123file.close
  123file.close
  release.overlay
;

release.overlay
: hint
cr  ."   Usages:"
cr  ."     (1) (old wks destroyed) [ array - ]     ato123.int"
cr  ."     (2) (old wks destroyed) [ array  row  col  'A/D' ]     ato123"
cr  ."     (3) (set.123 once)      [ wks.file.name  str array col ]    a2wks"
;

: app.ramp.col
  " $$[("  file.no 0 n>s "cat " ," "cat
  123.ary.n []size swap drop 10 + 0 n>s "cat
  " )<" "cat
  \ ---------------------------------------- ■
  \ \ $$[(5,1034)<1024>]
  123.ary.n []size swap drop 0 n>s
  \ ---------------------------------------- ■
  \ \ $$[(5,1034)<Level:2-Pt4,Power:10,Element:12>]
  \ " Level:" jwp.level 0 n>s "cat
  \ " -Pt" "cat 2 jwp.level ** 0 n>s "cat
  \ " ,Power:" "cat n.power 0 n>s "cat
  \ " ,Element:" "cat i.ele 0 n>s "cat
  \ ---------------------------------------- ■
  "cat
  " >]" "cat
  123.ary.s "[ 1 ] ":=
```

```
  \
  123.ary.n []size swap drop real ramp 1.0 * becomes> 123.ary.n
  \
  to.123
  file.no 1 + file.no := \ post inc of column
;
```

P-WCoh-A.shl

勇伯程式庫 A01520-20070125-A071 ❖ January 25, 2007 ❖ [Page count: 4]

```
: WC-A-Plt.sel ; \ %----- Asyst Prog

\ ^---------------------------------------------------------------
\ \\ ---------------------------------------------------------------
\ %% String assignment
\     0.  misce.str -- at Top Left, can be assigned in FB(2)
\     1.  spe.str.1 -- provided by wks file (element at first row and last column)
\     2.  spe.str.2 or spe.str.3 -- (CASE) above Legend, use spe.str.2.or.3
\           spe.str.2: can be assigned in FB(2)
\           spe.str.3: default to eps.file.name, not changeable, see plot.bat
\     3   123.str.ary or lgn.str.ary -- Legends for lines, use get.lgns.a.or.d
\           123.str.ary: provided by wks file (first row)
\           lgn.str.ary: can be assigned in FB2
\ \\ ---------------------------------------------------------------
\ ^---------------------------------------------------------------

fa sd swd

cdt load FigAPlot.shl \
\ \ ===============================================================
\ 1  string output.flag.str
\ 26 string d&t.string
\ 14 string new.d&t.string
\
\ integer scalar new.d&t.flag
\
\ 1 new.d&t.flag  := \ 0: default now, 1: assigned
\
\ : set.new.d&t \ [ "str -- ]
\   new.d&t.flag 1 =
\   if
\     new.d&t.string   ":=
\   else
\     "drop  \ use default
\   then
\ ;
\
\ cdtNow load WD.asy
\
\ : Chose.output
\   cr ." Chose output format:"
\   cr ."                      M: Monitor"
\   cr ."                      E: EPS"
\   cr ."                      " " M" get.str
\   "dup s.type
\   output.flag.str ":=
\ ;
\ Chose.output

\ ===============================================================
: Forget.mark ;
```

```
\ ==================================================================

: file.name
  stack.data " C1AC-CC.wks" \ Coh - Wave Current (Spectral) - A series
  stack.load " C-CC-A.bpf"
;
cdt load FigAPlot.asy

: file.name
  stack.data " C4FC-Px.wks" \ Coh - Wave Current (Spectral) - A series
  stack.load " C-Fx.bpf" \ No phase adjustment
;
cdt load FigAPlot.asy

: file.name
  stack.data " C-WCs-A.wks" \ Coh - Wave Current (Spectral) - A series
  stack.load " C-WCs-A.bpf"
;
" 08/20/98-12:30"  set.new.d&t
cdt load FigAPlot.asy
\ mon.plot

forget file.name
: file.name
  stack.data " C-WCs-9F.wks"
  stack.load " C-WCs-9F.bpf"
;
" 08/21/98-15:52"  set.new.d&t
cdt load FigAPlot.asy
\ mon.plot

forget file.name
: file.name
  stack.data " C-WCs-4F.wks"
  stack.load " C-WCs-4F.bpf"
;
" 08/21/98-15:52"  set.new.d&t
cdt load FigAPlot.asy
\ mon.plot

forget file.name
: file.name
  stack.data " C-WCs-2F.wks"
  stack.load " C-WCs-2F.bpf"
;
" 08/21/98-15:52"  set.new.d&t
cdt load FigAPlot.asy
\ mon.plot

forget file.name
: file.name
  stack.data " C-WCs-1F.wks"
  stack.load " C-WCs-1F.bpf"
;
" 08/21/98-15:52"  set.new.d&t
```

```
cdt load FigAPlot.asy
\ mon.plot

forget file.name
: file.name
\   stack.data " C1BnQxCC.wks" \ Coh - B & Q series - Phase X - Cur-Cur
\   stack.data " C1BnQ-CC.wks" \                   -Phase Optimize
  stack.data " C-CC-BnQ.wks" \ Coh - Current Current - B and Q series
  stack.load " C-CC-BnQ.bpf"
;
" 08/21/98-17:02"  set.new.d&t
cdt load FigAPlot.asy
\ mon.plot

forget file.name
: file.name
  stack.data " C-WC-BMC.wks" \ Coh - Wave Cur - B series - Modified Complex
  stack.load " C-WC-B-M.bpf" \
;
" 08/21/98-17:02"  set.new.d&t
cdt load FigAPlot.asy
\ mon.plot

forget file.name
: file.name
  stack.data " C-WC-1FC.wks"
  stack.load " C-WC-1F.bpf"
;
" 08/22/98-00:33"  set.new.d&t
cdt load FigAPlot.asy
\ mon.plot

forget file.name
: file.name
  stack.data " C-WC-2FC.wks"
  stack.load " C-WC-2F.bpf"
;
" 08/22/98-00:33"  set.new.d&t
cdt load FigAPlot.asy
\ mon.plot

forget file.name
: file.name
    stack.data " C-WC-4FC.wks"
    stack.load " C-WC-4F.bpf"
;
" 08/22/98-00:33"  set.new.d&t
cdt load FigAPlot.asy
\ mon.plot

forget file.name
: file.name
  stack.data " C-WC-A-R.wks" \ Coh - Wave Cur - A series - Real
  stack.load " C-WC-A.bpf"
;
```

```
" 08/22/98-14:30"  set.new.d&t
cdt load FigAPlot.asy \
\ mon.plot

\ -----------------------------------------------------------------
\ =================================================================
\ -----------------------------------------------------------------

: All.file.name
  \  " n"      1x1y.flag  ":=
  \ -----------------------------------------
  \   wks.file.name      FB2.file.name
  \ -----------------------------------------
  " C-WC-BRC.wks"      " C-WC-B-M.bpf"
  " C-WCs-A.wks"       " C-WCs-A.bpf"
  " C-WCs-9F.wks"      " C-WCs-9F.bpf"
  " C-WCs-4F.wks"      " C-WCs-4F.bpf"
  " C-WCs-2F.wks"      " C-WCs-2F.bpf"
  " C-WCs-1F.wks"      " C-WCs-1F.bpf"
  " C-SC-.wks"         " C-SC-.FB2"
  " C-WC-BRC.wks"      " C-WC-B-M.bpf"
  " C-CC-BnQ.wks"      " C-CC-BnQ.bpf"
  " C-WC-BMC.wks"      " C-WC-B-M.bpf"
  " C-WC-B_C.wks"      " C-WC-B.FB2"
  " C-WC-B_R.wks"      " C-WC-B.FB2"
  " C-WC-1FC.wks"      " C-WC-1F.bpf"
  " C-WC-2FC.wks"      " C-WC-2F.bpf"
  " C-WC-4FC.wks"      " C-WC-4F.bpf"
  " C-WC-F4r.wks"      " C-WC4F.FB2"      \ use ymax=0.5
  " C-WC-A_C.wks"      " C-WC-A.FB2"
  " C-WC-A-R.wks"      " C-WC-A.bpf"
  " C-model.wks"       " C-WC-A.FB2"      \ use " C-WC-A-R.wks"
  " C-psi1or.wk1"      " C-psi1or.FB2"    \ Re & Im parts of wavelets bat1
  " C-psi2or.wk1"      " C-psi2or.FB2"    \ Re & Im parts of wavelets bat2
  " C-psi-ol.wk1"      " C-psi-ol.FB2"    \ Re & Im parts of wavelets bat3
  \ " MorletRI.wk1"      " C-psi-ol.FB2"  \ Re & Im parts of wavelets
  " C-WC-R1A.wks"      " C-WC-A.FB2"      \ sai=50~5.5  a series
  " C-WC-R0A.wks"      " C-WC-A.FB2"      \    =11~5.5  a
  " C-WC-R-A.wks"      " C-WC-A.FB2"
  \ " C-WC-C2p.wks"
  \ " C-WC-C.wks"
  \ " C-WC-R.wks"
;
\ =================================================================
```

P-Pha.shl

勇伯程式庫 A01521-20070125-A072 ❖ January 25, 2007 ❖ [Page count: 6]

```
: Plot.Phase.shell ; \  -*- txt:asy -*-

\ ^-----------------------------------------------------------------
\ \\ -----------------------------------------------------------------
\ %% String assignment
\     0.  misce.str -- at Top Left, can be assigned in FB(2)
\     1.  spe.str.1 -- provided by wks file (element at first row and last column)
\     2.  spe.str.2 or spe.str.3 -- (CASE) above Legend, use spe.str.2.or.3
\           spe.str.2: can be assigned in FB(2)
\           spe.str.3: default to eps.file.name, not changeable, see plot.bat
\     3   123.str.ary or lgn.str.ary -- Legends for lines, use get.lgns.a.1r.d
\           123.str.ary: provided by wks file (first row)
\           lgn.str.ary: can be assigned in FB2
\ \\ -----------------------------------------------------------------
\ ^-----------------------------------------------------------------

fa sd swd

cdt load FigAPlot.sel \
\ \ ================================================================
\ 1  string output.flag.str
\ 26 string d&t.string
\ 14 string new.d&t.string
\
\ integer scalar new.d&t.flag
\
\ 1 new.d&t.flag  := \ 0: default now, 1: assigned
\
\ : set.new.d&t \ [ "str -- ]
\   new.d&t.flag 1 =
\   if
\     new.d&t.string   ":=
\   else
\     "drop  \ use default
\   then
\ ;
\
\ cdtNow load WD.asy
\
\ : Chose.output
\   cr ." Chose output format:"
\   cr ."                     M: Monitor"
\   cr ."                     E: EPS"
\   cr ."                     " " M" get.str
\   "dup s.type
\   output.flag.str ":=
\ ;
\ Chose.output

\ ================================================================
: Forget.mark ;
```

```
\ =================================================================

: file.name
\   stack.data " Phase-A.WKS" \ Coh - Wave Current (Spectral) - A series
  stack.data " Pha-LEME.WKS" \ Coh - Wave Current (Spectral) - A series
  stack.load " Pha-LEME.BPF"
;
cdt load FigAPlot.asy \
forget file.name

: file.name
\   stack.data " Phase-A.WKS" \ Coh - Wave Current (Spectral) - A series
  stack.data " Pha-SOD.WKS" \ Coh - Wave Current (Spectral) - A series
  stack.load " Pha-SOD.BPF"
;
cdt load FigAPlot.asy \
forget file.name

: file.name
  stack.data " Pha-ONA1.WKS" \ Coh - Wave Current (Spectral) - A series
  stack.load " Pha-ONA1.BPF"
;
cdt load FigAPlot.asy \
forget file.name

: file.name
  stack.data " Pha-ONA2.WKS" \ Coh - Wave Current (Spectral) - A series
  stack.load " Pha-ONA2.BPF"
;
cdt load FigAPlot.asy \
forget file.name

: file.name
  stack.data " Pha-ONS1.WKS" \ Coh - Wave Current (Spectral) - A series
  stack.load " Pha-ONS1.BPF"
;
cdt load FigAPlot.asy \
forget file.name

: file.name
  stack.data " Pha-ONS2.WKS" \ Coh - Wave Current (Spectral) - A series
  stack.load " Pha-ONS2.BPF"
;
cdt load FigAPlot.asy \
forget file.name

: file.name
  stack.data " Pha-ONC.WKS" \ Coh - Wave Current (Spectral) - A series
  stack.load " Pha-ONC.BPF"
;
cdt load FigAPlot.asy \
forget file.name

: file.name
  stack.data " Pha-BO11.WKS" \ Coh - Wave Current (Spectral) - A series
```

```
  stack.load " Pha-B011.BPF"
;
cdt load FigAPlot.asy \
forget file.name

: file.name
  stack.data " Pha-B012.WKS" \ Coh - Wave Current (Spectral) - A series
  stack.load " Pha-B012.BPF"
;
cdt load FigAPlot.asy \
forget file.name

: file.name
  stack.data " Pha-B021.WKS" \ Coh - Wave Current (Spectral) - A series
  stack.load " Pha-B021.BPF"
;
cdt load FigAPlot.asy \
forget file.name

: file.name
  stack.data " Pha-B022.WKS" \ Coh - Wave Current (Spectral) - A series
  stack.load " Pha-B022.BPF"
;
cdt load FigAPlot.asy \
forget file.name

: file.name
  stack.data " Pha-B031.WKS" \ Coh - Wave Current (Spectral) - A series
  stack.load " Pha-B031.BPF"
;
cdt load FigAPlot.asy \
forget file.name

: file.name
  stack.data " Pha-B032.WKS" \ Coh - Wave Current (Spectral) - A series
  stack.load " Pha-B032.BPF"
;
cdt load FigAPlot.asy \
forget file.name

\ ================================================================
\ ================================================================

\ : file.name
\   stack.data " Cs1A-CC.wks" \ Coh - Current Current - B and Q series
\   stack.load " CsCC-A.bpf"
\ ;
\ cdt load FigAPlot.asy

\ : file.name
\   stack.data " C4FC-Px.wks" \ Coh - Wave Current (Spectral) - A series
\   stack.load " C-Fx.bpf" \ No phase adjustment
\ ;
\ cdt load FigAPlot.asy
```

```
\ : file.name
\   stack.data " Cs1A-WC.wks" \ Coh - Wave Current (Spectral) - A series
\   stack.load " C-A.bpf"
\ ;
\ " 08/20/98-12:30"  set.new.d&t
\ cdt load FigAPlot.asy

\ : file.name
\   stack.data " Cs1A-WC.wks" \ Coh - Wave Current (Spectral) - A series
\   stack.load " C-WCs-A.bpf"
\ ;
\ " 08/20/98-12:30"  set.new.d&t
\ cdt load FigAPlot.asy

\ : file.name
\   stack.data " C-WCs-A.wks" \ Coh - Wave Current (Spectral) - A series
\   stack.load " C-WCs-A.bpf"
\ ;
\ " 08/20/98-12:30"  set.new.d&t
\ cdt load FigAPlot.asy

\
\ forget file.name
\ : file.name
\   stack.data " C-WCs-9F.wks"
\   stack.load " C-WCs-9F.bpf"
\ ;
\ " 08/21/98-15:52"  set.new.d&t
\ cdt load FigAPlot.asy
\ \ mon.plot
\
\ forget file.name
\ : file.name
\   stack.data " C-WCs-4F.wks"
\   stack.load " C-WCs-4F.bpf"
\ ;
\ " 08/21/98-15:52"  set.new.d&t
\ cdt load FigAPlot.asy
\ \ mon.plot
\
\ forget file.name
\ : file.name
\   stack.data " C-WCs-2F.wks"
\   stack.load " C-WCs-2F.bpf"
\ ;
\ " 08/21/98-15:52"  set.new.d&t
\ cdt load FigAPlot.asy
\ \ mon.plot
\
\ forget file.name
\ : file.name
\   stack.data " C-WCs-1F.wks"
\   stack.load " C-WCs-1F.bpf"
\ ;
\ " 08/21/98-15:52"  set.new.d&t
```

```
\ cdt load FigAPlot.asy
\ \ mon.plot
\
\ forget file.name
\ : file.name
\ \   stack.data " C1BnQxCC.wks" \ Coh - B & Q series - Phase X - Cur-Cur
\ \   stack.data " C1BnQ-CC.wks" \                    -Phase Optimize
\   stack.data " C-CC-BnQ.wks" \ Coh - Current Current - B and Q series
\   stack.load " C-CC-BnQ.bpf"
\ ;
\ " 08/21/98-17:02"  set.new.d&t
\ cdt load FigAPlot.asy
\ \ mon.plot
\
\ forget file.name
\ : file.name
\   stack.data " C-WC-BMC.wks" \ Coh - Wave Cur - B series - Modified Complex
\   stack.load " C-WC-B-M.bpf" \
\ ;
\ " 08/21/98-17:02"  set.new.d&t
\ cdt load FigAPlot.asy
\ \ mon.plot
\
\ forget file.name
\ : file.name
\   stack.data " C-WC-1FC.wks"
\   stack.load " C-WC-1F.bpf"
\ ;
\ " 08/22/98-00:33"  set.new.d&t
\ cdt load FigAPlot.asy
\ \ mon.plot
\
\ forget file.name
\ : file.name
\   stack.data " C-WC-2FC.wks"
\   stack.load " C-WC-2F.bpf"
\ ;
\ " 08/22/98-00:33"  set.new.d&t
\ cdt load FigAPlot.asy
\ \ mon.plot
\
\ forget file.name
\ : file.name
\     stack.data " C-WC-4FC.wks"
\     stack.load " C-WC-4F.bpf"
\ ;
\ " 08/22/98-00:33"  set.new.d&t
\ cdt load FigAPlot.asy
\ \ mon.plot
\
\ forget file.name
\ : file.name
\   stack.data " C-WC-A-R.wks" \ Coh - Wave Cur - A series - Real
\   stack.load " C-WC-A.bpf"
\ ;
```

```
\ " 08/22/98-14:30"  set.new.d&t
\ cdt load FigAPlot.asy
\ \ mon.plot
\
\ \ -------------------------------------------------------------
\ \ ==================================================================
\ \ -------------------------------------------------------------

: All.file.name
  \  " n"      1x1y.flag  ":=
  \ -----------------------------------------
  \   wks.file.name     FB2.file.name
  \ -----------------------------------------
  " C-WC-BRC.wks"     " C-WC-B-M.bpf"
  " C-WCs-A.wks"      " C-WCs-A.bpf"
  " C-WCs-9F.wks"     " C-WCs-9F.bpf"
  " C-WCs-4F.wks"     " C-WCs-4F.bpf"
  " C-WCs-2F.wks"     " C-WCs-2F.bpf"
  " C-WCs-1F.wks"     " C-WCs-1F.bpf"
  " C-SC-.wks"        " C-SC-.FB2"
  " C-WC-BRC.wks"     " C-WC-B-M.bpf"
  " C-CC-BnQ.wks"     " C-CC-BnQ.bpf"
  " C-WC-BMC.wks"     " C-WC-B-M.bpf"
  " C-WC-B_C.wks"     " C-WC-B.FB2"
  " C-WC-B_R.wks"     " C-WC-B.FB2"
  " C-WC-1FC.wks"     " C-WC-1F.bpf"
  " C-WC-2FC.wks"     " C-WC-2F.bpf"
  " C-WC-4FC.wks"     " C-WC-4F.bpf"
  " C-WC-F4r.wks"     " C-WC4F.FB2"      \ use ymax=0.5
  " C-WC-A_C.wks"     " C-WC-A.FB2"
  " C-WC-A-R.wks"     " C-WC-A.bpf"
  " C-model.wks"      " C-WC-A.FB2"      \ use " C-WC-A-R.wks"
  " C-psi1or.wk1"     " C-psi1or.FB2"    \ Re & Im parts of wavelets bat1
  " C-psi2or.wk1"     " C-psi2or.FB2"    \ Re & Im parts of wavelets bat2
  " C-psi-ol.wk1"     " C-psi-ol.FB2"    \ Re & Im parts of wavelets bat3
  \ " MorletRI.wk1"     " C-psi-ol.FB2"  \ Re & Im parts of wavelets
  " C-WC-R1A.wks"     " C-WC-A.FB2"      \ sai=50~5.5  a series
  " C-WC-R0A.wks"     " C-WC-A.FB2"      \    =11~5.5  a
  " C-WC-R-A.wks"     " C-WC-A.FB2"
  \ " C-WC-C2p.wks"
  \ " C-WC-C.wks"
  \ " C-WC-R.wks"
;
\ =================================================================
```

....

Plt_WBP.shl

勇伯程式庫 A01522-20070125-A073 ❖ January 25, 2007 ❖ [Page count: 3]

```
: Plt_TnD.shl ; \ Wavelet packet at different levels \ -*- txt:asy -*-
\ Wednesday, 2002/11/20 at 20:56:41
\ =================================================================
\ =================================================================
\ PhiPsi.str tmp.s ":=
fa
\ 4 string PhiPsi.str
\ tmp.s PhiPsi.str ":=
\ -----------------------------------------------------------------
4 string PhiPsi.str
: Choose.MW.FW
  cr
  ." Mother (psi) of Father (phi) wavelet"  " M" get.str
  " F" "=
  if
    " Phi-" PhiPsi.str ":=
  else
    " Psi-" PhiPsi.str ":=
  then
;
\ Choose.MW.FW
\ =================================================================
cdt load FigA-Ini.asy \
1.5 ps.linewidth.now :=
\ =================================================================
\ =================================================================
5 string wlet
" on11c" wlet ":=
" on22a" wlet ":=
\ -----------------------------------------------------------------
1 string wp.or.wb.str
\ stack.data and stack.load are dummies for input gathering only
" P" wp.or.wb.str ":=
: file.name
  stack.data " WPon22A1.WKS"
  stack.load " WPon22A1.BPF"
  "drop "drop
  " W" wp.or.wb.str "cat wlet "cat " 1.wks" "cat
  " W" wp.or.wb.str "cat wlet "cat " 1.bpf" "cat
;
cdt load FigACore.asy \
forget file.name

: file.name
  stack.data " WPon22A2.WKS"
  stack.load " WPon22A2.BPF"
  "drop "drop
  " W" wp.or.wb.str "cat wlet "cat " 2.wks" "cat
  " W" wp.or.wb.str "cat wlet "cat " 2.bpf" "cat
;
cdt load FigACore.asy \
forget file.name
```

```
: file.name
  stack.data " WPon22A3.WKS"
  stack.load " WPon22A3.BPF"
  "drop "drop
  " W" wp.or.wb.str "cat wlet "cat " 3.wks" "cat
  " W" wp.or.wb.str "cat wlet "cat " 3.bpf" "cat
;
cdt load FigACore.asy \
forget file.name

\ ==================================================================
" B" wp.or.wb.str ":=
: file.name
  " W" wp.or.wb.str "cat wlet "cat " 1.wks" "cat
  " W" wp.or.wb.str "cat wlet "cat " 1.bpf" "cat
;
cdt load FigACore.asy \
forget file.name

: file.name
  " W" wp.or.wb.str "cat wlet "cat " 2.wks" "cat
  " W" wp.or.wb.str "cat wlet "cat " 2.bpf" "cat
;
cdt load FigACore.asy \
forget file.name

: file.name
  " W" wp.or.wb.str "cat wlet "cat " 3.wks" "cat
  " W" wp.or.wb.str "cat wlet "cat " 3.bpf" "cat
;
cdt load FigACore.asy \
forget file.name

\ ==================================================================
" on22a" wlet ":=
" on11c" wlet ":=
\ ------------------------------------------------------------------
1 string wp.or.wb.str
\ stack.data and stack.load are dummies for input gathering only
" P" wp.or.wb.str ":=
: file.name
  " W" wp.or.wb.str "cat wlet "cat " 1.wks" "cat
  " W" wp.or.wb.str "cat wlet "cat " 1.bpf" "cat
;
cdt load FigACore.asy \
forget file.name

: file.name
  " W" wp.or.wb.str "cat wlet "cat " 2.wks" "cat
  " W" wp.or.wb.str "cat wlet "cat " 2.bpf" "cat
;
cdt load FigACore.asy \
forget file.name
```

```
: file.name
  " W" wp.or.wb.str "cat wlet "cat " 3.wks" "cat
  " W" wp.or.wb.str "cat wlet "cat " 3.bpf" "cat
;
cdt load FigACore.asy \
forget file.name

\ ==================================================================
" B" wp.or.wb.str ":=
: file.name
  " W" wp.or.wb.str "cat wlet "cat " 1.wks" "cat
  " W" wp.or.wb.str "cat wlet "cat " 1.bpf" "cat
;
cdt load FigACore.asy \
forget file.name

: file.name
  " W" wp.or.wb.str "cat wlet "cat " 2.wks" "cat
  " W" wp.or.wb.str "cat wlet "cat " 2.bpf" "cat
;
cdt load FigACore.asy \
forget file.name

: file.name
  " W" wp.or.wb.str "cat wlet "cat " 3.wks" "cat
  " W" wp.or.wb.str "cat wlet "cat " 3.bpf" "cat
;
cdt load FigACore.asy \
forget file.name

\ ==================================================================
\
\ : file.name
\   stack.data " ■"
\   stack.load " ■.BPF"
\ ;
\ cdt load FigACore.asy \
\ forget file.name

\ ==================================================================
```

PenPlot.asy

勇伯程式庫 A01523-20070125-A074 ❖ January 25, 2007 ❖ [Page count: 14]

```
\ Asyst text file       ----------- WORDauto .PLO -----------       Lee Yuen-Ron
\ Note              :: 1. Words for pen plotter
\                      2. Any plotting size ( single or double ports )
\                      3. Any X- or Y- plotting scale ( enlagement or reduction )
\                      4. Both sofeware and hardwave scalings can be synthesized
\                      5. Optimised for pleasing authentic graphs
\                      6. Modified auto plot routines ( rotating and scaling )
\                      7. centered.label command will cause label shifting
\                      8. 05/01/91
\ -----------------------------------------------------------------

echo.off

: xy.data.fit
  over  over
  []min/max   vertical     world.set
  []min/max   horizontal   world.set
;
: xyap.p
  \  vuport.clear    \   ---->  commented  for scaling up and down so that
  xy.data.fit        \          all parts can be drawn
  xy.axis.plot
  xy.data.plot
;
: yap.p
  index.array
  xyap.p
;

\ !0-----------------------------------------------------------------0!

HP7470  \  -- ansi A        7475 -- ansi B

" vs9;"    graph.command   \  ceramic pen = 10  , ink pen = 20

real scalar  x.vup.o          scalar  y.vup.o
     scalar  x.vup.s          scalar  y.vup.s
     scalar  x.pap.s          scalar  y.pap.s
     scalar  v.h.fla          scalar  tmp.v
     scalar  a4.sho           scalar  a4.lon
     scalar  x.axi.o          scalar  y.axi.o
     scalar  x.axi.s          scalar  y.axi.s
     scalar  x.cha.s          scalar  y.cha.s
     scalar  x.lab.a          scalar  y.lab.a
     scalar  x.til.a          scalar  y.til.a
     scalar  x.cap.a          scalar  y.cap.a
     scalar  sca.siz.x        scalar  ssx
     scalar  sca.siz.y        scalar  ssy
     scalar  r.p.len.x        scalar  r.p.len.y
     scalar  a.p.len.x        scalar  a.p.len.y
     scalar  r.p.l.sho        scalar  r.p.l.lon
     scalar  def.r.p.l.sho    scalar  def.r.p.l.lon
```

```
    scalar  sof.sca.x         scalar  sof.sca.y
    scalar  r.p.sca.x         scalar  r.p.sca.y
    scalar  x.vup.o.sf        scalar  y.vup.o.sf
    scalar  sla.deg           scalar  #.por

integer       scalar  v.h.fla   scalar  sca.u.d   scalar aut.plo.fla
              scalar  tmp.v.i   scalar  pen.1     scalar pen.2

80 string  case.name

 plotter.defaults
 normal.coords

 0.      aut.plo.fla  :=    \  1-auto
 9       v.h.fla  :=        \  0-V     9-H
 19.05   a4.sho   :=
 25.65   a4.lon   :=
 0.15    x.axi.o  :=
 0.15    y.axi.o  :=
 0.80    x.axi.s  :=
 0.80    y.axi.s  :=
 1.05    x.cap.a  :=    \ amplication factor for caption character
 1.10    y.cap.a  :=
 1.10    x.lab.a  :=    \ amplication factor for  label   character
 1.15    y.lab.a  :=
 1.15    x.til.a  :=    \ amplication factor for  label   character
 1.30    y.til.a  :=
 0.      sla.deg  :=
 1       pen.1    :=
 2       pen.2    :=
 a4.sho  y.pap.s  :=
 a4.lon  x.pap.s  :=
 20.     def.r.p.l.lon   :=
 15.     def.r.p.l.sho   :=

: ol
  " vs9;"   graph.command    \ ceramic pen = 10   , ink pen = 20
  " sp"  pen.2 "."  "cat   " ;"  "cat            graph.command
  outline
  " sp"  pen.1 "."  "cat   " ;"  "cat            graph.command
\  " vs20;"  graph.command
  " vs5;"  graph.command
;

: cha.siz   \  ssx and ssy   are used to incorperate hardware scaling
  normal.coords      color.on    3  foreground
  y.pap.s /   ssy *       swap    x.pap.s / ssx *              swap    CHAR.SIZE
;
: label
  label  color.off
;
: abs.pos      \    no scaling  absolute position
  normal.coords
  y.pap.s / ssy / y.vup.s /     swap
  x.pap.s / ssx / x.vup.s /     swap   position
```

```
;

: std.v.lab.for
  x.cha.s     y.cha.s       cha.siz
  horizontal
  \             0           5                label.points
  \             \                            world.set
                0.90        0.065            label.scale.point
               -0.5 ssx /  -0.7 ssy /    5   label.format
  vertical
  \             0           5                label.points
  \             \                            world.set
                0.035       0.95             label.scale.point
               -1.2 ssx /  -0.0 ssy /    5   label.format
;
: std.h.lab.for
  x.cha.s       y.cha.s     cha.siz
  horizontal
  \             0           5                label.points
  \             \                            world.set
                0.90        0.075            label.scale.point
                -0.5 ssx /  -0.5 ssy /    5  label.format
  vertical
  \             0           5                label.points
  \             \                            world.set
                0.05        0.95             label.scale.point
               -1.2 ssx /   -0.0 ssy /    5  label.format
;

\ !2^1-----------------------------------------------------------------1^2!

: vpor.to.hpor
  0   v.h.fla      =
  if
     PLOT.ROTATE
     9.   v.h.fla   :=
  then
     a4.lon x.pap.s   :=
     a4.sho y.pap.s   :=
     r.p.len.x    x.pap.s /    r.p.sca.x   :=
     r.p.len.y    y.pap.s /    r.p.sca.y   :=
  normal.coords
;
: hpor.to.vpor
  9   v.h.fla      =
  if
     PLOT.ROTATE
     0.   v.h.fla   :=
  then
     a4.sho x.pap.s   :=
     a4.lon y.pap.s   :=
     r.p.len.x    x.pap.s /    r.p.sca.x   :=
     r.p.len.y    y.pap.s /    r.p.sca.y   :=
  normal.coords
;
```

```
: o.v.p.inf.con
  0.0                       x.vup.o.sf   :=
  0.0                       y.vup.o.sf   :=
  0.20       sof.sca.x *    x.cha.s      :=
  0.40       sof.sca.y *    y.cha.s      :=
  hpor.to.vpor
;

: o.h.p.inf.con
  0.0                       x.vup.o.sf   :=
  0.0                       y.vup.o.sf   :=
  0.18       sof.sca.x *    x.cha.s      :=
  0.46       sof.sca.y *    y.cha.s      :=
  vpor.to.hpor
;
: u.v.p.inf.con
  1.5                       x.vup.o.sf   :=
  2.5                       y.vup.o.sf   :=
  0.15       sof.sca.x *    x.cha.s      :=
  0.30       sof.sca.y *    y.cha.s      :=
  hpor.to.vpor
;
: l.v.p.inf.con
  1.5                       x.vup.o.sf   :=
  2.5                       y.vup.o.sf   :=
  0.15       sof.sca.x *    x.cha.s      :=
  0.30       sof.sca.y *    y.cha.s      :=
  hpor.to.vpor
;
: u.h.p.inf.con
  2.5                       x.vup.o.sf   :=
  1.5                       y.vup.o.sf   :=
  0.15       sof.sca.x *    x.cha.s      :=
  0.38       sof.sca.y *    y.cha.s      :=
  vpor.to.hpor
;
: l.h.p.inf.con
  2.5                       x.vup.o.sf   :=
  1.5                       y.vup.o.sf   :=
  0.15       sof.sca.x *    x.cha.s      :=
  0.38       sof.sca.y *    y.cha.s      :=
  vpor.to.hpor
;

: p.inf.1
    v.h.fla   0   =
    if
      r.p.len.x   def.r.p.l.sho /    x.vup.o.sf *    x.vup.o.sf   :=
      r.p.len.y   def.r.p.l.lon /    y.vup.o.sf *    y.vup.o.sf   :=
    else
      r.p.len.x   def.r.p.l.lon /    x.vup.o.sf *    x.vup.o.sf   :=
      r.p.len.y   def.r.p.l.sho /    y.vup.o.sf *    y.vup.o.sf   :=
    then
;
```

```
\ !3^2************-----------------------------------------------------------************2^3!

integer        scalar p1x   scalar p1y   scalar p2x   scalar p2y   \ a c b d
               scalar p3x   scalar p3y   scalar p4x   scalar p4y   \ p r q s

: set.sca
  " IN;"   graph.command
  v.h.fla
  color.on   12   foreground
  nd
  cr ." ----------------------------------------------------------------------"
  cr ."   0 : Vertical           9 : Horizontal      \  {:set.sca;}
"
  cr ." ----------------------------------------------------------------------"
  cr ."   Choose V or H :         " #input   v.h.fla   :=     10 inten foreground
          v.h.fla  <>    if  plot.rotate  then
  cr ." **********************************************************************"
  cr ."   1 : Default vuports    2 : Set scales    3: Set scale without
"
  cr ."       ( 20 * 15 cm )         ( overall )        character scaling
"
  cr ." ----------------------------------------------------------------------"
\ cr ."
"
\ cr ."     ------ 25.7cm -----        --------------     --------------
"
\ cr ."     |                  |        | Horizontal |     | Vertical   |
"
\ cr ."     |  Original port 19.05cm    |     port   Y cm  |     port   X cm
"
\ cr ."     |                  |        |            |     |            |
"
\ cr cr     P------------------        P-- X cm -----     P---- Y cm ---
"
\ cr ."
"
\ cr ." ----------------------------------------------------------------------"
  cr ."   Choose OPTION        : " #input   sca.u.d   :=      3  foreground
  cr ." ----------------------------------------------------------------------"
  cr

1.          sof.sca.x  :=
1.          sof.sca.y  :=

1  sca.u.d  =
if
  0  v.h.fla  =
  if
    def.r.p.l.sho          r.p.len.x    :=
    def.r.p.l.lon          r.p.len.y    :=
  else
    def.r.p.l.lon          r.p.len.x    :=
    def.r.p.l.sho          r.p.len.y    :=
  then
then
```

```
2  sca.u.d  =     3  sca.u.d  =   or
if
  cr ."      REFERENCE  port's     X-dir        length : " #input  r.p.len.x
:=
  cr ."                            Y-dir        length : " #input  r.p.len.y
:=
  cr ."          -------------------------------------------"
then

0  v.h.fla  =
if
  r.p.len.x    r.p.l.sho    :=
  r.p.len.y    r.p.l.lon    :=
else
  r.p.len.x    r.p.l.lon    :=
  r.p.len.y    r.p.l.sho    :=
then

1    sca.u.d   =
if
      v.h.fla  0  =
      if    o.v.p.inf.con
      else  o.h.p.inf.con
      then
      r.p.sca.x             x.vup.o.sf  x.pap.s /  2. *  -  x.vup.s  :=
      r.p.sca.y             y.vup.o.sf  y.pap.s /  2. *  -  y.vup.s  :=
      r.p.len.x        x.pap.s /  x.vup.s /        ssx :=
      r.p.len.y        y.pap.s /  y.vup.s /        ssy :=
then

2  sca.u.d  =    3 sca.u.d  =   or
if    14  foreground
  cr ."     1    port   or    2 ports   : " #input   #.por     :=
  1  #.por    =
  if
    cr ."      set  O.H.P  or  O.V.P  ::     X_wnd_length = " #input  a.p.len.x :=
    cr ."                                    Y_wnd_length = " #input  a.p.len.y :=
    v.h.fla  0  =
    if    o.v.p.inf.con
    else  o.h.p.inf.con
    then
    p.inf.1
    r.p.sca.x             x.vup.o.sf  x.pap.s /  2. *  -  x.vup.s  :=
    r.p.sca.y             y.vup.o.sf  y.pap.s /  2. *  -  y.vup.s  :=
    a.p.len.x  x.pap.s /  x.vup.s /        ssx :=
    a.p.len.y  y.pap.s /  y.vup.s /        ssy :=
  else
    cr ."      set  U.h.p  or  L.v.p  ::     X_wnd_length = " #input  a.p.len.x :=
    cr ."                                    Y_wnd_length = " #input  a.p.len.y :=
    v.h.fla  0  =
    if    l.v.p.inf.con
    else  l.h.p.inf.con
    then
    p.inf.1
```

```
    r.p.sca.x              x.vup.o.sf  x.pap.s /  2. *  -  x.vup.s  :=
    r.p.sca.y    2. /      y.vup.o.sf  y.pap.s /        -  y.vup.s  :=
    a.p.len.x  x.pap.s /  x.vup.s /        ssx :=
    a.p.len.y  y.pap.s /  y.vup.s /        ssy :=
  then
then
cr ." ---------------------------------------------------------------------"
cr cr  color.off

3  sca.u.d  =
if
  v.h.fla  0  =
  if     r.p.len.x    def.r.p.l.sho  /
         r.p.len.y    def.r.p.l.lon  /    swap
  else   r.p.len.x    def.r.p.l.lon  /
         r.p.len.y    def.r.p.l.sho  /    swap
  then
  ssx   /     sof.sca.x  :=
  ssy   /     sof.sca.y  :=
then

0  v.h.fla  =
if                        \  set to  ROLAND  system  ( Hardware )
  ssx            sca.siz.y  :=
  ssy            sca.siz.x  :=
else
  ssx            sca.siz.x  :=
  ssy            sca.siz.y  :=
then

                           \       IP  :   a c b d
                           \       SC  :   p q r s
                           \
                           \
                           \
                           \       ---------------------------------   ( q,s )
                           \       |                              4|
                           \       |   --------------- ( b,d )     |
                           \       |   |             2|            |
                           \       |   |    IP        |            |
                           \       |   |1             |            |
                           \       |   ---------------             |
                           \       |   ( a,c )                     |
                           \       |                SC             |
                           \       |3                              |
                           \       ---------------------------------
                           \       ( p,r )
                           \       IP  :   a c b d
                           \       SC  :   p q r s      scale = IP / SC
                           \
11040     r.p.l.lon  400 *    -      2. /     p3x  :=     \ SC
11040     p3x -                               p4x  :=
7721      r.p.l.sho  400 *    -      2. /     p3y  :=
7721      p3y -                               p4y  :=
```

```
11040      p4x  p3x -   sca.siz.x * - 2. /     p1x  :=    \ IP
11040      p1x                    -            p2x  :=
7721       p4y  p3y -   sca.siz.y * - 2. /     p1y  :=
7721       p1y                    -            p2y  :=

  \  Shift SC to match SC port with IP port
  \  The center of IP port locates at paper center
  \  ( without any scaling restriction )
  \  { 37.1 , 2.70 } based on reference port sizes of ( 20*15 )  and  ( 5*5 )
  p4x  p3x   2 *dup
    -   p2x  p1x -  40. / /  37.1  sca.siz.x *  4. /  * neg  p3x +    p3x
:=
    -   p2x  p1x -  40. / /  37.1  sca.siz.x *  4. /  * neg  p4x +    p4x
:=
  p4y  p3y   2 *dup
    -   p2y  p1y -  40. / /  2.70  sca.siz.y *  3. /  * neg  p3y +    p3y
:=
    -   p2y  p1y -  40. / /  2.70  sca.siz.y *  3. /  * neg  p4y +    p4y
:=

;
set.sca

21  10  22  70  window  lower.inner.wnd
lower.inner.wnd  {border}

: scale.size
  " in;"                         graph.command
  " ip" p1x fix "." "cat  " ," "cat  p1y fix "." "cat  " ," "cat
        p2x fix "." "cat  " ," "cat  p2y fix "." "cat  " ;" "cat   graph.command
  " sc" p3x fix "." "cat  " ," "cat  p4x fix "." "cat  " ," "cat
        p3y fix "." "cat  " ," "cat  p4y fix "." "cat  " ;" "cat   graph.command
   cr  ."         IP       p1x = " p1x .   p2x .    p1y .   p2y .
   cr  ."         SC       p3x = " p3x .   p4x .    p3y .   p4y .

\ \ Maximum scale up can be achieved by not shifting IP   ---> This part has
\ \  been updated by the above SC shifting method to eliminate scale restriction
\ 0  v.h.fla  =
\ if
\ " ip" p1x float  37.1 40. *  4. /  sca.siz.x * + 120 - fix "." "cat  " ," "cat
\       p1y float  2.7  40. *  3. /  sca.siz.y * +       fix "." "cat  " ;" "cat
\ else
\ " ip" p1x float  37.1 40. *  4. /  sca.siz.x * + 240 - fix "." "cat  " ," "cat
\       p1y float  2.7  40. *  3. /  sca.siz.y * + 560 - fix "." "cat  " ;" "cat
\ then
\ graph.command

\  Shift IP port to a pleasing position ( impose a little scaling restriction )
0  v.h.fla  =
if
  " ip" p1x                                      80 -     "." "cat  " ," "cat
        p1y                                      80 -     "." "cat  " ;" "cat
else
  " ip" p1x                                     200 -     "." "cat  " ," "cat
        p1y                                     560 -     "." "cat  " ;" "cat
```

```
then
graph.command
;
\ !4^3***************-----------------------------------------**************3^4!

: p.inf.2
  1.  r.p.sca.x -  2. /    x.vup.o.sf  x.pap.s /        +        x.vup.o
:=
      r.p.sca.x            x.vup.o.sf  x.pap.s /        2. *  -  x.vup.s
:=
  x.vup.o  y.vup.o   VUPORT.ORIG
  x.vup.s  y.vup.s   VUPORT.SIZE
  scale.size
;
: o.p.inf
  p.inf.1
  1.  r.p.sca.y -  2. /    y.vup.o.sf  y.pap.s /        +        y.vup.o
:=
      r.p.sca.y            y.vup.o.sf  y.pap.s /        2. *  -  y.vup.s
:=
  p.inf.2
;
: 2.p.inf.l
  p.inf.1
  1.  r.p.sca.y -  2. /    y.vup.o.sf  y.pap.s /        +        y.vup.o
:=
      r.p.sca.y    2. /    y.vup.o.sf  y.pap.s /              -  y.vup.s
:=
  p.inf.2
;
: 2.p.inf.u
  p.inf.1
        0.5                y.vup.o.sf  y.pap.s /        +        y.vup.o
:=
      r.p.sca.y    2. /    y.vup.o.sf  y.pap.s /              -  y.vup.s
:=
  p.inf.2
;

\ !5^4------------------------------------------------------------------4^5!
: O.v.p
  o.v.p.inf.con
  o.p.inf
  std.v.lab.for
;
: O.h.p
  o.h.p.inf.con
  o.p.inf
  std.h.lab.for
;
: u.v.p
  u.v.p.inf.con
  2.p.inf.u
  std.v.lab.for
;
```

```
: l.v.p
  l.v.p.inf.con
  2.p.inf.l
  std.v.lab.for
;
: u.h.p
  u.h.p.inf.con
  2.p.inf.u
  std.h.lab.for
;
: l.h.p
  l.h.p.inf.con
  2.p.inf.l
  std.h.lab.for
;
: 1.p
  x.vup.o  x.vup.s  0.1 *   +   x.vup.o  :=
  y.vup.o  y.vup.s  0.1 *   +   y.vup.o  :=
  x.vup.s  0.8 *                x.vup.s  :=
  y.vup.s  0.8 *                y.vup.s  :=
  x.vup.o  y.vup.o   VUPORT.ORIG
  x.vup.s  y.vup.s   VUPORT.SIZE
  scale.size
;
: v.p
  o.v.p   1.p
;
: h.p
  o.h.p   1.p
;
\ !6^5-----------------------------------------------------------------------5^6!

real         scalar  tmp.v.1          scalar tmp.v.2

: all.cap           \  [ " CAP",  shift,  position -   ]
  sla.deg  slant
  dup  1  =
          if    v.h.fla 0 =    if u.v.p   else u.h.p    then
          cr  ."  CAPTION for UPPER port is : "  "dup  "type    cr
          then
  dup  2  =
          if    v.h.fla 0 =    if l.v.p   else l.h.p    then
          cr  ."  CAPTION for LOWER port is : "  "dup  "type    cr
          then
       3  =
          if    v.h.fla 0 =    if   v.p   else   h.p    then
          cr  ."  CAPTION for V / H port is : "  "dup  "type    cr
          then
  x.vup.o  x.vup.s  2. /  +    x.pap.s    *      tmp.v.1  :=
  y.vup.o  y.pap.s *                      +      tmp.v.2  :=
  \                    ? = -0.5
  v.h.fla  0  =
  if     o.v.p
  else   o.h.p
  then
```

```
  tmp.v.1   x.vup.o  x.pap.s *  -      x.pap.s /  x.vup.s /   tmp.v.1  :=
  tmp.v.2   y.vup.o  y.pap.s *  -      y.pap.s /  y.vup.s /   tmp.v.2  :=
  "dup "len  2. /  x.cha.s *  x.cap.a *  x.pap.s /  x.vup.s /      tmp.v
:=
  tmp.v.1  tmp.v -    tmp.v.1  :=
  tmp.v.1   tmp.v.2                       position
  x.cha.s  x.cap.a *    y.cha.s y.cap.a *   cha.siz
  label
  x.cha.s                y.cha.s             cha.siz
;
: u.cap
  -0.5   1  all.cap
;
: l.cap
  -0.5   2  all.cap
;
: 1.cap
  -0.6   3  all.cap
;
\ !7^6----------------------------------------------------------------------6^7!

: x.lab          \  [ x.lab -  ]
  scale.size
  "dup
  sla.deg  slant
  x.cha.s  x.lab.a *    y.cha.s y.lab.a        *  cha.siz
  x.axi.o    x.axi.s 2. /  +      y.axi.o  0.360 *     position
  "dup "len  2. /  x.cha.s *  x.lab.a *  x.pap.s /  x.vup.s /      tmp.v
:=
  x.axi.o    x.axi.s 2. /  + tmp.v -    y.axi.o  0.360 *     position
  cr  ." Input LABEL for X axis : "  "type   cr
  \  centered.label  \   causing shift of axis label
  label
  0. label.dir
  x.cha.s                y.cha.s             cha.siz
;
: y.lab          \  [ y.lab -  ]
  scale.size
  "dup
  sla.deg  slant
  x.cha.s  x.lab.a *    y.cha.s y.lab.a  *   cha.siz
  x.axi.o  .300 *       y.axi.o  y.axi.s  2. /  +     position
  cr  ." Input LABEL for Y axis : "  "type    cr
  "dup "len  2. /  x.cha.s *  x.lab.a *  y.pap.s /  y.vup.s /
  ssx *   ssy /         tmp.v   :=   \  the scales in x and y dir are different
  x.axi.o  .360 *       y.axi.o  y.axi.s 2. /  +  tmp.v -   position
  90. label.dir
  label
  0. label.dir
  x.cha.s                y.cha.s             cha.siz
;

: Lab.l
  0.       slant
  normal.coords
```

```
  0.4   0.3      abs.pos
  0.14  0.18     cha.siz
  " Lee Y.R."    label
  x.cha.s                 y.cha.s              cha.siz
;
: Lab.d
  0.       slant
  normal.coords
  0.4   0.3      abs.pos
  0.14  0.18     cha.siz
  "date          label
  x.cha.s                 y.cha.s              cha.siz
;
: Lab.l.d
  0.   slant
  normal.coords
  0.4   0.3      abs.pos
  0.14  0.18     cha.siz
  " Lee Y.R. "  "date  "cat        label
  x.cha.s                 y.cha.s              cha.siz
;

\ 40 string  data.filename.hp

: set.c.n        \  interactive   [  -  case.name ]
  cr cr
  ." Style of label ::                "          cr
  ."                 (1) CASE :       "          cr
  ."                 (2) other        "          cr
  ." Choose 1 or 2  :  "  #input cr
  1  =
  if
     " CASE : "  cr  inv.on
    ." CASE : "      inv.of          "input  "cat  case.name   ":=
  else           cr  inv.on
    ." Any label content :" inv.of "input        case.name   ":=
  then  cr
  case.name
;

: lab.c          \  [ case.name  -  ]
  scale.size
  "dup  case.name  ":=
  "dup  cr cr                                          color.on  4  foreground
  inv.on    ." Title is : " inv.of      "TYPE CR      color.off
  aut.plo.fla   0  =
  if
    inv.on    ." Is it OK ! " inv.of      ." to exit press <F>"
    pckey  drop  if  set.c.n  then
  then
  sla.deg  slant
  normal.coords
  \ data.filename.hp   case.name   ":=
  \ " CASE : "  case.name   "cat   "dup  "len   tmp.v  :=
                                         "len   tmp.v  :=
```

```
  x.vup.s x.pap.s *      dup  x.cha.s  x.til.a *  tmp.v *     -     swap    /
  y.axi.o y.axi.s +                                      position
  x.cha.s  x.til.a *     y.cha.s y.til.a *          cha.siz
  label
  x.cha.s       y.cha.s      cha.siz
;
\ !8^7--------------------------------------------------------------------7^8!

real scalar x.pos              scalar y.pos
     scalar x.c.siz            scalar y.c.siz

x.cha.s      x.c.siz   :=
y.cha.s      y.c.siz   :=

: LAB              \  interactive
  scale.size
  sla.deg  slant
  normal.coords
  cr  cr
  ." POSITION within port (abs in cm) : {;pos:}"
  ." X.p = " x.pos  .     ."    Y.p = " y.pos   .  x.pos   y.pos   abs.pos
CR
  ." CHAR.SIZE          (abs in cm) : {;cha:}"
  ." X.s = " x.c.siz .    ."    Y.s = " y.c.siz .  x.c.siz y.c.siz cha.siz cr cr
  ." LABEL     : "            "input  LABEL
;
: pos
  cr ." Input absolute position    ( X.p  ,  Y.p )   ::   " cr ."     "
 ." X.p = " #INPUT  x.pos :=         ."    Y.p = " #INPUT y.pos :=     cr
  x.pos   y.pos   abs.pos   CR
;
: cha
  cr ." Input CHAR.SIZE            ( X.s  ,  Y.s )   ::   " cr ."     "
  ." X.s = " #INPUT  x.c.siz :=      ."    Y.s = " #INPUT y.c.siz :=   cr
  x.c.siz y.c.siz cha.siz   cr
;

\ !9^8--------------------------------------------------------------------8^9!

: tom
  o.h.p
  ibm.graphics
  device.init
;

: change.paper
  cr  cr
b.t
  cr  ."  ***********  PLEASE CHANGE PAPER and PRESS ANY KEY ***********"
b.t
  BELL.LEE.3
  KEY DROP
inv.on
  cr  ."  --------- Be sure to change paper AND Press any key ----------" inv.of
  cr
```

```
  BELL.LEE.3
  KEY DROP
;
                                                                    686
nd integer scalar partial.clear

\ ^9-----------------------------------------------------------------9^

.................................................................................................
```

FigA-Ini.asy

勇伯程式庫 A01524-20070125-A075 ❖ January 25, 2007 ❖ [Page count: 2]

```
\   -*- txt:asy -*-   Wednesday, 02/11/20 at 20:52:00
\   The same as FigAPlot.sel (shl)
\ =================================================================
1   string output.flag.str
26  string d&t.string
14  string new.d&t.string

integer scalar new.d&t.flag
real scalar ps.linewidth.now
     scalar ps.linewidth.def
1.0            ps.linewidth.def        :=
1.0            ps.linewidth.now        :=

0   new.d&t.flag   :=

: set.new.d&t
  new.d&t.flag 1 =
  if
    new.d&t.string    ":=
  else
    "drop
  then
;

: set.lc   \   [ line.colors.flag - ]
  lc.flag :=
  lc.flag 0   =
  if
    0    lc1 :=
    lc1 lc2 :=   lc1 lc3 :=   lc1 lc4 :=   lc1 lc5 :=   lc1 lc6 :=
  else
    \   select.color
\       0 lc1  :=   9 lc2  := 12 lc3  := 13 lc4  := 11 lc5  := 10 lc6  := 0 lc7  :=
\       0 lc1  :=   1 lc2  :=   4 lc3  :=   5 lc4  :=   6 lc5  :=   3 lc6  := 0 lc7  :=
     0 lc1  :=   9 lc2  := 12 lc3  :=   5 lc4  :=   6 lc5  :=   3 lc6  := 0 lc7  :=
    \    9 lc1  :=    12 lc2  :=    10 lc3  :=
    \   13 lc4  :=     0 lc5  :=    11 lc6  :=     0 lc7  :=
    \ 0 lc.flag :=
  then
;
: Choose.output
  cr ." Chose output format:"
  cr ."                        M: Monitor"
  cr ."                        E: EPS"
  cr ."                        " " M" get.str
\    "dup s.type
  output.flag.str ":=
  cr ." Chose Color or Black:"
  cr ."                        C: Color"
  cr ."                        B: Black"
  cr ."                        " " C" get.str
  "dup
```

```
  "upper " C" "=
  if
    1 set.lc
  else
    0 set.lc
  then
;
Choose.output
```

FigAPlot.shl

勇伯程式庫 A01525-20070125-A076 ❖ January 25, 2007 ❖ [Page count: 2]

```
\ ==================================================================
1  string output.flag.str
26 string d&t.string
14 string new.d&t.string

integer scalar new.d&t.flag

0  new.d&t.flag  :=

: set.new.d&t
  new.d&t.flag 1 =
  if
    new.d&t.string   ":=
  else
    "drop
  then
;

: set.lc  \  [ line.colors.flag - ]
  lc.flag :=
  lc.flag 0  =
  if
    0   lc1 :=
    lc1 lc2 :=  lc1 lc3 :=  lc1 lc4 :=  lc1 lc5 :=  lc1 lc6 :=
  else
    \  select.color
\      0 lc1  :=  9 lc2  := 12 lc3  := 13 lc4  := 11 lc5  := 10 lc6  := 0 lc7  :=
\      0 lc1  :=  1 lc2  :=  4 lc3  :=  5 lc4  :=  6 lc5  :=  3 lc6  := 0 lc7  :=
     0 lc1  :=  9 lc2  := 12 lc3  :=  5 lc4  :=  6 lc5  :=  3 lc6  := 0 lc7  :=
    \   9 lc1  :=   12 lc2  :=   10 lc3  :=
    \  13 lc4  :=     0 lc5  :=   11 lc6  :=    0 lc7  :=
    \ 0 lc.flag :=
  then
;
: Chose.output
  cr ." Chose output format:"
  cr ."                       M: Monitor"
  cr ."                       E: EPS"
  cr ."                       " " M" get.str
\   "dup s.type
  output.flag.str ":=
  cr ." Chose Color or Black:"
  cr ."                       C: Color"
  cr ."                       B: Black"
  cr ."                       " " C" get.str
  "dup
  "upper " C" "=
  if
    1 set.lc
  else
    0 set.lc
  then
```

```
;
Chose.output
```

FigAPlot.asy

勇伯程式庫 A01526-20070125-A077 ❖ January 25, 2007 ❖ [Page count: 13]

```
: FigAPlot.asy ; \ %----- Asyst Prog

\ %=================================================================
\ For plotting psi curves
\
\ load Llet-0.asy
\ load Llet-cwt.asy
\ set.123
\    wks file
\ set.act
\    5
\    1024
\    start frequency:  0.6
\    # of frequency:1
\ load psiCwks.asy (uses wks file)
\
\ %-----------------------------------------------------------------
\ load this file: psiCplot.asy (need figaplot.asy)
\
\ (Equivalently --
\ load fig-plot.asy
\      and input psiCfb1.fb1 (needs PsiC.bpf)
\ )
\ %=================================================================

\ -----------------------------------------------------------------
\ for auto batch plot of figures
\ %% set line colors & eps/mon in this file  (1 set.lc;  set.eps)
\ %% need          fixed    FigAFB-W.fb
\ %%        interactive   (FigABP.bpf)
\ ^-----------------------------------------------------------------
\ %% Main code for Plotting figures using Asyst data or WKS data
\ %%  1.  Assign WKS file  and  FB2 file to be used
\ %%        Modify and Copy  ?.fb  to, such as,  WKS-FB-1.FB1
\ %%        ^^ (assign wks file to be used)
\ %%        Create xy-pairs batch file, such as, WKS-FB-2.FB2
\ %%        ^^ (assign FB2 file to be used)
\ %%        (design plot style, such as axis, grids, lgns, labels and strings)
\ %%  2.  load this file, which will execute do.plot or do.plot.wks
\ %%  3.  Use either
\ %%        do.plot     <-- Hil-edge.FB, etc (For fig file and render plot)
\ %%                    <-- WKS-FB-1.FB1, etc (for loading wks file)
\ %%        do.plot.wks <-- WKS-FB-2.FB2 (loading xy pairs and render plots)
\ ^-----------------------------------------------------------------

\ %% String assignment
\     0.  misce.str -- at Top Left, can be assigned in FB(2)
\     1.  spe.str.1 -- provided by wks file (element at row 1 column last)
\     2.  spe.str.2 or spe.str.3 -- (CASE) above Legend, use spe.str.2.or.3
\           spe.str.2: can be assigned in FB(2)
\           spe.str.3: default to eps.file.name, not changeable, see plot.bat
\     3   123.str.ary or lgn.str.ary -- Legends for lines, use get.lgns.a.or.d
```

```
\            123.str.ary: provided by wks file (first row)
\            lgn.str.ary: can be assigned in FB2
\ \\ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
\ dir.lee.txt  " def-anan.nam"  "cat defer> load

\ cdtNow
\ load WD.asy
\ cdt
\ \ load def-anan.nam
\ dir.lee.txt " def-anan.nam" "cat defer> load
cdt load def-anan.nam \

12  string   eps.file.name
12  string   FB2.file.name
64  string   Strange.str \ It is very strange!!! Needed for EPS.
64  string   spe.str.1    \ Loaded from wks file
\ 40  string   spe.str.1    \ Loaded from wks file
24  string   spe.str.2    \ assignable
24  string   spe.str.3    \ Default to eps.file.name
1   string   get.lgns.a.or.d
1   string   non.stop.flag
1   string   put.abc.str
26  string   pre.abc.str
26  string   post.abc.str
1   string   1x1y.flag
4   string   axis.fit.flag   \ " x0y1"
integer      scalar  put.abc.flag
             scalar  put.pre.abc.flag
             scalar  put.post.abc.flag
             scalar  put.d&t.flag
             scalar  put.spe.flag
             scalar  put.lgn.flag
             scalar  plot.#
             scalar  acc.plot.#   \  accumulated
             scalar  eps.file.010203.flag
             scalar  spe.str.2.or.3
             scalar  x.lp.pos
             scalar  x.lp.inc
             scalar  y.lp.pos
             scalar  y.lp.inc
             scalar  font.size.xy.label
             scalar  font.size.ll
20  string   pair.name
Dim[ 6 , 20 ] string.array  pair.name.ary
Dim[ 6 , 20 ] string.array  lgn.str.ary
integer      scalar line.#.tmp
             scalar #.line.@
real scalar x.min.w
     scalar x.max.w
     scalar y.min.w
     scalar y.max.w

\ -----------------------------------------------------------------------------
" d"           get.lgns.a.or.d       ":=
" Times-Roman" font.type.n           ":=
```

```
20              font.size.xy.label     :=
20              font.size.ll           :=
18              font.size.l            :=
16              font.size.n            :=
14              font.size.s            :=
0               x.lp.pos               :=
2               x.lp.inc               :=
0               y.lp.pos               :=
2               y.lp.inc               :=
\ ---------------------------------------------------------------------------------
integer scalar         #.line
integer dim[ 12 ] array line.#
integer scalar #.pts
\        scalar new.d&t.flag \ Moved to shell for auto plotling
real    scalar x.max

: change.d&t
  new.d&t.flag  1 =
  if
    new.d&t.string  d&t.string  ":=
  else
    "date   " -"  "cat   "time  "cat   d&t.string  ":=
  then
;
\ change.d&t
\    new.d&t.string  d&t.string  ":=
: labxy.xy
  x.label  y.label
  eps.flag 1 =
  if   0.04
  else 0.02
  then
  eps.flag 1 =
  if
    font.type.n  font.size.xy.label  ps.font
    labxy.stack
    font.type.n  font.size.n         ps.font
  else
    labxy.stack
  then
;
: put.d&t
  put.d&t.flag  1  =
  if
  normal.coords
  0.80  0.97 position  d&t.string  label
  world.coords
  then
;
: put.abc.part
  put.pre.abc.flag 1 =
  if
    pre.abc.str
  else
    "null
```

```
  then
  put.abc.flag  1  =
  if
    " ("  put.abc.str  "cat   " )" "cat
  else
    "null
  then
  put.post.abc.flag 1 =
  if
    post.abc.str
  else
    "null
  then
  "cat  "cat
;
: put.abc.part.res
  put.abc.flag  1  =
  \ eps.flag      1   =  and
  if
    pre.abc.str
    " ("  "cat    put.abc.str  "cat   " )" "cat
  else
    pre.abc.str
  then
;
: put.abc
  normal.coords
  put.abc.part
  Strange.str  ":=  \ It is very strange!!! Needed for EPS.
  Strange.str       \ It is very strange!!! Needed for EPS. Otherwise, No print
  0.20  0.89    position      label
  world.coords
;

: put.spe
  put.spe.flag  1  =
  if
  normal.coords
  spe.str.2.or.3  3 =
  if
    spe.str.3
  else
    spe.str.2
  then

  misce.str  \  spe.str.1 at top left
  2.wnd.flag  1 =   \ assigned by hp.new, dp.new, dp.new
  if
    0.15  0.97    position            label
    0.17
    eps.flag 1 =
    if      0.88
    else    0.91
    then          position            label
  else
```

```
    0.15  0.97    position            label
    \ \    0.17  0.93    position            label
    2.wnd.flag  1 =   \ assigned by hp.new, dp.new, dp.new
    if
      0.17  0.93    position            label
    else
      0.70  0.89    position            label
    then
 then
 world.coords
 then
;
: put.lgn  \ [ line.#, line.posi, "text"  -- ]
  put.lgn.flag  1  =
  if
  normal.coords
  2.wnd.flag  1 =   \ assigned by hp.new, dp.new, dp.new
  if
    0.70
    eps.flag 1 =
    if      0.88
    else    0.91
    then
  else
    0.70   0.85
  then
  \  \    0.70  0.92
  key.orig
      mkl.bat
      key.done
      world.coords
  then
;
\ ********************
: set.eps
  1  eps.flag :=
  " n"  my.pause.flag ":=
  \ 0  lc.flag :=
  \ 0  2in1.flag :=
;
: set.mon
  0  eps.flag :=
  \ 1  lc.flag :=
  \ 1  2in1.flag :=
;
: put.grid.lines   \  equivalent to xy.axis.plot
  grid.flag 1 =
  if
    0.5  ps.linewidth
    vertical
      grid.on
      no.labels
    horizontal
      grid.on
      no.labels
```

```
    xy.axis.plot
  then
  1.0  ps.linewidth
  vertical
    grid.off
    \    0  2  label.points
    x.lp.pos  x.lp.inc     label.points
  horizontal
    grid.off
    \    0  2  label.points
    y.lp.pos  y.lp.inc     label.points
  \  xy.axis.plot
;
\ *******************
\ : set.lc   \  [ line.colors.flag - ]
\   lc.flag :=
\   lc.flag 0  =
\   if
\     0   lc1 :=
\     lc1 lc2 :=  lc1 lc3 :=  lc1 lc4 :=  lc1 lc5 :=  lc1 lc6 :=
\   else
\     \  select.color
\     0  lc1 :=    9 lc2 :=   12 lc3 :=
\     13 lc4 :=   11 lc5 :=   10 lc6 :=    0 lc7 :=
\     \   9 lc1 :=   12 lc2 :=   10 lc3 :=
\     \  13 lc4 :=    0 lc5 :=   11 lc6 :=    0 lc7 :=
\     \ 0 lc.flag :=
\   then
\ ;
\ 1 set.lc
\ ----------------------------------------------------------------------------
\ ---------------------------------------- PS Words ---------------------------
\ vuport ps.hp.port
\    .90  .90 vuport.size
\    .05  .05 vuport.orig
\ vuport ps.up.port
\    .90  .42 vuport.size
\    .05  .51 vuport.orig
\ vuport ps.dp.port
\    .90  .42 vuport.size
\    .05  .07 vuport.orig
vuport ps.hp.port
   .900  .900 vuport.size
   .050  .050 vuport.orig
vuport ps.up.port
   .900  .440 vuport.size
   .050  .510 vuport.orig
vuport ps.dp.port
   .900  .440 vuport.size
   .050  .050 vuport.orig
\ vuport  h.port  0.0000 0.1500 vuport.orig  1.0000 0.8500 vuport.size  outline
: ps.hp
     ps.hp.port
     horizontal  normal.coords  0.85  0.10  label.scale.point  world.coords
\                             0     2     label.points
```

```
                                      x.lp.pos   x.lp.inc      label.points
       vertical     normal.coords   0.07   0.87   label.scale.point   world.coords
\                                     0      2      label.points
                                      y.lp.pos   y.lp.inc      label.points
       ps.graphics ;
: ps.up
       ps.up.port
       horizontal   normal.coords   0.85   0.10   label.scale.point   world.coords
\                                     0      2      label.points
                                      x.lp.pos   x.lp.inc      label.points
       vertical     normal.coords   0.07   0.87   label.scale.point   world.coords
\                                     0      2      label.points
                                      y.lp.pos   y.lp.inc      label.points
       ps.graphics ;
: ps.dp
       ps.dp.port
       horizontal   normal.coords   0.85   0.10   label.scale.point   world.coords
\                                     0      2      label.points
                                      x.lp.pos   x.lp.inc      label.points
       vertical     normal.coords   0.07   0.87   label.scale.point   world.coords
\                                     0      2      label.points
                                      y.lp.pos   y.lp.inc      label.points
       ps.graphics ;
: port.axis.1.m
          horizontal   normal.coords   0.85   0.10   label.scale.point   world.coords
                                      x.lp.pos   x.lp.inc      label.points
\                                         0      2      label.points
          vertical     normal.coords   0.07   0.87   label.scale.point   world.coords
                                      y.lp.pos   y.lp.inc      label.points ;
\                                         0      2      label.points ;
: port.axis.2.m
          horizontal   normal.coords   0.85   0.10   label.scale.point   world.coords
                                      x.lp.pos   x.lp.inc      label.points
\                                         0      2      label.points
          vertical     normal.coords   0.04   0.87   label.scale.point   world.coords
                                      y.lp.pos   y.lp.inc      label.points ;
\                                         0      2      label.points ;
: hp.tmp   h.port  {b.text}  t1.color tmp.c :=  port.axis.1.m  0.02 y.lab.pos :=
           port.color    0  2.wnd.flag := ;
: vp.tmp   v.port  {b.text}  t2.color tmp.c :=  port.axis.2.m  0.02 y.lab.pos :=
           port.color    0  2.wnd.flag := ;
: lp.tmp   l.port  {l.text}  t1.color tmp.c :=  port.axis.2.m  0.95 y.lab.pos :=
           port.color    1  2.wnd.flag := ;
: rp.tmp   r.port  {r.text}  t2.color tmp.c :=  port.axis.2.m  0.95 y.lab.pos :=
           port.color         1  2.wnd.flag := ;
: up.tmp   u.port  {l.text}  t1.color tmp.c :=  port.axis.1.m  0.02 y.lab.pos :=
           port.color    1  2.wnd.flag := ;
: dp.tmp   d.port  {r.text}  t2.color tmp.c :=  port.axis.1.m  0.02 y.lab.pos :=
           port.color    1  2.wnd.flag := ;
: fp.tmp   f.port  {f.text}  t1.color tmp.c :=  port.axis.1.m  0.02 y.lab.pos :=
           port.color    0  2.wnd.flag := ;
: hp.res   hp ;
: up.res   up ;
: dp.res   dp ;
: hp.new
```

```
  eps.flag 1 =
  if       ps.hp  0 2.wnd.flag :=
  else     hp.tmp
  then
\  outline.color color solid  outline
;
: up.new
  eps.flag 1 =
  if       ps.up  1 2.wnd.flag :=
  else     up.tmp
  then
\  outline.color color solid  outline
;
: dp.new
  eps.flag 1 =
  if       ps.dp  1 2.wnd.flag :=
  else     dp.tmp
  then
\  outline.color color solid  outline
;
: hint.tmp
  cr  ." Command:"
\  " 1 set.lc; set.mon; set.eps -> do.plot"  s.type.3
\  " (.wks)" s.type ."  "  " (wks.bat)"  s.type
  " 1 set.lc; set.mon; set.eps -> do.plot"  s.type.3
  ."  "  " [fig.FB]"  s.type
  cr  ." Command:"
  " 1 set.lc; set.mon; set.eps -> do.plot.wks"  s.type.3
  ."  "  " [?.bpf]"  s.type
;
\ -----------------------------------------------------------------------------

\ def.vuport
\ -----------------------------------------------------------------------------
\ ---------------------------------------- Pre Part ---------------------------
: ini.gr.out
  eps.flag 1 =
  if     sd
  \  else  sd
  else  gd
  then
  def.vuport
  \  ps.graphics
  11 8.5 ps.paper.size
  ps.encapsulate
  ps.landscape
  eps.flag 1 =
  if
    vertical
      grid.off
      \ no.labels
    horizontal
      grid.off
      \ no.labels
    \ ---------------
```

```
    \     cr cr   ." ---> EPS file: "
\     cr cr   ." ---> EPS file ["  2 1 fix.format  Plot.# 0 n>s s.type.3  ." ]: "
    cr cr   ." ---> EPS file ["
    put.abc.part  s.type.3  ." /"
    2 1 fix.format  acc.plot.# 0 n>s s.type.3  ." ]: "
    cdf.now.str
    eps.file.010203.flag  1 =
    if
      acc.plot.#    9  <=
      if
        " 0"
      else
      then
      acc.plot.#    0 n>s  "cat
      " .eps"              "cat
      eps.file.name  ":=
    then
    eps.file.name
    non.stop.flag "upper  " Y"  "=
    if
      "cat
    else
      get.str   "dup   eps.file.name ":=    "cat
    then
    \     ." ---> EPS file: "  " 01.eps"  get.str     "cat
    "dup   s.type.3
    defer> ps.file
    ps.graphics
    ps.start
    font.type.n  font.size.n  ps.font
    1.0 ps.linewidth
    ps.hp  ps.up  ps.dp
    \ initializing
  then
;
\ ---------------------------------------- End Part --------------------------
: end.gr.out
  normal.coords
  1  1  position
  world.coords
  eps.flag 1 =
  if
    ps.end
    def.vuport
    ibm.graphics
    \ gd
  else
  then
  eps.flag   0   =
  if
    my.pause
  else
    sd
    cr cr   ." ---> EPS file ["
    put.abc.part  s.type.3  ." /"
```

```
    2 1 fix.format  acc.plot.# 0 n>s s.type.3  ." ]: "
    cdf.now.str    eps.file.name     "cat     s.type.3  ."  "
    " All done!" s.type.2
  then
  hint.tmp
;
: for.mon.gd
  eps.flag 1  = not
  if
    gd
  then
;
\ !**************************************************************************
: do.plot
  plot.#  1  +   plot.#  :=
  acc.plot.# 1 +  acc.plot.# :=
  \
  \ sd   \ \% for debugging

  hint.tmp
  cr  ." Figure body file : "
  \ cr  ."  (Gen:Acc3w1i0.FB, WKS:WKS-1xAy.fb, etc.) "  " C-WC-R_F.FB"  get.str
  cr  ."  (Gen:Acc3w1i0.FB, WKS:WKS-1xAy.FB, etc.) "
  \  " WKS-FB-1.FB1"  get.str
  \  " WKS-FB-1.FB3"  get.str
  \  " WKSaFB-1.FB1"   \   get.str  \ \ FOR AUTO BATCH PLOT
  " FigAFB-W.fb" \ stack.load \ " FigAFB-W.fb" \ get.str \ FOR AUTO BATCH PLOT
  \ ---------------------------------------------------------------------------
  ini.gr.out
  \ ---------------------------------- Begin Body of graph --------------------
  \ dir.lee.txt  "swap   "cat defer> load
  cdt.now.str "swap "cat defer> load
  \
  \ ---------------------------------- End   Body of graph --------------------
  end.gr.out
  \ ---------------------------------------------------------------------------
;
\ " WKS-FB-2.FB2"  FB2.file.name  ":=
stack.load " FigABP.bpf"
FB2.file.name  ":=
\ : do.plot.wks
\   \ ---------------------------------------------------------------------------
\   \ cr  ." WKS Plot Batch file : (C-WC_B.Bat, etc.) "  " C-WC_B.Bat"  get.str
\   cr  ." WKS Plot Batch file : (WKS-FB-2.FB2, etc.) "
\ \  FB2.file.name  get.str  \ \ FOR INTERACTIVE PLOT
\   FB2.file.name  \  get.str  \ \ FOR AUTO BATCH PLOT
\ \   dir.lee.txt  "swap   "cat defer> load
\   cdt.now.str  "swap   "cat defer> load
\   \
\   \ ---------------------------------------------------------------------------
\ ;
: do.plot.wks
  \ ---------------------------------------------------------------------------
  \ cr  ." WKS Plot Batch file : (C-WC_B.Bat, etc.) "  " C-WC_B.Bat"  get.str
  cr  ." WKS Plot Batch file : (WKS-FB-2.FB2, etc.) "
```

```
  \  FB2.file.name  get.str  \ \ FOR INTERACTIVE PLOT
  FB2.file.name  \  get.str  \ \ FOR AUTO BATCH PLOT
  cdt.now.str "swap "cat defer> load
  \
  \ ------------------------------------------------------------------------
;
\ \\ ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
\ ^**********************************************************************

cdt
load 123-A2W.asy \
cdt
load 123-W2A.asy \

token  q.coef   exp.mem> q.coef
integer scalar  extracted.flag
\ ^**********************************************************************
\ ^**********************************************************************
: get.line.#    \ [ string -- ]
    \ %--------  Remember to initial #.line to 0
    "dup  cr  s.type.3
    "len   tmp.v.i.3  :=
    0      tmp.v.i.2  :=
    begin
      tmp.v.i.2  1 +   tmp.v.i.2  :=
      \  ."  " tmp.v.i.2  .
      "dup  "upper
      123.str.ary  "[ tmp.v.i.2 ]  1  tmp.v.i.3  "sub  "upper  "=
     tmp.v.i.2  #.col  =   or
    until
    \ "drop
    "upper  123.str.ary  "[ tmp.v.i.2 ]  1  tmp.v.i.3  "sub  "upper  "=
not
    tmp.v.i.2  #.col  =   and
    if
      cr  ." Array not found - Exit!"
      exit
    else
      #.line  1  +   #.line  :=
    then
    tmp.v.i.2      line.#  [ #.line ]  :=
    ."  No  " 2 0 fix.format   #.line   n.type
    ."  found at " tmp.v.i.2  n.type
;

dir.asy.sys       " 123io.sov" "cat  defer>  load.overlay
: extract.wks    \  [  123.file.name   -- ]
  123.file.name  ":=
  \ %--------------- Open WKS ---------------------------------------------
  cr ." Opening WKS file : "  123.file.name   s.type
  ."  --- "  "  Wait " s.type.2
  dir.asy.sys       " 123io.sov" "cat  defer>  load.overlay
  123.file.name  defer>   123file.open
  \ %--------------- Get all column names -------------------------------------
  cr  ." -->Str:"
```

```
  "  "   123.str.ary   ":=
  1        1
  1        100
  123read.range
  123.str.ary
  123file>"
  "  Done." s.type.3
  \ %--------------- Find the column number for the specification string ------
  cr ." -->Spe:"
  0  #.line  :=
  " $$"   \ cell flag for specification cell
  "dup  s.type
  get.line.#
  \ %%-------------  Get specification string --------------------------------
  1      line.# [ #.line ]
  1      1
  123read.range
  tmp.s
  123file>"
  tmp.s  "len  2 -  3  swap  "sub    tmp.s  ":=
  tmp.s  s.type.3
  \ %%-------------  Find specification parameters ----------------------------
  tmp.s spe.str.1 ":=
  tmp.s
  \ " 12345"  "cat    \  strange behavior here
  "null "cat          \  strange behavior here
  \  \ "dup s.type.2
  tmp.s    ascii ,  "number   #.col    :=   ?drop
  tmp.s   ":=
  tmp.s    ascii )  "number   #.row    :=   ?drop
  tmp.s   ":=
  \  tmp.s    ascii -  "number   tmp.v.i  :=   ?drop
  \  tmp.s   ":=
  tmp.s   ":=  \ recover tmp.s --- need the line : "null "cat !
  \ %%-------------  Read data -----------------------------------------------
  cr ." -->Data:"
  11       1
  #.row   #.col
  123read.range  \ \ in double precison

  123file>unnamed.array
  becomes> q.coef
  q.coef  trans[ 1 , 2 ]  becomes> q.coef
  \ %%-------------  Close WKS -----------------------------------------------
  123file.close
  release.overlay
  1  extracted.flag  :=
  "  Done." s.type.3
;
release.overlay

: set.def
\   1      grid.flag              :=
  0      grid.flag             :=
  1      put.lgn.flag          :=
```

```
  1      put.d&t.flag              :=
  1      put.spe.flag              :=
  1      put.abc.flag              :=
  0      plot.#                    :=
  0      acc.plot.#                :=
  0      #.line                    :=
  1      eps.file.010203.flag      :=
  " 01.eps" eps.file.name         ":=
  " y"   non.stop.flag            ":=   \ for eps.file
  3      spe.str.2.or.3            :=   \ use  spe.str.2  or  spe.str.3
  \ 1      new.d&t.flag               :=
  change.d&t
;

\ ^**************************************************************************
\ Don't change here
\ ^**************************************************************************
  set.def
\  1 set.lc
  set.mon
  sd
  do.plot
\ ^**************************************************************************
\ ^**************************************************************************
\ ^**************************************************************************

\ \ %=================================================================
\    0   set.lc
\    \ set.mon
\    set.eps
\    DO.PLOT.WKS   \ FOR AUTO PLOT
\ \ %=================================================================
\ %=================================================================
: eps.plot
  \ 0   set.lc
  \ set.mon
  set.eps
  DO.PLOT.WKS   \ FOR AUTO PLOT
;
\ %=================================================================
: mon.plot
  \ 1   set.lc
  set.mon
  \ set.eps
  DO.PLOT.WKS   \ FOR AUTO PLOT
;
\ %=================================================================
\ 1 string output.flag.str
: output.plot
  output.flag.str "upper
  " E" "= if eps.plot else mon.plot then
;
output.plot
```

WPon22A3.BPF

勇伯程式庫 A01532-20070125-A083 ❖ January 25, 2007 ❖ [Page count: 3]

```
\ %% Wavelet Blowup  -*- txt:asy -*-
\ -----------------------------------------------------------------
\      misce.str                          d&t   spe.str.1 : From wks $$Mark
\      -----------------------------------      spe.str.2 : Assigned
\      | X(a)Y      Case (spe.str.2.or.3) |     spe.str.3 : eps.file.name
\      |             lgn1 (put.lgn.flag)  |
\      |             lgn2                 |     123.str.ary : From wks 1st row (d)
\     Y|                                  |     lgn.str.ary : Assigned (a)
\      |                                  |
\      |                                  |     Case :
\      |                                  |     (put.spe.flag and spe.str.2.or.3)
\      -----------------------------------      Legend :
\                       X                       (put.lgn.flag and get.lgns.a.or.d)
\ -----------------------------------------------------------------
\ %% String assignment
\     0.  misce.str -- at Top Left, can be assigned in FB(2)
\     1.  spe.str.1 -- provided by wks file (element at row 1 column last)
\     2.  spe.str.2 or spe.str.3 -- For (CASE) above Legend, use spe.str.2.or.3
\           spe.str.2: can be assigned in FB(2)
\           spe.str.3: default to eps.file.name, not changeable, see plot.bat
\     3   123.str.ary or lgn.str.ary -- Legends for lines, use get.lgns.a.or.d
\           123.str.ary: provided by wks file (first row)
\           lgn.str.ary: can be assigned in FB2
\ -----------------------------------------------------------------
\ =================================================================
  " n"  my.pause.flag ":=
  " y"  non.stop.flag ":=  \ for eps.file
  1 eps.file.010203.flag :=
  0 #.line               :=
\ =================================================================
  \ ---------------------------------------- ■ figure annotation
  1 put.lgn.flag    :=
  1 put.d&t.flag    := \ at top right corner
  1 put.spe.flag    := \ (CASE) above Legend
  3 spe.str.2.or.3  := \ use spe.str.3 (eps.file.name) for (CASE)
  2 spe.str.2.or.3  := \ use spe.str.2 (assigned) for (CASE)
  0 x.lp.pos        := \ for label.points
  2 x.lp.inc        :=
  0 y.lp.pos        :=
  2 y.lp.inc        :=
 " d"               get.lgns.a.or.d ":= \ use Assigned or Default lgn
  spe.str.1         misce.str       ":= \ at top left corner
  "null             spe.str.2       ":=
  " 1"  lgn.str.ary  "[ 1 ]  ":=
  " 2"  lgn.str.ary  "[ 2 ]  ":=
  " 3"  lgn.str.ary  "[ 3 ]  ":=
  " 4"  lgn.str.ary  "[ 4 ]  ":=
  " 5"  lgn.str.ary  "[ 5 ]  ":=
  " 6"  lgn.str.ary  "[ 6 ]  ":=
  \ ---------------------------------------- ■ labeling a, b, .. subfigs
  0   put.abc.flag      := \ for labeling subfigs, e.g., (a)
  1   put.pre.abc.flag  := \ e.g. M-(a)
```

```
  1   put.post.abc.flag := \ e.g. M-(a)-5
  -32 plot.#            := \  A, B, .. (-32: 65 -> A)
  -0  plot.#            := \  a, b, .. (  0: 97 -> a)
  "null                pre.abc.str   ":=
  "null                post.abc.str ":= \ "null
\ =================================================================
  \ ---------------------------------------- ■ axes
  1       grid.flag              :=
  " x0y1" axis.fit.flag        ":=
  10   8  axis.divisions
  0.      x.min.w                :=
  500.    x.max.w                :=
  -4.     y.min.w                :=
  4.      y.max.w                :=
  0       Find.y.MinMax.flag     := \ 0: disable G.or.L.y.MinMax.Flag
  1       Find.y.MinMax.flag     := \ 1: recalculate y.min.w and y.max.w
  " G"    G.or.L.y.MinMax.Flag ":= \ full global q array (include unlisted cols)
  " L"    G.or.L.y.MinMax.Flag ":= \ all local listed and exclude unlisted cols
  " Point series"      x.label      ":=
  " Coefficient"       y.label      ":=
\ =================================================================
\ =================================================================
" Wavelet packet"         pre.abc.str   ":=
  "  (" G.or.L.y.MinMax.Flag "cat " )" "cat  post.abc.str       ":=
\ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
" WP1on22A"  eps.file.name  ":=
" ON22A"  eps.file.name  ":=
1 eps.file.010203.flag :=
2 spe.str.2.or.3 := \ use spe.str.2 (assigned) for (CASE)
" 3" eps.file.name "cat eps.file.010203.root ":= \ for eps output file name
eps.file.name spe.str.2            ":= \ for case labeling
" @-PAIRS"
" P1L0"
 plot.bat
\ =================================================================
" @-PAIRS"
" P1L1"
 plot.bat
\ =================================================================
" @-PAIRS"
" P1L2"
 plot.bat
\ =================================================================
" @-PAIRS"
" P1L3"
 plot.bat
\ =================================================================
" @-PAIRS"
" P1L4"
 plot.bat
\ =================================================================
" @-PAIRS"
" P1L5"
 plot.bat
\ =================================================================
```

APPENDIX — Mathematica Programs

(* TFR-WPT Shell *)

(* by Ron Lee *)

†

View Core File in NB Fmt | View Core File in M Fmt

View Original Single – File – Processing (the base of the core file)

†

COLOR SCHEME (* Hyperlinks: Top Colors Files Map Signal End *)

```
commondir = "g:\\all-mma\\common\\";
currentdir = "g:\\all-mma\\TFR-WP\\";
datadir = "c:\\per\\mat\\0-map\\data\\";
Get[commondir <> "SetNotebook_nb-m.m" ]
(* Set notebook options *)
Clear["noext*"];
noextbat = "yes";
colorstyle = 1;  (* 0:B& W ; 1:Color*)
ampflag = 1;
(* 0:normal below ; 1:db below ; 2:db above 3:Log below*)
dbcutoff = 40;
huebeg = 0.15;  (* 0.5:blue ; 0.2:yellow *)
xstr = "Time (sec)";
ystr = "Signal";
```

```
General::spell1 : Possible spelling error:
   new symbol name "ystr" is similar to existing symbol "xstr".
```

```
(* runmap[x_]:=
   {noextnew=x, <<"c:/per/mat/0-map/WP-Core_nb-m.m"}; *)
runmap[x_] := {noextnew = x,
   Get[currentdir <> "TFR-WP_Shell-Core.nb.m"]};
```

† BATCH PROCESSING (* Hyperlinks: Top Colors Files Map Signal End *)

```
datafilename = "0";
(* noext="a0m10205";
runmap["0"];
runmap["a0m10205"];
*)
(*
runmap["chip-al"];
runmap["0"];
*)
runmap[datafilename];
```

```
Remove::rmnsm : There are no symbols matching "*wp*".

Remove::rmnsm : There are no symbols matching "gf*".
```

Data file (Default-A) : c:\per\mat\0-map\data\Chirp-al

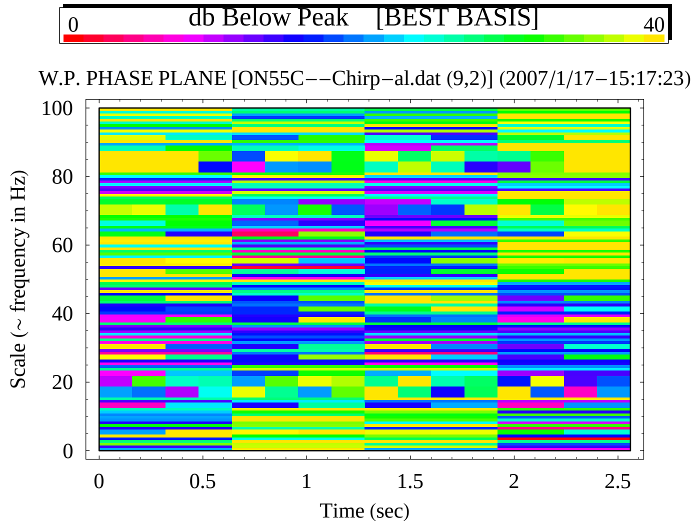
Frame Labels {x,y} : Time (sec)Signal

Sampling Rate : 200.

Freq index (beg, end, {secbeg, secend}) : {1, 512, {1, 9}}

```
StringTake::take : Cannot take positions 1 through 4 in "50.".
```

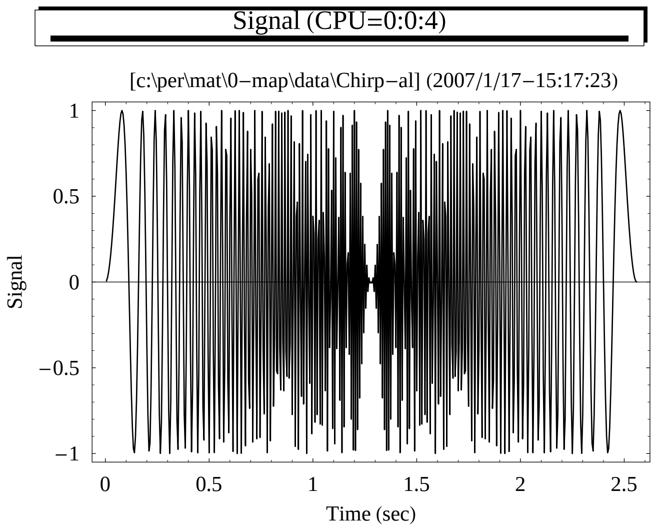
Map




```
Display::pserr : PostScript language error:
Warning…nt Utopia-Regular for Times-Roman
```

{2007, 1, 17, 15, 17, 20} CPU:(00, 00, 0.172);
Time:(00, 00, 04)

```
Signal
```




```
Display::pserr : PostScript language error:
Warning…nt Utopia-Regular for Times-Roman
```

```
{2007, 1, 17, 15, 17, 20}     CPU:(00, 00, 0.297);
Time:(00, 00, 06)
```

```
No TeX Equation.
```

```
(* Hyperlinks: Top  Colors  Files  Map  Signal  End *)
```

(* Time-Frequency Rendition of Wavelet Packet Transform*)

(* by Ron Lee *)

(*

Notes on `Shell Processing` OR `Single-file Stand-alone Processing`:

This file is also the base of the core file of TFR-WPT shell in ".nb" format `(View Shell File in NB Fmt)`. It needs to be converted to "text (.M)" format for use in the shell `(View Shell's Core File in M Fmt)`.

A: For `Single-file Stand-alone Processing` :
Uncomment all elements in cell labeled as 'Shell processing OR Single-file processing'
B: For `Shell Processing` :
Steps for Making M-file:
0: Rename this file as the Shell's core file in nb fmt (e.g., TFR-WPT_Shell-Core.nb).
1. Comment out all elements in cell labeled as 'Shell processing OR Single-file processing'
2. Delete all outputs
3. Remove In/Out names
4. Just "saved as special -> text ->, ?.m"
5. Put the file name in shell
Done!

*)

■ (* DISPLAY SCHEME *) (* Hyperlinks: Top Files Colors Map Signal End *)

```
commondir = "g:\\all-mma\\common\\";
currentdir = "g:\\all-mma\\TFR-WP\\";
datadir = "c:\\per\\mat\\0-map\\data\\";
FigDir = "G:\\Fig\\MMA\\TFR-WP\\";
Get[commondir <> "000-p1_nb-m.m"];
Get[commondir <> "Run-form-1w.m"];
Get[commondir <> "fig-fram_nb-m.m"];
Get[commondir <> "000-word_nb-m.m"];

Remove["p*", "*wp*", "gf*", "lgn*"]
<< "e:/math/4.0/addons/standardpackages/graphics/legend.m"
currentfile = ReadList[datadir <> "cur-map.fip", String];
(*
noext="chirpd";
noext="envelope";
noext="chirp-de";  (* inverse decreasing freq ~ 1/t*)
noext="chirp-ny";  (* Nyqust sampling *)
noext="chirp-al";  (* Nyqust sampling *)
noext="a0m10205";
Clear["noext*"]
*)
Clear["noext*"]
noext = currentfile[[1]];
Which[noextbat === "yes",
    Which[noextnew === "0", noext = currentfile[[1]],
          noextnew =!= "0", noext = noextnew ] ];
noext = datadir <> noext;
mouttt["Data file" <> Which[
          noextbat =!= "yes", " (Default-A) : ",
          noextbat === "yes", Which[noextnew === "0", " (Default-B)
                                   noextnew =!= "0", " (Assigned) :
```

Data file (Default-A) : c:\per\mat\0-map\data\Chirp-al

```
(* Shell processing  OR  single-file processing  *)
(* COLOR SCHEME *)  (* Hyperlinks: Top  Files  Colors  Map
Signal  End  *)
```

```
colorstyle=1; (*0:B&W ; 1:Color*)
ampflag=1;    (* 0:normal below; 1:db below ; 2:db above
3:Log below*)
dbcutoff=20;
huebeg=0.2;   (* 0.5:blue ; 0.2:yellow *)
xstr="Time (sec)"; ystr="Signal";
(*
*)
```

```
mouttt["Frame Labels {x,y} : ", {xstr, ystr}];
secbeg=1;
secend=9;
logbase=10.;
```

```
Frame Labels {x,y} : Time (sec)Signal
```

(* AUTO INPUT OF DETAILS *)

```
(* ===== Auto Input of Complete ASYST Program Details and Data =====
xymin=ReadList[noext<>".mip", Number, RecordLists->True];
xymax=ReadList[noext<>".map", Number, RecordLists->True];
wpsig=ReadList[noext<>".sip", Number];
wpamp=ReadList[noext<>".amp", Number];
wpspe=ReadList[noext<>".spp", Number];
wpstr=ReadList[noext<>".stp", String];
npower=wpspe[[1]];
jwplevel=wpspe[[2]];
bestflag=wpspe[[3]];
bestlevel=wpspe[[4]];
sfreq=wpspe[[5]];mouttn["Sampling Rate : ", sfreq];
(*  secbeg=wpspe[[6]] + 1 ; *) (* jwp.level + 1 *)
wpsigx=Table[ x/sfreq, {x, 1, 2^npower, 1}] ;
wpsigy=wpsig ;
wpsig=Table[ {wpsigx[[x]], wpsigy[[x]]}, {x, 1, 2^npower, 1} ] ;
wlet=wpstr[[1]];
nowfile=wpstr[[2]];

FigName=FigDir<>nowfile<>"_"<>wlet<>"_"

Sampling Rate : 200.

General::spell1 : Possible spelling error: new symbol name "wpsigx" is similar

General::spell : Possible spelling error: new symbol name "wpsigy" is similar
```

G:\Fig\MMA\TFR-WP\Chirp-al.dat_ON55C_

(* ZOOM IN *)

```
(* ==Parameters reset ( from 1 to n.power ; reset here ) === *)
(*
secbeg=1;
secend=npower
secend=8
*)
(*  secbeg=wpspe[[2]] + 1 ; *)
(* Set image size
myfontsize=10;
setis={imagesizex=432,imagesizey=imagesizex/1.,myfontsize=10,
       aspectratiox=1.11,lgnposy=0.99};
*)
setis={imagesizex=360,imagesizey=imagesizex/1.4,myfontsize=8,
       aspectratiox=1.55,lgnposy=0.70};
myfont = "Times";
$TextStyle = {FontFamily -> "Times", FontSize -> myfontsize};
```

(* LEGENDS *)

```
beststr=Which[ bestflag===1,  "    [BEST BASIS]"
              , bestflag===2,  "    [BEST LEVEL "<>"("<>
               ToString[bestlevel]<>")"<>"]" ];

(* ---------------------------------------------- *)
ibeg=If[ secbeg===1,  1, 2^(secbeg-1)+1 ];   (* index *)
iend=2^secend;
mouttn["Freq index (beg, end, {secbeg, secend}) : ",
                    {ibeg, iend, {secbeg, secend}} ];
```

```
 Freq index (beg, end, {secbeg, secend}) : {1, 512, {1,
9}}
```

```
tbeg=0. ;    (* absolute range of plot area *)
tend=2^npower ;
fbeg=If[ secbeg===1,  0, 2^(secbeg-1) ];
fend=2^secend ;

tbegstr="0.00";
tmidstr=StringTake[ToString[
```

```
             N[ 2^npower/sfreq/2+0.000001, 20]],4] ;
 tendstr=StringTake[ToString[
             N[ 2^npower/sfreq  +0.000001, 20]],4] ;
 fbegstr=If[ secbeg===1, "0.00", StringTake[ToString[
             N[ 2^(secbeg-1-npower)*sfreq/2+0.000001, 20]],4]  ] ;
 fmidstr=If[ secbeg===1, StringTake[ToString[
             N[ 2^(secend-npower)*sfreq/2/2+0.000001, 20]], 4],
                         StringTake[ToString[
             N[(2^(secend-secbeg-npower)+ 2^(secbeg-1-npower))
                                 *sfreq/2+0.000001, 20]],4] ] ;
 fendstr=StringTake[ToString[
             N[ 2^(secend-npower)*sfreq/2+0.000001, 20]],4];
 datelist=Date[];
 gfdate=
     "("<>ToString[datelist[[1]]]<>"/"<>ToString[datelist[[2]]]<>
     "/"<>ToString[datelist[[3]]]<>"-"<>ToString[datelist[[4]]]<>
     ":"<>ToString[datelist[[5]]]<>":"<>ToString[datelist[[6]]]<>")";
 lgnlb=Which[ ampflag===0, "Normalized Maganitude"
            , ampflag===1, "db Below Peak"
            , ampflag===2, "db Above Bottom"
            , ampflag===3, "Normalized Log Mag" (* poor resolution?!
            ]<>beststr;
 lgnlb=StyleForm[lgnlb, FontFamily->myfont, FontSize->myfontsize+2
 lgnup=Which[ ampflag===0, "1"
            , ampflag===1, "0"
            , ampflag===2, ToString[ dbcutoff ]
            , ampflag===3, "1"
            ];
 lgndn=Which[ ampflag===0, "0"
            , ampflag===1, ToString[ dbcutoff ]
            , ampflag===2, "0"
            , ampflag===3, "0"
            ];
```

```
StringTake::take : Cannot take positions 1 through 4 in "50.".
```

■ (* FUNCTIONS *)

□ (* PLOT 1 *)

```
(* ---------------------------------------------- *)
wpamp=Which[ ampflag===0, Abs[wpamp]
           , ampflag===1, Abs[wpamp]
           , ampflag===2, Abs[wpamp]
           , ampflag===3, Log[logbase, Abs[wpamp]]
           ];
(* shift log scale to >= 0 in the plot range *)
wpamp=Which[ ampflag===0, wpamp
           , ampflag===1, ( Log[logbase, Max[Take[wpamp, {ibeg, iend}]]]
                          - Log[logbase, wpamp] ) * 20. (*small->larger*)
           , ampflag===2, ( Log[logbase, wpamp]
                          - Log[logbase, Min[Take[wpamp, {ibeg, iend}]]]
                          ) * 20.  (* small->small*)
           , ampflag===3, Abs[Min[Take[wpamp, {ibeg, iend}]]]+wpamp
           ];
   (*ListPlot[Take[wpamp, {ibeg, iend}], PlotJoined->True, Frame->True]
   *)
(* normalizing to 0~1 *)
wpamp=Which[ ampflag===0, wpamp / Max[Take[wpamp, {ibeg, iend}]]
           , ampflag===1, ( dbcutoff - wpamp ) / dbcutoff
           , ampflag===2, ( dbcutoff - wpamp ) / dbcutoff
           , ampflag===3, wpamp / Max[Take[wpamp, {ibeg, iend}]]
           ];
(* trimming *)
champ=Which[ ampflag===0, wpamp
           , ampflag===1, Do[ If[wpamp[[x]]<0., wpamp[[x]]=0.
                          , wpamp[[x]]], {x, ibeg, iend, 1} ]
           , ampflag===2, Do[ If[wpamp[[x]]<0., wpamp[[x]]=0.
                          , wpamp[[x]]], {x, ibeg, iend, 1} ]
           , ampflag===3, wpamp
           ];
(* proper reverse *)
wpamp=Which[ ampflag===0, 1 - wpamp   (* higher amp is darker(0)
           , ampflag===1, 1 - wpamp
           , ampflag===2, wpamp
           , ampflag===3, 1 - wpamp
           ];
   (*Max[Take[wpamp, {ibeg, iend}]]
     Min[Take[wpamp, {ibeg, iend}]]
     ListPlot[Take[wpamp, {ibeg, iend}], PlotJoined->True, Frame->True]
   *)
wpamp=Which[ colorstyle===0, wpamp
           , colorstyle===1, 1-wpamp
           ];
(* scaling color *)
wpamp=Which[ colorstyle===0, wpamp
```

```
            , colorstyle===1, 1 - (1-wpamp) (1-huebeg)
            ];


(* ---------------------------------------------- *)
gffunc[x_]:= Graphics[
    Which[
           colorstyle===1
       (* , {Hue[0, 1-wpamp[[x]], 1-wpamp[[x]]] *)
         , {Hue[wpamp[[x]]]
                            , Rectangle[xymin[[x]], xymax[[x]]]}
         , colorstyle===0
         , {GrayLevel[wpamp[[x]]], Rectangle[xymin[[x]], xymax[[x]]]}
         ]
    ]


gftable=Table[gffunc[x], {x, ibeg, iend}];



(* ---------------------------------------------- *)
gfaxis={
(*   ------------  For interger x and y coordinates
      Graphics[Text[FontForm["Time (sec)", {aft, afs}]
      , {(tend-tbeg)/2+tbeg, (fend-fbeg)*-0.12+fbeg } ]]
    , Graphics[Text[FontForm["Scale (~ Frequency in Hz)",
{aft,afs}]
      , {(tend-tbeg)*-0.11+tbeg, (fend-fbeg)/2+fbeg }, {-1, 0},
{0, 1} ]]
    , Graphics[Text[FontForm[tbegstr,  {aft, afs}]
      , {  tbeg, (fend-fbeg)*-0.045+fbeg } ]]
    , Graphics[Text[FontForm[tmidstr, {aft, afs}]
      , {(tend-tbeg)/2+tbeg, (fend-fbeg)*-0.045+fbeg} ]]
    , Graphics[Line[{{(tend-tbeg)/2+tbeg, fbeg                 },
                     {(tend-tbeg)/2+tbeg, fbeg+(fend-
fbeg)*-0.02}}] ]
    , Graphics[Text[FontForm[tendstr,  {aft, afs}]
      , { tend, (fend-fbeg)*-0.045+fbeg } ]]
    , Graphics[Text[FontForm[fbegstr,  {aft, afs}]
      , {(tend-tbeg)*-0.045+tbeg, fbeg} ]]
    , Graphics[Text[FontForm[fmidstr,  {aft, afs}]
      , {(tend-tbeg)*-0.045+tbeg, (fend-fbeg)/2+fbeg} ]]
    , Graphics[Line[{{tbeg,                  (fend-fbeg)/2+fbeg},
                     {tbeg+(tend-tbeg)*-0.02, (fend-
fbeg)/2+fbeg}}] ]
    , Graphics[Text[FontForm[fendstr,  {aft, afs}]
      , {(tend-tbeg)*-0.045+tbeg, fend} ]]
    ,
      Graphics[
      Line[{{tbeg, fbeg}, {tend, fbeg},
            {tend, fend}, {tbeg, fend}, {tbeg, fbeg}}] ]
*)
      Graphics[
      Line[{{tbeg/sfreq, fbeg/fend*sfreq/2}, {tend/sfreq,
fbeg/fend*sfreq/2},
            {tend/sfreq, fend/fend*sfreq/2}, {tbeg/sfreq,
```

```
fend/fend*sfreq/2},
            {tbeg/sfreq, fbeg/fend*sfreq/2}}] ]
    };
```

(* PLOT 2 *)

```
(* ---------------------------------------------- *)
gfshowmap=Show[   (* ShowLegend[ *)
    {
      gftable
    , gfaxis
    }
    , AspectRatio->1/aspectratiox
    , TextStyle->{FontFamily->myfont, FontSize->myfontsize}
    , Frame->True
    , FrameLabel->{"Time (sec)", "Scale (~ frequency in Hz)"}
    , PlotLabel->
      StyleForm["W.P. PHASE PLANE "<>"["<>ToUpperCase[wlet]<>"--"<>
               nowfile<>" ("<>
               ToString[npower]<>","<>ToString[jwplevel]<>")"<>
               "] "<>gfdate,
               FontFamily->myfont, FontSize->myfontsize ]
    , DisplayFunction->Identity
  (*~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    , Background->Hue[0.6]
    , PlotRegion->{{-0.1,1.1},{-0.1,1.1}}
    ----------------------------------------------------------------------
  *)
   ];


pmap=ShowLegend[
    gfshowmap
    , Which[
          colorstyle===1
      (* , {Hue[0, #, #]&, 30, lgnup, lgndn *)
         , {Hue[1-(#)(1-huebeg)]&, 30, lgnup, lgndn
           , LegendBorderSpace -> 0.2
           , LegendOrientation->Horizontal
           , LegendPosition->{-0.81, +lgnposy}
           , LegendSize->{1.7, 0.10}
           , LegendTextOffset->{0.0, -0.70}
           , LegendLabel->lgnlb
           , LegendLabelSpace->+2.65
           , LegendShadow->{0.01, 0.01}
(*
           , LegendBorderSpace -> 0.2
           , LegendOrientation->Horizontal
           , LegendPosition->{-0.77, +0.75}
           , LegendSize->{1.5, 0.10}
           , LegendTextOffset->{0.0, -1.00}
           , LegendLabel->lgnlb
           , LegendLabelSpace->+2.65
```

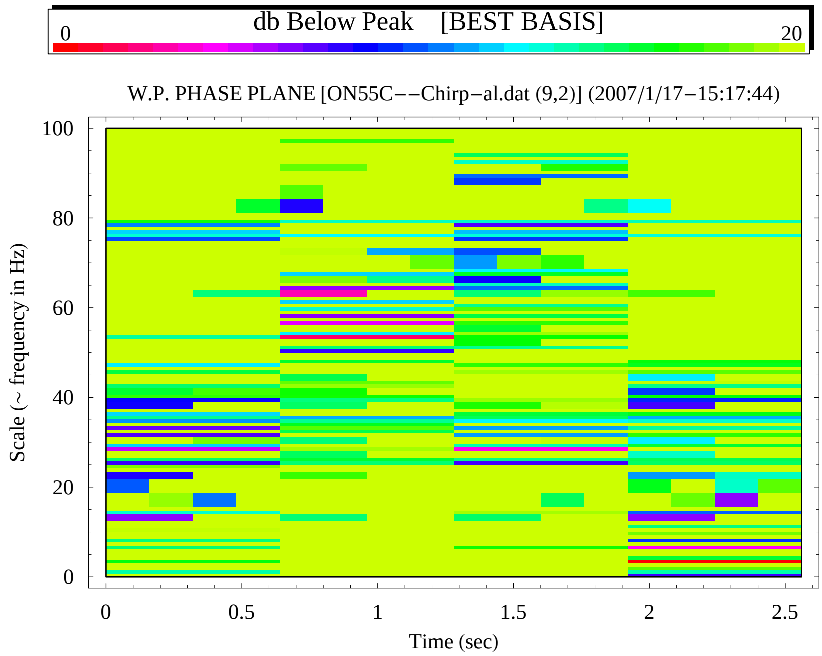
```
            , LegendShadow->{0.01, 0.01}
*)
            }
          , colorstyle===0
            , {GrayLevel[#]&, 30, lgnup, lgndn
            (*  , LegendBorderSpace -> 0.8  for 60  *)
            , LegendBorderSpace -> 0.2
            , LegendOrientation->Horizontal
            , LegendPosition->{-0.81, +lgnposy}
            , LegendSize->{1.7, 0.10}
            , LegendTextOffset->{0.0, -0.70}
            , LegendLabel->lgnlb
            , LegendLabelSpace->+2.65
            , LegendShadow->{0.01, 0.01}
            }
          ]
    , ImageSize->{imagesizex,imagesizey}
    , DisplayFunction->Identity
    ];



(* ------------------------------------------- *)
p2=Show[
   {
    ListPlot[ wpsig
    (* p2=ListPlot[ wpsig  *)
    , PlotJoined->True
    , AspectRatio->1/aspectratiox
    , Frame->True
    , FrameLabel->
      If[ StringMatchQ[StringTake[noext, -1 ], "4"
          ] || StringMatchQ[StringTake[noext, -1 ], "2"]
        , { "Time (sec)",
             If[ StringMatchQ[StringTake[noext, -1 ], "4"]
                , "Displacement (cm)"
                , "Acqueous flow (cm/sec)" ]
          }
      (*  , {"X value",  " Y value"}  *)
        , {xstr, ystr}
      ]
    , PlotLabel->
      StyleForm[" "<>"["<>noext<>
                "] "<>gfdate,
                FontFamily->myfont, FontSize->myfontsize ]
    , ImageSize->{imagesizex,imagesizey}
    , DisplayFunction->Identity
    ]
  }
  ] ;
(* ------------------------------------------- *)
prule=Graphics[Line[{{0.0,0.0},{0.5,0.5}}]
    , DisplayFunction->Identity
    ];
```

■ (* RENDERING *)

```
StylePrint["Map", "Graphics"
          , CellFrame->True, Background->GrayLevel[0.9], FontSize->10
          , CellTags->"Map"
          ];
Show[Graphics[pmap]
    , DisplayFunction->$DisplayFunction];
Display[FigName<>"Map"<>".eps", Graphics[pmap], "EPS"]
(*
StylePrint[
Show[Graphics[pmap]
    , DisplayFunction->$DisplayFunction];
          , "Graphics", CellTags->"Map"
          ];
*)

(*   ------------  For interger x and y coordinates   *)
(*
Show[Graphics[{ Rectangle[{0.045, 0.00}, {0.975, 1.00}, p2]
              , Rectangle[{0.000, 0.0}, {0.0000001, 0.0000001}, prule]
              , Rectangle[{0.999, 0.0}, {0.9990001, 0.0000001}, prule]
    } ]] ;
*)
Get[commondir<>"000-p2_nb-m.m"];
lgnlbsig=
  StyleForm["Signal"<>" (CPU="<>
           ToString[hrd]<>":"<>ToString[mind]<>":"<>
           ToString[secd]<>")",
           FontFamily->myfont, FontSize->myfontsize+2 ];
p4=ShowLegend[
    p2  (* %  *)
    , Which[
           colorstyle===1
           (* , {Hue[0, #, #]&, 10, lgnup, lgndn *)
           (* , {Hue[1-(#)(1-huebeg)]&, 30, lgnup, lgndn  *)
           (* , {Hue[0.75]&, 30, " ", " "  *)
           , {GrayLevel[0]&, 30, " ",  " "
           , LegendBorderSpace -> 0.8
           , LegendOrientation->Horizontal
           , LegendPosition->{-0.81, +lgnposy}
           , LegendSize->{1.7, 0.10}
           , LegendTextOffset->{0.0, -0.70}
           , LegendLabel->lgnlbsig
           , LegendLabelSpace->+2.65
           , LegendShadow->{0.01, 0.01}
(*
           , LegendBorderSpace -> 0.8
           , LegendOrientation->Horizontal
           , LegendPosition->{-0.77, +0.75}
           , LegendSize->{1.5, 0.10}
           , LegendTextOffset->{0.0, -1.00}
           , LegendLabel->"Signal"<>" (CPU="<>
```

```
              ToString[hrc]<>":"<>ToString[minc]<>":"<>
              ToString[secc]<>")"
            , LegendLabelSpace->+2.65
            , LegendShadow->{0.01, 0.01}
*)
            }
          , colorstyle===0
            , {GrayLevel[0]&, 30, " ",  " "
            , LegendBorderSpace -> 0.8
            , LegendOrientation->Horizontal
            , LegendPosition->{-0.81, +lgnposy}
            , LegendSize->{1.7, 0.10}
            , LegendTextOffset->{0.0, -0.70}
            , LegendLabel->lgnlbsig
            , LegendLabelSpace->+2.65
            , LegendShadow->{0.01, 0.01}
            }
          ]
    , ImageSize->{imagesizex,imagesizey}
    , DisplayFunction->Identity
    ];
StylePrint["Signal", "Graphics"
          , CellFrame->True, Background->GrayLevel[0.9], FontSize->10
          , CellTags->"Signal"
          ];
Show[Graphics[p4]
    , DisplayFunction->$DisplayFunction
    ];
Display[FigName<>"Signal"<>".eps", Graphics[p4], "EPS"]

Get[commondir<>"000-p2_nb-m.m"]
Get[commondir<>"000-p3.m"]  (* Hyperlinks: Top Files Colors
```

Map



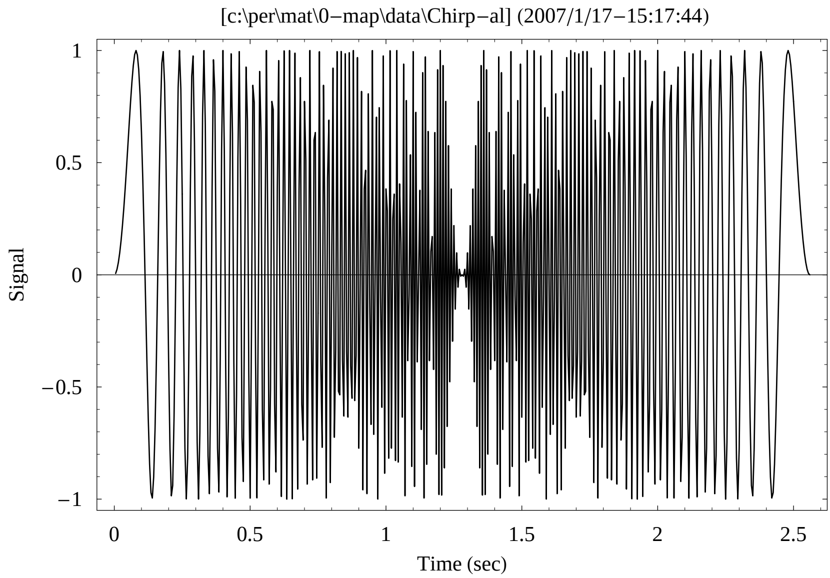

```
Display::pserr : PostScript language error:
Warning:…ont Utopia-Regular for Times-Roman
```

- Graphics -

{2007, 1, 17, 15, 17, 42} CPU:(00, 00, 0.172);
Time:(00, 00, 04)

```
Signal
```

Signal (CPU=0:0:4)




```
Display::pserr : PostScript language error:
Warning: …ont Utopia-Regular for Times-Roman
```

- Graphics -

{2007, 1, 17, 15, 17, 42}   CPU:(00, 00, 0.297);
Time:(00, 00, 06)

No TeX Equation.

■ (* Hyperlinks: Top Files Colors Map Signal End *)

(* WVTR Shell *)

(* by Ron Lee *)

(*! IMPORTANT: Firstly, run non-shell to initialze properly !*)

View Core File in NB Fmt | View Core File in M Fmt

View Stand Alone Single - File Processing (the base of the core file)

```
datafilename = WV-Demo (*0:forDefault;Allothers:specify*)
datafilename = 0;          (*0:forDefault;Allothers:specify*)
Hyperlinks :
```

Files | Parameters | Modulus | Phase | Signal | 3 DModulus | 3 DPhase | End

```
datafilename = "0"; (* "0" : for Default;  All others : specify  *)
```

```
commondir = "g:\\all-mma\\common\\";
currentdir = "g:\\all-mma\\TFR-WV\\";
datadir = "c:\\per\\mat\\0-CWT\\data\\";
Get[commondir <> "SetNotebook_nb-m.m" ]
(* Set notebook options *)
Clear["noext*"]; noextbat = "yes";
```

COLOR SCHEME

■ Modulus

```
colorstylemod = 1; (* 0:B& W; 1:Color *)
huebegmod = 0.5;   (* 0.5:Blue; 0.2:Yellow *)
ampflagmod = 1;    (* 0:Normal below;
 1:db Below; 3:db Above; 4:Log Below *)
dbcutoffmod = 40;
```

■ Phase

```
colorstylepha = 0 ;
(* 0:B& W; 1:Color *)huebegpha = 0.0 ;
(* 0.5:Blue; 0.2:Yellow *)
ampflagpha = 0;    (* 0:Normal below;
 1:db Below; 3:db Above; 4:Log Below *)
dbcutoffpha = 40;
(* Files  Parameters  Modulus
   Phase  Signal  3 DModulus  3 DPhase  End *)
```

■ 3D

```
flag3dpha = "yes"; (* yes/no *)
flag3dmod = "yes"; (* yes/no *)
xstr = "Time (sec)";
ystr = "Signal";
```

BATCH PROCESSING takes "data file" and ".M file".

```
(* runcwt[x_]:=
  {noextnew=x,<<"g:/all-mma/cwt/Acwt-core_nb-m.m"}; *)
(* runcwt[x_]:={noextnew=x,
   Get[currentdir<>"Acwt-core_nb-m.m"]}; *)
runcwt[x_] := {noextnew = x,
  Get[currentdir <> "TFR-WV_Shell-Core.nb.m"]};
runcwt[datafilename];
(* Files  Parameters  Modulus
   Phase  Signal  3 DModulus  3 DPhase  End *)
```

```
This is RUN No : 2

Data file (Default-B) : c:/per/mat/0-cwt/data/Chirp-al
```

Sampling Rate : 200.

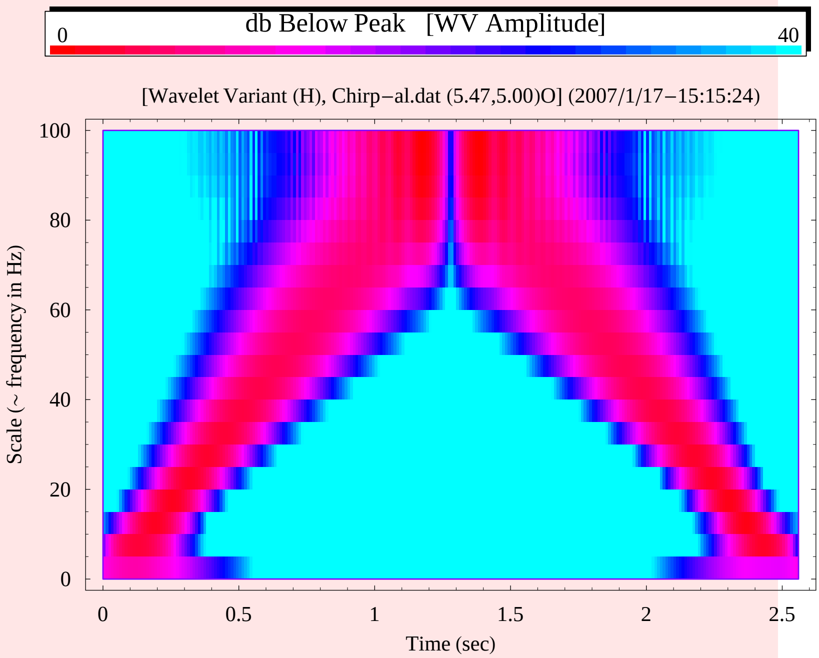
(mat_power, mat_inc, size) : {8, 2, {512}}

Dimension of WPamp : {256, 20}

Freq index range : {1, 20}

Time range : {0., 256}

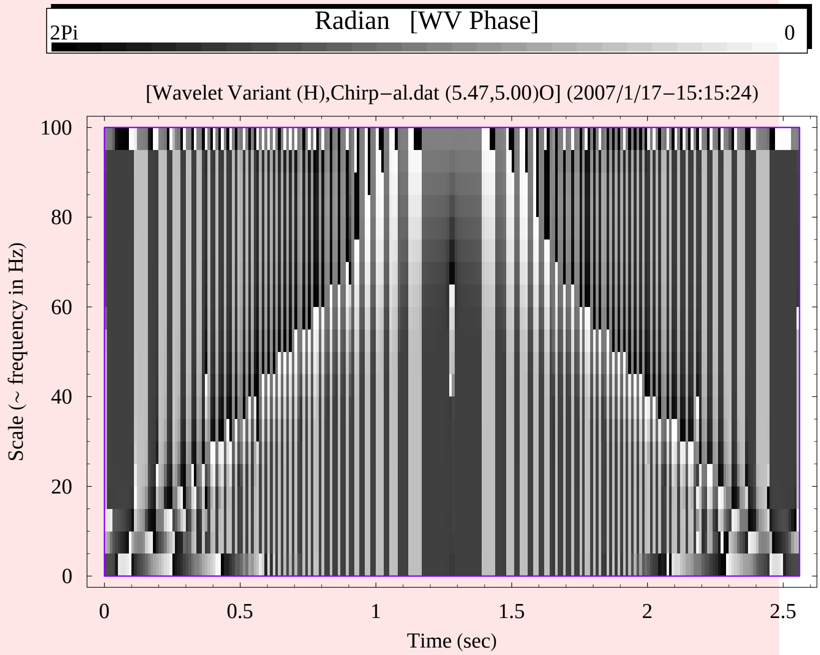
Beginning of Plots



- Graphics -

Display::pserr : PostScript language error:
Warn … Utopia-Regular for Times-Roman

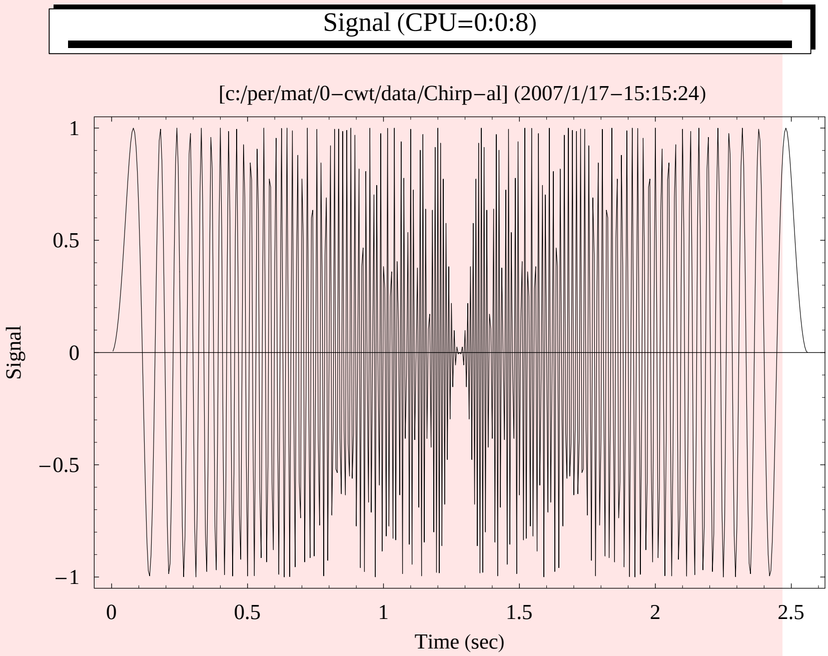
{2007, 1, 17, 15, 15, 23} CPU:(00, 00, 2.063);
Time:(00, 00, 05)



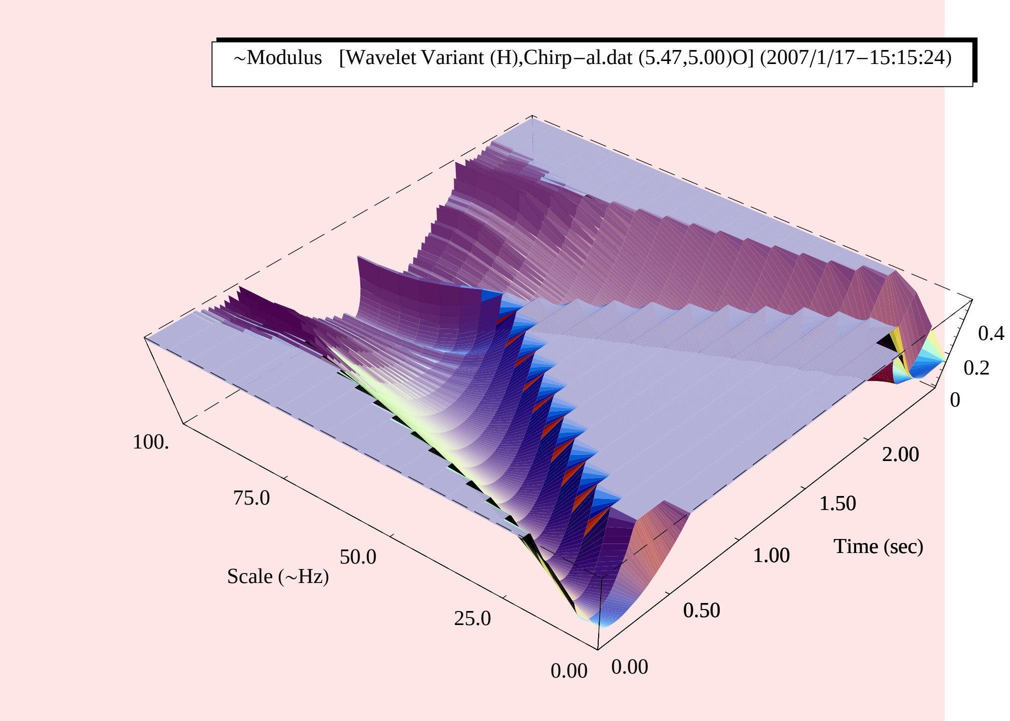

```
- Graphics -
```

```
Display::pserr : PostScript language error:
Warn … Utopia-Regular for Times-Roman
```

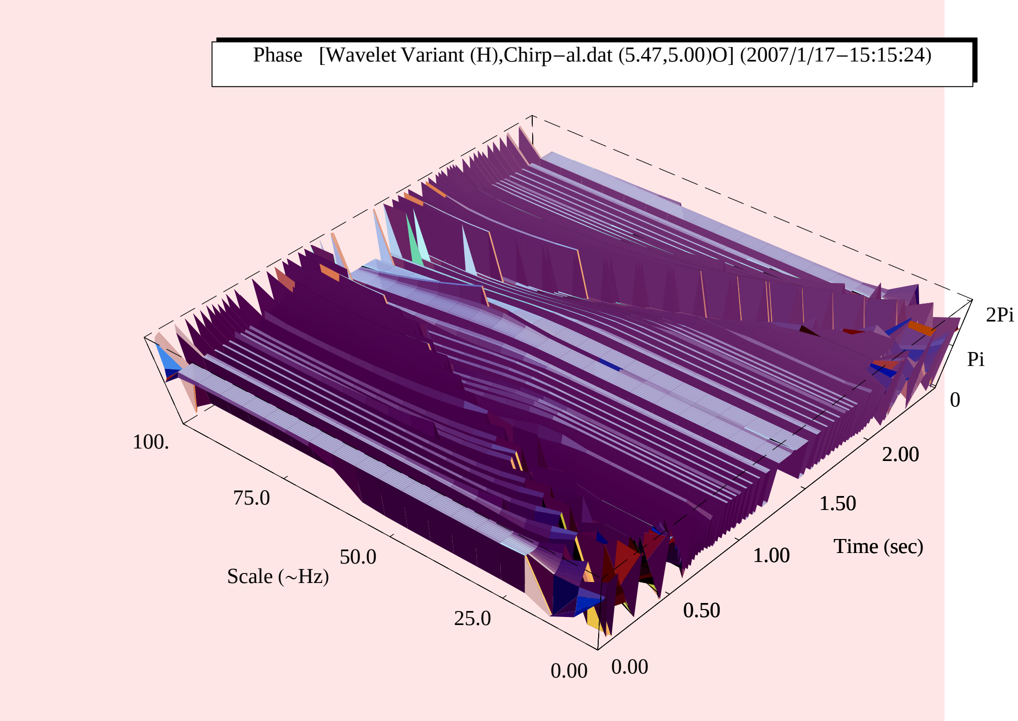
```
{2007, 1, 17, 15, 15, 23}     CPU:(00, 00, 2.594);
Time:(00, 00, 08)
```




```
- Graphics -

Display::pserr : PostScript language error:
Warn …Utopia-Regular for Times-Roman

General::stop :
 Further output of Display::pserr will
   be suppressed during this calculation.

 {2007, 1, 17, 15, 15, 23}      CPU:(00, 00, 2.656);
Time:(00, 00, 10)
```


~Modulus [Wavelet Variant (H),Chirp−al.dat (5.47,5.00)O] (2007/1/17−15:15:24)
0.4
0.2
0
100.
75.0
50.0
Scale (~Hz)
25.0
0.00
0.00
0.50
1.00
1.50
2.00
Time (sec)

```
{- Graphics -}
```



{- Graphics -}

{2007, 1, 17, 15, 15, 23} CPU:(00, 00, 2.922);
Time:(00, 00, 17)

No TeX Equation.

(* Files Parameters Modulus Phase Signal 3 DModulus 3 DPhase End *)

(* ---------------- Reserved ----
------------ *)

(* Wavelet Variant Transform and Adaptive CWT Renditions *)

(* by Ron Lee *)

(*
Notes on `Shell Processing` OR `Single-file Stand-alone Processing`:

This file is also the base of the core file for WVTR shell in ".nb" format `(View Shell File in NB Fmt)`. It needs to be converted to "text (.M)" format for use in WVTR shell `(View Shell's Core File in M Fmt)`.

A: For `Single-file Stand-alone Processing` :
Add ' noextbat="no" ' and optionally ' noext="datafilename" ' to cell labeled as 'Shell processing OR Single-file processing'
B: For `Shell Processing` :
Steps for Making the M-file:
0: Rename this file as the Shell's core file in nb fmt (e.g., WVTR_Shell-Core.nb).
1. Comment out all elements in the cell labeled as 'Shell Processing OR Single-file Processing
2. Delete all outputs
3. Remove In/Out names
4. Just "saved as special -> text ->, ?.m" (e.g., WVTR_Shell-Core.nb.m)
5. Put the M fmt file name in shell (e.g., WVTR_Shell.nb).
Done!
*)

```
(* DISPLAY SCHEME *)
  (* Hyperlinks:  Files   Parameters   Modulus

     Phase   Signal   3 DModulus   3 DPhase   End *)
```

```
commondir = "g:\\all-mma\\common\\";
currentdir = "g:\\all-mma\\TFR-WV\\";
datadir = "c:\\per\\mat\\0-CWT\\data\\";
FigDir = "G:\\Fig\\MMA\\TFR-WV\\";
Get[commondir <> "000-p1_nb-m.m"];
Get[commondir <> "Run-form-1w.m"];
Get[commondir <> "fig-fram_nb-m.m"];
Get[commondir <> "000-word_nb-m.m"];
If[ runcount > 0
  , { mouttn["This is RUN No : ", ++runcount],
    Remove["p*", "fun*", "*wp*", "gf*", "lgn*"] }
  , mouttn["No case shall ever go here.", 999]
  , { runcount = 1, mouttn["This is RUN No : ", runcount] }
 ];
Off[General::spell1]; Off[Generalspell];
SetOptions[ SelectedNotebook[], ShowPageBreaks -> True,
    WindowToolbars -> {"LinksBar" , "RulerBar", "EditBar"}];
<< "e:/math/4.0/addons/standardpackages/graphics/legend.m"
datadir = "c:/per/mat/0-cwt/data/";
currentfile = ReadList[datadir <> "cur-cwt.fip", String];
```

```
This is RUN No : 1
```

```
(*
noext="envelope";
noext="chirp-de";   (* inverse decreasing freq ~ 1/t *)
noext="chirp-ny";   (* Nyqust sampling *)
noext="q1w60202";
noext="chirp-n2";   (* Nyqust sampling -- variable  sai *)
noext="chirp-n1";   (* Nyqust sampling -- fixed     sai *)
noext="chirp-n3";
noext="chi-vari";
noext="chi-fixe";
noext="bif-fixe";
noext="bif-vari";
noext="test-cwt"
noext="test";
Clear["noext*"];
*)
```

```
noext=currentfile[[1]];
```

- (* Shell Processing OR Single-file Processing *)

```
(* (View Shell File in NB Fmt)  (View Shell's Core File in M Fmt)
*)
(*
(* Not Used in m file. And may not needed in nb file either. *)
noextbat="no";
noext="test";
noext="CWT-Demo";
*)
(* Hyperlinks:
  Files  Parameters  Modulus  Phase  Signal  3 DModulus  3 DPhase  End *)
noextbat="no";
```

```
Which[noextbat === "yes",
   Which[noextnew === "0", noext = currentfile[[1]],
         noextnew =!= "0", noext = noextnew ] ];
noext = datadir <> noext;
moutttt["Data file" <>
          Which[ noextbat =!= "yes",
     " (Default-A  or  Assigned) : ",
         noextbat === "yes",
                        Which[
      noextnew === "0", " (Default-B) : ",
                   noextnew =!= "0", " (Assigned) : "] ], noext];
```

Data file (Default-A or Assigned) : c:/per/mat/0-cwt/data/Chirp-al

```
(* Hyperlinks: Files Parameters Modulus
   Phase Signal 3 DModulus 3 DPhase End *)

(* ============== Parameters for Display Schemes
=================== *)
(* *)                       (*  "yes" in .m file *)
Which[noextbat=!="yes",
   {
   colorstylemod=0;    (* 0:B&W; 1:Color *)
   huebegmod=0.5;      (* 0.5:blue ; 0.2:yellow *)
   ampflagmod=1;       (* 0:normal below ; 1:DB below ; 2:db above 3:Log below*)
   dbcutoffmod=40;
   colorstylepha=0;    (* 0:B&W; 1:Color *)
   huebegpha=0.0;      (* 0.5:blue ; 0.2:yellow *)
   ampflagpha=0;       (* 0:normal below ; 1:DB below ; 2:db above 3:Log below*)
   dbcutoffpha=40;
   flag3dmod="yes";
   flag3dpha="yes";
   xstr="Time (sec)";
   ystr="Signal";
   } ];
(* *)
```

```
iinc=1 ;
jinc=1;logbase=10.;
secbeg=1;
colorstyle=colorstylemod;huebeg=huebegmod; (* initializing color
for modulus*)
ampflag=ampflagmod;dbcutoff=dbcutoffmod;
```

(* AUTO INPUT OF DETAILS *)

```
(* ===== Auto Input of Complete ASYST Program Details and Data =====
wpsig=ReadList[noext<>".sic", Number];
wpamp=ReadList[noext<>".amc", Number, RecordLists->True];
wppha=ReadList[noext<>".phc", Number, RecordLists->True];
wpsca=ReadList[noext<>".scc", Number, RecordLists->True];
wpspe=ReadList[noext<>".spc", Number];
wpstr=ReadList[noext<>".stc", String];
phaswitchflag=wpstr[[4]]; (* For ridge extraction *)
npower=wpspe[[1]];
jwplevel=wpspe[[2]];
bestflag=wpspe[[3]];
bestlevel=wpspe[[4]];
sfreq=wpspe[[5]];mouttn["Sampling Rate : ", sfreq];
matpower=wpspe[[6]];
matinc=wpspe[[7]];
wpsigx=Table[ x/sfreq, {x, 1, 2^npower, 1}] ;
(* wpsigx=Table[ x/sfreq, {x, 1, 2^npower, matinc}] ; *)
wpsigy=wpsig ;
ntmp=Dimensions[wpsig];
mouttn["(mat_power, mat_inc, size) : ", {matpower, matinc, ntmp}];
wpsig=Table[ {wpsigx[[x]], wpsigy[[x]]}, {x, 1, 2^npower, 1} ] ;
(* wpsig=Table[ {wpsigx[[x]], wpsigy[[x]]}, {x, 1, 2^matpower, matinc}
wlet=wpstr[[1]];
nowfile=wpstr[[2]];

FigName=FigDir<>nowfile<>"_"<>wlet<>"_"
```

```
Sampling Rate : 200.

(mat_power, mat_inc, size) : {8, 2, {512}}
```

```
G:\Fig\MMA\TFR-WV\Chirp-al.dat_Wavelet Variant (H)_
```

† (* RESET *)

```
ibeg=If[ secbeg===1,  1, 2^(secbeg-1)+1 ];    (* index *)
iend=2^matpower;
ntmp=Dimensions[ wpamp ];mouttn["Dimension of WPamp : ", ntmp];

jbeg=1;
jend=Dimensions[ wpamp ][[2]];mouttn["Freq index range : ", {jbeg,

(*
jbeg=1
jend=40
jinc=1
*)
(* Set image size
myfontsize=10;
setis={imagesizex=432,imagesizey=imagesizex/1.,myfontsize=10,
        aspectratiox=1.11,lgnposy=0.99};
*)
setis={imagesizex=360,imagesizey=imagesizex/1.4,myfontsize=8,
        aspectratiox=1.55,lgnposy=0.70};
myfont = "Times";
$TextStyle = {FontFamily -> "Times", FontSize -> myfontsize};
```

```
Dimension of WPamp : {256, 20}

Freq index range : {1, 20}
```

```
(* ------------------------------------------- *)

tbeg=0. ;     (* absolute range of plot area *)
tend=2^npower ;
tend=2^matpower;mouttn["Time range : ", {tbeg, tend}];
fbeg=Part[wpsca, 1, jbeg];
fend=Part[wpsca, 1, jend];
(* fbeg=fbeg- (fend-fbeg)/(Part[Dimensions[ wpsca[[1]] ], 1]-1);
fbeg=fbeg- (fend-fbeg)/(jend-jbeg);
saibeg=wpsca[[3,jbeg]];
saiend=wpsca[[3,jend]];
```

Time range : {0., 256}

† (* LEGENDS *)

```
(* beststr="   [CWT  Amplitude]"; (*   "   [CWT  Modulus]";  *) *)
Which[ phaswitchflag==="O", beststr="   [WV Amplitude]"
     , phaswitchflag==="N", beststr="   [ACWT Amplitude]"
     ];
tbegstr="0.00";
tmidstr=StringTake[ToString[
            N[ 2^npower/sfreq/2+0.000001, 20]],4] ;
tendstr=StringTake[ToString[
            N[ 2^npower/sfreq  +0.000001, 20]],4] ;
fbegstr=StringTake[ToString[
            N[ fbeg + 0.0010001, 20]],4];
fmidstr=StringTake[ToString[
            N[ fbeg+(fend-fbeg)/2. + 0.0010001, 20]], 4];
fendstr=StringTake[ToString[
            N[ fend + 0.001001, 20]],4];
datelist=Date[];
gfdate=
    "("<>ToString[datelist[[1]]]<>"/"<>ToString[datelist[[2]]]<>
    "/"<>ToString[datelist[[3]]]<>"-"<>ToString[datelist[[4]]]<>
    ":"<>ToString[datelist[[5]]]<>":"<>ToString[datelist[[6]]]<>")";
```

† (* FUNCTIONS *)

```
func1[fwpamp_]:=
      Which[ ampflag===0, Abs[fwpamp]
           , ampflag===1, Abs[fwpamp]
           , ampflag===2, Abs[fwpamp]
           , ampflag===3, Log[logbase, Abs[fwpamp]]
           ];
(* shift log scale to >= 0 in the plot range *)
func2[fwpamp_]:=
      Which[ ampflag===0, fwpamp
           , ampflag===1,
             ( Log[logbase,
                   Max[fwpamp[[ Range[ibeg, iend], Range[jbeg, jend]
                 - Log[logbase, fwpamp] ) * 20. (*small->larger*)
           , ampflag===2,
             ( Log[logbase, fwpamp]
             - Log[logbase,
                   Min[fwpamp[[ Range[ibeg, iend], Range[jbeg, jend]
             ) * 20.  (* small->small*)
           , ampflag===3,
             Abs[Min[fwpamp[[ Range[ibeg, iend], Range[jbeg, jend]
             + fwpamp
           ];
   (* ListPlot[Take[fwpamp, {ibeg, iend}], PlotJoined->True, Frame->True]
   *)

(* normalizing to 0~1 *)
func3[fwpamp_]:=
      Which[ ampflag===0, fwpamp /
                  Max[ fwpamp[[ Range[ibeg, iend], Range[jbeg, jend]
           , ampflag===1, ( dbcutoff - fwpamp ) / dbcutoff
           , ampflag===2, ( dbcutoff - fwpamp ) / dbcutoff
           , ampflag===3,
                  Max[ fwpamp[[ Range[ibeg, iend], Range[jbeg, jend]
           ];
(* trimming *)
func4[fwpamp_]:=  (* error -- not a function *)
(* champ=Which[ ampflag===0, fwpamp *)
      Which[ ampflag===0, fwpamp
           , ampflag===1, Do[Do[ If[ fwpamp[[x, y]]<0., fwpamp[[x,
                          , fwpamp[[x, y]] ], {x, ibeg, iend, 1}
                          , {y, jbeg, jend, 1} ]
           , ampflag===2, Do[Do[ If [fwpamp[[x, y]]<0., fwpamp[[x,
                          , fwpamp[[x, y]] ], {x, ibeg, iend, 1}
                          , {y, jbeg, jend, 1} ]
           , ampflag===3, fwpamp
           ];
(* proper reverse *)
```

```
func5[fwpamp_]:=
      Which[ ampflag===0, 1 - fwpamp   (* higher amp is darker(0)
           , ampflag===1, 1 - fwpamp
           , ampflag===2, fwpamp
           , ampflag===3, 1 - fwpamp
           ];
   (*Max[Take[fwpamp, {ibeg, iend}]]
     Min[Take[fwpamp, {ibeg, iend}]]
     ListPlot[Take[fwpamp, {ibeg, iend}], PlotJoined->True, Frame->True]
   *)
func6[fwpamp_]:=
      Which[ colorstyle===0, fwpamp
           , colorstyle===1, 1-fwpamp
           ];
(* scaling color *)
func7[fwpamp_]:=
      Which[ colorstyle===0, fwpamp
           , colorstyle===1, 1 - (1-fwpamp) (1-huebeg)
           ];
deltafre=fend-fbeg;
xymin=Table[{(i-1)*iinc*matinc/sfreq,
            ((j-1)-0)/(jend-(jbeg-1))*deltafre*jinc+fbeg},
            {i, (iend-(ibeg-1))/iinc},
            {j, (jend-(jbeg-1))/jinc}];
xymax=Table[{i    *iinc*matinc/sfreq,
           ((j  )-0)/(jend-(jbeg-1))*deltafre*jinc+fbeg},
            {i, (iend-(ibeg-1))/iinc},
            {j, (jend-(jbeg-1))/jinc}];
(*
xymin=Table[{(i-1)*iinc, ((j-1)-0)/(jend-(jbeg-1))*deltafre*jinc+fbeg},
            {i, (iend-(ibeg-1))/iinc}, {j, (jend-(jbeg-1))/jinc}];
xymax=Table[{i    *iinc, ((j  )-0)/(jend-(jbeg-1))*deltafre*jinc+fbeg},
            {i, (iend-(ibeg-1))/iinc}, {j, (jend-(jbeg-1))/jinc}];
*)
(* ------------------------------------------- *)
gffunc[x_, y_]:= Graphics[
    Which[
           colorstyle===1
      (* , {Hue[0, 1-wpamp[[x,y]], 1-wpamp[[x,y]]] *)
         , {Hue[wpamp[[(x-1)*iinc+ibeg, (y-1)*jinc+jbeg]]]
                    , Rectangle[xymin[[x,y]], xymax[[x,y]]]]}
         , colorstyle===0
         , {GrayLevel[wpamp[[(x-1)*iinc+ibeg, (y-1)*jinc+jbeg]]]
                    , Rectangle[xymin[[x,y]], xymax[[x,y]]]]}
         ]
    ]
gftable=Table[gffunc[x,y],
         {x,1,(iend-(ibeg-1))/iinc, 1}, {y,1,(jend-(jbeg-1))/jinc,1}];

(* ------------------------------------------- *)
gfaxis={
      Graphics[
      {
```

```
        Hue[0.75],
        Line[{{tbeg*matinc/sfreq, fbeg}, {tend*matinc/sfreq, fbeg},
            {tend*matinc/sfreq, fend}, {tbeg*matinc/sfreq, fend},
            {tbeg*matinc/sfreq, fbeg}}
        ]
      }
      ]
(*  , Graphics[ {GrayLevel[1.0], Rectangle[{tbeg,fbeg}, {tend,fend}]}
    };
```

† (* PLOT 1 *)

```
(*
wpamp=func1[wpamp];
wpamp=func2[wpamp];
wpamp=func3[wpamp];
*)
wpamp=func3[func2[func1[wpamp]]];

(* wpamp=func4[wpamp]; *)  (* error *)
(* func4[wpamp] *)   (* error *)
champ=Which[ ampflag===0, wpamp
           , ampflag===1, Do[Do[ If[ wpamp[[x, y]]<0., wpamp[[x,
                          , wpamp[[x, y]] ], {x, ibeg, iend, 1} ]
                          , {y, jbeg, jend, 1} ]
           , ampflag===2, Do[Do[ If [wpamp[[x, y]]<0., wpamp[[x,
                          , wpamp[[x, y]] ], {x, ibeg, iend, 1} ]
                          , {y, jbeg, jend, 1} ]
           , ampflag===3, wpamp
           ];
(*
wpamp=func5[wpamp];
wpamp=func6[wpamp];
wpamp=func7[wpamp];
*)
wpamp=func7[func6[func5[wpamp]]];

ampary=1-Transpose[wpamp];
gftable=Table[gffunc[x,y],
         {x,1,(iend-(ibeg-1))/iinc, 1}, {y,1,(jend-(jbeg-1))/jinc,1}];
lgnlbmod=Which[ ampflag===0, "Normalized Magnitude"
              , ampflag===1, "db Below Peak"
              , ampflag===2, "db Above Bottom"
              , ampflag===3, "Normalized Log Mag" (* poor resolution?!
              ]<>beststr;
(* lgnlbmod=FontForm[lgnlbmod, {myfont, myfontsize}];
lgnlbmod=StyleForm[lgnlbmod, FontFamily->"Times", FontSize->8 ];
lgnlbmod=StyleForm[lgnlbmod, FontFamily->myfont, FontSize->myfontsize+2
```

```
lgnupmod=Which[ ampflag===0, "1"
              , ampflag===1, "0"
              , ampflag===2, ToString[ dbcutoff ]
              , ampflag===3, "1"
              ];
(* lgnupmod=FontForm[lgnupmod, {myfont, myfontsize}] ---> error *)
lgndnmod=Which[ ampflag===0, "0"
              , ampflag===1, ToString[ dbcutoff ]
              , ampflag===2, "0"
              , ampflag===3, "0"
              ];

colorstyle=colorstylemod;huebeg=huebegmod;
gfshowmod=Show[   (* ShowLegend[ *)
    {
      gftable
    (* Graphics[ Raster[ Transpose[ wpamp ] ]]   *)
    , gfaxis
    }
    , AspectRatio->1/aspectratiox
(*
    , AspectRatio->1/1.8    (* 1.5    *)
    , AspectRatio->1/GoldenRatio    (* 1.5    *)
*)
    , Frame->True
    , FrameLabel->{"Time (sec)", "Scale (~ frequency in Hz)"}
    , PlotLabel->
    (*StyleForm["Modulus  "<>"["<>wlet<>", "<>nowfile<>" ("<> *)
      StyleForm[            "["<>wlet<>", "<>nowfile<>" ("<>
                StringTake[ToString[N[saibeg+ 0.0001, 10]],4]<>","<>
                StringTake[ToString[N[saiend+ 0.0001, 10]],4]<>")"<>
                phaswitchflag<>"] "<>gfdate,
                FontFamily->myfont, FontSize->myfontsize ]
    , DisplayFunction->Identity
    , TextStyle->{FontFamily->myfont, FontSize->myfontsize}
  (*~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    , Background->Hue[0.6]
    , PlotRegion->{{-0.1,1.1},{-0.1,1.1}}
    --------------------------------------------------------------------
  *)
   ];
```

† (* PLOT 2 *)

```
colorstyle=colorstylepha;huebeg=huebegpha;  (* initializing color
for phase*)
ampflag=ampflagpha;dbcutoff=dbcutoffpha;
(* colorstyle=0;  (* 0:B&W ; 1:Color*)
ampflag=0;     (* 0:normal below ; 1:DB below ; 2:db above 3:Log
```

```
below*)
dbcutoff=40;
huebeg=0.0;     (* 0.5:blue ; 0.2:yellow *)
huebeg=huebegpha;     (* 0.5:blue ; 0.2:yellow *) *)
```

```
wpamp=wppha;
wpamp=func3[func2[func1[wpamp]]];

champ=Which[ ampflag===0, wpamp
           , ampflag===1, Do[Do[ If[ wpamp[[x, y]]<0., wpamp[[x,
                          , wpamp[[x, y]] ], {x, ibeg, iend, 1} ]
                          , {y, jbeg, jend, 1} ]
           , ampflag===2, Do[Do[ If [wpamp[[x, y]]<0., wpamp[[x,
                          , wpamp[[x, y]] ], {x, ibeg, iend, 1} ]
                          , {y, jbeg, jend, 1} ]
           , ampflag===3, wpamp
           ];

wpamp=func7[func6[func5[wpamp]]];

(* phaary=Transpose[wpamp]; *)
phaary=Transpose[wppha];

gftable=Table[gffunc[x,y],
         {x,1,(iend-(ibeg-1))/iinc, 1}, {y,1,(jend-(jbeg-1))/jinc,1}];
(* beststr="   [CWT  Phase]"; *)
Which[ phaswitchflag==="O", beststr="   [WV Phase]"
     , phaswitchflag==="N", beststr="   [ACWT Phase]"
     ];
lgnlb=Which[ ampflag===0, "Radian"  (* "Normalized Magnitude" *)
           , ampflag===1, "db Below Peak"
           , ampflag===2, "db Above Bottom"
           , ampflag===3, "Normalized Log Mag" (* poor resolution?!
           ]<>beststr;
lgnlb=StyleForm[lgnlb, FontFamily->myfont, FontSize->myfontsize+2
lgnup=Which[ ampflag===0, "2Pi"   (* "1" *)
           , ampflag===1, "0"
           , ampflag===2, ToString[ dbcutoff ]
           , ampflag===3, "1"
           ];
lgndn=Which[ ampflag===0, "0"
           , ampflag===1, ToString[ dbcutoff ]
           , ampflag===2, "0"
           , ampflag===3, "0"
           ];

colorstyle=colorstylepha;huebeg=huebegpha;
gfshowpha=Show[   (* ShowLegend[ *)
    {
      gftable
    , gfaxis
    }
```

```
    , AspectRatio->1/aspectratiox
    , Frame->True
    , FrameLabel->{"Time (sec)", "Scale (~ frequency in Hz)"}
    , PlotLabel->
    (*StyleForm["Phase  "<>"["<>wlet<>","<>nowfile<>" ("<> *)
      StyleForm[           "["<>wlet<>","<>nowfile<>" ("<>
                 StringTake[ToString[N[saibeg+ 0.0001, 10]],4]<>","<>
                 StringTake[ToString[N[saiend+ 0.0001, 10]],4]<>")"<>
                 phaswitchflag<>"] "<>gfdate,
                 FontFamily->myfont, FontSize->myfontsize
               ]
    , DisplayFunction->Identity
  (*~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    , Frame->True
    , FrameLabel->{"Time", "Scale"}
    , AxesLabel->{"Time", "Scale"}
    , Background->Hue[0.6]
    , PlotRegion->{{-0.1,1.1},{-0.1,1.1}}
    ---------------------------------------------------------------------
  *)
   ];
```

† (* PLOT 3 *)

```
(* ---------------------------------------------- *)
(* ---------------------------------------------- *)
prule=Graphics[Line[{{0.0,0.0},{0.5,0.5}}]
    , DisplayFunction->Identity
    ];
```

† (* 3D PLOTS *)

```
xtdiv=4; ytdiv=4;
xtspa=Which[ matpower===6, 15
           , matpower===7, 25
           , matpower===8, 50
           , matpower===9, 100
           ];
(*
xttab=Table[
     {iend*x/xtdiv,
      StringTake[ToString[N[tend*matinc/sfreq*x/xtdiv+0.0001001,
      {x, 0, xtdiv, 1} ];
*)
xttab=Table[
     {If[ xtspa*x===0, 1, xtspa*x ],
      StringTake[ToString[N[tend*matinc/sfreq*xtspa*x/iend+0.000101,
      {x, 0, xtdiv, 1} ];
yttab=Table[
     {If[ jend*y/ytdiv===0, 1, jend*y/ytdiv ],
      StringTake[ToString[N[(fend-fbeg)*y/ytdiv+fbeg+0.000101, 6]],
      {y, 0, ytdiv, 1} ];
zttab={{0,"0"},{3.1416,"Pi"},{6.283,"2Pi"}};
```

```
(* $DefaultFont={aft,afs}; *)

colorstyle=colorstylemod;huebeg=huebegmod;
StylePrint["Beginning of Plots", "Graphics", CellTags->"Beginning

StylePrint[
pmod=   (* -----------------------------------------------------------
ShowLegend[
    gfshowmod
    , Which[
           colorstyle===1
      (* , {Hue[0, #, #]&, 10, lgnupmod, lgndnmod *)
         , {Hue[1-(#)(1-huebeg)]&, 30, lgnupmod, lgndnmod
           , LegendBorderSpace -> 0.2
           , LegendOrientation->Horizontal
           , LegendPosition->{-0.81, +lgnposy}
           , LegendSize->{1.7, 0.10}
           , LegendTextOffset->{0.0, -0.70}
           , LegendLabel->lgnlbmod
           , LegendLabelSpace->+2.65
           , LegendShadow->{0.01, 0.01}
           }
         , colorstyle===0
           , {GrayLevel[#]&, 30, lgnupmod, lgndnmod
```

```
              , LegendBorderSpace -> 0.2
              , LegendOrientation->Horizontal
              , LegendPosition->{-0.81, +lgnposy}
              , LegendSize->{1.7, 0.10}
              , LegendTextOffset->{0.0, -0.70}
              , LegendLabel->lgnlbmod
              , LegendLabelSpace->+2.65
              , LegendShadow->{0.01, 0.01}
              }
            ]
     , ImageSize->{imagesizex,imagesizey}
(*
     , TextStyle->{FontFamily->myfont, FontSize->myfontsize}
     , ImageSize->{432, 432 / 2 }
     , ImageSize->{432, 432 / GoldenRatio}
     , TextStyle->{FontFamily->myfont, FontSize->20}
     , TextStyle->{FontFamily->myfont, FontSize->myfontsize}
     , DefaultFont→{"Courier",20.}
              , LegendBorderSpace -> 0.2
              , LegendOrientation->Horizontal
              , LegendPosition->{-0.77, +0.75}
              , LegendSize->{1.5, 0.10}
              , LegendTextOffset->{0.0, -1.00}
              , LegendLabel->lgnlbmod
              , LegendLabelSpace->+2.65
              , LegendShadow->{0.01, 0.01}
*)
     (*   , DisplayFunction->Identity  *)
(*    ]; *)
 ], "Graphics", CellTags->"Modulus"];
Display[FigName<>"Modulus"<>".eps", Graphics[pmod], "EPS"]
Get[commondir<>"000-p2_nb-m.m"]

colorstyle=colorstylepha;huebeg=huebegpha;
StylePrint[
ppha=  (* -------------------------------------------------------------
ShowLegend[
    gfshowpha
    , Which[
          colorstyle===1
      (* , {Hue[0, #, #]&, 10, lgnup, lgndn *)
         , {Hue[1-(#)(1-huebeg)]&, 30, lgnup, lgndn
              , LegendBorderSpace -> 0.2
              , LegendOrientation->Horizontal
              , LegendPosition->{-0.81, +lgnposy}
              , LegendSize->{1.7, 0.10}
              , LegendTextOffset->{0.0, -0.70}
              , LegendLabel->lgnlb
              , LegendLabelSpace->+2.65
              , LegendShadow->{0.01, 0.01}
              }
         , colorstyle===0
              , {GrayLevel[#]&, 30, lgnup, lgndn
```

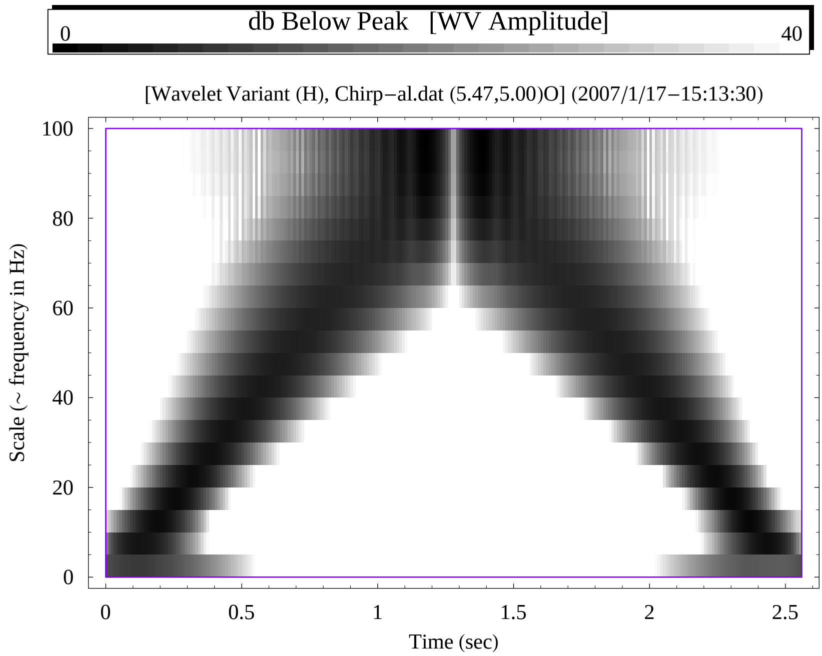
```
            , LegendBorderSpace -> 0.2
            , LegendOrientation->Horizontal
            , LegendPosition->{-0.81, +lgnposy}
            , LegendSize->{1.7, 0.10}
            , LegendTextOffset->{0.0, -0.70}
            , LegendLabel->lgnlb
            , LegendLabelSpace->+2.65
            , LegendShadow->{0.01, 0.01}
            }
          ]
    , ImageSize->{imagesizex,imagesizey}
    (*   , DisplayFunction->Identity  *)
(*    ]; *)
 ], "Graphics", CellTags->"Phase"];
Display[FigName<>"Phase"<>".eps", Graphics[ppha], "EPS"]
Get[commondir<>"000-p2_nb-m.m"]


StylePrint[
psig=  (* -------------------------------------------------------------
ShowLegend[
    ListPlot[ wpsig
    , PlotJoined->True
    , PlotStyle->{Thickness[0.0008] (* , Hue[0.0] *) }
    , Frame->True
    , FrameLabel->
      If[ StringMatchQ[StringTake[noext, -1 ], "4"
          ] || StringMatchQ[StringTake[noext, -1 ], "2"]
        , { "Time (sec)",
            If[ StringMatchQ[StringTake[noext, -1 ], "4"]
              , "Displacement (cm)"
              , "Acqueous flow (cm/sec)" ]
          }
        , {xstr, ystr}  (*   {"X value",   " Y value"}  *)
      ]
    , AspectRatio->1/aspectratiox / (1+0.0)
    , PlotLabel->
      StyleForm[" "<>"["<>noext<>
                 "] "<>gfdate,  FontFamily->myfont, FontSize->myfontsize
               ]
    , DisplayFunction->Identity
    ]
    ,
              {GrayLevel[0]&, 30, " ",  " "
            , LegendBorderSpace -> 0.8
            , LegendOrientation->Horizontal
            , LegendPosition->{-0.81, +lgnposy}
            , LegendSize->{1.7, 0.10}
            , LegendTextOffset->{0.0, -0.70}
(*
            , LegendPosition->{-0.77, +0.75}
            , LegendSize->{1.5, 0.10}
            , LegendTextOffset->{0.0, -1.00}
```

```
*)
           , LegendLabel->StyleForm["Signal"<>" (CPU="<>
             ToString[hrd]<>":"<>ToString[mind]<>":"<>
             ToString[secd]<>")",  FontFamily->myfont, FontSize->myfonts
             ]

           , LegendLabelSpace->+2.65
           , LegendShadow->{0.01, 0.01}
           }
    , ImageSize->{imagesizex,imagesizey}
(*    ]; *)
 ], "Graphics", CellTags->"Signal"];
Display[FigName<>"Signal"<>".eps", Graphics[psig], "EPS"]

Get[commondir<>"000-p2_nb-m.m"]

StylePrint[
If[flag3dmod=!="no", {
pmod3d=  (* ---------------------------------------------------------------
ShowLegend[
     ListPlot3D[
         ampary
       , Mesh->False
       , BoxStyle->{ Dashing[{0.02, 0.01}]
                   , Thickness[0.0008]
                   }
       , AxesLabel->{ "Time (sec)", "Scale (~Hz)", " " }
       , AxesEdge->{ Automatic, Automatic, Automatic }
       , Ticks->{ xttab
                , yttab
                , Automatic
                }
       , PlotRange->All
(*       , AspectRatio->1/1.406 / (1+0.0)    *)
       , AspectRatio->1/aspectratiox / (1+0.0)
    (* , PlotLabel->StyleForm["~Modulus   "<>"["<>noext<>"] "<>gfdate,
                            FontFamily->myfont, FontSize->myfontsize+2
       , DisplayFunction->Identity
       , ViewPoint->{-1.469,-1.244,2.783}
       ]
       , {
         GrayLevel[1]&, 30, " ",  " "
         , LegendBorderSpace -> .8
         , LegendOrientation->Horizontal
         , LegendPosition->{-0.81, +lgnposy}
         , LegendSize->{1.7, 0.10}
         , LegendTextOffset->{0.0, -0.70}
(*
         , LegendPosition->{-0.87, +0.70}
         , LegendSize->{1.5, 0.1}
         , LegendTextOffset->{0.0, -1.00}
*)
         , LegendLabel->StyleForm["~Modulus   "<>"["<>wlet<>","<>
```

```
                nowfile<>" ("<>
                StringTake[ToString[N[saibeg+ 0.0001, 10]],4]<>","<>
                StringTake[ToString[N[saiend+ 0.0001, 10]],4]<>")"<>
                phaswitchflag<>"] "<>gfdate,
                FontFamily->myfont, FontSize->myfontsize ]
              , LegendLabelSpace->+2.5
              , LegendShadow->{0.01, 0.01}
              }
         , ImageSize->{imagesizex,imagesizey}
         (* , DisplayFunction->Identity  *)
         ] (* ; *)
         (*  ListPlot3D[1-wpamp, Mesh->False,ViewPoint->{-1.689,1.484,2.416}]
(* } ]; *)
} ], "Graphics", CellTags->"3DModulus"];
If[flag3dmod=!="no", {
   Display[FigName<>"Modulus-3D"<>".eps", Graphics[pmod3d], "EPS"]
  } ];

StylePrint[
If[flag3dpha=!="no", {
ppha3d=  (* -------------------------------------------------------------------
ShowLegend[
     ListPlot3D[
         phaary
       , Mesh->False
       , BoxStyle->{ Dashing[{0.02, 0.01}]
                   , Thickness[0.0008]
                   }
       , AxesLabel->{ "Time (sec)", "Scale (~Hz)", " " }
       , AxesEdge->{ Automatic, Automatic, Automatic }
       , Ticks->{ xttab
                , yttab
                , zttab (*  Automatic  *)
                }
(*       , AspectRatio->1/1.406 / (1+0.0)    *)
       , AspectRatio->1/aspectratiox / (1+0.0)
    (* , PlotLabel->StyleForm[" "<>"["<>noext<>"] "<>gfdate,
                              FontFamily->myfont, FontSize->myfontsize
       , ViewPoint->{-1.469,-1.244,2.783}
       , PlotRange->All
       , DisplayFunction->Identity
       ]
       , {
           GrayLevel[1]&, 30, " ",  " "
         , LegendBorderSpace -> .8
         , LegendOrientation->Horizontal
         , LegendPosition->{-0.81, +lgnposy}
         , LegendSize->{1.7, 0.10}
         , LegendTextOffset->{0.0, -0.70}
(*
         , LegendPosition->{-0.87, +0.70}
         , LegendSize->{1.5, 0.1}
         , LegendTextOffset->{0.0, -1.00}
```

```
*)
          , LegendLabel->StyleForm["Phase   "<>"["<>wlet<>","<>
            nowfile<>" ("<>
            StringTake[ToString[N[saibeg+ 0.0001, 10]],4]<>","<>
            StringTake[ToString[N[saiend+ 0.0001, 10]],4]<>")"<>
            phaswitchflag<>"] "<>gfdate,
            FontFamily->myfont, FontSize->myfontsize ]
          , LegendShadow->{0.01, 0.01}
          }
     , ImageSize->{imagesizex,imagesizey}
       (* , DisplayFunction->Identity  *)
     ] (* ; *)
(* } ]; *)
} ], "Graphics", CellTags->"3DPhase"];
(* Run["mmawav.bat"]; *)
If[flag3dpha=!="no", {
   Display[FigName<>"Phase-3D"<>".eps", Graphics[ppha3d], "EPS"]
  } ];
Get[commondir<>"000-p2_nb-m.m"]

Get[commondir<>"000-p3.m"]
On[General::spell1];On[General::spell];
(* ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
(* Hyperlinks: Files  Parameters  Modulus  Phase  Signal  3 DModulus
```

Beginning of Plots

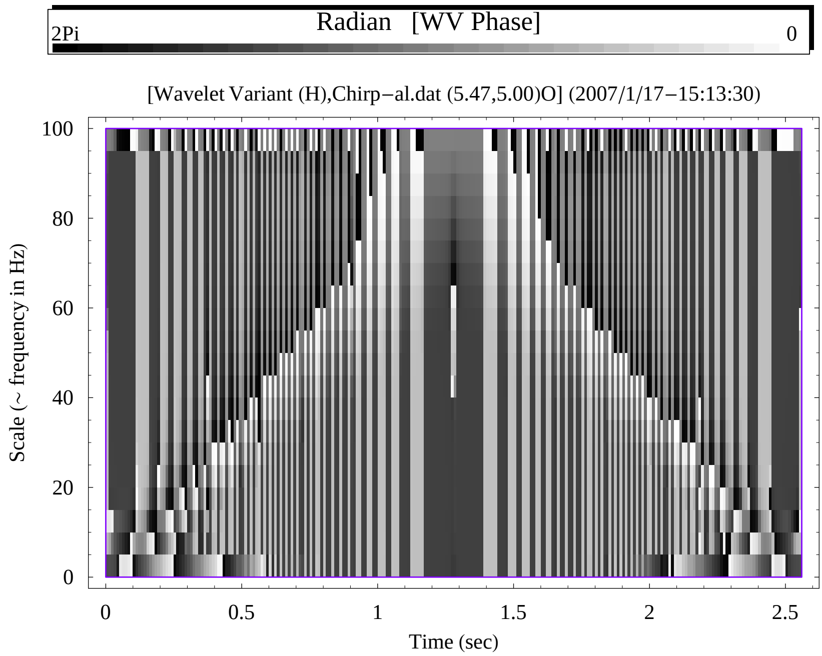


- Graphics -

```
Display::pserr : PostScript language error:
Warning…ont Utopia-Regular for Times-Roman
```

- Graphics -

{2007, 1, 17, 15, 13, 29}   CPU:(00, 00, 2.001);
Time:(00, 00, 05)



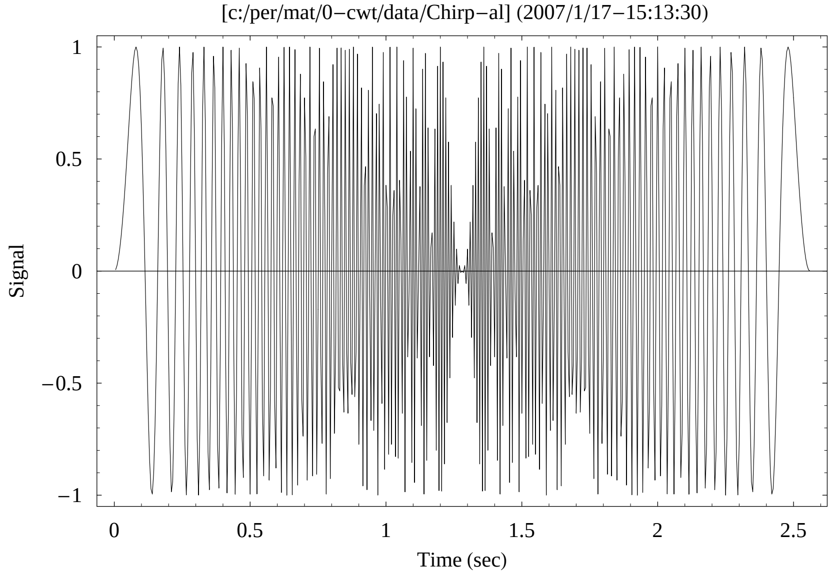
- Graphics -

```
Display::pserr : PostScript language error:
Warning…ont Utopia-Regular for Times-Roman
```

- Graphics -

{2007, 1, 17, 15, 13, 29} CPU:(00, 00, 2.345);
Time:(00, 00, 08)

Signal (CPU=0:0:8)



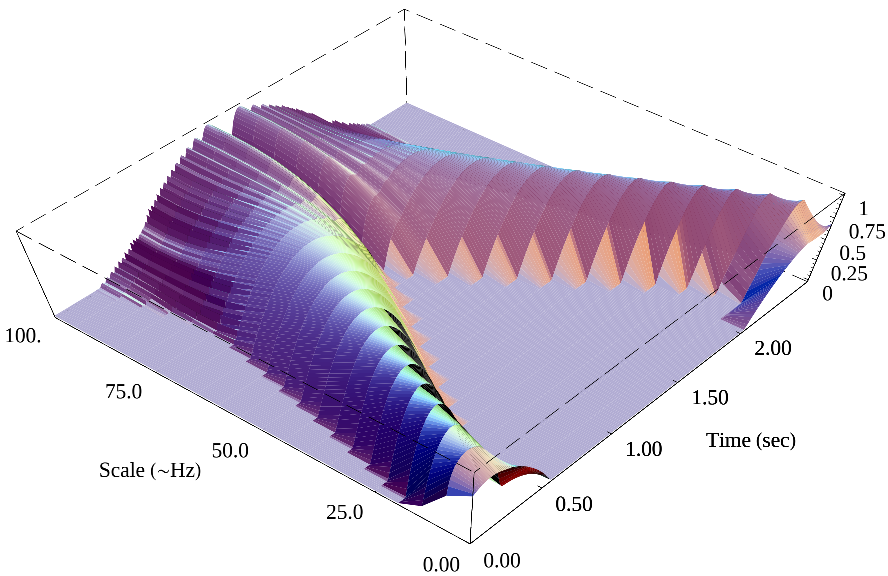
- Graphics -

```
Display::pserr : PostScript language error:
Warning…ont Utopia-Regular for Times-Roman
```

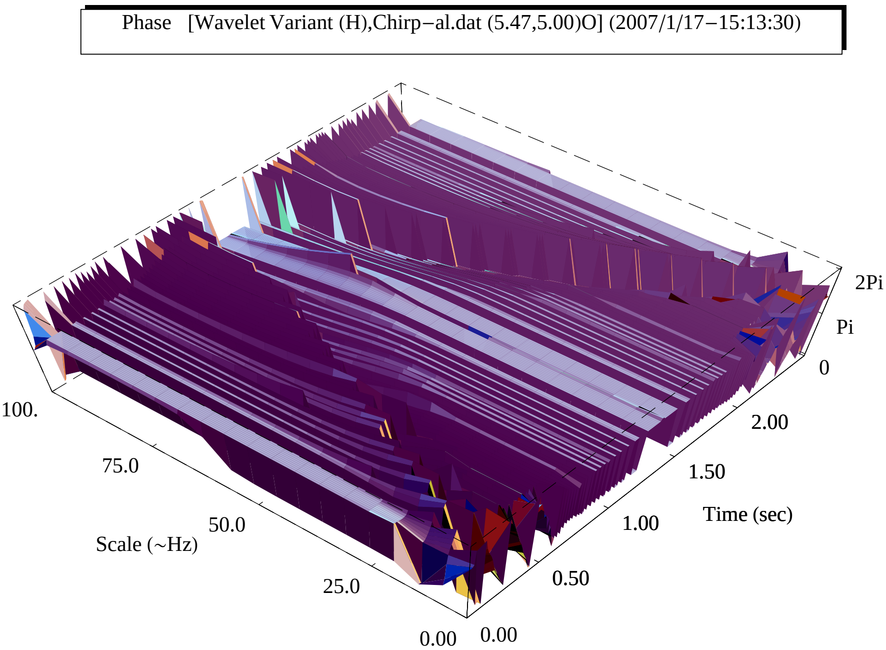
- Graphics -

{2007, 1, 17, 15, 13, 29}   CPU:(00, 00, 2.391);
Time:(00, 00, 10)

~Modulus [Wavelet Variant (H),Chirp−al.dat (5.47,5.00)O] (2007/1/17−15:13:30)




```
{- Graphics -}

Display::pserr : PostScript language error:
Warning…ont Utopia-Regular for Times-Roman
```




```
{- Graphics -}

        Display::pserr : PostScript language error:
        Warning…ont Utopia-Regular for Times-Roman

           {2007, 1, 17, 15, 13, 29}     CPU:(00, 00, 2.72);
Time:(00, 00, 16)

        No TeX Equation.

    (* Hyperlinks: Files  Parameters  Modulus  Phase  Signal  3 DModulus

 (* Hyperlinks:
   Files  Parameters  Modulus  Phase  Signal  3 DModulus  3 DPhase  End  *)
```

(* A Quasi Wavelet Basis Function for Refined Ridge Extraction*)

(* by Ron Lee *)

ü Files and Parameters

File Macro LeakOut LeakIn PhaseNoise Root Others End

```
<< "c:/lee/mat/Initial_SetNotebook.M" (* Set
  notebook options *)
<< "c:/lee/mat/Initial_Forms.M";
<< "c:/lee/mat/Initial_Frames.M";
<< "c:/lee/mat/Initial_Words.M";
FigDir =
  "G:\\Fig\\MMA\\WaveletVariantCharacterizations\\";
FigName = FigDir;
myfont = "Times";
$TextStyle =
  {FontFamily -> "Times", FontSize -> myfontsize};
timeflag = "Y";
dynamictimeflag = "N";
flabelflag = "Y";
flabelflag2 = "Y";
abcdflag = "N";
llabelflag = "y";
gridlineflag = "YY";
xyaxisflag = "LL";
stringposflag = 1 ;
xshiftflag = 0;
forcedstringaryflag = "n";
forcedflabelflag = "n" ;
stringindexmul  = 3;
stringindexsin  = 1;
abcdlabel  = "a";
timeflag = "Y"; tlabel = tlabelv;
```

ü Macros

File Macro LeakOut LeakIn PhaseNoise Root Others End

```
doshowxy := {
  moutt[flabel];
  moutt[flabel2];
  myplot=ListPlot[ dataxy
    , PlotJoined->True
    , PlotRange->All
    , PlotStyle->{Thickness[0.0008] (* , Hue[0.0] *) }
    , Frame->True
    , DisplayFunction->Identity
  ];
  (* myshow; *)
   Display[FigName<>EpsStrTmp<>".eps", Graphics[myshow],
"EPS"];
  (* Run["mmawav.bat"]; *)
  <<"c:/lee/mat/000-p2_nb-m.m"
};
```

```
flabel2 := typestr<>"α="<>StringTake[ToString[N[scap1,
9]],If[StringLength[ToString[N[scap1, 9]]]>=7, 7, 1] ]<>",
β="<>ToString[N[
peakshiftp1,2]]<>"π, ξ="<>StringTake[
ToString[N[xi,2]],1]<>",
(0↔"<>StringTake[ToString[N[xlimitp1, 2]],1]<>"π),
θ="<>ToString[N[phap1,3]]<>"π";
```

ü Leakage Out

File Macro LeakOut LeakIn PhaseNoise Root Others End

ü Leakage out (Wavelet variant)

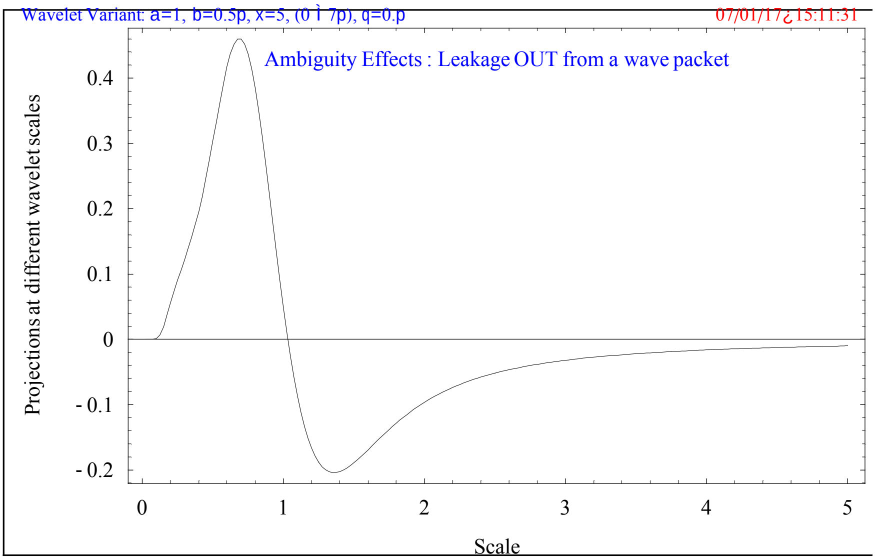
```
(* -------- Frequency Leakage Out (Variant)
--------- *)
typestr="Wavelet Variant: ";
xlabel="Scale";
ylabel="Projections at different wavelet scales";
flabel="Ambiguity Effects : Leakage OUT from a wave
packet";
<<"c:/lee/mat/000-p1_nb-m.m";
frelkgout [peakshiftv_, scav_, xlimitv_, phav_, xiv_] :=
  1 / (scav) * NIntegrate[Cos[xiv*x]*Sin[xiv*x/scav]*
     Exp[-((x - peakshiftv/xiv)^2/scav^2 + (x)^2)/(2)],
     {x, 0, xlimitv}
  , MinRecursion->3, MaxRecursion->10
];
  (*  1 / scav * NIntegrate[Cos[x]*Sin[xiv * x/scav]*
(* a=sca/xi *)
      Exp[-((x/scav - peakshiftv/xiv )^2 + (x)^2)/(2)],
  *)
peakshiftp1=0.5;peakshift= peakshiftp1 * Pi;
xlimitp1=7;xlimit=xlimitp1 * Pi;
phap1=0;pha=phap1 * Pi;
xi=5;
scap1=1;
datax= Table[ ni , {ni, 0.025, 5, 0.025}];
datay= Table[ frelkgout [peakshift, sca, xlimit, pha,
xi], {sca, 0.025 ,
5 , 0.025 } ];
dataxy=Table[ {datax[[ i ]], datay[[i]]}, {i,1
,Length[datax]}];
EpsStrTmp="WV_LO";
doshowxy;
```

```
General::spell1 :
 Possible spelling error: new symbol name "peakshift" is
   similar to existing symbol "peakshiftv".

General::spell1 : Possible spelling error: new symbol
   name "xlimit" is similar to existing symbol "xlimitv".

General::spell1 : Possible spelling error: new symbol
   name "datax" is similar to existing symbol "dataxy".

General::spell :
 Possible spelling error: new symbol name "datay" is
   similar to existing symbols {datax, dataxy}.
```

```
NIntegrate::slwcon :
 Numerical integration converging too slowly; suspect one
   of the following: singularity, value of the integration
   being 0, oscillatory integrand, or insufficient
   WorkingPrecision. If your integrand is oscillatory
   try using the option Method->Oscillatory in NIntegrate.
```

Ambiguity Effects : Leakage OUT from a wave packet

Wavelet Variant: α=1, β=0.5π, ξ=5, (0$\leftrightarrow$7π), θ=0.π




```
Display::pserr :  PostScript language error:
Warning…ont Utopia-Regular for Times-Roman
```

```
        {2007, 1, 17, 15, 11, 31}    CPU:(00, 00, 1.578);
Time:(00, 00, 06)
```

ü Leakage out (Morlet)

```
(* -------- Frequency Leakage Out (Morlet)
---------- *)
typestr="Morlet Wavelet: ";
xlabel="Scale";
ylabel="Projections at different wavelet scales";
flabel="Ambiguity Effects : Leakage OUT from a wave
packet";
<<"c:/lee/mat/000-p1_nb-m.m";
(*
frelkginM[peakshiftv_, scav_, xlimitv_, phav_, xiv_] :=
   NIntegrate[Cos[u/scav]*Cos[(u)]*
     Exp[-((u - peakshiftv)^2+ (u/scav)^2)/(2*xiv^2)],
     {u, 0, xlimitv}
     , MinRecursion->3, MaxRecursion->10
];
*)
(*  There exists analytical form  *)
frelkgoutM [peakshiftv_, scav_, xlimitv_, phav_, xiv_] :=
  2*  1 / (scav) * Integrate[Cos[xiv*x]*Cos[xiv*x/scav]*
     Exp[-((x - peakshiftv/xiv)^2/scav^2 + (x)^2)/(2)],
     {x, 0, Infinity}
];
peakshiftp1=0;  peakshift= peakshiftp1 * Pi;
xlimitp1=7;  xlimit=xlimitp1 * Pi;
phap1=0;  pha=phap1 * Pi;
xi=5;
scap1=1;
dofrelkgoutM=frelkgoutM [peakshift, sca, xlimit, pha,
xi](*analytical form*)

datax= Table[ ni , {ni, 0.000, 5, 0.025}];
datay= Table[ N[dofrelkgoutM], {sca, 0.000 , 5 ,
0.025 } ];
dataxy=Table[ {datax[[ i ]], datay[[i]]}, {i,1
,Length[datax]}];
EpsStrTmp="MW_LO";
doshowxy;
```

```
General::spell1 :
 Possible spelling error: new symbol name "frelkgoutM" is
   similar to existing symbol "frelkgout".
```

$$\frac{1}{\text{sca}}\left(2\;\text{If}\left[\text{Im}\left[\frac{1}{\text{sca}}\right]==0\;\&\&\;\text{Re}\left[\frac{1}{\text{sca}^2}\right]>-1,\;\frac{\left(e^{\frac{25\,(-1+\text{sca})^2}{2\,(1+\text{sca}^2)}}+e^{\frac{25\,(1+\text{sca})^2}{2\,(1+\text{sca}^2)}}\right)\sqrt{\frac{\pi}{2}}}{2\,e^{25}\sqrt{1+\frac{1}{\text{sca}^2}}},\right.\right.$$

$$\left.\left.\int_0^{\infty} e^{\frac{1}{2}\left(-x^2-\frac{x^2}{\text{sca}^2}\right)}\,\text{Cos}[5\,x]\;\text{Cos}\left[\frac{5\,x}{\text{sca}}\right]\,dx\right]\right)$$

Power::infy : Infinite expression $\frac{1}{0.}$ encountered.

Power::infy : Infinite expression $\frac{1}{0.}$ encountered.

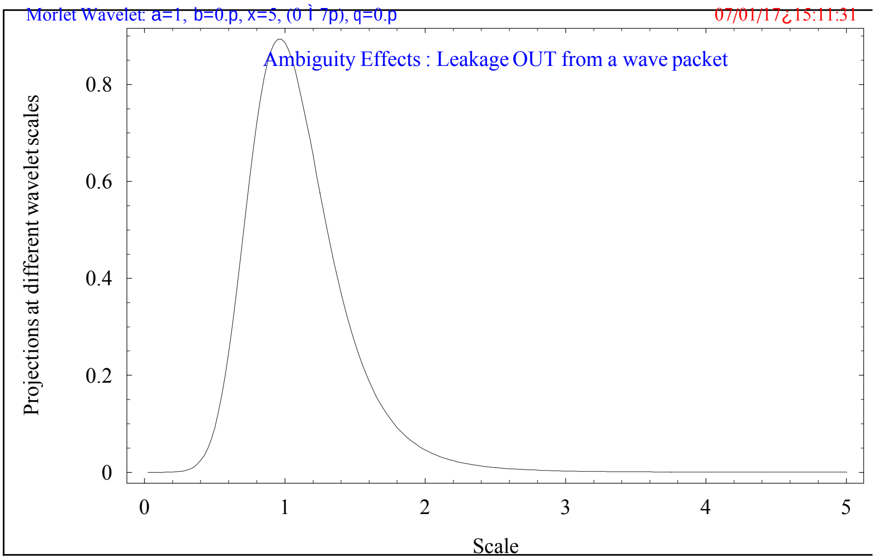
Power::infy : Infinite expression $\frac{1}{0.^2}$ encountered.

General::stop : Further output of
Power::infy will be suppressed during this calculation.

Ambiguity Effects : Leakage OUT from a wave packet

Morlet Wavelet: α=1, β=0.π, ξ=5, (0↔7π), θ=0.π

Graphics::gptn : Coordinate ComplexInfinity in
{0., ComplexInfinity} is not a floating-point number.




```
Graphics::gptn : Coordinate ComplexInfinity in
   {0., ComplexInfinity} is not a floating-point number.
```

```
Display::pserr : PostScript language error:
Warning…ont Utopia-Regular for Times-Roman
```

```
{2007, 1, 17, 15, 11, 38}   CPU:(00, 00, 0.5);
Time:(00, 00, 09)
```

■ `<< "c:\\lee\\mat\\000-p1.m";`

FindMinimum[

$$-1*1\,/\,sca*\left(\left(E^{-\frac{25\,(-1+sca)^2}{2\,(1+sca^2)}}+E^{-\frac{25\,(1+sca)^2}{2\,(1+sca^2)}}\right)\sqrt{\frac{\pi}{2}}\,\sqrt{1+\frac{1}{sca^2}}\;sca^2\right)\Big/$$

$(2\,(1+sca^2))$, {sca, 0.97}]

`<< "c:/lee/mat/000-p2_nb-m.m"`

{-0.447397, {sca → 0.962224}}

{2007, 1, 17, 15, 11, 48} CPU:(00, 00,

-16

4.44089 10); Time:(00, 00, 00)

Leakage In

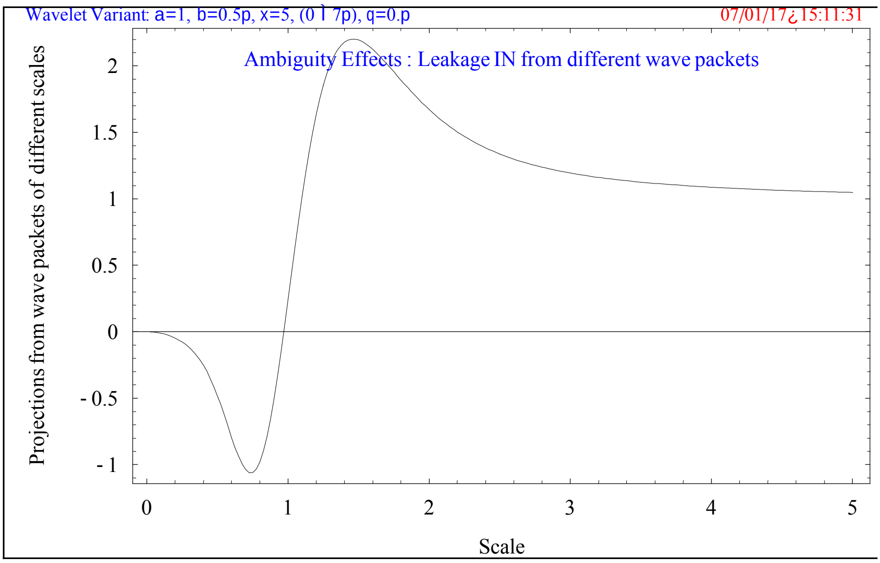
File Macro LeakOut LeakIn PhaseNoise Root Others End

Leakage in (Wavelet variant)

```
(* -------- Frequency Leakage In (Variant)
-------- *)
typestr="Wavelet Variant: ";
xlabel="Scale";
ylabel="Projections from wave packets of different
scales";
flabel="Ambiguity Effects : Leakage IN from different
wave packets";
<<"c:/lee/mat/000-p1_nb-m.m";
frelkgin[peakshiftv_, scav_, xlimitv_, phav_, xiv_] :=
   NIntegrate[Cos[u/scav]*Sin[(u)]*
     Exp[-((u - peakshiftv)^2+ (u/scav)^2)/(2*xiv^2)],
     {u, 0, xlimitv}
     , MinRecursion->3, MaxRecursion->10
];
(* gsca[b_, a_, s_] := NIntegrate[Cos[u/a]*Sin[(u)]*
                  Exp[-((u-Pi/2.)^2.+(u/a)^2.)/(2*s^2)],
{u, 0, b} ] *)
peakshiftp1=0.5;peakshift= peakshiftp1 * Pi;
xlimitp1=7;xlimit=xlimitp1 * Pi;
phap1=0;pha=phap1 * Pi;
xi=5;
scap1=1;
datax= Table[ ni , {ni, 0.025, 5, 0.025}];
datay= Table[ frelkgin [peakshift, sca, xlimit, pha, xi],
{sca, 0.025 ,
5 , 0.025 } ];
dataxy=Table[ {datax[[ i ]], datay[[i]]}, {i,1
,Length[datax]}];
EpsStrTmp="WV_LI";
doshowxy;
```

Ambiguity Effects : Leakage IN from different wave
packets

Wavelet Variant: α=1, β=0.5π, ξ=5, (0↔7π), θ=0.π




```
Display::pserr : PostScript language error:
Warning…ont Utopia-Regular for Times-Roman
```

```
        {2007, 1, 17, 15, 11, 48}    CPU:(00, 00, 0.468);
Time:(00, 00, 01)
```

ü Leakage in (Morlet)

```
(* --------- Frequency Leakage In (Morlet) --------- *)
typestr="Morlet Wavelet: ";
xlabel="Scale";
ylabel="Projections from wave packets of different scales";
flabel="Ambiguity Effects : Leakage IN from different wave packets";
<<"c:/lee/mat/000-p1_nb-m.m";
(*  There exists closed analytical form  *)
frelkginM[peakshiftv_, scav_, xlimitv_, phav_, xiv_] :=
   2 * Integrate[Cos[u/scav]*Cos[(u)]*
     Exp[-((u - peakshiftv)^2+ (u/scav)^2)/(2*xiv^2)],
     {u, 0, Infinity}
];
peakshiftp1=0;  peakshift= peakshiftp1 * Pi;
xlimitp1=7;  xlimit=xlimitp1 * Pi;
phap1=0;  pha=phap1 * Pi;
xi=5;
scap1=1;
dofrelkginM=frelkginM [peakshift, sca, xlimit, pha, xi](*analytical form*)

datax= Table[ ni , {ni, 0.000, 5, 0.025}];
datay= Table[ N[dofrelkginM], {sca, 0.000 , 5 , 0.025 } ];
dataxy=Table[ {datax[[ i ]], datay[[i]]}, {i,1 ,Length[datax]}];
EpsStrTmp="MW_LI";
doshowxy;
```

```
General::spell1 :
 Possible spelling error: new symbol name "frelkginM" is
   similar to existing symbol "frelkgin".
```

$$2\,\text{If}\Big[\text{Im}\Big[\frac{1}{\text{sca}}\Big] == 0 \;\&\&\; \text{Re}\Big[\frac{1}{\text{sca}^2}\Big] > -1,\ \frac{5\left(e^{\frac{25\,(-1+\text{sca})^2}{2\,(1+\text{sca}^2)}} + e^{\frac{25\,(1+\text{sca})^2}{2\,(1+\text{sca}^2)}}\right)\sqrt{\frac{\pi}{2}}}{2\,e^{25}\sqrt{1+\frac{1}{\text{sca}^2}}},$$
$$\int_0^{\infty} e^{\frac{1}{50}\left(-u^2-\frac{u^2}{\text{sca}^2}\right)}\,\text{Cos}[u]\,\text{Cos}\Big[\frac{u}{\text{sca}}\Big]\,du\Big]$$

Power::infy : Infinite expression $\frac{1}{0.}$ encountered.

Power::infy : Infinite expression $\frac{1}{0.^2}$ encountered.

Power::infy : Infinite expression $\frac{1}{0.^2}$ encountered.

General::stop : Further output of Power::infy will be suppressed during this calculation.

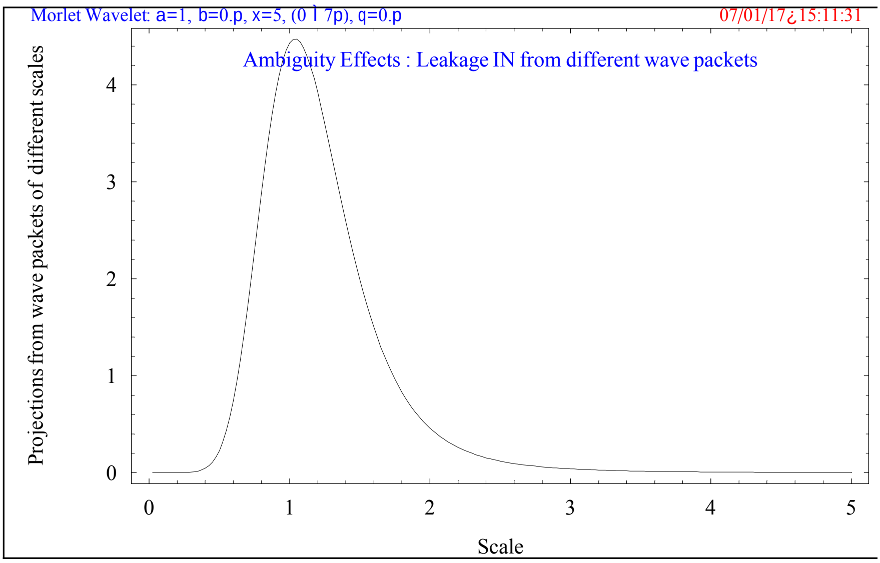
∞::indet : Indeterminate expression $e^{\text{ComplexInfinity}}$ encountered.

NIntegrate::inum : Integrand Indeterminate is not numerical at {u} = {1.}.

Ambiguity Effects : Leakage IN from different wave packets

Morlet Wavelet: α=1, β=0.π, ξ=5, (0↔7π), θ=0.π

Graphics::gptn : Coordinate 2. If[Indeterminate == 0. && Indeterminate > -1., <<1>>, <<1>>] in {0., 2. If[Indeterminate == 0. && Indeterminate > -1., <<1>>, NIntegrate[<<1>>]]} is not a floating-point number.



Graphics::gptn : Coordinate 2. If[Indeterminate == 0. && Indeterminate > -1., <<1>>, <<1>>] in {0., 2. If[Indeterminate == 0. && Indeterminate > -1., <<1>>, NIntegrate[<<1>>]]} is not a floating-point number.

```
Display::pserr : PostScript language error:
Warning…ont Utopia-Regular for Times-Roman

{2007, 1, 17, 15, 11, 50}    CPU:(00, 00, 0.235);
Time:(00, 00, 10)
```

- **<< "c:/lee/mat/000-p1_nb-m.m"**

FindMinimum[

$$-1*10*\left(\left(E^{-\frac{25\,(-1+sca)^2}{2\,(1+sca^2)}}+E^{-\frac{25\,(1+sca)^2}{2\,(1+sca^2)}}\right)\sqrt{\frac{\pi}{2}}\sqrt{1+\frac{1}{sca^2}}\;sca^2\right)\Big/$$

(2 (1 + sca²)), {sca, 1.01}]

<< "c:/lee/mat/000-p2_nb-m.m"

$\{-4.47397, \{sca \to 1.03926\}\}$

```
{2007, 1, 17, 15, 12, 1}     CPU:(00, 00, 0.015);
Time:(00, 00, 00)
```

ü Phase noise

File Macro LeakOut LeakIn PhaseNoise Root Others End

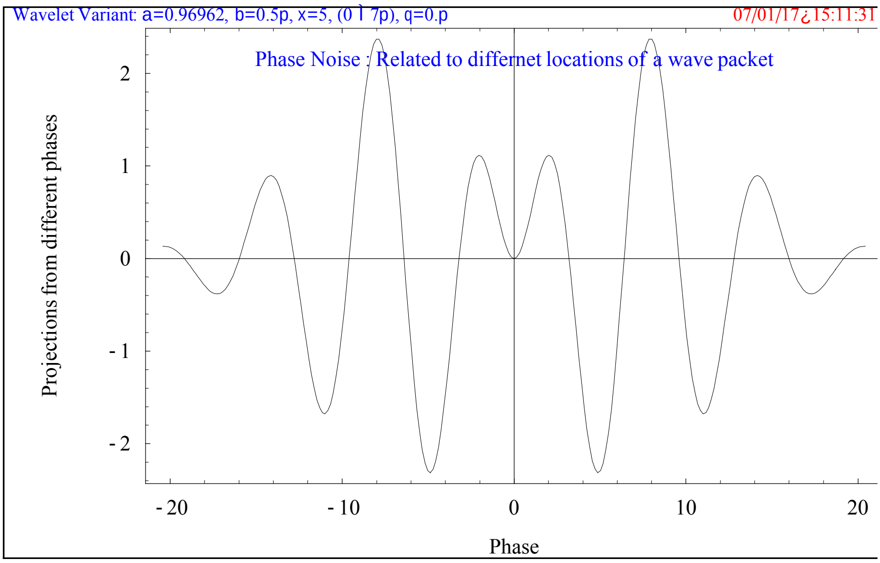
ü Phase noise (Wavelet variant)

```
(* -------------      Phase Noise
---------------- *)
typestr="Wavelet Variant: ";
xlabel="Phase";
ylabel="Projections from different phases";
flabel="Phase Noise : Related to differnet locations of a
wave packet";
<<"c:/lee/mat/000-p1_nb-m.m";
integright[peakshiftv_, scaadjv_, xlimitv_, phav_, xiv_]
:=
  NIntegrate[Cos[u/scaadjv-phav]*Sin[u]*
    Exp[-((u -peakshiftv )^2. + (u/scaadjv-
phav)^2.)/(2*xiv^2)],
    {u, 0, xlimitv}
    , MinRecursion->3, MaxRecursion->10 ];
integleft[peakshiftv_, scaadjv_, xlimitv_, phav_, xiv_]
:=
  NIntegrate[Cos[u/scaadjv-phav]*Sin[-u]*
    Exp[-((u + peakshiftv )^2. + (u/scaadjv-
phav)^2.)/(2*xiv^2)],
    {u, -1.*xlimitv, 0}
    , MinRecursion->3, MaxRecursion->10 ];
peakshiftp1=0.5;  peakshift= peakshiftp1 * Pi;
xlimitp1=7;  xlimit=xlimitp1 * Pi;
phap1=0;  pha=phap1 * Pi;
xi=5;
scaadj= 0.969621557058245997;  scap1=scaadj;
phaintp1=Table[  integright[peakshift, scaadj, xlimit,
phav, xi],
                  {phav, 0.05 Pi,  6.5  Pi , 0.05 Pi } ];
phaintp2=Table[  integleft[peakshift, scaadj, xlimit,
phav, xi],
                  {phav, 0.05 Pi,  6.5  Pi , 0.05 Pi } ];
phaintmid=2 * Table[  integleft[peakshift, scaadj,
xlimit, phav, xi],
                  {phav, 0.00 Pi,  0.00 Pi , 0.05 Pi } ];
posshiftsum=phaintp1+phaintp2;
midintsum=phaintmid;
datax=Join[ -1* Reverse[Table[ ni * Pi, {ni, 0.05, 6.5,
0.05}]], {0}, Table[ ni * Pi, {ni, 0.05, 6.5, 0.05}] ];
datay=Join[ Reverse[posshiftsum], midintsum, posshiftsum];
dataxy=Table[ {datax[[ i ]], datay[[i]]}, {i,1
,Length[datax]}];
EpsStrTmp="WV_PN";
doshowxy;
```

```
General::spell1 : Possible spelling error: new symbol
   name "scaadj" is similar to existing symbol "scaadjv".
```

Phase Noise : Related to differnet locations of a wave packet

Wavelet Variant: α=0.96962, β=0.5π, ξ=5, (0↔7π), θ=0.π




```
Display::pserr : PostScript language error:
Warning…ont Utopia-Regular for Times-Roman
```

```
        {2007, 1, 17, 15, 12, 1}    CPU:(00, 00, 0.812);
Time:(00, 00, 02)
```

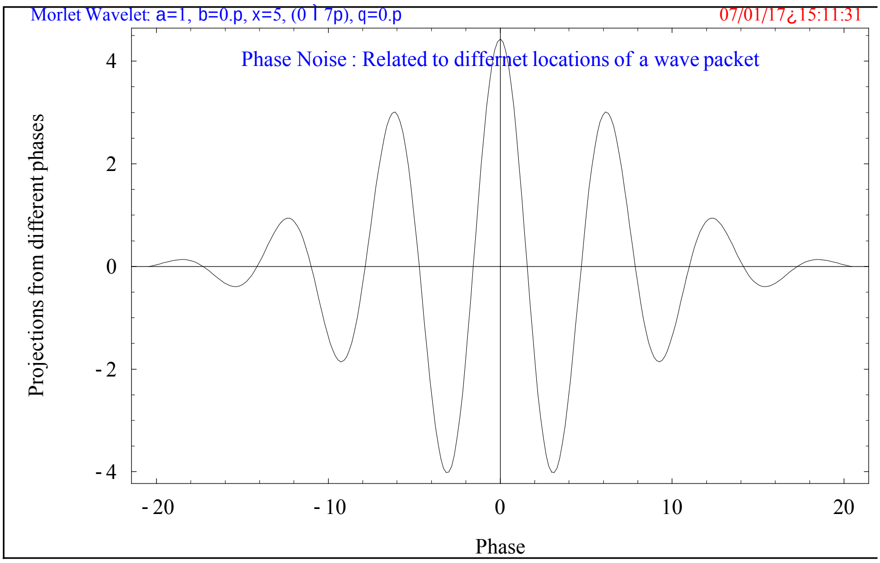
ü Phase noise (Morlet)

```
(* ---------     Phase Noise (Morlet)
----------- *)
typestr="Morlet Wavelet: ";
xlabel="Phase";
ylabel="Projections from different phases";
flabel="Phase Noise : Related to differnet locations of a
wave packet";
<<"c:/lee/mat/000-p1_nb-m.m";
integrightM[peakshiftv_, scaadjv_, xlimitv_, phav_, xiv_]
:=
  NIntegrate[Cos[u/scaadjv-phav]*Cos[u]*
    Exp[-((u -peakshiftv )^2. + (u/scaadjv-
phav)^2.)/(2*xiv^2)],
    {u, 0, xlimitv}
    , MinRecursion->3, MaxRecursion->10 ];
integleftM[peakshiftv_, scaadjv_, xlimitv_, phav_, xiv_]
:=
  NIntegrate[Cos[u/scaadjv-phav]*Cos[-u]*
    Exp[-((u + peakshiftv )^2. + (u/scaadjv-
phav)^2.)/(2*xiv^2)],
    {u, -1.*xlimitv, 0}
    , MinRecursion->3, MaxRecursion->10 ];
peakshiftp1=0.0;  peakshift= peakshiftp1 * Pi;
xlimitp1=7;  xlimit=xlimitp1 * Pi;
phap1=0;  pha=phap1 * Pi;
xi=5;
scaadj= 1;
scap1=scaadj;
phaintp1=Table[  integrightM[peakshift, scaadj, xlimit,
phav, xi],
                   {phav, 0.05 Pi,  6.50 Pi , 0.05 Pi } ];
phaintp2=Table[  integleftM[peakshift, scaadj, xlimit,
phav, xi],
                   {phav, 0.05 Pi,  6.50 Pi , 0.05 Pi } ];
phaintmid=2 * Table[  integleftM[peakshift, scaadj,
xlimit, phav, xi],
                   {phav, 0.00 Pi,  0.00 Pi , 0.05 Pi } ];
posshiftsum=phaintp1+phaintp2;
midintsum=phaintmid;
datax=Join[ -1* Reverse[Table[ ni * Pi, {ni, 0.05, 6.50,
0.05}]], {0}, Table[ ni * Pi, {ni, 0.05, 6.50, 0.05}] ];
datay=Join[ Reverse[posshiftsum], phaintmid, posshiftsum];
dataxy=Table[ {datax[[ i ]], datay[[i]]}, {i,1
,Length[datax]}];
EpsStrTmp="MW_PN";
doshowxy;
```

```
General::spell1 :
 Possible spelling error: new symbol name "integrightM" is
   similar to existing symbol "integright".

General::spell1 :
 Possible spelling error: new symbol name "integleftM" is
   similar to existing symbol "integleft".
```

Phase Noise : Related to differnet locations of a wave packet

Morlet Wavelet: α=1, β=0.π, ξ=5, (0↔7π), θ=0.π




```
Display::pserr : PostScript language error:
Warning…ont Utopia-Regular for Times-Roman
```

{2007, 1, 17, 15, 12, 4}   CPU:(00, 00, 0.766); Time:(00, 00, 03)

ü Roots

File | Macro | LeakOut | LeakIn | PhaseNoise | Root | Others | End

```
<< "c:/lee/mat/000-p1_nb-m.m"
g[b_, a_, s_] := Integrate[Cos[u / a] * Sin[(u)] *
```

```
        Exp[-((u - Pi / 2.) ^2. + u^2.) / (2 * s^2)], {u, 0, b}]
FindRoot[ g[ 5. * Pi, a, 5] == 0, {a,  0.95, 0.7,  1.2}]
<< "c:/lee/mat/000-p2_nb-m.m"
```

```
$Aborted
```

```
{2007, 1, 17, 15, 12, 8}    CPU:(00, 00, 32.219);
Time:(00, 00, 34)
```

```
<< "c:/lee/mat/000-p1_nb-m.m"
g[b_, a_, s_] := Integrate[Cos[u / a] * Sin[(u)] *
      Exp[-((u - Pi / 2.) ^2. + u^2.) / (2 * s^2)], {u, 0, b}]
FindRoot[g[7. * Pi, a, 5] == 0, {a, 0.9696, 0.7, 1.2}]
<< "c:/lee/mat/000-p2_nb-m.m"
```

```
<< "c:/lee/mat/000-p1_nb-m.m"
g[b_, a_, s_] := Integrate[Cos[u / a] * Sin[(u)] *
      Exp[-((u - Pi / 2.) ^2. + (u / a) ^2.) / (2 * s^2)],
  {u, 0, b}]
FindRoot[g[5. * Pi, a, 5] == 0, {a, 0.97, 0.7, 1.2}]
<< "c:/lee/mat/000-p2_nb-m.m"
(* Run["mmawav.bat"]; *)
```

ü Others

File | Macro | LeakOut | LeakIn | PhaseNoise | Root | Others | End

ü Close form integrations

$$g2(b\_, a\_, s\_) = \int_0^{\infty} \cos\left(\frac{u}{a}\right) \sin(u)\, e^{-\frac{\left(\frac{u}{a}\right)^{2.} + u^{2.}}{2 s^2}} \, du$$

```
<< "c:/lee/mat/000-p1_nb-m.m"
g2[b_, a_, s_] = Integrate[Cos[u / a] * Sin[(u)] *
      Exp[-((u) ^2. + (u / a) ^2.) / (2 * s^2)],
  {u, 0, Infinity}]
<< "c:/lee/mat/000-p2_nb-m.m"
```

```
<< "c:/lee/mat/000-p1_nb-m.m"
g3[b_, a_, s_] = Integrate[Cos[u / a] * Sin[(u)] *
      Exp[-((u - Pi / 2.) ^2. + (u / a) ^2.) / (2 * s^2)],
  {u, 0, Infinity}]
<< "c:/lee/mat/000-p2_nb-m.m"
```

```
<< "c:/lee/mat/000-p1_nb-m.m"
g4[a_, s_] := NIntegrate[Cos[u / a] * Sin[(u)] *
      Exp[-((u - Pi / 2.) ^2. + (u / a) ^2.) / (2 * s^2)],
```

```
  {u, 0, Infinity}]
g4[ 0.969621557058245997, 5]
<< "c:/lee/mat/000-p2_nb-m.m"
```

File Macro LeakOut LeakIn PhaseNoise Root Others End

■ Appendix

BIBLIOGRAPHY

[1] Battle, G. Cardinal spline interpolation and the block spin construction of wavelets. In C.K. Chui, editor, *Wavelets: A tutorial in Theory and Applications*, pages 73–90. Academic Press, Inc., San Diego, California, USA, 1992.

[2] Chui, C. K. *An Introduction to Wavelets*. Academic Press, Inc., San Diego, California, USA, 1992.

[3] Chui, C.K. On cardinal spline-wavelets. In M. B. Ruskai, G. Beylkin, R. Coifman, I. Daubechies, S. Mallat, Y. Meyer, and L. Raphael, editor, *Wavelets and Their Applications*, pages 439–452. Jones and Bartlett Publishers, Boston, New York, USA, 1992.

[4] Coifman, R., Y. Meyer, and M.V. Wickerhauser. Size properties of wavelet packets. In M. B. Ruskai, G. Beylkin, R. Coifman, I. Daubechies, S. Mallat, Y. Meyer, and L. Raphael, editor, *Wavelets and Their Applications*, pages 453–470. Jones and Bartlett Publishers, Boston, New York, USA, 1992.

[5] Daubechies, I. *Ten Lectures on Wavelets*. SIAM, Philadelphia, USA, 1992.

[6] Lamb, H. *Hydrodynamics*. Cambridge University Press, Cambridge, England, sixth edition, 1932.

[7] Lee, Y.R. Wave Characterizations Using Wavelet and Spectral Coherences. Technical report, No.2001–08, Institute of Harbor and Marine Technology, Taichung, Taiwan, 2000.

[8] Lee, Y.R. Signal Analysis from Wave Modulation Perspective. Technical report, No.2001–09, Institute of Harbor and Marine Technology, Taichung, Taiwan, 2001.

[9] Lee, Y.R. Wavelet Time-Frequency Analysis - An Optimum Basis and Its Applications to Water Waves. Technical report, No.93-79-7112, Institute of Harbor and Marine Technology, Taichung, Taiwan, 2003.

[10] Lee, Y.R., and J. Wu. A quasi-wavelet function bases for improved time-frequency characterizations. *Proc. 21th Conf. on Ocean Engineering in Taiwan*, pages 101–108, 1999.

[11] Lee, Y.R., and J. Wu. The wavelet optimum basis and the phase distribution of the characteristic function. *Proc. 25th Conf. on Ocean Engineering in Taiwan*, pages 149–156, 2003.

[12] Mallat, S. Multiresolution approximation and wavelets. *Trans. Amer. Math. Soc.*, 315:69–88, 1989.

[13] Phillips, O.M. *The Dynamics of the Upper Ocean*. Cambridge University Press, New York, USA, second edition, 1977.

[14] Press, W. H., S. A. Teukolsky, W. T. Vetterling and B. P. Flennery. *Numerical Recipes in Fortran*. Cambridge University Press, New York, USA, second edition, 1992.

[15] Wickerhauser, M.V. Comparison of picture compression methords: wavelet, wavelet packet, and local cosine. In C. K. Chui, editor, *Wavelets: Theory, Algorithms, and Applications*, pages 585–621. Academic Press, Inc., San Diego, California, USA, 1994.

審查意見回覆表

交通部運輸研究所港灣技術研究中心

自辦研究計畫期末審查意見回覆表

(訊號分析法比較研究暨其於水波應用探討 — Asyst 仔波程式集)

審查委員及審查意見	意見回覆
李兆芳教授：	
1. 本計畫所考慮仔波分析程式以執行自動化和使用方便爲主要考量， 相當實際。在程式中增加圖形顯示及使用資料表改進，相當用心。	Asyst 程式爲一 DOS-Based 語言， 其親和性和使用方便性有其先天之不足性， 在這一方面，此處程式已作部分改良及考慮。 而自動化之全面性與圖形顯示、圖檔自動產生、資料表一貫製作， 等、亦是程式考慮的重點。
2. 利用所發展程式計算出仔波之結果，在報告中顯示合理結果。 程式之可用信及正確性可以接受。	本報告所含括程式係逐步擴展增進，由根基起建， 確保其正確性是爲最高優先、最低要求。 一些結果也與其他研究者之結果加以比較驗証。
3. 第八章結論之內容應爲執行本計畫之心路歷程，可以加在計畫最後。結論應爲本計畫之內容摘要總結。	第八章已加以改寫，加入了本計畫之內容摘要總結。
林銘崇教授：	
1. 針對所提出之新型類仔波轉換應可進一步考慮專章處理，以突顯其意義。	有關於新型類仔波轉換之更詳細內容在先前研究報告中 有較詳盡之探討解說，並含蓋數值模擬與實訊應用。其進一步研究將爲未來工作之一。

郭一羽教授：	
1. 請加說明 Wavelet 在訊號分析（水波應用）上的優點及應用時機。	事實上這些問題正是多年來此處研究的重點。 由於此一報告係著眼於以 Asyst 語言 來開發含蓋內容廣泛與完整之仔波程式集。 故較著重程式面使用時機或功能特性說明， 而其於這方面的展示較簡略， 有關更詳細內容在先前數年的研究報告中有較詳盡之探討解說， 並含蓋數各種特性探討、數值模擬、與實訊應用。
2. 依計畫名稱，在 spectrum 與 wavelet 分析之比較應加以說明。	這一問題與上一問題都是多年來研究的關鍵核心。 多年來所出版的報告也多脫離不了這一問題。其簡要的相關探討內容爲： 熵值表現、相位糙音及移位不變性探討、水波分析最適函基之鑑取、 spectrum 共關協振與 wavelet 共關協振之表現差異、 水波不穩現象之鑑取、等。 由於此一報告著重於Asyst 語言程式集之功能化、 自動化、人性化、各類程式整合化、等， 故較著重程式面使用時機或功能特性展示， 而其於這方面的說明則較簡略。

李忠潘教授：	
1. 請於摘要增加說明如何應用於水波，並請於文內以例子說明。	如何應用於水波正是多年前研究的出發重點。 有關更詳細內容在先前數年的研究報告中有較詳盡之探討解說， 並含蓋數各種特性探討、數值模擬、與實訊應用。 此一報告因係著眼於以 Asyst 語言 來開發含蓋內容廣泛與完整之仔波程式集。 故較著重程式面使用時機或功能特性說明，而其於這方面的展示較簡略。
2. 爲了配合題目應有一節，專論於水波的實際應用。	相關探討內容如：熵值表現、水波分析最適函基之鑑取、 spectrum 共關協振與 wavelet 共關協振之表現差異、 水波不穩現象之鑑取、等，已部分加入。 由於此一報告著重於Asyst 語言程式集之功能化、 自動化、人性化、各類程式整合化、等， 故較著重程式面使用時機或功能特性展示， 而其於這方面的說明則較簡略。可以參閱多年來的報告書。

簡仲璟研究員：	
1. 本研究對於仔波（Wavelet）的分析自行開發建置一套程式。 惟本分析程式所需（或適用）之作業環境如何可再補充說明。	Asyst 程式語言其適用之作業環境是Microsoft DOS ， 在MS Windows中必需考量其EMS及 XMS 規格相容。相關之記憶體規劃， Overlay 模組需求已在報告中加以說明。
2. Wavelet 的分析程式目前已有多位學者建立。 本計劃所建立之程式其特色或重點、或優點和他人相較可再說明。	事實上就作者所知，當年開始撰寫仔波程式時， 還尚未有一家套裝應用軟體公司（如 MatLab 、Mathematica、等） 出版仔波分析工具套件。目前則都已含括在內，然而，就作者所知， 目前國內一些水波研究之仔波分析幾乎都是使用非自行開發之應用軟體。 比方而言，諸多研究對各類型仔波函基之水波模擬熵值表現並不了解， 對水波分析最適函基之認知亦相當薄弱，這些狀況有一些危險的義涵， 諸如不定性、不可重覆性、以偏概全、似是而非、甚而指鹿爲馬。 此處我們從根基起建，這於數學內涵了解與物理演伸有絕大的幫助， 也建立了對分析結果之信心。此外，此處程式之彈性、自動化性、 不同程式整合性、人爲錯誤性、大量分析批次化性、分析結果豐富性、 多元性、診斷確定性、等，都非使用套裝應用軟體所可比擬。

簡報資料

Wavelet Programming in Asyst

A basic research of IHMT

仔波Asyst程式集

李 勇 榮

運輸研究所
港灣技術研究中心
email: ronlee@ms4.hinet.net

http://www.ihmt.gov.tw/

1/86

回舊
回新
再見

內容

1. 以Asyst 語言開發廣泛之仔波分析程式。(包含多類屬之離散仔波與連續仔波，暨其諸多特性探討，如：正反轉換、爆展、熵值、常標值、最適函基、最適階函基、特性函數相位、共關協振、調適化時頻窗連續仔波轉換、及一新型類仔波轉換暨其特性探討、等。)
2. 資料之處理、檢視、呈現等功能。工作平台之開發、圖表文書管理系統。(整合一些外加程式及補助功能，如：Postfix、Lotus 介面、Mathematica 語言、WinEDT 巨集、LaTeX 程式及軟件集、等)

2/86

回舊
回新
再見

Wavelet bases tested

- Orthonormal (ON) ➲ [4, 1, 5, 7, 8]
  - Daubechies' most symmetric (ONxxS) ➲
  - Daubechies' most asymmetric (ONxxA) ➲
  - Coiflets (ONxxC) ➲
  - Meyer wavelet ➲
  - Battle-Lemarié (B&L) ➲
- Semi-orthogonal (SO) ➲ [1, 2]
  - Mother wavelet (SOxO)
  - Dual wavelet (SOxD)
- Bi-orthogonal (BO) ➲ [5]
  - Mother wavelet (BOxyO)
  - Dual wavelet (SOxyD)
- Wavelet Packet basis ➲ [3, 1, 5, 9, 10]
  - All ON groups (single or mixed) (WP best basis & WP best level)

ONxxA--M

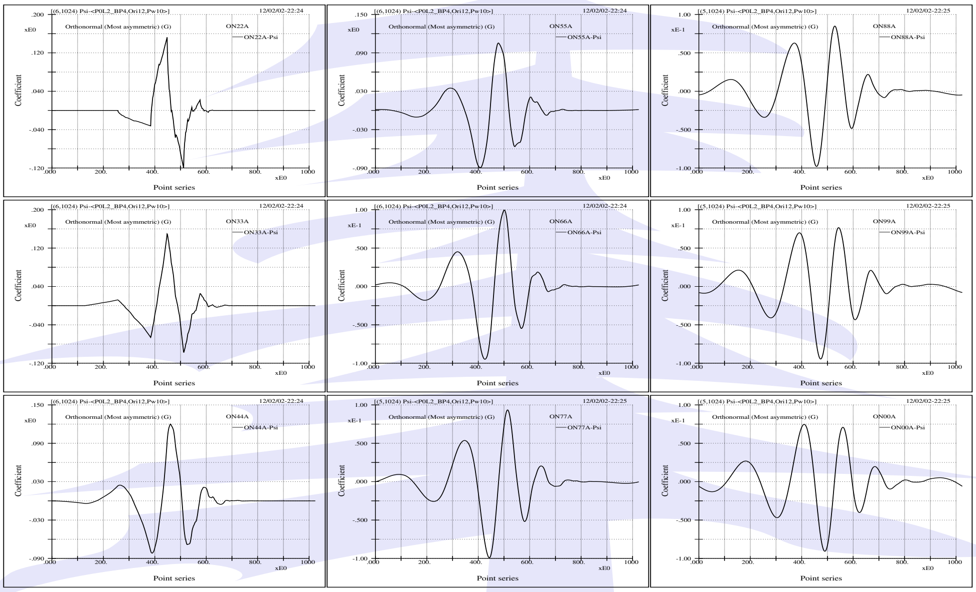


Figure 1: The mother wavelets of the ONxxA group originating from the point location of 12. Here the boundary point should be based on a level less or equal to 3.

BOxxO--F

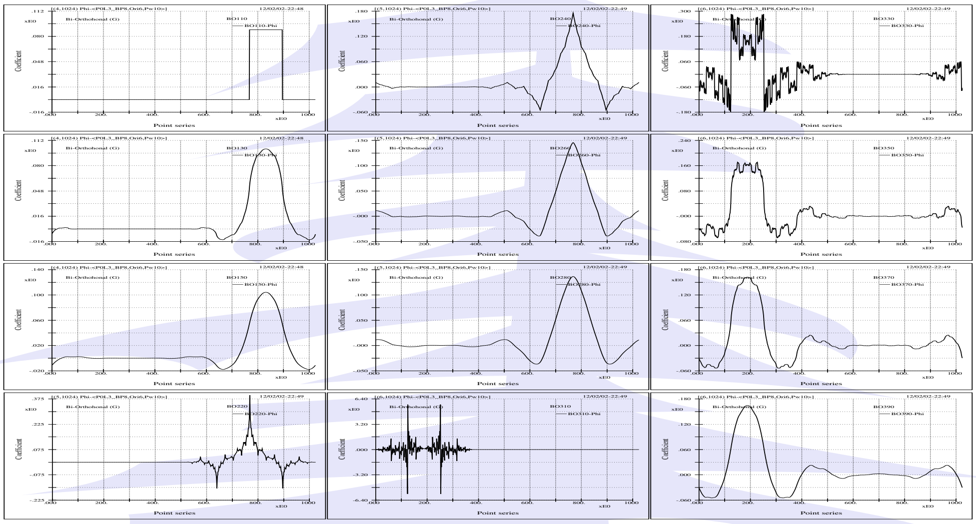


Figure 12: The farther wavelets of the BOxxO group originating from the point location of 6. Here the boundary point should be based on a level higher or equal to 3.

回舊
回新
再見

Concepts of time-frequency windows

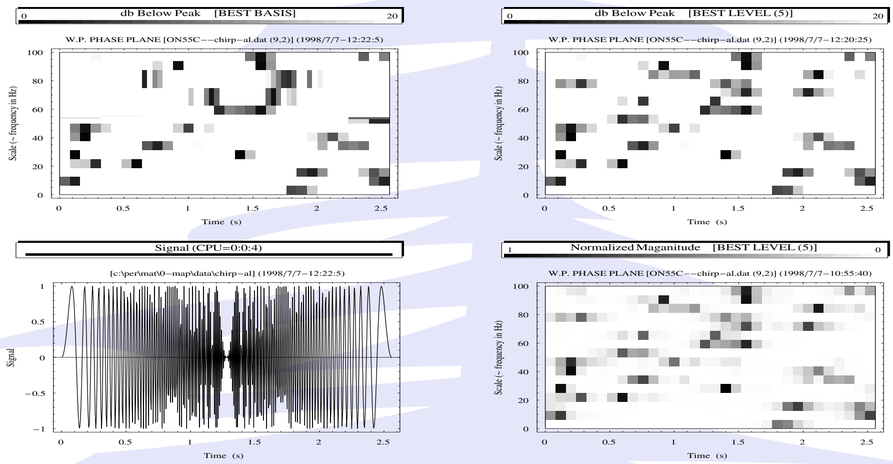


Figure 14: Phase plane of a wavelet packet's best basis time-frequency windows `(top left)` using a linear chirp signal with aliasing `(bottom left)`. Different representations are also shown in the right column (logarithmic measure (`top right`); linear measure (`bottom right`)) Here wavelet packets associated with coiflet of 30 convolution weights is used. The original signal, if not under-sampled, has linear instantaneous frequency distribution form 0 to 100 Hz. Note non-symmetric effects due to the nonlinear phase filtering and the scattering of windows due to composite frequency bands forming the wavelet.

回舊
回新
再見

The entropy criteria

- The general notions (p : pdf)

$$H(p) = -\sum_i p_i \log p_i$$

- The geometric notion ($p_i = |c_i|^2/\|C\|^2$)

$$H(p) = \log \|C\|^2 - \frac{\sum_i |c_i|^2 \log |c_i|^2}{\|C\|^2}$$

- The theoretical dimension notion

$$D(p) = e^{H(p)} = \prod_i (p_i^{-p_i})$$

- The different norm notions (displacement or energy norm)

回舊
回新
再見

Entropy of ON snd SO wavelet coefficients

Table 1: Entropy of orthonormal and semi-orthogonal wavelet coefficients as well as spectral coefficients under various statistic criteria.

Wavelet	L**2 coefficient entropy (0 division)	L**2 probability entropy (300 divisions)	L**1 probability entropy (300 divisions)	L**2 probability entropy (200 divisions)	Theotetical dimension (L**2 300 divisions)
B&L	4.691	1.330	3.417	1.179	3.782
Meyer	4.647	1.294	3.365	1.132	3.646
SO3O	4.833	1.669	3.756	1.488	5.307
SO3D	1.823	0.219	1.306	0.172	1.245
Spectrum	2.809	0.270	3.044	0.244	1.310
ON22A	4.993	1.761	3.891	1.516	5.815
ON33A	4.773	1.384	3.499	1.225	3.975
ON44A	4.790	1.517	3.596	1.363	4.559
ON55A	4.819	1.553	3.631	1.367	4.727
ON66A	4.790	1.373	3.456	1.203	3.946
ON77A	4.675	1.355	3.461	1.203	3.877
ON88A	4.645	1.229	3.283	1.082	3.418
ON99A	4.719	1.412	3.501	1.252	4.106
ON00A	4.787	1.423	3.511	1.244	4.149
ON44S	4.835	1.461	3.557	1.281	4.311
ON55S	4.758	1.492	3.576	1.298	4.426
ON66S	4.754	1.402	3.501	1.225	4.065
ON77S	4.751	1.336	3.331	1.188	3.804
ON88S	4.714	1.366	3.481	1.224	3.918
ON99S	4.755	1.469	3.570	1.288	4.345
ON00S	4.635	1.278	3.378	1.134	3.591
ON11C	4.938	1.696	3.832	1.457	5.452
ON22C	4.827	1.468	3.520	1.284	4.342
ON33C	4.756	1.488	3.573	1.333	4.427
ON44C	4.690	1.297	3.337	1.157	3.658
ON55C	4.644	1.309	3.405	1.154	3.703

回舊
回新
再見

Cumulative pdf curves of various transforms (II)

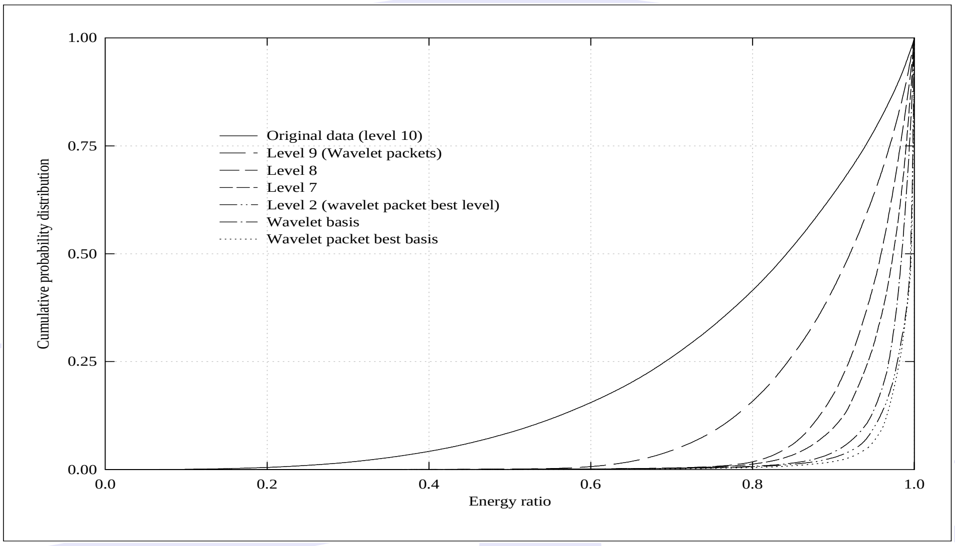


Figure 19: The cumulative probability distribution curves of the sorted wavelet and wavelet packet coefficients (L^2-norm squared, i.e., energy content) for various bases which all originate from a single mother wavelet. These bases include those of various wavelet packet levels, wavelet packet best basis, as well as the seeding wavelet basis ON77S; as are indicated in the legend.

31/86
回舊
回新
再見

$m_0(\xi)$ Phase Results

- The cardinal cubic B-spline wavelet is found to have constant phase distribution. ➲
- Wavelets that have similar visual appearance may show extremal phase differences. ➲
- BOxy0 $\approx$ BOxyD for any (x, y)
  Lengthening the support length $\longrightarrow$ no benefit ➲ ➲
- Lengthening of support lengthes of ONxxA and ONxxS $\longrightarrow$ more irregularities
  $\longrightarrow$ No need to further expand the construction ➲
- No other orthonormal wavelet may provide suitable (or better) characterizations ➲ ➲
- Linear distribution is not sufficient; a constant distribution seems to be a requirement. ➲ ➲

32/86
回舊
回新
再見

The Quasi Wavelet Basis Function

The devising of the basis function

The wavelet variant:

$$\psi(t) = \frac{1}{\pi^{\frac{1}{4}}} \left[\mathrm{sgn}(t) \sin \omega_0 t - i \cos \omega_0 t \right] e^{\frac{-t^2}{2}}. \tag{33}$$

The scaled and translated versions of the wavelet variant:

$$\psi_{a,b}(t) = \frac{1}{\sqrt{a}\pi^{\frac{1}{4}}} \left[\mathrm{sgn}(t) \sin \omega_0 \left(\frac{t-b}{a} \right) - i \cos \omega_0 \left(\frac{t-b}{a} \right) \right] e^{-\frac{\frac{t-b}{a}^2}{2}}, \tag{34}$$

The rendering of modulus and phase planes

Let $f(t)$ be a signal function, the modulus of the transform coefficient is defined either as

$$|\langle f(t), \mathbf{I_m}\psi(t)\rangle + i\mathcal{H}[\langle f(t), \mathbf{I_m}\psi(t)\rangle]|, \tag{35}$$

or

$$|\langle f(t), \mathbf{R_e}\psi(t)\rangle + i\mathcal{H}[\langle f(t), \mathbf{R_e}\psi(t)\rangle]|, \tag{36}$$

and the phase is defined as

$$\tan^{-1} \frac{\mathbf{R_e}\langle f(t), \psi(t)\rangle}{\mathbf{I_m}\langle f(t), \psi(t)\rangle} + \left(\frac{\pi}{2} \text{ or } 0 \right), \tag{37}$$

or

$$\tan^{-1} \frac{\mathbf{I_e}\langle f(t), \psi(t)\rangle}{\mathbf{R_m}\langle f(t), \psi(t)\rangle} + \left(\frac{\pi}{2} \text{ or } 0 \right). \tag{38}$$

The frequency leakage-out distribution curves

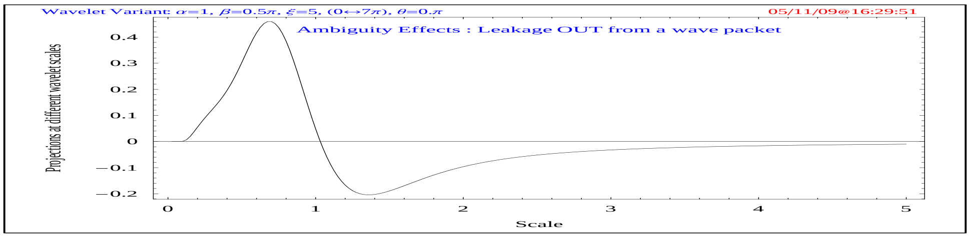


Figure 29: The frequency leakage-out distribution curve for the proposed basis function. The frequency leakage-out is the projection of the unit scale basis function into its neighboring scales. For $\omega_0 = 5$ the curve has a root at scale 0.969621. This zero value and the sharp steep slopes at both sides of the root make possible the easy identification of energy ridges.

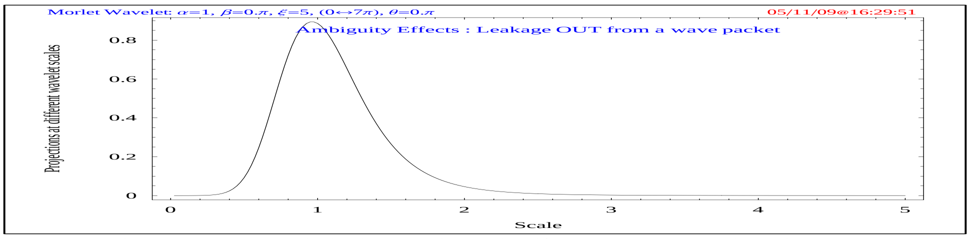


Figure 30: The frequency leakage-out distribution curve for the simplified Morlet wavelet. Again the frequency leakage-out is the projection of the unit scale basis function into its neighboring scales. The curve has a peak at scale nearly equal to 1. The weight centers around the peak and contributes to a relatively broader leakage of energy.

The frequency leakage-in distribution curves

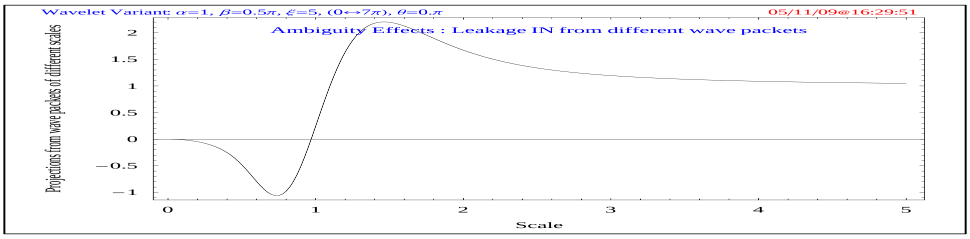


Figure 31: The frequency leakage-in distribution curve for the proposed basis function. The frequency leakage-in is the projection of a non-unit scale basis function into the unit scale basis function. It shows consistent results with the frequency leakage-out. Here the parameter value is the same as that of the previous figures.

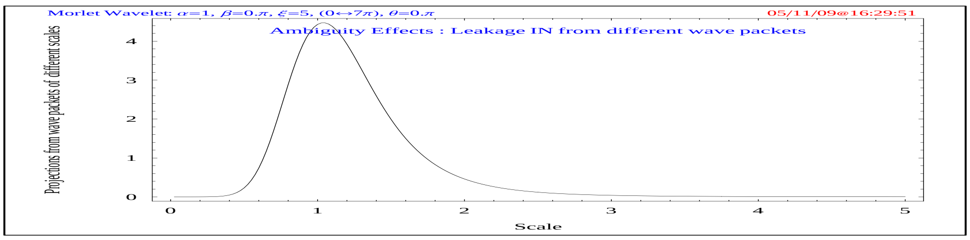


Figure 32: The frequency leakage-in distribution curve for the simplified Morlet wavelet. The frequency leakage-in is the projection of a non-unit scale wavelet function into the unit scale wavelet function. It shows consistent results with those of frequency leakage-out.

The analytic degree of ψ

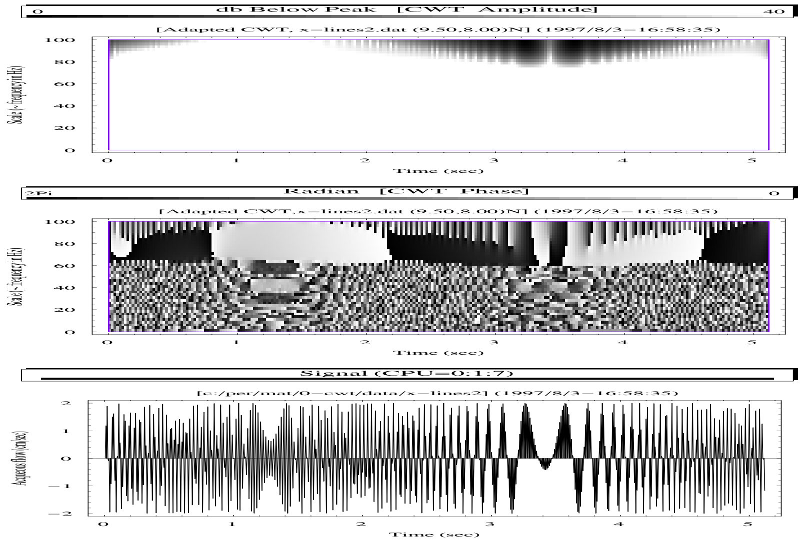


Figure 35: This figure shows the analytic degree of ψ related to equation 35. The top sub-figure shows the power (modulus squared) of the difference between $\langle f(t), \mathcal{A}[\psi]\rangle$ and $\langle f(t), \psi\rangle$, where $\mathcal{A}$ means the analytic counterpart. The mid sub-figure shows the corresponding phase. Here an X-signal composed of two linear chirps (bottom sub-figure, see figure *(Fig. 41)*) is used.

50/86

回舊 回新 再見

Tests and Applications

Simulated signals

- A parabolic chirp with a frequency range of zero to Nyquist rate of 100 Hz;
- A signal composed of two liner chirps with equal power contents
- An X-signal with a power ratio 0.01 between the two component signals;
- A signal composed of two liner chirps that are parallel and have the same power contents;
- A signal composed of two liner chirps that are parallel but with a power ratio of 0.04.

Water wave signals in laboratory tank

- Wind waves;
- Short wind waves with respective spectral peaks at about 2.0 to 2.6 Hz;
- Stokes waves with different fundamental harmonic frequencies and different wave steepness values.

51/86

回舊 回新 再見

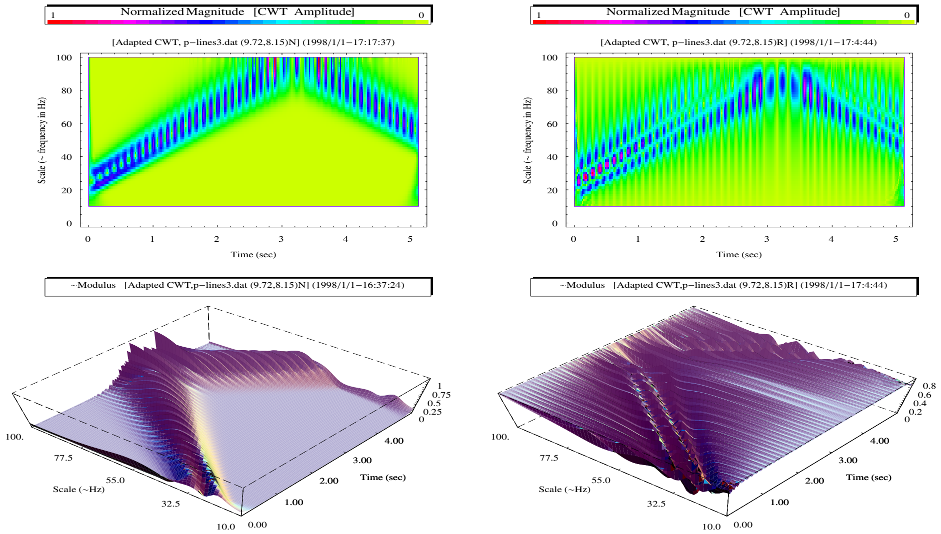


Figure 43: This figure shows the ridge extraction of a signal composed of a pair of parallel chirps that are with equal power contents. Here the frequency separation between the two chirps is one tenth Nyquist rate. The left sub-figures are for the simplified Morlet wavelet, and the right sub-figures are for the quasi wavelet basis function. It is clear that he power ridge given by the simplified Morlet wavelet is entirely misleading. But, for the quasi wavelet basis function, the two interfacial lines are clearly identifiable except at region near the Nyquist frequency. In the bottom right sub-figure, we have intentionally inverted the rendering, i.e., a trough in the 2-D plane (top right sub-figure) turns to a peak in the 3-D figure; and what are seen are the two clear spike lines.

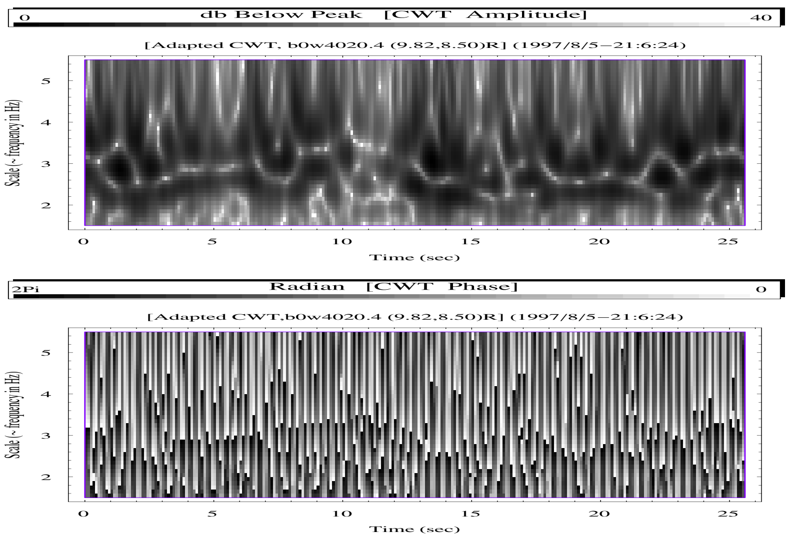


Figure 47: The modulus and phase planes for a lesser developed water wave signal (when compared to the previous figure, and it is due to a smaller wind speed in the tank) using the quasi wavelet basis function. The phase plane shows a very distinguish feature of multiple interfacing points along the time line. It reflects the existence of multiple scale components and the mutual interactions among them, and it also serves as the indication of instability or rapid wave modulation.

Wavelet Coherence (1/2)

- Spectral coherence
  - Cross correlation :

$$c(t) = \langle g(t+\tau), h(\tau) \rangle \tag{40}$$

  - Correlation coefficient :

$$r_s(t) = \frac{c(t)}{\|g(t)\| \|h(t)\|} \tag{41}$$

$$R_s^2(\omega) = \frac{|\mathbf{E}[G(\omega)\overline{H(\omega)}]|^2}{\left(\mathbf{E}[|G(\omega)|^2]\mathbf{E}[|H(\omega)|^2]\right)^{1/2}} \tag{42}$$

  - $\mathbf{E}$ is additional
  - Need to introduce one additional dimension

回舊 回新 再見

Wavelet Coherence (2/2)

- Wavelet coherence
  - The equation of resolution of identity

$$\langle g, h \rangle = \frac{1}{c_\psi} \int_0^\infty \frac{1}{a^2} \int_{-\infty}^{\infty} \langle g, \psi_{a,b} \rangle \overline{\langle h, \psi_{a,b} \rangle} dbda \tag{43}$$

$$\langle g_a, h_a \rangle = \frac{1}{c_\psi} \frac{1}{a^2} \int_{-\infty}^{\infty} \langle g, \psi_{a,b} \rangle \overline{\langle h, \psi_{a,b} \rangle} db \tag{44}$$

$$R_w^2(a) = \frac{|\mathbf{E}_b[\langle g, \psi_{a,b} \rangle \overline{\langle h, \psi_{a,b} \rangle}]|^2}{\left(\mathbf{E}_b[|\langle g, \psi_{a,b} \rangle|^2]\mathbf{E}_b[|\langle h, \psi_{a,b} \rangle|^2]\right)^{1/2}} \tag{45}$$

  - $\mathbf{E}$ is a natural extension of "$\int$"
  - No need to introduce additional dimension
  - Liu's problem [6] – Lack of $\mathbf{E}$

回舊 回新 再見

三種資料長度下波高與水流間之仔波與波譜共關協振

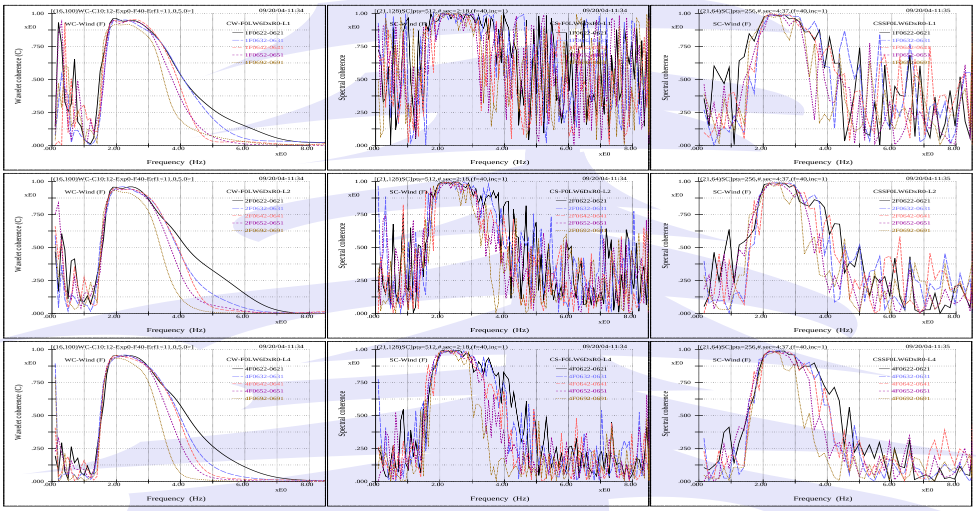


Figure 50: 三種資料長度下，波高與不同水深處水流間之仔波與波譜共關協振分佈曲線。上中下圖資料長度分別爲 1024、2048、4096 點。水速之量測點分別爲靜水面下 2、3、4、5、9 cm。圖中最左側分圖是爲仔波者，右側二行分圖是爲波譜者，其係選用兩種不同FFT 參數。

回舊
回新
再見

仔波實數函基共關協振與波譜者比較

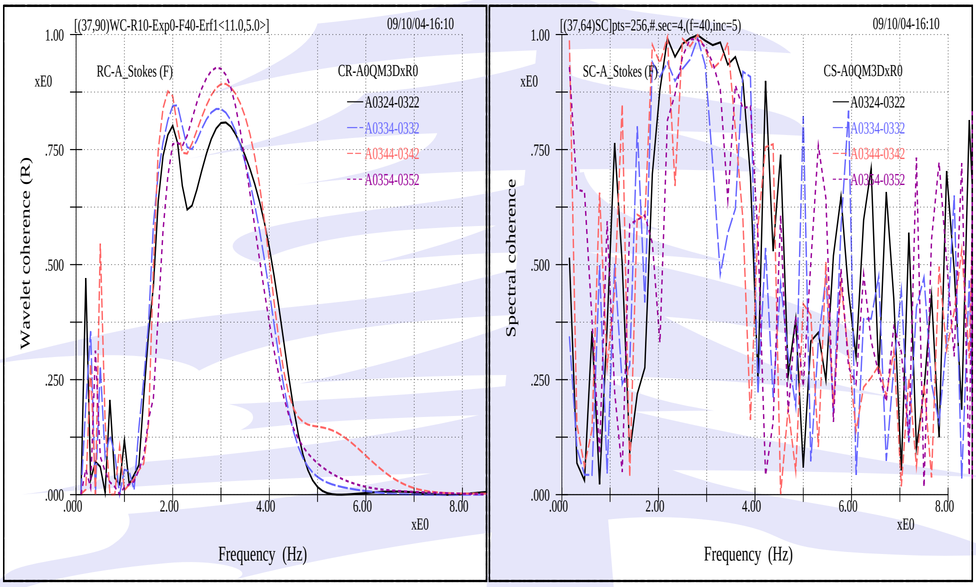


Figure 53: 仔波實數函基共關協振（左圖）與波譜者（右圖）比較。所用波浪爲一高波浪尖銳度之史脫克司波，共關協振爲波高與不同深度水流速之間，由仔波之圖可見較深處有最高之共關協振數值。

回舊
回新
再見