

AD-A197 035

DTIC FILE COPY

(4)

THEORETICAL STUDIES OF A TRANSIENT
STIMULATED RAMAN AMPLIFIER

Contract N00014-86-C-2341

SAIC Report No. 88/1674

by

Curtis R. Menyuk and Godehard Hilfer

April 19, 1988



Science Applications International Corporation

DTIC
ELECTE
JUN 16 1988
S H D

88 6 15 047

DISTRIBUTION STATEMENT A

Approved for public release;
Distribution Unlimited

4

**THEORETICAL STUDIES OF A TRANSIENT
STIMULATED RAMAN AMPLIFIER**

Contract N00014-86-C-2341

SAIC Report No. 88/1674

by

Curtis R. Menyuk and Godehard Hilfer

April 19, 1988

SAIC

Science Applications International Corporation

DTIC
ELECTE
JUN 16 1988
S H D

DISTRIBUTION STATEMENT A

Approved for public release;
Distribution Unlimited

Post Office Box 1303, 1710 Goodridge Drive, McLean, Virginia 22102, (703) 821-4300

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified			1b. RESTRICTIVE MARKINGS N/A		
2a. SECURITY CLASSIFICATION AUTHORITY N/A			3. DISTRIBUTION/AVAILABILITY OF REPORT Unlimited		
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A					
4. PERFORMING ORGANIZATION REPORT NUMBER(S) SAIC Report #88/1674			5. MONITORING ORGANIZATION REPORT NUMBER(S) N00014-86-C-2341		
6a. NAME OF PERFORMING ORGANIZATION Science Applications Int'l Corporation		6b. OFFICE SYMBOL (If applicable) N/A	7a. NAME OF MONITORING ORGANIZATION DCASMA - San Diego		
6c. ADDRESS (City, State, and ZIP Code) 1710 Goodridge Drive McLean, VA 22102			7b. ADDRESS (City, State, and ZIP Code) 7675 Dagget Street, Suite 200/300 San Diego, CA 92111-2241		
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Naval Research Laboratory		8b. OFFICE SYMBOL (If applicable) N/A	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER N00014-86-C-2341		
8c. ADDRESS (City, State, and ZIP Code) Scientific Office 4555 Overlook Avenue, S.W. Washington, D.C. 20375-5000			10. SOURCE OF FUNDING NUMBERS		
			PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.
11. TITLE (Include Security Classification) Theoretical Studies of a Transient Stimulated Raman Amplifier					
12. PERSONAL AUTHOR(S) Menyuk, Curtis Robert and Hilfer, Godehard					
13a. TYPE OF REPORT Final		13b. TIME COVERED FROM 11/86 TO 04/88		14. DATE OF REPORT (Year, Month, Day) 1988 April 19	
15. PAGE COUNT 339					
16. SUPPLEMENTARY NOTATION N/A					
17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number) Stimulated Raman, Nonlinear Optics, Light Amplification, Light Propagation,		
FIELD	GROUP	SUB-GROUP			
19. ABSTRACT (Continue on reverse if necessary and identify by block number) This final report summarizes Science Applications International Corporation's performance on contract no. N00014-86-C-2341 for the Naval Research Laboratory. Our principle deliverable, the codes RAM2D1 and PRAM1 have been completed on schedule and run successfully. Their operation is described in detail in this report as well as their application to cases of experimental importance. We have also carried out a number of analytical calculations in order to obtain greater insight into the code operation and the experiments and to make predictions in regimes of possible experimental interest which have not yet been explored. Some of these calculations were carried out in collaboration with Dr. John Reintjes of the Naval Research Laboratory. These calculations are summarized in this report. Relevant publications and presentations are also included.					
20. DISTRIBUTION/AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS			21. ABSTRACT SECURITY CLASSIFICATION Unclassified		
22a. NAME OF RESPONSIBLE INDIVIDUAL Curtis R. Menyuk			22b. TELEPHONE (Include Area Code) (703) 734-5957		22c. OFFICE SYMBOL N/A

FINAL REPORT

SAIC Report No. 88/1674

**THEORETICAL STUDIES OF A TRANSIENT
STIMULATED RAMAN AMPLIFIER**

April 19, 1988

by:

Curtis R. Menyuk and Godehard Hilfer

**Applied Physics Operation
Science Applications International Corporation
1710 Goodridge Drive
McLean, VA 22102**

Prepared under:

Contract N00014-86-C-2341

For:

Dr. John Reintjes (Code 6542)

**Laser Physics Branch
Naval Research Laboratory
4555 Overlook Avenue, SW
Washington, DC 20375-5000**

©

This report is copyrighted by Science Applications International Corporation

THEORETICAL STUDIES OF A TRANSIENT STIMULATED RAMAN AMPLIFIER

TABLE OF CONTENTS

ABSTRACT	iii
I. INTRODUCTION	1
II. CODE DEVELOPMENT	3
A. Basic Philosophy	3
B. Algorithmic Description of RAM2D1	5
C. Algorithmic Description of PRAM1	11
III. SCIENTIFIC STUDIES	21
A. Transient Raman Interactions	21
B. Beam Interactions in the Stationary Limit	27
C. Solitons and the Spectral Transformation	35
REFERENCES	42
APPENDICES	
A. Code Listings	43
RAM2D1 (Version C)	44
PRAM1 (Version CD)	65
B. Manual: Raman Amplifier Code RAM2D1 and Associated Diagnostic Program PRAM1	128
C. Publications	270
Application of Lie methods to autonomous Hamiltonian perturbations of the Korteweg-de Vries equation: Second-order calculation	271
Solitons in a birefringent Kerr medium	294
Pump replication in stimulated Raman scattering using a crossed-beam geometry	306
Asymptotic evolution of transient pulses undergoing stimulated Raman scattering	311

D. Presentations	324
Lie perturbation methods and their application to infinite-dimensional, Hamiltonian systems (abstract)	325
Numerical modelling of transient Raman amplification (abstract)	326
Linear theory of Raman beam cleanup and amplification in a crossing beam geometry (abstract)	326
Solitons in a Kerr medium (abstract)	327
Pump replication in stimulated Raman scattering using a crossed-beam geometry (abstract)	328
Numerical studies of transient Raman amplification (abstract)	329
Transient pulse evolution in the long distance limit of stimulated Raman scattering (abstract)	329
Formation of soliton-like structures in stimulated Raman scattering (abstract and summary)	330



Accession For	
NTIS GRA&I	☒
DTIC TAB	☐
Unannounced	☐
Justification	
By _____	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	

ABSTRACT

This final report summarizes Science Applications International Corporation's performance on contract no. N00014-86-C-2341 for the Naval Research Laboratory. Our principle deliverable, the codes RAM2D1 and PRAM1 have been completed on schedule and run successfully. Their operation is described in detail in this report as well as their application to cases of experimental importance. We have also carried out a number of analytical calculations in order to obtain greater insight into the code operation and the experiments and to make predictions in regimes of possible experimental interest which have not yet been explored. Some of these calculations were carried out in collaboration with Dr. John Rein-tjes of the Naval Research Laboratory. These calculations are summarized in this report. Relevant publications and presentations are also included.

I. INTRODUCTION

It is with pleasure and some pride that we present this summary of our accomplishments during this past year. We have completed the development of our principal deliverable, the code RAM2D1, which solves the basic equations governing transient, stimulated Raman interactions, accounting for both transient and diffractive effects. Using simple switches we can run the code in the transient regime where diffractive effects can be ignored or in the stationary regime where the transient effects can be ignored. Up to eight cases can be run simultaneously in these two limiting regimes. We have also developed a diagnostic code PRAM1 which uses DISSPLA routines to plot the results of our computer calculations. It can generate both ordinary plots and contour plots.

Both RAM2D1 and PRAM1 run on the NRL CRAY. They have also been tested on other CRAYs and should be easily modifiable to run on a variety of different machines.

In addition to developing and testing these codes, we have carried out a number of analytical and computational studies for the purpose of supporting existing experimental programs at the Naval Research Laboratory and exploring potential new ones. Some of these projects, but not all, make use of RAM2D1 and PRAM1. These studies have been marked by close cooperation with the experimentalists at the Naval Research Laboratory. We have explored transient phenomena in the long-distance limit both analytically and computationally. We have shown that the pump amplitude oscillates at a frequency proportional to $z^{1/2}$ and that the integrated intensity is proportional to $z^{-1/2}$ at long lengths. We have analytically studied stationary, multiple-beam interactions in a number of different limits. In collaboration with Dr. Reintjes of the Naval Research Laboratory, we have studied the conditions under which side beam replication occurs and have suggested a possible remedy. We have carried out computational studies of stationary, collinear beam propagation to determine the variation of the beam focal length due to nonlinear effects. In collaboration

with Dr. M. Duncan and Dr. R. Mahon, we have also carried out transient, computational studies to make detailed comparisons between theory and experiment. Finally, we have carried out studies of solitons aimed toward determining when they will appear and whether they are worth studying experimentally.

These studies have laid a firm foundation for work which we will undertake in the years to come. The codes RAM2D1 and PRAM1 will undergo further development to enlarge the range of phenomena which can be considered, improve efficiency, and improve our diagnostic capabilities. Several of the scientific studies we have undertaken, particularly those concerning side beam replication and focal point evolution, are likely to remain major concerns in the coming years.

II. CODE DEVELOPMENT

A. Basic Philosophy

In this section, we outline the basic philosophy governing our choice of algorithms and our choice of the plotting package DISSPLA.

The basic equations which we need to solve are

$$\frac{\partial E_L}{\partial z} - \frac{i}{2k_L} \frac{\partial^2 E_L}{\partial y^2} = -i\kappa_2 \frac{k_L}{k_S} Q E_S, \quad (2.1.a)$$

$$\frac{\partial E_S}{\partial z} - \frac{i}{2k_S} \frac{\partial^2 E_S}{\partial y^2} = -i\kappa_2 Q^* E_L, \quad (2.1.b)$$

$$\frac{\partial Q}{\partial t} + \Gamma Q = i\kappa_1 E_S^* E_L, \quad (2.1.c)$$

where k_L , k_S , Γ , κ_1 , and κ_2 are all physical parameters which are held constant in any individual computer run. Our boundary conditions are that $E_L(z, t)$ and $E_S(z, t)$ are fixed for all time at $z = 0$. We assume also that $Q(z, t) = 0$ at $t = -\infty$ for all z . Mathematically, Eqs. (2.1.a) and (2.1.b) are propagation equations while Eq. (2.1.c) is a constraint equation.

In solving these equations, our goal is to write a simple code which requires a minimum of space, runs with good efficiency, is robust, and is easily transportable.

The code RAM2D1 is written in FORTRAN and uses no canned routines except for the fast Fourier transform. Thus, this code is highly portable. The code PRAM1, being a plotting program, is dependent on the graphics package which is chosen. We use DISSPLA. While DISSPLA is somewhat difficult to learn, it is extremely powerful, and it exists on many different installations.

To solve the partial differential equations, we used a semi-spectral approach. For smooth initial conditions and infinite transverse boundaries, this approach has been shown for a large number of cases to be superior to finite difference or finite element methods.¹ The reason is that this approach is "infinite-order" in the transverse direction. It has the additional

advantage that the linear propagation is solved exactly (to within computer roundoff) so that in the limit of weak nonlinearity, a quite important limit in practice, step sizes in z can remain relatively large.

Use of the semi-spectral method places a premium on carrying out the fast Fourier transform efficiently. We have written it so that it vectorizes in different directions in the fully two-dimensional and stationary limits. Other portions of the code are also written to vectorize as efficiently as possible.

Another concern is reducing memory requirements. For this reason, we settled on a mid-step Euler approach, rather than a fourth order Runge-Kutta approach, although the latter is more accurate.² We have found nonetheless that in the fully two-dimensional limit, the code is often too large to run on the CRAY-XMP32 at NRL without modification. We have thus written a version of the code which allows us to move the data back and forth from core memory to the disk, keeping only what is needed for a single operation in core. While this approach solves the space problem, it necessitates a substantial amount costly I/O. A completely acceptable solution to this problem has not yet been found.

A final issue that requires discussion is robustness. The semi-spectral approach with a mid-step Euler advancement in z is extremely robust. As long as sufficient spectral bandwidth is provided through a sufficient number of node points, the method is never linearly unstable. The other place this issue arises is in the solution of the constraint equation. One must integrate Eq. (2.1.c) in a way which yields an accurate solution, independent of the ratio of $T_{\max}$ to T_2 , where $T_{\max}$ is the maximum $|t|$ -value of the t -region being kept. In the region where $T_2 \gg T_{\max}$ we use straightforward integration. When $T_2 \ll T_{\max}$, we use a Fourier transform approach. The critical point is that each approach does not work well at the extreme limits of the regime opposite to where it is applied. Hence, both are needed. We have set the crossover point at $T_{\max}/T_2 = 10$. At the crossover point, both methods

work well.

At present, the diagnostic code PRAM1 is set up to provide contour plots in $t - y$ space of the intensities, phases, and amplitudes of our principal fields, the pump, Stokes, and material excitation at fixed z -values. It also allows one to choose constant z and constant t sections for plotting purposes. These section plotting options are especially useful when plotting results obtained in the stationary and transient limits, as the contour plotting routines can no longer be used. The code PRAM1 does not plot z -history data; however, special, modified versions do exist for plotting this sort of data. More details can be found in the manual to RAM2D1 and PRAM1 which has been included as Appendix B.

The major concerns in designing this code have been flexibility and esthetics. Thus, the code has a large range of options, giving the user a large range of choices in how to plot the contour levels, where to choose the sections, how many to plot, and so on. Esthetic choices have included our insistence that the axis labels should be at "nice" values, the contour levels should be at "nice" values, and that the contour plots should lend themselves readily to the future production of movies.

B. Algorithmic Description of RAM2D1

The basic layout of RAM2D1 is shown in Fig. 2.1. In this section, we describe the basic algorithms used in each of the subroutines. The program listing is found in Appendix A, and more details on the variables and the program set-up can be found in Appendix B.

The main subroutine contains the input routine, routines to translate the input variables into variables used by the program, the basic stepping routine for the mid-step Euler method, and timing routines.

The variables NT and NY set the number of nodes in the t and y directions. These should be set equal to 2^N , where N is some integer, for the FFT routines to run properly.

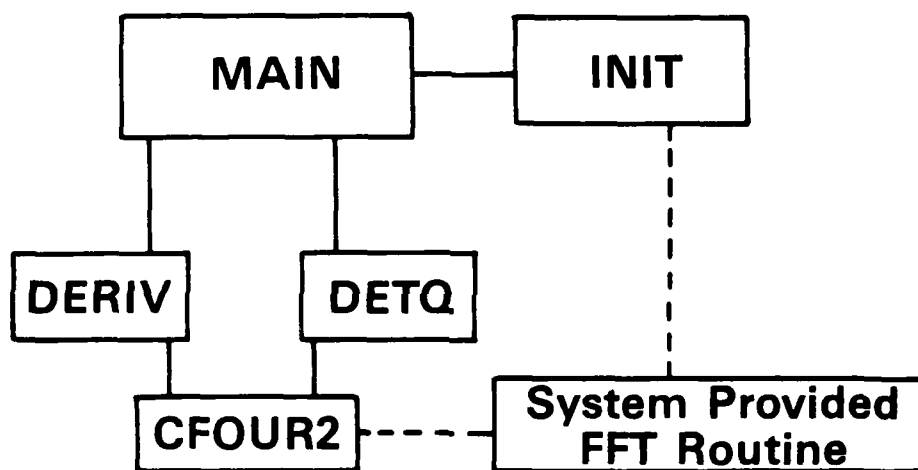


FIGURE 2.1. The structure of RAM2D1 is shown schematically.

Since they determine array sizes, they are set in a `PARAMETER` statement. The code must be re-compiled whenever they are changed. They also serve as switches. When $NT \leq 8$, the code assumes that the stationary limit is being used with the number of distinct cases set by NT ; similarly, when $NY \leq 8$, the code assumes that the transient limit is being used with up to 8 distinct cases.

Most other input parameters are read in through a `namelist` statement. (The only exceptions are `NP` and `NST` which set the maximum number of pump beams and the length of the timing data vector.) Parameters include: the actual number of pump beams, `NPUMP`; parameters which specify the box size, `TM` and `YM`; parameters to determine beam offsets, intensities, and widths, `YOFF`, `TOFF`, `YOST`, `TOST`, `YWIDTH`, `TWIDTH`, `YWST`, `TWST`, `RINT`, `RIST`; basic switches used by `INIT` to set beam type, `ICOND`, `RTYPE`, `ITYPE`; other parameters governing the beam shape, `RAMASM`, `RALASM`, `PHL`, `PHST`, `TOC`, `NHYP`, `RABAMP`, `RDSLIM`; parameters governing the beam intersection point, the final z -value, and the z -step, `ZINT`, `ZFINAL`, `ZSTEP`; a parameter setting the maximum number of z -steps `NMAX`; physical parameters, `RKP`, `RKS`, `TTWO`, `GAIN`; and a parameter governing how often z -data is recorded, `ZKEEP`.

From here, initialization of the derived quantities is carried out. The parameters `RKAP1` and `RKAP2` (κ_1 and κ_2) are determined from `GAIN` (g). The pump and Stokes amplitudes are determined from the input power fluxes. Other miscellaneous actions are carried out; in particular the parameters governing the final z -value and the z -values at which data are recorded are reduced slightly to avoid difficulties with roundoff when making equality comparisons.

Since the step size is fixed, so is the linear propagator. To save computer time, we calculate the propagator once and for all and store it in `CYVEC` where it is called as needed. Two propagators are needed, one for the pump and one for the Stokes. Specifically, referring

to Eq. 2.A.1, and defining the Fourier transform, of $X(y)$

$$\tilde{X}(k) = \frac{1}{2\pi} \int_{-\infty}^{\infty} dy e^{iky} X(y) , \quad (2.2)$$

we find that the linear propagator

$$\begin{aligned} \frac{\partial E_L}{\partial z} - \frac{i}{2k_L} \frac{\partial^2 E_L}{\partial y^2} &= 0 , \\ \frac{\partial E_S}{\partial z} - \frac{i}{2k_S} \frac{\partial^2 E_S}{\partial y^2} &= 0 , \end{aligned} \quad (2.3)$$

has the solution

$$\begin{aligned} \tilde{E}_L(k, z) &= \tilde{E}_L(k, 0) \exp(-ik^2 z / 2k_L) , \\ \tilde{E}_S(k, z) &= \tilde{E}_S(k, 0) \exp(-ik^2 z / 2k_S) . \end{aligned} \quad (2.4)$$

We note as well that the fast Fourier transform produces the k -values,

$$\begin{aligned} k &= \frac{2\pi(n-1)}{y_{\max} - y_{\min}} , \quad (1 \leq n \leq NY/2) \\ k &= \frac{2\pi(n-1-NY)}{y_{\max} - y_{\min}} , \quad (NY/2 < n \leq NY) \end{aligned} \quad (2.5)$$

where $y_{\min}$ and $y_{\max}$ are the minimum and maximum values of the y -box size.

We now initialize arrays which are used in the determination of Q . There are three methods used for determining Q , depending on the regime in which the code is being used. In method 1, which applies to the stationary limit, we set

$$Q = -i\kappa_1 \frac{E_S^* E_L}{\Gamma} . \quad (2.6)$$

In method 2, which applies when $\text{TRAT} = T_{\max}/T_2 < 10$, we use the integral expression

$$Q(z, t) = -i\kappa_1 e^{-\Gamma t} \int_{-\infty}^t e^{\Gamma t'} E_S^*(z, t') E_L(z, t') dt' , \quad (2.7)$$

The integrand has no t -dependence which allows us to calculate the integral through a running application of the trapezoidal rule. The vector $WQ1$ is initialized to contain $\exp(\Gamma t)$,

and the vector WQ2 is initialized to contain $\exp(-\Gamma t)$. Method 3 applies when $\text{TRAT} > 10$. Here, we define the Fourier transform

$$\bar{X}(\omega) = \frac{1}{2\pi} \int_{-\infty}^{\infty} dt e^{i\omega t} X(t) . \quad (2.8)$$

We then note

$$\bar{Q}(z, \omega) = -i\kappa_1 \frac{E_S^*(z, \omega) E_L(z, \omega)}{\Gamma - i\omega} . \quad (2.9)$$

Our approach is thus to multiply E_S^* by E_L , take the Fourier transform, multiply by $-i\kappa_1/(\Gamma - i\omega)$ and inverse transform. The vector CWQ contains the Fourier data, and COMVEC contains the factor $-i\kappa_1/(\Gamma - i\omega)$. Method 2 does not succeed when TRAT becomes very large due to underflow/overflow problems. Method 3 does not succeed when TRAT becomes very small because $(\Gamma - i\omega)^{-1}$ becomes singular.

In the code, we next store our initial data and enter the routine INIT which initializes the pump and Stokes fields. The ICOND value determines the basic form of the initialization. When $\text{ICOND}=1$, a sech amplitude profile is taken in both the t and y directions. When $\text{ICOND}=2$, a sech^2 amplitude profile is taken in the t -direction and a hyper-Gaussian profile in the y -direction. Additional parameters adjust the leading edge of the Stokes pulse so that it can be sharper than the pump pulse and a chirp, or phase variation, can be added to the Stokes pulse. When $\text{ICOND}=3$, a transient case is run. The amplitude profiles are governed by ITYPE and include sech, Lorentzian, square, exponential, Gaussian, and hyper-Gaussian profiles. The power of the amplitude or exponent is governed by RTYPE. When $\text{ICOND}=4$, a stationary case is run with a hyper-Gaussian profile. Other parameters allow non-zero phase and intensity aberrations to be introduced and govern their strength.

The subroutine INIT functions as a separate element and takes up increasing space as more alternative initializations are added. At present, however, the space allocated is negligible, and it is not worthwhile to separate INIT from RAM2D1.

We next record the initial fields and, except in the transient case where Fourier-transformed data is not used, calculate and record the Fourier data as well.

We now enter the main loop where the mid-step Euler method is applied to the forward stepping of $E_L(z, y, t)$, $E_S(z, y, t)$, and $Q(z, y, t)$ from z_n to $z_{n+1} = z_n + \Delta z$. We assume that E_L , E_S , and Q are known at $z = z_n$. We recall that the equations for $\tilde{E}_L$ and $\tilde{E}_S$ are

$$\begin{aligned}\frac{\partial \tilde{E}_L}{\partial z} + \frac{ik^2}{2k_L} \tilde{E}_L &= -i\kappa_2 \frac{k_L}{k_S} \widetilde{Q E_S} , \\ \frac{\partial \tilde{E}_S}{\partial z} + \frac{ik^2}{2k_S} \tilde{E}_S &= -i\kappa_2 \widetilde{Q^* E_L} .\end{aligned}\tag{2.10}$$

The routine DERIV calculates the right-hand sides of Eq. (2.10) by first determining the multiplicands $Q E_S$ and $Q^* E_L$ and then transforming. The quantities $\tilde{E}_L$ and $\tilde{E}_S$ are then advanced to the midpoint using the formula

$$\begin{aligned}\tilde{E}_L(z_{n+1/2}) &= \exp[(-ik^2/2k_L)(\Delta z/2)] \tilde{E}_L(z_n) \\ &\quad - i \frac{\Delta z}{2} \kappa_2 \frac{k_L}{k_S} Q(z_n) E_S(z_n) , \\ \tilde{E}_S(z_{n+1/2}) &= \exp[(-ik^2/2k_S)(\Delta z/2)] \tilde{E}_S(z_n) \\ &\quad - i \frac{\Delta z}{2} \kappa_2 Q^*(z_n) E_L(z_n) ,\end{aligned}\tag{2.11}$$

where $z_{n+1/2} = z_n + \Delta z/2$. We recall that the exponential factors are stored in CYVEC. The routine DETQ first determines $E_L(z_{n+1/2})$ and $E_S(z_{n+1/2})$ by inverse transforming $\tilde{E}_L$ and $\tilde{E}_S$. It then calculates $Q(z_{n+1/2})$ using the appropriate method. We now repeat the procedure, first using DERIV to calculate the right-hand side of Eq. (2.10) at $z = z_{n+1/2}$ and then using the formula

$$\begin{aligned}\tilde{E}_L(z_{n+1}) &= \exp[(-ik^2/2k_L)\Delta z] \tilde{E}_L(z_n) \\ &\quad - i\Delta z \kappa_2 \frac{k_L}{k_S} Q(z_{n+1/2}) E_S(z_{n+1/2}) , \\ \tilde{E}_S(z_{n+1}) &= \exp[(-ik^2/2k_S)\Delta z] \tilde{E}_S(z_n) \\ &\quad - i\Delta z \kappa_2 Q^*(z_{n+1/2}) E_L(z_{n+1/2}) ,\end{aligned}\tag{2.12}$$

to determine $\tilde{E}_L$ and $\tilde{E}_S$ at $z = z_{n+1}$. Using DETQ, we finally obtain E_L , E_S , and Q at $z = z_{n+1}$ and are ready for the next loop iteration. For transient runs, Fourier transform data is not calculated.

At the end of each loop iteration, a check is made to determine whether data should be recorded.

After exiting the main loop, the timing data and the number of data records is recorded.

C. Algorithmic Description of PRAM1

The purpose of PRAM1 is to display desired aspects of the data that RAM2D1 generates and files. Those data are the complex field amplitudes of the pump beams, Stokes beam, the material excitation, and their Fourier transform representations (=6 field arrays). Whenever Fourier transforms are mentioned it is understood to be the transform with regard to the transverse spatial dimension y . The desired format of display is that of contour plots of the intensity of these fields versus time coordinate and versus transverse spatial coordinate at a given point z along the path of propagation. In addition to that, cross-sections (of the contour plots) of the intensity and sections of the field phase and amplitude at user-defined z -values can be displayed. In the case of one-dimensional simulations no contour plots are available.

Intensity, phase, and real and the imaginary part of the amplitude can all be displayed. These three types of plots are desired of the three fields and their Fourier transforms. Hence, 18 different types of sections in addition to the intensity contour plots can be generated.

The user can generate any one or several graphs of the described type by specifying appropriate values for the elements of the flagging vector ISRF and array CSEC in the input data file NP-.DAT. For this purpose the field arrays are numbered (I through VI) and the sections (1-18). Which numeral corresponds to which type of graph and their sequence of

appearance in the output is as follows:

- I. contour plot of pump intensity
 - 1. sections of pump intensity
 - 2. sections of pump phase
 - 3. sections of pump amplitude (real/imag)
- II. contour plot of pump FFT intensity
 - 4. sections of pump FFT intensity
 - 5. sections of pump FFT phase
 - 6. sections of pump FFT amplitude (real/imag)
- III. contour plot of Stokes intensity
 - 7. sections of Stokes intensity
 - 8. sections of Stokes phase
 - 9. sections of Stokes amplitude (real/imag)
- IV. contour plot of Stokes FFT intensity
 - 10. sections of Stokes FFT intensity
 - 11. sections of Stokes FFT phase
 - 12. sections of Stokes FFT amplitude (real/imag)
- V. contour plot of mat. exct. intensity
 - 13. sections of mat. exct. intensity
 - 14. sections of mat. exct. phase
 - 15. sections of mat. exct. amplitude (real/imag)
- VI. contour plot of mat. exct. FFT intensity
 - 16. sections of mat. exct. FFT intensity
 - 17. sections of mat. exct. FFT phase
 - 18. sections of mat. exct. FFT amplitude (real/imag)

VII. contour plot of pump and Stokes intensity

19. sections of sum of pump and Stokes intensity

VIII. contour plot of pump and Stokes FFT intensity

Three more types of plots than the expected 6+18 were added on the bottom of the list. These are the surfaces VII and VIII and section 19. These graphs plot pump and Stokes data simultaneously on a common scale. Section 19 draws the sum of the fields weighted by a certain factor so as to compose a special invariant of the Raman interaction.

The roman numerals tell which element of the vector ISRF is the flag that determines if that particular contour plot will be generated or skipped.

ISRF(n) = 0 contour plot skipped

ISRF(n) = 1 contour plot drawn with contours labeled

ISRF(n) = -1 contour plot drawn; no labels on contours

The default value is zero which is set in PRAM1.

Each sectional plot is associated with one element of the complex array CSEC. The position of the element specifies uniquely one cross sectional plot. The arabic numerals in the list above indicate the row number (first array index) of CSEC with which each described type of section is associated. The column number (second index) of the elements of CSEC numbers the particular cross sectional plot of that type. The parameter NSEC (≤ 9) gives the maximally allowed number of sections of each type for the run.

The plotting of any cross-section is done depending on the values of the real and imaginary part of their representative element in CSEC. At the beginning of execution PRAM1 sets them all equal to zero as the default value. The values specified in the input data file replace these zeros.

In a two-dimensional simulation the value of the imaginary part of each element of the array CSEC means:

- = 0.0: that this sectional plot is not requested
- = 1.0: that this sectional plot is requested and that it shall be a cross-section parallel to the y -axis of the surface under question at a fixed t -value as given in physical units (psec) by the real part of the current element of CSEC. The first index of the array(s) $SRF(I)$ in PRAM1 is being held constant for this plot at the value ISEC which is the grid point that corresponds best to the fixed t -value;
- = 2.0: that this sectional plot is requested and that it shall be a cross section parallel to the t -axis of the surface under question at a fixed y -value as given in physical units (cm or 1/cm) by the real part of the element of CSEC in question. The second index of the array(s) $SRF(I)$ in PRAM1 is being held constant for this plot at the value ISEC which is the grid point that corresponds best to the fixed y -value.

In short: the imaginary part tells which variable to hold constant, and the real part tells at what value (in physical units).

In a one-dimensional simulations the location of the array elements within CSEC has the same correspondence with cross-sectional plots as in two-dimensional simulations. The exact values of the imaginary parts of CSEC no longer matter except that they must be larger than 0.001 for the corresponding sections to be generated. In the diagnosis of a run by RAM2D1 where only one one-dimensional case was simulated the exact value of the real parts of CSEC also do not matter; however, they must be larger than 0.5 and less than 8.5 for the plot to be generated. When several cases have been simulated in one run simultaneously the value of the real part (1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, or 8.0) of each element of array CSEC identifies which of these cases is meant to be diagnosed.

The program has the structure shown in Fig. 2.2. The program starts by setting default values for the graphics output parameters as specified in the data statements. These

default values are updated by reading customized values from the namelist file NPRAM1. The updated parameter set is then written, depending on the value of the 4 elements of the vector LPRMT(n), onto the first 4 graphics frames in the output META-file call PLT2.DAT. The value of the n-th element of LPRMT should be equal to 1 if the n-th page of parameters is to be plotted, and equal to 0 if not. The content of the four parameter graphics frames is shown in the examples of the appendices.

The program continues by calculating several constants. Among these constants are the end values and interval sizes for the plotting of the frequently used y - and t -coordinate axes. Then the large DO-loop 500 is entered. It reads the data for each requested plot and converts it into a device-independent graphics frame that is stored in the META-file. Once all data are scanned with respect to the requested graphs the generated graphics frames in the META-file are transferred to the VAX storage disk under the name PLT2.DAT.

In detail, the ensuing main part of this program acquires the electric field data from the input data file F—— by reading sequentially the i -th record specified by the value i of the consecutive elements of the vector KZ. These complex amplitude data are converted into real intensity data (array SRF), or are split into their real (array SRF) and imaginary (array SRFI) parts for plotting of the phase and/or amplitudes. Following their acquisition, the real arrays, SRF and SRFI, are handed, like other necessary parameters, through FORTRAN COMMON BLOCKS to the subroutine CNTR (for contour plotting) and to the subroutine CRSSCT (for cross sectional plots).

The subroutine CNTR is then called depending on the value of the relevant element of ISRF. Interleaved in these calls are the calls to the subroutine CRSSCT, depending on the sum of values of all elements of the line of CSEC in question. If this sum is non-zero CRSSCT is called to generate the requested graphs, if this sum is zero the DO-loop proceeds. Following the end of DO-loop 500 the program just closes the META-file and then stops execution.

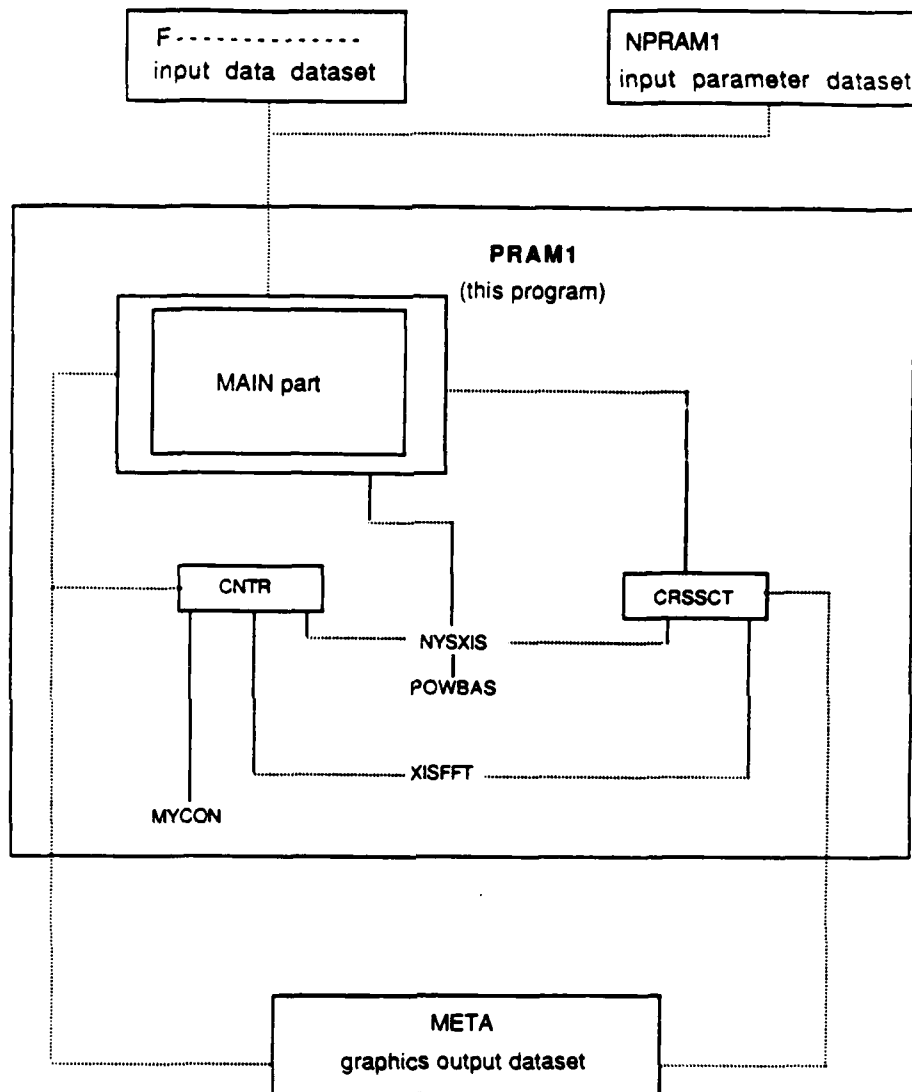


FIGURE 2.2. The structure of PRAM1 is shown schematically.

The contouring subroutine CNTR makes use of the subroutine MYCON which generates a customized dotted line for the half-height contour. The cross section subroutine CRSSCT calls repeatedly on the subroutine NYSXIS which finds "nice" values for coordinate axis limits and intervals. NYSXIS in turn uses subroutine POWBAS to find the next lower integral power of 10 for maximas and minimas. Both subroutines CNTR and CRSSCT share subroutine XISFFT when making secondary axes for FFT-plots.

The subroutine CNTR contains the calls to the special routines of the DISSPLA graphics library that create the graphics data for each contour plot. Several features of those plots are customized with respect to the standard that DISSPLA provides. For example, the software that generates axis tick marks has been amended by the non-DISSPLA subroutine NYSXIS. The subroutine NYSXIS computes "nice" tick marks along the coordinate axes. The subroutine XISFFT computes the location, extrema and intervals of the transformed variable axis in FFT-plots. The subroutine changes the field data in order to plot the logarithmic intensity as desired. Therefore, the intensity data in SRF (SRFI) have to be restored in the main part of the program before the intensity cross sections can be generated.

The call to subroutine CNTR contains the parameter KSRF which identifies the type of graph. Depending on its values, various titles, coordinate axes, and labels are selected and drawn. The sign of KSRF toggles the labeling option of the main contour lines (positive KSRF labels, negative KSRF no labels). The main contour lines are solid lines representing integral powers of 10. A number NDEC such lines will be drawn below the peak of the surface maximum. A number ILN (≤ 8) other contour lines (dashed lines) are drawn between the main contour lines corresponding to the integral multiples of the next lower integral power of ten. Which integral multiples are to be drawn is determined by the first ILN elements of the vector LEVEL. If the input parameter ISHM = 1 a dotted contour will mark the half-height level, if ISHM = 0 this line will not be drawn, if ISHM = -1 the half-height contour and a

dot at the surface maximum will be drawn.

The subroutine CRSSCT contains the calls to the special routines of the DISSPLA graphics library that create the graphics data for each cross sectional plot from the modified field data in the array(s) SRF (SRFI). Just as in the case of CNTR, several features of those plots are customized with respect to the standard that DISSPLA provides.

The three categories of cross sectional plots are: intensity plots (following statement label 300), phase plots, and amplitude plots (both following statement label 400). When intensity cross sections are called for, this subroutine executes DO-loop 390 that does all cross sections specified in row MSRF of array CSEC and thereafter returns control to the main program. When phase or amplitude cross sections are called for, this subroutine executes DO-loop 490 which generates all phase sections specified in row MSRF of array CSEC. Immediately afterwards, since phase plots and amplitude plots are derived from the same data in the arrays SRF and SRFI, DO-loop 590 is executed which generates all amplitude cross sections that are specified in row MSRF+1 of array CSEC. After these actions, control is returned to the main program.

Each type of cross sections is prepared in a similar fashion. In the case of one-dimensional data (NT or NY less than or equal to 8), only one argument of the array(s) SRF (and SRFI) is an independent variable the other argument serves as a label to allow distinction between up to eight one-dimensional datasets. Which one of these eight datasets is to be graphed is determined by the value of the real part of the element of CSEC under consideration. When NT and NY are larger than 8, then SRF and SRFI contain one two-dimensional function, a surface. Which of the two functional arguments is to be held constant for each cross sectional plots is determined by the imaginary part of its corresponding element of CSEC. Therefore, in 2-d cases the imaginary part of the current element of CSEC is tested. If it is 2.0 a horizontal cross section (second variable of array(s) SRF (SRFI) fixed) follows; if it

is 1.0 a vertical cross section (first variable of array(s) SRF (SRFI) fixed) follows; otherwise the next element in the current row of CSEC will be considered in the same way. A selected plot starts by writing its headline and axis labels onto a new graphics frame. Then the data of the sectional curve are computed, the coordinate system is sized accordingly and then drawn. Finally the cross sectional curve is itself drawn. If the plot displays FFT-data the drawing of the FFT-axis that would be drawn as part of the coordinate system (CALL GRAF) will be suppressed in order to avoid the tick mark labels which generally exhibit "messy looking" numbers. This axis of the coordinate system is suppressed. Instead of it a "secondary" (DISSPLA nomenclature) axis will be drawn immediately after the cross sectional curve is drawn. This secondary axis exhibits tick marks with "nice" values as determined by the subroutine NYSXIS.

The cross sectional curves are the functional values of the field data arrays at the grid point ISEC that is the closest to the locations specified by the real part of the current element of CSEC. While the data of the intensity and amplitude can readily be plotted as they are available in the array(s) SRF (SRFI), the data for the phase sections have to be calculated first by this subroutine.

The plotted phase data are calculated as follows: The field magnitude at the fixed grid point ISEC is computed. If its maximum is less than 10^{-30} the field information is deemed unreliable and no phase curve will be drawn. Furthermore all locations where the magnitude is less than the maximum magnitude divided by 10^8 are deemed unreliable and no phase curve points are shown. The arctangent of the ratio of the imaginary to real field amplitudes provides the raw phase data. It is assumed that the numerical resolution of RAM2D1 is sufficient to provide raw phase data that do not vary by more than $\pm\pi$ from grid point to grid point. The first raw data point is placed within $\pm\pi$ of zero phase. All consecutive raw data points are tested if they were reached by a phase change that implies a crossing of

the negative real axis of the amplitude vector in which case 2π will be added or subtracted to all following phase points depending on an implied phase wind-up or wind-down. By this method phase variations crossing multiple 2π -intervals can be followed. In case of intermittent unreliable data points the next reliable phase is placed within the 2π -interval of the previous reliable phase point.

The subroutine NYSXIS finds "nice" end values and interval step sizes (used for customized axis labeling) outside of the range that is specified by a choice of two from the following four values in the subroutine arguments: the maximum value in the vector VEC, the minimum value in VEC, VECBOT, and VECTOP. The decision which quantities constitute the reference interval depends on the value of the argument NECLEC.

If NECLEC = -1: VECBOT and VECTOP are chosen and the vector VEC is neglected.

= 0: then the maximum and minimum of all four are chosen

= 1: then the extrema of VEC are chosen and VECBOT and VECTOP are neglected.

It is also possible to "hard-wire" the lower (upper) end-value to the current value of VECBOT (VECTOP) by setting the argument VECGAP to -1.0 (1.0) as input. If VECGAP = 2.0 on input both end values are "hard-wired."

The subroutine NYSXIS finds the extrema of the input data. Then it determines the largest integral power of ten (XTRPOW) that is still smaller than the larger of the absolute values of the extrema. Based on XTRPOW the leading two decimal places of the extrema are compared with each other. The possible difference is placed into one of seven interval classes with the following interval sizes: 0.005, 0.05, 0.1, 0.2, 0.5, 1.0, 2.0 times XTRPOW. The end values that will be returned are chosen to be one interval beyond the integer that is closest to the original extrema. If the hard-wiring option was chosen the hard-wired end value is re-instated before the interval and end values are returned to the calling routine.

III. SCIENTIFIC STUDIES

III.A. Transient Raman Interactions

The study of transient Raman effects has been an important focus of scientific activity since the original papers of Wang³ and of Carmen, *et al.*⁴ Recent experiments at the Naval Research Laboratory⁵ have reinvigorated basic research in this area and opened up new questions relating to the evolution of pulses in this regime. The basic equations are

$$\begin{aligned}\frac{\partial E_L}{\partial z} &= -i \frac{k_L}{k_S} \kappa_2 Q E_S, \\ \frac{\partial E_S}{\partial z} &= -i \kappa_2 Q^* E_L, \\ \frac{\partial Q}{\partial t} + \Gamma Q &= -i \kappa_1 E_S^* E_L,\end{aligned}\tag{3.1}$$

where E_L and E_S are the pump and Stokes fields, z and t are axial distance along the Raman interaction cell and time with z -dependent origin, k_L and k_S are the pump and Stokes wavenumbers, and κ_1 and κ_2 are Raman coefficients.

We have carried out analytical and numerical calculations, both to gain insight into behavior that has been observed experimentally at NRL and to predict the behavior that would be observed in regimes not yet accessed by the experiments. Virtually all the analytical work is summarized in the preprint "Asymptotic evolution of transient pulses undergoing stimulated Raman scattering," which is included in Appendix C of this report and will not be repeated in detail here. Basically, we have shown that if the amplitude of the initial Stokes is small compared to the amplitude of the initial pump, then the pulse evolution passes through two main regimes. Initially, the Stokes grows exponentially while the pump is essentially undepleted. During this growth, the phase of the Stokes pulse locks onto the pump phase. This regime was studied by Carmen, *et al.*⁴ using simple linear theory. We call it the *I*-regime. This regime is followed by a transition regime in which pump depletion becomes significant. Finally, there is the *J*-regime in which the Stokes intensity remain

almost constant while the pump slowly depletes.

We now turn to our computational studies, considering first the J -regime. In Figs. 3.1–3.2, we show the variation of

$$R = \left[\int_{-\infty}^{\infty} dt |E_L|^2(0) / \int_{-\infty}^{\infty} dt |E_L|^2(\zeta) \right]^2$$

vs. $\zeta = \kappa_1 \kappa_2 z \int_{-\infty}^{\infty} K(t) dt$, where

$$K(t) = |E_L|^2(\zeta, t) + \frac{k_L}{k_S} |E_S|^2(\zeta, t). \quad (3.2)$$

The theory indicates that R should vary linearly with ζ at sufficiently large ζ . This trend is observed in Fig. 3.1. Moreover, linear behavior is observed when $R \gtrsim 10$ which corresponds to approximately 70% depletion of the pump. In Fig. 3.2, we show on a parabolic scale vs. ζ the number of zero-crossings N of the pump amplitude. Theory indicates that N^2 should be proportional to ζ . This result is confirmed in Fig. 3.2. We note that in all cases the expected asymptotic behavior is observed for $\zeta \gtrsim 120$, and the original Stokes has an intensity 0.001 that of the pump.

In Fig. (3.3) and (3.4), we show the effect of varying the Stokes offset for pulses with an initial sech^2 amplitude and a FWHM of 40 ps. At negative offsets there is a tendency for depletion to be delayed while the number of zero-crossings increases linearly beyond a relatively short distance. When $t_{o\sigma} = -40$ ps, the pump must be 90% depleted before linear variation of R is observed. At $t_{o\sigma} = -20$ ps, one finds that R begins to scale linearly when the pump is 85% depleted. At $t_{o\sigma} \geq 0$ ps, this requirement reduces to 70% depletion. In all cases, linear behavior is observed to set in when $\zeta = 100$ –200, with the highest values occurring at the most negative offsets. Conversely, when $t_{o\sigma} > 0$, there is a tendency for the oscillations of the pump amplitude to be delayed. If we add the effect of chirp onto the pump pulses, there is little effect until the chirp becomes quite sizable. With a phase

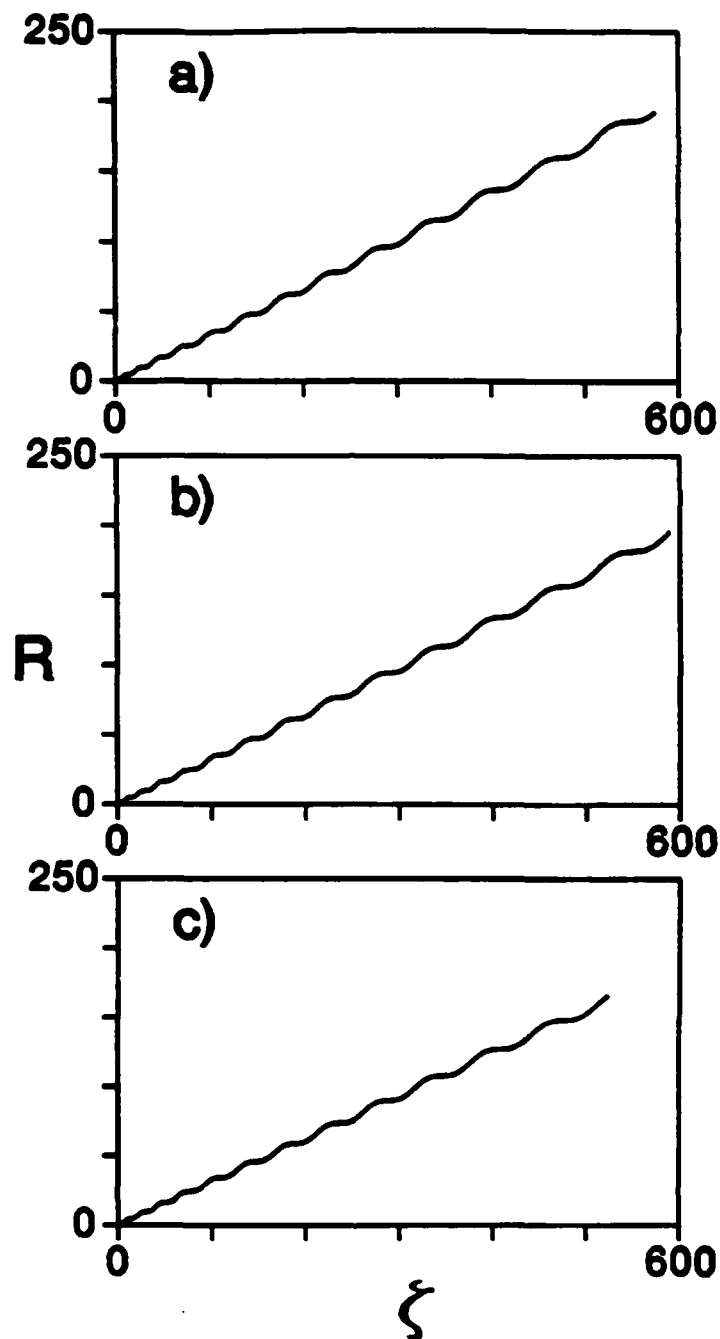


FIGURE 3.1. Plots of R vs. ζ for different pulse shapes. a) sech-squared amplitude, FWHM = 40 ps; b) Lorentzian-squared amplitude, FWHM = 39 ps; c) Square pulse, FWHM = 43.8 ps.

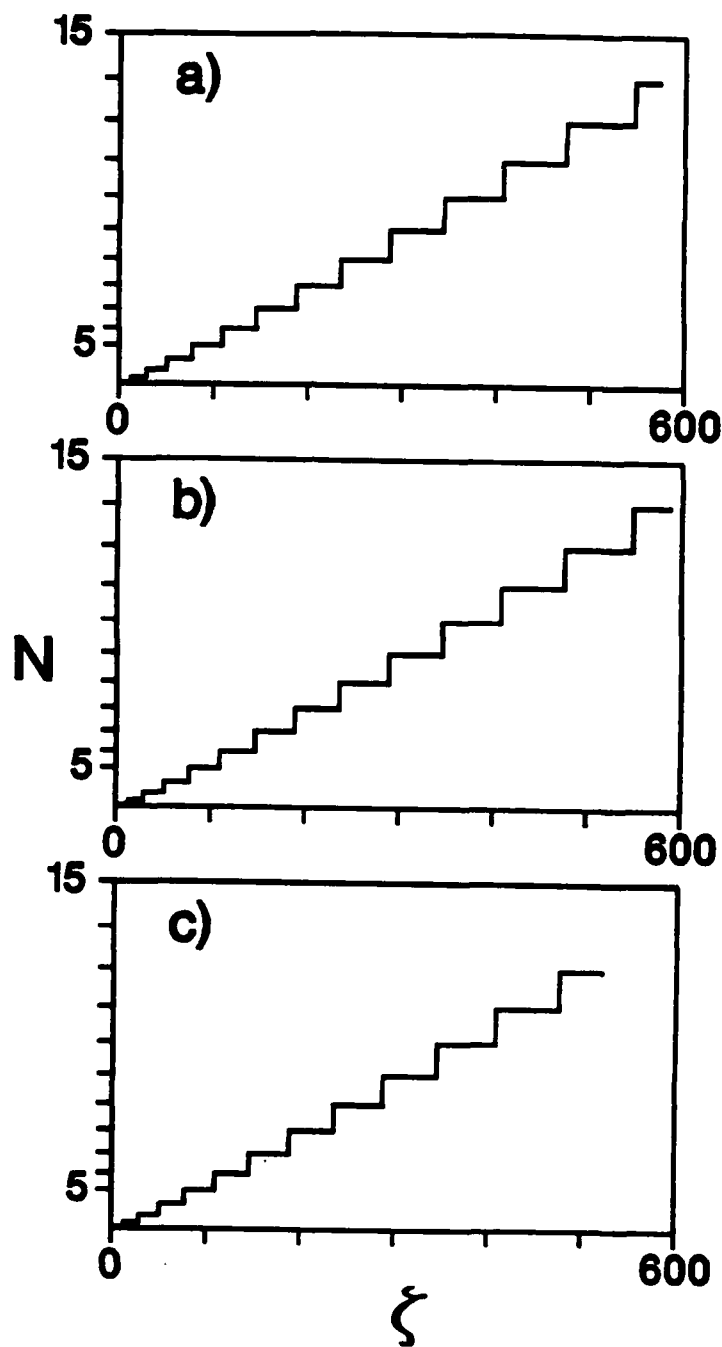


FIGURE 3.2. Plots of N vs. ζ ; N is plotted on a parabolic scale. Shapes and parameters are the same as in Fig. 3.1.

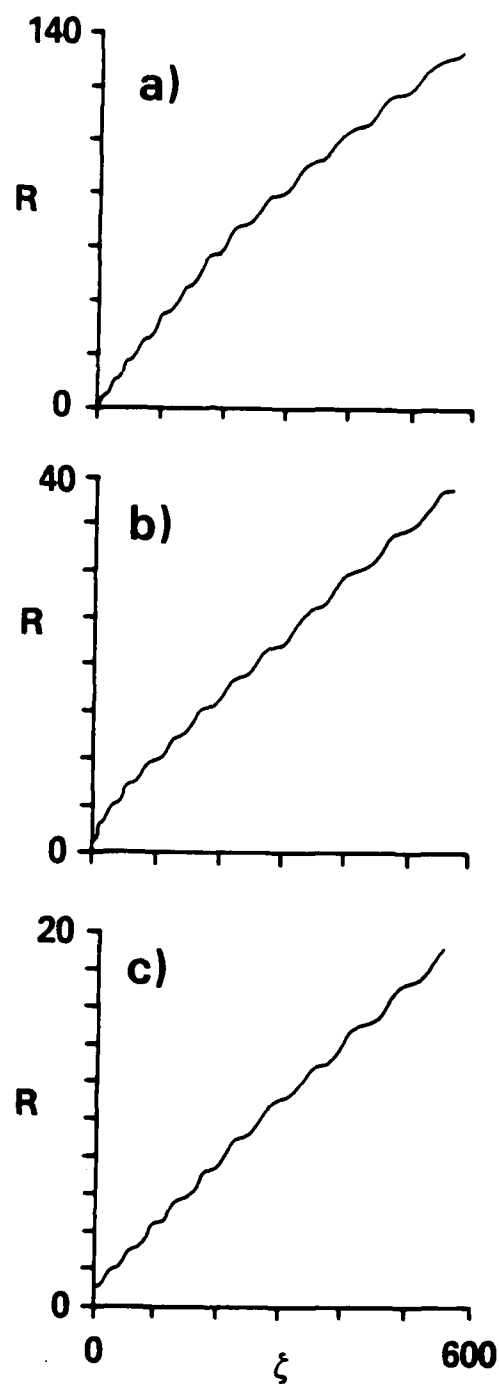


FIGURE 3.3. Effect of Stokes offset on the scaling of R with ζ . In all cases the pulses have a sech^2 amplitude profile and a FWHM of 40 ps. a) $t_{\text{off}} = -20$ ps; b) $t_{\text{off}} = 0$ ps; c) $t_{\text{off}} = 20$ ps.

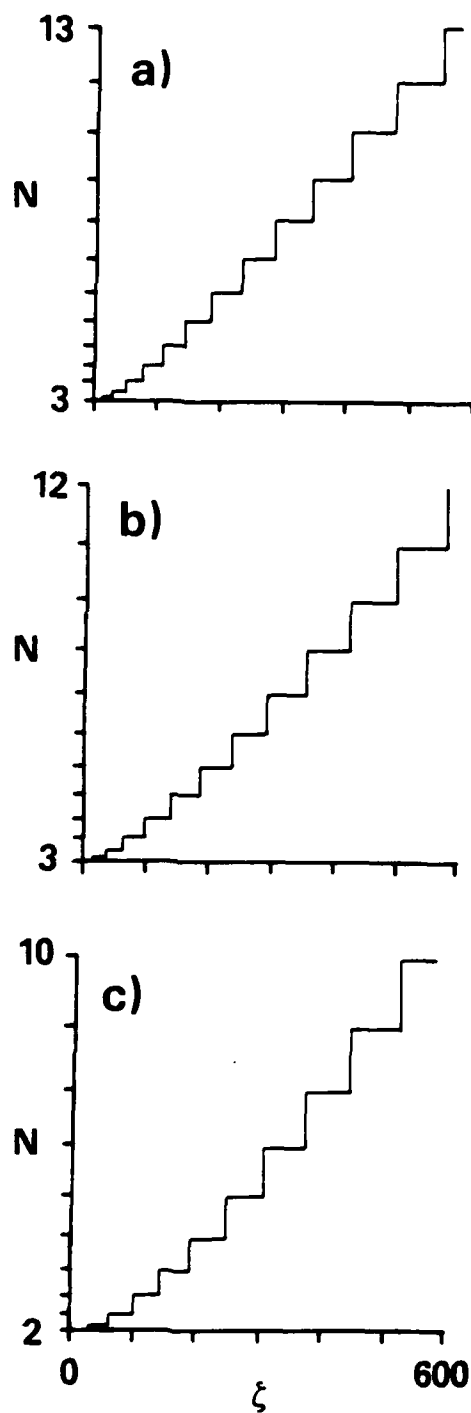


FIGURE 3.4 Effect of Stokes offset on the scaling of N with ζ . Parameters are the same as in Figure 3.3.

variation of 9.5π , we find an increase by under 50 in the ζ -values at which linear behavior of R sets in. Otherwise, the qualitative behavior remains the same.

Turning now to consideration of the I -regime, we consider the effect of the Stokes pulse offset on its gain over a fixed distance (40 cm) and its ability to phase lock to the pump. For a symmetric pulse with an initial sech^2 amplitude, a 40 ps FWHM, an initial maximum pump intensity of 1.0 Gwatts/cm², and an initial Stokes intensity 0.001 the pump intensity, we list the dependence of gain and locking on offset in Table III.1. The chirp referred to is approximately π , which is the experimental magnitude. In Fig. 3.5, we show the phases at $z = 40$ cm for three values of $t_{0\pi}$. Phase locking is complete at $t_{0\pi} = -20$ ps and $t_{0\pi} = 0$ cm but is incomplete at $t_{0\pi} = 0$ cm.

Finally, we have carried out numerical calculations aimed at understanding the rapid phase flip which is observed experimentally to travel from the back to the front of the pulse in the I -regime. We generally observe from our numerical studies that a fast phase flip can be obtained at a particular gain for a fixed chirp. As we raise the gain, this fast phase flip disappears and phase locking occurs in contrast to the experimental observations. It appears at this point that we will have to take into account diffractive effects in order to have any hope of explaining the experimental observations, and we will carry out those studies shortly.

III.B. Beam Interactions in the Stationary Limit

In this section we outline a number of analytical calculations which we have made in the stationary limit to clarify the effect of aberrations and a finite interaction length on the interaction of pump beams with a Stokes beam in both collinear and crossing beam geometries.

In studying a multiple beam geometry, we may assume that the pump beam has the

TABLE III.1

Variation of Gain and Locking With Offset

	No Chirp	Chirp	
Offset	Gain	Gain	Locking
-100	1.0	1.0	locked
-75	2.5	1.25	locked
-50	15	5.8	locked
-25	42	15	locked
0	43	23	locked
25	17	11	partially locked
50	2.7	1.4	none
75	1.1	1.1	none

Stokes Phase

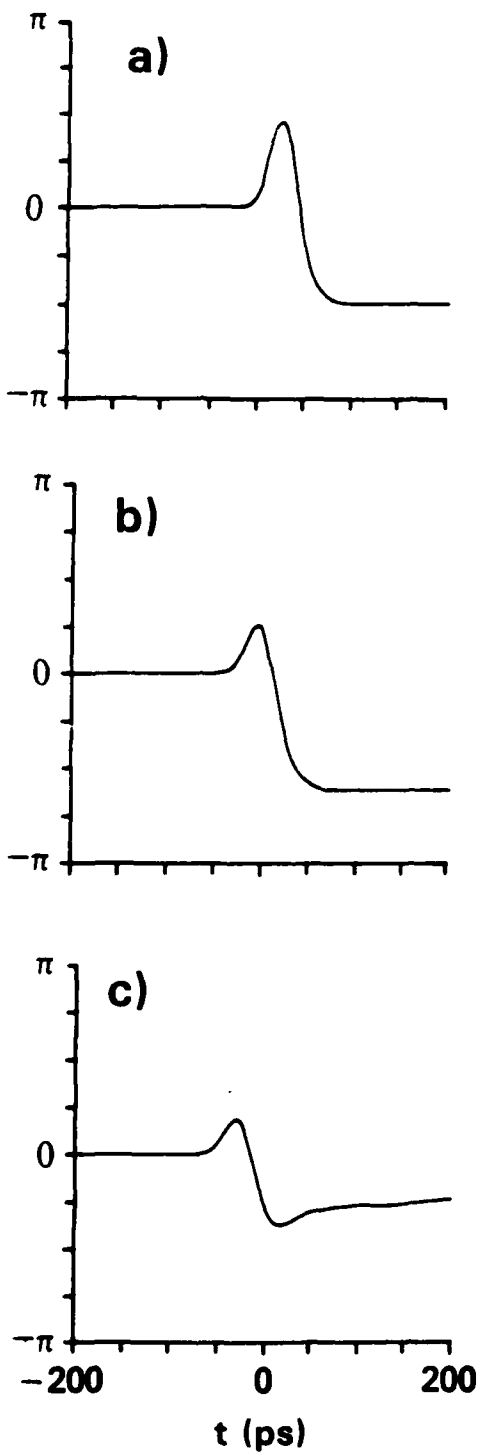


FIGURE 3.5. Phase-locking of a pulse with sech^2 amplitude is shown at three different offsets. a) $t_{0\pi} = -20$ ps; b) $t_{0\pi} = 0$ ps; c) $t_{0\pi} = 20$ ps.

form

$$E_L = \sum_{n=-N}^N f_n \left[y + (ny_0/z_0)z - ny_0, z \right] \exp \left\{ -ik_L(ny_0/z_0) \left[y + \frac{1}{2}(ny_0/z_0)z \right] \right\} , \quad (3.3)$$

where the f_n give the shapes of the individual beams. The first argument gives the rapid transverse variation. The second argument gives the slow z -variation due to diffraction. The quantity y_0 gives the intrinsic separation between the beams when $z = 0$ while z_0 is the z -value at which all the beams converge. Noting that

$$\tilde{X}(k) = \frac{1}{2\pi} \int_{-\infty}^{\infty} X(y) \exp(-iky) dy , \quad (3.4)$$

we find

$$\begin{aligned} \tilde{E}_L(k) = \sum_{n=-N}^N \tilde{f}_n[k = k_L(ny_0/z_0), z] \\ \exp \left\{ i[k + k_L(ny_0/z_0)] [(ny_0/2z_0)z - ny_0] \right\} . \end{aligned} \quad (3.5)$$

In the Fourier domain, $\tilde{E}_L$ consists of a set of offset peaks. In all experiments, the width of these peaks $(\Delta K)_{\text{beam}}$ can be assumed small compared to the fundamental separation between the peaks $(\Delta K)_{\text{sep}} = y_0/z_0$. When the beams are well-separated in the coordinate domain, they have rapid amplitude modulations in the Fourier domain; when the beams are nearly overlapping, these amplitude modulations are slow.

We note that the condition $(\Delta K)_{\text{beam}} \ll (\Delta K)_{\text{sep}}$ results in the nonlinear terms combining like convolutions. The basic equations reduce in the stationary limit to

$$\begin{aligned} \frac{\partial E_L}{\partial z} - \frac{i}{2k_L} \frac{\partial^2 E_L}{\partial y^2} &= -\frac{g}{2} \frac{k_L}{k_S} |E_S|^2 E_L , \\ \frac{\partial E_S}{\partial z} - \frac{i}{2k_S} \frac{\partial^2 E_S}{\partial y^2} &= \frac{g}{2} |E_L|^2 E_S . \end{aligned} \quad (3.6)$$

We now find, letting $y_n = y + (ny_0/z_0)z - ny_0$, $z_n = z$, and

$$E_S = \sum_{n=-N}^N g_n(y_n, z_n) \exp \left\{ -ik_S(ny_0/z_0) \left[y + \frac{1}{2}(ny_0/z_0)z \right] \right\} , \quad (3.7)$$

that

$$\frac{\partial g_n}{\partial z_n} = \sum_{k=-N}^N \sum_{l=-N}^N \sum_{m=-N}^N f_k(y_n, z_n) f_l^*(y_n, z_n) g_m(y_n, z_n) \cdot \exp \left\{ -ik_L \left[\left(\frac{ky_0}{z_0} \right)^2 - \left(\frac{ly_0}{z_0} \right)^2 \right] z - ik_S \left[\left(\frac{my_0}{z_0} \right)^2 - \left(\frac{ny_0}{z_0} \right)^2 \right] z \right\} , \quad (3.8)$$

where we impose the condition $k - l + m - n = 0$. Terms which do not satisfy this condition are too rapidly varying in y to be consistent with the condition $(\Delta K)_{\text{sep}} \ll (\Delta K)_{\text{beam}}$.

In general, the exponential factor in Eq. (3.8) is also rapidly varying. Exceptions are when $k^2 = l^2$ and $m^2 = n^2$, in which case this factor disappears. It turns out that only these cases make non-zero contributions to dg_n/dz_n . This issue is discussed in great detail in the paper "Pump replication in stimulated Raman scattering using a crossed-beam geometry" which we presented at the 1988 SPIE meeting in Los Angeles in session 874. This paper is included in Appendix C, and we do not present the details here. This paper has a discussion of the effect of pump aberrations on side beam replication which we also do not repeat.

At this point we discuss the effect of geometry on Stokes beam amplification in the low Fresnel number regime where diffraction can be ignored. We consider as a simple example two crossing beams with Gaussian profiles interacting with a single Stokes. Thus, $k = -l = \pm 1$ and $m = n = 0$ in Eq. (3.8). Writing

$$\begin{aligned} f_1 &= \frac{E_0}{(2\pi)^{1/2} w} \exp \left\{ -[y - (y_0/z_0)z + y_0]^2 / 2w^2 \right\} , \\ f_{-1} &= \frac{E_0}{(2\pi)^{1/2} w} \exp \left\{ -[y + (y_0/z_0)z - y_0]^2 / 2w^2 \right\} , \end{aligned} \quad (3.9)$$

we conclude that

$$\begin{aligned} \frac{dg_0}{dz} &= \frac{E_0^2}{2\pi w^2} \left(\exp \left\{ -[y - (y_0/z_0)z + y_0]^2 / w^2 \right\} \right. \\ &\quad \left. + \exp \left\{ -[y + (y_0/z_0)z - y_0]^2 / w^2 \right\} \right) g_0 . \end{aligned} \quad (3.10)$$

Integrating Eq. (3.10), we find

$$g_0(y, z) = g_0(y, 0) \exp \left\{ \frac{z_0 g E_0^2}{8\sqrt{\pi} y_0 w^2} \operatorname{erf} \left(\frac{y_0 z}{z_0 w} + \frac{y - y_0}{w} \right) + \operatorname{erf} \left(\frac{y_0 z}{z_0 w} - \frac{y + y_0}{w} \right) - \operatorname{erf} \left(\frac{y - y_0}{w} \right) + \operatorname{erf} \left(\frac{y + y_0}{w} \right) \right\} . \quad (3.11)$$

The maximum gain which is achieved in the limit $y_0 \gg w$ and $z \gtrsim 2z_0$ is

$$g_0(y, z) = g_0(y, 0) \exp \left(\frac{g z_0 E_0^2}{2\sqrt{\pi} y_0 w^2} \right) . \quad (3.12)$$

The gain saturates because the pump and Stokes beams only interact over a limited length.

When a pump beam is aberrated, its linear propagation is strongly affected. Suppose we consider a Gaussian beam with phase aberrations

$$E_L(z=0) = \frac{E_0}{\sqrt{2\pi} w} \exp(-y^2/2w) \exp[i\varphi(y)] , \quad (3.13)$$

where $\varphi(y)$ is a randomly varying phase. Then E_L for $z > 0$ can be determined by solving the linear equation

$$\frac{\partial E_L}{\partial z} - \frac{i}{2k_L} \frac{\partial^2 E_L}{\partial y^2} = 0 . \quad (3.14)$$

We find that

$$|E_L(z, y)|^2 = \frac{1}{(2\pi)^2} \int_{-\infty}^{\infty} dk \int_{-\infty}^{\infty} dk' \int_{-\infty}^{\infty} dy' \int_{-\infty}^{\infty} dy'' \left(\frac{E_0^2}{2\pi w^2} \right) \exp[i(k - k')y] \exp[-iky' + ik'y''] \exp \left\{ -[(y')^2 + (y'')^2]/2w^2 \right\} \exp \{ i[\varphi(y') - \varphi(y'')] \} \exp \left[-i \left(\frac{k^2}{2k_L} - \frac{(k')^2}{2k_L} \right) z \right] . \quad (3.15)$$

If we assume a Gaussian autocorrelation length,

$$\langle \exp \{ i[\varphi(y') - \varphi(y'')] \} \rangle = \exp [-(y' - y'')^2/2l^2] , \quad (3.16)$$

we conclude

$$\langle |E_L|^2 \rangle = \frac{E_0^2}{2\pi w^2} \frac{w}{\left[w^2 + \left(\frac{l^2 + 2l^2 w^2}{l^4 w^2} \right) \frac{z^2}{k_L^2} \right]^{1/2}} \exp \left(- \frac{y^2}{\left[w^2 + \left(\frac{l^2 + 2l^2 w^2}{l^4 w^2} \right) \frac{z^2}{k_L^2} \right]} \right) . \quad (3.17)$$

In the limit where $l \gg w$, we see that Eq. (3.17) goes over to the standard result where the pulse spreads to $\sqrt{2}$ times its width over a length $z = k_L w^2$. In the limit where $w \gg l$, so that the beam is highly aberrated, the pulse spreads to $\sqrt{2}$ times its original length over a distance $z = k_L w l / \sqrt{2}$.

The phase aberrations rapidly translate into amplitude aberrations so that the intensity fluctuates as it spreads. To determine the size of these fluctuations, one must in principal calculate

$$\langle |E_L|^4 \rangle - (\langle |E_L|^2 \rangle)^2 .$$

A detailed calculation of this quantity is rather messy, but intuitively we expect that

$\langle |E_L|^4 \rangle \simeq (\langle |E_L|^2 \rangle)^2$ at a distance short compared to the aberration Fresnel length, $d_F = l^2 k_L$ and $\langle |E_L|^4 \rangle \simeq 2(\langle |E_L|^2 \rangle)^2$ at distances long compared to d_F . Roughly speaking, we can consider E_L viewed as a function of y to vary on a length scale l . If $w = nl$, where n is some integer, then the original beam has n independent emitters, E_i , $i = 1, n$. At a distance $z = Nl$, the number of emitters which contributes is roughly $N = z/l$ if $N < n$ or n otherwise. We now find

$$E_L = \sum_{i=1}^N E_i \quad (3.18)$$

$$\Rightarrow \langle |E_L(z)|^2 \rangle = \sum_{i=1}^N \langle |E_i(z)|^2 \rangle = N \langle |E_i(z)|^2 \rangle ,$$

where we assume that the expectation for each individual emitter is the same. Writing now

$$|E_L|^4 = \left(\sum_{i=1}^N \mathbf{E}_i \right) \cdot \left(\sum_{j=1}^N \mathbf{E}_j^* \right) \left(\sum_{k=1}^N \mathbf{E}_k \right) \cdot \left(\sum_{l=1}^N \mathbf{E}_l^* \right) , \quad (3.19)$$

we conclude

$$\begin{aligned} \langle |E_L|^4 \rangle &= 2 \left(\sum_{i=1}^N \mathbf{E}_i \cdot \mathbf{E}_i^* \right) \left(\sum_{k=1}^N \mathbf{E}_k \cdot \mathbf{E}_k^* \right) - \sum_{i=1}^N (\mathbf{E}_i \cdot \mathbf{E}_i^*)^2 \\ &= 2(N^2 - N) \langle |\mathbf{E}_i|^2 \rangle^2 \simeq 2 \langle |E_L|^2 \rangle^2 \end{aligned} \quad (3.20)$$

when N is large. To pin down the connection with the Fresnel length, we note that when z is small

$$E_L(z) - E_L(0) = \frac{i}{2k_L} \frac{\partial^2 E_L}{\partial y^2} \Big|_{z=0} z \quad (3.21)$$

where the second derivative is evaluated at $z = 0$. Using Eq. (3.13), we now find

$$|E_L(z)|^2 = |E_L(0)|^2 - \frac{\varphi''(y)}{2k_L} z |E_L(0)|^2 . \quad (3.22)$$

Noting that $|\varphi''(y)| \sim 1/l^2$, we conclude that the amplitude aberrations appear over a length d_F .

Once amplitude aberrations appear, they can have a deleterious affect on collinear beam amplification, particularly in the high gain regime. When $gE_0^2 d_F / 4\pi w^2 \gg 1$ and $w \gg l$, then we are in the high gain, highly aberrated regime. We may ignore to lowest order the effect of diffraction on the Stokes beam amplification. In the central part of the beam where $y \ll w$, we may write

$$\begin{aligned} \frac{dE_S}{dz} &= \frac{gE_0^2}{4\pi w^2} \exp(-y^2/w^2) a(y) E_S \\ &\simeq \frac{gE_0^2}{4\pi w^2} a(y) E_S , \end{aligned} \quad (3.23)$$

where $a(y) = |E_L|^2 / \langle |E_L|^2 \rangle$. The quantity $a(y)$ is Gaussian-distributed and $\langle a(y) \rangle = 1$. Specifically,

$$f(a) = \frac{\pi}{2} a \exp(-\pi a^2/4) \quad (3.24)$$

gives the probability distribution function of a . We then find

$$\frac{\langle |E_S|^2(z) \rangle}{\langle |E_S|^2(0) \rangle} \simeq \frac{gE_0^2 z}{2\pi w^2} \exp\left(\frac{g^2 E_0^4 z^2}{16\pi^3 w^4}\right) \quad (III.25)$$

at large z . In effect, the amplitude aberrations in the pump lead to high amplitude spikes which in turn leads to differential growth in the Stokes and substantial spikiness. We have not carried out a calculation in which we determine the increase in the Stokes bandwidth, but it is clear that the Stokes can become more aberrated than the original pump, a case sometimes observed in practice.⁶

In the future, we intend to carry out a series of numerical studies using RAM2D1, aimed at verifying some of these theoretical results in the stationary regime.

III.C. Solitons and the Spectral Transformation

If we consider the usual transient equations, Eq. (II.A.1), in the limit of pulses very short compared to T_2 so that we may set $\Gamma = 0$, we obtain the following solitary wave solution

$$\begin{aligned} E_L &= a \operatorname{sech}(\alpha z - \beta t) \ , \\ E_S &= \sqrt{\frac{k_S}{k_L}} \kappa_1 a \tanh(\alpha z - \beta t) \ , \\ Q &= -\sqrt{\frac{k_S}{k_L}} \kappa_1 \frac{a^2}{\beta} \operatorname{sech}(\alpha z - \beta t) \ , \end{aligned} \quad (3.26)$$

where $\beta = \kappa_1 \kappa_2 a^2 / \alpha$ which implies that the pulse is sub-luminous. This pulse has the remarkable property that $|E_S|$ tends to a non-zero value as $t \rightarrow \pm\infty$. Unfortunately, this property is not physical in the limit of short pulses. Nonetheless, it can be effectively true in situations where pulses are initially long compared to T_2 , and the Stokes pulse undergoes a rapid phase flip at some point. It is in this context that solitons have been observed experimentally.^{7,8} Indeed, there are theoretical considerations which indicate that dissipation plays an important role in the soliton's formation.^{9,10} Virtually all theoretical

work to date has focussed on the case where the original Stokes pulse has a 180° phase flip. It is of some interest to determine how close to 180° the phase flip must be before a soliton, or more precisely a soliton-like structure, will form. We have considered this issue and intended to report on it at the July '88 IQEC meeting in Tokyo, Japan. (The high cost of travel to Japan has prevented us from attending.) The summary for this meeting is included in Appendix D, and we do not repeat the details here. We conclude that if

$$E_S = K\Gamma_S t + iK_S \quad (3.27)$$

in the neighborhood of the phase flip, then a soliton will form if

$$\frac{\Gamma}{\Gamma_S} \frac{K_S}{K} < 1 . \quad (3.28)$$

Another issue of some importance is the possible generation of a series of solitons when a series of pump pulses is injected into a Raman cell. The evolution of a series of pulses has been considered by Reintjes, *et al.*¹¹ and it is of some interest to determine whether a series of solitons can emerge from these pulses. We have not considered this issue in any detail, but it is of some interest to point out that the transient equations do have periodic solutions when $\Gamma = 0$. Typical solutions are

$$\begin{aligned} E_L &= a \operatorname{cn}(\alpha z - \beta t|m) , \\ E_S &= \sqrt{\frac{k_S}{k_L}} a \operatorname{sn}(\alpha z - \beta t|m) , \\ Q &= -\sqrt{\frac{k_S}{k_L}} \frac{a^2}{m\beta} \operatorname{dn}(\alpha z - \beta t|m) , \end{aligned} \quad (3.29)$$

where $\alpha\beta = \kappa_1\kappa_2 a^2/m$. Whether this solution is realizable in practice will be determined in future investigations.

We now turn to a discussion of the spectral transform method which applies in the limit of short, transient pulses. There are theoretical reasons to suspect that solitons in

these systems are always transient. On physical grounds, we might anticipate this result as the quantity

$$K = |E_L|^2 + \frac{k_L}{k_S} |E_S|^2 \quad (3.30)$$

is constant at every t -point while solitons are sub-luminous. Hence, we expect them to disappear at the back end of the pulse. We shall see that the spectral transform method has peculiarities in our case which result in solutions of very different character from those which are normally found when spectral methods can be used.

We first make a change of variables so that our notation follows that normally used in this field.^{9,12,13} We let $A_1 = E_L$, $A_2 = (k_L/k_S)^{1/2} E_S$, $X = i(k_L/k_S)^{1/2} Q$, $\varepsilon = \Gamma/\kappa_1$, $\tau = \kappa_1 t$, and $\chi = \kappa_2 z$, yielding

$$\begin{aligned} \frac{\partial A_1}{\partial \chi} &= -X A_2 , \\ \frac{\partial A_L}{\partial \chi} &= X A_1 , \\ \frac{\partial X}{\partial \tau} + \varepsilon X &= A_1 A_2^* . \end{aligned} \quad (3.31)$$

We apply the spectral transform approach in the limit where we may set $\varepsilon = 0$. Following Kaup,⁹ we first consider two new quantities u_1 and u_2 which satisfy the equations

$$\begin{aligned} \frac{\partial u_1}{\partial \chi} - \frac{i}{\zeta} u_1 &= \chi u_2 , \\ \frac{\partial u_2}{\partial X} + \frac{i}{\zeta} u_2 &= -X^* u_1 , \end{aligned} \quad (3.32)$$

and

$$\begin{aligned} \frac{\partial u_1}{\partial \tau} + i\zeta S_3 u_1 &= \zeta S_+ u_2 , \\ \frac{\partial u_2}{\partial \tau} - i\zeta S_3 u_2 &= -\zeta S_- u_1 , \end{aligned} \quad (3.33)$$

where

$$\begin{aligned} S_3 &= \frac{1}{4} (A_1 A_1^* - A_2 A_2^*) , \\ S_+ &= \frac{i}{2} A_2^* A_1 , \\ S_- &= S_+^* . \end{aligned} \quad (3.34)$$

Equations (3.32) and (3.33) are *compatible*, i.e., their cross-derivatives are equal, only if Eq. (3.31) holds with $\varepsilon = 0$. At this point, we define the quantities

$$A = \frac{1}{4}(A_1 A_1^* + A_2 A_2^*) , \quad (3.35)$$

the angles β and θ through the relations

$$S_3 = A \cos \beta , \quad (3.36)$$

$$S_+ = A e^{i\theta} \sin \beta ,$$

and the angle γ through the compatibility relations

$$\begin{aligned} \frac{\partial \gamma}{\partial r} &= \cos \beta \frac{\partial \theta}{\partial r} , \\ \frac{\partial \gamma}{\partial \chi} &= \frac{2}{\sin \beta} (X_1 \cos \theta - X_2 \sin \theta) , \end{aligned} \quad (3.37)$$

where we have decomposed

$$X = X_1 - iX_2 . \quad (3.38)$$

We define as well the matrices,

$$\begin{aligned} \Gamma &= I \cos(\gamma/2) + i\sigma_3 \sin(\gamma/2) , \\ B &= I \cos(\beta/2) + i\sigma_1 \sin(\beta/2) , \\ \Theta &= I \cos(\theta/2) + i\sigma_3 \sin(\theta/2) , \end{aligned} \quad (3.39)$$

where σ_1 , σ_2 , and σ_3 are the Pauli matrices. Finally, we let

$$V = \begin{pmatrix} u_1 & \hat{u}_1 \\ u_2 & \hat{u}_2 \end{pmatrix} , \quad (3.40)$$

where (u_1, u_2) and $(\hat{u}_1, \hat{u}_2)$ are two independent solutions of Eq. (3.33). Making the transformation

$$V = \Gamma B \Theta^{-1} U , \quad (3.33)$$

we have verified after substantial algebra that V satisfies the equation

$$\left(I \frac{\partial}{\partial T} + i\zeta \sigma_3 \right) V = \begin{pmatrix} 0 & q \\ -q^* & 0 \end{pmatrix} V , \quad (3.42)$$

where

$$\begin{aligned} T &= \int_{-\infty}^r A dr' , \\ q &= \frac{i}{2 \cos \beta} \frac{\partial}{\partial T} [e^{i\gamma} \sin \beta] . \end{aligned} \quad (3.43)$$

Equation (3.42) has the standard form of the AKNS systems.¹⁴ We impose the boundary condition $V(r \rightarrow -\infty) = \sigma_3$ and let $V(r \rightarrow +\infty) = Y S^t \sigma_3$, where

$$Y = \begin{pmatrix} e^{-i\zeta T_\infty} & 0 \\ 0 & e^{i\zeta T_\infty} \end{pmatrix} , \quad S = \begin{pmatrix} a & b \\ -\bar{b} & \bar{a} \end{pmatrix} , \quad (3.44)$$

$a, \bar{a}, b, \bar{b}$ are the usual scattering coefficients, and $T_\infty = T(r = +\infty)$. We now find

$$\frac{\partial V}{\partial \chi} = \frac{i}{\zeta} \Gamma B \sigma_3 B^{-1} \Gamma^{-1} V - \frac{i}{\zeta} V \sigma_3 \Gamma_0 B_0 \sigma_3 B_0^{-1} \Gamma_0^{-1} \sigma_3 , \quad (3.45)$$

where the subscripted matrices, B_0 and Γ_0 , are B and Γ in the limit $r \rightarrow -\infty$. Once the evolution of V is known in the limit $r = \infty$, we can determine $a(\chi)$, $\bar{a}(\chi)$, $b(\chi)$, and $\bar{b}(\chi)$.

Where q is compact, i.e. T_∞ is finite, $a, \bar{a}, b$, and $\bar{b}$ are all analytic as a function of ζ . We now define

$$\begin{aligned} G(x) &= \frac{1}{2\pi} \int_C \frac{\bar{b}}{a} e^{-i\zeta x} d\zeta , \\ \bar{G}(x) &= \frac{1}{2\pi} \int_{\bar{C}} \frac{b}{\bar{a}} e^{i\zeta x} d\zeta , \end{aligned} \quad (3.46)$$

where the contour C goes over all the zeroes of a and $\bar{C}$ goes under all the zeroes of $\bar{a}$.

Solving the linear equations

$$\begin{aligned} \bar{L}(x, y) + \begin{pmatrix} 1 \\ 0 \end{pmatrix} G(x+y) - \int_{-\infty}^x L(x, s) G(s+y) ds &= 0 , \\ L(x, y) + \begin{pmatrix} 0 \\ 1 \end{pmatrix} \bar{G}(x+y) + \int_{-\infty}^x L(x, s) \bar{G}(s+y) ds &= 0 , \end{aligned} \quad (3.47)$$

we can find $q(x)$ using the relation

$$q(x) = 2L_1(x, x) . \quad (3.48)$$

In standard AKNS systems, the evolution of the scattering is quite simple, while the zeroes of a and $\bar{a}$ are fixed and correspond to solitons.¹⁴ In our problem, that is no longer the case

as $X \neq 0$ in general in the limit $\tau \rightarrow \infty$, and, hence, the evolution of the spectral data cannot be easily determined. Fortunately, Kaup¹² has shown that solving the equation

$$\frac{\partial V}{\partial \chi} = -\frac{i}{\zeta} V \sigma_3 \Gamma_0 B_0 \sigma_3 B_0^{-1} \Gamma_0 \sigma_3 , \quad (3.49)$$

will still yield the correct answer for q when the previously outlined procedure is followed. However, the spectral data obtained in this way is not true spectral data. The zeroes of a and $\bar{a}$ are not fixed and no longer correspond to solitons. To illustrate this point, we consider a simple example already studied by Duncan, *et al.*⁵ We suppose that the Stokes is initially a multiple of the pump. We then find that

$$q(\chi = 0) = 0 . \quad (3.50)$$

From Eq. (3.49), we find

$$\begin{aligned} a_\chi &= -\frac{i}{\zeta} [a \cos \beta_0 - i \bar{b} \sin \beta_0] , \\ \bar{b}_\chi &= -\frac{i}{\zeta} [i a \sin \beta_0 - \bar{b} \cos \beta_0] , \\ \bar{a}_\chi &= \frac{i}{\zeta} [\bar{a} \cos \beta_0 + i b \sin \beta_0] , \\ b_\chi &= -\frac{i}{\zeta} [i \bar{a} \sin \beta_0 + b \cos \beta_0] . \end{aligned} \quad (3.51)$$

Using Eq. (3.50), it now follows that

$$G(2T) = \frac{1}{2\pi} \int_C \frac{i \sin \beta_0 [e^{-ix/\zeta} - e^{ix/\zeta}]}{(1 - \cos \beta_0) e^{ix/\zeta} + (1 + \cos \beta_0) e^{-ix/\zeta}} e^{2isT} d\zeta . \quad (3.52)$$

When $\beta_0 \simeq 0$, the zeroes of the integrand lie in the upper half plane, there are an infinite number of them clustering about the essential singularity at $\zeta = 0$, and they explode outward as χ increases. We have yet to make a complete evaluation of Eq. (3.52), not to mention a determination of $L(x, y)$ and $\bar{L}(x, y)$. We may consider this problem in the future.

It is possible to show however that when χ is small, the usual linear result is reproduced. Expanding the integrand of Eq. (3.52) as a power series in β_0 , assuming that it is small, we

find

$$G(2T) \simeq \frac{\beta_0}{4\pi i} \int_{+} (1 - e^{2i\chi/\zeta}) e^{-2i\zeta T} d\zeta , \quad (3.53)$$

where the contour $+$ is a small, positive circle around $\zeta = 0$. Recalling the relation

$$\exp \left[\frac{1}{2} y(t + 1/t) \right] = \sum_{k=-\infty}^{\infty} t^k I_k(y) , \quad (3.54)$$

we find

$$G(2T) \simeq -\frac{i\beta_0}{2} \left(\frac{\chi}{T} \right)^{1/2} I_1 \left[4(\chi T)^{1/2} \right] . \quad (3.55)$$

Writing now

$$i \frac{d\beta}{dT} = q(T) = 2\bar{L}_1(T, T) \simeq -2G(2T) , \quad (3.56)$$

we finally conclude

$$\beta(T) = \beta_0 I_0 \left[4(\chi T)^{1/2} \right] , \quad (3.57)$$

a result which had earlier been obtained by Duncan *et al.*⁵ using more elementary methods.

REFERENCES

- 1 See, *e.g.*, T. R. Taha and M. Ablowitz, *J. Comp. Phys* **55**, 203 (1984).
- 2 See, *e.g.*, C. W. Gear, *Numerical Initial Value Problems in Ordinary Differential Equations*, (Prentice-Hall, Englewood Cliffs, NJ, 1971).
- 3 C. S. Wang, *Phys. Rev* **182**, 482 (1969).
- 4 R. L. Carmen, F. Shimuzu, C. S. Wang, and N. Bloembergen, *Phys. Rev A* **2**, 60 (1970).
- 5 M. D. Duncan, R. Mahon, L. L. Tankersley, and J. Reintjes, *JOSA B* **5**, 37 (1988).
- 6 Dr. J. Reintjes, private communication.
- 7 K. J. Druhl, R. G. Wenzel, and J. L. Carlsten, *Phys. Rev. Lett.* **51**, 1171 (1983).
- 8 R. G. Wenzel, J. L. Carlsten, and K. J. Druhl, *J. Stat. Phys.* **39**, 621 (1985).
- 9 D. J. Kaup, *Physica* **19D**, 125 (1986).
- 10 K. J. Druhl, J. L. Carlsten, and R. G. Wenzel, *J. Stat. Phys.* **39**, 615 (1985).
- 11 J. Reintjes, M. D. Duncan, G. Calame, L. L. Tankersley, and R. Mahon, *Opt. News* **39**, 101 (1987).
- 12 D. Kaup, *Physica* **6D**, 143 (1983).
- 13 H. Steudel, *Physica* **6D**, 155 (1983).
- 14 See, *e.g.*, M.J. Ablowitz and H. Segur, *Solitons and the Inverse Scattering Transform* (SIAM, Philadelphia, 1981).

APPENDIX A

Code Listings

Typical namelists and output are included in the manual (Appendix B).

RAM2D1 (version C)

```

1      PROGRAM RAM2D1C
2      C
3      C THIS IS VERSION C OF THE TRANSIENT RAMAN AMPLIFIER CODE
4      C R A M 2 D 1. THIS VERSION IS ADAPTED TO RUN ON THE CENTRAL
5      C COMPUTING FACILITY AT THE NAVAL RESEARCH LABORATORY.
6      C
7      C RAM2D1 WAS WRITTEN BY CURTIS R. MENYUK 11/86 TO SOLVE THE
8      C COUPLED RAMAN EQUATIONS WITH BOTH TRANSIENT AND DIFFRACTIVE
9      C PHENOMENA ACCOUNTED FOR. THE EQUATIONS ARE ADVANCED IN THE
10     C Z-DIRECTION. THE BEHAVIOR IN THE TIME "DIRECTION" AND THE
11     C TRANSVERSE (Y) DIRECTION ARE DETERMINED AT EACH Z-STEP. HENCE,
12     C THIS CODE IS 2+1D. THE TWO QUANTITIES THAT ARE ADVANCED AT EACH
13     C STEP ARE THE PUMP AND THE STOKES AMPLITUDES. IN ADDITION, THE
14     C MATERIAL EXCITATION MUST BE DETERMINED AT EACH STEP THROUGH A
15     C CONSTRAINT EQUATION.
16     C
17     C THE SYSTEM OF EQUATIONS IS SOLVED USING A SEMI-SPECTRAL APPROACH.
18     C THE Y-DERIVATIVES OF THE DYNAMICAL EQUATIONS (THE EQUATIONS
19     C GOVERNING THE PUMP AND STOKES WAVES) ARE DETERMINED IN KY-SPACE,
20     C AND THE NONLINEAR TERMS ARE DETERMINED IN Y-SPACE. THERE ARE NO
21     C T-DERIVATIVES IN THE DYNAMICAL EQUATIONS AND HENCE NO NEED TO USE
22     C OMEGA SPACE. THE CONSTRAINT EQUATION (THE EQUATION GOVERNING
23     C THE MATERIAL EXCITATION) DOES CONTAIN A T-DERIVATIVE BUT NO
24     C Y-DERIVATIVE.
25     C
26     C THE CONSTRAINT EQUATION IS SOLVED FOR Q(EL,ES), SUBJECT TO THE
27     C APPROPRIATE BOUNDARY CONDITION (THE MATERIAL EXCITATION IS ZERO
28     C WHEN T GOES TO MINUS INFINITY). IT IS SOLVED FOR IN ONE OF THREE
29     C WAYS, DEPENDING ON THE PARAMETER REGIME: 1) WHEN NT IS EIGHT OR
30     C LESS, A SET OF 1-D STATIONARY CASES IS RUN. 2) WHEN MAX(ABS(T/TTWO))
31     C < 10.0 AND NT > 8, A RUNNING SUM IS PERFORMED. IN THIS APPROACH, IT
32     C IS NOT NECESSARY THAT Q=0 AT TMAX. 3) WHEN MAX(ABS(T/TTWO)) > 10.0
33     C AND NT > 8, A FOURIER TRANSFORM APPROACH IS USED.
34     C
35     C THE DYNAMICAL EQUATIONS ARE ADVANCED IN Z USING A MIDPOINT EULER
36     C METHOD WITH ONE SPECIAL MODIFICATION - THE LINEAR PORTION OF EACH
37     C EQUATION IS ADVANCED IN SUCH A WAY THAT IT IS SOLVED EXACTLY TO
38     C WITHIN ROUNDOFF. IN THIS VERSION, THE STEP SIZE IS FIXED.
39     C
40     C
41     C TIME IS DIMENSIONED IN PICOSECONDS; DISTANCE IS DIMENSIONED IN
42     C CENTIMETERS; AND POWER IS DIMENSIONED IN GIGAWATTS. ALL OTHER
43     C QUANTITIES ARE CORRESPONDINGLY DIMENSIONED.
44     C
45     C
46     C *****
47     C MODIFICATION 4/87:
48     C THIS PROGRAM ASSUMES THAT WHEN NY IS EIGHT OR LESS, A SET OF 1-D
49     C TRANSIENT CASES WITH NO Y-VARIATION ARE BEING RUN.
50     C NOTE: ONE MUST SET ICOND=3 IN THIS CASE FOR THE PROGRAM TO
51     C INITIATE PROPERLY
52     C *****
53     C MODIFICATION 5/87:
54     C THIS PROGRAM ASSUMES THAT WHEN NT IS EIGHT OR LESS, A SET OF
55     C STATIONARY CASES WITH NO T-VARIATION ARE BEING RUN. IN THIS
56     C CASE CFFT2 IS CALLED TO CARRY OUT THE FOURIER TRANSFORMS
57     C SERIALY, RATHER THAN CARRYING THEM OUT IN PARALLEL AS IN THE
58     C 2-D CASE.
59     C NOTE: ONE MUST SET ICOND=4 IN THIS CASE FOR THE PROGRAM TO
60     C INITIATE PROPERLY
61     C *****
62     C MODIFICATION 9/87:
63     C THE DATA OUTPUT FILE NAME WAS CHANGED FROM 'FRAM' TO THE FOLLOWING:

```

RAM2D1 (version C)

```

64 C THE DATA FILE NAME'S FIRST CHARACTER (F) STANDS FOR THE OLD DATA
65 C FILE NAME 'FRAM'. THE SECOND CHARACTER INDICATES THE T-DIMENSION.
66 C THE THIRD THE Y-DIMENSION. THE DIMENSIONS ARE REPRESENTED BY THEIR
67 C NUMBER (1-8) IF LESS THAN 9. IF GREATER THAN 8 THE DIMENSIONS ARE
68 C ASSUMED TO BE INTEGRAL POWERS OF 2. THE N-TH POWER OF 2 IS
69 C REPRESENTED BY THE N-TH CHARACTER OF RLFBE. THE FOURTH THROUGH
70 C NINTH CHARACTER IN THE FILE NAME ENCODES THE MONTH, DAY, AND YEAR
71 C THE PROGRAM WAS STARTED. A TENTH THROUGH TWELFTH CHARACTER IS
72 C APPENDED, NUMBERING THE PARTIAL DATA FILES THAT ARE GENERATED
73 C WHEN THE PROGRAM RUNS TWO-DIMENSIONALLY (MAXIMALLY 999 NEW FILES).
74 C *****
75 C
76 C
77 C
78 C
79 C
80 C
81 C
82 C
83 C
84 C
85 C
86 C
87 C
88 C
89 C
90 C
91 C
92 C
93 C
94 C
95 C
96 C
97 C
98 C
99 C
100 C
101 C
102 C
103 C
104 C
105 C
106 C
107 C
108 C
109 C
110 C
111 C
112 C
113 C
114 C
115 C
116 C
117 C
118 C
119 C
120 C
121 C
122 C
123 C
124 C
125 C
126 C

```

—VARIABLES—

```

NY = NUMBER OF Y POINTS (MUST BE A POWER OF 2)
NT = NUMBER OF T POINTS (MUST BE A POWER OF 2)
NP = MAXIMUM NUMBER OF PUMP BEAMS
NPUMP = ACTUAL NUMBER OF PUMPS
YM = DELIMITING Y-VALUES (CM)
TM = DELIMITING T-VALUES (PS)
ZINT = BEAM INTERSECTION POINT (CM)
RKP = PUMP WAVENUMBER (CM**-1)
RKS = STOKES WAVENUMBER (CM**-1)
YOFF = Y-OFFSETS OF THE PUMP BEAMS (CM)
TOFF = T-OFFSETS OF THE PUMP BEAMS (PS)
YWIDTH = Y-WIDTHS OF THE PUMP BEAMS (CM)
TWIDTH = T-WIDTHS OF THE PUMP BEAMS (PS)
YOST = Y-OFFSET OF THE STOKES BEAM (CM)
TOST = T-OFFSET OF THE STOKES BEAM (PS)
YWST = Y-WIDTH OF THE STOKES BEAM (CM)
TWST = T-WIDTH OF THE STOKES BEAM (PS)
RINT = INTENSITY OF THE PUMP BEAMS (GW/CM**2)
RIST = INTENSITY OF THE STOKES BEAM (GW/CM**2)
RAMP = AMPLITUDE OF THE PUMP BEAMS [SQRT(PS*GW/CM**3)]
RIST = AMPLITUDE OF THE STOKES BEAM [SQRT(PS*GW/CM**3)]
RAMASM = AMPLITUDE OF THE STOKES ASSYMETRY
RALASM = STRETCH OF THE STOKES ASSYMETRY
NHYP = EXPONENT OF THE HYPERGAUSSIAN DISTRIBUTION IN THE
      Y-DIRECTION (MUST BE AN EVEN INTEGER)
PHL = FACTOR MULTIPLYING THE INITIAL PUMP CHIRP
PHST = FACTOR MULTIPLYING THE INITIAL STOKES CHIRP
TOC = CHIRP PULSE TIME OFFSET (PS)
TWC = CHIRP PULSE T-WIDTH [SET TO TWIDTH(1)] (PS)
YWC = CHIRP PULSE Y-WIDTH [SET TO YWIDTH(1)] (PS)
ICOND = TYPE OF INITIAL PUMP AND STOKES PROFILES
        = 1: DOUBLE-SECH PROFILE
        = 2: SECH**2-HYPERGAUSSIAN PROFILE
        = 3: 1-D TRANSIENT CASES (NO Y-VARIATION)
        = 4: STATIONARY CASE (TO T-VARIATION)
ZSTEP = STEP SIZE (CM)
ZH = ZSTEP/2 (CM)
ZFINAL = FINAL Z-VALUE (CM)
ZKEEP = Z-VALUE INCREMENT BETWEEN POINTS WHERE DATA IS
        STORED (CM)
NMAX = MAXIMUM NUMBER OF ALLOWED STEPS IN Z (MUST BE LESS THAN
        OR EQUAL TO NST)
TTWO = DAMPING TIME OF THE MATERIAL EXCITATION (PS)
GAIN = RAMAN GAIN FACTOR ASSUMING NO PUMP DEPLETION (CM/GW)
RKAP1 = KAPPA-1: NONLINEAR COEFFICIENT IN THE MATERIAL
        EXCITATION EQUATION [SQRT(CM**3/GW*PS)]
RKAP2 = KAPPA-2: NONLINEAR COEFFICIENT IN THE STOKES
        EQUATION [SQRT(CM**3/GW*PS)/CM*PS]
SPEED = SPEED OF LIGHT IN VACUUM (USED IN THIS CODE TO

```


RAM2D1 (version C)

```

127 C      APPROXIMATE THE SPEED OF LIGHT IN THE MATERIAL) (CM/PS)
128 C      EL = PUMP ARRAY (NT*NY)
129 C      ES = STOKES ARRAY (NT*NY)
130 C      Q = MATERIAL EXCITATION ARRAY (NT*NY)
131 C      AEL,AES,AQ = SORRESPONDING FOURIER ARRAYS
132 C      AW = STORAGE ARRAY FOR THE MIDPOINT EULER METHOD (NT*NY*4)
133 C      CW = WORKING ARRAY FOR THE Y-DIRECTION FFT (5*NY/2)
134 C      (LENGTH MODIFICATION MADE 5/87 TO ALLOW CFFT2 TO RUN)
135 C      CWQ = WORKING ARRAY FOR THE T-DIRECTION FFT (NT)
136 C      USED WITH METHOD 2 OF CONSTRAINT EQ. SOLN.
137 C      WQ1,WQ2 = WORKING ARRAYS FOR CONSTRAINT EQ. SOLN. WITH
138 C      METHOD 1 (NT). THEY ARE EQUIVALENCED WITH CWQ
139 C      TO CONSERVE SPACE
140 C      COMVEC = INVERSE KERNEL FOR THE MATERIAL EXCITATION EQUATION
141 C      WHEN METHOD 2 IS USED (NT)
142 C      CYVEC = KERNEL FOR THE SECOND ORDER Y-DERIVATIVE OPERATOR IN
143 C      THE DYNAMICAL EQUATIONS. THE FINITE LENGTH IS ACCOUNTED
144 C      FOR SO THAT IN THE LINEAR LIMIT THE PROPAGATOR IS EXACT.
145 C      TWO VECTORS ARE NEEDED (NY*2)
146 C      NWRT = NUMBER OF RECORD GROUPS IN UNIT 4
147 C
148 C      —VARIABLES, BOTH ALTERED AND NEW, 1-D TRANSIENT CASE—
149 C
150 C      NY = ACTUAL NUMBER OF CASES RUN
151 C      ITYPE = TYPE OF INITIAL PROFILE
152 C      = 1: SECH PROFILE
153 C      = 2: RECTANGULAR PROFILE
154 C      = 3: LORENTZIAN PROFILE
155 C      = 4: EXPONENTIAL PROFILE
156 C      RTYPE = POWER TO WHICH PROFILE IS TAKEN (ITYPE = 1 & 3)
157 C      = POWER TO WHICH EXPONENT IS TAKEN (ITYPE = 4)
158 C
159 C      —VARIABLES, BOTH ALTERED AND NEW, STATIONARY CASE—
160 C
161 C      NT = ACTUAL NUMBER OF CASES RUN
162 C      AW1,AW2 = WORKING ARRAYS USED BY CFFT2 (NY)
163 C      RABAMP = FRACTIONAL CONTRIBUTION OF THE AMPLITUDE ABERRATIONS
164 C      RDSLIM = NUMBER OF TIMES DISPERSION LIMITED THE PUMP BEAMS ARE
165 C      DUE TO ABERRATIONS
166 C      [SET WITH RESPECT TO YWIDTH(1)]
167 C
168 C
169 C
170 C
171 C      PARAMETER(NT=256,NY=256,NTHP=1+NT/2,NP=10,NPM2=NP-2,NST=4000,
172 C      1 NS=5*NY/2)
173 C
174 C      IMPLICIT COMPLEX(A-E,Q)
175 C      DIMENSION EL(NT,NY),ES(NT,NY),Q(NT,NY),AEL(NT,NY),AES(NT,NY),
176 C      1 AQ(NT,NY),AW(NT,NY,4),CW(NS),AW1(NY),AW2(NY),CYVEC(NY,2),
177 C      2 COMVEC(NT),CWQ(NT),WQ1(NT),WQ2(NT),YWIDTH(NP),TWIDTH(NP),
178 C      3 YOFF(NP),TOFF(NP),RAMP(NP),RINT(NP),PHL(NP),ITYPE(8),RTYPE(8),
179 C      4 RABAMP(8),RDSLIM(8),SFE(NST),SQ(NST),SSTEP(NST),YM(2),TM(2)
180 C      CHARACTER*1 D1
181 C      CHARACTER*2 D1A
182 C      CHARACTER*2 D2
183 C      CHARACTER*2 D3
184 C      CHARACTER*7 DTFL0
185 C      CHARACTER*7 DTFL1
186 C      CHARACTER*6 DTFL1D
187 C      CHARACTER*7 DTFL2D
188 C      CHARACTER*8 FDATE
189 C      CHARACTER*9 FRAM

```

RAM2D1 (version C)

```

190 CHARACTER*6 FRM
191 CHARACTER*1 ISTEP1
192 CHARACTER*1 ISTEP2
193 CHARACTER*1 ISTEP3
194 CHARACTER*10 NUMRAL
195 CHARACTER*9 PDN1D
196 CHARACTER*12 PDN2D
197 CHARACTER*12 PDN0
198 CHARACTER*12 PDN1
199 CHARACTER*26 RLFBET
200 CHARACTER*1 TDIM
201 CHARACTER*1 YDIM
202 EQUIVALENCE (CWQ,WQ1), (CWQ(NTHP),WQ2)
203 NAMELIST/NAML/NPUMP,YM,TM,ZINT,RKP,RKS,YOFF,TOFF,YWIDTH,TWIDTH,
204 1 YOST,TOST,YWST,TWST,RINT,RIST,RAMASM,RALASM,NHYP,PHL,PHST,TOC,
205 2 ITYPE,RTYPE,RABAMP,RDSLIM,ICOND,ZSTEP,ZFINAL,ZKEEP,NMAX,TTWO,GAIN
206 COMMON/VINIT/NPUMP,YM,TM,ZINT,YOFF,TOFF,YWIDTH,TWIDTH,YOST,TOST,
207 1 YWST,TWST,RAMP,RAST,RAMASM,RALASM,NHYP,PHL,PHST,TOC,ITYPE,RTYPE,
208 2 RABAMP,RDSLIM
209 COMMON/VARTWO/EL,ES,Q,AW1,AW2,CW,RKP,RKS
210 COMMON/VWORK/AEL,AES,AQ,AW,CWQ,COMVEC,RKAP1,RKAP2,TTWO,YFAC,RDT
211 C
212 DATA PI/3.14159265358979/,SPEED/0.0299779/
213 DATA YM/-0.3,0.3/,TM/-100.0,100.0/,NPUMP/2/,GAIN/3.0/,
214 1 RINT/NP*0.55/,RIST/0.003/,TTWO/633.0/,YOFF/0.14,-0.14,NPM2*0.0/,
215 2 TOFF/NP*0.0/,YWIDTH/NP*0.10/,TWIDTH/NP*40.0/,YOST/0.0/,
216 3 TOST/-40.0/,YWST/0.10/,TWST/40.0/,RAMASM/1.5/,RALASM/5.0/,
217 4 NHYP/8/,PHL/NP*0.0/,TOC/5.0/,PHST/0.0/,ICOND/2/,ITYPE/8*1/,
218 5 RTYPE/8*2.0/,RABAMP/8*0.0/,RDSLIM/8*1.0/,ZINT/20.0/,
219 6 RKP/1.180E+5/,RKS/0.91893E+5/,ZSTEP/0.05/,ZFINAL/50.0/,
220 7 ZKEEP/1.0/,NMAX/4000/
221 C
222 CALL ASSIGN(IRRE,'DN'L,'ERRM'L,'A'L,'FT59'L)
223 RLFBET='ABCDEFGHIJKLMNOPQRSTUVWXYZ'
224 C
225 12345678901234567890123456
226 NUMRAL='0123456789'
227 IF (NT.GT.8) THEN
228 ITDIM=NINT(ALOG(FLOAT(NT))/ALOG(2.0))
229 TDIM=RLFBET (ITDIM:ITDIM)
230 ELSE
231 TDIM=NUMRAL (NT+1:NT+1)
232 ENDIF
233 IF (NY.GT.8) THEN
234 IYDIM=NINT(ALOG(FLOAT(NY))/ALOG(2.0))
235 YDIM=RLFBET (IYDIM:IYDIM)
236 ELSE
237 YDIM=NUMRAL (NY+1:NY+1)
238 ENDIF
239 CALL DATE(NDATE)
240 WRITE (FDATE,'(A8)') NDATE
241 D1=FDATE (2:2)
242 D1A=FDATE (1:2)
243 IF (D1A.EQ.'10') D1='A'
244 IF (D1A.EQ.'11') D1='B'
245 IF (D1A.EQ.'12') D1='C'
246 D2=FDATE (4:5)
247 D3=FDATE (7:8)
248 FRM='F'//TDIM//YDIM//D1//D2
249 FRAM='F'//TDIM//YDIM//D1A//D2//D3
250 IF (NT.GT.8.AND.NY.GT.8) THEN
251 ISTEP1=NUMRAL(1:1)
252 ISTEP2=NUMRAL(2:2)
253 DTFL0=FRM//ISTP1

```

RAM2D1 (version C)

```

253      DTFL1=FRM//ISTP2
254      CALL ASSIGN(IRRE,'DN'L,DTFL0,'A'L,'FT02'L)
255      WRITE (59,*) 'ASSIGN FT02= ',DTFL0
256      CALL ASSIGN(IRRE,'DN'L,DTFL1,'A'L,'FT03'L)
257      WRITE (59,*) 'ASSIGN FT03= ',DTFL1
258      PDN0=FRAM//ISTP1//ISTP1//ISTP1
259      PDN1=FRAM//ISTP1//ISTP1//ISTP2
260      IZNO=1
261      ELSE
262      DTFL1D=FRM
263      CALL ASSIGN(IRRE,'DN'L,DTFL1D,'A'L,'FT04'L)
264      WRITE (59,*) 'ASSIGN FT04= ',DTFL1D
265      PDN1D=FRAM
266      ENDIF
267      CALL ASSIGN(IRRE,'DN'L,'NRAM'L,'A'L,'FT01'L)
268      READ (1,NAML)
269      CALL SECOND(STOT1)
270      C
271      C - SET KAPPA-FACTORS
272      RKAP1=SQRT(GAIN/(RKS*(RKP-RKS)*TTWO))/(8.0*PI)
273      RKAP2=4.0*PI*RKS*(RKP-RKAP1)*SPEED/RKAP1
274      C
275      C - SET PUMP AND STOKES AMPLITUDES
276      R1=8.0*PI/SPEED
277      NAMP=NPUMP
278      IF (NY.LE.8) NAMP=NY
279      DO 5 I1=1,NAMP
280      5      RAMP(I1)=SQRT(R1*RINT(I1))
281      RAST=SQRT(R1*RIST)
282      C
283      C - MISCELLANEOUS INITIALIZATIONS, INCLUDING THE WORKING ARRAY FOR THE
284      C      Y-DIRECTION FFT
285      ZFIN=ZFINAL-1.0E-06
286      ZKP=ZKEEP-1.0E-06
287      N999=AMOD(ZFINAL,ZKEEP)
288      IF (N999.GE.998) THEN
289      WRITE (59,*) 'DATA FILES IN EXCESS OF 999'
290      CALL EXIT(1)
291      ENDIF
292      IF (NY.GT.8) CALL CFOUR2(EL,CW,NY,NT,1,0,AW1,AW2)
293      ZVAL=0.0
294      ZH=0.5*ZSTEP
295      C
296      C - DETERMINE Y-SECOND-ORDER-DERIVATIVE KERNEL
297      C
298      IF (NY.GT.8) THEN
299      YFAC=2.0*PI/(YM(2)-YM(1))
300      DO 8 I2=1,NY/2
301      8      CYVEC(I2,2)=-0.5*(0.0,1.0)*ZH*((I2-1)*YFAC)**2
302      DO 9 I2=1+NY/2,NY
303      9      CYVEC(I2,2)=-0.5*(0.0,1.0)*ZH*((-NY+I2-1)*YFAC)**2
304      DO 10 I2=1,NY
305      CYVEC(I2,1)=CEXP(CYVEC(I2,2)/RKP)
306      CYVEC(I2,2)=CEXP(CYVEC(I2,2)/RKS)
307      10      CONTINUE
308      ELSE
309      YFAC=1
310      DO 12 I2=1,NY
311      CYVEC(I2,1)=1.0
312      CYVEC(I2,2)=1.0
313      12      CONTINUE
314      ENDIF
315      C

```

RAM2D1 (version C)

```

316 C — SET TRAT AND THE WORKING ARRAYS FOR DETQ
317 C
318     IF (NT.GT.8) TRAT=AMAX1(-TM(1)/TTWO,TM(2)/TTWO)
319 C
320 C - IF METHOD 2, SET WQ1 AND WQ2
321     IF (TRAT.LE.10.0.AND.NT.GT.8) THEN
322         RDT=(TM(2)-TM(1))/NT
323         DO 15 I3=1,NT
324             TVAL=TM(1)+RDT*(I3-1)
325             WQ1(I3)=EXP(TVAL/TTWO)
326             WQ2(I3)=1.0/WQ1(I3)
327     15     CONTINUE
328 C
329 C - IF METHOD 3, SET CWQ AND COMVEC
330     ELSEIF (TRAT.GT.10.AND.NT.GT.8) THEN
331         CALL CFOUR2(Q,CWQ,NT,NY,1.0,AW1,AW2)
332         R1=1.0/TTWO
333         R2=2.0*PI/(TM(2)-TM(1))
334         R3=NT
335         DO 17 I3=1,NT/2
336     17     COMVEC(I3)=-(0.0,1.0)*RKAP1/((R1-(0.0,1.0)*(I3-1)*R2)*R3)
337         DO 18 I3=NT/2+1,NT
338     18     COMVEC(I3)=-(0.0,1.0)*RKAP1/((R1-(0.0,1.0)*(-NT+I3-1)*R2)*R3)
339     ENDIF
340 C
341 C — RECORD INITIAL DATA
342 C
343     IF (NT.GT.8.AND.NY.GT.8) THEN
344         WRITE (2) NPUMP,YM,TM,ZINT,RKP,RKS,YOFF,TOFF,YWIDTH,TWIDTH,
345     1     YOST,TOST,YWST,TWST,RINT,RIST,RAMASM,RALASM,NHYP,PHL,PHST,
346     2     TOC,ICOND,ITYPE,RTYPE,RABAMP,RDSLIM,ZSTEP,ZFINAL,ZKEEP,NMAX,
347     3     TTWO,GAIN
348         WRITE (3) NPUMP,YM,TM,ZINT,RKP,RKS,YOFF,TOFF,YWIDTH,TWIDTH,
349     1     YOST,TOST,YWST,TWST,RINT,RIST,RAMASM,RALASM,NHYP,PHL,PHST,
350     2     TOC,ICOND,ITYPE,RTYPE,RABAMP,RDSLIM,ZSTEP,ZFINAL,ZKEEP,NMAX,
351     3     TTWO,GAIN
352     ELSE
353         WRITE (4) NPUMP,YM,TM,ZINT,RKP,RKS,YOFF,TOFF,YWIDTH,TWIDTH,
354     1     YOST,TOST,YWST,TWST,RINT,RIST,RAMASM,RALASM,NHYP,PHL,PHST,
355     2     TOC,ICOND,ITYPE,RTYPE,RABAMP,RDSLIM,ZSTEP,ZFINAL,ZKEEP,NMAX,
356     3     TTWO,GAIN
357     ENDIF
358     NWRT=1
359 C
360 C - DETERMINE CPU TIME FOR INIT
361     CALL SECOND(SINIT1)
362     CALL INIT(ICOND)
363     CALL SECOND(SINIT2)
364     SINIT=SINIT2-SINIT1
365 C
366 C - RECORD INITIAL COORDINATE DATA AND FOURIER DATA:  NOTE AQ=Q=0.0
367     IF (NY.GT.8) THEN
368         CALL SHFT(EL,NY,NT)
369         CALL SHFT(ES,NY,NT)
370         IF (NT.GT.8) THEN
371             WRITE (2) ZVAL,EL
372             WRITE (2) ZVAL,ES
373             WRITE (2) ZVAL,Q
374             WRITE (3) ZVAL,EL
375             WRITE (3) ZVAL,ES
376             WRITE (3) ZVAL,Q
377         CLOSE (2)
378         CALL SAVE(IRRE,'DN'L,DTFLO,'PDN'L,PDNO,

```

RAM2D1 (version C)

```

379      1      'RESIDE'L,'OFFLINE'L)
380      CALL RELEASE(IRRE,'DN'L,DTFL0)
381      ELSE
382      WRITE (4) ZVAL,EL
383      WRITE (4) ZVAL,ES
384      WRITE (4) ZVAL,Q
385      ENDIF
386      CALL SHFT(EL,NY,NT)
387      CALL SHFT(ES,NY,NT)
388      ELSE
389      WRITE (4) ZVAL,EL
390      WRITE (4) ZVAL,ES
391      WRITE (4) ZVAL,Q
392      ENDIF
393      NWRT=NWRT+3
394      C
395      C — DETERMINE INITIAL FOURIER DATA
396      C
397      DO 20 I2=1,NY
398      DO 20 I3=1,NT
399      AEL(I3,I2)=EL(I3,I2)
400      AES(I3,I2)=ES(I3,I2)
401      AQ(I3,I2)=Q(I3,I2)
402      20      CONTINUE
403      IF (NY.GT.8) THEN
404      CALL CF0UR2(AEL,CW,NY,NT,-1.1,AW1,AW2)
405      CALL CF0UR2(AES,CW,NY,NT,-1.1,AW1,AW2)
406      R1=1.0/(YFAC*NY)
407      DO 30 I2=1,NY
408      DO 30 I3=1,NT
409      AEL(I3,I2)=R1*AEL(I3,I2)
410      AES(I3,I2)=R1*AES(I3,I2)
411      30      CONTINUE
412      C
413      C — RECORD INITIAL FOURIER DATA: NOTE AQ=0.0
414      CALL SHFT(AEL,NY,NT)
415      CALL SHFT(AES,NY,NT)
416      IF (NT.GT.8) THEN
417      WRITE (2) ZVAL,AEL
418      WRITE (2) ZVAL,AES
419      WRITE (2) ZVAL,AQ
420      WRITE (3) ZVAL,AEL
421      WRITE (3) ZVAL,AES
422      WRITE (3) ZVAL,AQ
423      ELSE
424      WRITE (4) ZVAL,AEL
425      WRITE (4) ZVAL,AES
426      WRITE (4) ZVAL,AQ
427      ENDIF
428      CALL SHFT(AEL,NY,NT)
429      CALL SHFT(AES,NY,NT)
430      NWRT=NWRT+3
431      ENDIF
432      C
433      C — ENTER THE LOOP OVER STEPS IN Z
434      C
435      DO 500 I0=1,NST
436      CALL SECOND(SSTEP1)
437      C
438      C — EXIT CONDITION: STORAGE IS FILLED
439      IF (I0.GT.NMAX) THEN
440      WRITE(59,50)
441      50      FORMAT(' NMAX REACHED')
```

RAM2D1 (version C)

```

442      GO TO 510
443      ENDIF
444      ZVAL=10*ZSTEP
445      C
446      C — CALCULATE THE FIRST EULER STEP
447      C
448      CALL SECOND(SFE1)
449      CALL DERIV(1)
450      CALL SECOND(SFE2)
451      DO 100 I2=1,NY
452      DO 100 I3=1,NT
453      C1=AW(I3,I2,1)
454      C2=AW(I3,I2,2)
455      AW(I3,I2,1)=AEL(I3,I2)*CYVEC(I2,1)
456      AW(I3,I2,2)=AES(I3,I2)*CYVEC(I2,2)
457      AEL(I3,I2)=AW(I3,I2,1)+ZH*C1
458      AES(I3,I2)=AW(I3,I2,2)+ZH*C2
459      100 CONTINUE
460      C
461      C — SOLVE CONSTRAINT EQUATION FOR THE MATERIAL EXCITATION
462      CALL SECOND(SQ1)
463      CALL DETQ(TRAT)
464      CALL SECOND(SQ2)
465      C
466      C — CALCULATE THE SECOND EULER STEP
467      C
468      CALL SECOND(SFE3)
469      CALL DERIV(2)
470      CALL SECOND(SFE4)
471      DO 110 I2=1,NY
472      DO 110 I3=1,NT
473      AEL(I3,I2)=AW(I3,I2,1)*CYVEC(I2,1)+ZSTEP*AW(I3,I2,3)
474      AES(I3,I2)=AW(I3,I2,2)*CYVEC(I2,2)+ZSTEP*AW(I3,I2,4)
475      110 CONTINUE
476      C
477      C — SOLVE CONSTRAINT EQUATION FOR THE MATERIAL EXCITATION
478      CALL SECOND(SQ3)
479      CALL DETQ(TRAT)
480      CALL SECOND(SQ4)
481      C
482      C — RECORD DATA
483      C
484      IF (ZVAL.GE.ZKP) THEN
485      ZKP=ZKP+ZKEEP
486      IF (NT.GT.8.AND.NY.GT.8) THEN
487      IZNO=IZNO+1
488      INUMRL=AIN(0.01*IZNO)
489      ISTEP1=NUMRL (INUMRL+1:INUMRL+1)
490      IRST=IZNO-100*INUMRL
491      INUMRL=AIN(0.1*IRST)
492      ISTEP2=NUMRL (INUMRL+1:INUMRL+1)
493      INUMRL=IRST-10*INUMRL
494      ISTEP3=NUMRL (INUMRL+1:INUMRL+1)
495      DTFL2D=FRM//'2'
496      CALL ASSIGN(IRRE,'DN'L,DTFL2D,'A'L,'FT04'L)
497      WRITE (59,*) 'ASSIGN FT04=' ,DTFL2D
498      PDN2D=FRM//ISTEP1//ISTEP2//ISTEP3
499      ENDIF
500      IF (NY.GT.8) THEN
501      DO 115 I2=1,NY
502      DO 115 I3=1,NT
503      EL(I3,I2)=YFAC*EL(I3,I2)
504      ES(I3,I2)=YFAC*ES(I3,I2)

```

RAM2D1 (version C)

```

505      AQ(I3,I2)=Q(I3,I2)
506      115      CONTINUE
507      CALL SHFT(EL,NY,NT)
508      CALL SHFT(ES,NY,NT)
509      CALL SHFT(Q,NY,NT)
510      WRITE (4) ZVAL,EL
511      WRITE (4) ZVAL,ES
512      WRITE (4) ZVAL,Q
513      CALL SHFT(EL,NY,NT)
514      CALL SHFT(ES,NY,NT)
515      CALL SHFT(Q,NY,NT)
516      CALL CF0UR2(AQ,CW,NY,NT,-1,1,AW1,AW2)
517      CALL SHFT(AEL,NY,NT)
518      CALL SHFT(AES,NY,NT)
519      CALL SHFT(AQ,NY,NT)
520      WRITE (4) ZVAL,AEL
521      WRITE (4) ZVAL,AES
522      WRITE (4) ZVAL,AQ
523      NWRT=NWRT+6
524      CALL SHFT(AEL,NY,NT)
525      CALL SHFT(AES,NY,NT)
526      IF (NT.GT.8) THEN
527        CLOSE (4)
528        CALL SAVE(IRRE,'DN'L,DTFL2D,'PDN'L,PDN2D,
529          1      'RESIDE'L,'OFFLINE'L)
530        CALL RELEASE (IRRE,'DN'L,DTFL2D)
531      ENDIF
532      ELSE
533        WRITE (4) ZVAL,EL
534        WRITE (4) ZVAL,ES
535        WRITE (4) ZVAL,Q
536        NWRT=NWRT+3
537      ENDIF
538    ENDIF
539    CALL SECOND(SSTEP2)
540  C
541  C — SET TIMING DATA
542  C
543      SSTEP(I0)=SSTEP2-SSTEP1
544      SFE(I0)=SFE4-SFE3+SFE2-SFE1
545      SQ(I0)=SQ4-SQ3+SQ2-SQ1
546  C
547  C - EXIT CONDITION: ZFINAL IS REACHED
548      IF (ZVAL.GE.ZFINAL) GO TO 510
549      500 CONTINUE
550  C
551  C — EXIT ROUTINES
552  C
553      510 CONTINUE
554  C
555  C - SET STOT; RECORD CPU TIMING DATA
556      CALL SECOND(STOT2)
557      STOT=STOT2-STOT1
558      NWRT=NWRT+1
559      IF (NT.GT.8.AND.NY.GT.8) THEN
560        WRITE (3) NWRT,ZVAL,STOT,SINIT,SSTEP,SFE,SQ
561        CLOSE (3)
562        CALL SAVE(IRRE,'DN'L,DTFL1D,'PDN'L,PDN1,'RESIDE'L,'OFFLINE'L)
563      ELSE
564        WRITE (4) NWRT,ZVAL,STOT,SINIT,SSTEP,SFE,SQ
565        CLOSE (4)
566        CALL SAVE(IRRE,'DN'L,DTFL1D,'PDN'L,PDN1D,'RESIDE'L,'OFFLINE'L)
567      ENDIF

```

RAM2D1 (version C)

```

568      CALL EXIT(1)
569      END
570      C
571      C
572      C
573      C
574      C
575      SUBROUTINE DETQ(TRAT)
576      C
577      C THIS SUBROUTINE FIRST CALCULATES THE PUMP AND AND STOKES FIELDS IN
578      C THE COORDINATE SPACE FROM THEIR FOURIER SPACE REPRESENTATIONS.
579      C IT THEN DETERMINES THE MATERIAL EXCITATION (Q) IN THREE DIFFERENT
580      C PARAMETER REGIMES: 1) IF NT IS LESS THAN OR EQUAL TO 8, A SET OF
581      C 1-D STATIONARY CASES IS RUN. 2) IF MAX(ABS(T/TTWO)) < 10.0 AND
582      C NT > 8, A RUNNING SUM IS PERFORMED. 3) IF MAX(T/TTWO) > 10.0
583      C AND NT > 8, AN FFT APPROACH IS USED.
584      C
585      C
586      C
587      C
588      C
589      C
590      C
591      C
592      C
593      C
594      C
595      C
596      C
597      C
598      C
599      C
600      C
601      C
602      C
603      C
604      C
605      C
606      C
607      C
608      C
609      C
610      C
611      C
612      C
613      C
614      C
615      C
616      C
617      C
618      C
619      C
620      C
621      C
622      C
623      C
624      C
625      C
626      C
627      C
628      C
629      C
630      C

```

--VARIABLES--

```

      TRAT = MAX(T/TTWO)

      PARAMETER(NT=256,NY=256,NTHP=1+NT/2,NS=5*NY/2)
      IMPLICIT COMPLEX(A-E,Q)
      DIMENSION EL(NT,NY),ES(NT,NY),Q(NT,NY),AEL(NT,NY),AES(NT,NY),
1  AQ(NT,NY),AW(NT,NY,4),CW(NS),AW1(NY),AW2(NY),COMVEC(NT),CWQ(NT),
2  WQ1(NT),WQ2(NT)
      EQUIVALENCE (CWQ,WQ1),(CWQ(NTHP),WQ2)
      COMMON/VARTWO/EL,ES,Q,AW1,AW2,CW,RKP,RKS
      COMMON/VWORK/AEL,AES,AQ,AW,CWQ,COMVEC,RKAP1,RKAP2,TTWO,YFAC,RDT
      C
      DO 10 I2=1,NY
      DO 10 I3=1,NT
      EL(I3,I2)=AEL(I3,I2)
      ES(I3,I2)=AES(I3,I2)
10      CONTINUE
      IF (NY.GT.8) THEN
      CALL CFOUR2(EL,CW,NY,NT,1,1,AW1,AW2)
      CALL CFOUR2(ES,CW,NY,NT,1,1,AW1,AW2)
      ENDIF
      IF (NT.LE.8) THEN
      DO 20 I2=1,NY
      DO 20 I3=1,NT
20      Q(I3,I2)=-(0.0,1.0)*RKAP1*TTWO*CONJG(ES(I3,I2))*EL(I3,I2)
      ELSEIF (TRAT.LE.10.0) THEN
      DO 30 I2=1,NY
      DO 30 I3=2,NT
30      Q(I3,I2)=Q(I3-1,I2)-(0.0,1.0)*RKAP1*RDT*CONJG(ES(I3,I2))
1      *EL(I3,I2)*WQ1(I3)
      DO 35 I2=1,NY
      DO 35 I3=1,NT
35      Q(I3,I2)=WQ2(I3)*Q(I3,I2)
      ELSE
      DO 40 I2=1,NY
      DO 40 I3=1,NT
40      Q(I3,I2)=CONJG(ES(I3,I2))*EL(I3,I2)
      CALL INVERT(Q,AQ,NT,NY)
      CALL CFOUR2(AQ,CWQ,NT,NY,1,1,AW1,AW2)
      DO 45 I3=1,NT
      DO 45 I2=1,NY
45      AQ(I2,I3)=COMVEC(I3)*AQ(I2,I3)
      CALL CFOUR2(AQ,CWQ,NT,NY,-1,1,AW1,AW2)
      CALL INVERT(AQ,Q,NY,NT)
      ENDIF

```


RAM2D1 (version C)

```

631      IF (NY.GT.8) THEN
632          R1=YFAC**2
633          DO 50 I2=1,NY
634              DO 50 I3=1,NT
635      50      Q(I3,I2)=R1*Q(I3,I2)
636      ENDIF
637      RETURN
638      END
639      C
640      C
641      C
642      C
643      C
644      SUBROUTINE DERIV(IFILL)
645      C
646      C THIS SUBROUTINE CALCULATES THE Z-DERIVATIVES OF THE PUMP AND STOKES
647      C FIELDS. THIS CALCULATION IS DONE IN KY-SPACE. THE LINEAR PORTION OF
648      C THE SECOND-ORDER-DERIVATIVE OPERATOR HAS A FINITE STEP CORRECTION
649      C (CONTAINED IN CYVEC) SO THAT THE LINEAR CONTRIBUTION IS EXACT.
650      C
651      C
652      C
653      C
654      C
655      C
656      C
657      C
658      C
659      C
660      C
661      C
662      C
663      C
664      C
665      C
666      C
667      C
668      C
669      C
670      C
671      C
672      C
673      C
674      C
675      C
676      C
677      C
678      C
679      C
680      C
681      C
682      C
683      C
684      C
685      C
686      C
687      C
688      C
689      C
690      C
691      C
692      C
693      C

```

—VARIABLES—

```

IFILL = DERIVATIVE NUMBER
      = 1: INITIAL STEP
      = 2: MID-POINT STEP

PARAMETER(NT=256,NY=256,NS=5*NY/2)
IMPLICIT COMPLEX(A-E,Q)
DIMENSION EL(NT,NY),ES(NT,NY),Q(NT,NY),AEL(NT,NY),AES(NT,NY),
1 AQ(NT,NY),AW(NT,NY,4),CW(NS),AW1(NY),AW2(NY),COMVEC(NT),CWQ(NT)
COMMON/VARTWO/EL,ES,Q,AW1,AW2,CW,RKP,RKS
COMMON/VWORK/AEL,AES,AQ,AW,CWQ,COMVEC,RKAP1,RKAP2,TTWO,YFAC,RDT

C1=-(0.0,1.0)*(RKP/RKS)*RKAP2
IF (NY.GT.8) C1=C1/NY
DO 10 I2=1,NY
DO 10 I3=1,NT
10 AQ(I3,I2)=C1*Q(I3,I2)*ES(I3,I2)
IF (NY.GT.8) CALL CFOUR2(AQ,CW,NY,NT,-1,1,AW1,AW2)
IV=2*IFILL-1
DO 20 I2=1,NY
DO 20 I3=1,NT
20 AW(I3,I2,IV)=AQ(I3,I2)
C1=-(0.0,1.0)*RKAP2
IF (NY.GT.8) C1=C1/NY
DO 30 I2=1,NY
DO 30 I3=1,NT
30 AQ(I3,I2)=C1*CONJG(Q(I3,I2))*EL(I3,I2)
IF (NY.GT.8) CALL CFOUR2(AQ,CW,NY,NT,-1,1,AW1,AW2)
IV=2*IFILL
DO 40 I2=1,NY
DO 40 I3=1,NT
40 AW(I3,I2,IV)=AQ(I3,I2)
RETURN
END

```

SUBROUTINE SHFT(FDATA,NF,NV)

```

THIS SUBROUTINE SHIFTS THE FOURIER DATA SO THAT ZERO FREQUENCY IS AT
THE 1+NF/2 LOCATION (THE CENTER OF THE ARRAY)

```

RAM2D1 (version C)

```

694 C      DIMENSION FDATA(2*NV,NF)
695      NFH=NF/2
696      DO 100 I2=1,NFH
697      DO 100 I3=1,2*NV
698      TEMP=FDATA(I3,I2)
699      FDATA(I3,I2)=FDATA(I3,I2+NFH)
700      FDATA(I3,I2+NFH)=TEMP
701      100 CONTINUE
702      RETURN
703      END
704 C
705 C
706 C
707 C
708 C
709      SUBROUTINE INVERT(EDATA,EWORK,NF,NV)
710 C      THIS SUBROUTINE INVERTS THE INNER AND OUTER ARRAY VARIABLES
711 C
712 C      COMPLEX EDATA(NF,NV),EWORK(NV,NF)
713      IF (NF.LE.1.OR.NV.LE.1) RETURN
714      DO 50 I3=1,NF
715      DO 50 I2=1,NV
716      EWORK(I2,I3)=EDATA(I3,I2)
717      50 CONTINUE
718      RETURN
719      END
720 C
721 C
722 C
723 C
724 C
725      SUBROUTINE INIT(ICOND)
726 C      THIS SUBROUTINE DETERMINES THE INITIAL PROFILES FOR THE STOKES AND
727 C      PUMP WAVES. MOST VARIABLES ARE DECLARED IN THE MAIN ROUTINE.
728 C
729 C      ---VARIABLES---
730 C
731 C      ICOND = 1: DOUBLE-SECH PROFILE
732 C      2: SECH**2-HYPERGAUSSIAN PROFILE WITH STOKES ASYMMETRY
733 C      IN TIME AND IMPOSED CHIRP
734 C      3: 1-D TRANSIENT PROFILES: TYPE DETERMINED BY ITYPE
735 C      ITYPE = 1: SECH**N PROFILES
736 C      2: RECTANGULAR PROFILES
737 C      3: LORENTZIAN**N PROFILES
738 C      4: EXP(|AT|**N) PROFILES
739 C      4: STATIONARY HYPERGAUSSIAN PROFILES WITH PUMP
740 C      ABERRATION INCLUDED
741 C
742 C      PARAMETER(NT=256,NY=256,NP=10,NYH=NY/2,NYHP=NYH+1,NS=5*NY/2)
743      IMPLICIT COMPLEX(A-E,Q)
744      DIMENSION EL(NT,NY),ES(NT,NY),Q(NT,NY),CW(NS),AW1(NY),AW2(NY),
745      1 YSTOR1(NY),YSTOR2(NY),YSTOR3(NY),TSTORE(NT),TSC(NT),PHL(NP),
746      2 YWIDTH(NP),TWIDTH(NP),YOFF(NP),TOFF(NP),RAMP(NP),ITYPE(8),
747      3 RTYPE(8),RABAMP(8),RDSLIM(8),YM(2),TM(2)
748      COMMON/VINIT/NPUMP,YM,TM,ZINT,YOFF,TOFF,YWIDTH,TWIDTH,YOST,TOST,
749      1 YWST,TWST,RAMP,RAST,RAMASM,RALASM,NHYP,PHL,PHST,TOC,ITYPE,RTYPE,
750      2 RABAMP,RDSLIM
751      COMMON/VARTWO/EL,ES,Q,AW1,AW2,CW,RKP,RKS
752      DATA PI/3.14159265358979/,SQ2/1.41421356237309/,
753      2 SQ4/1.18920711500272/,RAL2/0.693147180559945/,
754      3 SQ10/3.16227766016838/,SQ12/3.46410161513775/
755 C
756 C

```

RAM2D1 (version C)

```

757       IF (ICOND.LT.1.OR.ICOND.GT.4) THEN
758         WRITE (59,5)
759       5   FORMAT(' ONLY TYPES 1,2,3 AND 4 ARE INITIALIZED')
760         CALL EXIT(1)
761       ENDF
762     C
763     C - INITIALIZE VARIABLE ARRAYS
764       DO 8 I2=1,NY
765       DO 8 I3=1,NT
766         EL(I3,I2)=(0.0,0.0)
767         ES(I3,I2)=(0.0,0.0)
768         Q(I3,I2)=(0.0,0.0)
769       8   CONTINUE
770       IF (ICOND.NE.3.AND.NY.LE.8) THEN
771         WRITE (59,10)
772       10   FORMAT(' ICOND MUST EQUAL 3 IN 1-D TRANSIENT RUNS (NY = 8 OR',
773         1   ' LESS):'/'.' ICOND IS RESET TO 3')
774         ICOND=3
775       ENDF
776       IF (ICOND.NE.4.AND.NT.LE.8) THEN
777         WRITE (59,12)
778       12   FORMAT(' ICOND MUST EQUAL 4 IN STATIONARY RUNS (NT = 8 OR',
779         1   ' LESS):'/'.' ICOND IS RESET TO 4')
780         ICOND=4
781       ENDF
782       IF (ICOND.EQ.3) GO TO 210
783     C
784     C - INITIALIZE Y-QUANTITIES
785       IF (ABS(YM(2)+YM(1)).GT.1.0E-06) WRITE (59,14)
786       14   FORMAT(' YM(2) MUST EQUAL -YM(1)')
787         YM(2)=-YM(1)
788         RDY=(YM(2)-YM(1))/NY
789       DO 16 I2=1,NYH
790       16   YSTOR2(I2)=RDY*(I2-1)
791       DO 18 I2=NYHP,NY
792       18   YSTOR2(I2)=RDY*(I2-1-NY)
793       IF (ICOND.EQ.4) GO TO 310
794     C
795       RDT=(TM(2)-TM(1))/NT
796       IF (ICOND.EQ.2) GO TO 110
797       RFAC=2.0*ALOG(1.0+SQ2)
798     C
799     C
800     C — DOUBLE-SECH PROFILE
801     C
802     C
803     C - DETERMINE PUMP FACTORS
804       DO 50 I1=1,NPUMP
805       ALPHA=-(0.0,1.0)*YOFF(I1)*RKP/ZINT
806       YFAC=RFAC/YWIDTH(I1)
807       TFAC=RFAC/TWIDTH(I1)
808       DO 20 I2=1,NY
809       Y1=YSTOR2(I2)
810       YV=EXP(YFAC*(Y1-YOFF(I1)))
811       YSTOR1(I2)=(1.0/(YV+1.0/YV))
812       20   CONTINUE
813         T1=TM(1)-TOFF(I1)
814         DO 30 I3=1,NT
815         TV=EXP(TFAC*(T1+RDT*(I3-1)))
816         TSTORE(I3)=4.0*RAMP(I1)/(TV+1.0/TV)
817       30   CONTINUE
818       DO 50 I2=1,NY
819       C1=CEXP(ALPHA*YSTOR2(I2))

```

RAM2D1 (version C)

```

820      DO 50 I3=1,NT
821      EL(I3,I2)=EL(I3,I2)+YSTOR1(I2)*TSTORE(I3)*C1
822      50 CONTINUE
823      C
824      C - DETERMINE STOKES FACTORS
825      ALPHA=-(0.0,1.0)*YOST*RKS/ZINT
826      YFAC=RFAC/YWST
827      DO 70 I2=1,NY
828      Y1=YSTOR2(I2)
829      YV=EXP(YFAC*(Y1-YOST))
830      YSTOR1(I2)=(1.0/(YV+1.0/YV))
831      70 CONTINUE
832      T1=TM(1)-TOST
833      DO 80 I3=1,NT
834      TV=EXP(TFAC*(T1+RDT*(I3-1)))
835      TSTORE(I3)=4.0*RAST/(TV+1.0/TV)
836      80 CONTINUE
837      DO 100 I2=1,NY
838      C1=CEXP(ALPHA*YSTOR2(I2))
839      DO 100 I3=1,NT
840      ES(I3,I2)=YSTOR1(I2)*TSTORE(I3)*C1
841      100 CONTINUE
842      RETURN
843      C
844      C
845      C --- SECH**2-HYPERGAUSSIAN PROFILE WITH ASYMMETRIC STOKES WAVE
846      C
847      110 CONTINUE
848      RFACY=2.0***(NHYP-1)*RAL2
849      RFACT=2.0*ALOG(SQ4+SQRT(SQ2-1.0))
850      C
851      C - DETERMINE PUMP FACTORS
852      DO 150 I1=1,NPUMP
853      ALPHA=-(0.0,1.0)*YOFF(I1)*RKP/ZINT
854      YFAC=RFACY/YWIDTH(I1)**NHYP
855      TFAC=RFACT/TWIDTH(I1)
856      DO 120 I2=1,NY
857      YSTOR1(I2)=EXP(-YFAC*(YSTOR2(I2)-YOFF(I1))**NHYP)
858      120 CONTINUE
859      T1=TM(1)-TOFF(I1)
860      DO 130 I3=1,NT
861      TV=EXP(TFAC*(T1+RDT*(I3-1)))
862      TSTORE(I3)=4.0/(TV+1.0/TV)**2
863      130 CONTINUE
864      DO 150 I2=1,NY
865      DO 150 I3=1,NT
866      R1=YSTOR1(I2)*TSTORE(I3)
867      EL(I3,I2)=EL(I3,I2)+RAMP(I1)*R1*CEXP((0.0,1.0)*PHL(I1)*R1**2
868      1 + ALPHA*YSTOR2(I2))
869      150 CONTINUE
870      C
871      C - DETERMINE STOKES CHIRP FACTORS
872      C AT PRESENT, TWC=TWIDTH(1), YWC=YWIDTH(1)
873      TWC=TWIDTH(1)
874      YWC=YWIDTH(1)
875      YFAC=RFACY/YWC**NHYP
876      TFAC=RFACT/TWC
877      DO 160 I2=1,NY
878      YSC(I2)=EXP(-YFAC*YSTOR2(I2)**NHYP)
879      160 CONTINUE
880      T1=TM(1)-TOST-TOC
881      DO 165 I3=1,NT
882      TV=EXP(TFAC*(T1+RDT*(I3-1)))

```

RAM2D1 (version C)

```

883      TSC(I3)=4.0/(TV+1.0/TV)**2
884 165  CONTINUE
885  C
886  C - DETERMINE STOKES FACTORS
887      ALPHA=-(0.0,1.0)*YOST*RKS/ZINT
888      YFAC=RFAC/YWST**NHYP
889      TFAC=RFAC/TWST
890      DO 170 I2=1,NY
891      YSTOR1(I2)=EXP(-YFAC*(YSTOR2(I2)-YOST)**NHYP)
892 170  CONTINUE
893      DO 180 I3=1,NT
894      T1=TM(1)+RDT*(I3-1)-TOST
895  C
896  C - SET STOKES ASYMMETRY
897      TV=EXP(-TFAC*RALASM*T1)
898      T1=T1*(1.0+RAMASM*TV/(TV+1.0/TV))
899      TV=EXP(TFAC*T1)
900      TSTORE(I3)=4.0/(TV+1.0/TV)**2
901 180  CONTINUE
902      DO 190 I2=1,NY
903      DO 190 I3=1,NT
904      R1=YSTOR1(I2)*TSTORE(I3)
905      R2=YSC(I2)*TSC(I3)
906      ES(I3,I2)=RAST*R1*CEXP((0.0,1.0)*PHST*R2**2
907      1 +ALPHA*YSTOR2(I2))
908 190  CONTINUE
909  RETURN
910  C
911  C
912  C — ONE-DIMENSIONAL TRANSIENT CASES (NO Y-VARIATION)
913  C
914 210  CONTINUE
915      IF (NY.GT.8) THEN
916          WRITE (59,212)
917212      FORMAT(' IN TRANSIENT STUDIES, ONLY UP TO 8 CASES CAN BE KEPT')
918          CALL EXIT(1)
919      ENDIF
920      RDT=(TM(2)-TM(1))/NT
921  C
922  C — LOOP OVER CASES
923  C
924      DO 290 I1=1,NY
925  C
926  C — SECH PROFILE
927  C
928      IF (ITYPE(I1).EQ.1) THEN
929          R2=1.0/RTYPE(I1)
930          R1=0.5*R2
931          RFAC=2.0*ALOG(EXP(R1*RAL2)+SQRT(EXP(R2*RAL2)-1.0))
932  C
933  C - DETERMINE PUMP PROFILE
934      TFAC=RFAC/TWIDTH(I1)
935      T1=TM(1)-TOFF(I1)
936      DO 215 I3=1,NT
937      TV=EXP(TFAC*(T1+RDT*(I3-1)))
938      TV=2.0/(TV+1.0/TV)
939      TV=EXP(RTYPE(I1)*ALOG(TV))
940      EL(I3,I1)=RAMP(I1)*TV*CEXP((0.0,1.0)*PHL(I1)*TV**2)
941215  CONTINUE
942  C
943  C - DETERMINE STOKES CHIRP FACTOR
944      TWC=TWIDTH(1)
945      TFAC=RFAC/TWC

```

RAM2D1 (version C)

```

946      T1=TM(1)-TOST-TOC
947      DO 220 I3=1,NT
948      TV=EXP(TFAC*(T1+RDT*(I3-1)))
949      TV=2.0/(TV+1.0/TV)
950      TSC(I3)=EXP(RTYPE(I1)*ALOG(TV))
951      220  CONTINUE
952  C
953  C - DETERMINE STOKES PROFILE
954      TFAC=RFACT/TWST
955      DO 225 I3=1,NT
956      T1=TM(1)+RDT*(I3-1)-TOST
957      TV=EXP(-TFAC*RALASM*T1)
958      T1=T1*(1.0+RAMASM*TV/(TV+1.0/TV))
959      TV=EXP(TFAC*T1)
960      TV=2.0/(TV+1.0/TV)
961      TV=EXP(RTYPE(I1)*ALOG(TV))
962      ES(I3,I1)=RAST*TV*CEXP((0.0,1.0)*PHST*TSC(I3)**2)
963      225  CONTINUE
964  C
965  C --- RECTANGULAR PROFILE (ASSYMETRY AND CHIRP ARE IGNORED)
966  C
967      ELSEIF (ITYPE(I1).EQ.2) THEN
968  C
969  C - DETERMINE PUMP PROFILE
970      IMIN=NINT((TOFF(I1)-TM(1)-0.5*TWIDTH(I1))/RDT) + 1
971      IF (IMIN.LT.1) IMIN=1
972      IMAX=NINT((TOFF(I1)-TM(1)+0.5*TWIDTH(I1))/RDT) + 1
973      IF (IMAX.GT.NT) IMAX=NT
974      DO 230 I3=IMIN,IMAX
975      EL(I3,I1)=RAMP(I1)
976  C
977  C - DETERMINE STOKES PROFILE
978      IMIN=NINT((TOST-TM(1)-0.5*TWST)/RDT) + 1
979      IF (IMIN.LT.1) IMIN=1
980      IMAX=NINT((TOST-TM(1)+0.5*TWST)/RDT) + 1
981      IF (IMAX.GT.NT) IMAX=NT
982      DO 235 I3=IMIN,IMAX
983      ES(I3,I1)=RAST
984  C
985  C --- LORENTZIAN PROFILE
986  C
987      ELSEIF (ITYPE(I1).EQ.3) THEN
988      RFACT=2.0*SQRT(EXP((0.5/RTYPE(I1))*RAL2)-1.0)
989  C
990  C - DETERMINE PUMP PROFILE
991      TFAC=RFACT/TWIDTH(I1)
992      T1=TM(1)-TOFF(I1)
993      DO 240 I3=1,NT
994      TV=T1+RDT*(I3-1)
995      TV=1.0/(1.0+(TFAC*TV)**2)
996      TV=EXP(RTYPE(I1)*ALOG(TV))
997      EL(I3,I1)=RAMP(I1)*TV*CEXP((0.0,1.0)*PHL(I1)*TV**2)
998      240  CONTINUE
999  C
1000  C - DETERMINE STOKES CHIRP FACTOR
1001      TWC=TWIDTH(1)
1002      TFAC=RFACT/TWC
1003      T1=TM(1)-TOST-TOC
1004      DO 245 I3=1,NT
1005      TV=T1+RDT*(I3-1)
1006      TV=1.0/(1.0+(TFAC*TV)**2)
1007      TSC(I3)=EXP(RTYPE(I1)*ALOG(TV))
1008      245  CONTINUE

```

RAM2D1 (version C)

```

1009 C
1010 C - DETERMINE STOKES PROFILE
1011     TFAC=RFACT/TWST
1012     DO 250 I3=1,NT
1013         T1=TM(1)+RDT*(I3-1)-TOST
1014         TV=EXP(-TFAC*RALASM*T1)
1015         T1=T1*(1.0+RAMASM*TV/(TV+1.0/TV))
1016         TV=1.0/(1.0+(TFAC*T1)**2)
1017         TV=EXP(RTYPE(I1)*ALOG(TV))
1018         ES(I3,I1)=RAST*TV*CEXP((0.0,1.0)*PHST*TSC(I3)**2)
1019     250 CONTINUE
1020 C
1021 C — EXPONENTIAL PROFILE (EXPONENT IS TAKEN TO THE POWER RTYPE(I1))
1022 C
1023     ELSEIF (ITYPE(I1).EQ.4) THEN
1024         RFACT=2.0*EXP((1.0/RTYPE(I1))*ALOG(0.5*RAL2))
1025 C
1026 C - DETERMINE PUMP PROFILE
1027     TFAC=RFACT/TWIDTH(I1)
1028     T1=TM(1)-TOFF(I1)
1029     DO 255 I3=1,NT
1030         TV=ABS(TFAC*(T1+RDT*(I3-1)))+1.0E-10
1031         TV=EXP(-EXP(RTYPE(I1)*ALOG(TV)))
1032         EL(I3,I1)=RAMP(I1)*TV*EXP((0.0,1.0)*PHL(I1)*TV**2)
1033     255 CONTINUE
1034 C
1035 C - DETERMINE STOKES CHIRP FACTOR
1036     TWC=TWIDTH(1)
1037     TFAC=RFACT/TWC
1038     T1=TM(1)-TOST-TOC
1039     DO 260 I3=1,NT
1040         TV=ABS(TFAC*(T1+RDT*(I3-1)))+1.0E-10
1041         TSC(I3)=EXP(-EXP(RTYPE(I1)*ALOG(TV)))
1042     260 CONTINUE
1043 C
1044 C - DETERMINE STOKES PROFILE
1045     TFAC=RFACT/TWST
1046     DO 265 I3=1,NT
1047         T1=TM(1)+RDT*(I3-1)-TOST+1.0E-10
1048         TV=EXP(-TFAC*RALASM*T1)
1049         T1=T1*(1.0+RAMASM*TV/(TV+1.0/TV))
1050         TV=EXP(-EXP(RTYPE(I1)*ALOG(ABS(TFAC*T1))))
1051         ES(I3,I1)=RAST*TV*CEXP((0.0,1.0)*PHST*TSC(I3)**2)
1052     265 CONTINUE
1053 C
1054 C — ERROR
1055 C
1056     ELSE
1057         WRITE (59,270) I1,ITYPE(I1)
1058     270     FORMAT(' ONLY TRANSIENT TYPES 1-4 ARE INITIALIZED',
1059     1         ' ON PUMP NO.',I4,' TYPE NO. =',I4)
1060     ENDIF
1061     290 CONTINUE
1062     RETURN
1063 C
1064 C
1065 C — STATIONARY CASE (NO T-VARIATION)
1066 C (AT PRESENT THE DISPERSION LIMIT IS SET WITH RESPECT TO YWIDTH(1))
1067 C
1068     310 CONTINUE
1069     RFACY=2.0*(NHYP-1)*RAL2
1070     RFACK=(0.125*YWIDTH(1)**2/(EXP((2.0/NHYP)*ALOG(RAL2))))
1071     1*(2.0*PI/(NY*RDY))**2

```

RAM2D1 (version C)

```

1072 C
1073 C - DETERMINE PUMP FACTORS
1074 DO 320 I1=1,NPUMP
1075 ALPHA=-(0.0,1.0)*YOFF(I1)*RKP/ZINT
1076 YFAC=RFACY/YWIDTH(I1)*NHYP
1077 DO 315 I2=1,NY
1078 YSTOR1(I2)=EXP(-YFAC*(YSTOR2(I2)-YOFF(I1))*NHYP)
1079 315 CONTINUE
1080 DO 320 I3=1,NT
1081 DO 320 I2=1,NY
1082 EL(I3,I2)=EL(I3,I2)+RAMP(I1)*YSTOR1(I2)*CEXP(ALPHA*YSTOR2(I2))
1083 320 CONTINUE
1084 C
1085 C - DETERMINE RANDOMIZING FACTORS
1086 C
1087 DO 350 I3=1,NT
1088 IF (RDSLIM(I3).LT.1.1) GO TO 350
1089 RKFAC=RFACK/(2.0*RDSLIM(I3)*2-1.0)
1090 C
1091 C - PHASE FACTORS
1092 DO 322 I2=1,NY
1093 322 AW1(I2)=CEXP((0.0,1.0)*2.0*PI*RANF(1))
1094 CALL CFFT2(0,1,NY,AW1,CW,AW2)
1095 DO 326 I2=1,NYH
1096 AW2(I2)=EXP(-RKFAC*(I2-1)*2)*AW2(I2)
1097 AW2(NYH+I2)=EXP(-RKFAC*(NYH-I2+1)*2)*AW2(NYH+I2)
1098 326 CONTINUE
1099 CALL CFFT2(0,-1,NY,AW2,CW,AW1)
1100 DO 330 I2=1,NY
1101 R1=CABS(AW1(I2))
1102 IF (R1.GT.1.0E-10) AW1(I2)=AW1(I2)/R1
1103 330 EL(I3,I2)=EL(I3,I2)*AW1(I2)
1104 C
1105 C - AMPLITUDE FACTORS
1106 IF (RABAMP(I3).LT.0.01) GO TO 350
1107 DO 332 I2=1,NY
1108 332 AW1(I2)=-5.0
1109 DO 334 I2=1,10
1110 DO 334 I2=1,NY
1111 334 AW1(I2)=AW1(I2)+RANF(1)
1112 R1=0.0
1113 DO 335 I2=1,NY
1114 335 R1=R1+AW1(I2)*AW1(I2)
1115 CALL CFFT2(0,1,NY,AW1,CW,AW2)
1116 DO 336 I2=1,NYH
1117 AW2(I2)=EXP(-RKFAC*(I2-1)*2)*AW2(I2)
1118 AW2(NYH+I2)=EXP(-RKFAC*(NYH-I2+1)*2)*AW2(NYH+I2)
1119 336 CONTINUE
1120 CALL CFFT2(0,-1,NY,AW2,CW,AW1)
1121 R2=0.0
1122 DO 337 I2=1,NY
1123 337 AW1(I2)=AW1(I2)/NY
1124 R2=R2+AW1(I2)*AW1(I2)
1125 R1=SQRT(R1/R2)*RABAMP(I3)*SQ12/SQ10
1126 R2=1.0-RABAMP(I3)
1127 DO 340 I2=1,NY
1128 340 EL(I3,I2)=EL(I3,I2)*(R2+R1*AW1(I2))
1129 350 CONTINUE
1130 C
1131 C - DETERMINE STOKES PROFILE
1132 ALPHA=-(0.0,1.0)*YOST*RKS/ZINT
1133 YFAC=RFACY/YWST*NHYP
1134 DO 370 I2=1,NY

```


RAM2D1 (version C)

```

1135 YSTOR1(I2)=EXP(-YFAC*(YSTOR2(I2)-YOST)*NHYP)
1136 370 CONTINUE
1137 DO 390 I3=1,NT
1138 DO 390 I2=1,NY
1139 ES(I3,I2)=RAST*YSTOR1(I2)*CEXP(ALPHA*YSTOR2(I2))
1140 390 CONTINUE
1141 RETURN
1142 C
1143 END
1144 C
1145 C
1146 C
1147 C
1148 SUBROUTINE CFOR2(FDATA,RWORK,NF,NV,ISIGN,ITYPE,RW1,RW2)
1149 C
1150 C THIS SUBROUTINE WAS WRITTEN BY CURTIS R. MENYUK 11/86. IT
1151 C CALCULATES THE FOURIER TRANSFORM OF A SET OF VECTORS STORED IN A TWO-
1152 C DIMENSIONAL ARRAY. THE ROUTINE TRANSFORMS OVER THE OUTER VARIABLE
1153 C (SLOWLY VARYING) AND VECTORIZES OVER THE INNER VARIABLE (RAPIDLY
1154 C VARYING). THE ALGORITHM USED IS DESCRIBED IN NUMERICAL RECIPES
1155 C BY PRESS, ET AL., CHAP. 12. THE ROUTINE HERE IS BASED ON FOUR1.
1156 C
1157 C MODIFIED 5/87:
1158 C IF NV=8 OR LESS THIS SUBROUTINE USES THE OMNILIB ROUTINE CFFT2
1159 C TO CARRY OUT THE FOURIER TRANSFORM SERIALLY.
1160 C
1161 C —VARIABLES—
1162 C
1163 C FDATA = DATA ARRAY. IN THIS PROGRAM, IT IS TREATED AS A REAL
1164 C ARRAY WITH 2*NV X NF ELEMENTS. THE CORRESPONDING
1165 C COMPLEX ARRAY HAS NV X NF ELEMENTS.
1166 C RWORK = WORK ARRAY WHERE THE NEEDED COSINES AND SINES ARE STORED.
1167 C IT HAS 2*(NF-1) ELEMENTS
1168 C NF = THE OUTER DIMENSION OVER WHICH THE ROUTINE TRANSFORMS
1169 C NV = THE INNER DIMENSION OVER WHICH THE PROGRAM VECTORIZES
1170 C ISIGN = SIGN OF THE FOURIER TRANSFORM
1171 C ITYPE = 0: INITIALIZE THE WORK ARRAY (NF IS THE ONLY SIGNIFICANT
1172 C PARAMETER; NV AND ISIGN ARE IGNORED)
1173 C 1: CARRY OUT THE FOURIER TRANSFORM
1174 C RW1,RW2 = WORK ARRAYS WITH 2*NF ELEMENTS USED BY CFFT2
1175 C (INACTIVE WHEN NV > 8)
1176 C ICR,ICI = REFERENCES TO THE WORK ARRAY
1177 C MMAX = SUMMATION SEPARATION IN THE DANIELSON-LANCZOS ROUTINE
1178 C
1179 C DATA TWOPI/6.28318530717959/
1180 C DIMENSION FDATA(2*NV,NF),RWORK(5*NF/2),RW1(2*NF),RW2(2*NF)
1181 C IER=-1
1182 C IF (ITYPE.EQ.0) GO TO 1000
1183 C IF (ITYPE.NE.1) THEN
1184 C
1185 C - ERROR CHECK
1186 C IER=-1
1187 C RETURN
1188 C ENDIF
1189 C
1190 C — IF NV = 8 OR LESS, CALCULATE FOURIER TRANSFORM SERIALLY
1191 C
1192 C IF (NV.LE.8) THEN
1193 C DO 20 I3=1,NV
1194 C DO 10 I2=1,NF
1195 C RW1(2*I2)=FDATA(2*I3,I2)
1196 C RW1(2*I2-1)=FDATA(2*I3-1,I2)
1197 10 CONTINUE

```

RAM2D1 (version C)

```

1198      CALL CFFT2(0,ISIGN,NF,RW1,RWORK,RW2)
1199      DO 20 I2=1,NF
1200      FDATA(2*I3,I2)=RW2(2*I2)
1201      FDATA(2*I3-1,I2)=RW2(2*I2-1)
1202      20      CONTINUE
1203      RETURN
1204      ENDIF
1205      C
1206      C — NV > 8
1207      C
1208      C — BIT REVERSAL ROUTINE
1209      C
1210      J=1
1211      DO 100 I=1,NF
1212      IF (J.GT.I) THEN
1213      DO 40 I1=1,2*NV
1214      TEMP=FDATA(I1,J)
1215      FDATA(I1,J)=FDATA(I1,I)
1216      FDATA(I1,I)=TEMP
1217      40      CONTINUE
1218      ENDIF
1219      M=NF/2
1220      50      CONTINUE
1221      IF ((M.GE.1).AND.(J.GT.M)) THEN
1222      J=J-M
1223      M=M/2
1224      GO TO 50
1225      ENDIF
1226      J=J+M
1227      100     CONTINUE
1228      C
1229      C — DANIELSON-LANCZOS ROUTINE. THE FOURIER DATA IS RECOMBINED.
1230      C
1231      MMAX=1
1232      ICR=1
1233      ICI=2
1234      FSIGN=FLOAT(ISIGN)
1235      120     CONTINUE
1236      IF (NF.GT.MMAX) THEN
1237      ISTEP=2*MMAX
1238      DO 200 M=1,MMAX
1239      DO 180 I=M,NF,ISTEP
1240      J=I+MMAX
1241      DO 180 I1=1,2*NV,2
1242      TEMPR=RWORK(ICR)*FDATA(I1,J)-FSIGN*RWORK(ICI)*FDATA(I1+1,J)
1243      TEMPI=RWORK(ICR)*FDATA(I1+1,J)+FSIGN*RWORK(ICI)*FDATA(I1,J)
1244      FDATA(I1,J)=FDATA(I1,I)-TEMPR
1245      FDATA(I1+1,J)=FDATA(I1+1,I)-TEMPI
1246      FDATA(I1,I)=FDATA(I1,I)+TEMPR
1247      FDATA(I1+1,I)=FDATA(I1+1,I)+TEMPI
1248      180     CONTINUE
1249      ICR=ICR+2
1250      ICI=ICI+2
1251      200     CONTINUE
1252      MMAX=ISTEP
1253      GO TO 120
1254      ENDIF
1255      RETURN
1256      C
1257      C
1258      C — ENTER THE INITIALIZATION ROUTINE
1259      C
1260      1000    CONTINUE

```

RAM2D1 (version C)

```

1261 C
1262 C — IF NV = 8 OR LESS
1263 C
1264     IF (NV.LE.8) THEN
1265         CALL CFFT2(1,1,NF,RW1,RWORK,RW2)
1266         RETURN
1267     ENDIF
1268 C
1269 C — IF NV > 8
1270 C
1271     MMAX=1
1272     ICR=1
1273     ICI=2
1274     1120 CONTINUE
1275     IF (NF.GT.MMAX) THEN
1276         ISTEP=2*MMAX
1277         THETA=TWOPI/ISTEP
1278         WPR=-2.0*SIN(0.5*THETA)**2
1279         WPI=SIN(THETA)
1280         WR=1.0
1281         WI=0.0
1282         DO 1200 M=1,MMAX
1283             RWORK(ICR)=WR
1284             RWORK(ICI)=WI
1285             ICR=ICR+2
1286             ICI=ICI+2
1287             TEMP=WR
1288             WR=WR+WPR-WI*WPI+WR
1289             WI=WI+WPR+TEMP*WPI+WI
1290         1200 CONTINUE
1291         MMAX=ISTEP
1292         GO TO 1120
1293     ENDIF
1294     RETURN
1295     END

```

PRAM1 (version CD)

```

1      PROGRAM PRAM1CD
2      C
3      C
4      C This program was written by Godehard Hilfer (3/87). It generates
5      C contour and cross sectional plots from the data generated by the
6      C transient RAMAN amplifier code RAM2D1 written by Prof. Curtis R.
7      C Menyuk. To execute this program it has to be linked to the
8      C DISSPLA graphics package.
9      C
10     C This is version CD which is adapted for the Central Computing
11     C Facility Cray computer at the Naval Research Laboratory (Fall
12     C 1987). This version reads a record at a time and processes the
13     C field data contained in it. The field data of the next record
14     C that is being read over-writes the previous data in memory such
15     C as to minimize the memory requirements and to accommodate large
16     C dimensional field data arrays. This process entails large Input/
17     C Output transfer costs. Whenever possible, hence, version C should
18     C be used which stores all field data of one z-location. This is
19     C recommended particularly for one-dimensional transient (ny < 9)
20     C operation of the code.
21     C
22     C The program has the following structure:
23     C
24     C
25     C      F...      NPRAM1
26     C      Input data file      Input parameter file
27     C
28     C
29     C
30     C
31     C
32     C
33     C      PRAM1
34     C      (this program)
35     C
36     C      Main part
37     C
38     C      |
39     C      |
40     C      |
41     C      |
42     C      |
43     C      |
44     C      |
45     C      |
46     C      |
47     C      |
48     C      |
49     C      |
50     C      |
51     C      |
52     C      |
53     C      |
54     C      |
55     C      |
56     C      |
57     C      |
58     C      |
59     C      |
60     C      |
61     C      |
62     C      |
63     C

```

PRAM1 (version CD)

```

64 C
65 C
66 C ..... | PLT2.PLT | .....
67 C ..... | graphics output file | .....
68 C
69 C
70 C The program starts by setting default values for the graphics output
71 C parameters as specified in the data statements. These default values
72 C are updated by the values in the input file N'PRAM1 which allows
73 C format-free input through the two namelists condatt and zplot. The
74 C updated set is then written, depending on the value of the flag
75 C parameters lprmt, onto the first 4 graphics frames in the output file
76 C F3RAM00X. The value of lprmt(n) should be equal to 1 if the nth
77 C page of parameter output is desired, and equal to 0 if not
78 C
79 C Several constants are precalculated before the large DO-loop 500
80 C reads through and plots the data in file F... . Among these constants
81 C are the end values and interval sizes for the frequently plotted y
82 C and t coordinate axes.
83 C
84 C The following main part of this program acquires the electric field
85 C data from the input data file F... by reading sequentially the i-th
86 C record specified by the value i of the consecutive elements of the
87 C vector kz. These amplitude data are converted into intensity data
88 C if necessary and then handed through the arrays srl and srlf to the
89 C subroutine cntr (for contour plotting) and to the subroutine crsect
90 C (for cross sectional plots). The sequence of the resulting plots is
91 C as follows:
92 C      I contours pump intensity
93 C        1 sections pump intensity
94 C        2 sections pump phase
95 C        3 sections pump amplitude (real/imag)
96 C      II contours pump FFT intensity
97 C        4 sections pump FFT intensity
98 C        5 sections pump FFT phase
99 C        6 sections pump FFT amplitude (real/imag)
100 C     III contours Stokes intensity
101 C        7 sections Stokes intensity
102 C        8 sections Stokes phase
103 C        9 sections Stokes amplitude (real/imag)
104 C     IV contours Stokes FFT intensity
105 C       10 sections Stokes FFT intensity
106 C       11 sections Stokes FFT phase
107 C       12 sections Stokes FFT amplitude (real/imag)
108 C     V contours mat. exct. intensity
109 C       13 sections mat. exct. intensity
110 C       14 sections mat. exct. phase
111 C       15 sections mat. exct. amplitude (real/imag)
112 C     VI contours mat. exct. FFT intensity
113 C       16 sections mat. exct. FFT intensity
114 C       17 sections mat. exct. FFT phase
115 C       18 sections mat. exct. FFT amplitude (real/imag)
116 C     VII contours pump and Stokes intensity
117 C       19 sections sum of pump and Stokes intensity
118 C     VIII contours pump and Stokes FFT intensity
119 C
120 C The roman numerals tell which element of the vector isrlf is the
121 C flag that determines if that particular contour plot will be done
122 C or skipped:
123 C       isrlf(n) = 0 plot skipped
124 C       isrlf(n) = 1 plot drawn with labeled contours
125 C       isrlf(n) = -1 plot drawn; no labels on contours
126 C The arabic numerals of the sections indicate the row of the complex

```

PRAM1 (version CD)

```

127 C array cseC whith which this section is associated. The column number
128 C (second index) of the elements of cseC numbers the cross sectional
129 C plots of that particular surface. A maximum of nseC (< 9) cross
130 C sections of each surface can be drawn. The imaginary part of the
131 C elements of cseC determines: if =0.0 that this sectional plot is
132 C not requested
133 C     if =1.0 that this is a cross section parallel to the y-axis
134 C     of the surface under question at a fixed x-value as
135 C     given in real units by the real part of the
136 C     element of cseC; i.e. the first index of the
137 C     data array(s) srf(i) is being held constant for this
138 C     plot at the value lseC which is the grid point that
139 C     corresponds best to the fixed x-value;
140 C     if =2.0 that this is a cross section parallel to the x-axis
141 C     of the surface under question at a fixed y-value as
142 C     given in real units by the real part of the element
143 C     of cseC in question; i.e. the second index of the
144 C     data array(s) srf(i) is being held constant for this
145 C     plot at the value lseC which is the grid point that
146 C     corresponds best to the fixed y-value.
147 C In short: the imaginary part tells which variable to hold constant,
148 C and the real part tells at what value (in physical units).
149 C
150 C When one dimensional transient cases (ny.le.8) are being investigated
151 C the real part of the element of cseC under question has to be set
152 C equal to the number (1.0,through 8.0) of the element of the vector
153 C ltype in subroutine INIT in the code RAM2D1 in order for the
154 C sectional plot to contain the correct data and label of that
155 C particular case. Recall that the imaginary part has to be nonzero
156 C for the section to be drawn. Note that the sections #19 are sofar
157 C intended only for the check of the total electromagnetic intensity
158 C in the one-dimensional transient cases (ny.le.8). Set the real and
159 C imaginary part of the elements in row 19 of the array cseC as
160 C described above in this paragraph to obtain these total
161 C electromagnetic intensity sectional plots.
162 C
163 C More details on how the individual subroutines work precedes their
164 C listings. The contouring subroutine cntr makes use of the
165 C subroutine mycon which generates a customized dotted line for the
166 C half-height contour. The cross section subroutine crsct calls
167 C frequently on the subroutine nysxis which finds 'nice' values for
168 C coordinate axis limits and intervals. nysxis in turn uses
169 C subroutine powbas to find the next lower integral power of 10 for
170 C maximas and minimas. Both subroutines cntr and crsct share
171 C subroutine xisFFT when making secondary axes for FFT-plots.
172 C
173 C —variables—
174 C
175 C gain = see RAM2D1
176 C grfz = physical size of graphics plots
177 C i2 = y-coordinate index in do-loops 125,128,133,136,145,148,153,
178 C     156,165,168,173,176,185,188,193,196,205,208,213,225,228,
179 C     233,236,250,260
180 C i3 = t-coordinate index in same do-loops as i2
181 C icond = see RAM2D1
182 C iflip = 0/1 summand checks next row of cseC in do-loops 130,150,
183 C     170,190,210,230
184 C iln = number of dashed contours between solid contours in
185 C     sub=cntr
186 C is = cseC column index in do-loops 120,140,160,180,200,220;
187 C     dummy index in do-loops 130,150,170,190,210,230
188 C ishm = flag for half-height contour option in sub=cntr
189 C isrf = flag vector that indicates which contour plots are desired

```

PRAM1 (version CD)

```

190 C iss = cseC column index in do-loops 130,150,170,190,210,230
191 C jarf = signed contour plot index; >0 for contour labels, <0 no
192 C labels
193 C kz = vector contains the iteration numbers at which graphics
194 C plots are desired
195 C level = vector containing desired level heights for dashed
196 C contours
197 C lprmt = flag; if nonzero indicates list of parameters is desired
198 C necveC = data switch for subroutine nyaxis
199 C ndeC = desired number of solid contours representing powers of 10
200 C nhyp = see RAM2D1
201 C nmax = see RAM2D1; index limit in do-loop 500
202 C np = see RAM2D1
203 C npump = see RAM2D1
204 C nsC = cseC row index in do-loops 130,150,170,190,210,230
205 C nseC = number of elements tested in rows of csec
206 C nt = see RAM2D1
207 C ntp = nt+1
208 C nwrt = number of records in unit 4
209 C ny = see RAM2D1
210 C nyp = ny +1
211 C nyh = ny/2
212 C nyhp = nyh+1
213 C phi = see RAM2D1
214 C phst = see RAM2D1
215 C pi = 3.14159265358979
216 C r1 = intensity normalization factor 8*pi/speed
217 C rabamp = see RAM2D1
218 C ralasm = see RAM2D1
219 C ramasm = see RAM2D1
220 C ramp = see RAM2D1
221 C rdelim = see RAM2D1
222 C rdt = step size in time
223 C rdy = step size in transverse spatial variable y
224 C rint = see RAM2D1
225 C rist = see RAM2D1
226 C rkp = see RAM2D1
227 C rks = see RAM2D1
228 C sC = sum of imaginary parts of a row of csec; test variable
229 C sfe = see RAM2D1
230 C sinit = see RAM2D1
231 C speed = see RAM2D1
232 C sq = see RAM2D1
233 C srf = array of data from which contours and sections are plotted
234 C srfi = imaginary part of amplitude data for cross sectional plots
235 C sstep = see RAM2D1
236 C stot = see RAM2D1
237 C tm = see RAM2D1
238 C tm1 = time coordinate lower limit
239 C tm2 = time coordinate upper limit
240 C tmax = value at end of time axis
241 C toC = see RAM2D1
242 C toff = see RAM2D1
243 C torig = value at beginning of time axis
244 C tost = see RAM2D1
245 C tstp = time axis interval
246 C ttwo = see RAM2D1
247 C twidth = see RAM2D1
248 C twst = see RAM2D1
249 C wfmax = nice spatial FFT axis end value
250 C wforig = nice spatial FFT axis beginning value
251 C wfstp = nice spatial FFT axis interval
252 C yfmax = value at end of spatial FFT axis

```

PRAM1 (version CD)

```

253 C      yforig = value at beginning of spatial FFT axis
254 C      yfstp = spatial FFT axis interval
255 C      ym = see RAM2D1
256 C      ym1 = y-coordinate lower limit
257 C      ym2 = y-coordinate upper limit
258 C      ym2m1 = ym2-ym1
259 C      ymax = value at end of transverse spatial axis
260 C      yoff = see RAM2D1
261 C      yorig = value at beginning of transverse spatial axis
262 C      yost = see RAM2D1
263 C      ystp = transverse spatial axis interval
264 C      ywidth = see RAM2D1
265 C      ywst = see RAM2D1
266 C      zfinal = see RAM2D1
267 C      zint = see RAM2D1
268 C      zkeep = see RAM2D1
269 C      zstep = see RAM2D1
270 C      zval = value of z-coordinate of current data/plot
271 C *****
272 C      MODIFICATION 9/87:
273 C      THE DATA OUTPUT FILE NAME WAS CHANGED FROM 'FRAM' TO THE FOLLOWING:
274 C      THE DATA FILE NAME'S FIRST CHARACTER (F) STANDS FOR THE OLD DATA
275 C      FILE NAME 'FRAM'. THE SECOND CHARACTER INDICATES THE T-DIMENSION,
276 C      THE THIRD THE Y-DIMENSION. THE DIMENSIONS ARE REPRESENTED BY THEIR
277 C      NUMBER (1-8) IF LESS THAN 9. IF GREATER THAN 8 THE DIMENSIONS ARE
278 C      ASSUMED TO BE INTEGRAL POWERS OF 2. THE N-TH POWER OF 2 IS
279 C      REPRESENTED BY THE N-TH CHARACTER OF RLFBE. THE FOURTH THROUGH
280 C      NINTH CHARACTER IN THE FILE NAME ENCODES THE MONTH, DAY, AND YEAR
281 C      THE PROGRAM WAS STARTED. A TENTH THROUGH TWELFTH CHARACTER IS
282 C      APPENDED, NUMBERING THE PARTIAL DATA FILES THAT ARE GENERATED
283 C      WHEN THE PROGRAM RUNS TWO-DIMENSIONALLY.
284 C *****
285 C
286 C      PARAMETER (NP=10,NST=4000,NT=256,NTP=NT+1,NX=8,NXI=19*NX,NY=128,
287 C      1          NYH=NY/2,NYHP=NYH+1,NYP=NY+1,NZ=20)
288 C
289 C      IMPLICIT COMPLEX(A-E,Q)
290 C      DIMENSION INDEX(NP),ISRF(3),ISTAT(2),ITYPE(8),IWHEN(NYP),
291 C      1          IWORK(257),KZ(NZ),LEVEL(8),LPRMT(4),AEQ(NT,NY),
292 C      2          AER(NT,NY),CSEC(19,NX),PHL(NP),RABAMP(8),RAMP(NP),
293 C      3          RDSLM(8),RINT(NP),RTYPE(8),SFE(NST),SRF(NTP,NYP),
294 C      4          SRFI(NTP,NYP),SRTOF(1,NP),SQ(NST),SSTEP(NST),TIK(NY),
295 C      5          TM(2),TOFF(NP),TWIDTH(NP),YM(2),YOFF(NP),YWIDTH(NP)
296 C      CHARACTER*1 D1
297 C      CHARACTER*2 D1A
298 C      CHARACTER*2 D2
299 C      CHARACTER*2 D3
300 C      CHARACTER*7 DTFL1
301 C      CHARACTER*6 DTFL1D
302 C      CHARACTER*7 DTFL2D
303 C      CHARACTER*1 DUM1
304 C      CHARACTER*1 DUM2
305 C      CHARACTER*2 EDN
306 C      CHARACTER*9 FRAM
307 C      CHARACTER*6 FRM
308 C      CHARACTER*1 ISTEP1
309 C      CHARACTER*1 ISTEP2
310 C      CHARACTER*1 ISTEP3
311 C      CHARACTER*10 NUMRAL
312 C      CHARACTER*9 PDN1D
313 C      CHARACTER*12 PDN2D
314 C      CHARACTER*12 PDN0
315 C      CHARACTER*12 PDN1

```


PRAM1 (version CD)

```

316 CHARACTER*26 RLF BET
317 CHARACTER*1 TDIM
318 CHARACTER*1 YDIM
319 INTEGER DONYET, DAY, YEAR
320 NAMELIST /FLDATE/ DONYET, MONTH, DAY, YEAR, IPART, NEDN
321 NAMELIST /CONDAT/ ILN, ISHM, LEVEL, LPRMT, NDEC, NSEC, ISRF, CSEC
322 NAMELIST /ZPLOT/ KZ
323 COMMON /GRAPHS/ ILN, ISHM, ISRF, ITYPE, LEVEL, NDEC, NHYP, NSEC, CSEC,
324 1 GRFSZ, PI, RTYPE, SRF, SRFI, TMAX, TORG, TSTP, YFMAX, YFORIG,
325 2 YFSTP, YMAX, YORIG, YSTP, WFMAX, WFORIG, WFSTP, ZBOT, ZMAX, ZSTEP,
326 3 ZVAL
327 COMMON /NUM/ RDT, RDY, RDYF, TM1, TM2, YM1, YM2, YM2M1
328 EQUIVALENCE (YOFF, SRTYOF)
329 C
330 DATA PI/3.14159265358979/, SPEED/0.0299779/,
331 1 WDLIM/0.632120558828558/
332 DATA DONYET/1/, MONTH/09/, DAY/28/, YEAR/87/, IPART/002/, NEDN/01/
333 DATA ILN/8/, ISHM/0/, LEVEL/2,3,4,5,6,7,8,9/, LPRMT/4+1/, NDEC/2/,
334 1 NSEC/5/, ISRF/8+0/, CSEC/NXI*(0.0,0.0)/, GRFSZ/7.0/
335 DATA KZ/NZ+0/
336 C
337 CALL ASSIGN (IRRE, 'DN'L, 'NPRAM1'L, 'A'L, 'FT01'L)
338 CALL ASSIGN (IRRE, 'DN'L, 'EPRM'L, 'A'L, 'FT59'L)
339 READ (1, FLDATE)
340 WRITE (59, *) 'READ (1, FLDATE)'
341 WRITE (59, FLDATE)
342 READ (1, CONDAT)
343 WRITE (59, *) 'READ (1, CONDAT)'
344 WRITE (59, CONDAT)
345 READ (1, ZPLOT)
346 WRITE (59, *) 'READ (1, ZPLOT)'
347 WRITE (59, ZPLOT)
348 C
349 C - DETERMINE DATA FILE NAME
350 RLF BET='ABCDEFGH IJKLMN O PQRSTU VWXYZ'
351 C 12345678901234567890123456
352 NUMRAL='0123456789'
353 IF (NT.GT.0) THEN
354 ITDIM=NINT(ALOG(FLOAT(NT))/ALOG(2.0))
355 TDIM=RLF BET (ITDIM:ITDIM)
356 ELSE
357 TDIM=NUMRAL (NT+1:NT+1)
358 ENDIF
359 IF (NY.GT.0) THEN
360 IYDIM=NINT(ALOG(FLOAT(NY))/ALOG(2.0))
361 YDIM=RLF BET (IYDIM:IYDIM)
362 ELSE
363 YDIM=NUMRAL (NY+1:NY+1)
364 ENDIF
365 WRITE (59, *) 'ITDIM= ', ITDIM, ' TDIM= ', TDIM
366 WRITE (59, *) 'IYDIM= ', IYDIM, ' YDIM= ', YDIM
367 IF (MONTH.GT.0.AND.MONTH.LT.10) THEN
368 IDUM1=1
369 IDUM2=MONTH+1
370 D1=NUMRAL (IDUM2:IDUM2)
371 ELSEIF (MONTH.EQ.10) THEN
372 IDUM1=2
373 IDUM2=1
374 D1='A'
375 ELSEIF (MONTH.EQ.11) THEN
376 IDUM1=2
377 IDUM2=2
378 D1='B'

```

PRAM1 (version CD)

```

379     ELSEIF (MONTH.EQ.12) THEN
380         IDUM1=2
381         IDUM2=3
382         D1='C'
383     ELSE
384         WRITE (59,*) 'MONTH INPUT = ',MONTH,' IS OUT OF RANGE'
385         CALL EXIT(1)
386     ENDIF
387     DUM1=NUMRAL (IDUM1:IDUM1)
388     DUM2=NUMRAL (IDUM2:IDUM2)
389     D1A=DUM1//DUM2
390     IF (DAY.LT.1.OR.DAY.GT.31) THEN
391         WRITE (59,*) 'DAY INPUT = ',DAY,' IS OUT OF RANGE'
392         CALL EXIT(1)
393     ENDIF
394     IDUM1=INT(DAY/10)
395     IDUM2=DAY-10*IDUM1
396     DUM1=NUMRAL (IDUM1+1:IDUM1+1)
397     DUM2=NUMRAL (IDUM2+1:IDUM2+1)
398     D2=DUM1//DUM2
399     IF (YEAR.LT.0.OR.YEAR.GT.99) THEN
400         WRITE (59,*) 'YEAR INPUT = ',YEAR,' IS OUT OF RANGE'
401         CALL EXIT(1)
402     ENDIF
403     IDUM1=INT(YEAR/10)
404     IDUM2=YEAR-10*IDUM1
405     DUM1=NUMRAL (IDUM1+1:IDUM1+1)
406     DUM2=NUMRAL (IDUM2+1:IDUM2+1)
407     D3=DUM1//DUM2
408     FRM='F'//TDIM//YDIM//D1//D2
409     FRAM='F'//TDIM//YDIM//D1A//D2//D3
410     IDUM1=INT(NEDN/10)
411     IDUM2=NEDN-IDUM1*10
412     DUM1=NUMRAL (IDUM1+1:IDUM1+1)
413     DUM2=NUMRAL (IDUM2+1:IDUM2+1)
414     EDN=DUM1//DUM2
415     IUNIT=4
416     IF (NT.GT.8.AND.NY.GT.8) THEN
417         ISTEP1=NUMRAL(1:1)
418         IF (DONYET.EQ.0) THEN
419             ISTEP2=ISTEP1
420         ELSE
421             ISTEP2=NUMRAL(2:2)
422         ENDIF
423         DTFL1=FRM//ISTEP2
424         PDN1=FRAM//ISTEP1//ISTEP2
425         WRITE (59,*) 'DTFL1= ',DTFL1
426         WRITE (59,*) 'PDN1= ',PDN1
427         WRITE (59,*) 'NEDN,EDN= ',NEDN,EDN
428         CALL ACCESS(IRRE,'DN'L,DTFL1,'PDN'L,PDN1,'ED'L,EDN)
429         CALL ASSIGN(IRRE,'DN'L,DTFL1,'A'L,'FT03'L)
430         ISPCT=1
431         IUNIT=3
432         IF (DONYET.EQ.0) GO TO 40
433     ELSE
434         DTFL1D=FRM
435         PDN1D=FRAM
436         WRITE (59,*) 'DTFL1D= ',DTFL1D
437         WRITE (59,*) 'PDN1D= ',PDN1D
438         CALL ACCESS(IRRE,'DN'L,DTFL1D,'PDN'L,PDN1D,'ED'L,EDN)
439         CALL ASSIGN(IRRE,'DN'L,DTFL1D,'A'L,'FT04'L)
440     ENDIF
441 C

```

PRAM1 (version CD)

```

442 C — READ TIMING INFORMATION AND SET OF INPUT PARAMETERS
443 C
444 C - SKIP TO EOF IN UNIT IUNIT
445     CALL SKIPR(IUNIT,6*NST+3,ISTAT)
446     WRITE (59,*) 'SKIPPED ',ISTAT(1),' RECORDS AND ',ISTAT(2),' FILES
447     1 IN UNIT ',IUNIT,' '
448 C
449 C - BACKUP ONE RECORD IN UNIT IUNIT
450     CALL SKIPR(IUNIT,-1,ISTAT)
451     WRITE (59,*) 'SKIPPED BACK ',ISTAT(1),' RECORDS AND ',ISTAT(2),'
452     1 FILES IN UNIT ',IUNIT,' '
453 C
454 C - READ NUMBER OF RECORDS AND TIMING INFORMATION FROM FILE RAM2D1.FOR
455 C   AND REWIND DATA FILE
456     READ (IUNIT) NWRT,ZVAL,STOT,SINIT,SSTEP,SFE,SQ
457     WRITE (59,*) 'READ (IUNIT) NWRT,ZVAL,STOT,SINIT,SSTEP,SFE,SQ'
458     WRITE (59,*) 'NWRT,ZVAL,STOT,SINIT',NWRT,ZVAL,STOT,SINIT
459     WRITE (59,*) 'RAM2D1 RAN ',STOT,' SECONDS.'
460     REWIND IUNIT
461 40  CONTINUE
462 C
463 C - READ CODE INPUT PARAMETER
464     READ (IUNIT) NPUMP,YM,TM,ZINT,RKP,RKS,YOFF,TOFF,YWIDTH,TWIDTH,
465     1 YOST,TOST,YWST,TWST,RINT,RIST,RAMASM,RALASM,NHYP,PHL,PHST,
466     2 TOC,ICOND,ITYPE,RTYPE,RABAMP,RDSLIM,ZSTEP,ZFINAL,ZKEEP,
467     3 NMAX,TTWO,GAIN
468     WRITE (59,*) 'READ (IUNIT) NPUMP,YM,TM,ZINT...'
469     WRITE(59,*) 'NPUMP,YM,TM,ZINT,RKP,RKS,YOFF,TOFF,YWIDTH,TWIDTH,YOST,
470     1 YOST,YWST,TWST,RINT,RIST,RAMASM,RALASM,NHYP,PHL,PHST,TOC,
471     2 ICOND,ITYPE,RTYPE,RABAMP,RDSLIM,ZSTEP,ZFINAL,ZKEEP,NMAX,TTWO,
472     3 GAIN'
473     WRITE(59,*) 'NPUMP,YM,TM,ZINT,RKP,RKS,YOFF,TOFF,YWIDTH,TWIDTH,YOST,
474     1 YOST,YWST,TWST,RINT,RIST,RAMASM,RALASM,NHYP,PHL,PHST,TOC,
475     2 ICOND,ITYPE,RTYPE,RABAMP,RDSLIM,ZSTEP,ZFINAL,ZKEEP,NMAX,TTWO,
476     3 GAIN'
477 C
478 C - ERROR CONDITIONS
479     IF (NSEC.LT.1.OR.NSEC.GT.NX) THEN
480         WRITE (59,*) 'NSEC = ',NSEC,' IS OUT OF RANGE'
481         NSEC=NX
482     ENDIF
483     IF (NT.LE.8.AND.NY.LE.8) THEN
484         WRITE (59,*) 'NT AND NY BOTH LESS THAN 9; STOP'
485     ENDIF
486     IF (ICOND.NE.3.AND.NY.LE.8) THEN
487         WRITE (59,*) 'WHEN ICOND=3 NY MUST BE LESS THAN 9; STOP'
488         CALL EXIT(1)
489     ENDIF
490     IF (ICOND.NE.4.AND.NT.LE.8) THEN
491         WRITE (59,*) 'WHEN ICOND=4 NY MUST BE LESS THAN 8; STOP'
492         CALL EXIT(1)
493     ENDIF
494 C
495 C - RENAME CONSTANTS
496     TM1=TM(1)
497     TM2=TM(2)
498     YM1=YM(1)
499     YM2=YM(2)
500     YM2M1=YM2-YM1
501 C
502 C - INITIALIZE DISSPLA GRAPHICS
503     CALL COMPRS
504 C

```

PRAM1 (version CD)

```

505 C - LIST INPUT PARAMETERS ON 3 GRAPHICS FRAMES UPON REQUEST
506 C
507     IF (LPRMT(1).EQ.0) GO TO 80
508 C
509 C - FIRST FRAME OF PARAMETERS
510     CALL RESET('ALL')
511     CALL AREA2D(8.0,10.5)
512     CALL HEIGHT(0.17)
513     SLT=2.0
514     ZL=10.2
515     CALL MESSAG('LIST OF INPUT PARAMETERS$',100,SLT,ZL)
516     CALL RESET('HEIGHT')
517     SL1=1.8
518     SLT=0.0
519     ZL=9.5
520     CALL MESSAG('ICOND   = $',100,SLT,ZL)
521     CALL INTNO(ICOND,SL1,ZL)
522     IF (NT.GT.8.AND.NY.GT.8) THEN
523         ZL=ZL-0.3
524         CALL MESSAG('ILN    = $',100,SLT,ZL)
525         CALL INTNO(ILN,SL1,ZL)
526         ZL=ZL-0.3
527         CALL MESSAG('ISHM   = $',100,SLT,ZL)
528         CALL INTNO(ISHM,SL1,ZL)
529         ZL=ZL-0.3
530         CALL MESSAG('NDEC   = $',100,SLT,ZL)
531         CALL INTNO(NDEC,SL1,ZL)
532     ENDIF
533     IF (ICOND.EQ.2.OR.ICOND.EQ.4) THEN
534         ZL=ZL-0.3
535         CALL MESSAG('NHYP   = $',100,SLT,ZL)
536         CALL INTNO(NHYP,SL1,ZL)
537     ENDIF
538     ZL=ZL-0.3
539     CALL MESSAG('NMAX    = $',100,SLT,ZL)
540     CALL INTNO(NMAX,SL1,ZL)
541     ZL=ZL-0.3
542     CALL MESSAG('NPUMP   = $',100,SLT,ZL)
543     CALL INTNO(NPUMP,SL1,ZL)
544     NZLT=NZL+1
545     ZL=ZL-0.3
546     CALL MESSAG('NT      = $',100,SLT,ZL)
547     CALL INTNO(NT,SL1,ZL)
548     NZLT=NZL+1
549     ZL=ZL-0.3
550     CALL MESSAG('NY      = $',100,SLT,ZL)
551     CALL INTNO(NY,SL1,ZL)
552     ZL=ZL-0.3
553     CALL MESSAG('GAIN    = $',100,SLT,ZL)
554     CALL REALNO(GAIN,105,SL1,ZL)
555     IF (ICOND.EQ.2.OR.ICOND.EQ.3) THEN
556         ZL=ZL-0.3
557         CALL MESSAG('PHST   = $',100,SLT,ZL)
558         CALL REALNO(PHST,105,SL1,ZL)
559         ZL=ZL-0.3
560         CALL MESSAG('RALASM  = $',100,SLT,ZL)
561         CALL REALNO(RALASM,105,SL1,ZL)
562         ZL=ZL-0.3
563         CALL MESSAG('RAMASM  = $',100,SLT,ZL)
564         CALL REALNO(RAMASM,105,SL1,ZL)
565     ENDIF
566     ZL=ZL-0.3
567     CALL MESSAG('RIST    = $',100,SLT,ZL)

```

PRAM1 (version CD)

```

568 IPLACE=2
569 IF (ABS(RIST).GT.9999.0.OR.ABS(RIST).LT.0.01) IPLACE=-2
570 CALL REALNO(RIST,IPLACE,SL1,ZL)
571 ZL=ZL-0.3
572 CALL MESSAG('RKP      = $',100,SLT,ZL)
573 IPLACE=2
574 IF (ABS(RKP).GT.9999.0.OR.ABS(RKP).LT.0.01) IPLACE=-2
575 CALL REALNO(RKP,IPLACE,SL1,ZL)
576 ZL=ZL-0.3
577 CALL MESSAG('RKS      = $',100,SLT,ZL)
578 IPLACE=2
579 IF (ABS(RKS).GT.9999.0.OR.ABS(RKS).LT.0.01) IPLACE=-2
580 CALL REALNO(RKS,IPLACE,SL1,ZL)
581 IF (ICOND.EQ.2.OR.ICOND.EQ.3) THEN
582   ZL=ZL-0.3
583   CALL MESSAG('TOC      = $',100,SLT,ZL)
584   CALL REALNO(TOC,105,SL1,ZL)
585   ZL=ZL-0.3
586   CALL MESSAG('TOST     = $',100,SLT,ZL)
587   CALL REALNO(TOST,105,SL1,ZL)
588 ENDIF
589 ZL=ZL-0.3
590 CALL MESSAG('TTWO      = $',100,SLT,ZL)
591 CALL REALNO(TTWO,105,SL1,ZL)
592 IF (NT.GT.8) THEN
593   ZL=ZL-0.3
594   CALL MESSAG('TWST     = $',100,SLT,ZL)
595   CALL REALNO(TWST,105,SL1,ZL)
596 ENDIF
597 IF (NY.GT.8) THEN
598   ZL=ZL-0.3
599   CALL MESSAG('YOST      = $',100,SLT,ZL)
600   CALL REALNO(YOST,105,SL1,ZL)
601   ZL=ZL-0.3
602   CALL MESSAG('YWST      = $',100,SLT,ZL)
603   CALL REALNO(YWST,105,SL1,ZL)
604 ENDIF
605 ZL=ZL-0.3
606 CALL MESSAG('ZFINAL    = $',100,SLT,ZL)
607 CALL REALNO(ZFINAL,105,SL1,ZL)
608 IF (NY.GT.8) THEN
609   ZL=ZL-0.3
610   CALL MESSAG('ZINT      = $',100,SLT,ZL)
611   CALL REALNO(ZINT,105,SL1,ZL)
612 ENDIF
613 ZL=ZL-0.3
614 CALL MESSAG('ZKEEP      = $',100,SLT,ZL)
615 CALL REALNO(ZKEEP,105,SL1,ZL)
616 ZL=ZL-0.3
617 CALL MESSAG('ZSTEP      = $',100,SLT,ZL)
618 CALL REALNO(ZSTEP,105,SL1,ZL)
619 CALL ENDPL(0)
620 80 CONTINUE
621 IF (LPRMT(2).EQ.0) GO TO 85
622 C
623 C - SECOND FRAME OF PARAMETERS
624 CALL AREA2D(8.0,10.5)
625 ZL=10.2
626 CALL MESSAG('LIST OF INPUT PARAMETERS (CONTD)$',100,SLT,ZL)
627 SL1=2.2
628 SL2=2.9
629 SL3=3.6
630 SL4=4.3

```

PRAM1 (version CD)

```

631 SL5=5.0
632 SL6=5.7
633 SL7=6.4
634 SL8=7.1
635 ZL=9.5
636 IF (NT.GT.8.AND.NY.GT.8) THEN
637 CALL MESSAG('ISRF(1-8) = $',100,SLT,ZL)
638 CALL INTNO(ISRF(1),SL1,ZL)
639 CALL INTNO(ISRF(2),SL2,ZL)
640 CALL INTNO(ISRF(3),SL3,ZL)
641 CALL INTNO(ISRF(4),SL4,ZL)
642 CALL INTNO(ISRF(5),SL5,ZL)
643 CALL INTNO(ISRF(6),SL6,ZL)
644 CALL INTNO(ISRF(7),SL7,ZL)
645 CALL INTNO(ISRF(8),SL8,ZL)
646 ZL=ZL-0.3
647 CALL MESSAG('LEVEL = $',100,SLT,ZL)
648 CALL INTNO(LEVEL(1),SL1,ZL)
649 CALL INTNO(LEVEL(2),SL2,ZL)
650 CALL INTNO(LEVEL(3),SL3,ZL)
651 CALL INTNO(LEVEL(4),SL4,ZL)
652 CALL INTNO(LEVEL(5),SL5,ZL)
653 CALL INTNO(LEVEL(6),SL6,ZL)
654 CALL INTNO(LEVEL(7),SL7,ZL)
655 CALL INTNO(LEVEL(8),SL8,ZL)
656 ENDIF
657 IF (ICOND.EQ.3) THEN
658 ZL=ZL-0.3
659 CALL MESSAG('ITYPE = $',100,SLT,ZL)
660 CALL INTNO(ITYPE(1),SL1,ZL)
661 CALL INTNO(ITYPE(2),SL2,ZL)
662 CALL INTNO(ITYPE(3),SL3,ZL)
663 CALL INTNO(ITYPE(4),SL4,ZL)
664 CALL INTNO(ITYPE(5),SL5,ZL)
665 CALL INTNO(ITYPE(6),SL6,ZL)
666 CALL INTNO(ITYPE(7),SL7,ZL)
667 CALL INTNO(ITYPE(8),SL8,ZL)
668 ENDIF
669 SL1=2.1
670 SL2=3.2
671 SL3=4.3
672 SL4=5.4
673 SL5=6.5
674 IF (ICOND.EQ.2.OR.ICOND.EQ.3) THEN
675 ZL=ZL-0.3
676 CALL MESSAG('PHL(1-10) = $',100,SLT,ZL)
677 CALL REALNO(PHL(1),105,SL1,ZL)
678 CALL REALNO(PHL(2),105,SL2,ZL)
679 CALL REALNO(PHL(3),105,SL3,ZL)
680 CALL REALNO(PHL(4),105,SL4,ZL)
681 CALL REALNO(PHL(5),105,SL5,ZL)
682 ZL=ZL-0.3
683 CALL REALNO(PHL(6),105,SL1,ZL)
684 CALL REALNO(PHL(7),105,SL2,ZL)
685 CALL REALNO(PHL(8),105,SL3,ZL)
686 CALL REALNO(PHL(9),105,SL4,ZL)
687 CALL REALNO(PHL(10),105,SL5,ZL)
688 ENDIF
689 IF (ICOND.EQ.4) THEN
690 ZL=ZL-0.3
691 CALL MESSAG('RABAMP(1-8)= $',100,SLT,ZL)
692 CALL REALNO(RABAMP(1),105,SL1,ZL)
693 CALL REALNO(RABAMP(2),105,SL2,ZL)

```

PRAM1 (version CD)

```

694      CALL REALNO(RABAMP(3),105,SL3,ZL)
695      CALL REALNO(RABAMP(4),105,SL4,ZL)
696      CALL REALNO(RABAMP(5),105,SL5,ZL)
697      ZL=ZL-0.3
698      CALL REALNO(RABAMP(6),105,SL1,ZL)
699      CALL REALNO(RABAMP(7),105,SL2,ZL)
700      CALL REALNO(RABAMP(8),105,SL3,ZL)
701      ZL=ZL-0.3
702      CALL MESSAG('RDSLIM(1-8) = $',100,SLT,ZL)
703      CALL REALNO(RDSLIM(1),105,SL1,ZL)
704      CALL REALNO(RDSLIM(2),105,SL2,ZL)
705      CALL REALNO(RDSLIM(3),105,SL3,ZL)
706      CALL REALNO(RDSLIM(4),105,SL4,ZL)
707      CALL REALNO(RDSLIM(5),105,SL5,ZL)
708      ZL=ZL-0.3
709      CALL REALNO(RDSLIM(6),105,SL1,ZL)
710      CALL REALNO(RDSLIM(7),105,SL2,ZL)
711      CALL REALNO(RDSLIM(8),105,SL3,ZL)
712      ENDIF
713      ZL=ZL-0.3
714      CALL MESSAG('RINT(1-10) = $',100,SLT,ZL)
715      IPLACE=4
716      IF (ABS(RINT(1)).GT.9999.0.OR.ABS(RINT(1)).LT.0.01) IPLACE=-2
717      CALL REALNO(RINT(1),IPLACE,SL1,ZL)
718      IPLACE=4
719      IF (ABS(RINT(2)).GT.9999.0.OR.ABS(RINT(2)).LT.0.01) IPLACE=-2
720      CALL REALNO(RINT(2),IPLACE,SL2,ZL)
721      IPLACE=4
722      IF (ABS(RINT(3)).GT.9999.0.OR.ABS(RINT(3)).LT.0.01) IPLACE=-2
723      CALL REALNO(RINT(3),IPLACE,SL3,ZL)
724      IPLACE=4
725      IF (ABS(RINT(4)).GT.9999.0.OR.ABS(RINT(4)).LT.0.01) IPLACE=-2
726      CALL REALNO(RINT(4),IPLACE,SL4,ZL)
727      IPLACE=4
728      IF (ABS(RINT(5)).GT.9999.0.OR.ABS(RINT(5)).LT.0.01) IPLACE=-2
729      CALL REALNO(RINT(5),IPLACE,SL5,ZL)
730      ZL=ZL-0.3
731      IPLACE=4
732      IF (ABS(RINT(6)).GT.9999.0.OR.ABS(RINT(6)).LT.0.01) IPLACE=-2
733      CALL REALNO(RINT(6),IPLACE,SL1,ZL)
734      IPLACE=4
735      IF (ABS(RINT(7)).GT.9999.0.OR.ABS(RINT(7)).LT.0.01) IPLACE=-2
736      CALL REALNO(RINT(7),IPLACE,SL2,ZL)
737      IPLACE=4
738      IF (ABS(RINT(8)).GT.9999.0.OR.ABS(RINT(8)).LT.0.01) IPLACE=-2
739      CALL REALNO(RINT(8),IPLACE,SL3,ZL)
740      IPLACE=4
741      IF (ABS(RINT(9)).GT.9999.0.OR.ABS(RINT(9)).LT.0.01) IPLACE=-2
742      CALL REALNO(RINT(9),IPLACE,SL4,ZL)
743      IPLACE=4
744      IF (ABS(RINT(10)).GT.9999.0.OR.ABS(RINT(10)).LT.0.01) IPLACE=-2
745      CALL REALNO(RINT(10),IPLACE,SL5,ZL)
746      IF (ICOND.EQ.3) THEN
747      ZL=ZL-0.3
748      CALL MESSAG('RTYPE = $',100,SLT,ZL)
749      CALL REALNO(RTYPE(1),105,SL1,ZL)
750      CALL REALNO(RTYPE(2),105,SL2,ZL)
751      CALL REALNO(RTYPE(3),105,SL3,ZL)
752      CALL REALNO(RTYPE(4),105,SL4,ZL)
753      CALL REALNO(RTYPE(5),105,SL5,ZL)
754      ZL=ZL-0.3
755      CALL REALNO(RTYPE(6),105,SL1,ZL)
756      CALL REALNO(RTYPE(7),105,SL2,ZL)

```

PRAM1 (version CD)

```

757      CALL REALNO(RTYPE(8),105,SL3,ZL)
758  ENDIF
759  IF (NT.GT.8) THEN
760      ZL=ZL-0.3
761      CALL MESSAG('TM(1,2) = $',100,SLT,ZL)
762      CALL REALNO(TM1,105,SL1,ZL)
763      CALL REALNO(TM2,105,SL2,ZL)
764      ZL=ZL-0.3
765      CALL MESSAG('TOFF(1-10) = $',100,SLT,ZL)
766      CALL REALNO(TOFF(1),105,SL1,ZL)
767      CALL REALNO(TOFF(2),105,SL2,ZL)
768      CALL REALNO(TOFF(3),105,SL3,ZL)
769      CALL REALNO(TOFF(4),105,SL4,ZL)
770      CALL REALNO(TOFF(5),105,SL5,ZL)
771      ZL=ZL-0.3
772      CALL REALNO(TOFF(6),105,SL1,ZL)
773      CALL REALNO(TOFF(7),105,SL2,ZL)
774      CALL REALNO(TOFF(8),105,SL3,ZL)
775      CALL REALNO(TOFF(9),105,SL4,ZL)
776      CALL REALNO(TOFF(10),105,SL5,ZL)
777      ZL=ZL-0.3
778      CALL MESSAG('TWIDTH = $',100,SLT,ZL)
779      CALL REALNO(TWIDTH(1),105,SL1,ZL)
780      CALL REALNO(TWIDTH(2),105,SL2,ZL)
781      CALL REALNO(TWIDTH(3),105,SL3,ZL)
782      CALL REALNO(TWIDTH(4),105,SL4,ZL)
783      CALL REALNO(TWIDTH(5),105,SL5,ZL)
784      ZL=ZL-0.3
785      CALL REALNO(TWIDTH(6),105,SL1,ZL)
786      CALL REALNO(TWIDTH(7),105,SL2,ZL)
787      CALL REALNO(TWIDTH(8),105,SL3,ZL)
788      CALL REALNO(TWIDTH(9),105,SL4,ZL)
789      CALL REALNO(TWIDTH(10),105,SL5,ZL)
790  ENDIF
791  IF (NY.GT.8) THEN
792      ZL=ZL-0.3
793      CALL MESSAG('YOFF(1-10) = $',100,SLT,ZL)
794      CALL REALNO(YOFF(1),105,SL1,ZL)
795      CALL REALNO(YOFF(2),105,SL2,ZL)
796      CALL REALNO(YOFF(3),105,SL3,ZL)
797      CALL REALNO(YOFF(4),105,SL4,ZL)
798      CALL REALNO(YOFF(5),105,SL5,ZL)
799      ZL=ZL-0.3
800      CALL REALNO(YOFF(6),105,SL1,ZL)
801      CALL REALNO(YOFF(7),105,SL2,ZL)
802      CALL REALNO(YOFF(8),105,SL3,ZL)
803      CALL REALNO(YOFF(9),105,SL4,ZL)
804      CALL REALNO(YOFF(10),105,SL5,ZL)
805      ZL=ZL-0.3
806      CALL MESSAG('YM(1,2) = $',100,SLT,ZL)
807      CALL REALNO(YM1,105,SL1,ZL)
808      CALL REALNO(YM2,105,SL2,ZL)
809      ZL=ZL-0.3
810      CALL MESSAG('YWIDTH = $',100,SLT,ZL)
811      CALL REALNO(YWIDTH(1),105,SL1,ZL)
812      CALL REALNO(YWIDTH(2),105,SL2,ZL)
813      CALL REALNO(YWIDTH(3),105,SL3,ZL)
814      CALL REALNO(YWIDTH(4),105,SL4,ZL)
815      CALL REALNO(YWIDTH(5),105,SL5,ZL)
816      ZL=ZL-0.3
817      CALL REALNO(YWIDTH(6),105,SL1,ZL)
818      CALL REALNO(YWIDTH(7),105,SL2,ZL)
819      CALL REALNO(YWIDTH(8),105,SL3,ZL)

```


PRAM1 (version CD)

```

820          CALL REALNO(YWIDTH(9),105,SL4,ZL)
821          CALL REALNO(YWIDTH(10),105,SL5,ZL)
822      ENDIF
823      CALL ENDPL(0)
824      85      CONTINUE
825          IF (LPRMT(3).EQ.0) GO TO 95
826      C
827      C - THIRD FRAME OF PARAMETERS
828          CALL AREA2D(8.0,11.0)
829          ZL=10.2
830          CALL MESSAG('LIST OF INPUT PARAMETERS (CONTD)$',100,SLT,ZL)
831          ZL=9.8
832          CALL MESSAG('CSEC(1-19,1-8) = $',100,SLT,ZL)
833          SL1=0.1
834          SL2=1.0
835          SL3=1.9
836          SL4=2.8
837          SL5=3.7
838          SL6=4.6
839          SL7=5.5
840          SL8=6.4
841          ZL=ZL-0.3
842          CALL REALNO(REAL(CSEC(1,1)),104,SL1,ZL)
843          CALL REALNO(AIMAG(CSEC(1,1)),104,SL2,ZL)
844          CALL REALNO(REAL(CSEC(1,2)),104,SL3,ZL)
845          CALL REALNO(AIMAG(CSEC(1,2)),104,SL4,ZL)
846          CALL REALNO(REAL(CSEC(1,3)),104,SL5,ZL)
847          CALL REALNO(AIMAG(CSEC(1,3)),104,SL6,ZL)
848          CALL REALNO(REAL(CSEC(1,4)),104,SL7,ZL)
849          CALL REALNO(AIMAG(CSEC(1,4)),104,SL8,ZL)
850          ZL=ZL-0.2
851          CALL REALNO(REAL(CSEC(1,5)),104,SL1,ZL)
852          CALL REALNO(AIMAG(CSEC(1,5)),104,SL2,ZL)
853          CALL REALNO(REAL(CSEC(1,6)),104,SL3,ZL)
854          CALL REALNO(AIMAG(CSEC(1,6)),104,SL4,ZL)
855          CALL REALNO(REAL(CSEC(1,7)),104,SL5,ZL)
856          CALL REALNO(AIMAG(CSEC(1,7)),104,SL6,ZL)
857          CALL REALNO(REAL(CSEC(1,8)),104,SL7,ZL)
858          CALL REALNO(AIMAG(CSEC(1,8)),104,SL8,ZL)
859          ZL=ZL-0.3
860          CALL REALNO(REAL(CSEC(2,1)),104,SL1,ZL)
861          CALL REALNO(AIMAG(CSEC(2,1)),104,SL2,ZL)
862          CALL REALNO(REAL(CSEC(2,2)),104,SL3,ZL)
863          CALL REALNO(AIMAG(CSEC(2,2)),104,SL4,ZL)
864          CALL REALNO(REAL(CSEC(2,3)),104,SL5,ZL)
865          CALL REALNO(AIMAG(CSEC(2,3)),104,SL6,ZL)
866          CALL REALNO(REAL(CSEC(2,4)),104,SL7,ZL)
867          CALL REALNO(AIMAG(CSEC(2,4)),104,SL8,ZL)
868          ZL=ZL-0.2
869          CALL REALNO(REAL(CSEC(2,5)),104,SL1,ZL)
870          CALL REALNO(AIMAG(CSEC(2,5)),104,SL2,ZL)
871          CALL REALNO(REAL(CSEC(2,6)),104,SL3,ZL)
872          CALL REALNO(AIMAG(CSEC(2,6)),104,SL4,ZL)
873          CALL REALNO(REAL(CSEC(2,7)),104,SL5,ZL)
874          CALL REALNO(AIMAG(CSEC(2,7)),104,SL6,ZL)
875          CALL REALNO(REAL(CSEC(2,8)),104,SL7,ZL)
876          CALL REALNO(AIMAG(CSEC(2,8)),104,SL8,ZL)
877          ZL=ZL-0.3
878          CALL REALNO(REAL(CSEC(3,1)),104,SL1,ZL)
879          CALL REALNO(AIMAG(CSEC(3,1)),104,SL2,ZL)
880          CALL REALNO(REAL(CSEC(3,2)),104,SL3,ZL)
881          CALL REALNO(AIMAG(CSEC(3,2)),104,SL4,ZL)
882          CALL REALNO(REAL(CSEC(3,3)),104,SL5,ZL)

```

PRAM1 (version CD)

```

883 CALL REALNO(AIMAG(CSEC(3,3)),104,SL6,ZL)
884 CALL REALNO(REAL(CSEC(3,4)),104,SL7,ZL)
885 CALL REALNO(AIMAG(CSEC(3,4)),104,SL8,ZL)
886 ZL=ZL-0.2
887 CALL REALNO(REAL(CSEC(3,5)),104,SL1,ZL)
888 CALL REALNO(AIMAG(CSEC(3,5)),104,SL2,ZL)
889 CALL REALNO(REAL(CSEC(3,6)),104,SL3,ZL)
890 CALL REALNO(AIMAG(CSEC(3,6)),104,SL4,ZL)
891 CALL REALNO(REAL(CSEC(3,7)),104,SL5,ZL)
892 CALL REALNO(AIMAG(CSEC(3,7)),104,SL6,ZL)
893 CALL REALNO(REAL(CSEC(3,8)),104,SL7,ZL)
894 CALL REALNO(AIMAG(CSEC(3,8)),104,SL8,ZL)
895 ZL=ZL-0.3
896 CALL REALNO(REAL(CSEC(4,1)),104,SL1,ZL)
897 CALL REALNO(AIMAG(CSEC(4,1)),104,SL2,ZL)
898 CALL REALNO(REAL(CSEC(4,2)),104,SL3,ZL)
899 CALL REALNO(AIMAG(CSEC(4,2)),104,SL4,ZL)
900 CALL REALNO(REAL(CSEC(4,3)),104,SL5,ZL)
901 CALL REALNO(AIMAG(CSEC(4,3)),104,SL6,ZL)
902 CALL REALNO(REAL(CSEC(4,4)),104,SL7,ZL)
903 CALL REALNO(AIMAG(CSEC(4,4)),104,SL8,ZL)
904 ZL=ZL-0.2
905 CALL REALNO(REAL(CSEC(4,5)),104,SL1,ZL)
906 CALL REALNO(AIMAG(CSEC(4,5)),104,SL2,ZL)
907 CALL REALNO(REAL(CSEC(4,6)),104,SL3,ZL)
908 CALL REALNO(AIMAG(CSEC(4,6)),104,SL4,ZL)
909 CALL REALNO(REAL(CSEC(4,7)),104,SL5,ZL)
910 CALL REALNO(AIMAG(CSEC(4,7)),104,SL6,ZL)
911 CALL REALNO(REAL(CSEC(4,8)),104,SL7,ZL)
912 CALL REALNO(AIMAG(CSEC(4,8)),104,SL8,ZL)
913 ZL=ZL-0.3
914 CALL REALNO(REAL(CSEC(5,1)),104,SL1,ZL)
915 CALL REALNO(AIMAG(CSEC(5,1)),104,SL2,ZL)
916 CALL REALNO(REAL(CSEC(5,2)),104,SL3,ZL)
917 CALL REALNO(AIMAG(CSEC(5,2)),104,SL4,ZL)
918 CALL REALNO(REAL(CSEC(5,3)),104,SL5,ZL)
919 CALL REALNO(AIMAG(CSEC(5,3)),104,SL6,ZL)
920 CALL REALNO(REAL(CSEC(5,4)),104,SL7,ZL)
921 CALL REALNO(AIMAG(CSEC(5,4)),104,SL8,ZL)
922 ZL=ZL-0.2
923 CALL REALNO(REAL(CSEC(5,5)),104,SL1,ZL)
924 CALL REALNO(AIMAG(CSEC(5,5)),104,SL2,ZL)
925 CALL REALNO(REAL(CSEC(5,6)),104,SL3,ZL)
926 CALL REALNO(AIMAG(CSEC(5,6)),104,SL4,ZL)
927 CALL REALNO(REAL(CSEC(5,7)),104,SL5,ZL)
928 CALL REALNO(AIMAG(CSEC(5,7)),104,SL6,ZL)
929 CALL REALNO(REAL(CSEC(5,8)),104,SL7,ZL)
930 CALL REALNO(AIMAG(CSEC(5,8)),104,SL8,ZL)
931 ZL=ZL-0.3
932 CALL REALNO(REAL(CSEC(6,1)),104,SL1,ZL)
933 CALL REALNO(AIMAG(CSEC(6,1)),104,SL2,ZL)
934 CALL REALNO(REAL(CSEC(6,2)),104,SL3,ZL)
935 CALL REALNO(AIMAG(CSEC(6,2)),104,SL4,ZL)
936 CALL REALNO(REAL(CSEC(6,3)),104,SL5,ZL)
937 CALL REALNO(AIMAG(CSEC(6,3)),104,SL6,ZL)
938 CALL REALNO(REAL(CSEC(6,4)),104,SL7,ZL)
939 CALL REALNO(AIMAG(CSEC(6,4)),104,SL8,ZL)
940 ZL=ZL-0.2
941 CALL REALNO(REAL(CSEC(6,5)),104,SL1,ZL)
942 CALL REALNO(AIMAG(CSEC(6,5)),104,SL2,ZL)
943 CALL REALNO(REAL(CSEC(6,6)),104,SL3,ZL)
944 CALL REALNO(AIMAG(CSEC(6,6)),104,SL4,ZL)
945 CALL REALNO(REAL(CSEC(6,7)),104,SL5,ZL)

```

PRAM1 (version CD)

```

946 CALL REALNO(AIMAG(CSEC(6,7)),104,SL6,ZL)
947 CALL REALNO(REAL(CSEC(6,8)),104,SL7,ZL)
948 CALL REALNO(AIMAG(CSEC(6,8)),104,SL8,ZL)
949 ZL=ZL-0.3
950 CALL REALNO(REAL(CSEC(7,1)),104,SL1,ZL)
951 CALL REALNO(AIMAG(CSEC(7,1)),104,SL2,ZL)
952 CALL REALNO(REAL(CSEC(7,2)),104,SL3,ZL)
953 CALL REALNO(AIMAG(CSEC(7,2)),104,SL4,ZL)
954 CALL REALNO(REAL(CSEC(7,3)),104,SL5,ZL)
955 CALL REALNO(AIMAG(CSEC(7,3)),104,SL6,ZL)
956 CALL REALNO(REAL(CSEC(7,4)),104,SL7,ZL)
957 CALL REALNO(AIMAG(CSEC(7,4)),104,SL8,ZL)
958 ZL=ZL-0.2
959 CALL REALNO(REAL(CSEC(7,5)),104,SL1,ZL)
960 CALL REALNO(AIMAG(CSEC(7,5)),104,SL2,ZL)
961 CALL REALNO(REAL(CSEC(7,6)),104,SL3,ZL)
962 CALL REALNO(AIMAG(CSEC(7,6)),104,SL4,ZL)
963 CALL REALNO(REAL(CSEC(7,7)),104,SL5,ZL)
964 CALL REALNO(AIMAG(CSEC(7,7)),104,SL6,ZL)
965 CALL REALNO(REAL(CSEC(7,8)),104,SL7,ZL)
966 CALL REALNO(AIMAG(CSEC(7,8)),104,SL8,ZL)
967 ZL=ZL-0.3
968 CALL REALNO(REAL(CSEC(8,1)),104,SL1,ZL)
969 CALL REALNO(AIMAG(CSEC(8,1)),104,SL2,ZL)
970 CALL REALNO(REAL(CSEC(8,2)),104,SL3,ZL)
971 CALL REALNO(AIMAG(CSEC(8,2)),104,SL4,ZL)
972 CALL REALNO(REAL(CSEC(8,3)),104,SL5,ZL)
973 CALL REALNO(AIMAG(CSEC(8,3)),104,SL6,ZL)
974 CALL REALNO(REAL(CSEC(8,4)),104,SL7,ZL)
975 CALL REALNO(AIMAG(CSEC(8,4)),104,SL8,ZL)
976 ZL=ZL-0.2
977 CALL REALNO(REAL(CSEC(8,5)),104,SL1,ZL)
978 CALL REALNO(AIMAG(CSEC(8,5)),104,SL2,ZL)
979 CALL REALNO(REAL(CSEC(8,6)),104,SL3,ZL)
980 CALL REALNO(AIMAG(CSEC(8,6)),104,SL4,ZL)
981 CALL REALNO(REAL(CSEC(8,7)),104,SL5,ZL)
982 CALL REALNO(AIMAG(CSEC(8,7)),104,SL6,ZL)
983 CALL REALNO(REAL(CSEC(8,8)),104,SL7,ZL)
984 CALL REALNO(AIMAG(CSEC(8,8)),104,SL8,ZL)
985 ZL=ZL-0.3
986 CALL REALNO(REAL(CSEC(9,1)),104,SL1,ZL)
987 CALL REALNO(AIMAG(CSEC(9,1)),104,SL2,ZL)
988 CALL REALNO(REAL(CSEC(9,2)),104,SL3,ZL)
989 CALL REALNO(AIMAG(CSEC(9,2)),104,SL4,ZL)
990 CALL REALNO(REAL(CSEC(9,3)),104,SL5,ZL)
991 CALL REALNO(AIMAG(CSEC(9,3)),104,SL6,ZL)
992 CALL REALNO(REAL(CSEC(9,4)),104,SL7,ZL)
993 CALL REALNO(AIMAG(CSEC(9,4)),104,SL8,ZL)
994 ZL=ZL-0.2
995 CALL REALNO(REAL(CSEC(9,5)),104,SL1,ZL)
996 CALL REALNO(AIMAG(CSEC(9,5)),104,SL2,ZL)
997 CALL REALNO(REAL(CSEC(9,6)),104,SL3,ZL)
998 CALL REALNO(AIMAG(CSEC(9,6)),104,SL4,ZL)
999 CALL REALNO(REAL(CSEC(9,7)),104,SL5,ZL)
1000 CALL REALNO(AIMAG(CSEC(9,7)),104,SL6,ZL)
1001 CALL REALNO(REAL(CSEC(9,8)),104,SL7,ZL)
1002 CALL REALNO(AIMAG(CSEC(9,8)),104,SL8,ZL)
1003 ZL=ZL-0.3
1004 CALL REALNO(REAL(CSEC(10,1)),104,SL1,ZL)
1005 CALL REALNO(AIMAG(CSEC(10,1)),104,SL2,ZL)
1006 CALL REALNO(REAL(CSEC(10,2)),104,SL3,ZL)
1007 CALL REALNO(AIMAG(CSEC(10,2)),104,SL4,ZL)
1008 CALL REALNO(REAL(CSEC(10,3)),104,SL5,ZL)

```

PRAM1 (version CD)

```

1009 CALL REALNO(AIMAG(CSEC(10,3)),104,SL6,ZL)
1010 CALL REALNO(REAL(CSEC(10,4)),104,SL7,ZL)
1011 CALL REALNO(AIMAG(CSEC(10,4)),104,SL8,ZL)
1012 ZL=ZL-0.2
1013 CALL REALNO(REAL(CSEC(10,5)),104,SL1,ZL)
1014 CALL REALNO(AIMAG(CSEC(10,5)),104,SL2,ZL)
1015 CALL REALNO(REAL(CSEC(10,6)),104,SL3,ZL)
1016 CALL REALNO(AIMAG(CSEC(10,6)),104,SL4,ZL)
1017 CALL REALNO(REAL(CSEC(10,7)),104,SL5,ZL)
1018 CALL REALNO(AIMAG(CSEC(10,7)),104,SL6,ZL)
1019 CALL REALNO(REAL(CSEC(10,8)),104,SL7,ZL)
1020 CALL REALNO(AIMAG(CSEC(10,8)),104,SL8,ZL)
1021 ZL=ZL-0.3
1022 CALL REALNO(REAL(CSEC(11,1)),104,SL1,ZL)
1023 CALL REALNO(AIMAG(CSEC(11,1)),104,SL2,ZL)
1024 CALL REALNO(REAL(CSEC(11,2)),104,SL3,ZL)
1025 CALL REALNO(AIMAG(CSEC(11,2)),104,SL4,ZL)
1026 CALL REALNO(REAL(CSEC(11,3)),104,SL5,ZL)
1027 CALL REALNO(AIMAG(CSEC(11,3)),104,SL6,ZL)
1028 CALL REALNO(REAL(CSEC(11,4)),104,SL7,ZL)
1029 CALL REALNO(AIMAG(CSEC(11,4)),104,SL8,ZL)
1030 ZL=ZL-0.2
1031 CALL REALNO(REAL(CSEC(11,5)),104,SL1,ZL)
1032 CALL REALNO(AIMAG(CSEC(11,5)),104,SL2,ZL)
1033 CALL REALNO(REAL(CSEC(11,6)),104,SL3,ZL)
1034 CALL REALNO(AIMAG(CSEC(11,6)),104,SL4,ZL)
1035 CALL REALNO(REAL(CSEC(11,7)),104,SL5,ZL)
1036 CALL REALNO(AIMAG(CSEC(11,7)),104,SL6,ZL)
1037 CALL REALNO(REAL(CSEC(11,8)),104,SL7,ZL)
1038 CALL REALNO(AIMAG(CSEC(11,8)),104,SL8,ZL)
1039 ZL=ZL-0.3
1040 CALL REALNO(REAL(CSEC(12,1)),104,SL1,ZL)
1041 CALL REALNO(AIMAG(CSEC(12,1)),104,SL2,ZL)
1042 CALL REALNO(REAL(CSEC(12,2)),104,SL3,ZL)
1043 CALL REALNO(AIMAG(CSEC(12,2)),104,SL4,ZL)
1044 CALL REALNO(REAL(CSEC(12,3)),104,SL5,ZL)
1045 CALL REALNO(AIMAG(CSEC(12,3)),104,SL6,ZL)
1046 CALL REALNO(REAL(CSEC(12,4)),104,SL7,ZL)
1047 CALL REALNO(AIMAG(CSEC(12,4)),104,SL8,ZL)
1048 ZL=ZL-0.2
1049 CALL REALNO(REAL(CSEC(12,5)),104,SL1,ZL)
1050 CALL REALNO(AIMAG(CSEC(12,5)),104,SL2,ZL)
1051 CALL REALNO(REAL(CSEC(12,6)),104,SL3,ZL)
1052 CALL REALNO(AIMAG(CSEC(12,6)),104,SL4,ZL)
1053 CALL REALNO(REAL(CSEC(12,7)),104,SL5,ZL)
1054 CALL REALNO(AIMAG(CSEC(12,7)),104,SL6,ZL)
1055 CALL REALNO(REAL(CSEC(12,8)),104,SL7,ZL)
1056 CALL REALNO(AIMAG(CSEC(12,8)),104,SL8,ZL)
1057 ZL=ZL-0.3
1058 CALL REALNO(REAL(CSEC(13,1)),104,SL1,ZL)
1059 CALL REALNO(AIMAG(CSEC(13,1)),104,SL2,ZL)
1060 CALL REALNO(REAL(CSEC(13,2)),104,SL3,ZL)
1061 CALL REALNO(AIMAG(CSEC(13,2)),104,SL4,ZL)
1062 CALL REALNO(REAL(CSEC(13,3)),104,SL5,ZL)
1063 CALL REALNO(AIMAG(CSEC(13,3)),104,SL6,ZL)
1064 CALL REALNO(REAL(CSEC(13,4)),104,SL7,ZL)
1065 CALL REALNO(AIMAG(CSEC(13,4)),104,SL8,ZL)
1066 ZL=ZL-0.2
1067 CALL REALNO(REAL(CSEC(13,5)),104,SL1,ZL)
1068 CALL REALNO(AIMAG(CSEC(13,5)),104,SL2,ZL)
1069 CALL REALNO(REAL(CSEC(13,6)),104,SL3,ZL)
1070 CALL REALNO(AIMAG(CSEC(13,6)),104,SL4,ZL)
1071 CALL REALNO(REAL(CSEC(13,7)),104,SL5,ZL)

```

PRAM1 (version CD)

```

1072 CALL REALNO(AIMAG(CSEC(13,7)),104,SL6,ZL)
1073 CALL REALNO(REAL(CSEC(13,8)),104,SL7,ZL)
1074 CALL REALNO(AIMAG(CSEC(13,8)),104,SL8,ZL)
1075 ZL=ZL-0.3
1076 CALL REALNO(REAL(CSEC(14,1)),104,SL1,ZL)
1077 CALL REALNO(AIMAG(CSEC(14,1)),104,SL2,ZL)
1078 CALL REALNO(REAL(CSEC(14,2)),104,SL3,ZL)
1079 CALL REALNO(AIMAG(CSEC(14,2)),104,SL4,ZL)
1080 CALL REALNO(REAL(CSEC(14,3)),104,SL5,ZL)
1081 CALL REALNO(AIMAG(CSEC(14,3)),104,SL6,ZL)
1082 CALL REALNO(REAL(CSEC(14,4)),104,SL7,ZL)
1083 CALL REALNO(AIMAG(CSEC(14,4)),104,SL8,ZL)
1084 ZL=ZL-0.2
1085 CALL REALNO(REAL(CSEC(14,5)),104,SL1,ZL)
1086 CALL REALNO(AIMAG(CSEC(14,5)),104,SL2,ZL)
1087 CALL REALNO(REAL(CSEC(14,6)),104,SL3,ZL)
1088 CALL REALNO(AIMAG(CSEC(14,6)),104,SL4,ZL)
1089 CALL REALNO(REAL(CSEC(14,7)),104,SL5,ZL)
1090 CALL REALNO(AIMAG(CSEC(14,7)),104,SL6,ZL)
1091 CALL REALNO(REAL(CSEC(14,8)),104,SL7,ZL)
1092 CALL REALNO(AIMAG(CSEC(14,8)),104,SL8,ZL)
1093 ZL=ZL-0.3
1094 CALL REALNO(REAL(CSEC(15,1)),104,SL1,ZL)
1095 CALL REALNO(AIMAG(CSEC(15,1)),104,SL2,ZL)
1096 CALL REALNO(REAL(CSEC(15,2)),104,SL3,ZL)
1097 CALL REALNO(AIMAG(CSEC(15,2)),104,SL4,ZL)
1098 CALL REALNO(REAL(CSEC(15,3)),104,SL5,ZL)
1099 CALL REALNO(AIMAG(CSEC(15,3)),104,SL6,ZL)
1100 CALL REALNO(REAL(CSEC(15,4)),104,SL7,ZL)
1101 CALL REALNO(AIMAG(CSEC(15,4)),104,SL8,ZL)
1102 ZL=ZL-0.2
1103 CALL REALNO(REAL(CSEC(15,5)),104,SL1,ZL)
1104 CALL REALNO(AIMAG(CSEC(15,5)),104,SL2,ZL)
1105 CALL REALNO(REAL(CSEC(15,6)),104,SL3,ZL)
1106 CALL REALNO(AIMAG(CSEC(15,6)),104,SL4,ZL)
1107 CALL REALNO(REAL(CSEC(15,7)),104,SL5,ZL)
1108 CALL REALNO(AIMAG(CSEC(15,7)),104,SL6,ZL)
1109 CALL REALNO(REAL(CSEC(15,8)),104,SL7,ZL)
1110 CALL REALNO(AIMAG(CSEC(15,8)),104,SL8,ZL)
1111 ZL=ZL-0.3
1112 CALL REALNO(REAL(CSEC(16,1)),104,SL1,ZL)
1113 CALL REALNO(AIMAG(CSEC(16,1)),104,SL2,ZL)
1114 CALL REALNO(REAL(CSEC(16,2)),104,SL3,ZL)
1115 CALL REALNO(AIMAG(CSEC(16,2)),104,SL4,ZL)
1116 CALL REALNO(REAL(CSEC(16,3)),104,SL5,ZL)
1117 CALL REALNO(AIMAG(CSEC(16,3)),104,SL6,ZL)
1118 CALL REALNO(REAL(CSEC(16,4)),104,SL7,ZL)
1119 CALL REALNO(AIMAG(CSEC(16,4)),104,SL8,ZL)
1120 ZL=ZL-0.2
1121 CALL REALNO(REAL(CSEC(16,5)),104,SL1,ZL)
1122 CALL REALNO(AIMAG(CSEC(16,5)),104,SL2,ZL)
1123 CALL REALNO(REAL(CSEC(16,6)),104,SL3,ZL)
1124 CALL REALNO(AIMAG(CSEC(16,6)),104,SL4,ZL)
1125 CALL REALNO(REAL(CSEC(16,7)),104,SL5,ZL)
1126 CALL REALNO(AIMAG(CSEC(16,7)),104,SL6,ZL)
1127 CALL REALNO(REAL(CSEC(16,8)),104,SL7,ZL)
1128 CALL REALNO(AIMAG(CSEC(16,8)),104,SL8,ZL)
1129 ZL=ZL-0.3
1130 CALL REALNO(REAL(CSEC(17,1)),104,SL1,ZL)
1131 CALL REALNO(AIMAG(CSEC(17,1)),104,SL2,ZL)
1132 CALL REALNO(REAL(CSEC(17,2)),104,SL3,ZL)
1133 CALL REALNO(AIMAG(CSEC(17,2)),104,SL4,ZL)
1134 CALL REALNO(REAL(CSEC(17,3)),104,SL5,ZL)

```

PRAM1 (version CD)

```

1135 CALL REALNO(AIMAG(CSEC(17,3)),104,SL6,ZL)
1136 CALL REALNO(REAL(CSEC(17,4)),104,SL7,ZL)
1137 CALL REALNO(AIMAG(CSEC(17,4)),104,SL8,ZL)
1138 ZL=ZL-0.2
1139 CALL REALNO(REAL(CSEC(17,5)),104,SL1,ZL)
1140 CALL REALNO(AIMAG(CSEC(17,5)),104,SL2,ZL)
1141 CALL REALNO(REAL(CSEC(17,6)),104,SL3,ZL)
1142 CALL REALNO(AIMAG(CSEC(17,6)),104,SL4,ZL)
1143 CALL REALNO(REAL(CSEC(17,7)),104,SL5,ZL)
1144 CALL REALNO(AIMAG(CSEC(17,7)),104,SL6,ZL)
1145 CALL REALNO(REAL(CSEC(17,8)),104,SL7,ZL)
1146 CALL REALNO(AIMAG(CSEC(17,8)),104,SL8,ZL)
1147 ZL=ZL-0.3
1148 CALL REALNO(REAL(CSEC(18,1)),104,SL1,ZL)
1149 CALL REALNO(AIMAG(CSEC(18,1)),104,SL2,ZL)
1150 CALL REALNO(REAL(CSEC(18,2)),104,SL3,ZL)
1151 CALL REALNO(AIMAG(CSEC(18,2)),104,SL4,ZL)
1152 CALL REALNO(REAL(CSEC(18,3)),104,SL5,ZL)
1153 CALL REALNO(AIMAG(CSEC(18,3)),104,SL6,ZL)
1154 CALL REALNO(REAL(CSEC(18,4)),104,SL7,ZL)
1155 CALL REALNO(AIMAG(CSEC(18,4)),104,SL8,ZL)
1156 ZL=ZL-0.2
1157 CALL REALNO(REAL(CSEC(18,5)),104,SL1,ZL)
1158 CALL REALNO(AIMAG(CSEC(18,5)),104,SL2,ZL)
1159 CALL REALNO(REAL(CSEC(18,6)),104,SL3,ZL)
1160 CALL REALNO(AIMAG(CSEC(18,6)),104,SL4,ZL)
1161 CALL REALNO(REAL(CSEC(18,7)),104,SL5,ZL)
1162 CALL REALNO(AIMAG(CSEC(18,7)),104,SL6,ZL)
1163 CALL REALNO(REAL(CSEC(18,8)),104,SL7,ZL)
1164 CALL REALNO(AIMAG(CSEC(18,8)),104,SL8,ZL)
1165 ZL=ZL-0.3
1166 CALL REALNO(REAL(CSEC(19,1)),104,SL1,ZL)
1167 CALL REALNO(AIMAG(CSEC(19,1)),104,SL2,ZL)
1168 CALL REALNO(REAL(CSEC(19,2)),104,SL3,ZL)
1169 CALL REALNO(AIMAG(CSEC(19,2)),104,SL4,ZL)
1170 CALL REALNO(REAL(CSEC(19,3)),104,SL5,ZL)
1171 CALL REALNO(AIMAG(CSEC(19,3)),104,SL6,ZL)
1172 CALL REALNO(REAL(CSEC(19,4)),104,SL7,ZL)
1173 CALL REALNO(AIMAG(CSEC(19,4)),104,SL8,ZL)
1174 ZL=ZL-0.2
1175 CALL REALNO(REAL(CSEC(19,5)),104,SL1,ZL)
1176 CALL REALNO(AIMAG(CSEC(19,5)),104,SL2,ZL)
1177 CALL REALNO(REAL(CSEC(19,6)),104,SL3,ZL)
1178 CALL REALNO(AIMAG(CSEC(19,6)),104,SL4,ZL)
1179 CALL REALNO(REAL(CSEC(19,7)),104,SL5,ZL)
1180 CALL REALNO(AIMAG(CSEC(19,7)),104,SL6,ZL)
1181 CALL REALNO(REAL(CSEC(19,8)),104,SL7,ZL)
1182 CALL REALNO(AIMAG(CSEC(19,8)),104,SL8,ZL)
1183 CALL RESET('HEIGHT')
1184 CALL ENDPL(0)
1185 95 CONTINUE
1186 C
1187 C — PRECALCULATE 'NICE' END VALUES AND INTERVALS FOR FREQUENTLY USED
1188 C COORDINATE AXES, NUMERICAL STEP SIZE
1189 C
1190 C - T-AXIS
1191 IF (NT.GT.8) THEN
1192 TORIG=TM1
1193 TSTP=2.0
1194 TMAX=TM2
1195 NECLEC=-1
1196 CALL NYSXIS(TORIG,1,NECLEC,TORIG,TSTP,TMAX)
1197 RDT=(TM2-TM1)/NT

```

PRAM1 (version CD)

```

1198         ENDIF
1199         IF (NY.GT.8) THEN
1200     C
1201     C - Y-AXIS
1202         RDY=(YM2M1)/NY
1203         YORIG=YM1
1204         YSTP=2.0
1205         YMAX=YM2
1206         NECLEC=-1
1207         CALL NYSXIS(YORIG,1,NECLEC,YORIG,YSTP,YMAX)
1208     C
1209     C - FFT-AXIS
1210         YFMAX=0.5/RDY
1211         YFORIG=-YFMAX
1212         YFSTP=YFMAX-YFORIG
1213         RDYF=YFSTP/NY
1214         WFORIG=YFORIG
1215         WFSTP=0.0
1216         WFMAX=YFMAX
1217         NECLEC=-1
1218         CALL NYSXIS(WFORIG,1,NECLEC,WFORIG,WFSTP,WFMAX)
1219         WFSTP=2.0
1220         NECLEC=-1
1221         CALL NYSXIS(WFORIG,1,NECLEC,WFORIG,WFSTP,WFMAX)
1222         WFORIG=WFORIG+WFSTP
1223         WFMAX=WFMAX-WFSTP
1224         ENDIF
1225     C
1226     C - INTENSITY NORMALIZATION FACTOR
1227         R1=8.0*PI/SPEED
1228         R2=R1*YM2M1*YM2M1
1229     C
1230     C --- LOOP THROUGH: ALL RECORDS IN DATA FILE / ALL DATA FILES
1231     C
1232         IF (NT.GT.8.AND.NY.GT.8) THEN
1233             IF (DONYET.EQ.0) THEN
1234                 NWRT=4
1235                 NI0=2
1236             ELSE
1237                 NI0=(NWRT-2)/6
1238             ENDIF
1239         ELSE
1240             IF (NY.GT.8) THEN
1241                 NI0=(NWRT-2)/6
1242             ELSE
1243                 NI0=(NWRT-2)/3
1244             ENDIF
1245         ENDIF
1246         WRITE (59,*)'NI0= ',NI0
1247         IZ=1
1248         KZOLD=0
1249         KZNEW=KZ(1)
1250     C
1251         DO 500 I0=1,NI0
1252             WRITE (59,*)'I0= ',I0
1253             WRITE (59,*)'KZOLD= ',KZOLD
1254             WRITE (59,*)'KZNEW= ',KZNEW
1255             WRITE (59,*)'IZ= ',IZ
1256             IF (KZNEW.LE.KZOLD.AND.I0.GT.2) THEN
1257                 WRITE (59,*)'KZNEW .LE. KZOLD; STOP AT I0= ',I0
1258                 GOTO 501
1259             ENDIF
1260             IUNIT=4

```

PRAM1 (version CD)

```

1261       IF (NT.GT.8.AND.NY.GT.8.AND.KZNEW.EQ.1) IUNIT=3
1262       IF (I0.NE.KZNEW) THEN
1263   C
1264   C - NO PLOTTING AT CURRENT ZVAL, MOVE ON IN DATA FILE/S
1265       IF (NT.GT.8.AND.NY.GT.8) THEN
1266       IF (I0.GT.1) THEN
1267       ISPCT=ISPCT+1
1268       ENDIF
1269       GO TO 500
1270       ENDIF
1271       GO TO 290
1272   ENDIF
1273   IF (I0.NE.1) THEN
1274   IF (NT.GT.8.AND.NY.GT.8) THEN
1275   IF (DONYET.NE.0) THEN
1276   ISPCT=ISPCT+1
1277   ELSE
1278   ISPCT=IPART
1279   ENDIF
1280   INUMRL=INT(ISPCT/100)
1281   ISTEP1=NUMRAL (INUMRL+1:INUMRL+1)
1282   IRST=ISPCT-100*INUMRL
1283   INUMRL=INT(IRST/10)
1284   ISTEP2=NUMRAL (INUMRL+1:INUMRL+1)
1285   INUMRL=IRST-10*INUMRL
1286   ISTEP3=NUMRAL (INUMRL+1:INUMRL+1)
1287   DTFL2D=FRM//'.2'
1288   PDN2D=FRM//ISTEP1//ISTEP2//ISTEP3
1289   WRITE (59,*) 'DTFL2D ',DTFL2D
1290   WRITE (59,*) 'PDN2D= ',PDN2D
1291   CDIR$ BLOCK
1292       CALL ACCESS(IRRE,'DN'L,DTFL2D,'PDN'L,PDN2D,'ED'L,EDN)
1293       CALL ASSIGN(IRRE,'DN'L,DTFL2D,'A'L,'FT04'L)
1294       ENDIF
1295   ENDIF
1296   IF (KZNEW.GT.1.OR.LPRMT(4).NE.1.OR.NY.LE.8) GO TO 175
1297   C
1298   C - INITIALLY, WHEN NY.GT.8, COMPUTE DIVERSE WAVE NUMBER AVERAGES
1299   C AND WRITE THEM ONTO A SEPARATE GRAPHICS OUTPUT FRAME
1300       CALL AREA2D(8.0,10.5)
1301       SLT=2.0
1302       ZL=10.2
1303       CALL MESSAG('LIST OF OUTPUT PARAMETERS$',100,SLT,ZL)
1304       SLT=0.5
1305       ZL=ZL-0.7
1306       IF (NT.GT.8) THEN
1307   C
1308   C - READ INITIAL PUMP DATA, COMPUTE TOTAL PUMP ENERGY
1309       READ (IUNIT) ZVAL,AEQ
1310       WRITE (59,*) 'READ (IUNIT) ZVAL,AEQ'
1311       TPI=0.0
1312       DO 105 I2=1,NY
1313       DO 105 I3=1,NT
1314       TPI=TPI+AEQ(I3,I2)*CONJG(AEQ(I3,I2))
1315   105 CONTINUE
1316       TPI=TPI*RDY*RDT/R1
1317       CALL MESSAG('COMBINED LINEAR ENERGY DENSITY OF PUMPS IN MILLIJ
1318   10ULE/CM = $',100,SLT,ZL)
1319       SLTR=SLT+3.0
1320       ZL=ZL-0.35
1321       IPLACE=2
1322       IF (ABS(TPI).GT.9999.9.OR.ABS(TPI).LT.0.01) IPLACE=-2
1323       CALL REALNO(TPI,IPLACE,SLTR,ZL)

```


PRAM1 (version CD)

```

1324      WRITE (59,*) 'TOTAL LINEAR ENERGY DENSITY IN PUMPS IN ',
1325      1      'MILLIJOULES/CM = ', TPI
1326      C
1327      C - READ INITIAL STOKES DATA, COMPUTE TOTAL STOKES ENERGY
1328      READ (IUNIT) ZVAL,AEQ
1329      WRITE (59,*) 'READ (IUNIT) ZVAL,AEQ'
1330      TPI=0.0
1331      DO 107 I2=1,NY
1332      DO 107 I3=1,NT
1333      TPI=TPI+AEQ(I3,I2)*CONJG(AEQ(I3,I2))
1334      107      CONTINUE
1335      TPI=TPI*RDY*RT/R1
1336      ZL=ZL-0.5
1337      CALL MESSAG('COMBINED LINEAR ENERGY DENSITY OF STOKES IN MILLI
1338      1 JOULE/CM = $',100,SLT,ZL)
1339      IPLACE=2
1340      IF (ABS(TPI).GT.9999.9.OR.ABS(TPI).LT.0.01) IPLACE=-2
1341      ZL=ZL-0.35
1342      CALL REALNO(TPI,IPLACE,SLTR,ZL)
1343      WRITE (59,*) 'TOTAL LINEAR ENERGY DENSITY IN STOKES IN ',
1344      1      'MILLIJOULES/CM = ', TPI
1345      CALL SKIPR(IUNIT,1,ISTAT)
1346      WRITE (59,*) 'SKIPPED ',ISTAT(1),' RECORDS AND ',ISTAT(2),
1347      1      ' FILES IN UNIT ',IUNIT,' '
1348      ELSE
1349      CALL SKIPR(IUNIT,3,ISTAT)
1350      WRITE (59,*) 'SKIPPED ',ISTAT(1),' RECORDS AND ',ISTAT(2),
1351      1      ' FILES IN UNIT ',IUNIT,' '
1352      ENDIF
1353      C
1354      C - INITIAL PUMP BEAMS FFT INTENSITY DATA, READ PUMP FFT DATA AND
1355      C RESET FILE POINTER
1356      READ (IUNIT) ZVAL,AEQ
1357      WRITE (59,*) 'READ (IUNIT) ZVAL,AEQ'
1358      CALL SKIPR(IUNIT,-4,ISTAT)
1359      WRITE (59,*) 'SKIPPED BACK ',ISTAT(1),' RECORDS AND ',ISTAT(2),
1360      1      ' FILES IN UNIT ',IUNIT,' '
1361      DO 110 I2=1,NY
1362      DO 110 I3=1,NT
1363      SRFI(I3,I2)=AEQ(I3,I2)*CONJG(AEQ(I3,I2))/R2
1364      110      CONTINUE
1365      C
1366      C - DETERMINE INITIAL SEQUENCE OF INDICES OF PUMP BEAMS ALONG Y-AXIS
1367      C FROM LEFT TO RIGHT
1368      IF (NPUMP.EQ.1) INDEX(1)=1
1369      IF (NPUMP.EQ.2) THEN
1370      IF (YOFF(1).LT.YOFF(2)) THEN
1371      INDEX(1)=1
1372      INDEX(2)=2
1373      ELSE
1374      INDEX(1)=2
1375      INDEX(2)=1
1376      ENDIF
1377      ELSE IF (NPUMP.GT.2) THEN
1378      MODE=2
1379      CALL ORDERS(MODE,IWORK,SRTYOF,INDEX,NPUMP,1,8,1)
1380      ENDIF
1381      WRITE (59,*) 'PUMP INDICES SEQUENTIALLY',INDEX
1382      C
1383      C - ERROR CONDITIONS AND WARNINGS
1384      IF (YM(1).GT.YOFF(INDEX(1))) THEN
1385      WRITE (59,*) 'FIRST BEAM OUTSIDE Y-WINDOW; STOP'
1386      CALL EXIT(1)

```

PRAM1 (version CD)

```

1387     ENDIF
1388     IF (YM(2).LT.YOFF(INDEX(NPUMP))) THEN
1389         WRITE (59,*) 'LAST BEAM OUTSIDE Y-WINDOW; STOP'
1390         CALL EXIT(1)
1391     ENDIF
1392     IF (NPUMP.GT.1) THEN
1393         DO 120 INP=1,NPUMP-1
1394             IB=INDEX(INP)
1395             IBNX=INDEX(INP+1)
1396             IF (YOFF(IBNX)-YOFF(IB).LE.YWIDTH(IBNX)+YWIDTH(IB)) THEN
1397                 WRITE (59,*) 'BEAM ',IB,' AND BEAM ',IBNX,
1398                     1 ' ARE TOO CLOSE FOR AVERAGE K CALCULATIONS'
1399             ENDIF
1400             RIB=RINT(IBNX)/RINT(IB)
1401             IF (RIB.LT.0.1.OR.RIB.GT.10.0) THEN
1402                 WRITE (59,*) 'DISAPPEAR INTENSITIES OF BEAM ',IB,' AND ',
1403                     1 IBNX,'MAY OBSCURE SIGNATURES IN AVRG. K CALCS.'
1404             ENDIF
1405             CONTINUE
1406         ENDIF
1407     C
1408     C - CONSIDER EACH BEAM AT TIME TOFF WHEN 2-D; CONSIDER ALL NT CASES
1409     C WHEN 1-D (STATIONARY)
1410         SL=SL+0.5
1411         ZL=ZL-0.5
1412         CALL MESSAG('PUMP$',100,SLT,ZL)
1413         SLT=SLT+1.2
1414         CALL MESSAG('TOTAL INTENSITY$',100,SLT,ZL)
1415         SLT=SLT+2.6
1416         CALL MESSAG('K-WIDTH$',100,SLT,ZL)
1417         IFRM=0
1418         NCASES=NT
1419         IF (NT.GT.8) NCASES=1
1420         DO 170 ICS=1,NCASES
1421             IF (NCASES.GT.1) THEN
1422                 IF (ZL.LT.NPUMP*0.3+0.5) THEN
1423                     IFRM=IFRM+1
1424                     CALL ENDPL(0)
1425                     CALL AREA2D(8.0,10.5)
1426                     ZL=10.2
1427                     CALL MESSAG('LIST OF OUTPUT PARAMETERS (CONTD)$',
1428                         1 100,SLT,ZL)
1429                     ZL=9.5
1430                     CALL MESSAG('PUMP$',100,SLT,ZL)
1431                     SLT=SLT+1.2
1432                     CALL MESSAG('TOTAL INTENSITY$',100,SLT,ZL)
1433                     SLT=SLT+2.6
1434                     CALL MESSAG('K-WIDTH$',100,SLT,ZL)
1435                     IF (IFRM.GT.8) THEN
1436                         WRITE (59,*) 'LIST OF OUTPUT PARAMETER INTERRUPTED'
1437                         GO TO 170
1438                     ENDIF
1439                 ENDIF
1440                 SLT=0.1
1441                 ZL=ZL-0.4
1442                 CALL MESSAG('CASE$',100,SLT,ZL)
1443                 SLT=SLT+0.8
1444                 CALL INTNO(ICS,SLT,ZL)
1445             ENDIF
1446             IT=ICS
1447             ZL=ZL-0.1
1448     C
1449     C - CONSIDER ONE BEAM AFTER THE OTHER (ALONG K-AXIS FROM RIGHT TO LEFT)

```

PRAM1 (version CD)

```

1450      DO 160 INP=1,NPUMP
1451      IB=INDEX(INP)
1452      C
1453      C - RIGHT BEAM LIMIT
1454      IF (INP.EQ.1) THEN
1455          JR=NY
1456      ELSE
1457          JR=JL
1458      ENDIF
1459      C
1460      C - LEFT BEAM LIMIT
1461      IF (INP.EQ.NPUMP) THEN
1462          JL=1
1463      ELSE
1464          IBNX=INDEX(INP+1)
1465          YKL=-0.5*(YOFF(IB)+YOFF(IBNX))*RKP/ZINT
1466          JL=ANINT((YKL-YFORIG)*YM2M1)
1467      ENDIF
1468      C
1469      C - SELECT TEMPORAL PEAK OF EACH BEAM IN 2-D CASES
1470      IF (NT.GT.8) THEN
1471          TOFIB=TOFF(IB)
1472          IF (TOFIB.LT.TM1.OR.TO FIB.GT.TM2) THEN
1473              WRITE (59,*) 'BEAM ',IB,' IS OUTSIDE T-RANGE'
1474              GO TO 160
1475          ENDIF
1476          IT=ANINT(NT*(TOFIB-TM1)/(TM2-TM1))
1477      ENDIF
1478      C
1479      C - COMPUTE TOTAL INTENSITY OF BEAM IN K-SPACE
1480      TIK(JL)=0.0
1481      DO 140 J=JL+1,JR
1482          TIK(J)=TIK(J-1)+SRFI(IT,J)*RDYF
1483      140 CONTINUE
1484      WRITE (59,*) 'TIK = ',TIK
1485      TIKBMX=TIK(JR)
1486      WRITE (59,*) 'JL= ',JL,' JR= ',JR,' TIKBMX= ',TIKBMX,' IT= ',IT
1487      ZL=ZL-0.3
1488      SLT=0.7
1489      CALL INTNO(IB,SLT,ZL)
1490      SLT=SLT+1.3
1491      IPLACE=2
1492      IF (ABS(TIKBMX).GT.9999.0.OR.ABS(TIKBMX).LT.0.01) IPLACE=-2
1493      CALL REALNO(TIKBMX,IPLACE,SLT,ZL)
1494      WRITE (59,*) 'TOTAL INTENSITY IN K= ',TIKBMX
1495      C
1496      C - COMPUTE K-WIDTH OF BEAM (LINEAR INTERPOLATION)
1497      IF (TIKBMX.GT.1.0E-64) THEN
1498          DO 150 J=JL,JR
1499              TIK(J)=ABS((2.0*TIK(J)-TIKBMX)/TIKBMX)-WDLIM
1500      150 CONTINUE
1501      ELSE
1502          WRITE (59,*) 'POWER INTERGRAL IN K-SPACE VANISHES'
1503      ENDIF
1504      JSRCH=JR-JL+1
1505      CALL WHENFLE(JSRCH,TIK(JL),1,0.0,IWHEN,NVAL)
1506      WRITE (59,*) 'IWHEN = ',IWHEN
1507      IWN1=IWHEN(1)+JL-1
1508      IWN2=IWHEN(NVAL)+JL-1
1509      IF (NPUMP.GT.1.AND.(IWN1.EQ.1.OR.IWN2.EQ.NY)) THEN
1510          WRITE (59,*) 'AVERAGE K CALCULATION FAILED'
1511          GOTO 170
1512      ENDIF

```

PRAM1 (version CD)

```

1513 IF (IWN2-IWN1.LT.4) THEN
1514 WRITE (59,*) 'INSUFFICIENT RESOLUTION FFT BEAM ',IB
1515 GOTO 170
1516 ENDIF
1517 TIKDL1=TIK(IWN1-1)-TIK(IWN1)
1518 TIKDL2=TIK(IWN2+1)-TIK(IWN2)
1519 WRITE (59,*) 'TIKDL1= ',TIKDL1,' TIKDL2= ',TIKDL2
1520 YKL=YFORIG+RDYF*(IWN1+TIK(IWN1))/(TIK(IWN1-1)-TIK(IWN1)))
1521 YKR=YFORIG+RDYF*(IWN2-TIK(IWN2))/(TIK(IWN2+1)-TIK(IWN2)))
1522 WDHKB=YKR-YKL
1523 SLT=SLT+2.4
1524 IPLACE=2
1525 IF (ABS(WDHKB).GT.9999.0.OR.ABS(WDHKB).LT.0.01) IPLACE=-2
1526 CALL REALNO(WDHKB,IPLACE,SLT,ZL)
1527 160 CONTINUE
1528 170 CONTINUE
1529 CALL ENDPL(0)
1530 175 CONTINUE
1531 C
1532 C --- GENERATE DESIRED GRAPHICS DATA AND CALL PLOTTING SUBROUTINES
1533 C
1534 IPLTGP=1
1535 IF (NY.LE.8) IPLTGP=2
1536 DO 220 IPLT=1,8,IPLTGP
1537 C
1538 C - CHECK WHICH GRAPHS ARE REQUESTED
1539 IF (NY.GT.8.AND.NT.GT.8) THEN
1540 JSRF=ISRF(IPLT)
1541 ELSE
1542 JSRF=0
1543 ENDIF
1544 SCI=0.0
1545 LCSEC=3*(IPLT-1)+1
1546 DO 180 IS=1,NSEC
1547 SCI=SCI+ABS(AIMAG(CSEC(LCSEC,IS)))
1548 180 CONTINUE
1549 SCA=0.0
1550 ICSEC=LCSEC+1
1551 DO 190 IS=1,2*NSEC
1552 IFLIP=INT((IS-1)/NSEC)
1553 NSC=LCSEC+IFLIP
1554 ISS=IS-IFLIP*NSEC
1555 SCA=SCA+ABS(AIMAG(CSEC(NSC,ISS)))
1556 190 CONTINUE
1557 C
1558 C - WHEN A GRAPH IS REQUESTED READ ITS AMPLITUDE DATA IN AND RESET
1559 C FILE POINTER TO FIRST OF THE RECORDS WITH THE CURRENT ZVAL
1560 WRITE (59,*) 'SCI,SCA,I0,IPLT,LCSEC,JSRF,IZ,KZNEW,NWRT'
1561 WRITE (59,*) SCI,SCA,I0,IPLT,LCSEC,JSRF,IZ,KZNEW,NWRT
1562 IF (SCI.GT.0.001.OR.SCA.GT.0.001.OR.JSRF.NE.0) THEN
1563 C
1564 C - MATCH UP EL,ES,Q,AEL,AES,AQ STORAGE WITH EL,AEL,ES,AES,Q,AQ
1565 C PLOTTING
1566 IF (IPLT.EQ.1) IRCD=1
1567 IF (IPLT.EQ.2) IRCD=4
1568 IF (IPLT.EQ.3) IRCD=2
1569 IF (IPLT.EQ.4) IRCD=5
1570 IF (IPLT.EQ.5) IRCD=3
1571 IF (IPLT.EQ.6) IRCD=6
1572 C IRCD=1+MOD(IPLT+2*(IPLT-1)+INT(IPLT/2)-1,6)
1573 CALL SKIPR(IUNIT,IRCD-1,ISTAT)
1574 WRITE (59,*) 'SKIPPED ',ISTAT(1),' RECORDS AND ',ISTAT(2),
1575 1 ' FILES IN UNIT ',IUNIT

```

PRAM1 (version CD)

```

1576      READ (IUNIT) ZVAL,AEQ
1577      WRITE (59,*) 'READ (IUNIT) ZVAL,AEQ'
1578      CALL SKIPR(IUNIT,-IRCD,ISTAT)
1579      WRITE (59,*) 'SKIPPED BACK ',ISTAT(1),' RECORDS AND ',ISTAT(2),
1580      1 ' FILES IN UNIT ',IUNIT,' '
1581      ELSE
1582      GO TO 220
1583      ENDIF
1584  C - INTENSITY CONTOURS
1585      IF (JSRF.NE.0) THEN
1586      DO 200 I2=1,NY
1587      DO 200 I3=1,NT
1588      SRF(I3,I2)=AEQ(I3,I2)*CONJG(AEQ(I3,I2))/R1
1589      200 CONTINUE
1590      CALL CNTR(IPLT*JSRF)
1591      ENDIF
1592  C
1593  C - INTENSITY SECTIONS
1594      IF (SCI.GT.0.001) THEN
1595      X1=R1
1596      IF (IPLT.EQ.2.OR.IPLT.EQ.4.OR.IPLT.EQ.6) X1=R2
1597      DO 205 I2=1,NY
1598      DO 205 I3=1,NT
1599      SRF(I3,I2)=AEQ(I3,I2)*CONJG(AEQ(I3,I2))/X1
1600      205 CONTINUE
1601      CALL CRSSCT(LCSEC-1)
1602      ENDIF
1603  C
1604  C - LOAD PHASE AND AMPLITUDE DATA
1605      IF (SCA.GT.0.001) THEN
1606      DO 210 I2=1,NY
1607      DO 210 I3=1,NT
1608      SRF(I3,I2)=REAL(AEQ(I3,I2))
1609      SRFI(I3,I2)=AIMAG(AEQ(I3,I2))
1610      210 CONTINUE
1611  C
1612  C - PHASE AND AMPLITUDE SECTIONS
1613      CALL CRSSCT(LCSEC)
1614      ENDIF
1615      220 CONTINUE
1616  C
1617  C — PLOTS WITH SUM OF THE INTENSITIES OF PUMP BEAMS AND STOKES BEAM
1618  C AND LONGITUDINAL INVARIANTS
1619  C
1620      SC=0.0
1621      DO 230 IS=1,NSEC
1622      SC=SC+ABS(AIMAG(CSEC(10,IS)))
1623      230 CONTINUE
1624  C
1625  C - DATA OF PUMPS AND STOKES INTENSITY COMBINED
1626      IFLG=0
1627      IF (ISRF(7).NE.0.AND.NT.GT.8.AND.NY.GT.8) THEN
1628      JSRF=ISRF(7)
1629      READ (IUNIT) ZVAL,AEQ
1630      READ (IUNIT) ZVAL,AER
1631      WRITE (59,*) 'READ (IUNIT) ZVAL,AEQ'
1632      WRITE (59,*) 'READ (IUNIT) ZVAL,AER'
1633      IFLG=1
1634      CALL SKIPR(IUNIT,-2,ISTAT)
1635      WRITE (59,*) 'SKIPPED BACK ',ISTAT(1),' RECORDS AND ',
1636      1 ISTAT(2),' FILES IN UNIT ',IUNIT,' '
1637      DO 250 I2=1,NY
1638      DO 250 I3=1,NT

```

PRAM1 (version CD)

```

1639      SRF(13,12)=(AEQ(13,12)*CONJG(AEQ(13,12))
1640      +AER(13,12)*CONJG(AER(13,12)) )/R1
1641 250 1 CONTINUE
1642      CALL CNTR(7*JSRF)
1643      ENDIF
1644 C
1645 C - DATA OF INVARIANT ALONG Z
1646      IF (SC.GT.0.001) THEN
1647 C
1648 C - LONGITUDINAL INVARIANT IN 1-D TRANSIENT CASE
1649      IF (IFLG.EQ.0) THEN
1650          READ (IUNIT) ZVAL,AEQ
1651          READ (IUNIT) ZVAL,AER
1652          WRITE (59,*) 'READ (IUNIT) ZVAL,AEQ'
1653          WRITE (59,*) 'READ (IUNIT) ZVAL,AER'
1654          CALL SKIPR(IUNIT,-2,ISTAT)
1655          WRITE (59,*) 'SKIPPED BACK ',ISTAT(1),' RECORDS AND ',
1656          1 ISTAT(2),' FILES IN UNIT ',IUNIT,' '
1657      ENDIF
1658      DO 255 I2=1,NY
1659      DO 255 I3=1,NT
1660          SRF(13,12)=(RKS*AEQ(13,12)*CONJG(AEQ(13,12))
1661          +RKP*AER(13,12)*CONJG(AER(13,12)) )/R1
1662 255 1 CONTINUE
1663      IF (NY.GT.8) THEN
1664 C
1665 C - LONGITUDINAL INVARIANT IN 1-D STATIONARY CASE AND IN 2-D
1666      DO 257 I3=1,NT
1667          SRF(13,1)=0.0
1668 257 CONTINUE
1669      DO 258 I2=2,NY
1670      DO 258 I3=1,NT
1671          SRF(13,12)=SRF(13,12)*RDY+SRF(13,12-1)
1672 258 CONTINUE
1673      ENDIF
1674      CALL CRSSCT(19)
1675      ENDIF
1676      IF (ISRF(8).NE.0.AND.NT.GT.8.AND.NY.GT.8) THEN
1677          JSRF=ISRF(8)
1678 C
1679 C - DATA OF PUMPS AND STOKES FFT INTENSITY COMBINED
1680      CALL SKIPR(IUNIT,3,ISTAT)
1681      WRITE (59,*) 'SKIPPED ',ISTAT(1),' RECORDS AND ',ISTAT(2),
1682      1 ' FILES IN UNIT ',IUNIT
1683      READ (IUNIT) ZVAL,AEQ
1684      READ (IUNIT) ZVAL,AER
1685      WRITE (59,*) 'READ (IUNIT) ZVAL,AEQ'
1686      WRITE (59,*) 'READ (IUNIT) ZVAL,AER'
1687      CALL SKIPR(IUNIT,-5,ISTAT)
1688      WRITE (59,*) 'SKIPPED BACK ',ISTAT(1),' RECORDS AND ',
1689      1 ISTAT(2),' FILES IN UNIT ',IUNIT,' '
1690      DO 260 I2=1,NY
1691      DO 260 I3=1,NT
1692          SRF(13,12)=( AEQ(13,12)*CONJG(AEQ(13,12))
1693          +AER(13,12)*CONJG(AER(13,12)) )/R2
1694 260 1 CONTINUE
1695      CALL CNTR(8*JSRF)
1696      ENDIF
1697 C
1698 C — END OF PLOTTING DATA SET
1699 C
1700 C
1701      IF (NT.GT.8.AND.NY.GT.8.AND.I0.GT.1) THEN

```

PRAM1 (version CD)

```

1702          CALL RELEASE(IRRE,'DN'L,DTFL2D)
1703          WRITE (59,*)'RELEASED DN= ',DTFL2D
1704          ENDIF
1705          KZOLD=KZNEW
1706          IZ=IZ+1
1707          KZNEW=KZ(IZ)
1708          290 CONTINUE
1709      C
1710      C — MOVE POINTER IN DATA FILE ON TO FIRST RECORD WITH NEXT ZVAL
1711      C
1712          IF (NY.GT.8) THEN
1713              CALL SKIPR(IUNIT,6,ISTAT)
1714              WRITE (59,*) 'SKIPPED ',ISTAT(1),' RECORDS AND ',ISTAT(2),
1715              1 ' FILES IN UNIT ',IUNIT
1716          ELSE
1717              CALL SKIPR(IUNIT,3,ISTAT)
1718              WRITE (59,*) 'SKIPPED ',ISTAT(1),' RECORDS AND ',ISTAT(2),
1719              1 ' FILES IN UNIT ',IUNIT
1720          ENDIF
1721          500 CONTINUE
1722      C
1723      C — CLOSE GRAPHICS SURFACES, END PROGRAM
1724      C
1725          501 CONTINUE
1726          CALL DONEPL
1727          CALL EXIT(1)
1728          END
1729      c
1730      c
1731      c
1732      c
1733          SUBROUTINE CNTR(KSRF)
1734      c
1735      C This subroutine was written by Godehard Hilfer and Curtis R. Menyuk
1736      C (2/87). It uses the commercial graphics package DISSPLA (SDSS) to
1737      C generate a contour plot of the data in array srf.
1738      c
1739      C This subroutine employs the subroutine nyaxis to compute 'nice' tick
1740      C marks along the coordinate axes and the subroutine xisFFT to compute
1741      C the location, extremes and intervals of the transformed variable axis
1742      C in FFT-plots. Depending on the value of ksrfl various titles,
1743      C coordinate axes and labels are selected and drawn. The sign of
1744      C ksrfl toggles the labeling option of the main contour lines
1745      C (positive ksrfl labels, negative ksrfl no labels). The main contour
1746      C lines are solid lines representing integral powers of 10. ndeC such
1747      C lines will be drawn below the surface maximum. lin (<9) other
1748      C contour lines (dashed lines) are drawn between the main contour
1749      C lines corresponding to the integral multiples of the next lower
1750      C integral power of ten. Which integral multiples are drawn is
1751      C determined by the first lin elements of the vector level. If
1752      C ishm = 1 a dotted contour will mark the half-height level. If
1753      C ishm = 0 this line will not be drawn, if ishm = -1 the half-height
1754      C contour and a dot at the surface maximum should be drawn.
1755      C
1756      C
1757      C
1758      C
1759      C
1760      C
1761      C
1762      C
1763      C
1764      C

```

—variables—

```

      grfsz = physical size of graphics plots
      i2 = y-coordinate index in do-loops 228,230
      i3 = t-coordinate index in do-loops 225,230
      lin = number of dashed contours between solid contours
      ishm = flag for half-height contour option in sub=cntr
      ksrfl = index number of surface that is being contoured
      labl = labelling variable

```

PRAM1 (version CD)

```

1765 C      level = vector with desired level heights for dashed contours
1766 C      lv = index do-loop 240
1767 C      ndec = desired number of solid contours representing powers of 10
1768 C      necveC = data switch for subroutine nyaxis
1769 C      nt = see RAM2D1
1770 C      ntp = nt+1
1771 C      ny = see RAM2D1
1772 C      nyp = ny+1
1773 C      rdy = step size in transverse spatial variable y
1774 C      srf = array of data from which contours are plotted
1775 C      tm1 = time coordinate lower limit
1776 C      tm2 = time coordinate upper limit
1777 C      tmax = value at end of time axis
1778 C      torig = value at beginning of time axis
1779 C      tstp = time axis interval
1780 C      wfmax = nice spatial FFT axis end value
1781 C      wforig = nice spatial FFT axis beginning value
1782 C      wfstp = nice spatial FFT axis interval
1783 C      xdum = dummy variable holding the x-coordinate of two points
1784 C      ydum = dummy variable holding the y-coordinate of two points
1785 C      yfmax = value at end of spatial FFT axis
1786 C      yforig = value at beginning of spatial FFT axis
1787 C      yfstp = spatial FFT axis interval
1788 C      ymax = value at end of transverse spatial axis
1789 C      yorig = value at beginning of transverse spatial axis
1790 C      ystp = transverse spatial axis interval
1791 C      ym1 = y-coordinate lower limit
1792 C      ym2 = y-coordinate upper limit
1793 C      zbot = logarithmic data cutoff
1794 C      zincr = special contour separation
1795 C      zlev = integral power of 10 next to data maximum
1796 C      zmax = data maximum
1797 C      zplane = reference level for contours
1798 C      zval = value of z-coordinate of current data/plot
1799 C
1800 C
1801 C      PARAMETER (NT=256,NTP=NT+1,NX=8,NY=128,NYP=NY+1,NYTP=NYP*NTP)
1802 C
1803 C      IMPLICIT COMPLEX(A-E,Q)
1804 C      DIMENSION ISRF(8),ITYPE(8),LEVEL(8),CSEC(19,NX),RTYPE(8),
1805 C      1 SRF(NTP,NYP),SRFI(NTP,NYP),SRFSEQ(NYTP),XDUM(2),YDUM(2)
1806 C      COMMON /GRAPHS/ ILN,ISHM,ISRF,ITYPE,LEVEL,NDEC,NHYP,NSEC,CSEC,
1807 C      1 GRFSZ,PI,RTYPE,SRF,SRFI,TMAX,TORIG,TSTP,YFMAX,YFORIG,
1808 C      2 YFSTP,YMAX,YORIG,YSTP,WFMAX,WFORIG,WFSTP,ZBOT,ZMAX,ZSTEP,
1809 C      3 ZVAL
1810 C      COMMON /NUM/ RDT,RDY,RDYF,TM1,TM2,YM1,YM2,YM2M1
1811 C      COMMON WORK(25000)
1812 C      EQUIVALENCE (SRF,SRFSEQ)
1813 C
1814 C - NO CONTOURING IN ONE DIMENSIONAL CASES
1815 C      IF (NT.LE.8.OR.NY.LE.8) RETURN
1816 C
1817 C - TOGGLE LABELLING DEPENDING ON SIGN OF KSRF
1818 C      LABL='LABELS'
1819 C      IF (KSRF.LT.0) LABL='NOLABELS'
1820 C      KSRF=ABS(KSRF)
1821 C
1822 C - SURFACE DATA
1823 C - MAKE DATA ARRAY SYMMETRIC TO OBTAIN AXIS LABELS ('NICE') AT AXIS END
1824 C      DO 200 I3=1,NT
1825 C      SRF(I3,NYP)=SRF(I3,1)
1826 C      200 CONTINUE
1827 C      DO 210 I2=1,NY

```


PRAM1 (version CD)

```

1828      SRF(NTP,12)=SRF(1,12)
1829      210 CONTINUE
1830      C
1831      C - FIND DATA MAXIMUM, ITS INTEGRAL POWER OF TEN AND CORRESPONDING
1832      C MANTISSA
1833      ZMAX=SRFSEQ(ISMAX(NYTP-1,SRFSEQ,1))
1834      IF(ZMAX.LE.0.0)THEN
1835          WRITE (59,*) 'warning: ZMAX IS ZERO OR NEGATIVE WHEN KSRF = ',
1836          1      KSRF
1837      RETURN
1838      ENDIF
1839      C
1840      C - DETERMINE LOWER DATA CUTOFF
1841      ZLEV=10.0**INT(ALOG10(ZMAX))
1842      ZBOT=(INT(ZMAX/ZLEV)+0.5)*ZLEV/10.0**NDEC
1843      IF(ZBOT.LE.0.0) WRITE (59,*) 'warning: ZBOT IS ZERO OR NEGATIVE ',
1844      1      ' WHEN KSRF = ',KSRF
1845      SRF(NTP,NYP)=ZBOT
1846      C
1847      C - START A NEW GRAPHICS FRAME FOR THIS CONTOUR PLOT
1848      CALL RESET('ALL')
1849      CALL INTXS
1850      CALL AREA2D(GRFSZ,GRFSZ)
1851      C
1852      C - HEADLINE, LABELS, AND COORDINATE SYSTEM
1853      GO TO (211,212,213,214,215,216,217,218) KSRF
1854      211 CALL HEADIN('TRANSIENT RAMAN: PUMP (PWR)$',100,1.5,1)
1855      GO TO 222
1856      212 CALL HEADIN('TRANSIENT RAMAN: PUMP (FFT, PWR)$',100,1.5,1)
1857      GO TO 222
1858      213 CALL HEADIN('TRANSIENT RAMAN: STOKES (PWR)$',100,1.5,1)
1859      GO TO 222
1860      214 CALL HEADIN('TRANSIENT RAMAN: STOKES (FFT, PWR)$',100,1.5,1)
1861      GO TO 222
1862      215 CALL HEADIN('TRANSIENT RAMAN: MATERIAL EXCITATION$',100,1.5,1)
1863      GO TO 222
1864      216 CALL HEADIN('TRANSIENT RAMAN: MATERIAL EXCITATION (FFT)$',
1865      1      100,1.5,1)
1866      GO TO 222
1867      217 CALL HEADIN('TRANSIENT RAMAN: PUMP AND STOKES (PWR)$',100,1.5,1)
1868      GO TO 222
1869      218 CALL HEADIN('TRANSIENT RAMAN: PUMP AND STOKES (FFT, PWR)$',
1870      1      100,1.5,1)
1871      222 CONTINUE
1872      CALL MSGAG('Z = $',100,5.9,7.1)
1873      IPLACE=2
1874      IF (ABS(ZVAL).GT.9999.0.OR.ABS(ZVAL).LT.0.01) IPLACE=-2
1875      CALL REALNO(ZVAL,IPLACE,6.4,7.1)
1876      CALL XNAME('TIME (PICO-SECONDS)$',100)
1877      IF(KSRF.EQ.2.OR.KSRF.EQ.4.OR.KSRF.EQ.6.OR.KSRF.EQ.8) THEN
1878          CALL YNONUM
1879          CALL YTICKS(0)
1880          VORIG=YFORIG
1881          VSTP=YFSTP
1882          VMAX=YFMAX
1883      ELSE
1884          CALL YNAME ('Y-DIMENSION (CM)$',100)
1885          VORIG=YORIG
1886          VSTP=YSTP
1887          VMAX=YMAX
1888      C
1889      C - AXIS LINE AND TICKMARKS ON THE RIGHT IN NO-FFT PLOTS
1890      NTIK=NINT((VMAX-VORIG)/VSTP)

```

PRAM1 (version CD)

```

1891      XDUM(1)=TMAX-(TMAX-TORIG)/50.0
1892      XDUM(2)=TMAX
1893      YDM=VORIG
1894      DO 225 ITK=1,NTIK-1
1895      YDM=YDM+VSTP
1896      YDUM(1)=YDM
1897      YDUM(2)=YDM
1898      CALL CURVE(XDUM,YDUM,2.0)
1899 225      CONTINUE
1900      ENDIF
1901      CALL GRAF(TORIG,TSTP,TMAX,VORIG,VSTP,VMAX)
1902  C
1903  C - COMPLETE COORDINATE FRAME AND TICKMARKS
1904      NTIK=NINT((TMAX-TORIG)/TSTP)
1905      YDUM(1)=VMAX
1906      YDUM(2)=VMAX-(VMAX-VORIG)/50.0
1907      XDM=TMAX
1908      DO 226 ITK=1,NTIK-1
1909      XDM=XDM-TSTP
1910      XDUM(1)=XDM
1911      XDUM(2)=XDM
1912      CALL CURVE(XDUM,YDUM,2.0)
1913 226      CONTINUE
1914      XDUM(1)=TORIG
1915      XDUM(2)=TMAX
1916      YDUM(1)=VMAX
1917      YDUM(2)=VMAX
1918      CALL CURVE(XDUM,YDUM,2.0)
1919      XDUM(1)=TMAX
1920      XDUM(2)=TMAX
1921      YDUM(1)=VMAX
1922      YDUM(2)=VORIG
1923      CALL CURVE(XDUM,YDUM,2.0)
1924      XDUM(1)=TORIG
1925      XDUM(2)=TORIG
1926      YDUM(1)=VMAX
1927      YDUM(2)=VORIG
1928      CALL CURVE(XDUM,YDUM,2.0)
1929  C
1930  C - PREPARE CONTOUR FINDING
1931      CALL BCOMON(25000)
1932      CALL CONANG(90.0)
1933      CALL PSPLIN
1934  C
1935  C - DRAW HALF-HEIGHT CONTOUR WITH LINE TYPE SPECIFIED IN
1936  C SUBROUTINE MYCON
1937      IF(ISHM.EQ.1.OR.ISHM.EQ.-1) THEN
1938          ZPLANE=0.1*ZMAX
1939          ZINCR=8.0*ZPLANE
1940          IF (ISHM.EQ.-1) THEN
1941              ZINCR=ZINCR*(1.0-1.0E-5)
1942          ELSE
1943              ZINCR=ZINCR*(1.0+1.0E-5)
1944          ENDIF
1945          CALL ZBASE(ZPLANE)
1946          CALL CONMAK(SRF,NTP,NYP,ZINCR)
1947          CALL CONLIN(0,'MYCON','NOLABELS',1,5)
1948          CALL CONTUR(1,'NOLABELS','DRAW')
1949          ZPLANE=0.5*ZMAX
1950          ZINCR=ZPLANE
1951          IF (ISHM.EQ.-1) THEN
1952              ZINCR=ZINCR*(1.0-1.0E-5)
1953          ELSE

```

PRAM1 (version CD)

```

1954         ZINCR=ZINCR*(1.0+1.0E-5)
1955     ENDIF
1956     CALL ZBASE(ZPLANE)
1957     CALL CONMAK(SRF,NTP,NYP,ZINCR)
1958     CALL CONLIN(0,'MYCON','NOLABELS',1.5)
1959     CALL CONTUR(1,'NOLABELS','DRAW')
1960 ENDIF
1961 C
1962 C - REPLACE ALL DATA SMALLER THAN THE CUTOFF ZBOT BY ZBOT
1963 C TO ELIMINATE UNDESIED LOGARITHMIC CONTOURS
1964     DO 230 I2=1,NYP
1965     DO 230 I3=1,NTP
1966     SRF(I3,I2)=MAX(ZBOT,SRF(I3,I2))
1967 C
1968 C - REPLACE ALL DATA BY THEIR DECADIC LOGARITHM FOR LOGARITHMIC INCREMENTS
1969 C BETWEEN CONTOURS
1970     SRF(I3,I2)=ALOG10(SRF(I3,I2))
1971 230 CONTINUE
1972 C
1973 C - COMPUTE AND DRAW A SOLID CONTOUR LINE EVERY INTEGER STARTING AT ZERO
1974     ZPLANE=0.0
1975     CALL ZBASE(ZPLANE)
1976     CALL CONLIN(0,'SOLID',LABL,2.8)
1977     CALL CONMAK(SRF,NTP,NYP,1.0)
1978     CALL CONTUR(1,LABL,'DRAW')
1979 C
1980 C - COMPUTE AND DRAW A DASHED CONTOUR LINE EVERY INTEGER STARTING AT EVERY
1981 C LOGARITHM OF THE ELEMENTS OF THE VECTOR LEVEL
1982     CALL CONLIN(0,'DASH','NOLABELS',1.3)
1983     DO 240 LV=1,ILN
1984     ZPLANE=ALOG10(FLOAT(LEVEL(LV)))
1985     CALL ZBASE(ZPLANE)
1986     CALL CONMAK(SRF,NTP,NYP,1.0)
1987     CALL CONTUR(1,'NOLABELS','DRAW')
1988 240 CONTINUE
1989 C
1990 C - SPECIAL AXIS AND LABEL FOR FFT COORDINATE
1991     IF ((KSRF.EQ.2.OR.KSRF.EQ.4.OR.KSRF.EQ.6.OR.KSRF.EQ.8)
1992     1 .AND.NY.GT.8) CALL XISFFT('Y',TORIG,TMAX)
1993     CALL ENDPL(0)
1994     RETURN
1995     END
1996 c
1997 c
1998 c
1999 c
2000 SUBROUTINE MYCON(DUMMY,IDUMMY)
2001 C
2002 C This subroutine makes a customized dotted contourline as described
2003 C in the DISSPLA manual.
2004 c
2005     DIMENSION RATRAY(2)
2006     TLENG=0.14
2007     NMRKSP=2
2008     RATRAY(1)=1.0/6.0
2009     RATRAY(2)=5.0/6.0
2010     CALL MRSCOD(TLENG,NMRKSP,RATRAY)
2011     RETURN
2012     END
2013 c
2014 c
2015 c
2016 c

```

PRAM1 (version CD)

```

2017      SUBROUTINE XISFFT(OORD,YMNDMY,YMXDMY)
2018      C
2019      C This subroutine calculates the values for the call x/y-graxs which
2020      C labels the axis of the FFT-variable.
2021      C
2022      C      grfsz = physical size of graphics plots
2023      C      tmax = value at end of time axis
2024      C      torig = value at beginning of time axis
2025      C      tstp = time axis interval
2026      C      wfmaz = nice spatial FFT axis end value
2027      C      wforig = nice spatial FFT axis beginning value
2028      C      wfstp = nice spatial FFT axis interval
2029      C      yfmaz = value at end of spatial FFT axis
2030      C      yforig = value at beginning of spatial FFT axis
2031      C      yfstp = spatial FFT axis interval
2032      C      ymax = value at end of transverse spatial axis
2033      C      yorig = value at beginning of transverse spatial axis
2034      C      ystp = transverse spatial axis interval
2035      C      uaxor = x- or y-distance of secondary axis from physical origin
2036      C      udiff = difference of original axis end values
2037      C      ulnth = length of customized axis in inches
2038      C      vaxor = x- or y-distance of secondary axis from physical origin
2039      C      xdum = dummy variable holding the x-coordinate of two points
2040      C      ydum = dummy variable holding the y-coordinate of two points
2041      C
2042      C      PARAMETER (NT=256,NTP=NT+1,NX=8,NY=128,NYP=NY+1)
2043      C
2044      C      IMPLICIT COMPLEX(A-E,O)
2045      C      DIMENSION ISRF(8),ITYPE(8),LEVEL(8),CSEC(19,NX),RTYPE(8),
2046      C      1      SRF(NTP,NYP),SRFI(NTP,NYP),XDUM(2),YDUM(2)
2047      C      COMMON /GRAPHS/ ILN,ISHM,ISRF,ITYPE,LEVEL,NDEC,NHYP,NSEC,CSEC,
2048      C      1      GRFSZ,PI,RTYPE,SRF,SRFI,TMAX,TORIG,TSTP,YFMAY,YFORIG,
2049      C      2      YFSTP,YMAX,YORIG,YSTP,WFMAY,WFORIG,WFSTP,ZBOT,ZMAX,ZSTEP,
2050      C      3      ZVAL
2051      C
2052      C - NO FFT-AXIS IN ONE-DIMENSIONAL TRANSIENT CASE
2053      C      IF (NY.LE.8) THEN
2054      C          WRITE (59,*) 'NO FFT-AXIS WHEN NY.LE.8'
2055      C          RETURN
2056      C      ENDIF
2057      C
2058      C - WARNING NOT ONE INTERVAL FITS BETWEEN EXTREMA
2059      C      IF (WFORIG+WFSTP.GE.WFMAY) WRITE (59,*) 'AXIS ON FFT PLOTS WRONG'
2060      C
2061      C - COMPUTE AXIS LENGTH AND ORIGIN
2062      C      UDIFF=YFMAY-YFORIG
2063      C      ULNTH=GRFSZ*(WFMAY-WFORIG)/UDIFF
2064      C      UAXOR=GRFSZ*(WFORIG-YFORIG)/UDIFF
2065      C      VAXOR=0.0
2066      C      ORD='Y'
2067      C      IF (OORD.EQ.ORD) THEN
2068      C
2069      C - Y-AXIS (IN CONTOUR PLOTS)
2070      C
2071      C - YAXIS TICKMARKS ON THE RIGHT
2072      C      NTIK=NINT((WFMAY-WFORIG)/WFSTP)
2073      C      XDUM(1)=TMAX-(TMAX-TORIG)/50.0
2074      C      XDUM(2)=TMAX
2075      C      YDM=WFORIG-WFSTP
2076      C      DO 250 ITK=1,NTIK+1
2077      C          YDM=YDM+WFSTP
2078      C          YDUM(1)=YDM
2079      C          YDUM(2)=YDM

```

PRAM1 (version CD)

```

2080      CALL CURVE(XDUM,YDUM,2,0)
2081 250      CONTINUE
2082          CALL RESET('YNONUM')
2083          CALL RESET('YTICKS')
2084          CALL YGRAXS(WFORIG,WFSTP,WFMAX,ULNTH,
2085      1      'INVERSE WAVE LENGTH (1/CM)$',100,VAXOR,UAXOR)
2086      ELSE
2087          WRITE (59,*)'A FFT'
2088  C
2089  C - X-AXIS (IN CROSS SECTIONAL PLOTS)
2090          CALL RESET('XNONUM')
2091          CALL RESET('XTICKS')
2092  C
2093  C - DRAW LINE FOR AXIS
2094          XDUM(1)=YFORIG
2095          XDUM(2)=YFMAX
2096          YDUM(1)=YMNDMY
2097          YDUM(2)=YMNDMY
2098          CALL CURVE(XDUM,YDUM,2,0)
2099          WRITE (59,*)'B FFT'
2100  C
2101  C - YAXIS TICKMARKS ON THE RIGHT
2102          NTIK=NINT((WFMAX-WFORIG)/WFSTP)
2103          WRITE (59,*)'C FFT'
2104          YDUM(1)=YMXDMY-(YMXDMY-YMNDMY)/50.0
2105          YDUM(2)=YMXDMY
2106          XDM=WFORIG-WFSTP
2107          DO 255 ITK=1,NTIK+1
2108              XDM=XDM+WFSTP
2109              XDUM(1)=XDM
2110              XDUM(2)=XDM
2111              CALL CURVE(XDUM,YDUM,2,0)
2112 255      CONTINUE
2113          WRITE (59,*)'D FFT'
2114  C
2115  C - DRAW CUSTOMIZED TICK MARKS
2116          CALL XGRAXS(WFORIG,WFSTP,WFMAX,ULNTH,
2117      1      'INVERSE WAVE LENGTH (1/CM)$',100,UAXOR,VAXOR)
2118      ENDIF
2119      RETURN
2120      END
2121  C
2122  C
2123  C
2124  C
2125      SUBROUTINE CRSSCT(MSRF)
2126  C
2127  C This subroutine was written by Godehard Hilfer (3/87). It generates
2128  C cross sectional plots of the data in the two dimensional array(s)
2129  C srf (srfl).
2130  C
2131  C Three types of cross sectional plots are available: intensity plots
2132  C (following statement label 300), phase plots, and amplitude plots
2133  C (both following statement label 400). When intensity cross sections
2134  C are called for, this subroutine executes do-loop 390 that does all
2135  C cross sections specified in row msrf of array cseC and thereafter
2136  C returns control to the main program. When phase or amplitude cross
2137  C sections are called for, this subroutine executes do-loop 490 which
2138  C generates all phase sections specified in row msrf of array csec.
2139  C immediately afterwards do-loop 590 is executed which generates all
2140  C amplitude cross sections that are specified in row msrf+1 of array
2141  C csec. After this control is returned to the main program.
2142  C

```

PRAM1 (version CD)

```

2143 C Each type of cross sections is prepared in a similar fashion. In
2144 C the case of one dimensional data (ny or nt less than 9) only one
2145 C argument of the arrays srf and srfl is an independent variable
2146 C the other argument serves as a label to allow distinction between
2147 C up to eight one dimensional data sets. Which one of these eight
2148 C data sets is to be graphed is determined by the value of the real
2149 C part of the element of cseC under consideration (the imaginary
2150 C part is meaningless in these cases). When nt and ny are larger than
2151 C 8 srf and srfl contain one two-dimensional data set, a surface.
2152 C Which of the two free variables is to be held constant for
2153 C cross sectional plots is determined by the imaginary part of the
2154 C element of cseC under consideration. Therefore, in 2-d cases
2155 C the imaginary part of the current element of cseC is tested. If it
2156 C is 2.0 a horizontal cross section (second variable of array(s) srf
2157 C (srfl) fixed) follows, if it is 1.0 a vertical cross section (first
2158 C variable of array(s) srf (srfl) fixed) follows, otherwise the next
2159 C element of cseC will be considered in the same way. For the present
2160 C graph the headline and axis labels are written onto a new graphics
2161 C frame, the curve data are computed, the coordinate system is drawn
2162 C and finally the cross sectional curve itself. If the plot displays
2163 C FFT data the drawing of the FFT-axis that would be drawn by the call
2164 C graf will be suppressed in order to avoid the tick mark labels at the
2165 C very end of this axis which would exhibit messy numbers. In the
2166 C place of the suppressed axis a 'secondary' (DISPLA nomenclature)
2167 C axis will be drawn immediately after the cross sectional curve
2168 C is drawn. This secondary axis exhibits 'nicely' valued tick marks.
2169 C
2170 C The cross sectional curves represent the functional values at the
2171 C grid point iseC that is closest to the locations specified by the
2172 C real part of the current element of cseC. While the data of the
2173 C intensity and amplitude plots are readily available from the
2174 C array(s) srf (srfl) the data for the phase sections have to be
2175 C calculated first by this subroutine.
2176 C
2177 C The phase data are calculated as follows. The field magnitude at the
2178 C fixed grid point iseC is computed. If its maximum is less than
2179 C 10**(-30) the field information is determined unreliable and no
2180 C phase curve will be drawn. Furthermore all locations where the
2181 C magnitude is less than the maximum magnitude divided by 10**8 are
2182 C determined as points of unreliable field information and will
2183 C exhibit no phase curve point. The arctangent of the ratio of the
2184 C imaginary to real field amplitudes provides the raw phase data. It
2185 C is assumed that the numerical resolution of RAM2D1 is sufficient to
2186 C provide raw phase data that do not vary by more than +/- pi from
2187 C grid point to grid point. The first raw data point falls within
2188 C +/- pi of zero phase. All consecutive raw data points are tested if
2189 C they were reached by a phase change that implies a crossing of the
2190 C negative real axis of the amplitude vector in which case 2 pi will
2191 C be added or subtracted to all following phase points depending on an
2192 C implied phase windup or wind-down. By this method phase variations
2193 C over multiples of 2 pi can be followed. In case of intermittent
2194 C unreliable data points the next reliable phase is placed within the
2195 C same 2 pi interval as the previous reliable phase.
2196 C
2197 C
2198 C
2199 C
2200 C
2201 C
2202 C
2203 C
2204 C
2205 C

```

—variables—

```

cseC = 2-dim array with cross sectional information
grfz = physical size of graphics plots
iseC = srf(i) grid point corresponds closest to real part of
      cseC
k = index in do-loops 390,490,590
k1 = index in do-loops 328,375,410,460,510,520,555,560
k2 = index in do-loops 330,380,411,465,530,565

```

PRAM1 (version CD)

```

2206 C      k3 = index in do-loops 423,473
2207 C      k4 = index in do-loops 429,475
2208 C      kcnt = index when calculating and plotting phase data
2209 C      lpi = multiples of 2 pi counter
2210 C      mserf = index number of surface of which cross sections are drawn
2211 C      nab = index of the first of a string of reliable phase data
2212 C           points
2213 C      nan = index of the last of a string of reliable phase data points
2214 C      necveC = data switch for subroutine nyaxis
2215 C      npoints = number of data points to be drawn
2216 C      nseC = number of elements tested in rows of cseC
2217 C      nserf = index number of amplitude surface of which cross sections
2218 C           are drawn
2219 C      nt = see RAM2D1
2220 C      ntp = nt+1
2221 C      ny = see RAM2D1
2222 C      nyhp = ny/2+1
2223 C      nyp = ny+1
2224 C      phasdf = test variable deciding phase axis interval
2225 C      phasmx = phase axis end value
2226 C      phasor = phase axis beginning value
2227 C      phastp = phase axis interval
2228 C      phamxi = integer closest to phasmx
2229 C      phsori = integer closest to phsori
2230 C      pi = 3.14159265358979
2231 C      psik = phase being tested for 2 pi interval
2232 C      psip = previous phase referencing in 2 pi interval test
2233 C      rdt = step size in time
2234 C      rdy = step size in transverse spatial variable y
2235 C      rdyx = grid point spacing on horizontal axis
2236 C      scmem = array containing initial longitudinal invariant data
2237 C      scold = last reliable phase before unreliable phase data
2238 C      segi = imaginary part of current cseC element
2239 C      secr = real part of current cseC element
2240 C      secti = vector containing phase data or imaginary amplitude data
2241 C      sectn = vector containing intensity data, magnitude data, or real
2242 C           amplitude data
2243 C      srf = source data array from main program
2244 C      srfi = source data array from main program (imaginary part)
2245 C      tm1 = time coordinate lower limit
2246 C      tm2 = time coordinate upper limit
2247 C      tmax = value at end of time axis
2248 C      torig = value at beginning of time axis
2249 C      tstp = time axis interval
2250 C      wfmax = nice spatial FFT axis end value
2251 C      wforig = nice spatial FFT axis beginning value
2252 C      wfstp = nice spatial FFT axis interval
2253 C      wmax = nice vertical axis end value
2254 C      worig = nice vertical axis beginning value
2255 C      wstp = nice vertical axis interval
2256 C      yfmax = value at end of spatial FFT axis
2257 C      yforig = value at beginning of spatial FFT axis
2258 C      yfstp = spatial FFT axis interval
2259 C      ymax = value at end of transverse spatial axis
2260 C      yorig = value at beginning of transverse spatial axis
2261 C      ystp = transverse spatial axis interval
2262 C      xi = imaginary amplitude value
2263 C      xmx = section maximum
2264 C      xr = real amplitude value
2265 C      xthrsh = fraction of intensity below which data are considered
2266 C           unreliable
2267 C      xx = vector containing physical x-axis values for plotting
2268 C      xdum = dummy variable holding the x-coordinate of two points

```

PRAM1 (version CD)

```

2269 C      ydum = dummy variable holding the y-coordinate of two points
2270 C      ym1 = y-coordinate lower limit
2271 C      ym2 = y-coordinate upper limit
2272 C      ym2m1 = ym2-ym1
2273 C      zval = current z-location
2274 C
2275 C
2276 C      PARAMETER (NP=10,NT=256,NTP=NT+1,NX=8,NY=128,NYH=NY/2,NYHP=NYH+1,
2277 C      1      NYP=NY+1,NTPY=NT+NY)
2278 C
2279 C      IMPLICIT COMPLEX(A-E,Q)
2280 C      DIMENSION ISRF(8),ITYPE(8),IWHEN(NYHP),LEVEL(8),CSEC(19,NX),
2281 C      1      PSMEM(NTPY,8),PPMEM(NTPY,8),RTYPE(8),SCMEM(NTPY,8),
2282 C      2      SECTI(NTPY),SECTJ(NTPY),SECTN(NTPY),SRF(NTP,NYP),
2283 C      3      SRFI(NTP,NYP),TIK(NY),XDUM(2),XX(NTPY),YDUM(2)
2284 C      COMMON /GRAPHS/ ILN,ISHM,ISRF,ITYPE,LEVEL,NDEC,NHYP,NSEC,CSEC,
2285 C      1      GRFSZ,PI,RTYPE,SRF,SRFI,TMAX,TORIG,TSTP,YFMAX,YFORIG,
2286 C      2      YFSTP,YMAX,YORIG,YSTP,WFMAX,WFORIG,WFSTP,ZBOT,ZMAX,ZSTEP,
2287 C      3      ZVAL
2288 C      COMMON /NUM/ RDT,RDY,RDYF,TM1,TM2,YM1,YM2,YM2M1
2289 C      COMMON WORK(25000)
2290 C
2291 C - ERROR CONDITIONS
2292 C      IF (MSRF.LT.1.OR.MSRF.GT.19) THEN
2293 C          WRITE (59,*) 'MSRF = ',MSRF,' IN CRSSCT OUT OF RANGE'
2294 C          RETURN
2295 C      ENDIF
2296 C
2297 C      IF (NY.LE.8) THEN
2298 C          GO TO (290,290,290, 280,280,280, 290,290,290,
2299 C      1      280,280,280, 290,290,290, 280,280,280, 290) MSRF
2300 C      280      RETURN
2301 C      290      CONTINUE
2302 C      ENDIF
2303 C      GO TO (300,400,400, 300,400,400, 300,400,400,
2304 C      1      300,400,400, 300,400,400, 300,400,400, 300) MSRF
2305 C      300      CONTINUE
2306 C
2307 C - CROSS SECTIONS OF INTENSITY SURFACES OR MATERIAL EXCITATION
2308 C
2309 C - CHECK EACH ELEMENT IN ROW MSRF OF ARRAY CSEC
2310 C      DO 390 K=1,NSEC
2311 C          SECR=REAL(CSEC(MSRF,K))
2312 C
2313 C - ONE-DIMENSIONAL CASES
2314 C      IF (NY.LE.8) THEN
2315 C          IF (SECR.LT.0.5.OR.SECR.GT.8.5) GO TO 390
2316 C          GO TO 310
2317 C      ELSE IF (NT.LE.8) THEN
2318 C          IF (SECR.LT.0.5.OR.SECR.GT.8.5) GO TO 390
2319 C          GO TO 340
2320 C      ENDIF
2321 C
2322 C - TWO DIMENSIONAL CASES; SECTION ONLY IF IMAGINARY PART OF CSEC-
2323 C      ELEMENT IS EQUAL TO 1.0 OR 2.0;
2324 C      OTHERWISE GO TO NEXT LOOP COUNTER K, I.E. NEXT ELEMENT OF LINE MSRF
2325 C      IN ARRAY CSEC
2326 C      SECI=AIMAG(CSEC(MSRF,K))
2327 C      IF (SECI.GT.0.9.AND.SECI.LT.1.1) GO TO 340
2328 C      IF (SECI.GT.1.9.AND.SECI.LT.2.1) GO TO 310
2329 C      GO TO 390
2330 C      310      CONTINUE
2331 C

```


PRAM1 (version CD)

```

2332 C -- HORIZONTAL CROSS SECTION (SECOND ARGUMENT OF SRF FIXED); INTENSITY
2333 C
2334 C - START A NEW GRAPHICS FRAME FOR THIS CROSS SECTION
2335     CALL RESET('ALL')
2336     CALL AREA2D(GRFSZ,GRFSZ)
2337     CALL INTAXS
2338 C
2339 C - HEADLINE, LABELS, AND PARAMETER
2340     GO TO (311,390,390, 312,390,390, 313,390,390,
2341 1      314,390,390, 315,390,390, 316,390,390, 317) MSRF
2342 311 CALL HEADIN('RAMAN PUMP: INTENSITY$',100,1.5,1)
2343     GO TO 321
2344 312 CALL HEADIN('RAMAN PUMP: MODE INTENSITY$',100,1.5,1)
2345     GO TO 321
2346 313 CALL HEADIN('RAMAN STOKES: INTENSITY$',100,1.5,1)
2347     GO TO 321
2348 314 CALL HEADIN('RAMAN STOKES: MODE INTENSITY$',100,1.5,1)
2349     GO TO 321
2350 315 CALL HEADIN('RAMAN MAT. EXC.: INTENSITY$',100,1.5,1)
2351     GO TO 321
2352 316 CALL HEADIN('RAMAN MAT. EXC.: MODE INTENSITY$',100,1.5,1)
2353     GO TO 321
2354 317 CALL HEADIN('RAMAN AMPLIFIER Z-INVARIANT$',100,1.5,1)
2355 321 CONTINUE
2356     CALL MESSAG('Z = $',100,5.9,7.1)
2357     IPLACE=2
2358     IF (ABS(ZVAL).GT.9999.0.OR.ABS(ZVAL).LT.0.01) IPLACE=-2
2359     CALL REALNO(ZVAL,IPLACE,6.4,7.1)
2360     CALL XNAME('TIME (PICO-SECONDS)$',100)
2361     IF (MSRF.EQ.4.OR.MSRF.EQ.10.OR.MSRF.EQ.16) THEN
2362         CALL YNAME('INTENSITY IN MODE KY$',100)
2363         CALL MESSAG('KY = $',100,4.2,7.1)
2364         IPLACE=2
2365         IF (ABS(SECR).GT.9999.0.OR.ABS(SECR).LT.0.01) IPLACE=-2
2366         CALL REALNO(SECR,IPLACE,4.7,7.1)
2367         CALL MESSAG('2 - DIM.$',100,5.9,7.35)
2368         ISEC=INT((SECR+NYHP/YM2M1)*YM2M1)
2369     ELSE
2370         IF (MSRF.EQ.19.) THEN
2371             CALL YNAME('LONGITUDINAL INVARIANT$',100)
2372             IF (ZVAL.GT.ZSTEP) THEN
2373                 CALL MESSAG('DASHED = INVARIANT AT Z=0$',100,0.1,7.35)
2374             ENDIF
2375         ELSE
2376             CALL YNAME('INTENSITY$',100)
2377         ENDIF
2378         IF (NY.LE.8) THEN
2379             ISEC=NINT(SECR)
2380             GO TO (322,323,324,325) ITYPE(ISEC)
2381 322 CALL MESSAG('SEC-HYPERB. , EXP = $',100,0.1,7.1)
2382             GO TO 326
2383 323 CALL MESSAG('RECTANGULAR$',100,0.1,7.1)
2384             GO TO 326
2385 324 CALL MESSAG('LORENTZIAN , EXP = $',100,0.1,7.1)
2386             GO TO 326
2387 325 CALL MESSAG('EXPONENTIAL , EXP = $',100,0.1,7.1)
2388 326 CONTINUE
2389             IF (ITYPE(ISEC).NE.2) THEN
2390                 XRTYPE=RTYPE(ISEC)
2391                 IPLACE=2
2392                 IF (ABS(XRTYPE).GT.9999.0.OR.ABS(XRTYPE).LT.0.01)
2393 1                 IPLACE=-2
2394                 CALL REALNO(XRTYPE,IPLACE,2.4,7.1)

```

PRAM1 (version CD)

```

2395         ENDIF
2396         IF (NY.GT.1) THEN
2397             CALL MESSAG('CASE$',100,4.0,7.1)
2398             CALL INTNO(ISEC,4.7,7.1)
2399         ENDIF
2400         CALL MESSAG('1 - D TRA.$',100,5.9,7.35)
2401     ELSE
2402         CALL MESSAG('Y = $',100,4.2,7.1)
2403         IPLACE=2
2404         IF (ABS(SECR).GT.9999.0.OR.ABS(SECR).LT.0.01) IPLACE=-2
2405         CALL REALNO(SECR,IPLACE,4.7,7.1)
2406         CALL MESSAG('2 - DIM.$',100,5.9,7.35)
2407         ISEC=INT((SECR+RDY*NYHP)/RDY)
2408     ENDIF
2409 ENDIF
2410 C
2411 C - CROSS SECTION DATA
2412 DO 327 K1=1,NT
2413     SECTN(K1)=SRF(K1,ISEC)
2414     XX(K1)=TM1+RDT*(K1-1)
2415 327 CONTINUE
2416 C
2417 C - TOTAL INTENSITY INTEGRAL IN 1-D
2418 IF (NY.LE.8) THEN
2419     TOTI=0.0
2420     DO 329 K1=1,NT
2421         TOTI=TOTI+SECTN(K1)
2422 329 CONTINUE
2423     TOTI=TOTI*RDT
2424     IF (ZVAL.LT.ZSTEP) THEN
2425 C
2426 C - INTEGRAL VALUE ONTO GRAPH WHEN Z=0
2427         IF (MSRF.EQ.1.AND.TOTI.GT.0.0) THEN
2428             TTPSTO=TOTI
2429             CALL MESSAG('INTEGRAL= $',100,0.1,6.7)
2430             IPLACE=4
2431             IF (TOTI.GT.9999.0.OR.TOTI.LT.0.01) IPLACE=-2
2432             CALL REALNO(TOTI,IPLACE,1.4,6.7)
2433         ENDIF
2434         IF (MSRF.EQ.7.AND.TOTI.GT.0.0) THEN
2435             TTSSTO=TOTI
2436             CALL MESSAG('INTEGRAL= $',100,0.1,6.7)
2437             IPLACE=4
2438             IF (TOTI.GT.9999.0.OR.TOTI.LT.0.01) IPLACE=-2
2439             CALL REALNO(TOTI,IPLACE,1.4,6.7)
2440         ENDIF
2441         IF (MSRF.EQ.7) TTSSTO=TOTI
2442     ELSE
2443 C
2444 C - DEPLETION/GAIN VALUE ONTO GRAPH WHEN Z>0
2445         IF (MSRF.EQ.1.AND.TTPSTO.GT.0.0) THEN
2446             RINTEG=TOTI/TTPSTO
2447             CALL MESSAG('DEPLETION= $',100,0.1,6.7)
2448             IPLACE=4
2449             IF (RINTEG.GT.9999.0.OR.RINTEG.LT.0.01) IPLACE=-2
2450             CALL REALNO(RINTEG,IPLACE,1.4,6.7)
2451         ENDIF
2452         IF (MSRF.EQ.7.AND.TTSSTO.GT.0.0) THEN
2453             RINTEG=TOTI/TTSSTO
2454             CALL MESSAG('GAIN= $',100,0.1,6.7)
2455             IPLACE=4
2456             IF (RINTEG.GT.9999.0.OR.RINTEG.LT.0.01) IPLACE=-2
2457             CALL REALNO(RINTEG,IPLACE,0.7,6.7)

```

PRAM1 (version CD)

```

2458             ENDIF
2459         ENDIF
2460     ENDIF
2461 C
2462 C - MEMORIZE INVARIANT AT Z=0
2463     IF (MSRF.EQ.19.AND.ZVAL.LT.ZSTEP) THEN
2464         DO 335 K1=1,NT
2465             SCMEM(K1,K)=SECTN(K1)
2466         335     CONTINUE
2467     ENDIF
2468 C
2469 C - DRAW COORDINATE SYSTEM
2470     NECLEC=1
2471     WSTP=-1.0
2472     WORIG=0.0
2473     CALL NYSXIS(SECTN,NT,NECLEC,WORIG,WSTP,WMAX)
2474     CALL GRAF(TORIG,TSTP,TMAX,WORIG,WSTP,WMAX)
2475 C
2476 C - COMPLETE COORDINATE FRAME AND TICKMARKS
2477     XDUM(1)=TORIG
2478     XDUM(2)=TMAX
2479     YDUM(1)=WMAX
2480     YDUM(2)=WMAX
2481     CALL CURVE(XDUM,YDUM,2,0)
2482     NTIK=NINT((TMAX-TORIG)/TSTP)
2483     YDUM(1)=WMAX
2484     YDUM(2)=WMAX-(WMAX-WORIG)/50.0
2485     XDM=TORIG
2486     DO 336 ITK=1,NTIK-1
2487         XDM=XDM+TSTP
2488         XDUM(1)=XDM
2489         XDUM(2)=XDM
2490         CALL CURVE(XDUM,YDUM,2,0)
2491     336     CONTINUE
2492     XDUM(1)=TMAX
2493     XDUM(2)=TMAX
2494     YDUM(1)=WMAX
2495     YDUM(2)=WORIG
2496     CALL CURVE(XDUM,YDUM,2,0)
2497     NTIK=NINT((WMAX-WORIG)/WSTP)
2498     XDUM(1)=TMAX-(TMAX-TORIG)/50.0
2499     XDUM(2)=TMAX
2500     YDM=WORIG
2501     DO 337 ITK=1,NTIK-1
2502         YDM=YDM+WSTP
2503         YDUM(1)=YDM
2504         YDUM(2)=YDM
2505         CALL CURVE(XDUM,YDUM,2,0)
2506     337     CONTINUE
2507 C
2508 C - DRAW CROSS SECTION CURVE
2509     NPOINTS=NT
2510     CALL CURVE(XX,SECTN,NPOINTS,0)
2511 C
2512 C - DRAW Z=0 INVARIANT FOR COMPARISON
2513     IF (MSRF.EQ.19.AND.ZVAL.GT.ZSTEP) THEN
2514         DO 338 K1=1,NT
2515             SECTN(K1)=SCMEM(K1,K)
2516         338     CONTINUE
2517         CALL DASH
2518         CALL CURVE(XX,SECTN,NPOINTS,0)
2519         CALL RESET('DASH')
2520     ENDIF

```

PRAM1 (version CD)

```

2521 C
2522 C - END OF PLOT
2523 CALL ENDPL(0)
2524 GO TO 390
2525 340 CONTINUE
2526 C
2527 C - VERTICAL CROSS SECTION ( FIRST VARIABLE FIXED IN SRF); INTENSITY
2528 C
2529 C - START A NEW GRAPHICS FRAME FOR THIS CROSS SECTION
2530 CALL RESET('ALL')
2531 CALL AREA2D(GRFSZ,GRFSZ)
2532 CALL INTAXS
2533 C
2534 C - HEADLINE, LABELS, AND PARAMETER
2535 GO TO (355,390,390, 356,390,390, 357,390,390,
2536 1 358,390,390, 359,390,390, 360,390,390, 361) MSRF
2537 355 CALL HEADIN('RAMAN PUMP: INTENSITY$',100,1.5,1)
2538 GO TO 371
2539 356 CALL HEADIN('RAMAN PUMP: INTENSITY (FFT)$',100,1.5,1)
2540 GO TO 371
2541 357 CALL HEADIN('RAMAN STOKES: INTENSITY$',100,1.5,1)
2542 GO TO 371
2543 358 CALL HEADIN('RAMAN STOKES: INTENSITY (FFT)$',100,1.5,1)
2544 GO TO 371
2545 359 CALL HEADIN('RAMAN MAT. EXC.: INTENSITY$',100,1.5,1)
2546 GO TO 371
2547 360 CALL HEADIN('RAMAN MAT. EXC.: INTENSITY (FFT)$',100,1.5,1)
2548 GO TO 371
2549 361 CALL HEADIN('RAMAN LONGITUDINAL INVARIANT$',100,1.5,1)
2550 371 CONTINUE
2551 CALL MSGAG('Z = $',100,5.9,7.1)
2552 IPLACE=2
2553 IF (ABS(ZVAL).GT.9999.0.OR.ABS(ZVAL).LT.0.01) IPLACE=-2
2554 CALL REALNO(ZVAL,IPLACE,6.4,7.1)
2555 IF (MSRF.EQ.4.OR.MSRF.EQ.10.OR.MSRF.EQ.10) THEN
2556 CALL XNONUM
2557 CALL XTICKS(0)
2558 CALL YNAME('FFT INTENSITY$',100)
2559 RDYX=RDYF
2560 XORIG=YFORIG
2561 XSTP=YFSTP
2562 XMAX=YFMAX
2563 ELSE
2564 CALL XNAME('Y-DIMENSION (CM)$',100)
2565 IF (MSRF.EQ.10.) THEN
2566 CALL YNAME('LONGITUDINAL INVARIANT$',100)
2567 IF (ZVAL.GT.ZSTEP) THEN
2568 CALL MSGAG('DASHED = INVARIANT AT Z=0$',100,0.1,7.35)
2569 ENDIF
2570 ELSE
2571 CALL YNAME('INTENSITY$',100)
2572 ENDIF
2573 RDYX=RDY
2574 XORIG=YORIG
2575 XSTP=YSTP
2576 XMAX=YMAX
2577 ENDIF
2578 IF (NT.LE.8) THEN
2579 ISEC=NINT(SECR)
2580 CALL MSGAG('EXPON.. NHYP = $',100,0.1,7.1)
2581 CALL INTNO(NHYP,2.0,7.1)
2582 CALL MSGAG('1 - D STA.$',100,5.9,7.35)
2583 IF (NT.GT.1) THEN

```

PRAM1 (version CD)

```

2584         CALL MESSAG('CASE$',100,+.0,7.1)
2585         CALL INTNO(ISEC,4.7,7.1)
2586     ENDIF
2587 ELSE
2588     ISEC=INT((SECR+RDT*(NT/2+1))/RDT)
2589     CALL MESSAG('T = $',100,4.2,7.1)
2590     IPLACE=2
2591     IF (ABS(SECR).GT.9999.0.OR.ABS(SECR).LT.3.01) IPLACE=-2
2592     CALL REALNO(SECR,IPLACE,4.7,7.1)
2593     CALL MESSAG('2 - DIM.$',100,5.9,7.35)
2594 ENDIF
2595 C
2596 C - CROSS SECTION DATA
2597 DO 373 K1=1,NY
2598     SECTN(K1)=SRF(ISEC,K1)
2599     XX(K1)=XORIG+RDYX*(K1-1)
2600 373 CONTINUE
2601 C
2602 C - TOTAL INTENSITY INTEGRAL IN 1-D
2603 IF (NT.LE.8) THEN
2604     TOTI=0.0
2605     DO 374 K1=1,NY
2606         TOTI=TOTI+SECTN(K1)
2607 374 CONTINUE
2608     TOTI=TOTI*RDY
2609     IF (ZVAL.LT.ZSTEP) THEN
2610 C
2611 C - INTEGRAL VALUE ONTO GRAPH WHEN Z=0
2612 IF (MSRF.EQ.1.AND.TOTI.GT.0.0) THEN
2613     TTPSTO=TOTI
2614     CALL MESSAG('INTEGRAL= $',100,0.1,6.7)
2615     IPLACE=4
2616     IF (TOTI.GT.9999.0.OR.TOTI.LT.0.01) IPLACE=-2
2617     CALL REALNO(TOTI,IPLACE,1.4,6.7)
2618 ENDIF
2619 IF (MSRF.EQ.7.AND.TOTI.GT.0.0) THEN
2620     TTSSTO=TOTI
2621     CALL MESSAG('INTEGRAL= $',100,0.1,6.7)
2622     IPLACE=4
2623     IF (TOTI.GT.9999.0.OR.TOTI.LT.0.01) IPLACE=-2
2624     CALL REALNO(TOTI,IPLACE,1.4,6.7)
2625 ENDIF
2626 IF (MSRF.EQ.7) TTSSTO=TOTI
2627 ELSE
2628 C
2629 C - DEPLETION/GAIN VALUE ONTO GRAPH WHEN Z>0
2630 IF (MSRF.EQ.1.AND.TTPSTO.GT.0.0) THEN
2631     RINTEG=TOTI/TTPSTO
2632     CALL MESSAG('DEPLETION= $',100,0.1,6.7)
2633     IPLACE=4
2634     IF (RINTEG.GT.9999.0.OR.RINTEG.LT.0.01) IPLACE=-2
2635     CALL REALNO(RINTEG,IPLACE,1.4,6.7)
2636 ENDIF
2637 IF (MSRF.EQ.7.AND.TTSSTO.GT.0.0) THEN
2638     RINTEG=TOTI/TTSSTO
2639     CALL MESSAG('GAIN= $',100,0.1,6.7)
2640     IPLACE=4
2641     IF (RINTEG.GT.9999.0.OR.RINTEG.LT.0.01) IPLACE=-2
2642     CALL REALNO(RINTEG,IPLACE,0.7,6.7)
2643 ENDIF
2644 ENDIF
2645 ENDIF
2646 C

```

PRAM1 (version CD)

```

2647 C - MEMORIZE INVARIANT AT Z=0
2648 IF (MSRF.EQ.19.AND.ZVAL.LT.ZSTEP) THEN
2649 DO 375 K1=1,NY
2650 SCMEM(K1,K)=SECTN(K1)
2651 375 CONTINUE
2652 ENDIF
2653 IF (MSRF.EQ.10) THEN
2654 C
2655 C - COMPUTE TOTAL INTENSITY OF STOKES BEAM IN K-SPACE
2656 TIK(1)=0.0
2657 DO 376 J=2,NY
2658 TIK(J)=TIK(J-1)+SECTN(J)*RDYF
2659 376 CONTINUE
2660 TIKBMX=TIK(NY)
2661 WRITE (59,*) 'TIKBMX= ',TIKBMX
2662 CALL MESSAG('TOT. INT. = $',100,0.2,6.7)
2663 IPLACE=2
2664 IF (ABS(TIKBMX).GT.9999.0.OR.ABS(TIKBMX).LT.0.01) IPLACE=-2
2665 CALL REALNO(TIKBMX,IPLACE,1.6,6.7)
2666 WRITE (59,*) 'TOTAL INTENSITY OF STOKES IN K= ',TIKBMX
2667 C
2668 C - COMPUTE K-WIDTH OF STOKES (LINEAR INTERPOLATION)
2669 DO 377 J=1,NY
2670 TIK(J)=ABS((2.0*TIK(J)-TIKBMX)/TIKBMX)-2.0/PI
2671 377 CONTINUE
2672 CALL WHENFLE(NY,TIK(1),1.0,0,IWHEN,NVAL)
2673 IWN1=IWHEN(1)+JL-1
2674 IWN2=IWHEN(NVAL)+JL-1
2675 IF (IWN2-IWN1.LT.4) THEN
2676 WRITE (59,*) 'INSUFFICIENT RESOLUTION FFT BEAM ',IB
2677 ENDIF
2678 WRITE (59,*) 'IWHEN = ',IWHEN
2679 YKL=YFORIG+RDYF*(IWN1+TIK(IWN1))/(TIK(IWN1-1)-TIK(IWN1)))
2680 YKR=YFORIG+RDYF*(IWN2-TIK(IWN2))/(TIK(IWN2+1)-TIK(IWN2)))
2681 WDTKKB=YKR-YKL
2682 CALL MESSAG(' K WIDTH = $',100,4.2,6.7)
2683 IPLACE=2
2684 IF (ABS(WDTKKB).GT.9999.0.OR.ABS(WDTKKB).LT.0.01) IPLACE=-2
2685 CALL REALNO(WDTKKB,IPLACE,5.6,6.7)
2686 ENDIF
2687 C
2688 C - DRAW COORDINATE SYSTEM
2689 NECLEC=1
2690 WSTP=-1.0
2691 WORIG=0.0
2692 CALL NYSXIS(SECTN,NY,NECLEC,WORIG,WSTP,WMAX)
2693 CALL GRAF(XORIG,XSTP,XMAX,WORIG,WSTP,WMAX)
2694 C
2695 C - COMPLETE COORDINATE FRAME AND TICKMARKS
2696 XDUM(1)=XORIG
2697 XDUM(2)=XMAX
2698 YDUM(1)=WMAX
2699 YDUM(2)=WMAX
2700 CALL CURVE(XDUM,YDUM,2,0)
2701 IF (NY.GT.8.AND.(MSRF.EQ.4.OR.MSRF.EQ.10.OR.MSRF.EQ.16)) GOTO 380
2702 NTIK=NINT((XMAX-XORIG)/XSTP)
2703 YDUM(1)=WMAX
2704 YDUM(2)=WMAX-(WMAX-WORIG)/50.0
2705 XDM=XORIG
2706 DO 378 ITK=1,NTIK-1
2707 XDM=XDM+XSTP
2708 XDUM(1)=XDM
2709 XDUM(2)=XDM

```

PRAM1 (version CD)

```

2710 CALL CURVE(XDUM,YDUM,2,0)
2711 378 CONTINUE
2712 380 CONTINUE
2713 XDUM(1)=XMAX
2714 XDUM(2)=XMAX
2715 YDUM(1)=WMAX
2716 YDUM(2)=WORIG
2717 CALL CURVE(XDUM,YDUM,2,0)
2718 NTIK=NINT((WMAX-WORIG)/WSTP)
2719 XDUM(1)=XMAX-(XMAX-XORIG)/50.0
2720 XDUM(2)=XMAX
2721 YDM=WORIG
2722 DO 382 ITK=1,NTIK-1
2723 YDM=YDM+WSTP
2724 YDUM(1)=YDM
2725 YDUM(2)=YDM
2726 CALL CURVE(XDUM,YDUM,2,0)
2727 382 CONTINUE
2728 C
2729 C - DRAW CROSS SECTION CURVE
2730 NPOINTS=NY
2731 CALL CURVE(X,SECTN,NPOINTS,0)
2732 C
2733 C - DRAW Z=0 INVARIANT FOR COMPARISON
2734 IF (MSRF.EQ.19.AND.ZVAL.GT.ZSTEP) THEN
2735 DO 385 K1=1,NY
2736 SECTN(K1)=SCMEM(K1,K)
2737 385 CONTINUE
2738 CALL DASH
2739 CALL CURVE(X,SECTN,NPOINTS,0)
2740 CALL RESET('DASH')
2741 ENDIF
2742 C
2743 C - SPECIAL AXIS AND LABEL FOR FFT COORDINATE
2744 IF (NY.GT.8.AND.(MSRF.EQ.4.OR.MSRF.EQ.10.OR.MSRF.EQ.16))
2745 1 CALL XISFFT('X',WORIG,WMAX)
2746 WRITE (59,*)'END OF STATIONARY INTENSITY PLOT'
2747 C
2748 C - END OF PLOT
2749 CALL ENDPL(0)
2750 390 CONTINUE
2751 RETURN
2752 C
2753 C — END OF INTENSITY SECTION
2754 C
2755 400 CONTINUE
2756 C
2757 C — PHASE AND AMPLITUDE SECTIONS
2758 C
2759 C — CHECK EACH ELEMENT IN ROW MSRF OF ARRAY CSEC
2760 DO 490 K=1,NSEC
2761 SECR=REAL(CSEC(MSRF,K))
2762 C
2763 C - ONE DIMENSIONAL CASES
2764 IF (NY.LE.8) THEN
2765 IF (SECR.LT.0.5.OR.SECR.GT.8.5) GO TO 490
2766 GO TO 401
2767 ELSE IF (NT.LE.8) THEN
2768 IF (SECR.LT.0.5.OR.SECR.GT.8.5) GO TO 490
2769 GO TO 450
2770 ENDIF
2771 C
2772 C - TWO DIMENSIONAL CASES; SECTION ONLY IF IMAGINARY PART OF CSEC-

```

PRAM1 (version CD)

```

2773 C ELEMENT IS EQUAL TO 1.0 OR 2.0;
2774 C OTHERWISE GO TO NEXT LOOP COUNTER K, I.E. NEXT ELEMENT OF LINE MSRF
2775 C IN ARRAY CSEC
2776 SEC1=AIMAG(CSEC(MSRF,K))
2777 IF (SEC1.GT.0.9.AND.SEC1.LT.1.1) GO TO 450
2778 IF (SEC1.GT.1.9.AND.SEC1.LT.2.1) GO TO 401
2779 GO TO 490
2780 401 CONTINUE
2781 C
2782 C — HORIZONTAL CROSS SECTION (SECOND ARGUMENT OF SRF FIXED); PHASE
2783 C
2784 C - START A NEW GRAPHICS FRAME FOR THIS CROSS SECTION
2785 CALL RESET('ALL')
2786 CALL INTAXS
2787 CALL MX1ALF('STANDARD','1')
2788 CALL MX2ALF('L/CGRK','1')
2789 CALL AREA2D(GRFSZ,GRFSZ)
2790 IF (ZVAL.GE.ZSTEP.AND.(MSRF.EQ.2.OR.MSRF.EQ.8)) THEN
2791 CALL MESSAG('SOLID = INTERFER.$',100,0.1,7.35)
2792 CALL MESSAG('DASHED = ACTUAL$',100,2.3,7.35)
2793 ENDIF
2794 C
2795 C - HEADLINE, LABELS, AND PARAMETER
2796 GO TO (490,402,490, 490,403,490, 490,404,490,
2797 1 490,405,490, 490,406,490, 490,407,490) MSRF
2798 402 CALL HEADIN('RAMAN PUMP: PHASE$',100,1.5,1)
2799 GO TO 409
2800 403 CALL HEADIN('RAMAN PUMP: MODE PHASE$',100,1.5,1)
2801 GO TO 409
2802 404 CALL HEADIN('RAMAN STOKES: PHASE$',100,1.5,1)
2803 GO TO 409
2804 405 CALL HEADIN('RAMAN STOKES: MODE PHASE$',100,1.5,1)
2805 GO TO 409
2806 406 CALL HEADIN('RAMAN MAT. EXC.: PHASE$',100,1.5,1)
2807 GO TO 409
2808 407 CALL HEADIN('RAMAN MAT. EXC.: MODE PHASE$',100,1.5,1)
2809 409 CONTINUE
2810 CALL MESSAG('Z = $',100,5.9,7.1)
2811 IPLACE=2
2812 IF (ABS(ZVAL).GT.9999.0.OR.ABS(ZVAL).LT.0.01) IPLACE=-2
2813 CALL REALNO(ZVAL,IPLACE,6.4,7.1)
2814 CALL XNAME('TIME (PICO-SECONDS)$',100)
2815 IF (MSRF.EQ.5.OR.MSRF.EQ.11.OR.MSRF.EQ.17) THEN
2816 CALL YNAME('MODE PHASE (MULTIPLES OF #PI)$',100)
2817 CALL MESSAG('KY = $',100,4.2,7.1)
2818 IPLACE=2
2819 IF (ABS(SECR).GT.9999.0.OR.ABS(SECR).LT.0.01) IPLACE=-2
2820 CALL REALNO(SECR,IPLACE,4.7,7.1)
2821 CALL MESSAG('2 - DIM.$',100,5.9,7.35)
2822 ISEC=INT((SECR+NYHP/YM2M1)*YM2M1)
2823 ELSE
2824 CALL YNAME('PHASE (MULTIPLES OF #PI)$',100)
2825 IF (NY.LE.8) THEN
2826 CALL MESSAG('1 - D TRA.$',100,5.9,7.35)
2827 ISEC=NINT(SECR)
2828 GO TO (412,413,414,415) ITYPE(ISEC)
2829 412 CALL MESSAG('SEC-HYPERB. , EXP = $',100,0.1,7.1)
2830 GO TO 416
2831 413 CALL MESSAG('RECTANGULAR$',100,0.1,7.1)
2832 GO TO 416
2833 414 CALL MESSAG('LORENTZIAN , EXP = $',100,0.1,7.1)
2834 GO TO 416
2835 415 CALL MESSAG('EXPONENTIAL , EXP = $',100,0.1,7.1)

```


PRAM1 (version CD)

```

2836      416      CONTINUE
2837      IF (ITYPE(ISEC).NE.2) THEN
2838          XRTYPE=RTYPE(ISEC)
2839          IPLACE=2
2840          IF (ABS(XRTYPE).GT.9999.0.OR.ABS(XRTYPE).LT.0.01)
2841      1          IPLACE=-2
2842          CALL REALNO(XRTYPE,IPLACE,2.4,7.1)
2843      ENDIF
2844      IF (NY.GT.1) THEN
2845          CALL MESSAG('CASE$',100,4.0,7.1)
2846          CALL INTNO(ISEC,4.7,7.1)
2847      ENDIF
2848      ELSE
2849          CALL MESSAG('Y = $',100,4.2,7.1)
2850          IPLACE=2
2851          IF (ABS(SECR).GT.9999.0.OR.ABS(SECR).LT.0.01) IPLACE=-2
2852          CALL REALNO(SECR,IPLACE,4.7,7.1)
2853          CALL MESSAG('2 - DIM.$',100,5.9,7.35)
2854          ISEC=INT((SECR+RDY*NYHP)/RDY)
2855      ENDIF
2856      ENDIF
2857      C
2858      C - MAGNITUDE DATA, ABSCISSA VECTOR
2859      DO 418 K1=1,NT
2860          XR=SRF(K1,ISEC)
2861          XI=SRFI(K1,ISEC)
2862          SECTN(K1)=SQRT(XR*XR+XI*XI)
2863          XX(K1)=TM1+RDT*(K1-1)
2864      418      CONTINUE
2865      C
2866      C - UNCERTAIN PHASE THRESHOLD
2867          XMX=SECTN(ISMAX(NT,SECTN,1))
2868          IF (XMX.LT.1.0E-30) THEN
2869              WRITE (59,*) 'note: UNRELIABLE PHASE, MAGNITUDES ARE ZERO'
2870              GO TO 490
2871          ENDIF
2872          XTHRSX=XMX/1.0E8
2873      C
2874      C - CALCULATE PHASE DATA
2875      C
2876      C - PHASE OF FIRST DATA POINT WITHIN +/- PI OF ZERO PHASE
2877          SCTOLD=0.0
2878      C
2879      C - INITIALIZE LOOP VARIABLES
2880          NAB=1
2881          NAN=0
2882          KINCR=0
2883      C
2884      C - LOOP OVER ALL GRID POINTS; KINCR LOOP COUNTER
2885      420      CONTINUE
2886          KINCR=KINCR+1
2887      C
2888      C - CLEAR VECTOR FOR PHASE DATA
2889          SECTI(KINCR)=0.0
2890      C
2891      C - FIND STRING OF GRID POINTS (NAB TO NAN) WHERE MAGNITUDE OF FIELD
2892      C DATA EXCEEDS THRESHOLD
2893          IF (SECTN(KINCR).GE.XTHRSX) THEN
2894              NAN=KINCR
2895          C
2896      C - PLACE MARKER (SECTN=-1.0) WHERE FIELD MAGNITUDE IS BELOW THRESHOLD
2897      C (UNCERTAIN PHASE INFORMATION)
2898          ELSE

```

PRAM1 (version CD)

```

2899          SECTN(KINCR)=1.0
2900          IF (NAN.GE.NAB) THEN
2901 C
2902 C - EXIT LOOP TEMPORARILY TO CALCULATE PHASE DATA FOR STRING OF GRID
2903 C   POINTS NAB TO NAN
2904         GO TO 421
2905       ELSE
2906 C
2907 C - CURRENT DATA POINT STILL UNCERTAIN; INCREMENT NAB (BEGINNING OF NEXT
2908 C   STRING)
2909         NAB=KINCR+1
2910       ENDIF
2911     ENDIF
2912 C
2913 C -- END LOOP OR CONTINUE IN LOOP UNTIL NT
2914       IF (KINCR.LT.NT) GO TO 420
2915 C
2916 C - SKIP PHASE CALCULATION, LAST DATA POINTS ARE UNCERTAIN
2917       IF (NAN.LT.NAB) GO TO 432
2918 C
2919 C - CALCULATE RAW PHASE MODULO 2*PI
2920     421 CONTINUE
2921         DO 423 K3=NAB,NAN
2922           SECTI(K3)=ATAN2(SRFI(K3,ISEC),SRF(K3,ISEC))/PI
2923     423 CONTINUE
2924 C
2925 C - CALCULATE EXACT PHASE KEEPING TRACK OF MULTIPLES OF 2*PI COUNTED BY
2926 C   LPI
2927         LPI=0
2928         IF (NAN.EQ.NAB) THEN
2929 C
2930 C - SINGLE DATA POINT
2931         SECTI(NAN)=SECTI(NAN)+SCTOLD
2932         GO TO 431
2933       ENDIF
2934 C
2935 C - PHASE OF FIRST DATA POINT IN STRING
2936         PSIP=SECTI(NAB)
2937         SECTI(NAB)=PSIP+SCTOLD
2938 C
2939 C - PHASE OF FOLLOWING DATA POINTS IN STRING
2940         DO 429 K4=NAB+1,NAN
2941           PSIK=SECTI(K4)
2942           IF (PSIP.GE.0.0) THEN
2943 C
2944 C - INCREMENT LPI IF PRESENT RAW PHASE PSIK DIFFERS BY MORE THAN PI FROM
2945 C   THE PREVIOUS POINT PSIP (WHICH WAS POSITIVE)
2946           IF (ABS(Psik-PSIP).GT.1.0) LPI=LPI+2
2947         ELSE
2948 C
2949 C - DECREMENT LPI IF PRESENT RAW PHASE PSIK DIFFERS BY MORE THAN PI FROM
2950 C   THE PREVIOUS POINT PSIP (WHICH WAS NEGATIVE)
2951           IF (ABS(Psik-PSIP).GT.1.0) LPI=LPI-2
2952         ENDIF
2953 C
2954 C - EXACT PHASE
2955         SECTI(K4)=PSIK+LPI+SCTOLD
2956 C
2957 C - CURRENT RAW PHASE BECOMES PREVIOUS RAW PHASE NEXT TIME THROUGH THE
2958 C   LOOP
2959         PSIP=PSIK
2960     429 CONTINUE
2961     431 CONTINUE

```

PRAM1 (version CD)

```

2962 C
2963 C - STORE PHASE OF LAST DATA POINT AS REFERENCE VALUE FOR NEXT STRING OF
2964 C RELIABLE DATA
2965 C SCTOLD=SECTI(NAN)
2966 C
2967 C - INCREMENT LABEL OF BEGINNING OF NEXT STRING
2968 C NAB=KINCR+1
2969 C
2970 C - FIND NEXT STRING OF PHASE DATA
2971 C IF (NAN.LT.NT) GO TO 420
2972 432 CONTINUE
2973 C
2974 C - PLOT COORDINATE SYSTEM
2975 C IF ((MSRF.EQ.2.OR.MSRF.EQ.8).AND.ZVAL.LT.ZSTEP) THEN
2976 C
2977 C - MEMORIZE ORIGINAL PHASE
2978 C DO 433 I3=1,NT
2979 C IF (MSRF.EQ.2) THEN
2980 C PPMEM(I3,K)=SECTI(I3)
2981 C ELSE
2982 C PSMEM(I3,K)=SECTI(I3)
2983 C ENDIF
2984 433 CONTINUE
2985 C ELSE
2986 C
2987 C - INTERFEROMETRIC PHASE (CURRENT PHASE MINUS ORIGINAL PHASE)
2988 C DO 434 I3=1,NT
2989 C IF (MSRF.EQ.2) THEN
2990 C SECTJ(I3)=SECTI(I3)-PPMEM(I3,K)
2991 C ELSE
2992 C SECTJ(I3)=SECTI(I3)-PSMEM(I3,K)
2993 C ENDIF
2994 434 CONTINUE
2995 C ENDIF
2996 C
2997 C - SCALE AXIS BY COMBINATION OF BOTH CURVES
2998 C NECLEC=1
2999 C PHASTP=0.0
3000 C CALL NYSXIS(SECTJ,NT,NECLEC,PHASOR,PHASTP,PHASMX)
3001 C NECLEC=0
3002 C CALL NYSXIS(SECTI,NT,NECLEC,PHASOR,PHASTP,PHASMX)
3003 C
3004 C - MAKE FRACTIONAL Y-AXIS LIMITS INTEGRAL
3005 C IF (PHASTP.LE.1.0) THEN
3006 C PHSORI=ANINT(PHASOR)
3007 C PHSMXI=ANINT(PHASMIX)
3008 C IF (PHASOR.LE.PHSORI) THEN
3009 C PHASOR=PHSORI-1.0
3010 C ELSE
3011 C PHASOR=PHSORI
3012 C ENDIF
3013 C IF (PHASMX.GE.PHSMXI) THEN
3014 C PHASMX=PHSMXI+1.0
3015 C ELSE
3016 C PHASMX=PHSMXI
3017 C ENDIF
3018 C
3019 C - SMALLEST Y-INTERVAL SIZES SHOULD BE PI/4 AND PI/2
3020 C PHASDF=PHASMX-PHASOR
3021 C IF (PHASDF.LE.2.0) THEN
3022 C PHASTP=0.25
3023 C ELSE IF (PHASDF.LE.4.0) THEN
3024 C PHASTP=0.5

```

PRAM1 (version CD)

```

3025         ENDIF
3026     ENDIF
3027     CALL GRAF(TORIG,TSTP,TMAX,PHASOR,PHASTP,PHASMX)
3028 C
3029 C - COMPLETE COORDINATE FRAME AND TICKMARKS
3030     XDUM(1)=TORIG
3031     XDUM(2)=TMAX
3032     YDUM(1)=PHASMX
3033     YDUM(2)=PHASMX
3034     CALL CURVE(XDUM,YDUM,2,0)
3035     NTIK=NINT((TMAX-TORIG)/TSTP)
3036     YDUM(1)=PHASMX
3037     YDUM(2)=PHASMX-(PHASMX-PHASOR)/50.0
3038     XDM=TORIG
3039     DO 435 ITK=1,NTIK-1
3040     XDM=XDM+TSTP
3041     XDUM(1)=XDM
3042     XDUM(2)=XDM
3043     CALL CURVE(XDUM,YDUM,2,0)
3044 435 CONTINUE
3045     XDUM(1)=TMAX
3046     XDUM(2)=TMAX
3047     YDUM(1)=PHASMX
3048     YDUM(2)=PHASOR
3049     CALL CURVE(XDUM,YDUM,2,0)
3050     NTIK=NINT((PHASMX-PHASOR)/PHASTP)
3051     XDUM(1)=TMAX-(TMAX-TORIG)/50.0
3052     XDUM(2)=TMAX
3053     YDM=PHASOR
3054     DO 436 ITK=1,NTIK-1
3055     YDM=YDM+PHASTP
3056     YDUM(1)=YDM
3057     YDUM(2)=YDM
3058     CALL CURVE(XDUM,YDUM,2,0)
3059 436 CONTINUE
3060 C
3061 C - PLOT PHASE CURVE SEGMENTS; RESET COUNTERS
3062     NPOINTS=0
3063     KINCR=0
3064 C
3065 C — LOOP OVER DATA POINTS; LOOP COUNTER KINCR
3066 437 CONTINUE
3067     KINCR=KINCR+1
3068     IF (SECTN(KINCR).LT.-0.99) THEN
3069 C
3070 C - UNRELIABLE PHASE MARKER ENCOUNTERED; PLOT DATA STRING OF LENGTH
3071 C     NPOINTS
3072     IF (NPOINTS.GT.0) GO TO 438
3073     ELSE
3074 C
3075 C - INCREMENT DATA STRING COUNTER; PUSH RELIABLE DATA TO FRONT OF VECTOR
3076 C     FOR PLOTTING
3077     NPOINTS=NPOINTS+1
3078     SECTI(NPOINTS)=SECTI(KINCR)
3079     IF (ZVAL.GE.ZSTEP.AND.(MSRF.EQ.2.OR.MSRF.EQ.8)) THEN
3080     SECTJ(NPOINTS)=SECTJ(KINCR)
3081     ENDIF
3082     XX(NPOINTS)=XX(KINCR)
3083     ENDIF
3084 C
3085 C — END LOOP OR CONTINUE IN LOOP UNTIL NT
3086     IF (KINCR.LT.NT) GO TO 437
3087 C

```

PRAM1 (version CD)

```

3088 C - LAST DATA POINTS UNRELIABLE; END PHASE PLOTTING
3089 IF (NPOINTS.EQ.0) GO TO 440
3090 438 CONTINUE
3091 C
3092 C - PLOT DATA STRING; RESET DATA COUNTER
3093 C - DRAW INTERFEROMETRIC PHASE SOLID
3094 IF (ZVAL.GE.ZSTEP.AND.(MSRF.EQ.2.OR.MSRF.EQ.8)) THEN
3095 CALL CURVE(XX,SECTJ,NPOINTS,0)
3096 ENDIF
3097 C
3098 C - DRAW CURRENT PHASE DASHED
3099 CALL DASH
3100 CALL CURVE(XX,SECTI,NPOINTS,0)
3101 CALL RESET('DASH')
3102 NPOINTS=0
3103 C
3104 C - JUMP BACK INTO LOOP FOR NEXT STRING OF PHASE DATA
3105 IF (KINCR.LT.NT) GO TO 437
3106 440 CONTINUE
3107 CALL ENDPL(0)
3108 GO TO 490
3109 450 CONTINUE
3110 C
3111 C — VERTICAL CROSS SECTION ( FIRST VARIABLE FIXED IN SRF); PHASE
3112 C
3113 C - START A NEW GRAPHICS FRAME FOR THIS CROSS SECTION
3114 CALL RESET('ALL')
3115 CALL INTAXS
3116 CALL MX1ALF('STANDARD',1)
3117 CALL MX2ALF('L/CGRK',1)
3118 CALL AREA2D(GRFSZ,GRFSZ)
3119 IF (ZVAL.GE.ZSTEP.AND.(MSRF.EQ.2.OR.MSRF.EQ.8)) THEN
3120 CALL MESSAG('SOLID = INTERFER.$',100,0.1,7.35)
3121 CALL MESSAG('DASHED = ACTUAL$',100,2.4,7.35)
3122 ENDIF
3123 C
3124 C - HEADLINE, LABELS, AND PARAMETER
3125 GO TO (490,452,490, 490,453,490, 490,454,490,
3126 1 490,455,490, 490,456,490, 490,457,490) MSRF
3127 452 CALL HEADIN('RAMAN PUMP: PHASE$',100,1.5,1)
3128 GO TO 459
3129 453 CALL HEADIN('RAMAN PUMP: PHASE (FFT)$',100,1.5,1)
3130 GO TO 459
3131 454 CALL HEADIN('RAMAN STOKES: PHASE$',100,1.5,1)
3132 GO TO 459
3133 455 CALL HEADIN('RAMAN STOKES: PHASE (FFT)$',100,1.5,1)
3134 GO TO 459
3135 456 CALL HEADIN('RAMAN MAT. EXC.: PHASE$',100,1.5,1)
3136 GO TO 459
3137 457 CALL HEADIN('RAMAN MAT. EXC.: PHASE (FFT)$',100,1.5,1)
3138 459 CONTINUE
3139 CALL MESSAG('Z = $',100,5.9,7.1)
3140 IPLACE=2
3141 IF (ABS(ZVAL).GT.9999.0.OR.ABS(ZVAL).LT.0.01) IPLACE=-2
3142 CALL REALNO(ZVAL,IPLACE,6.4,7.1)
3143 IF (MSRF.EQ.5.OR.MSRF.EQ.11.OR.MSRF.EQ.17) THEN
3144 CALL XNONUM
3145 CALL XTICKS(0)
3146 CALL YNAME('FFT PHASE (MULTIPLES OF #PI)$',100)
3147 RDYX=RDYF
3148 XORIG=YFORIG
3149 XSTP=YFSTP
3150 XMAX=YFMAX

```

PRAM1 (version CD)

```

3151     ELSE
3152         CALL XNAME('Y-DIMENSION (CM)$',100)
3153         CALL YNAME('PHASE (MULTIPLES OF #PI)$',100)
3154         RDYX=RDY
3155         XORIG=YORIG
3156         XSTP=YSTP
3157         XMAX=YMAX
3158     ENDIF
3159     IF (NT.LE.8) THEN
3160         CALL MESSAG('EXPON., NHYP = $',100,0.1,7.1)
3161         CALL INTNO(NHYP,2,0,7.1)
3162         CALL MESSAG('1 - D STA.$',100,5.9,7.35)
3163         ISEC=NINT(SECR)
3164         IF (NT.GT.1) THEN
3165             CALL MESSAG('CASE$ ',100,4.0,7.1)
3166             CALL INTNO(ISEC,4,7,7.1)
3167         ENDIF
3168     ELSE
3169         CALL MESSAG('2 - DIM.$',100,5.9,7.35)
3170         CALL MESSAG('T = $',100,4.2,7.1)
3171         IPLACE=2
3172         IF (ABS(SECR).GT.9999.0.OR.ABS(SECR).LT.0.01) IPLACE=-2
3173         CALL REALNO(SECR,IPLACE,4,7,7.1)
3174         ISEC=INT((SECR+RDT*(NT/2+1))/RDT)
3175     ENDIF
3176 C
3177 C - MAGNITUDE DATA, ABSCISSA VECTOR
3178     DO 460 K1=1,NY
3179         XR=SRF(ISEC,K1)
3180         XI=SRFI(ISEC,K1)
3181         SECTN(K1)=SQRT(XR*XR+XI*XI)
3182         XX(K1)=XORIG+RDYX*(K1-1)
3183     460 CONTINUE
3184 C
3185 C - UNCERTAIN PHASE THRESHOLD
3186     XMX=SECTN(ISMAX(NY,SECTN,1))
3187     IF (XMX.LT.1.0E-30) THEN
3188         WRITE (59,*) 'note: UNRELIABLE PHASE, MAGNITUDES ARE ZERO'
3189         GO TO 490
3190     ENDIF
3191     XTHRSX=XMX/1.0E8
3192 C
3193 C - CALCULATE PHASE DATA
3194 C
3195 C - PHASE OF FIRST DATA POINT WITHIN +/- PI OF ZERO PHASE
3196     SCTOLD=0.0
3197 C
3198 C - INITIALIZE LOOP VARIABLES
3199     NAB=1
3200     NAN=0
3201     KINCR=0
3202 C
3203 C - LOOP OVER ALL GRID POINTS; KINCR LOOP COUNTER
3204     470 CONTINUE
3205     KINCR=KINCR+1
3206 C
3207 C - CLEAR VECTOR FOR PHASE DATA
3208     SECTI(KINCR)=0.0
3209 C
3210 C - FIND STRING OF GRID POINTS (NAB TO NAN) WHERE MAGNITUDE OF FIELD
3211 C   DATA EXCEEDS THRESHOLD
3212     IF (SECTN(KINCR).GE.XTHRSX) THEN
3213         NAN=KINCR

```

PRAM1 (version CD)

```

3214 C
3215 C - PLACE MARKER (SECTN=-1.0) WHERE FIELD MAGNITUDE IS BELOW THRESHOLD
3216 C (UNCERTAIN PHASE INFORMATION)
3217 ELSE
3218     SECTN(KINCR)=-1.0
3219     IF (NAN.GE.NAB) THEN
3220 C
3221 C - EXIT LOOP TEMPORARILY TO CALCULATE PHASE DATA FOR STRING OF GRID
3222 C POINTS NAB TO NAN
3223     GO TO 471
3224 ELSE
3225 C
3226 C - CURRENT DATA POINT STILL UNCERTAIN; INCREMENT NAB (BEGINNING OF NEXT
3227 C STRING)
3228     NAB=KINCR+1
3229     ENDIF
3230 ENDIF
3231 C
3232 C - END LOOP OR CONTINUE IN LOOP UNTIL NT
3233     IF (KINCR.LT.NY) GO TO 470
3234 C
3235 C - SKIP PHASE CALCULATION, LAST DATA POINTS ARE UNCERTAIN
3236     IF (NAN.LT.NAB) GO TO 479
3237 471 CONTINUE
3238 C
3239 C - CALCULATE RAW PHASE MODULO 2*PI
3240     DO 473 K3=NAB,NAN
3241     SECTI(K3)=ATAN2(SRF1(ISEC,K3),SRF(ISEC,K3))/PI
3242 473 CONTINUE
3243 C
3244 C - CALCULATE EXACT PHASE KEEPING TRACK OF MULTIPLES OF 2*PI COUNTED BY
3245 C LPI
3246     LPI=0
3247     IF (NAN.EQ.NAB) THEN
3248 C
3249 C - SINGLE DATA POINT
3250     SECTI(NAN)=SECTI(NAN)+SCTOLD
3251     GO TO 477
3252     ENDIF
3253 C
3254 C - PHASE OF FIRST DATA POINT IN STRING
3255     PSIP=SECTI(NAB)
3256     SECTI(NAB)=PSIP+SCTOLD
3257 C
3258 C - PHASE OF FOLLOWING DATA POINTS IN STRING
3259     DO 475 K4=NAB+1,NAN
3260     PSIK=SECTI(K4)
3261     IF (PSIP.GE.0.0) THEN
3262 C
3263 C - INCREMENT LPI IF PRESENT RAW PHASE PSIK DIFFERS BY MORE THAN PI FROM
3264 C THE PREVIOUS POINT PSIP (WHICH WAS POSITIVE)
3265     IF (ABS(PSIK-PSIP).GT.1.0) LPI=LPI+2
3266     ELSE
3267 C
3268 C - DECREMENT LPI IF PRESENT RAW PHASE PSIK DIFFERS BY MORE THAN PI FROM
3269 C THE PREVIOUS POINT PSIP (WHICH WAS POSITIVE)
3270     IF (ABS(PSIK-PSIP).GT.1.0) LPI=LPI-2
3271     ENDIF
3272 C
3273 C - EXACT PHASE
3274     SECTI(K4)=PSIK+LPI+SCTOLD
3275 C
3276 C - CURRENT RAW PHASE BECOMES PREVIOUS RAW PHASE NEXT TIME THROUGH THE

```

PRAM1 (version CD)

```

3277 C LOOP
3278 PSIP=PSIK
3279 475 CONTINUE
3280 477 CONTINUE
3281 C
3282 C - STORE PHASE OF LAST DATA POINT AS REFERENCE VALUE FOR NEXT STRING OF
3283 C RELIABLE DATA
3284 SCTOLD=SECTI(NAN)
3285 C
3286 C - INCREMENT LABEL OF BEGINNING OF NEXT STRING
3287 NAB=KINCR+1
3288 C
3289 C - FIND NEXT STRING OF PHASE DATA
3290 IF (NAN.LT.NY) GO TO 470
3291 479 CONTINUE
3292 C
3293 C - MEMORIZE ORIGINAL PHASE
3294 IF (ZVAL.LT.ZSTEP) THEN
3295 IF (MSRF.EQ.2) THEN
3296 DO 480 I3=1,NY
3297 PPMEM(I3,K)=SECTI(I3)
3298 480 CONTINUE
3299 ELSE
3300 DO 481 I3=1,NY
3301 PSMEM(I3,K)=SECTI(I3)
3302 481 CONTINUE
3303 ENDIF
3304 ELSE
3305 C
3306 C - INTERFEROMETRIC PHASE (CURRENT PHASE MINUS ORIGINAL PHASE)
3307 IF (MSRF.EQ.2) THEN
3308 DO 482 I3=1,NY
3309 SECTJ(I3)=SECTI(I3)-PPMEM(I3,K)
3310 482 CONTINUE
3311 ENDIF
3312 IF (MSRF.EQ.8) THEN
3313 DO 483 I3=1,NY
3314 SECTJ(I3)=SECTI(I3)-PSMEM(I3,K)
3315 483 CONTINUE
3316 ENDIF
3317 ENDIF
3318 C
3319 C - COMPUTE COORDINATE SYSTEM
3320 C
3321 C - SCALE Y-COORDINATE AXIS
3322 NECLEC=1
3323 PHASTP=0.0
3324 CALL NYSXIS(SECTI,NY,NECLEC,PHASOR,PHASTP,PHASMX)
3325 C
3326 C - INCLUDE INTERFERENCE PHASE IN SCALE OF Y-AXIS LIMITS
3327 IF (MSRF.EQ.2.OR.MSRF.EQ.8) THEN
3328 NECLEC=0
3329 CALL NYSXIS(SECTJ,NY,NECLEC,PHASOR,PHASTP,PHASMX)
3330 ENDIF
3331 C
3332 C - MAKE FRACTIONAL Y-AXIS LIMITS INTEGRAL
3333 IF (PHASTP.LE.1.0) THEN
3334 PHSORI=ANINT(PHASOR)
3335 PHSMXI=ANINT(PHASMIX)
3336 IF (PHASOR.LE.PHSORI) THEN
3337 PHASOR=PHSORI-1.0
3338 ELSE
3339 PHASOR=PHSORI

```


PRAM1 (version CD)

```

3340         ENDIF
3341         IF (PHASMX.GE.PHSMXI) THEN
3342             PHASMX=PHSMXI+1.0
3343         ELSE
3344             PHASMX=PHSMXI
3345         ENDIF
3346     C
3347     C - SMALLEST Y-INTERVAL SIZES SHOULD BE PI/4 AND PI/2
3348         PHASDF=PHASMX-PHASOR
3349         IF (PHASDF.LE.2.0) THEN
3350             PHASTP=0.25
3351         ELSE IF (PHASDF.LE.4.0) THEN
3352             PHASTP=0.5
3353         ENDIF
3354     ENDIF
3355     C
3356     C — PLOT COORDINATE SYSTEM
3357         CALL GRAF(XORIG,XSTP,XMAX,PHASOR,PHASTP,PHASMX)
3358     C
3359     C - COMPLETE COORDINATE SYSTEM BY A FRAME AND TICKMARKS
3360         XDUM(1)=XORIG
3361         XDUM(2)=XMAX
3362         YDUM(1)=PHASMX
3363         YDUM(2)=PHASMX
3364         CALL CURVE(XDUM,YDUM,2.0)
3365         IF (MSRF.EQ.5.OR.MSRF.EQ.11.OR.MSRF.EQ.17) GOTO 485
3366         NTIK=NINT((XMAX-XORIG)/XSTP)
3367         YDUM(1)=PHASMX
3368         YDUM(2)=PHASMX-(PHASMX-PHASOR)/50.0
3369         XDM=XORIG
3370         DO 484 ITK=1,NTIK-1
3371             XDM=XDM+XSTP
3372             XDUM(1)=XDM
3373             XDUM(2)=XDM
3374             CALL CURVE(XDUM,YDUM,2.0)
3375     484     CONTINUE
3376     485     CONTINUE
3377         XDUM(1)=XMAX
3378         XDUM(2)=XMAX
3379         YDUM(1)=PHASMX
3380         YDUM(2)=PHASOR
3381         CALL CURVE(XDUM,YDUM,2.0)
3382         NTIK=NINT((PHASMX-PHASOR)/PHASTP)
3383         XDUM(1)=XMAX-(XMAX-XORIG)/50.0
3384         XDUM(2)=XMAX
3385         YDM=PHASOR
3386         DO 486 ITK=1,NTIK-1
3387             YDM=YDM+PHASTP
3388             YDUM(1)=YDM
3389             YDUM(2)=YDM
3390             CALL CURVE(XDUM,YDUM,2.0)
3391     486     CONTINUE
3392     C
3393     C - PLOT PHASE CURVE SEGMENTS; RESET COUNTERS
3394         NPOINTS=0
3395         KINCR=0
3396     C
3397     C — LOOP OVER DATA POINTS; LOOP COUNTER KINCR
3398     487     CONTINUE
3399         KINCR=KINCR+1
3400         IF (SECTN(KINCR).LT.-0.99) THEN
3401             C
3402             C - UNRELIABLE PHASE MARKER ENCOUNTERED; PLOT DATA STRING OF LENGTH

```

PRAM1 (version CD)

```

3403 C   NPOINTS
3404       IF (NPOINTS.GT.0) GO TO 488
3405       ELSE
3406 C
3407 C - INCREMENT DATA STRING COUNTER; PUSH RELIABLE DATA TO FRONT OF VECTOR
3408       NPOINTS=NPOINTS+1
3409       SECTI(NPOINTS)=SECTI(KINCR)
3410       IF (ZVAL.GE.ZSTEP.AND.(MSRF.EQ.2.OR.MSRF.EQ.8)) THEN
3411         SECTJ(NPOINTS)=SECTJ(KINCR)
3412       ENDIF
3413       XX(NPOINTS)=XX(KINCR)
3414     ENDIF
3415 C
3416 C — END LOOP OR CONTINUE IN LOOP UNTIL NY
3417       IF (KINCR.LT.NY) GO TO 487
3418 C
3419 C - LAST DATA POINTS UNRELIABLE; END PHASE PLOTTING
3420       IF (NPOINTS.EQ.0) GO TO 489
3421     488 CONTINUE
3422 C
3423 C - PLOT DATA STRING; RESET DATA COUNTER
3424 C - DRAW INTERFEROMETRIC PHASE SOLID
3425       IF (ZVAL.GE.ZSTEP.AND.(MSRF.EQ.2.OR.MSRF.EQ.8)) THEN
3426         CALL CURVE(XX,SECTJ,NPOINTS,0)
3427       ENDIF
3428 C
3429 C - DRAW CURRENT PHASE DASHED
3430       CALL DASH
3431       CALL CURVE(XX,SECTI,NPOINTS,0)
3432       CALL RESET('DASH')
3433       NPOINTS=0
3434 C
3435 C - JUMP BACK INTO LOOP FOR NEXT STRING OF PHASE DATA
3436       IF (KINCR.LT.NY) GO TO 487
3437     489 CONTINUE
3438 C
3439 C — SPECIAL AXIS AND LABEL FOR FFT COORDINATE
3440 C
3441       IF (NY.GT.8.AND.(MSRF.EQ.5.OR.MSRF.EQ.11.OR.MSRF.EQ.17))
3442     1 CALL XISFFT('X',PHASOR,PHASMX)
3443 C
3444 C - END OF PHASE SECTION PLOT
3445       CALL ENDPL(0)
3446     490 CONTINUE
3447 C
3448 C — CROSS SECTIONS OF AMPLITUDE SURFACES (REAL/IMAGINARY
3449 C REPRESENTATION)
3450       NSRF=MSRF+1
3451 C
3452 C — CHECK EACH ELEMENT IN ROW MSRF OF ARRAY CSEC
3453       DO 590 K=1,NSEC
3454         SECR=REAL(CSEC(NSRF,K))
3455 C
3456 C - ONE-DIMENSIONAL CASES
3457       IF (NY.LE.8) THEN
3458         IF (SECR.LT.0.5.OR.SECR.GT.8.5) GO TO 590
3459         GO TO 501
3460       ELSE IF (NY.LE.8) THEN
3461         IF (SECR.LT.0.5.OR.SECR.GT.8.5) GO TO 590
3462         GO TO 540
3463       ENDIF
3464 C
3465 C - TWO DIMENSIONAL CASES; SECTION ONLY IF IMAGINARY PART OF CSEC--

```

PRAM1 (version CD)

```

3466 C ELEMENT IS EQUAL TO 1.0 OR 2.0;
3467 C OTHERWISE GO TO NEXT LOOP COUNTER VALUE K, I.E. NEXT ELEMENT OF LINE
3468 C MSRF IN ARRAY CSEC
3469 SEC1=AIMAG(CSEC(NSRF,K))
3470 IF (SEC1.GT.0.9.AND.SEC1.LT.1.1) GO TO 540
3471 IF (SEC1.GT.1.9.AND.SEC1.LT.2.1) GO TO 501
3472 GO TO 590
3473 501 CONTINUE
3474 C
3475 C — HORIZONTAL CROSS SECTION (SECOND ARGUMENT OF SRF FIXED); AMPLITUDE
3476 C
3477 C - START A NEW GRAPHICS FRAME FOR THIS CROSS SECTION
3478 CALL RESET('ALL')
3479 CALL AREA2D(GRFSZ,GRFSZ)
3480 CALL INTAXS
3481 CALL MESSAG('SOLID = REAL$',100,0.1,7.35)
3482 CALL MESSAG('DASHED = IMAG$',100,1.7,7.35)
3483 C
3484 C - HEADLINE, LABELS, AND PARAMETER
3485 GO TO(590,590,502, 590,590,503, 590,590,504,
3486 1 590,590,505, 590,590,506, 590,590,507) NSRF
3487 502 CALL HEADIN('RAMAN PUMP: AMPLITUDE$',100,1.5,1)
3488 GO TO 509
3489 503 CALL HEADIN('RAMAN PUMP: MODE AMPLITUDE$',100,1.5,1)
3490 GO TO 509
3491 504 CALL HEADIN('RAMAN STOKES: AMPLITUDE$',100,1.5,1)
3492 GO TO 509
3493 505 CALL HEADIN('RAMAN STOKES: MODE AMPLITUDE$',100,1.5,1)
3494 GO TO 509
3495 506 CALL HEADIN('RAMAN MAT. EXC.: AMPLITUDE$',100,1.5,1)
3496 GO TO 509
3497 507 CALL HEADIN('RAMAN MAT. EXC.: MODE AMPLITUDE$',100,1.5,1)
3498 509 CONTINUE
3499 CALL MESSAG('Z = $',100,5.9,7.1)
3500 IPLACE=2
3501 IF (ABS(ZVAL).GT.9999.0.OR.ABS(ZVAL).LT.0.01) IPLACE=-2
3502 CALL REALNO(ZVAL,IPLACE,6.4,7.1)
3503 CALL XNAME('TIME (PICO-SECONDS)$',100)
3504 IF (NSRF.EQ.6.OR.NSRF.EQ.12.OR.NSRF.EQ.18) THEN
3505 CALL YNAME('MODE AMPLITUDE$',100)
3506 CALL MESSAG('KY = $',100,4.2,7.1)
3507 IPLACE=2
3508 IF (ABS(SECR).GT.9999.0.OR.ABS(SECR).LT.0.01) IPLACE=-2
3509 CALL REALNO(SECR,IPLACE,4.7,7.1)
3510 ISEC=INT((SECR+NYHP/YM2M1)*YM2M1)
3511 ELSE
3512 CALL YNAME('AMPLITUDE$',100)
3513 IF (NY.LE.8) THEN
3514 CALL MESSAG('1 - D TRA.$',100,5.9,7.35)
3515 ISEC=NINT(SECR)
3516 GO TO (512,513,514,515) ITYPE(ISEC)
3517 512 CALL MESSAG('SEC-HYPERB. , EXP = $',100,0.1,7.1)
3518 GO TO 516
3519 513 CALL MESSAG('RECTANGULAR$',100,0.1,7.1)
3520 GO TO 516
3521 514 CALL MESSAG('LORENTZIAN , EXP = $',100,0.1,7.1)
3522 GO TO 516
3523 515 CALL MESSAG('EXPONENTIAL , EXP = $',100,0.1,7.1)
3524 516 CONTINUE
3525 IF (ITYPE(ISEC).NE.2) THEN
3526 XRTYPE=RTYPE(ISEC)
3527 IPLACE=2
3528 IF (ABS(XRTYPE).GT.9999.0.OR.ABS(XRTYPE).LT.0.01)

```

PRAM1 (version CD)

```

3529      1      IPLACE=-2
3530      CALL REALNO(XRTYPE,IPLACE,2.4,7.1)
3531      ENDIF
3532      IF (NY.GT.1) THEN
3533          CALL MESSAG('CASE$',100,4.0,7.1)
3534          CALL INTNO(ISEC,4.7,7.1)
3535      ENDIF
3536      ELSE
3537          CALL MESSAG('Y = $',100,4.2,7.1)
3538          IPLACE=2
3539          IF (ABS(SECR).GT.9999.0.OR.ABS(SECR).LT.0.01) IPLACE=-2
3540          CALL REALNO(SECR,IPLACE,4.7,7.1)
3541          CALL MESSAG('2 - DIM.$',100,5.9,7.35)
3542          ISEC=INT((SECR+RDY*NYHP)/RDY)
3543      ENDIF
3544      ENDIF
3545      C
3546      C - CROSS SECTION DATA
3547      DO 520 K1=1,NT
3548          SECTN(K1)=SRF(K1,ISEC)
3549          SECTI(K1)=SRFI(K1,ISEC)
3550          XX(K1)=TM1+RDT*(K1-1)
3551      520 CONTINUE
3552      C
3553      C - DRAW COORDINATE SYSTEM
3554      NECLEC=1
3555      WSTP=0.0
3556      CALL NYSXIS(SECTN,NT,NECLEC,WORIG,WSTP,WMAX)
3557      NECLEC=0
3558      WSTP=0.0
3559      CALL NYSXIS(SECTI,NT,NECLEC,WORIG,WSTP,WMAX)
3560      CALL GRAF(TORIG,TSTP,TMAX,WORIG,WSTP,WMAX)
3561      C
3562      C - COMPLETE COORDINATE FRAME AND TICKMARKS
3563      XDUM(1)=TORIG
3564      XDUM(2)=TMAX
3565      YDUM(1)=WMAX
3566      YDUM(2)=WMAX
3567      CALL CURVE(XDUM,YDUM,2.0)
3568      NTIK=NINT((TMAX-TORIG)/TSTP)
3569      YDUM(1)=WMAX
3570      YDUM(2)=WMAX-(WMAX-WORIG)/50.0
3571      XDM=TORIG
3572      DO 536 ITK=1,NTIK-1
3573          XDM=XDM+TSTP
3574          XDUM(1)=XDM
3575          XDUM(2)=XDM
3576          CALL CURVE(XDUM,YDUM,2.0)
3577      536 CONTINUE
3578      XDUM(1)=TMAX
3579      XDUM(2)=TMAX
3580      YDUM(1)=WMAX
3581      YDUM(2)=WORIG
3582      CALL CURVE(XDUM,YDUM,2.0)
3583      NTIK=NINT((WMAX-WORIG)/WSTP)
3584      XDUM(1)=TMAX-(TMAX-TORIG)/50.0
3585      XDUM(2)=TMAX
3586      YDM=WORIG
3587      DO 537 ITK=1,NTIK-1
3588          YDM=YDM+WSTP
3589          YDUM(1)=YDM
3590          YDUM(2)=YDM
3591          CALL CURVE(XDUM,YDUM,2.0)

```

PRAM1 (version CD)

```

3592 537 CONTINUE
3593 C
3594 C - DRAW CROSS SECTION CURVES
3595 NPOINTS=NT
3596 CALL CURVE(XX,SECTN,NPOINTS,0)
3597 CALL DASH
3598 CALL CURVE(XX,SECTI,NPOINTS,0)
3599 CALL RESET('DASH')
3600 CALL ENDPL(0)
3601 GO TO 590
3602 540 CONTINUE
3603 C
3604 C - VERTICAL CROSS SECTION ( FIRST VARIABLE FIXED IN SRF); AMPLITUDE
3605 C
3606 C - START A NEW GRAPHICS FRAME FOR THIS CROSS SECTION
3607 CALL RESET('ALL')
3608 CALL AREA2D(GRFSZ,GRFSZ)
3609 CALL INTAXS
3610 C
3611 C - HEADLINE, LABELS, AND PARAMETER
3612 GO TO (590,590,542, 590,590,543, 590,590,544,
3613 1 590,590,545, 590,590,546, 590,590,547) NSRF
3614 542 CALL HEADIN('RAMAN PUMP: AMPLITUDE$',100,1.5,1)
3615 GO TO 549
3616 543 CALL HEADIN('RAMAN PUMP: AMPLITUDE (FFT)$',100,1.5,1)
3617 GO TO 549
3618 544 CALL HEADIN('RAMAN STOKES: AMPLITUDE$',100,1.5,1)
3619 GO TO 549
3620 545 CALL HEADIN('RAMAN STOKES: AMPLITUDE (FFT)$',100,1.5,1)
3621 GO TO 549
3622 546 CALL HEADIN('RAMAN MAT. EXC.: AMPLITUDE$',100,1.5,1)
3623 GO TO 549
3624 547 CALL HEADIN('RAMAN MAT. EXC.: AMPLITUDE (FFT)$',100,1.5,1)
3625 549 CONTINUE
3626 CALL MESSAG('SOLID = REAL$',100,0.1,7.35)
3627 CALL MESSAG('DASHED = IMAG.$',100,1.7,7.35)
3628 CALL MESSAG('Z = $',100,5.9,7.1)
3629 IPLACE=2
3630 IF (ABS(ZVAL).GT.9999.0.OR.ABS(ZVAL).LT.0.01) IPLACE=-2
3631 CALL REALNO(ZVAL,IPLACE,0.4,7.1)
3632 IF (NSRF.EQ.6.OR.NSRF.EQ.12.OR.NSRF.EQ.18) THEN
3633 CALL XNONUM
3634 CALL XTICKS(0)
3635 CALL YNAME('FFT AMPLITUDE$',100)
3636 RDYX=RDYF
3637 XORIG=YFORIG
3638 XSTP=YFSTP
3639 XMAX=YFMAX
3640 ELSE
3641 CALL XNAME('Y-DIMENSION (CM)$',100)
3642 CALL YNAME('AMPLITUDE$',100)
3643 RDYX=RDY
3644 XORIG=YORIG
3645 XSTP=YSTP
3646 XMAX=YMAX
3647 ENDIF
3648 IF (NT.LE.8) THEN
3649 CALL MESSAG('EXPON.. NHYP = $',100,0.1,7.1)
3650 CALL INTNO(NHYP,2.0,7.1)
3651 CALL MESSAG('1 - D STA.$',100,5.9,7.35)
3652 ISEC=NINT(SECR)
3653 IF (NT.GT.1) THEN
3654 CALL MESSAG('CASE$',100,4.0,7.1)

```

PRAM1 (version CD)

```

3655      CALL INTNO(ISEC,4.7,7.1)
3656      ENDIF
3657      ELSE
3658      CALL MSGAG('T = $',100,4.2,7.1)
3659      IPLACE=2
3660      IF (ABS(SECR).GT.9999.0.OR.ABS(SECR).LT.0.01) IPLACE=-2
3661      CALL REALNO(SECR,IPLACE,4.7,7.1)
3662      CALL MSGAG('2 - DIM.$',100,5.9,7.35)
3663      ISEC=INT((SECR+RDT*(NT/2+1))/RDT)
3664      ENDIF
3665      C
3666      C - CROSS SECTION DATA
3667      DO 555 K1=1,NY
3668      SECTN(K1)=SRF(ISEC,K1)
3669      SECTI(K1)=SRFI(ISEC,K1)
3670      XX(K1)=XORIG+RDYX*(K1-1)
3671      555 CONTINUE
3672      C
3673      C - DRAW COORDINATE SYSTEM
3674      NECLEC=1
3675      WSTP=0.0
3676      CALL NYSXIS(SECTN,NY,NECLEC,WORIG,WSTP,WMAX)
3677      NECLEC=0
3678      WSTP=0.0
3679      CALL NYSXIS(SECTI,NY,NECLEC,WORIG,WSTP,WMAX)
3680      CALL GRAF(XORIG,XSTP,XMAX,WORIG,WSTP,WMAX)
3681      C
3682      C - COMPLETE COORDINATE FRAME AND TICKMARKS
3683      XDUM(1)=XORIG
3684      XDUM(2)=XMAX
3685      YDUM(1)=WMAX
3686      YDUM(2)=WMAX
3687      CALL CURVE(XDUM,YDUM,2,0)
3688      IF (NY.GT.8.AND.(NSRF.EQ.6.OR.NSRF.EQ.12.OR.NSRF.EQ.18)) GOTO 568
3689      NTIK=NINT((XMAX-XORIG)/XSTP)
3690      YDUM(1)=WMAX
3691      YDUM(2)=WMAX-(WMAX-WORIG)/50.0
3692      XDM=XORIG
3693      DO 567 ITK=1,NTIK-1
3694      XDM=XDM+XSTP
3695      XDUM(1)=XDM
3696      XDUM(2)=XDM
3697      CALL CURVE(XDUM,YDUM,2,0)
3698      567 CONTINUE
3699      568 CONTINUE
3700      XDUM(1)=XMAX
3701      XDUM(2)=XMAX
3702      YDUM(1)=WMAX
3703      YDUM(2)=WORIG
3704      CALL CURVE(XDUM,YDUM,2,0)
3705      NTIK=NINT((WMAX-WORIG)/WSTP)
3706      XDUM(1)=XMAX-(XMAX-XORIG)/50.0
3707      XDUM(2)=XMAX
3708      YDM=WORIG
3709      DO 569 ITK=1,NTIK-1
3710      YDM=YDM+WSTP
3711      YDUM(1)=YDM
3712      YDUM(2)=YDM
3713      CALL CURVE(XDUM,YDUM,2,0)
3714      569 CONTINUE
3715      C
3716      C - DRAW CROSS SECTION CURVES
3717      NPOINTS=NY

```

PRAM1 (version CD)

```

3718      CALL CURVE(XX,SECTN,NPOINTS,0)
3719      CALL DASH
3720      CALL CURVE(XX,SECTI,NPOINTS,0)
3721      CALL RESET('DASH')
3722  C
3723  C - AXIS AND LABEL FOR FFT COORDINATE
3724      IF (NY.GT.8.AND.(NSRF.EQ.6.OR.NSRF.EQ.12.OR.NSRF.EQ.18))
3725      1  CALL XISFFT('X',WORIG,WMAX)
3726  C
3727  C - END AMPLITUDE SECTIONS
3728      CALL ENDPL(0)
3729  590  CONTINUE
3730      RETURN
3731      END
3732  C
3733  C
3734  C
3735  C
3736      SUBROUTINE NYSXIS(VEC,NPOINTS,NECLEC,VECBOT,VECGAP,VECTOP)
3737  C
3738  C This subroutine was written by Godehard Hilfer (3/87). It finds
3739  C 'nice' end-values (vecbot, vectop) and intervals (vecgap) for
3740  C linear coordinate axes.
3741  C
3742  C The subroutine can find such values around the extremas of the
3743  C argument veC and/or around the input values of the arguments vecbot
3744  C and vectop. This is determined by the argument necveC. If
3745  C      necveC = -1 then the input vector veC is neglected and
3746  C      'nice' limits and interval are only based on
3747  C      the current values of vecbot and vectop. If
3748  C      necveC = 0 then vecbot and vectop are also incorporated
3749  C      in the search for the extrema of vec,
3750  C      thereby allowing user controlled lower
3751  C      limits for these extrema. If
3752  C      necveC = 1 then current values of vecbot and vectop are
3753  C      neglected and 'nice' limits and interval
3754  C      based on the npoints values in veC alone.
3755  C It is also possible to 'hard-wire' the lower (upper) end-value to
3756  C the current value of vecbot (vectop) by setting the argument vecgap
3757  C to -1.0 (1.0) as input. If vectop=2.0 on input both end values are
3758  C 'hard-wired'.
3759  C
3760  C The subroutine finds the extrema of the input data. Then it
3761  C determines the largest integral power of ten (xtrpow) that is still
3762  C smaller than the larger of the absolute values of the extrema.
3763  C Based on xtrpow the leading two decimal places of the extrema are
3764  C compared with each other. The possible difference in the leading
3765  C decimal places the extrema belong to one of seven interval classes
3766  C with the following interval sizes: 0.005, 0.05, 0.1, 0.2, 0.5, 1.0,
3767  C 2.0 times xtrpow. The extremal values are one interval beyond the
3768  C integer that is closest to the extrema. If the hard-wiring option
3769  C was chosen the hard-wired end value is reinstated before the
3770  C interval and end values are returned to the calling routine.
3771  C
3772  C
3773  C
3774  C      mantdif = difference in integral mantissa of extrema
3775  C      mantlw = lower extremum integral mantissa
3776  C      mantup = upper extremum integral mantissa
3777  C      nechrd = hard-wiring flag
3778  C      necveC = flag that picks input data
3779  C      npoints = number of elements to be considered in data vector vec
3780  C      vcevnl = even lower extremum guide

```

PRAM1 (version CD)

```

3781 C      vcevnu = even upper extremum guide
3782 C      vchdbt = hard-wired bottom value
3783 C      vchdtp = hard-wired top value
3784 C      vcmntl = lower extremum divided by dominant power of 10
3785 C      vcmntu = upper extremum divided by dominant power of 10
3786 C      vec = data vector
3787 C      vecbot = data minimum and returns 'nice' lower value
3788 C      vecgap = hard wiring flag on input; 'nice' interval on output
3789 C      vecmax = upper data extremum
3790 C      vecmin = lower data extremum
3791 C      vectop = data maximum and returns 'nice' upper value
3792 C      xtrpow = dominant power of 10
3793 C      xtrpwl = next integral power of 10 below lower extremum
3794 C      xtrpwu = next integral power of 10 below upper extremum
3795 C
3796 C
3797 C      PARAMETER (NT=256,NY=128,NTPY=NT+NY)
3798 C
3799 C      DIMENSION VEC(NTPY)
3800 C
3801 C - STORE INPUT VALUES
3802 C      VCHDBT=VECBOT
3803 C      VCHDTP=VECTOP
3804 C      NECHRD=NINT(100.0*VECGAP)
3805 C
3806 C - CORRECT OR RETURN UPON ERRONEOUS INPUT
3807 C      IF (NECHRD.NE.-100.AND.NECHRD.NE.100.AND.NECHRD.NE.200) NECHRD=0
3808 C      IF (NECLEC.NE.-1.AND.NECLEC.NE.0.AND.NECLEC.NE.1) THEN
3809 C          WRITE (59,*) 'note: NECLEC IN SUBROUTINE NYSXIS OUT OF RANGE'
3810 C          RETURN
3811 C      ENDIF
3812 C      IF (NECLEC.LT.1.AND.VECBOT.GE.VECTOP) THEN
3813 C          WRITE (59,*) 'note: VECBOT IS GREATER THAN OR EQUAL TO VECTOP
3814 C          IN NYSXIS'
3815 C          VECBOT=AMIN1(VECBOT,VECTOP)
3816 C          VECTOP=AMAX1(VECBOT,VECTOP)
3817 C      ENDIF
3818 C
3819 C - FIND EXTREMA
3820 C      NECLEC=NECLEC+2
3821 C      GO TO (810,820,830) NECLEC
3822 C      810 CONTINUE
3823 C      VECMIN=VECBOT
3824 C      VECMAX=VECTOP
3825 C      GO TO 840
3826 C      820 CONTINUE
3827 C      VECMIN=VEC(ISMIN(NPOINTS,VEC,1))
3828 C      IF (NECHRD.EQ.-100.OR.NECHRD.EQ.200.AND.VECMIN.LT.VECBOT) THEN
3829 C          WRITE (59,*) 'warning: FUNCTION EXTENDS BELOW AXIS'
3830 C      ENDIF
3831 C      VECMAX=VEC(ISMAX(NPOINTS,VEC,1))
3832 C      IF (NECHRD.EQ.100.OR.NECHRD.EQ.200.AND.VECMAX.GT.VECTOP) THEN
3833 C          WRITE (59,*) 'warning: FUNCTION EXTENDS ABOVE AXIS'
3834 C      ENDIF
3835 C      VECMIN=AMIN1(VECMIN,VECBOT)
3836 C      VECMAX=AMAX1(VECMAX,VECTOP)
3837 C      GO TO 840
3838 C      830 CONTINUE
3839 C      VECMIN=VEC(ISMIN(NPOINTS,VEC,1))
3840 C      VECMAX=VEC(ISMAX(NPOINTS,VEC,1))
3841 C      840 CONTINUE
3842 C
3843 C - CONSIDER HARDWIRED VALUES AS EXTREMA

```


PRAM1 (version CD)

```

3844     IF (NECHRD.EQ.-100) THEN
3845         VECMIN=AMIN1(VECBOT,VECMIN)
3846     ELSE IF (NECHRD.EQ.100) THEN
3847         VECMAX=AMAX1(VECTOP,VECMAX)
3848     ELSE IF (NECHRD.EQ.200) THEN
3849         VECMIN=AMIN1(VECBOT,VECMIN)
3850         VECMAX=AMAX1(VECTOP,VECMAX)
3851     ENDIF
3852 C
3853 C - FIND DOMINANT INTEGRAL POWER OF TEN FOR THE EXTREMA
3854     RCUT=1.0E-35
3855     IF (ABS(VECMAX).GT.RCUT) THEN
3856         CALL POWBAS(VECMAX,XTRPWU)
3857     IF (ABS(VECMIN).GT.RCUT) THEN
3858         CALL POWBAS(VECMIN,XTRPWL)
3859         XTRPOW=MAX(XTRPWU,XTRPWL)
3860     ELSE
3861         XTRPOW=XTRPWU
3862     ENDIF
3863     ELSE
3864         CALL POWBAS(VECMIN,XTRPOW)
3865     ENDIF
3866 C
3867 C - FIND MANTISSA OF THE EXTREMA
3868     VCMNTU=VECMAX/XTRPOW
3869     VCMNTL=VECMIN/XTRPOW
3870 C
3871 C - CONSTANTS OR EXTREMA THAT DIFFER BY LESS THAN ONE IN THE
3872 C   THIRD SIGNIFICANT PLACE
3873     IF (ABS(VCMNTU-VCMNTL).LE.0.01) THEN
3874         VCEVNU=0.01*(NINT(100.0*VCMNTU)+1)
3875         VCEVNL=0.01*(NINT(100.0*VCMNTL)-1)
3876         VECGAP=0.005*XTRPOW
3877         GO TO 880
3878     ENDIF
3879 C
3880 C - MAKE INTEGER OUT OF THE LEADING TWO SIGNIFICANT PLACES
3881     MANTUP=NINT(10.0*VCMNTU)
3882     MANTLW=NINT(10.0*VCMNTL)
3883     MANTDIF=ABS(MANTUP-MANTLW)
3884 C
3885 C - EXTREMA DIFFER BY LESS THAN 2 PERCENT
3886     IF (MANTDIF.LT.2) THEN
3887         VCEVNU=0.05*(INT(NINT(100.0*VCMNTU)/5)+1)
3888         VCEVNL=0.05*(INT(NINT(100.0*VCMNTL)/5)-1)
3889         VECGAP=0.05*XTRPOW
3890 C
3891 C - EXTREMA DIFFER BY LESS THAN 10 PERCENT
3892     ELSE IF (MANTDIF.LT.10) THEN
3893         VCEVNU=0.1*(MANTUP+1)
3894         VCEVNL=0.1*(MANTLW-1)
3895         VECGAP=0.1*XTRPOW
3896 C
3897 C - EXTREMA DIFFER BY LESS THAN 20 PERCENT
3898     ELSE IF (MANTDIF.LT.20) THEN
3899         VCEVNU=0.2*(INT(MANTUP/2)+1)
3900         VCEVNL=0.2*(INT(MANTLW/2)-1)
3901         VECGAP=0.2*XTRPOW
3902 C
3903 C - EXTREMA DIFFER BY LESS THAN 50 PERCENT
3904     ELSE IF (MANTDIF.LT.50) THEN
3905         VCEVNU=0.5*(INT(MANTUP/5)+1)
3906         VCEVNL=0.5*(INT(MANTLW/5)-1)

```

PRAM1 (version CD)

```

3907          VECGAP=0.5*XTRPOW
3908      C
3909      C - EXTREMA DIFFER BY LESS THAN 100 PERCENT
3910          ELSE IF (MANTDIF.LT.100) THEN
3911              VCEVNU=1.0*(INT(MANTUP/10)+1)
3912              VCEVNL=1.0*(INT(MANTLW/10)-1)
3913              VECGAP=XTRPOW
3914      C
3915      C - EXTREMA DIFFER BY MORE THAN 100 PERCENT (E.G. OPPOSITE SIGN)
3916          ELSE
3917              VCEVNU=2.0*(INT(MANTUP/20)+1)
3918              VCEVNL=2.0*(INT(MANTLW/20)-1)
3919              VECGAP=2.0*XTRPOW
3920          ENDIF
3921      880  CONTINUE
3922      C
3923      C - HARD-WIRED LOWER END VALUE
3924          IF (NECHRD.EQ.-100) THEN
3925              VECTOP=VCEVNU*XTRPOW
3926      C
3927      C - NO HARD-WIRED END VALUE
3928          ELSE IF (NECHRD.EQ.0) THEN
3929              VECTOP=VCEVNU*XTRPOW
3930              VECBOT=VCEVNL*XTRPOW
3931      C
3932      C - HARD-WIRED UPPER END VALUE
3933          ELSE IF (NECHRD.EQ.100) THEN
3934              VECBOT=VCEVNL*XTRPOW
3935          ENDIF
3936          RETURN
3937      END
3938      c
3939      c
3940      c
3941      c
3942          SUBROUTINE POWBAS(VARBLE,PWDECN)
3943      c
3944      C This subroutine was written by Godehard Hilfer (3/87). It determines
3945      C the next lower integral power of 10, pwdecn, of the quantity varble.
3946      C If varble vanishes pwdecn returns 1.0.
3947      c
3948          RCUT=1.0E-35
3949          VABS=ABS(VARBLE)
3950          IF (VABS.GT.RCUT) GO TO 10
3951          PWDECN=1.0E-36
3952          RETURN
3953      10  CONTINUE
3954          XPLOG=ALOG10(VABS)
3955          PWDECN=10.0**INT(XPLOG)
3956          IF (XPLOG.LT.0.0) PWDECN=PWDECN/10.0
3957          RETURN
3958          END

```

APPENDIX B

Manual

MANUAL
RAMAN AMPLIFIER CODE RAM2D1
AND
ASSOCIATED DIAGNOSTIC PROGRAM PRAM1

INTRODUCTION	132
CHAPTER I: GETTING STARTED	134
Part 1.A Computer Access / LOGIN	134
I.A.1: Telephone Access	134
I.A.2: Hardwired Terminal Access	135
I.A.3: Building A68 Access	135
I.A.4: Dec-Server	136
I.A.5: VAX Login	136
I.A.6: VAX Logout	137
I.A.7: CRAY Login/Logout	137
Part I.B File Management	137
I.B.1: Names of the Game	137
I.B.1.1 Code File Names in VAX/VMS	137
I.B.1.2 Relevant Groups of Files	138
I.B.1.3 Modes of Operation and Encryption of Dimensions	139
I.B.1.4 Names of Adjunct Files on the VAX Computer	139
I.B.1.5 CRAY Dataset Names	141
I.B.2: Obtaining the Necessary Files	143
I.B.3: Editing Files	145
I.B.3.1 EDT Screen Editor	145
I.B.3.2 TEDI Line Editor	148

I.B.4: Directories / Delete / Purge	150
I.B.4.1 VAX	150
I.B.4.2 CRAY	151
I.B.5: Running a Batch Job	152
I.B.5.1 The Batch Job Command File	152
I.B.5.2 Submitting a Batch Job	153
I.B.5.3 Batch Job Execution and Termination	154
I.B.5.4 VAX Job Interruption	155
 CHAPTER II: RUNNING RAM2D1 AND PRAM1	156
 CHAPTER III: VIEWING THE RESULTS	158
Part III.A Terminal Output	158
Part III.B Hardcopies	159
III.B.1: Printed Graphs	159
III.B.2: Printed ASCII-Files	160
 CHAPTER IV: SUSTAINING OPERATIONS	162
Part IV.A File Storage	162
IV.A.1: VAX Disk	162
IV.A.2: VAX Tape	163
IV.A.3: CRAY Disk	164
IV.A.4: CRAY Tape	164
Part IV.B Operator Relief	165
IV.B.1: Login Command File	165
IV.B.2: Edit-Aid	167
IV.B.3: VAX/CRAY Status	168
IV.B.4: CRAY Grease	168
IV.B.5: Money Savers	171

Part IV.C Trouble Shooting	171
Part IV.D CRAY Run Specifications	173
APPENDICES	175
APPENDIX A 1-D Transient Limit; Examples	175
EXAMPLE A1	175
EXAMPLE A2	191
APPENDIX B 1-D Stationary Limit; Examples	214
EXAMPLE B1	214
EXAMPLE B2	229
APPENDIX C 2-D Operation; Examples	248
EXAMPLE C	248
APPENDIX D Special Versions of the Codes	269

INTRODUCTION

The manual at hand is intended to introduce the reader to the use of the (2+1)-dimensional Raman amplifier code RAM2D1 and the accompanying diagnostic program PRAM1 as installed on the CRAY X-MP 24¹ computer of the Central Computing Facility (CCF) of the U.S. Naval Research Laboratory (NRL).

Both programs are written in CRAY-FORTRAN (CFT) and run under the CRAY operating system (COS). The computational setup at NRL favors batch job operation. In this mode, the user does not interact directly with the CRAY computer while working with RAM2D1 or PRAM1. Four Digital Equipment Corporation (DEC) VAX² computers (called NRL1, NRL2, NRL3, NRL4) process independently and simultaneously the requests of all users for communication, editing, storage, etc. Any of the four machines can be used interchangeably. Due to size and speed requirements most computations when using RAM2D1 and PRAM1 are done on the CRAY computer. Presently the only computational use of the VAXes is the post processing of the graphics data files that are generated by PRAM1. These data files contain device independent graphics data which the VAX software converts into data that can be displayed on a VT240-type terminal or a laser printer. All other computing is done on the CRAY.

Data storage is available⁴ separately both on the CRAY and on the VAX computers. Both primarily utilize quickly accessible hard disk storage devices. However, both locations offer also the more economical long term tape storage option. All files (datasets) in memory during computation on the CRAY computer are volatile. That means that a computational process has to be given explicitly all the necessary datasets and the results have to be retrieved explicitly from it; otherwise, the datasets disappear upon completion of the job. The resulting data can be sent to the VAX for storage or can be stored on devices that are reserved for CRAY use only.

The data files resulting from the execution of RAM2D1 are programmed to be stored on the CRAY tape storage device (= off-line; CRAY disk = on-line). The code PRAM1 uses these data to produce a DISSPLA-META³ file which is the device independent data file mentioned above. A batch job command transfers this file to the VAX computer post mortem of PRAM1 for storage and/or post-processing. Through the VAX, the data can be displayed or printed.

The remainder of this manual contains explicit instructions and examples pertaining to the use of the computers and the programs RAM2d1 and PRAM1 so that the user can, with a particular input parameter choice in hand, run the codes and carry the results home on paper.

- 1 CRAY X-MP (and other CRAY logos) is a registered trademark of CRAY Research, Incorporated, Mendota Heights, MN.
- 2 VAX, DEC, and others are registered trademarks of the Digital Equipment Corporation, Maynard, MA.
- 3 DISSPLA is a registered trademark of the Integrated Software Systems Corporation, San Diego, CA.

CHAPTER I

GETTING STARTED

PART 1.A COMPUTER ACCESS / LOGIN

Section I.A.1: Telephone Access

For remote access, by means of a personal computer and the telephone network, find appropriate communications software (e.g. VTEK, KERMIT, or other) to dial Washington, D.C., metropolitan area phone number 767-2000 for a 1200 baud connection to the Naval Research Laboratory Central Computing Facility (NRL CCF). For a 2400 baud connection dial the number 767- 1240.

Should you have problems call 767-3512 for a status information on the CCF, or call the consultants desk 767-3542 for assistance Mondays through Fridays from 9am to 5pm.

After the connection is made type: < (carriage return)

<

.

.

.

two or more carriage returns until the computer prompts: # From here proceed to section I.A.4: DEC-server.

Section I.A.2: Hardwired Terminal Access

Access through one of the terminals at the CCF is obtained in the following way:
Turn the power on.

```
type:      <      (return)
prompts:    You may now enter Net/One commands
            >
            >

type:      c cts <
prompts:    connecting ... (1) ----- success
            #
```

From here proceed to section I.A.4: DEC-server.

Section I.A.3: Building A68 Access

The terminals in building A68 at NRL (John Reintjes' section) are connected to the communications server CS/200T which in turn is hardwired directly to the front-end VAX computers. Thereby the DEC-server involved in all other access paths is circumvented. Proceed as follows: Turn the power on.

```
type:      < (return)
prompts:    CS/200T>
type:      c nrl <
```

which will establish connection to NRL3 (alternatively type: c nrl1 <, or c nrl2 <, or c nrl3 <). From here proceed to section I.A.5: VAX login. In building A68 NRL4 can only be accessed through the DEC-server. For that, turn the power switch of the terminal to ON

```
type:      < (return)
prompts:    CS/200T>
type:      c lat-gw <
prompts:    Querying Primary Name Server...
            Connecting... session 1 -- connected to lat-gw

type:      <
prompts:    Local>
```

which indicates successful access to the DEC-server. From here continue with what follows the prompt Local> in section I.A.4: DEC-server below.

Section I.A.4: Dec-Server

When the computer

```
prompts:  #  
type:     n < (Note: the letter n will not show up on the screen!)  
prompts:  Enter username>  
type:     'your username' < (=name under which you may use the VAXes)  
prompts:  Local>
```

Now you have accessed the so-called DEC-server. (type: `help` < if you wish on-line information about this networking facility; otherwise:)

```
type:     c nrl <
```

to be connected with one of the four VAX front-end computers. You may explicitly specify the VAX or your choice (e.g. `c nrl3` < , to get onto the NRL3 computer etc.). A standard VAX-login ensues. Proceed to section I.A.5: VAX Login.

Section I.A.5: VAX Login

The last pressing of the return key should effect that the system

```
prompts:  Username:  
  
whereupon it is necessary to  
type:     'your username' <  
prompts:  Password:  
type:     'your password' < (will not echo)
```

Then the VAX computer executes the login which will be finished when a \$-sign appears on a line by itself following all other text on the screen.

From here proceed to Section I.B.2: Obtaining the Necessary Files, if you do not have them; or continue with Section I.B.3: Editing Files, if you do have all the files but need to change something; or turn to Chapter II, if you have the correct set of files for the intended simulation, or, if the simulation was previously done and the results need to be converted into graphs; or Chapter III, Viewing the Results, to see the graphs actually come out on the terminal screen or on paper; or turn to Part IV.A., File Storage, if programs or data need to be moved into storage or removed from it.

Section I.A.6: VAX Logout

To leave the computer system one has to logout. This procedure terminates all access to and responses from the computer. To logout

type: log <

which, e.g.,

```
prompts:  USER      logged out at d-a-t-e t:i:m:e
          Local - Session x disconnected from NRL
          Local>
```

This is the prompt of the DEC-server network node. A second LOG is necessary to signoff from it and to free the port of access. Thus,

type: log <

which will be acknowledged by telling from which port was logged off. In all it takes two log to finish the computing session. The second log is not necessary but neither harmful when using the building A68 communication server CS/200T. After that the terminal is disconnected from the CCF.

If the front-end VAXes do not obtain new input from the terminal within roughly 8 minutes, a ten minute countdown associated with two warnings, 5 minutes apart, ensues followed by an automatic logout of the inactive user.

Section I.A.7: CRAY Login/Logout

Access of the batch job to the CRAY is authorized by means of the first two batch job file command lines: 'JOB,---.' and 'ACCOUNT,---.' (see also subsection I.B.5.1: The Batch Job Command File). Access to the CRAY computer and its storage facilities is limited to the command lines in the batch job command file.

PART I.B FILE MANAGEMENT

Section I.B.1: Names of the Game

I.B.1.1 Code File Names in VAX/VMS

The nomenclature of all relevant files is as follows. The names of the source codes as indicated above, are RAM2D1 and PRAM1. The name RAM-2D- 1 abbreviates that this code solves the Raman amplification problem in 2-D. *i.e.*, the two spatial

dimensions: z (linear coordinate along the central Stokes beam ray path) and y (linear coordinate orthogonally transverse to z). The character 1 in the name indicates that this is the first generation of this code. The diagnostic program name P-RAM-1 abbreviates: plots of the Raman amplifier code, 1st generation.

Both source codes reside on the VAX computers. Following the VAX/VMS operating system particulars, their full VAX-file names are:

DUA107:[HILFER.FOR]RAM2D1C.FOR;1
DUA107:[HILFER.FOR]PRAM1CD.FOR;1.

According to VAX/VMS conventions the name elements and their meanings are: DUA107: indicates the specific storage disk name on which the file is stored. [HILFER.FOR] indicates that the file belongs to the subdirectory FOR of the HILFER directory of files on that disk. The code name RAM2D1 was supplemented by suffix C to indicate version C of the code (see Appendix D for details and other versions). The arbitrarily chosen file extension .FOR is a reminder that the file contains a FORTRAN code. The file version ;1 is a number that serves the VAX computer to distinguish files of the same name. Every time the file is amended or changed, the VAX computer will keep the old file with its full old name and will create a new file with amendments and/or changes that will be given the same name but a version number one greater than that of the old file. Therefore, the highest version number indicates the most up-to-date version of the same file. The characters CD in PRAM1CD indicate that this version of PRAM1 works with RAM2D1C and RAM2D1D (i.e. on the NRL-CRAY, as opposed to PRAM1AB, which works with RAM2D1A and RAM2D1B on the NMFEC-CRAYs).

All relevant files are stored by default in the same directory on the same disk. Hence, the file name portion DUA107:[HILFER.FOR] is the same for all files and will be dropped in this manual for brevity's sake. Since the version number may be larger than 1 depending on, and only significant for, code development, the user can neglect it and it will be dropped also. Thus, the code names reduce to, simply,

RAM2D1C.FOR
PRAM1CD.FOR

I.B.1.2 Relevant Groups of Files

The relevant files can be grouped by their file name extensions (i.e., three characters following the dot in the full file name analogous to what was described in subsection I.B.1.1: Code File Names in VAX/VMS). There are the following groups:

- .FOR (the 2 source code files mentioned in subsection I.B.1.1)
- .DAT (input and output data files)
- .JOB (batch job command files containing the user's commands for the CRAY computer)
- .CPR (message files generated by the CRAY computer system during job execution containing listings, messages, and a batch job log)
- .MSG (message files generated by the FORTRAN code during job execution containing formatted and unformatted output as programmed by the code developers.)
- .TMP (device specific graphics data files that can be printed on the laser printer)

I.B.1.3 Modes of Operation and Encryption of Dimensions

The code's operation as a two-dimensional or one-dimensional model is switched by the field array dimension parameters NT and NY. If both integers are larger than eight, two-dimensional operation is indicated and the algorithm expects that the parameters are set to integral powers of 2. If one of the parameters is 8 or smaller, the variable (*t* or *y*) associated with that parameter ceases to be a variable, and refers instead to the number of cases being run in the one dimensional mode. Both NT and NY must never be 8 or less simultaneously.

In short, the values of NT and NY are salient characteristics of any simulation and serve, therefore, to distinguish data files and code versions by contributing two characters to every file name. The first of both characters indicates the value of NT, the second that of NY according to the following scheme. If the value is 8 or less, that value is used as one file name character. The one (or both) parameter(s) that is larger than 8, which must be an integral power n of 2, is represented by the n -th character of the alphabet. For example, if NT=5 and NY=1024=2¹⁰, one finds the character 5 followed by the tenth character of the alphabet (=J) as a two character block (--5J--.--), in all relevant file names. For a list of typical encrypted dimensions and their NT $\times$ NY equivalence, see the table in section V.D.2.

I.B.1.4 Names of Adjunct Files on the VAX Computer

The other relevant files that reside on the VAX besides the FORTRAN source codes are data files, message files and batch job command files.

INPUT DATA FILE

The input data files can be distinguished by a file name beginning with the character N followed by three more characters and ending with the extension .DAT. E.g.

NR1J.DAT

NPGI.DAT

The second file name character is either R, if this is an input data file for RAM2D1, or P, if this is an input data file for PRAM1. The third and fourth character are the two character block that contains the values of the code parameters NT and NY encrypted as described in subsection I.B.1.3 .

GRAPHICS DATA FILES

There can be two types of output data files on the VAX. One is the so-called META-file by the name

PLT2.DAT

which is generated by the DISSPLA-graphics subroutines in PRAM1. The other is the data file that the DISSPLA-postprocessing software on the VAX generates with the name

INTSCRT.TMP

when the graphs in PLT2.DAT are requested as laser printer hardcopies. It is the duty of the user to find a means of distinction for these equally named output data files from a series of simulations. It is suggested to rename these files mnemonically. This is easily done by the VAX command RENAME,

type: RENAME PLT2.DAT 'new file name'

following the VAX-prompt \$.

MESSAGES FILES

Two types of message files can be found in the VAX user directory. Except for a varying file extensions, these files have the same name as the batch job command file (see next paragraph) from which they originated. There are MSG-files. One such file is created if a code generates output due to formatted and/or unformatted write statements. These statements constitute the sole content of this file. The file is identified by its .MSG file extension. For example,

CR1J.MSG

XPGI.MSG

Secondly, there are CPR-files one of which is generated by the CRAY computer every time a job is run. For example,

CR1J.CPR
XPGL.CPR

These files document the batch job execution by recording information such as: program listing, error messages from the CRAY operating system and the CRAY compiler timing information regarding batch job execution, space and cost information and more esoteric information relating to the CRAY computer usage.

JOB FILES

The batch job command files have a name similar to the input data files. The only two differences being the .JOB file extension instead of .DAT, and the initial letter being C or X instead of N. For example,

CR1J.JOB
XR1J.JOB
CP1J.JOB
XP1J.JOB

A first letter X indicates a job file that executes the code associated with it (see second letter of job file name: R for RAM2D1, P for PRAM1). A first letter C indicates a job file that will first compile and assemble the source code before running the newly created executable file.

All file names mentioned above apply to the VAX directory of files [HILFER.FOR]. When a batch job fetches a file (source code or input data file) from VAX storage and transfers it to the CRAY during job execution, the VAX name (specified by TEXT='---' on the FETCH command line in the job file) is changed to a CRAY dataset name (as given by DN='---' on the same FETCH command line).

I.B.1.5 CRAY Dataset Names

SOURCE CODE

The source codes have a three character dataset name when used on the CRAY computer. The first character is R (or P) for RAM2D1 (or PRAM1). The second and third character give the NT and NY parameter values as described in subsection I.B.1.3. For example,

R1J is RAM2D1 on the CRAY with. NT=1, NY=2¹⁰
PGI is PRAM1 on the CRAY with. NT=2⁷, NY=2⁹

Either dataset appears on the CRAY following a FETCH command line in a C—.JOB file and disappears automatically following completion of the job.

EXECUTABLE DATASET

The executable dataset resulting from compilation of either source code is usually kept (SAVE command line in JOB-file) under the same dataset name as its parental source code, but amended by a preceeding X. For example,

XR1J

XPGI

INPUT DATASET

The input dataset to RAM2D1 following a FETCH form the VAX is named
NRAM,

the input dataset to PRAM1 is named

NPRAM1

on the CRAY computer.

OUTPUT DATASET

The output resulting from execution of RAM2D1 is contained in a single CRAY dataset when running the code one-dimensionally. When operating the code two-dimensionally, the number of output datasets is proportional to the number of

z-locations at which field data are kept. All of these data files are saved automatically in the CRAY off-line storage facility.

All output dataset names begin with the letter F followed by eight alphanumeric characters if the file results from one-dimensional code operation, and followed by eleven alphanumeric characters if the file results from two-dimensional code operation. The second and third character in these dataset names are the two character block that contains the values of the code parameters NT and NY encrypted as described in subsection I.B.1.3. The following six characters contain the date at which the execution of RAM2D1 began. In two-dimensional operation three more numerals (a counter) are appended to this same name which number the individual field datasets consecutively as they are created. For example,

F1J101587 (field dataset with arrays dimensioned NT=1, NY=2¹⁰, started on
October 15, 1987)

FGI101587000 (field datasets with arrays
FGI101587001 dimensioned $NT=2^8$, $NY=2^9$, started
FGI101587002 on October 15, 1987, at different
FGI101587003 z-values)

This counter is 000 for the dataset that contains the list of setup parameters and initial field data. Its purpose is to enable the user to view output data with the diagnostic code PRAM1 immediately as they become available during an extensive run. Such concurrent diagnosis has to be indicated to PRAM1 by setting its input parameter DONYET to 0 (DONYET should be 1 during regular post mortem diagnosis).

This counter is 001 for the dataset that contains the setup parameters (like -000 dataset), the field data at $ZVAL=0.0$ (like -000 dataset), and the timing information gathered at the end of the run (unlike -000 dataset). This counter is 002 for the dataset that contains the field data at $ZVAL=1*ZKEEP$, 003 at $ZVAL=2*ZKEEP$, 004 at $ZVAL=3*ZKEEP$, etc.

MESSAGE DATASET

User defined messages (mostly conditional error messages) from RAM2D1 (PRAM1) are gathered in dataset ERRM (EPRM) which is transferred to the VAX under the name of the current JOB-file but with the file extension .MSG . The other message dataset from each run, the CPR-file, is created by the operating system and not accessible to the user until after it is transferred to the VAX post mortem of the run.

Section I.B.2: Obtaining the Necessary Files

Six files are required to simulate the Raman interaction numerically. These are the FORTRAN source codes

RAM2D1 and
PRAM1

(see subsection I.B.1.1 for full VAX/VMS file names), their respective input data files

NR--.DAT and
NP--.DAT,

and their respective batch job command files

CR--.JOB and
CP--.JOB.

The dashes -- stand for the particular 2-character block as the choice of dimensions, described in subsection I.B.1.3, necessitates.

Unless the user has immediate access (password) to the [HILFER.FOR]- subdirectory it will be necessary to copy these files from there into the user's own directory. The VAX/VMS copy command serves this purpose. When the VAX

```
prompts:      $
type:         COPY DUA107:[HILFER.FOR]RAM2D1C.FOR *.* <
prompts:      $
```

(Should an error message appear, e.g. copy protection violation or insufficient privilege, contact the CCF consultants desk at (202)767-3542 or Godehard Hilfer at (202)767-2028).

```
type:         COPY DUA107:[HILFER.FOR]PRAM1CD.FOR *.* <
prompts:      $
type:         COPY DUA107:[HILFER.FOR]NR--.DAT *.* <
prompts:      $
type:         COPY DUA107:[HILFER.FOR]NP--.DAT *.* <
prompts:      $
type:         COPY DUA107:[HILFER.FOR]CR--.JOB *.* <
prompts:      $
type:         COPY DUA107:[HILFER.FOR]CP--.JOB *.* <
prompts:      $
```

Now all necessary files are in the user's current directory. From this directory the batch job should be submitted in order for the automatic substitution of default values for user disk, default directory etc. in the abbreviated file names as they appear in the batch job command file to work. The message and data files that the job sheds will be send to this directory from which the job was submitted.

Once the dimensionality of the intended simulation is known, the corresponding NT and NY values will have to be encoded as described in subsection I.B.1.3 and filled into all the file names of this section. Remember to insert/replace these two characters also into/in appropriate positions in all file names and dataset names contained in the two JOB-files! Remember also to verify/change all occurrences of NT=--- and NY=--- in both source codes accordingly.

The process of inserting/replacing these characters is called 'editing the file.' The computer software that accomplishes this task is called an 'editor.' A rudimentary description of two selected editors is described below in section I.B.3.

Section I.B.3: Editing Files

I.B.3.1 EDT Screen Editor

In order to make amendments, deletions or any other changes in a file (e.g. an input data file), that file needs to be accessed by an editor program. The preferred editor of the VAX/VMS operating system is called EDT. It accesses any file in the following way. When the VAX

prompts: \$
type: SET TERMINAL/VT100 <

to identify to the editor what industry standard terminal to expect. This setting needs to be made only once after login, not every time the editor is invoked. Giving this setting repeatedly is merely redundant. However, it needs to be set once for the editor to work properly. The terminal used should actually be a DEC VT100 terminal as indicated by the command, or at least emulating such; otherwise, the appropriate setting will have to be found from the VAX/VMS reference manual. Ideally, the user should have a VT240-type terminal to work with. Without its graphics capability it will not be possible to view the output from PRAM1 on the screen. Such terminal is otherwise fully compatible with the VT100 industry standard and will, therefore, work fine in the editor given the above setting. This setting is taken by the VAX without any special response, it just

prompts: \$
Then
type: EDIT/EDT 'filename.extension' <

and fill in for 'filename.extension' the name of the file that shall be edited.

CREATING/EDITING A NEW FILE

The same command

EDIT/EDT 'filename.extension'

can also be used to create a new file by filling in a filename that is not yet in the directory. (To see which files are already in the directory see below in section I.B.4.)

In that case the system

prompts: Input file does not exist
 [EOB]
 *

The star indicates that the editor is in its default mode which is the line editing mode. However, the power and primary function of EDT is its screen editing capability. To change to screen editing mode

type: c <

following the star prompt. Then the screen will be erased and in the top left corner appears the [EOB] indicating the end of the buffer. Buffer is the name for storage space that is volatile. The characters stored in it will disappear after the process to which the buffer belongs is terminated unless the buffer is purposely saved. Anything that the file contains, can now be typed into the buffer. The 'end of the buffer' indicator moves automatically down the screen as characters are inserted. The buffer is saved and becomes the desired file if the editing session is ended with the END instruction. The alternative would be to finish editing with the QUIT instruction where upon the buffer is discarded leaving no trace of the editing session whatsoever. To finish either way

type: ^ z (Ctrl z ; i.e. while holding the Ctrl key on the
 keyboard down type a 'z', then release both keys;
 no additional return key stroke is necessary;
 although it would do no harm)

The editor will return to the line editing mode that

prompts: *

To exit

type: exit < (to exit and to save the buffer content in a disk file)

or

type: quit < (to exit and to lose the buffer content)

EDITING AN EXISTING FILE

If the 'filename.extension' in the EDIT/EDT command line

EDIT/EDT 'filename.extension' <

matches one, or several, entries in the current directory the editor will access the one of these files that has the highest version number. Access is accomplished when the computer

prompts: 1 ----- 'text of first line in file'-----
 *

This star is the line editor mode prompt.

type: c <

to get into screen editor mode.

SCREEN EDITING TOOLS

Most screen editing consists in moving the cursor to the desired position on the screen and then entering characters there, by typing them, or deleting characters there. For this the essential tools are the special keyboard keys:

arrows (*left, right, up, down*; move the cursor one field at the time by pressing the key shortly; scroll the cursor in that direction by holding the key down)

delete (erases a character to the left of the current cursor position)

PF4 (erases a whole line following the current cursor position at once)

PF1 PF4 (undoes the last delete of the *PF4* key)

The set of 18 keys in the lower right corner of the keyboard is called keypad. Its keys, designated in this manual by a preceding *P* (e.g. *P4* is keypad key 4), have special functions in EDT (e.g. *PF1* and *PF4* described above). To view a description of these functions press the *PF2* key. For the extensive user of the VAX, it is desirable to memorize the use of the keypad. For the occasional user it shall suffice to mention the block delete/move procedure: select desired block of text by marking invisibly one end by hitting *P*. (that is the . key on the keypad) (undo erroneous use of that key by pressing *PF1* followed by *P*.); Use the arrow keys to move the cursor to the other end of the intended block boundary; press *P6*; now the block is moved from the displayed text buffer into a hidden text buffer. From there it can be copied to the current cursor position as often as desired by pressing *PF1* followed by *P6*. The block will remain in the hidden buffer until another block delete overwrites it, or until the editor is exited.

Standard editing shows a maximum of 80 characters per column. To view CPR-files it is appropriate to display 132-characters per line. To change to that format

type: *PF1 P7 SET SCREEN 132 PEnter* (*PEnter* is the *enter*-key on the keypad)

Very, useful particularly when viewing a CPR-file, are the EDT-commands for fast scroll to end or beginning of the file:

type: *PF1 P4* (for fast scroll to the end of the file),
type: *PF1 P5* (for fast scroll to the beginning of the file),

The key *P8* is not quite that fast, but still faster than the arrow keys, in scrolling forward or backward in the file. If preceded by *P4*, *P8* will scroll 16 lines forward, if preceded by *P5*, *P8* will scroll 16 lines backward. The direction key *P4* or *P5* needs to be pressed only once. *P8* can be applied repeatedly thereafter.

These are the basic EDT screen editing commands that the user will need. Further detail can be found on line (press *PF2*) or in the VAX/VMS reference manual on EDT.

1.B.3.2 TEDI Line Editor

The widely used line editor TEDI shall be introduced because of its convenient pattern search and replace operation. Line editing consists in displaying and modifying a particular line or several lines at the same time.

For the TEDI editor to access the file 'filename.extension',
type: TEDI 'filename.extension' <.
This
prompts: DUA107: [DIRECTORY]filename.extension;1 ---LINES
*

The star is, just like in the EDT editor, the line mode prompt.

TEDI commands consist of one or a few acronymic letters accompanied by one to three line numbers separated by commas and, separated by semicolons, followed by one or two character strings, depending on the particular command.

The TEDI editor can list and replace efficiently all occurrences of a given character pattern. This is useful when checking and/or changing the dimensionality of the field arrays in the source codes. To accomplish this

type: TP1,500;NY=; <

following the star prompt. This instructs the computer to type all lines between line 1 and line 500 in the currently accessed file that contain the pattern: NY=. Note that TEDI distinguishes letters also by their capitalization. To search the whole file one needs to replace 500 by a number equal to or larger than the total number of lines in the file or, if unknown, to replace 1,500 by the wildcard symbol *. For example,

'tp*;NT='. The command accronyms can be small or large case letters. The last semicolon may be and was omitted.

To replace all occurrences of NY=1 by NY=512, for example,
type: RP1,500;NY=1;NY=512; <

The type pattern (TP) command preceeding the replacement (RP) is somewhat tedious but efficient if there is any doubt about possibly unwanted replacements like: ISNY=1. Therefore, TP should be employed to make sure that the intended pattern string is unique.

Portions of a file can be viewed by the type command:

T1,500 <

would scroll lines 1 through 500 across the screen. The command

T* <

scrolls the whole file. An individual line (e.g. line 500) can be deleted by

DL500 <

Several lines are deleted by giving the range (e.g. line 1 through 500)

DL1,500 <

Caution! Deletes cannot be restored in TEDI except for the price of giving up all the other editing that was done beforehand through an emergency exit from the editing session (type: quit).

New lines can be added before (BL) or after (AL) any specified line number. For example,

BL1 <

starts the insertion of lines before the current line number 1. Insertion mode is indicated by the '>'-prompt. All following characters will be inserted sequentially as typed. Another new line is inserted with every return '<'. Insertion mode is ended by typing a '.' by itself on a new line.

A detailed description of the TEDI editor is on file in the CCF consultants office or can be purchased from the CCF operator desk.

Section I.B.4: Directories / Delete / Purge

I.B.4.1 VAX

DIRECTORY

A listing of the directory of files on the VAX can be viewed in the following way: Change, if necessary, the directory information that is contained in the omitted portion of the complete file name to the desired directory DISK: [USER.SUBDIRECTORY]. To this end

```
type:          SET DEFAULT DISK: [USER.SUBDIRECTORY] <
prompts:       $
```

Then the listing of files in that subdirectory appears after you

```
type:          DIRECTORY <
```

may be shortened to DIR <.

The DISK: specification may be omitted if unchanged. The .SUBDIRECTORY specification has to be omitted to see the main [USER] directory list of files. Multiple level subdirectories can be listed in the same way by continuing the path of directories starting with the main directory in the analog fashion:

```
SET DEFAULT DISK: [USER.SUBDIR.SUBSUBDIR.SUBSUBSUBDIR] <
```

The plain listing of all files can be more elaborate by means of file name portion, filters, and options following the DIRECTORY command. For details

```
type:          HELP DIRECTORY <
```

which can be terminated by one or several '<' returns.

DELETE

To delete an entry from the directory of files and thereby destroy that file

```
type:          DELETE 'filename.extension;version' <
```

The specified file name is removed from the default directory (see I.B.1.1) only. It is necessary to specify the version number otherwise no deletion will take place but rather an error message will appear on the screen. The three pieces in the name of the file: filename, extension, and version can be substituted with the wild card character '*' in order to generalize the command to delete all files that match the specification except for the name piece represented by the '*'. For example,

```
type:          DELETE NRAM.DAT;*
```

to delete all versions of the file NRAM.DAT (contrary to PURGE NRAM.DAT which leaves the highest version). For example,

type: DELETE N*.DAT;*

to delete all files whose names begin with the letter N, by the file extension .DAT from the directory. For more sophisticated usage of the DELETE command

type: HELP DELETE

which can be exited by one or several '<' returns.

PURGE

To purge the default directory of files is to remove all file versions except for the last (highest) one. To purge the current VAX default directory simply

type: PURGE <

The PURGE command can be made more specific. For details

type: HELP PURGE <

which can be exited by one or several '<' returns.

I.B.4.2 CRAY

The simple functions of listing the file directory, purging it and deleting particular entries are somewhat time consuming on the CRAY computer due to the batch job setup. Therefore, a batch job has to be submitted to accomplish these tasks. How to submit a batch job will be demonstrated in the next Section I.B.5: Running a Batch Job.

DIRECTORY

The listing of the files in the user directory on the CRAY disk is obtained in the CPR-output file of any CRAY job if the job command file contains the command line with the command:

AUDIT. .

This is usually the case with every batch job, hence, the need for at-will CRAY directory information is small. Nevertheless, the job command file

DUA107:[HILFER.FOR]CAUDIT.JOB

can be copied to do only that when submitted as a batch job.

DELETE

In order to delete a file in the CRAY directory a batch job command file needs to be submitted that contains the appropriate DELETE command line. For example,

```
DELETE,PDN='filename'.
```

The user may wish read the details of DELETE command line in the CRAY operating system (COS) manual. The quickest path for the new user is simply to copy the file DUA107:[HILFER.FOR]CDELET.JOB into the current directory, to change the file name contained in it as desired, and to submit it for execution. Notice that the '-' character serves as the wild card character of COS representing any string of characters.

PURGE

In order to purge files in the CRAY directory, i.e. delete all versions but the latest of each file, a batch job command file needs to be submitted to the CRAY that accomplishes to delete in a selective way. For example,

```
DELETE,PDN=-,ED=-1.
```

The user can find the details of the DELETE command in the COS-manual. A simpler path is to copy the file DUA107:[HILFER.FOR]CPURGE.JOB into the user directory, and to edit the contained file names such that all file names to be purged are covered by the specified file name pieces in combination with wild cards. Recall that on the CRAY the symbol '-' is the wild card for any string of characters.

Section I.B.5: Running a Batch Job

I.B.5.1 The Batch Job Command File

The execution of a computation on the CRAY computer as a batch job requires several steps which are listed as command lines in the batch job's .JOB-file. Once this file is transferred to the CRAY it will be queued in the batch job queue. When its turn for execution comes around, the operating system will execute all command lines sequentially, waiting for each command to finish before picking up the next one. Should a terminal error occur, execution will be stopped. At the end of each job a log-file will be sent to the user's VAX directory.

There are a few rules concerning the form of the batch job command file: Beginning in column 1, every line must start with a command verb that is known and accepted by the operating system. Every line must end with a period ('.'). Several parameters

may follow the command verb separated by commas. The first command line in the file must be the JOB-statement:

JOB,JN='job name'.

(CBATCH processing waives this requirement, see section V.B.4. The name that the job shall have has to be inserted. The second command line must be the ACCOUNT-statement:

ACCOUNT,AC='account number',US='user
number',UPW='user password'.

(CBATCH processing waives this requirement, see section V.B.4 which has to be completed by the three appropriate fill-ins: account number, user number, and user password. The next command lines contain the desired CRAY action followed by the command line:

EXIT.

Note that all JOB-files that are copied from the DUA107[HILFER.FOR] directory lack the JOB and ACCOUNT command line which will have to be supplied by the user.

I.B.5.2 Submitting a Batch Job

It is recommended to precede the submission of the first batch job, when the VAX prompts: \$

with the following VAX command,

type: CRAY SET TERMINAL INFORM <

This will inform the CRAY computer of the location of the user's terminal and, hence, enable forwarding of the messages that accompany the execution of the job.

For the actual submission of the JOB-file
type: CRAY SUBMIT 'filename'.JOB <

where the JOB-file's filename has to be inserted. This will queue the job file for transfer to the CRAY and subsequently queue it for execution. For example

type: CRAY SUBMIT CAUDIT.JOB <

prompts: \$

% CX-S-SUB_OK, Job: CAUDIT queued for submission

\$

VAX TO CRAY: % SYSTEM-S-NOMRAL, normal successful completion

VAX TO CRAY: FILE=CAUDIT

VAX TO CRAY: 4608 BYTES TRANSFERRED

\$

which are the standard messages of verification for the queuing for submission and for the transfer of the JOB-file for the CRAY computer.

I.B.5.3 Batch Job Execution and Termination

The execution of the job is determined by the CRAY operating system. During the execution of RAM2D1 and PRAM1 other files are transferred from the VAX to the CRAY. Each transfer is accompanied by a message of the type

VAX TO CRAY: % SYSTEM-S-NOMRAL, normal successful completion

VAX TO CRAY: FILE=NRJ1

VAX TO CRAY: 4608 BYTES TRANSFERRED

Progress of execution can be monitored through on-demand status messages. For this purpose

type: CRAY STATUS/OWN <

prompts:

cray	system	status	EIORS	PRIMARY	17-feb-1988	11:39:50.19				
jsd	dc	dataset	class	status	pri	used	limit	length	id	tid
12596	IN	CAUDIT	SMALL	QUEUED	6.0	0	60	512	V2	HILFER

the explanation of each detail for which all would break the frame of this manual. The important points, however, are the STATUS, the number of seconds USED, and the number of seconds LIMIT for the job. Those three items are self-evident.

The termination of a job occurs usually automatically when the EXIT. command line in the JOB-file is executed. Such normal (and other unusual) termination is indicated by the transfer of the CPR-file from the CRAY to the VAX as notified of by a message of the following type:

CRAY TO VAX: % RMS-S-NORMAL, normal successful completion

CRAY TO VAX: FILE=1DUA107:[HILFER.FOR]CAUDIT.CPR;1

CRAY TO VAX: 1706 BYTES TRANSFERRED

Another definite indication is when the response to the status request explained just above is responded by only the first two headlines, showing no job sequence number. The successful transfer of the CPR-file does not indicate that the program ran successfully. This can only be seen from the bottom portion of the CPR-file.

Unusual termination can be due to, e.g., programming errors, command line errors, too small a time limit (job needs more CPU-time than the allocated amount; =60sec by default), forced by the user and other reasons. When a submitted job needs to be stopped, obtain at first the jsq-number from the CRAY STATUS/OWN report, then type:

```
CRAY KILL'jsq-number' <.
```

This will result in the termination of the job that is documented as such in the subsequently issued CPR-file.

1.9.5.4 VAX Job Interruption

An emergency stop of any VAX DCL-command can be forced by typing ^ Y (=Ctrl Y). This causes the VAX computer to interrupt whatever it was engaged in and to return to the \$-prompt, ready for a new command.

CHAPTER II

RUNNING RAM2D1 AND PRAM1

To perform the actual Raman amplifier simulation, one only needs to submit a JOB-file that compiles the source code RAM2D1 and runs the resulting executable file. Hence

type: `CRAY SUBMIT CR--.JOB`

where the '--' holds the place for the appropriate dimensionality characters (see PART I.B).

To diagnose the results of a Raman amplifier simulation, one only needs to submit a JOB-file that compiles the source code PRAM1 and runs the resulting executable file. Hence

type: `CRAY SUBMIT CP--.JOB`

where the '--' holds the place for the appropriate dimensionality characters (see PART I.B).

As a reminder, we repeat several points: 1) ensure that RAM2D1 and PRAM1 have the desired dimensions in all its subroutines; 2) ensure that the input data file NR--.DAT, contains the desired input parameters; 3) ensure that the JOB-file transfers the desired set of files.

If all appears well, submit the job as shown above. Monitor the job progress by reading the messages on the screen and/or inquire the status as described in section I.B.5. Job termination is indicated by the transfer of the CR--.CPR file from the CRAY to the VAX. Use EDT's 132 column screen editing mode to check the CPR-file for error-free execution of the whole job. In case of error messages, turn to PART V.C. or call Godehard Hilfer at (202)-767-2028.

In a series of simulations, it is unnecessary to recompile the source code for each simulation over again. Instead one can copy, or create, the XR--.JOB file and type: CRAY SUBMIT XR--.JOB

to submit the next simulation. The XR--.JOB file is a copy of the CR--.JOB that lacks the compilation and loading command lines. Hence, it will only run the executable dataset XR--. The corresponding CPR-file is XR--.CPR .

CHAPTER III

VIEWING THE RESULTS

PART III.A TERMINAL OUTPUT

The data file that arrives in the VAX user directory at the end of PRAM1's execution, PLT2.DAT, is a device independent graphics data file generated by the DISSPLA library routines contained in PRAM1. In order to see the graphs on the terminal screen, DISSPLA postprocessing software needs to be applied. For this purpose, unless previously done during this login,

type: GRAPHICS.LOGICALS

prompts: \$

type: PUBLIC.LOGICALS

prompts: \$

(to make use of site specific software and setups)

Then attach the data file to the post-processing software and run it

type: RUN VT240\$POP <

prompts: THIS IS THE VT240 POST-PROCESSOR ENTER YOUR POST-
PROCESSOR DIRECTIVES OR A CARRIAGE-RETURN FOR
DEFAULTS

To view all graphs one only needs to

type: <

a carriage return. This will produce the first graph on the screen. Another carriage return will erase the first graph and draw the second graph. Any more carriage returns will sequentially display the rest of the graphs until the last carriage return

prompts: END OF DISSPOP 2.2 -- 2057 VECTORS IN 1 PLOTS RUN ON 2/17/88
 USING SERIAL NUMBER 60 AT NRL PCC VAX PROPRIETARY
 SOFTWARE PRODUCT OF ISSCO, SAN DIEGO, CA
 \$

which automatically finishes the post-processing.

To be more selective in which graphs shall actually be displayed, one has to enter those graph numbers explicitly when asked for the post-processor directives. For example,

type: DRAW=5-9,12,17-20 <<

to display graphs numbered 5, 6, 7, 8, 9, 12, 17, 18, 19, 20. Notice that it takes two carriage returns to continue the postprocessing. If a few in a large series of graphs shall be excluded from viewing, one can, rather than listing all the others, 'delete' those particular graphs from the display. Hence,

type: DELE=1-4,10,11,13-16,21-END <<

to display the same graphs numbered 5, 6, 7, 8, 9, 12, 17, 18, 19, 20 as before. Note, deletes supersede draws, and the sequence of listing is immaterial.

For more details, see the DISSPLA users manual part F. DISSPOP post-processing.

PART III.B HARDCOPIES

Section III.B.1: Printed Graphs

The data file that arrives in the VAX user directory at the end of PRAM1's execution, PLT2.DAT, is a device independent graphics data file generated by the DISSPLA library routines contained in PRAM1. In order to obtain the graphs on paper, DISSPLA postprocessing software needs to be applied. For this purpose, unless previously done during this login,

type: GRAPHICS.LOGICALS

prompts: \$

type: PUBLIC.LOGICALS

prompts: \$

(to make use of site specific software and setups)

Then attach the data file to the post-processing software and run it.

type: RUN LNO1\$POP <

prompts: THIS IS THE VT240 POST-PROCESSOR ENTER YOUR POST-
 PROCESSOR DIRECTIVES OR A CARRIAGE-RETURN FOR
 DEFAULTS

To process all graphs for printing one only needs to

type: <

a carriage return. This will produce a new data file called INTSCRT.TMP in the user's directory which then can be printed straightforwardly. At the end of processing the computer

prompts: END OF DISSPOP 2.2 -- 2057 VECTORS IN 1 PLOTS RUN ON 2/17/88
 USING SERIAL NUMBER 60 AT NRL PCC VAX PROPRIETARY
 SOFTWARE PRODUCT OF ISSCO, SAN DIEGO, CA
 \$

which automatically finishes the post-processing.

To be more selective in which graphs shall actually be post-processed, one has to enter those graph numbers explicitly when asked for the post-processor directives. For example,

type: DRAW=5-9,12,17-20 <<

to graph frames numbered 5, 6, 7, 8, 9, 12, 17, 18, 19, 20. Notice that it takes *two* carriage returns to continue the postprocessing. One can also delete particular graphs,

type: DELE=1-4,10,11,13-16,21-END <<

to process the same graphs, numbered 5, 6, 7, 8, 9, 12, 17, 18, 19, 20, as before. The command DELEte supersedes DRAW and the sequence is immaterial. For more details, see the DISSPLA users manual part F. DISSPOP post-processing.

The device specific file INTSCRT2.TMP (last version is default version number) can be send directly to the CCF or A49 laser printer. Thence,

type: LASER/PLOT/CCF/NOTIFY INTSCRT2.TMP <

or

type: LASER/PLOT/A49/NOTIFY INTSCRT2.TMP <

The terminal will notify of the completion of printing with a beep and a message. The print-out can then be picked up in building A49 either at the CCF-desk (output from the CCF laser printer) or in the Remote-Print-Station room in building A49 (output from the A49 laser printer).

Section III.B.2: Printed ASCII-Files

The printing of regular text files is done with either one of the following two commands,

type: LASER/PORT/CCF/NOTIFY <

for a print-out on the CCF laser printer in building A49 or

type: LASER/PORT/CCF/NOTIFY <

for a print-out on the 'remote print station' laser printer in building A49. Both commands will cause the terminal to notify of the completion of the printing job with a beep and a message. For files with lines of more than 80 characters length, the printing can be turned by 90 degrees from the high format to the wide format of the 8.5×11 inch pages. For this

type: LASER/LAND/CCF/NOTIFY <

or

type: LASER/LAND/A49/NOTIFY <

CHAPTER IV

SUSTAINING OPERATIONS

PART IV.A FILE STORAGE

Section IV.A.1: VAX Disk

The most essential and only necessary storage device for the code operation is the VAX disk storage space. Since it is the default storage location, no special steps need to be taken to store files in the VAX computer on those disks. The essential files (source codes, input data files, batch job command files, and graphics output data files) are stored here. However, the space allocation for the user is limited and can be restrictive when producing graphs. To obtain a quotation of the allocated and used disk storage space,

type: `SHOW QUOTA <`

which will respond with a message that gives the total allocated storage space, the portion used, the portion remaining, and the overflow margin. If the allocated space is continuously insufficient, turn to the CCF system manager or consultant for an increase. If the shortage of storage space is expected to be only occasional, then turn also to the consultant for access to the so-called scratch disk. This whole disk is available on a first come, first serve basis. Files will be kept on it for at least 24 hours but at most 48 hours. Hence all post-processing and printing of particularly large graphics data files can be done before the system software wipes out all scratch disk files routinely.

The size of individual files can be obtained when listing the directory of files by specifying `/SIZE=USED` in the `DIRECTORY` command. Hence,

type: `DIR/SIZE=USED 'filename.ext' <`

Furthermore, the date the file was created can be inquired by specifying /DATE=CREATED in the DIRECTORY command. For this

type: DIR/DATE=USED 'filename.ext' <

This way, a more selective clean up of the user directory is possible, hopefully, maintaining sufficient space for all user activity.

VAX storage space is measured in units called BLOCKS,

1 Block = 512 Bytes.

(Recall 1 Byte = 8 Bits)

The price for VAX disk storage is currently \$0.00016 per Block per day.

Section IV.A.2: VAX Tape

For more economical storage and to keep the VAX disk quota sufficient it is recommended to store files on VAX tape. This is called archiving the file. For on-line documentation regarding the archiving options

type: HELP ARCHIVE <

The most important features will be listed here.

To archive a file means that that file is physically removed from the VAX disk to the tape. Hence, to keep a copy on the disk an explicit copy must be made,

type: COPY 'file-to-archive.ext' 'remaining-file.ext' <

To archive the desired file

type: ARCHIVE 'file-to-archive.ext' <

This removes the file from the directory.

To list the files that had previously been archived

type: ARCHIVE/DIR <

Since the archiving is done overnight by the operating system, the newly archived file, although gone from the directory, does not yet show up as archived. It is queued for archival. To list the files awaiting archival

type: ARCHIVE/LIST <

If an error occurred, the file can be retrieved from this queue of files bound for archival; For this

type: ARCHIVE/CANCEL 'file-to-archive.ext' <

To remove a file from the archive

type: REMOVE 'archived-file.ext' <

prompts: REMOVE DUA107: [USER.DIR]'archived-file.ext;x' ?

type: yes <

then the file disappears.

VAX storage space is measured in units called *blocks*,

1 *block* = 512 *bytes*.

(Recall

1 *byte* = 8 *bits*)

The price for VAX disk storage is currently \$0.00001 per block per day.

Section IV.A.3: CRAY Disk

As was mentioned in the introduction, all files on the CRAY computer are volatile. That is, they will not be stored by default, rather they will be destroyed by default. Therefore, it is necessary to save explicitly all files that need to be saved. To this end, after compilation of the source codes, the executable files are saved by the batch job command file on CRAY disk. Unless otherwise specified the system's storage (*save*) commands save the datasets on the CRAY disks. Examples for two procedures are:

SAVE, DN=XR--.

(example for batch job command line for storage),

CALL SAVE(IRRE, 'DN'L, DTFL1D, 'PDN'L, PDN1D)

(example for FORTRAN statement for file storage, where 'DN'L, DTFL1D indicates that the character variable DTFL1D is the dataset name when the program is running, and 'PDN'L, DTFL1D is the permanent dataset name by which the file will be listed on the disk.

Storage space on the CRAY is not allocated individually, but always on a first come, first serve basis. Operating system software ensures, by moving big, old files automatically from disk to tape, that there is always storage space available. When listing the directory file names by means of the AUDIT. command, the right hand column indicates whether the listed file is on disk (on-line) or on tape (off-line).

The price for CRAY file storage is the same as that for VAX file storage. Hence, CRAY disk storage is charged at \$0.00016 per *block* per day.

(Recall

1 *block* = 512 *bytes*.

1 *byte* = 8 *bits*)

The standard measure for CRAY storage is 1 *sector* = 8 Blocks = 512 CRAY words of 64 *bits* each)

Section IV.A.4: CRAY Tape

As was mentioned in the introduction and in section IV.A.3 above, all files on the CRAY computer are volatile. They will not be stored by default; they will be destroyed by default. Therefore, it is necessary to explicitly save all files that

need to be saved. To this end, the programs RAM2D1 (and PRAM1) contain CRAY operating system calls that save the data files on CRAY tape (called off-line). To save a file on CRAY tape, one has specify to that location on the SAVE command line. For example:

```
CALL SAVE(IRRE, 'DN'L,DTFL1D, 'PDN'L,PDN1D, 'RESIDE'L, 'OFFLINE'L)
```

This is a CRAY FORTRAN statement for file storage, where 'DN'L,DTFL1D indicates that the character variable DTFL1D holds the dataset name when the program is running, and 'PDN'L,DTFL1D contains the permanent dataset name by which the file will be listed on the disk. Residency off-line is explicitly mentioned.

The storage space on the CRAY tape is not allocated individually, but always sequentially used on first come first serve basis available. Operating system software ensures, by moving big old files automatically from disk to tape, that there is always storage space available. When listing the directory file names by means of the AUDIT. command, the right hand column indicates residency of the file on CRAY disk as on-line.

The price for CRAY file storage is the same as that for VAX file storage. Hence CRAY disk storage is charged at \$0.00016 per *block* per day.

(Recall 1 *block* = 512 *bytes*.
 1 *byte* = 8 *bits*)

The standard measure for CRAY storage is 1 *sector* = 8 *blocks*)

PART IV.B OPERATOR RELIEF

Section IV.B.1: Login Command File

Many settings and definitions should be repeated every time a user logs into the front-end VAX computers. To save the user the typing effort of these settings, it is possible and recommendable to let the computer repeat this sequence of definitions and commands automatically. This can be done by means of the LOGIN.COM file. This file, which has to be in the user's root (login default) directory, is a command file that the computer executes automatically every time the user's logs into the VAX computer. For details on the meaning and syntax of command lines in this file, see the VAX/VMS DCL-manual. The following list is an example for some of the login commands and definitions which typically appear in a LOGIN.COM-file.

COMMANDS

```
$ SET TERMINAL/VT100
```

informs the operating system of the terminal's industry standard

\$ GRAPHICS LOGICALS

invokes site-specific system definitions

\$ PUBLIC LOGICALS

invokes site-specific system definitions

\$ CRAY SET TERMINAL INFORM

advise operating system to output CRAY messages to terminal

\$ SHOW TIME

show current time on terminal

\$ SHOW QUOTA

show current VAX disk storage distribution of the owner.

DEFINITIONS

a) of acronyms of customized directory lists of file groups

\$ DIRALL ::= DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING

\$ DCPR ::= DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *.CPR

\$ DDAT ::= DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *.DAT

\$ DFOR ::= DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *.FOR

\$ DJOB ::= DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *.JOB

\$ ETA ::= DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *PLT*.DAT

\$ DMSG ::= DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *.MSG

\$ DTMP ::= DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *.TMP

b) of acronyms of customized directory lists of files of a standard dimensionality

\$ DEE ::= DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *EE.*

\$ DG1 ::= DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *G1.*

\$ DH1 ::= DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *H1.*

\$ DI1 ::= DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *I1.*

\$ DJ1 ::= DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *J1.*

\$ DK1 ::= DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *K1.*

\$ D1G ::= DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *1G.*

\$ D1H ::= DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *1H.*

\$ D1I ::= DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *1I.*

```

$ D1J ::= DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *1J.*
$ D1K ::= DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *1K.*
$ D1L ::= DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *1L.*
$ D1M ::= DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *1M.*
$ D1N ::= DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *1N.*
$ D1O ::= DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *1O.*
$ D1P ::= DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *1P.*
$ DGI ::= DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *GI.*

```

- c) of acronyms of other customized commands that have been explained elsewhere in this manual

```

$ H ::= SET DEFAULT DUA107:[HILFER]
$ HFOR ::= SET DEFAULT DUA107:[HILFER.FOR]
$ PLOTMPL ::= LASER/PLOT/CCF/NOTIFY INTSCRT2.TMP
$ PLOTMPS ::= LASER/PLOT/A49/NOTIFY INTSCRT2.TMP
$ POP240 ::= RUN VT240$POP
$ POPL ::= RUN LNO1$POP

```

Section IV.B.2: Edit-Aid

Certain customized features of the EDT editor can be made standard if the LOGIN.COM-file (see section IV.B.1) contains the following definition.

```
$ E ::= EDIT/EDT/COMMAND=DUA107:[USER]EDTINI.EDT
```

This shortens the command line that starts the EDT editor to the letter *e* plus the file-name and at the same time implements the definitions contained in the file [USER]:EDTINI.EDT for which an example follows:

```
DEFINE KEY GOLD N AS ``EXT SET SCREEN 80.``
```

(defines the two key strokes PF1 N to change the terminal display to 80 column width)

```
DEFINE KEY GOLD W AS ``EXT SET SCREEN 132.``
```

(defines the two key strokes PF1 W to change the terminal display to 132 column width)

```
SET SCREEN 72
```

(sets the display width to 72 columns)

SET WRAP 72

(sets the editor's feature of swapping terminal entries beyond column 72 into the next line)

SET MODE CHANGE

(causes the EDT editor to change to screen editing mode automatically at the onset of the editing session)

Section IV.B.3: VAX/CRAY Status

It is useful to define mnemonic acronyms for the monitoring functions of the VAX and CRAY computer systems. Some frequently used definitions from the LOGIN.COM-file of the author are:

CRAY:

\$ CSO ::= 'CRAY STATUS/OWN

(see subsection I.B.5.3 for details)

VAX:

\$ CNRL == ''MON CLU/INT=1''

(to monitor the work load distribution between the four front-end VAXes)

\$ MSYS == ''MONITOR SYSTEM''

(to monitor the CPU (computing), memory, and I/O (data input/output) load of the VAX in use)

\$ MTOP == ''MONITOR PROCESSES /TOPCPU''

(to monitor the list of VAX-processes with the highest CPU demand)

\$ SQ == ''SHOW QUEUE/DEVICE''

(to list the queue entries on all devices)

Section IV.B.4: CRAY Grease

There are ways of making the interaction with the CRAY computer more convenient and speedy. For example, the submission of batch jobs can be simplified by using the command CBATCH in combination of several mnemonic acronyms. CBATCH is a CCF command that allows one to eliminate the obligatory JOB and ACCOUNT command line from all job files by automatically attaching a file that contains those two command lines in an encrypted form. The file is named \$CRAY\$.ACCOUNT and resides in the user's root directory. It is automatically generated the first time CBATCH is used. For details on this command

type: HELP CBATCH <

Every batch job that is submitted to the CRAY computer is placed into a class of jobs that are similar in memory size and priority requirements. There are four classes with regard

to job-size: small = up to 511000 CRAY words
 medium = up to 1023000 CRAY words
 large = up to 1535000 CRAY words
 x-large = up to 3071000 CRAY words

and three subclasses of the four with regards to the requested
priority level: express charged 1.5,
 normal charged 1.0,
 deferred charged 0.7

times the price of \$900.00 per hour of CPU usage.

To submit a job quickly and conveniently while at the same time specifying these details, the following definitions may be found helpful when present in the LOGIN.COM-file:

```
$ CBDS ::= CBATCH/JUS=DEFER/MFL=511000/AC=-----  
$ CBDM ::= CBATCH/JUS=DEFER/MFL=1023000/AC=-----  
$ CBDL ::= CBATCH/JUS=DEFER/MFL=1535000/AC=-----  
$ CBDXL ::= CBATCH/JUS=DEFER/MFL=3071000/AC=-----  
$ CBNS ::= CBATCH/JUS=NORMAL/MFL=511000/AC=-----  
$ CBNM ::= CBATCH/JUS=NORMAL/MFL=1023000/AC=-----  
$ CBNL ::= CBATCH/JUS=NORMAL/MFL=1535000/AC=-----  
$ CBNXL ::= CBATCH/JUS=NORMAL/MFL=3071000/AC=-----  
$ CBXS ::= CBATCH/JUS=EXPRESS/MFL=511000/AC=-----  
$ CBXM ::= CBATCH/JUS=EXPRESS/MFL=1023000/AC=-----  
$ CBXL ::= CBATCH/JUS=EXPRESS/MFL=1535000/AC=-----  
$ CBXXL ::= CBATCH/JUS=EXPRESS/MFL=3071000/AC=-----
```

which have to be completed with the appropriate charge account number following the AC= parameter. With these acronyms, the submission of a batch job becomes quite simple. Since the CBATCH command expects a .JOB-file, the file extension (which is .JOB) can even be omitted. So, for example, to submit a small batch job that deletes a file from CRAY storage, one need only

type: CBDS CDELET <

or to run a particular full scale Raman interaction simulation

type: CBDXL CRGI <

These batch jobs are assumed to take less than 60 seconds of CPU time for completion. Should more CPU time be required, provide the /T=400 parameter to allow maximally 400 seconds of CPU time for execution. For example

CBDXL/T=400 CRGI

Note: be generous with the time limit to avoid having to rerun (and pay again) the whole job for lack of time allocation.

If the maximal memory requirement is known from the CPR-file of a previous run with the same dimensions, the right job class can be chosen. To choose the right class, one should consider also the system's limit of how many jobs of a certain class can run simultaneously. These are:

service class	resource class	max. jobs	priority
express	small	12	9
express	medium	6	9
express	large	2	9
express	xlarge	2	9
normal	small	10	6
normal	medium	4	6
normal	large	2	6
normal	xlarge	2	6
normal, long time	small	10	6
normal, long time	medium	4	6
normal, long time	large	2	6
normal, long time	xlarge	2	6
deferred	small	5	3
deferred	medium	2	3
deferred	large	2	3
deferred	xlarge	2	3

The meaning of the normal, long time class is subtle and should not concern the user, except that the job will be counted in the long time class if more than 300 seconds CPU time are requested. To find out how full the desired class currently is

type: CRAY <

following the standard VAX/VMS DCL prompt: \$. Then the screen

prompts: CRAY>

type: STATCLASS <

which will be responded with a table of the current job class demand. To scroll down

type: + <

To scroll up

type: - <

To exit the display

type: EXIT <

prompts: \$

Section IV.B.5: Money Savers

Some methods follow that will reduce the cost of computing. Generally, The most important money saver is the algorithm itself. To use the most efficient numerical scheme for obtaining the results of any computation is the key to low cost. Other methods usually provide only a fraction of the possible savings. Some of those methods are mentioned here.

The program RAM2D1 comes in two versions: RAM2D1C and RAM2D1D. They differ only in their memory requirements. In two-dimensional operation the three megaword random access memory (RAM) capacity of the CRAY X-MP machine often is exceeded. For simulations of this size one needs to use RAM2D1D. That version keeps only two work arrays in memory and stores intermediate results of the computation on CRAY disk memory at the expense of voluminous data input and output. The associated high I/O charges can be saved by using RAM2D1C whenever possible.

One can save 30 percent of the CPU-charges by running the job with priority=deferred rather than priority=normal (see section IV.B.4 for details). This change did not appear to alter the job turnaround time noticeably. Running a small job with priority=express is more expensive but also not very noticeable in terms of job turnaround time, since only the CPU processing is prioritized, not the file transfer.

Savings result also from the use of tape storage rather than disk, for long term file storage. These savings can be significant if the file is stored for a long length of time. The biggest savings are obtained if the file under consideration for storage can be discarded altogether rather than stored. This decision requires extreme prudence. Here, one can easily save pennies but waste dollars when having to regenerate the discarded dataset.

PART IV.C TROUBLE SHOOTING

For all sorts of invincible obstacles, the user will find ample support from the CCF consultants. They can be reached by phone: (202) 767-3542 or (202) 767-1374. One can

also send them a message over the VAX MAIL facility (type: HELP MAIL for details; when prompted for the recipient type: CONSULTANT). For all problems, especially regarding the codes RAM2D1 and PRAM1, the user may wish to call Dr. Godehard Hilfer at (202)-767-2028.

Problems that arise during the execution of the programs on the CRAY computer will be documented at the end of the CPR-file before the accounting section. For details on the error message, one may wish to read the description for the given error number in the CRAY operating system (COS) message manual.

Problems that arise from the use of the VAX are usually indicated by on-the-screen error messages that will indicate the nature of the problem.

If RAM2D1 compiles and runs without any apparent error, then one has no indication that the datafile is incorrect. If then PRAM1 also compiles and runs fine without any apparent error but fails to produce a graph, or produces some graphs as expected but others not at all or only in part, then one should first check the input data to PRAM1, especially the elements of the array CSEC. Another suspicion should be that the wrong dataset on the CRAY disk was used. Hence, one should check the existence of the desired datafile, the dimensions, date, and edition number of the file as specified in the input data file, the input namelist, and the program. If all looks well, one can rerun PRAM1, but requesting only one of those graphs that did not come out right previously to narrow the possible sources of error.

If PRAM1 ran without producing a PLT2.DAT file the CPR-file might report: SY001 - RLS COULD NOT FIND A DNT FOR META, which indicates that the META-file (=DIS-SPLA terminology for the device independent graphics file PLT2.DAT) was not created or was created, remained empty, and was as such discarded, and hence unavailable for transfer. It may also turn out that the file was held back on the CRAY since there was no room in the VAX directory. Confirm the latter by typing the command SHOW QUOTA and delete old files if necessary.

If the execution of the program seems to take an unexpected amount of time, it is likely to be due to the general overload of the computer system or the network rather than due to a problem with the codes. To inquire the computer system performance use the commands presented in section IV.B.3. In addition to those commands, one can check if the submitted process is being worked on which is reflected in an increase in CPU time used. The VAX CPU time can be monitored at any time by typing ^ T (=Ctrl T). Caution must be exercised that the T-key is hit and not, by accident, the

Y-key, which would terminate the execution. Analogously, the CRAY CPU time is listed when monitoring the status of one's own CRAY job by typing CRAY STATUS/OWN (CS0). Both, ^ T on the VAX and CS0 on the CRAY indicate what the computer is currently doing.

PART IV.D CRAY RUN SPECIFICATIONS

The following contains the vital statistics of RAM2D1C examples. The columns are numbered and contain the following information: (recall 1 CRAY word = 8 bytes = 64 bits)

1. encrypted dimensions
2. time dimension (NT)
3. transverse space dimension (NY)
4. maximum job size when executing (*mega-words*)
5. time executing in CPU for first z-step, 2 data drops (*seconds*)
6. time executing in CPU for 2000 z-steps, 2 data drops in 1-D, 21 data drops in 2-D (*seconds*)
7. CPU time required for compilation (*seconds*)
8. maximum job size when compiling (*mega-words*)
9. size of typical output data file with 2 data drops in 1-D, and 1 data set in 2-D (*mega-words*)

RAM2D1C

1.	2.	3.	4.	5.	6.	7.	8.	9.
IG	512	128	1.40	.730	799.75	2.23	1.46	.79
HH	256	256	1.40	.712	852.00	2.21	1.46	.79
HG	256	128	.74	.333	405.77	2.14	.81	.39
GG	128	128	.41	.158	201.67	2.01	.48	.20
KL	2048	1	.12	.017	12.08	2.06	.20	.037
JL	1024	1	.10	.009	6.11	2.07	.17	.024
IL	512	1	.10	.006	3.11	2.06	.16	.018
HL	256	1	.10	.004	1.63	2.06	.15	.015
1K	1	2048	.14	.062	28.63	2.11	.22	.061
1J	1	1024	.11	.030	14.93	2.17	.18	.037
1I	1	512	.10	.017	7.39	2.18	.16	.024
1H	1	256	.10	.010	3.60	2.15	.15	.018

The following contains the vital statistics of PRAM1CD examples. The columns are numbered and contain the following information:

(recall 1 CRAY word = 8 bytes = 64 bits)

1 VAX block = 512 bytes

1. encrypted dimensions
2. time dimension (NT)
3. transverse space dimension (NY)
4. maximum job size when executing (*mega-words*)
5. time executing in CPU for one graph (*seconds*)
6. time executing in CPU for 10 graphs (*seconds*)
7. CPU time required for compilation (*seconds*)
8. maximum job size when compiling (*mega-words*)
9. size of typical PLT2.DAT file with 1 graph (*blocks*)
10. size of typical PLT2.DAT file with 10 graphs (*blocks*)

PRAM1CD

1.	2.	3.	4.	5.	6.	7.	8.	9.	10.
IG	512	128	.60	1.917	11.33	5.917	.675	46	402
HH	256	256	.60	1.279	8.88	5.596	.672	32	288
HG	256	128	.40	.827	6.80	5.63	.46	37	288
GG	128	128	.29	.523	3.78	5.504	.36	26	206
K1	2048	1	.25	.355	3.08	9.20	.34	27	240
J1	1024	1	.22	.253	2.23	9.23	.30	20	177
I1	512	1	.20	.116	1.74	9.24	.28	16	147
H1	256	1	.19	.162	1.43	9.19	.27	13	130
1K	1	2048	.26	.345	2.70	8.699	.34	27	219
1J	1	1024	.22	.257	2.17	9.223	.30	19	164
1I	1	512	.20	.197	1.65	9.20	.28	15	133
1H	1	256	.19	.165	1.56	8.81	.27	13	120

APPENDICES

Appendix A

The appendices A-C present five examples of what the typical input to and output from RAM2D1 and PRAM1 looks like. The input data files, N---.DAT, must not contain any character in the first column of any line! (This is not visible in the examples shown.) All characters start in column 2 and/or the following columns. The input data are grouped in so called **namelists** (variables between two consecutive \$-signs. The character strings following the first \$-sign is the name of the namelist. The complete list of variables of each namelist is evident from the code listings. The possible values for these variables and the implications of these values are explained there in the commentary preceeding the routine (or subroutine) where the variable is used. For brevity's sake, four pages of the PLT2.DAT graphics output file are reproduced on a single page here. Even this reduction of volume was insufficient in Example B2. Hence, example B2 shows only a choice of the plots that result from the given input data.

APPENDIX A 1-D Transient Limit; Examples

Two examples are appended to show code operation in the transient limit. The illustration features the batch job command files, the input data files, the output CPR-files and the resulting output. The first example is a run that illustrates the basic use of the codes without complications or finess. The second example illustrates how several one-dimensional simulations can be done while running the programs only once.

EXAMPLE A1

XRJ1.JOB

AUDIT.
ACCESS, DN=XRJ1.
FETCH, DN=NRAM, TEXT='NRJ1.DAT'.
XRJ1.
DISPOSE, DN=ERRM, DF=BB, WAIT, TEXT='XRJ1.MSG.'.
AUDIT.
EXIT.

XPJ1.JOB

```
AUDIT.  
ACCESS, DN=XPJ1.  
ACCESS, DN=DISLIB, ID=DISSPLA, OWN=LIBRARY.  
ACCESS, DN=INTLIB, ID=DISSPLA, OWN=LIBRARY.  
ACCESS, DN=DVSD, ID=DISSPLA, OWN=LIBRARY.  
FETCH, DN=NPRAM1, TEXT='NPJ1.DAT'.  
XPJ1.  
DISPOSE, DN=META, DF=BB, WAIT, TEXT='PLT2.DAT'.  
DISPOSE, DN=EPRM, DF=BB, WAIT, TEXT='XPJ1.MSG.'.  
DISPOSE, DN=DISOUT, DF=BB, WAIT, TEXT='XPJ1.DSP.'.  
AUDIT.  
EXIT.  
DISPOSE, DN=EPRM, DF=BB, WAIT, TEXT='XPJ1.MSG.'.  
DISPOSE, DN=META, DF=BB, WAIT, TEXT='PLT2.DAT'.  
DISPOSE, DN=DISOUT, DF=BB, WAIT, TEXT='XPJ1.DSP.'.  
DUMPJOB.  
DEBUG, BLOCKS=GRAPHS.
```

```
AUDIT.  
ACCESS, DN=XPJ1.  
ACCESS, DN=DISLIB, ID=DISSPLA, OWN=LIBRARY.  
ACCESS, DN=INTLIB, ID=DISSPLA, OWN=LIBRARY.  
ACCESS, DN=DVSD, ID=DISSPLA, OWN=LIBRARY.  
FETCH, DN=NPRAM1, TEXT='NPJ1.DAT'.  
XPJ1.  
DISPOSE, DN=META, DF=BB, WAIT, TEXT='PLT2.DAT'.  
DISPOSE, DN=EPRM, DF=BB, WAIT, TEXT='XPJ1.MSG.'.  
DISPOSE, DN=DISOUT, DF=BB, WAIT, TEXT='XPJ1.DSP.'.  
AUDIT.  
EXIT.  
DISPOSE, DN=EPRM, DF=BB, WAIT, TEXT='XPJ1.MSG.'.  
DISPOSE, DN=META, DF=BB, WAIT, TEXT='PLT2.DAT'.  
DISPOSE, DN=DISOUT, DF=BB, WAIT, TEXT='XPJ1.DSP.'.  
DUMPJOB.  
DEBUG, BLOCKS=GRAPHS.
```

NRJ1.DAT

```

$NAML
  RINT(1)=1.0,
  RIST=1.0E-8,
  ICOND=3,
  ZFINAL=100.0,
  ZKEEP=50.0,
$
  NAMLIST/NAML/NPUMP, YM, TM, ZINT, RKP, RKS, YOFF, TOFF, YWIDTH, TWIDTH,
  1 YOST, TOST, YWST, TWST, RINT, RIST, RAMASM, RALASM, NHYP, PHL, PHST, TOC,
  2 ITYPE, RTYPE, RABAMP, RDSLIM, ICOND, ZSTEP, ZFINAL, ZKEEP, NMAX, TTWO, GAIN

```

NPJ1.DAT

```

$FLDATE
  DONYET=1,
  MONTH=04,
  DAY=18,
  YEAR=88,
  IPART=1,
  NEDN=1,
$
$CONDAT
  LPRMT(1)=1,
  LPRMT(2)=1,
  LPRMT(3)=1,
  LPRMT(4)=1,
  NSEC=1,
  CSEC(1,1)=(1.0,2.0),
  CSEC(2,1)=(1.0,2.0),
  CSEC(3,1)=(1.0,2.0),
  CSEC(7,1)=(1.0,2.0),
  CSEC(8,1)=(1.0,2.0),
  CSEC(9,1)=(1.0,2.0),
  CSEC(13,1)=(1.0,2.0),
  CSEC(14,1)=(1.0,2.0),
  CSEC(15,1)=(1.0,2.0),
$
$ZPLOT
  KZ(1)=1,
  KZ(2)=2,
  KZ(3)=3,
$
$RMPLT
$

```

XRJ1.CPR

```

09:02:06 0512 0 0000 CSP
09:02:06 0515 0 0000 CSP
09:02:06 0518 0 0000 CSP
09:02:06 0521 0 0000 CSP
09:02:06 0523 0 0001 CSP
09:02:06 0526 0 0001 CSP
09:02:06 0529 0 0001 CSP
09:02:06 0531 0 0001 CSP
09:02:06 0534 0 0001 CSP
09:02:06 0537 0 0001 CSP
09:02:06 0539 0 0001 CSP
09:02:06 0542 0 0002 CSP
09:02:06 0544 0 0002 CSP
09:02:06 0546 0 0002 CSP
09:02:06 0548 0 0002 CSP
09:02:06 0570 0 0002 CSP
09:02:06 0573 0 0002 CSP
09:02:06 0575 0 0002 CSP
09:02:06 0585 0 0002 CSP
09:02:06 0542 0 0012 CSP
09:02:07 0422 0 1095 USER
09:02:07 0360 0 1126 USER
09:02:22 0940 0 3453 USER
09:02:22 0944 0 3454 USER
09:02:22 0949 0 3455 USER
09:02:22 7022 0 3457 CSP
09:02:22 9635 0 3457 PDM
09:02:22 9637 0 3457 PDM
09:02:22 9652 0 3458 CSP
09:02:28 3559 0 3459 SCP
09:02:28 3562 0 3459 SCP
09:02:28 3564 0 3459 SCP
09:02:30 9535 0 3459 SCP
09:02:31 2485 0 3461 CSP
09:02:46 8580 13 0298 PDM
09:02:46 8582 13 0298 PDM
09:02:46 8683 13 0298 USER
09:02:46 8718 13 0299 CSP
09:02:53 8929 13 0301 SCP
09:02:53 8932 13 0301 SCP
09:02:53 8935 13 0301 SCP
09:02:57 5904 13 0305 USER
09:03:08 8528 13 2640 USER
09:03:08 8532 13 2641 USER
09:03:08 8536 13 2642 USER
09:03:08 8599 13 2642 CSP
09:03:08 8617 13 2643 CSP
09:03:08 8619 13 2643 CSP
09:03:08 8622 13 2643 CSP
09:03:08 8352 13 2644 USER
09:03:08 8355 13 2644 USER
09:03:08 8359 13 2644 USER
09:03:08 8362 13 2644 USER
09:03:08 8367 13 2645 USER
09:03:08 8370 13 2645 USER
09:03:08 8373 13 2645 USER
09:03:08 8376 13 2645 USER
09:03:08 8379 13 2645 USER
09:03:08 8383 13 2645 USER
09:03:08 8386 13 2646 USER
09:03:08 8389 13 2646 USER

```

```

.....
WELCOME TO THE NRL CRAY XMP
.....
The CRAY will be unavailable Sunday April 24 from 8:00 A.M. to 4:00 P.M.
for software testing
.....
There will be no CRAY off-line data set recalls on Tuesday or Wednesday
mornings between 2:00 AM and 7:00 AM in order for us to perform CLEANUP
runs on our CRAY archive tape library.
.....
CRAY X-MP SERIAL-415 65      NAVAL RESEARCH LABORATORY      04 21 88
CRAY OPERATING SYSTEM      COS 1.15  ASSEMBLY DATE 01 04 88

JOB JN-XRJ1.MFL-511000.US-DEFER.
ACCOUNT.AC-US-UPW-APW-
AC213  ' ' TOTAL BUDGET WARNING LEVEL REACHED FOR THIS ACCOUNT NUMBER
AUDIT
AU003 -      213 DATASETS.      226201 BLOCKS.      115746098 WORDS
AU003 -      63 DATASETS.      46310 BLOCKS.      23694963 WORDS ONLINE
AU003 -      150 DATASETS.      179891 BLOCKS.      92051135 WORDS OFFLINE
ACCESS. DN-XRJ1
PD000 - PDN - XRJ1      ID -      ED -      5 OWN - HILFER
PD000 - ACCESS COMPLETE
FETCH. DN-NRAM.TEXT- 'NRJ1.DAT'
VAX TO CRAY: $SYSTEM-S-NORMAL. normal successful completion
VAX TO CRAY: FILE-$1$DUAL07:[HILFER.FR2]NRJ1.DAT:21
VAX TO CRAY: 416 BYTES TRANSFERRED
SS004 - DATASET RECEIVED FROM FRONT END
XRJ1
PD000 - PDN - FK1042188      ID -      ED -      1 OWN - HILFER
PD000 - SAVE COMPLETE
UT003 - EXIT CALLED BY RAH2D1C
DISPOSE. DN-ERRM.DF-BB.WAIT.TEXT- 'XRJ1 MSG.'
CRAY TO VAX: $RMS-S-NORMAL. normal successful completion
CRAY TO VAX: FILE-$1$DUAL07:[HILFER.FR2]XRJ1.MSG:1
CRAY TO VAX: 20 BYTES TRANSFERRED
AUDIT
AU003 -      214 DATASETS.      226297 BLOCKS.      115795201 WORDS
AU003 -      84 DATASETS.      46406 BLOCKS.      23744066 WORDS ONLINE
AU003 -      150 DATASETS.      179891 BLOCKS.      92051135 WORDS OFFLINE
EXIT
END OF JOB

JOB NAME -      XRJ1
USER NUMBER -      HILFER
JOB SEQUENCE NUMBER -      40324

TIME EXECUTING IN CPU -      0000:00:13.2644
TIME WAITING TO EXECUTE -      0000:00:26.4189
TIME WAITING FOR I O -      0000:00:22.0685
TIME WAITING IN INPUT QUEUE -      0000:00:00.2267
MEMORY ' CPU TIME (MWDS*SEC) -      1.62992
MEMORY ' I O WAIT TIME (MWDS*SEC) -      2.17570
MINIMUM JOB SIZE (WORDS) -      44544
MAXIMUM JOB SIZE (WORDS) -      124416

```

XRJ1.CPR

09:03:08 8392	13 2646	USER	MINIMUM FL (WORDS)	40960
09:03:08 8395	13 2646	USER	MAXIMUM FL (WORDS) -	119808
09:03:08 8398	13 2646	USER	MINIMUM JTA (WORDS) -	3584
09:03:08 8401	13 2646	USER	MAXIMUM JTA (WORDS) -	4608
09:03:08 8405	13 2646	USER	DISK SECTORS MOVED -	2302
09:03:08 8408	13 2646	USER	FSS SECTORS MOVED -	0
09:03:08 8411	13 2646	USER	USER I O REQUESTS -	1397
09:03:08 8414	13 2646	USER	USER I O SUSPENSIONS -	1544
09:03:08 8417	13 2646	USER	OPEN CALLS -	27
09:03:08 8421	13 2647	USER	CLOSE CALLS -	28
09:03:08 8424	13 2647	USER	MEMORY RESIDENT DATASETS -	0
09:03:08 8427	13 2647	USER	TEMPORARY DATASET SECTORS USED -	1
09:03:08 8430	13 2647	USER	PERMANENT DATASET SECTORS ACCESSED -	1600
09:03:08 8434	13 2647	USER	PERMANENT DATASET SECTORS SAVED -	96
09:03:08 8437	13 2647	USER	SECTORS RECEIVED FROM FRONT END -	1
09:03:08 8440	13 2647	USER	SECTORS QUEUED TO FRONT END -	1
09:03:09 1518	13 2724	USER		
09:03:09 1520	13 2724	USER		
09:03:09 1524	13 2725	USER		
09:03:09 1527	13 2725	USER		
09:03:09 1531	13 2726	USER		
09:03:09 1534	13 2727	USER		
09:03:09 1578	13 2728	USER		
09:03:09 1582	13 2729	USER		
09:03:09 1585	13 2730	USER		
09:03:09 1589	13 2732	USER		
09:03:09 1593	13 2733	USER		
09:03:09 1597	13 2734	USER		
09:03:09 1600	13 2735	USER		
09:03:09 1604	13 2736	USER		
09:03:09 1606	13 2736	USER		
09:03:09 1609	13 2736	USER		

*** COST TABLE FOR THIS JOB ***									
JOBNAME -----				XRJ1					
USER IDENT.-----				HILFER					
BEGAN EXECUTION ----				THU APR 21, 1988		09:02:05		HOURS	
AT A PRIORITY OF --						3			
AND JOB CLASS OF --						DSMALL			
13 271129	SECONDS OF CPU TIME	@ \$ 630.00	HR	--	\$	2	32		
1 630306	MEMORY CPU (MWRD-SEC)	@ \$ 84.00	HR	--	\$	0	04		
2 177428	MEMORY I O (MWRD-SEC)	@ \$ 84.00	HR	--	\$	0	05		
0 002303	I O MEGASECTORS MOVED	@ \$ 84.00	EA	--	\$	0	19		
0 000000	TAPE MOUNT(S)	@ \$ 5.00	EA	--	\$	0	00		
**** TOTAL COST FOR THIS JOB ****						--	\$	2.60	

XPJ1.CPR

```

09:03:27 5941 0 0000 CSP
09:03:27 5944 0 0000 CSP
09:03:27 5947 0 0000 CSP
09:03:27 5950 0 0001 CSP
09:03:27 5953 0 0001 CSP
09:03:27 5955 0 0001 CSP
09:03:27 5958 0 0001 CSP
09:03:27 5961 0 0001 CSP
09:03:27 5963 0 0001 CSP
09:03:27 5966 0 0001 CSP
09:03:27 5969 0 0001 CSP
09:03:27 5972 0 0002 CSP
09:03:27 5983 0 0002 CSP
09:03:27 5996 0 0002 CSP
09:03:27 5999 0 0002 CSP
09:03:27 6002 0 0002 CSP
09:03:27 6004 0 0002 CSP
09:03:27 6122 0 0002 CSP
09:03:27 6476 0 0014 CSP
09:03:28 7831 0 1099 USER
09:03:29 0537 0 1131 USER
09:03:40 1793 0 3464 USER
09:03:40 1797 0 3465 USER
09:03:40 1801 0 3466 USER
09:03:40 1875 0 3468 CSP
09:03:40 4632 0 3468 PDM
09:03:40 4634 0 3468 PDM
09:03:40 4652 0 3472 CSP
09:03:40 7388 0 3472 PDM
09:03:40 7390 0 3472 PDM
09:03:40 7408 0 3476 CSP
09:03:40 9784 0 3476 PDM
09:03:40 9787 0 3476 PDM
09:03:40 9805 0 3479 CSP
09:03:41 2152 0 3480 PDM
09:03:41 2154 0 3480 PDM
09:03:41 2170 0 3480 CSP
09:03:43 0104 0 3482 SCP
09:03:43 0107 0 3482 SCP
09:03:43 0111 0 3482 SCP
09:03:47 1131 0 3482 SCP
09:03:47 3965 0 3483 CSP
09:03:47 9514 0 3516 PDM
09:03:47 9516 0 3516 PDM
09:04:02 1912 13 5244 USER
09:04:02 1938 13 5244 CSP
09:04:17 8375 13 5247 SCP
09:04:17 8378 13 5247 SCP
09:04:17 8381 13 5247 SCP
09:04:24 4376 13 5247 CSP
09:04:29 4986 13 5249 SCP
09:04:29 4988 13 5249 SCP
09:04:29 4991 13 5249 SCP
09:04:33 7466 13 5250 CSP
09:04:38 5871 13 5252 SCP
09:04:38 5874 13 5252 SCP
09:04:38 5877 13 5252 SCP
09:04:43 1980 13 5256 USER
09:04:54 3707 13 7600 USER
09:04:54 3711 13 7601 USER

```

```

.....
WELCOME TO THE NRL CRAY XMP
.....
The CRAY will be unavailable Sunday April 24 from 8:00 A.M. to 4:00 P.M.
for software testing.
.....
There will be no CRAY off-line data set recalls on Tuesday or Wednesday
mornings between 2:00 AM and 7:00 AM in order for us to perform CLEANUP
runs on our CRAY archive tape library.
.....
CRAY X-MP SERIAL-415.65    NAVAL RESEARCH LABORATORY    04 21 88
CRAY OPERATING SYSTEM    COS 1.15    ASSEMBLY DATE 01 04 88

JOB JN-XPJ1.MFL-511000.US-DEFER.
ACCOUNT.AC-US-UPW-APW-
AC213 - ** TOTAL BUDGET WARNING LEVEL REACHED FOR THIS ACCOUNT NUMBER
AUDIT.
AU003 -      214 DATASETS.    226297 BLOCKS.    115795201 WORDS
AU003 -      64 DATASETS.    46406 BLOCKS.    23744066 WORDS ONLINE
AU003 -     150 DATASETS.    179891 BLOCKS.    92051135 WORDS OFFLINE
ACCESS. DN-XPJ1
PD000 - PDM - XPJ1          ID -          ED -   39 OWN - HILFER
PD000 - ACCESS COMPLETE
ACCESS. DN-DISLIB.ID-DISSPLA.OWN-LIBRARY
PD000 - PDM - DISLIB      ID - DISSPLA    ED -    1 OWN - LIBRARY
PD000 - ACCESS COMPLETE
ACCESS. DN-INTLIB.ID-DISSPLA.OWN-LIBRARY
PD000 - PDM - INTLIB      ID - DISSPLA    ED -    1 OWN - LIBRARY
PD000 - ACCESS COMPLETE
ACCESS. DN-DVSD.ID-DISSPLA.OWN-LIBRARY
PD000 - PDM - DVSD        ID - DISSPLA    ED -    1 OWN - LIBRARY
PD000 - ACCESS COMPLETE
FETCH. DN-NPRAM1.TEXT- NPFJ1.DAT
VAX TO CRAY: %SYSTEM-S-NORMAL. normal successful completion
VAX TO CRAY: FILE-$1SDUA107:[HILFER.FR2]NPFJ1.DAT:39
VAX TO CRAY: 768 BYTES TRANSFERRED
SS004 - DATASET RECEIVED FROM FRONT END
XPJ1.
PD000 - PDM - FK1042188      ID -          ED -    1 OWN - HILFER
PD000 - ACCESS COMPLETE
UT003 - EXIT CALLED BY PRAM1CD
DISPOSE. DN-META.DF-BB.WAIT.TEXT-PLT2.DAT
CRAY TO VAX: %RMS-S-NORMAL. normal successful completion
CRAY TO VAX: FILE-$1SDUA107:[HILFER.FR2]PLT2.DAT:2
CRAY TO VAX: 496240 BYTES TRANSFERRED
DISPOSE. DN-EPRM.DF-BB.WAIT.TEXT- XPJ1.MSG
CRAY TO VAX: %RMS-S-NORMAL. normal successful completion
CRAY TO VAX: FILE-$1SDUA107:[HILFER.FR2]XPJ1.MSG:1
CRAY TO VAX: 3101 BYTES TRANSFERRED
DISPOSE. DN-DISOUT.DF-BB.WAIT.TEXT-XPJ1.DSP
CRAY TO VAX: %RMS-S-NORMAL. normal successful completion
CRAY TO VAX: FILE-$1SDUA107:[HILFER.FR2]XPJ1.DSP:1
CRAY TO VAX: 688 BYTES TRANSFERRED
AUDIT.
AU003 -      214 DATASETS.    226297 BLOCKS.    115795201 WORDS
AU003 -      64 DATASETS.    46406 BLOCKS.    23744066 WORDS ONLINE

```


XPJ1.CPR

09:04:54 3716 13 7603 USER AV003 - 150 DATASETS. 179891 BLOCKS. 92051135 WORDS OFFLINE
 09:04:54 3792 13 7603 CSP EXIT.
 09:04:54 3808 13 7603 CSP END OF JOB
 09:04:54 3809 13 7603 CSP
 09:04:54 3811 13 7603 USER
 09:04:54 5244 13 7605 USER
 09:04:54 5247 13 7605 USER
 09:04:54 5250 13 7605 USER
 09:04:54 5253 13 7605 USER
 09:04:54 5257 13 7605 USER
 09:04:54 5260 13 7605 USER
 09:04:54 5263 13 7606 USER
 09:04:54 5268 13 7606 USER
 09:04:54 5272 13 7606 USER
 09:04:54 5275 13 7606 USER
 09:04:54 5278 13 7606 USER
 09:04:54 5282 13 7606 USER
 09:04:54 5285 13 7606 USER
 09:04:54 5288 13 7607 USER
 09:04:54 5291 13 7607 USER
 09:04:54 5294 13 7607 USER
 09:04:54 5298 13 7607 USER
 09:04:54 5301 13 7607 USER
 09:04:54 5373 13 7607 USER
 09:04:54 5376 13 7607 USER
 09:04:54 5379 13 7607 USER
 09:04:54 5382 13 7607 USER
 09:04:54 5385 13 7607 USER
 09:04:54 5389 13 7607 USER
 09:04:54 5392 13 7607 USER
 09:04:54 5395 13 7608 USER
 09:04:54 5398 13 7608 USER
 09:04:54 5401 13 7608 USER
 09:04:54 5573 13 7685 USER
 09:04:54 5577 13 7685 USER
 09:04:54 5581 13 7686 USER
 09:04:54 5584 13 7686 USER
 09:04:54 5588 13 7687 USER
 09:04:54 5591 13 7688 USER
 09:04:54 5593 13 7689 USER
 09:04:54 5598 13 7690 USER
 09:04:54 5602 13 7691 USER
 09:04:54 5606 13 7693 USER
 09:04:54 5610 13 7694 USER
 09:04:54 5613 13 7695 USER
 09:04:54 5617 13 7696 USER
 09:04:54 5621 13 7697 USER
 09:04:54 5623 13 7697 USER
 09:04:54 5626 13 7697 USER

JOB NAME -
 USER NUMBER -
 JOB SEQUENCE NUMBER -

TIME EXECUTING IN CPU -
 TIME WAITING TO EXECUTE -
 TIME WAITING FOR I O -
 TIME WAITING IN INPUT QUEUE -
 MEMORY CPU TIME (MWDS*SEC) -
 MEMORY I O WAIT TIME (MWDS*SEC) -
 MINIMUM JOB SIZE (WORDS) -
 MAXIMUM JOB SIZE (WORDS) -
 MINIMUM FL (WORDS) -
 MAXIMUM FL (WORDS) -
 MINIMUM JTA (WORDS) -
 MAXIMUM JTA (WORDS) -
 DISK SECTORS MOVED -
 FSS SECTORS MOVED -
 USER I O REQUESTS -
 USER I O SUSPENSIONS -
 OPEN CALLS -
 CLOSE CALLS -
 MEMORY RESIDENT DATASETS -
 TEMPORARY DATASET SECTORS USED -
 PERMANENT DATASET SECTORS ACCESSED -
 PERMANENT DATASET SECTORS SAVED -
 SECTORS RECEIVED FROM FRONT END -
 SECTORS QUEUED TO FRONT END -

0000:00:13.7605
 0000:00:49.1493
 0000:00:22.9771
 0000:00:00.5499
 3.39919
 2.47661
 44544
 253952
 40940
 249344
 3584
 5120
 3072
 0
 1464
 1714
 31
 31
 0
 127
 2821
 0
 1
 126

.....
 COST TABLE FOR THIS JOB
 JOBNAME
 USER IDENT
 BEGAN EXECUTION
 AT A PRIORITY OF
 AND JOB CLASS OF
 SECONDS OF CPU TIME
 MEMORY CPU (MWDS*SEC)
 MEMORY I O (MWDS*SEC)
 I O MEGASECTORS MOVED
 TAPE MOUNT(S)

13 767250
 3.399409
 2.479123
 0.003074
 0.000000

THU APR 21, 1988
 @ \$ 630.00 HR
 @ \$ 84.00 HR
 @ \$ 84.00 HR
 @ \$ 84.00 EA
 @ \$ 5.00 EA

XPJ1
 HILFER
 09:03:27 HOURS.
 3
 DSHALL
 \$ 2.41
 \$ 0.08
 \$ 0.06
 \$ 0.26
 \$ 0.00
 \$ 2.81

.....
 TOTAL COST FOR THIS JOB

PLT2.DAT (Example A1)

LIST OF INPUT PARAMETERS

ICOND	-	3
MMAX	-	1000
NPLUR	-	2
NT	-	1024
NY	-	1
GAIN	-	3.0000
PMST	-	0.0000
AQLASH	-	5.0000
RAMASH	-	1.5000
RIST	-	1.00-10 ⁺
RKP	-	1.18-10 ⁺
RKS	-	9.19-10 ⁺
TGC	-	5.0000
TOST	-	-40.000
TTMO	-	633.00
TWST	-	-40.000
ZFINAL	-	100.00
ZKEEP	-	50.300
ZSTEP	-	0.0500

LIST OF INPUT PARAMETERS CONC.

TYPE	1	2	3	4	5
PHL 1-101	0.0000	0.0000	0.0000	0.0000	0.0000
	0.0000	0.0000	0.0000	0.0000	0.0000
RINT11-101	1.0000	0.5500	0.5500	0.5500	0.5500
	0.5500	0.5500	0.5500	0.5500	0.5500
RTYPE	2.0000	2.0000	2.0000	2.0000	2.0000
	2.0000	2.0000	2.0000		
TH11,21	-100.00	100.00			
TOFF11-101	0.0000	0.0000	0.0000	0.0000	0.0000
	0.0000	0.0000	0.0000	0.0000	0.0000
TWIDTH	40.000	40.000	40.000	40.000	40.000
	40.000	40.000	40.000	40.000	40.000

LIST OF INPUT PARAMETERS (CONTD.)

[illegible]

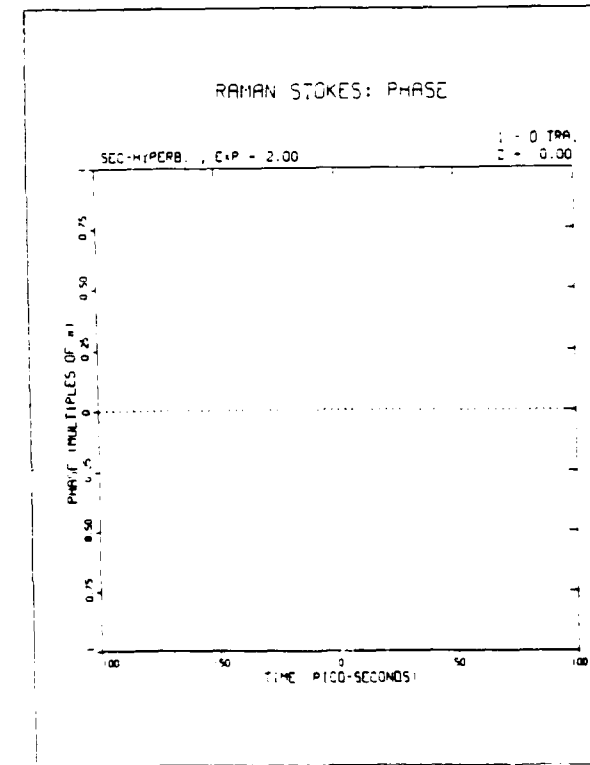
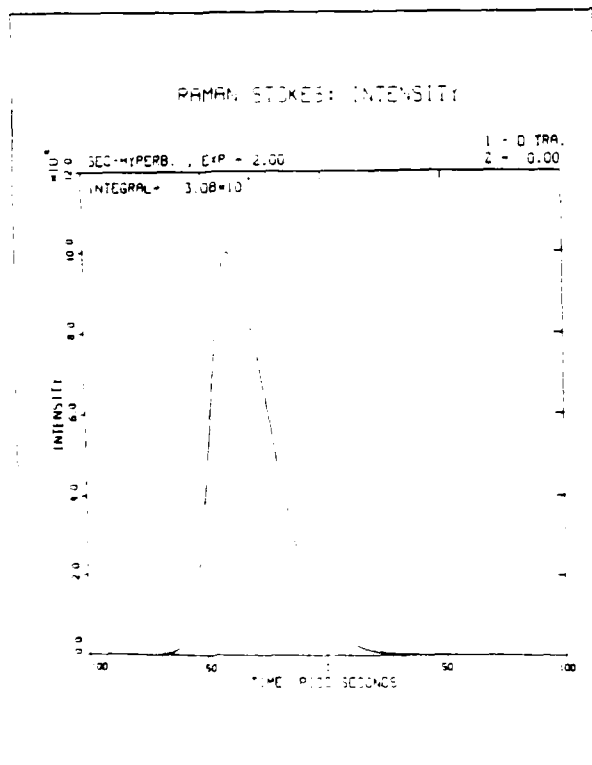
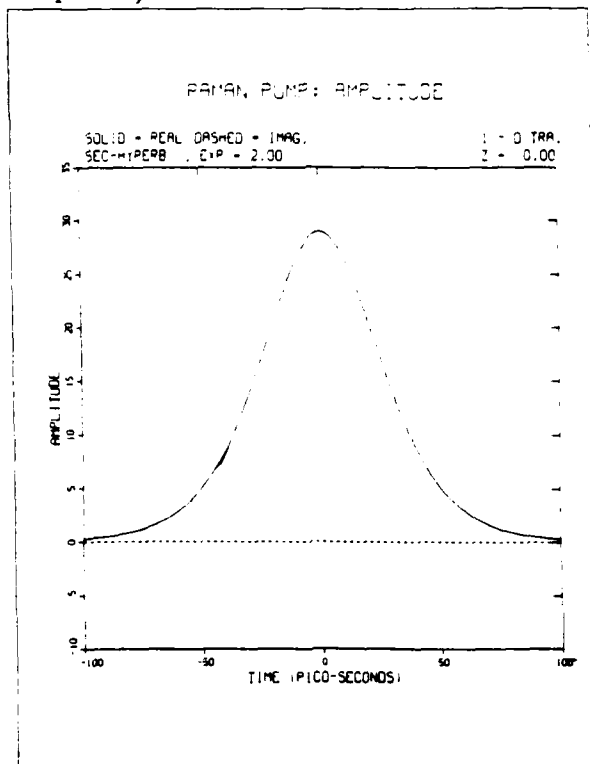
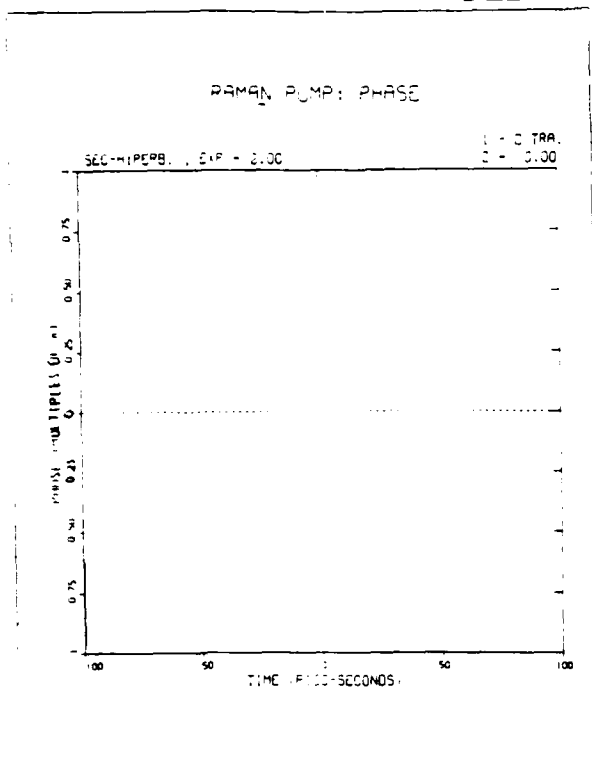
PAMAN PUMP: INTENSITY

SEC-HYPERB., EXP - 2.00
INTEGRAL = 44.0142
1.0000

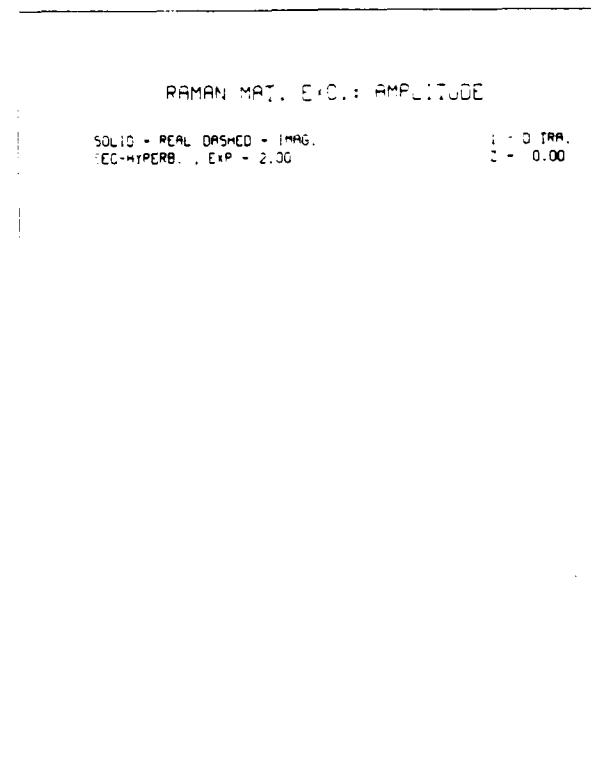
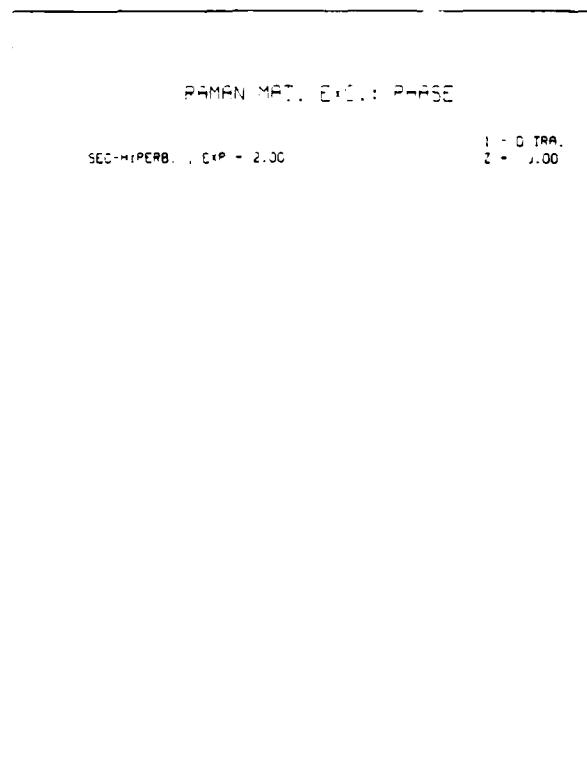
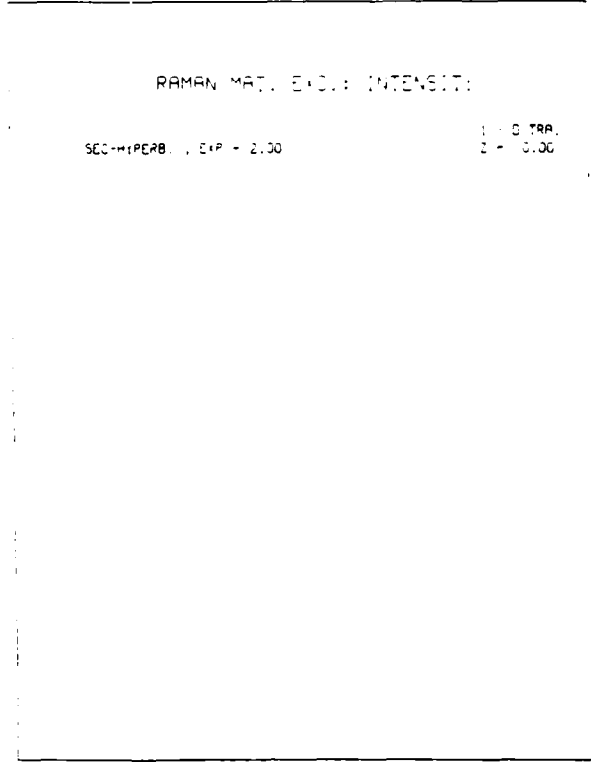
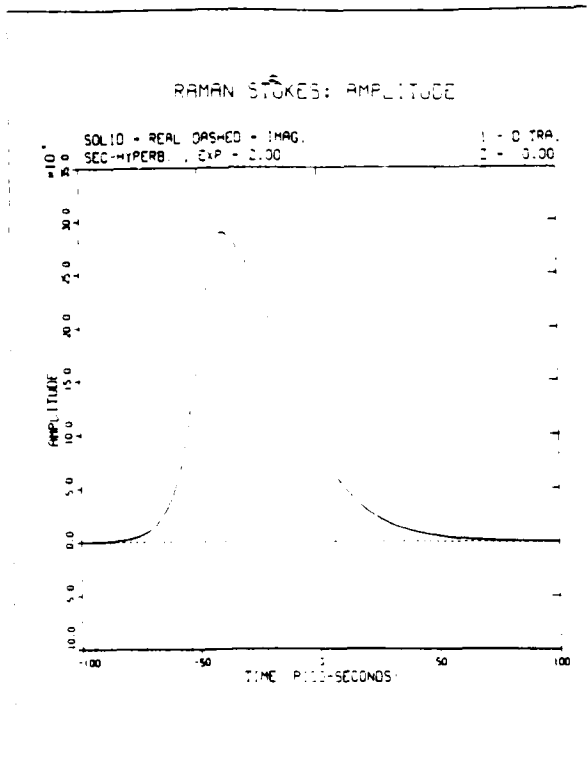
INTENSITY

TIME (PICO-SECONDS)

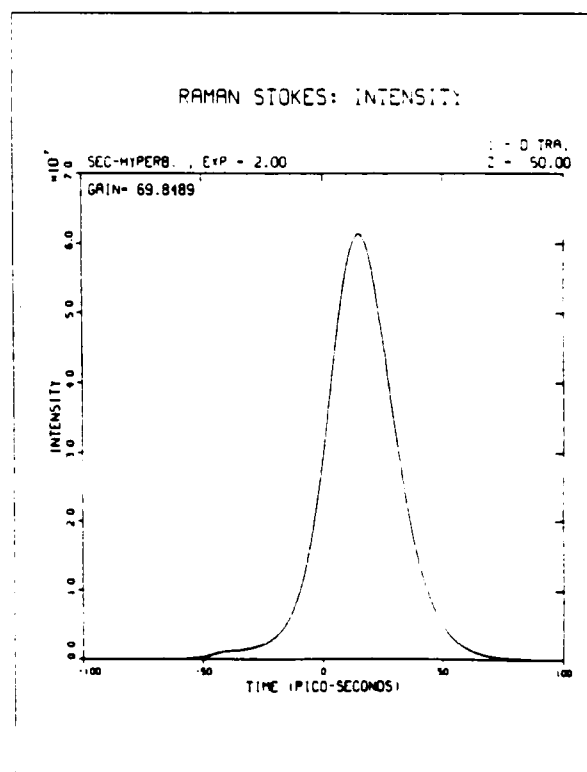
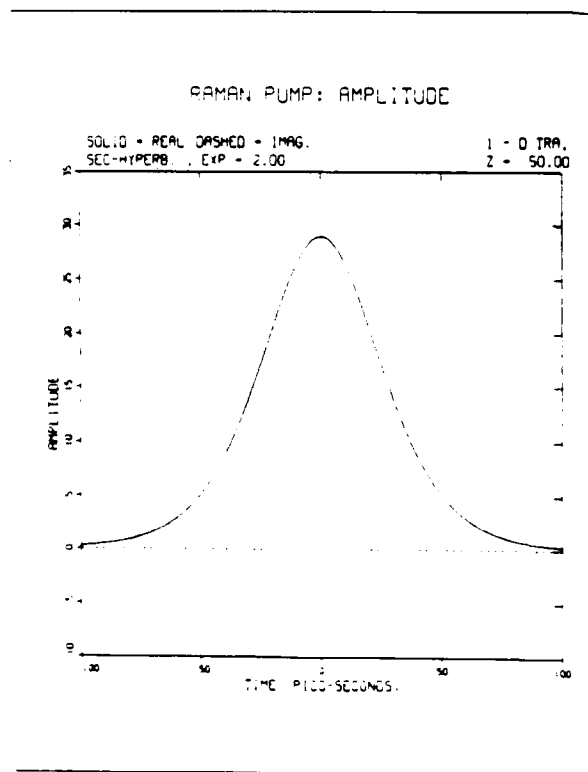
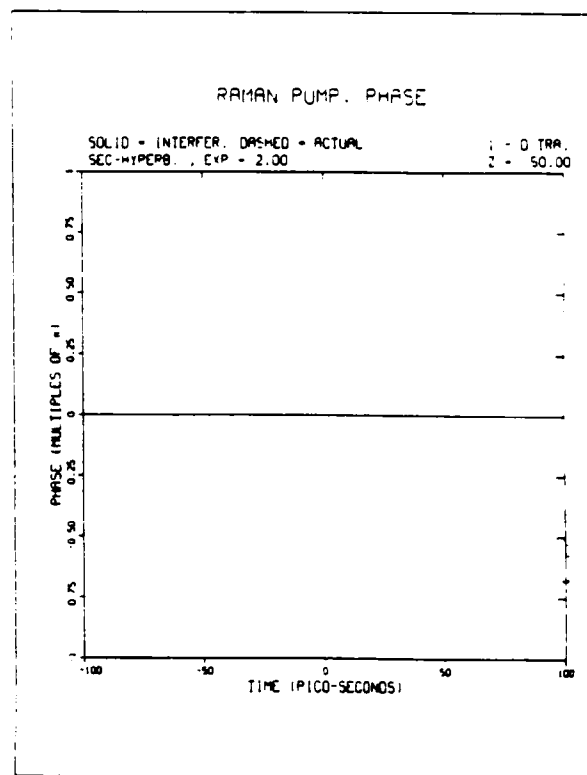
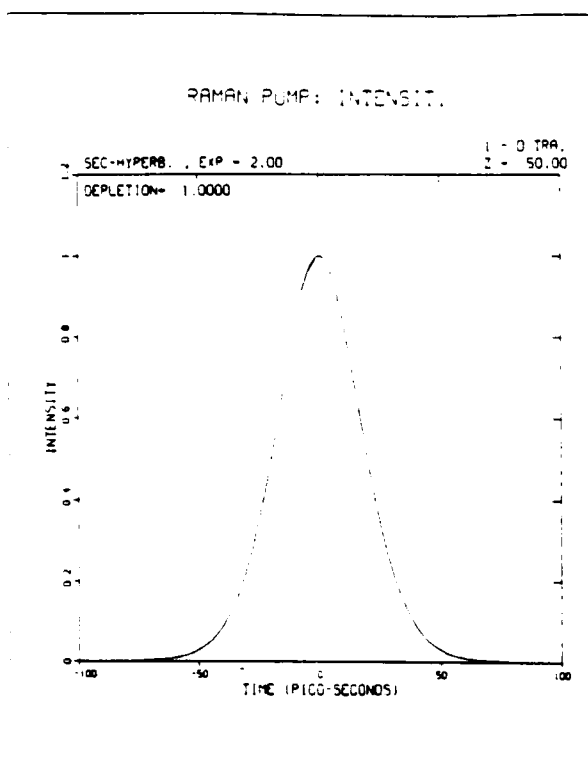
PLT2.DAT (Example A1)



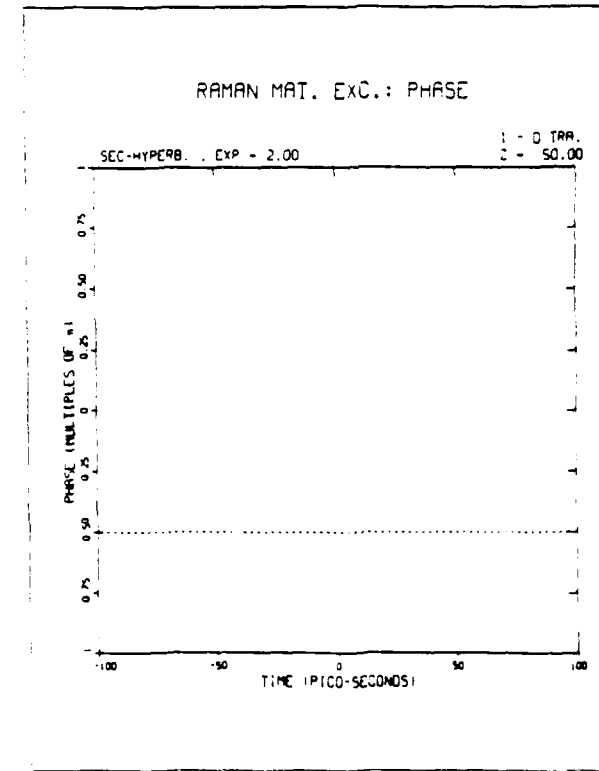
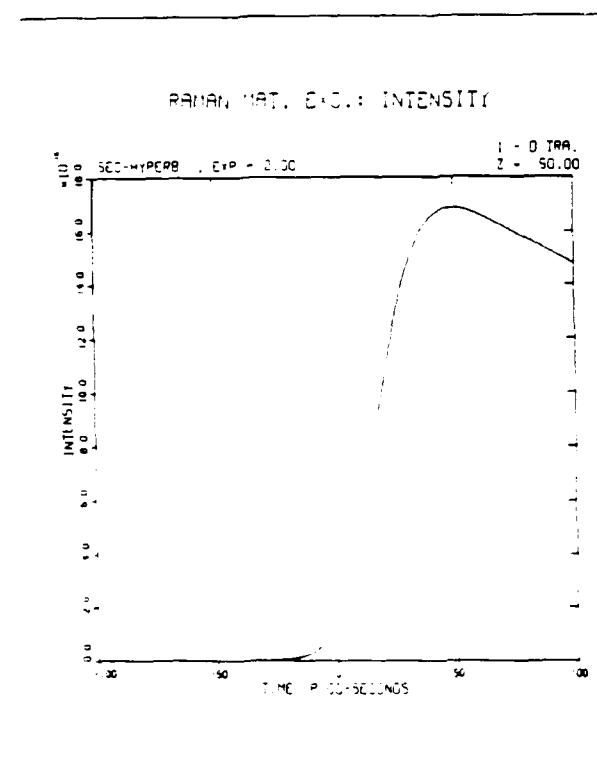
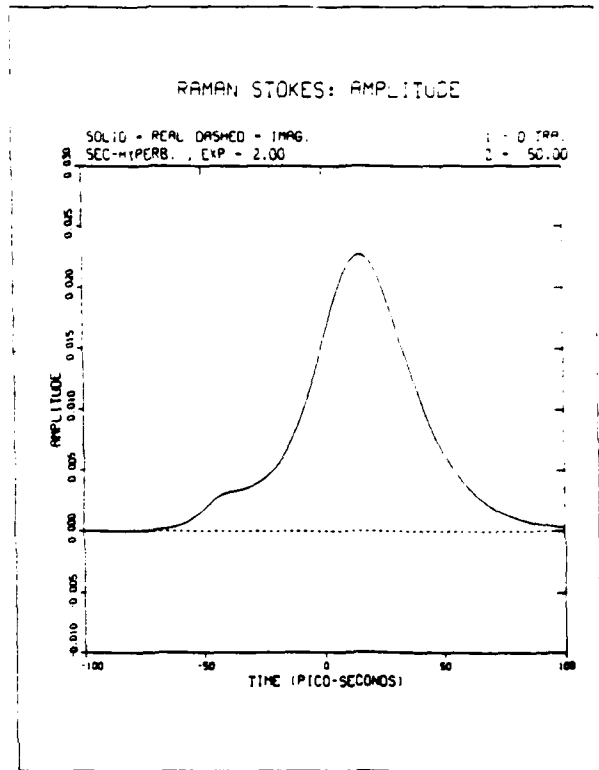
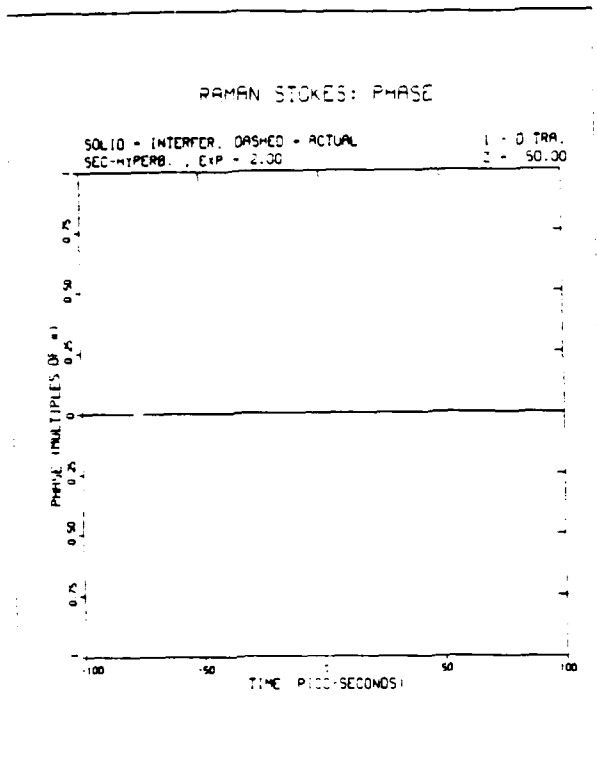
PLT2.DAT (Example A1)



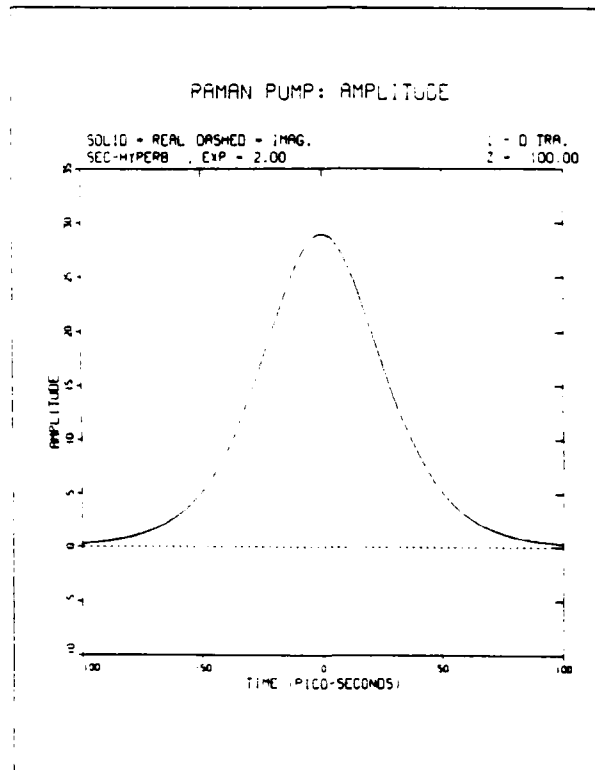
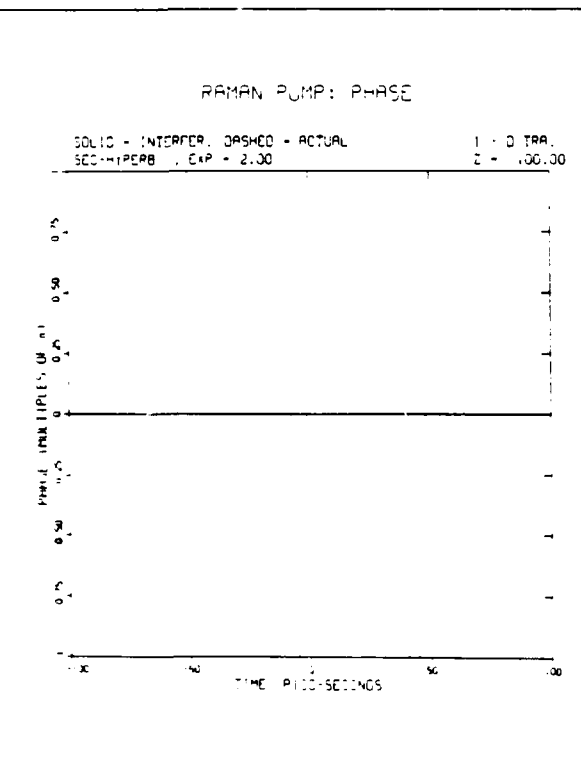
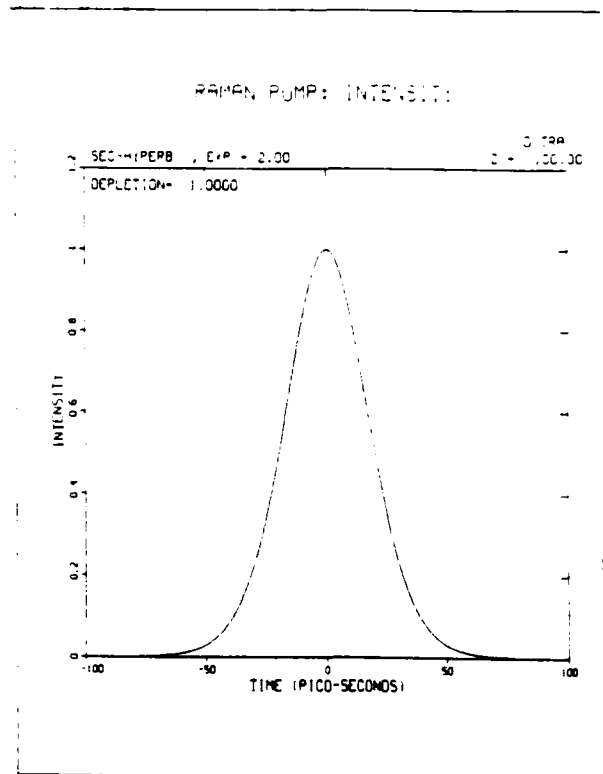
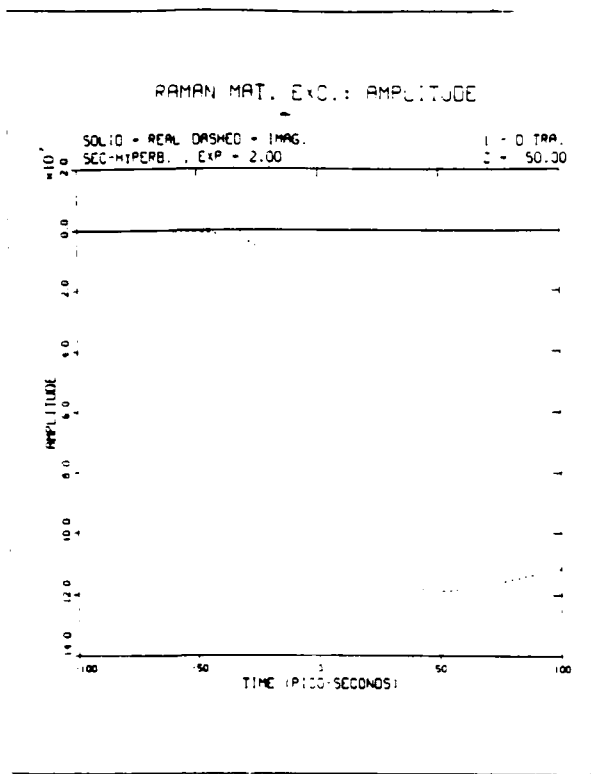
PLT2.DAT (Example A1)



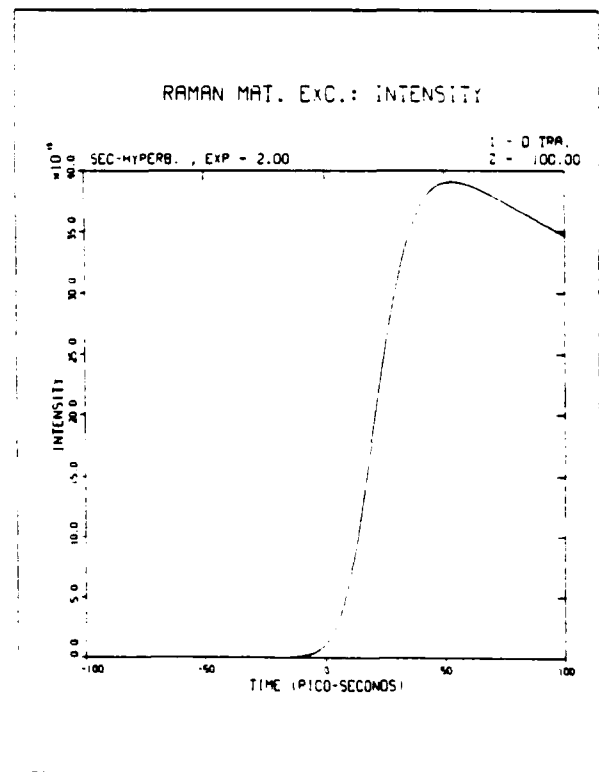
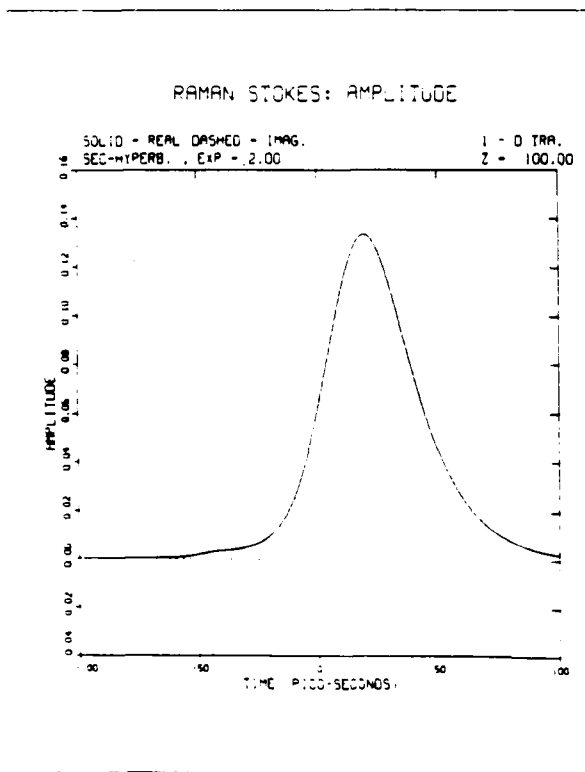
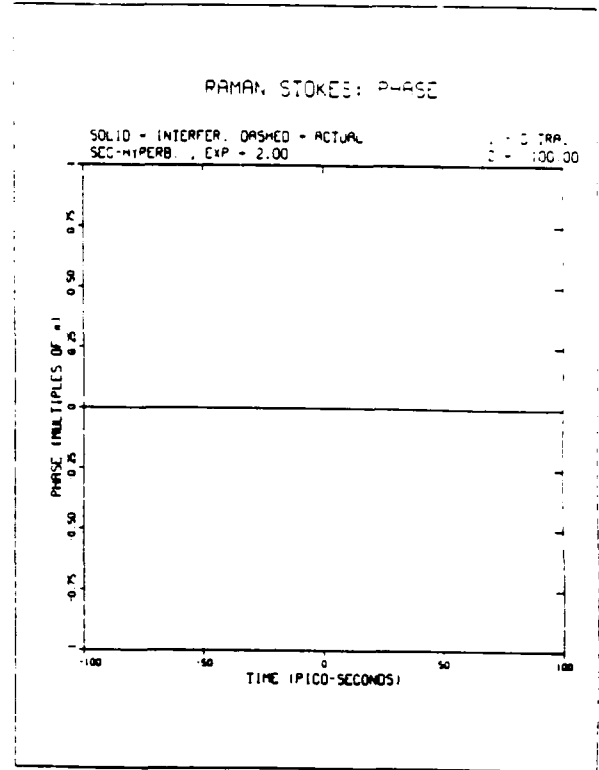
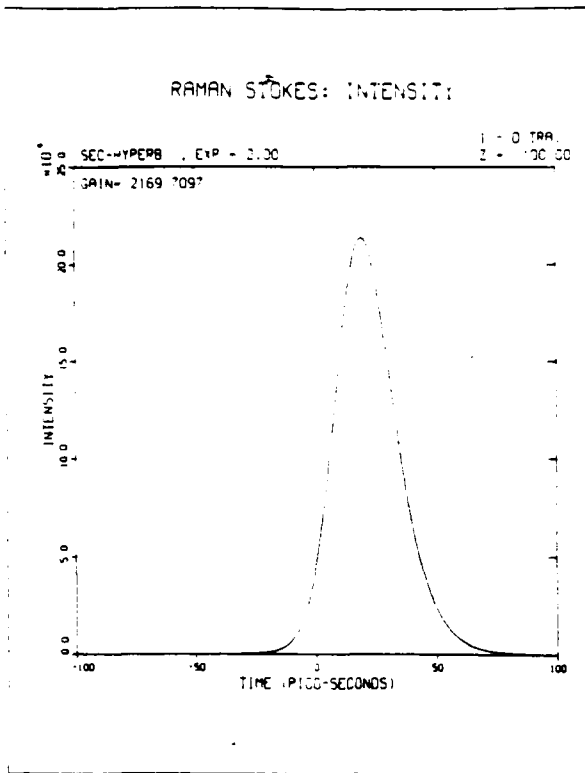
PLT2.DAT (Example A1)



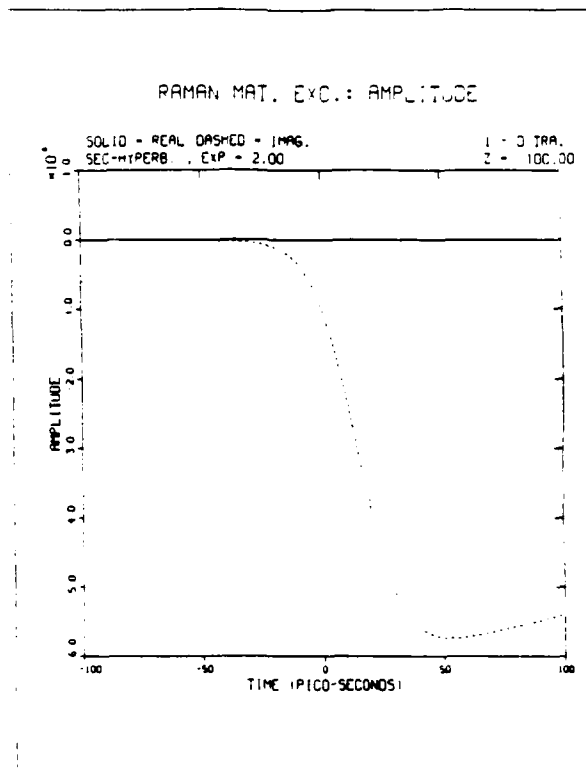
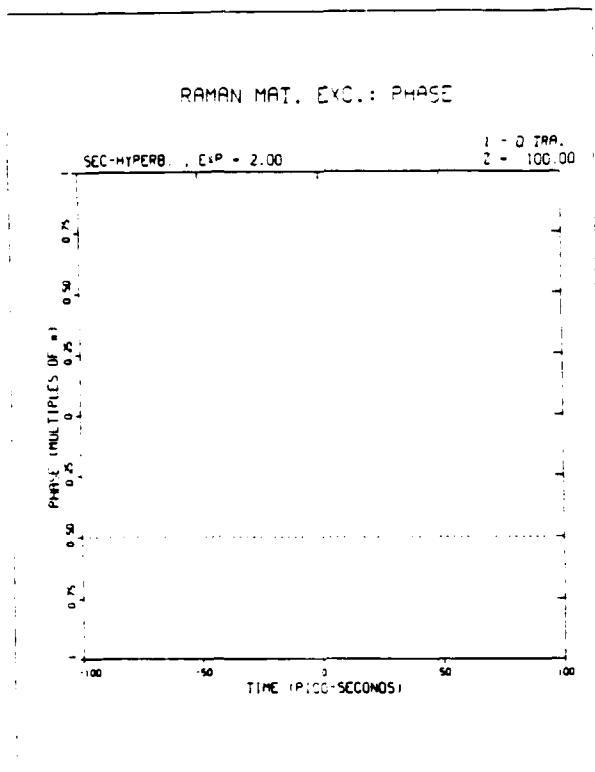
PLT2.DAT (Example A1)



PLT2.DAT (Example A1)



PLT2.DAT (Example A1)



EXAMPLE A2

CRJ3.JOB

```
AUDIT.  
FETCH,   DN=RJ3,TEXT='RAM2D1C.FOR'.  
CFT,     I=RJ3,ON=INZ.  
LDR,     AB=XRJ3,NX.  
SAVE,    DN=XRJ3.  
FETCH,   DN=NRAM,TEXT='NRJ3.DAT'.  
XRJ3.  
DISPOSE, DN=ERRM,DF=BB,WAIT,TEXT='CRJ3.MSG.'.  
AUDIT.  
EXIT.  
DISPOSE, DN=ERRM,DF=BB,WAIT,TEXT='CRJ3.MSG.'.  
DUMPJOB.  
DEBUG,   BLOCKS=VINIT.
```

CPJ3.JOB

AUDIT.
ACCESS, DN=DISLIB, ID=DISSPLA, OWN=LIBRARY.
ACCESS, DN=INTLIB, ID=DISSPLA, OWN=LIBRARY.
ACCESS, DN=DVSD, ID=DISSPLA, OWN=LIBRARY.
FETCH, DN=PJ3, TEXT='PRAM1CD.FOR'.
CFT, I=PJ3, ON=INZ.
LDR, LIB=INTLIB:DISLIB, NX, AB=XPJ3.
SAVE, DN=XPJ3.
RELEASE, DN=PJ3.
FETCH, DN=NPRAM1, TEXT='NPJ3.DAT'.
XPJ3.
DISPOSE, DN=META, DF=BB, WAIT, TEXT='PLT2.DAT'.
DISPOSE, DN=EPRM, DF=BB, WAIT, TEXT='CPJ3.MSG.'.
DISPOSE, DN=META, DF=BB, WAIT, TEXT='CPJ3.DSP'.
AUDIT.
EXIT.
DISPOSE, DN=EPRM, DF=BB, WAIT, TEXT='CPJ3.MSG.'.
DISPOSE, DN=META, DF=BB, WAIT, TEXT='PLT2.DAT'.
DISPOSE, DN=META, DF=BB, WAIT, TEXT='CPJ3.DSP'.
DUMPJOB.
DEBUG, BLOCKS=GRAPHS.

NRJ3.DAT

\$NAML

TOFF(2)=30.0,
 TWIDTH(2)=20.0,
 RINT(1)=1.0,
 RINT(2)=10.0,
 RINT(3)=1.0,
 PHL(3)=6.28,
 ITYPE(3)=4,
 RTYPE(3)=8.0,
 RIST=1.0E-8,
 ICOND=3,
 ZFINAL=100.0,
 ZKEEP=50.0,

\$

NAMELIST/NAML/NPUMP, YM, TM, ZINT, RKP, RKS, YOFF, TOFF, YWIDTH, TWIDTH,
 1 YOST, TOST, YWST, TWST, RINT, RIST, RAMASM, RALASM, NHYP, PHL, PHST, TOC,
 2 ITYPE, RTYPE, RABAMP, RDSLIM, ICOND, ZSTEP, ZFINAL, ZKEEP, NMAX, TTWO, GAIN

NPJ3.DAT

SFLDATE

DONYET=1,
 MONTH=03,
 DAY=28,
 YEAR=88,
 IPART=2,
 NEDN=2,

SCONDAT

LPRMT 1 =1,
 LPRMT 2 =1,
 LPRMT 3 =1,
 LPRMT 4 =1,
 NSEC=3,
 CSEC(1,1)=1.0,2.0),
 CSEC(1,2)=2.0,2.0),
 CSEC(1,3)=3.0,2.0),
 CSEC(2,1)=1.0,2.0),
 CSEC(2,2)=2.0,2.0),
 CSEC(2,3)=3.0,2.0),
 CSEC(7,1)=1.0,2.0),
 CSEC(7,2)=2.0,2.0),
 CSEC(7,3)=3.0,2.0),
 CSEC(8,1)=1.0,2.0),
 CSEC(8,2)=2.0,2.0),
 CSEC(8,3)=3.0,2.0),
 CSEC(13,1)=1.0,2.0),
 CSEC(13,2)=2.0,2.0),
 CSEC(13,3)=3.0,2.0),
 CSEC(14,1)=1.0,2.0),
 CSEC(14,2)=2.0,2.0),
 CSEC(14,3)=3.0,2.0),

S

SZPLOT

KZ 1 =1,
 KZ 2 =2,
 KZ 3 =3,

S

CRJ3.CPR

```

10:45:04 7223 0 0000 CSP
10:45:04 7226 0 0000 CSP
10:45:04 7229 0 0000 CSP
10:45:04 7232 0 0000 CSP
10:45:04 7235 0 0001 CSP
10:45:04 7238 0 0001 CSP
10:45:04 7241 0 0001 CSP
10:45:04 7244 0 0001 CSP
10:45:04 7246 0 0001 CSP
10:45:04 9833 0 0001 CSP
10:45:04 9836 0 0001 CSP
10:45:04 9840 0 0001 CSP
10:45:04 9843 0 0001 CSP
10:45:04 9846 0 0001 CSP
10:45:05 0688 0 0002 CSP
10:45:05 7984 0 0012 CSP
10:45:07 5805 0 1104 USER
10:45:08 5479 0 1135 USER
10:45:41 5632 0 3197 USER
10:45:41 5637 0 3198 USER
10:45:41 5642 0 3199 USER
10:45:41 5711 0 3200 CSP
10:45:45 7731 0 3201 SCP
10:45:45 7736 0 3201 SCP
10:45:45 7740 0 3201 SCP
10:45:50 1769 0 3201 SCP
10:45:50 4813 0 3205 CSP
10:45:50 9744 0 3209 USER
10:45:58 7978 2 1721 USER
10:45:58 7982 2 1721 USER
10:45:58 7988 2 1721 USER
10:45:58 7994 2 1723 USER
10:45:58 9784 2 1727 CSP
10:46:17 2556 2 4336 CSP
10:46:17 5188 2 4336 PDM
10:46:17 5191 2 4336 PDM
10:46:17 5211 2 4337 CSP
10:46:21 7099 2 4338 SCP
10:46:21 7102 2 4338 SCP
10:46:21 7106 2 4338 SCP
10:46:25 3771 2 4338 SCP
10:46:25 6713 2 4340 CSP
10:47:45 8254 21 3838 PDM
10:47:45 8257 21 3838 PDM
10:47:45 8266 21 3838 USER
10:47:45 8290 21 3838 CSP
10:47:50 6917 21 3841 SCP
10:47:50 6920 21 3841 SCP
10:47:50 6923 21 3841 SCP
10:47:53 8852 21 3845 USER
10:48:10 3499 21 5929 USER
10:48:10 3504 21 5930 USER
10:48:10 3509 21 5931 USER
10:48:10 3580 21 5931 CSP
10:48:10 3605 21 5932 CSP
10:48:10 3610 21 5932 CSP
10:48:10 3612 21 5932 CSP
10:48:10 5246 21 5933 USER
10:48:10 5252 21 5933 USER
10:48:10 5865 21 5933 USER

```

```

.....
WELCOME TO THE NRL CRAY XMP
.....
There will be no CRAY off line data set recalls on Tuesday or Wednesday
mornings between 2:00 AM and 7:00 AM in order for us to perform CLEANUP
runs on our CRAY archive tape library.
.....
CRAY X-MP SERIAL-415 65      NAVAL RESEARCH LABORATORY      04 20 88

CRAY OPERATING SYSTEM      COS 1 15  ASSEMBLY DATE 01 04 88

JOB JN-CRJ3.MPL-511000.US-DEFER
ACCOUNT AC-US-UPW-APW-
AC213 - ** TOTAL BUDGET WARNING LEVEL REACHED FOR THIS ACCOUNT NUMBER
AUDIT
AU003 -      187 DATASETS.      208621 BLOCKS.      106751139 WORDS
AU003 -      37 DATASETS.      28730 BLOCKS.      14700004 WORDS ONLINE
AU003 -      150 DATASETS.      179891 BLOCKS.      92051135 WORDS OFFLINE
FETCH.  DN-RJ3.TEXT- RAM2D1C FOR
VAX TO CRAY: 4SYSTEM-S-NORMAL. normal successful completion
VAX TO CRAY: FILE-$1SDUAL07:[HILFER.FR2]RAM2D1C FOR:46
VAX TO CRAY: 54936 BYTES TRANSFERRED
SS004 - DATASET RECEIVED FROM FRONT END
CFT.    I-RJ3.ON-INZ.
CF000 - CFT VERSION -      06 16 87 1.15BF2
CF001 - COMPILE TIME -      1.8512 SECONDS
CF002 -      1295 LINES.      803 STATEMENTS
CF003 -      75082 WORDS.      14540 I O BUFFERS USED
CF017 -      2 WARNINGS
LDR.    AB-XRJ3.NX.
SAVE.   DN-XRJ3.
PD000 - PDM - XRJ3      ID -      ED -      5 OWN - HILFER
PD000 - SAVE      COMPLETE
FETCH.  DN-NRAM.TEXT- NRJ3 DAT
VAX TO CRAY: 4SYSTEM-S-NORMAL. normal successful completion
VAX TO CRAY: FILE-$1SDUAL07:[HILFER.FR2]NRJ3.DAT:3
VAX TO CRAY: 592 BYTES TRANSFERRED
SS004 - DATASET RECEIVED FROM FRONT END
XRJ3.
PD000 - PDM - FJ3042088      ID -      ED -      1 OWN - HILFER
PD000 - SAVE      COMPLETE
UT003 - EXIT CALLED BY RAM2D1C
DISPOSE. DN-ERRH.DF-BB.WAIT.TEXT- CRJ3.MSG.
CRAY TO VAX: 4RHS-S-NORMAL. normal successful completion
CRAY TO VAX: FILE-$1SDUAL07:[HILFER.FR2]CRJ3.MSG:1
CRAY TO VAX: 20 BYTES TRANSFERRED
AUDIT.
AU003 -      189 DATASETS.      208967 BLOCKS.      106927943 WORDS
AU003 -      39 DATASETS.      28076 BLOCKS.      14876808 WORDS ONLINE
AU003 -      150 DATASETS.      179891 BLOCKS.      92051135 WORDS OFFLINE
EXIT.
END OF JOB

JOB NAME -      CRJ3
USER NUMBER -      HILFER
JOB SEQUENCE NUMBER -      38880

```

CRJ3.CPR

10 48 10 5259	21 5933	USER	TIME EXECUTING IN CPU -	0000:00:21	5933
10 48 10 5265	21 5934	USER	TIME WAITING TO EXECUTE -	0000:02:03	6841
10 48 10 5270	21 5934	USER	TIME WAITING FOR I O -	0000:00:39	5798
10 48 10 5274	21 5934	USER	TIME WAITING IN INPUT QUEUE	0000:00:00	0054
10 48 10 5278	21 5934	USER	MEMORY ' CPU TIME (MWDS' SEC) -	2	99854
10 48 10 5282	21 5934	USER	MEMORY ' I O WAIT TIME (MWDS' SEC) -	3	99978
10 48 10 5286	21 5935	USER	MINIMUM JOB SIZE (WORDS) -	43008	
10 48 10 5289	21 5935	USER	MAXIMUM JOB SIZE (WORDS) -	215040	
10 48 10 5293	21 5935	USER	MINIMUM FL (WORDS) -	38400	
10 48 10 5297	21 5935	USER	MAXIMUM FL (WORDS) -	210432	
10 48 10 5301	21 5935	USER	MINIMUM JTA (WORDS) -	3584	
10 48 10 5305	21 5935	USER	MAXIMUM JTA (WORDS) -	5120	
10 48 10 5308	21 5935	USER	DISK SECTORS MOVED -	2952	
10 48 10 5312	21 5935	USER	FSS SECTORS MOVED -	0	
10 48 10 5316	21 5935	USER	USER I O REQUESTS -	1379	
10 48 10 5320	21 5935	USER	USER I O SUSPENSIONS -	1599	
10 48 10 5323	21 5935	USER	OPEN CALLS -	45	
10 48 10 5327	21 5936	USER	CLOSE CALLS -	44	
10 48 10 5331	21 5936	USER	MEMORY RESIDENT DATASETS -	0	
10 48 10 5335	21 5936	USER	TEMPORARY DATASET SECTORS USED -	1	
10 48 10 5338	21 5936	USER	PERMANENT DATASET SECTORS ACCESSED -	1414	
10 48 10 5342	21 5936	USER	PERMANENT DATASET SECTORS SAVED -	346	
10 48 10 5346	21 5936	USER	SECTORS RECEIVED FROM FRONT END -	15	
10 48 10 5350	21 5936	USER	SECTORS QUEUED TO FRONT END -	1	
10 48 10 5354	21 5936	USER			
10 48 10 8906	21 6012	USER			
10 48 10 8909	21 6012	USER			
10 48 10 8913	21 6012	USER			
10 48 10 8916	21 6013	USER			
10 48 10 8920	21 6014	USER			
10 48 10 8924	21 6015	USER			
10 48 10 8928	21 6016	USER			
10 48 10 8931	21 6017	USER			
10 48 10 8935	21 6018	USER			
10 48 10 8939	21 6020	USER			
10 48 10 8943	21 6021	USER			
10 48 10 8947	21 6022	USER			
10 48 10 8951	21 6023	USER			
10 48 10 8955	21 6024	USER			
10 48 10 8958	21 6024	USER			
10 48 10 8961	21 6024	USER			

*** COST TABLE FOR THIS JOB ***					
JOBNAME -----			CRJ3		
USER IDENT -----			HILFER		
BEGAN EXECUTION ----			WED APR 20, 1988		
AT A PRIORITY OF --			10:45:04 HOURS		
AND JOB CLASS OF --			3		
			DSHALL		
21 599948	SECONDS OF CPU TIME	@ \$ 630.00	HR	--	\$ 3.78
2 998950	MEMORY'CPU (MWRD-SEC)	@ \$ 84.00	HR	--	\$ 0.07
4 004100	MEMORY'I O (MWRD-SEC)	@ \$ 84.00	HR	--	\$ 0.09
0 002954	I O MEGASECTORS MOVED	@ \$ 84.00	EA	--	\$ 0.25
0.000000	TAPE MOUNT(S)	@ \$ 5.00	EA	--	\$ 0.00
*** TOTAL COST FOR THIS JOB ***					
				--	\$ 4.19

CPJ3.CPR

```

10:52:01 9667 0 0000 CSP
10:52:01 9670 0 0000 CSP
10:52:01 9673 0 0000 CSP
10:52:01 9676 0 0000 CSP
10:52:01 9679 0 0001 CSP
10:52:01 9682 0 0001 CSP
10:52:01 9685 0 0001 CSP
10:52:01 9688 0 0001 CSP
10:52:01 9691 0 0001 CSP
10:52:01 9715 0 0001 CSP
10:52:01 9718 0 0001 CSP
10:52:01 9722 0 0001 CSP
10:52:01 9725 0 0001 CSP
10:52:01 9728 0 0002 CSP
10:52:01 9949 0 0002 CSP
10:52:02 0243 0 0014 CSP
10:52:03 4287 0 1114 USER
10:52:04 0746 0 1146 USER
10:52:31 6486 0 3240 USER
10:52:31 6491 0 3241 USER
10:52:31 6496 0 3242 USER
10:52:31 6581 0 3246 CSP
10:52:31 9239 0 3246 PDM
10:52:31 9242 0 3246 PDM
10:52:31 9261 0 3250 CSP
10:52:32 1834 0 3250 PDM
10:52:32 1838 0 3250 PDM
10:52:32 1862 0 3253 CSP
10:52:32 4003 0 3254 PDM
10:52:32 4006 0 3254 PDM
10:52:32 4022 0 3254 CSP
10:52:39 7022 0 3256 SCP
10:52:39 7025 0 3256 SCP
10:52:39 7028 0 3256 SCP
10:52:42 1554 0 3256 SCP
10:52:42 5372 0 3260 CSP
10:52:43 0746 0 3263 USER
10:53:01 3381 8 8122 USER
10:53:01 3428 8 8122 USER
10:53:01 3432 8 8122 USER
10:53:01 4860 8 8128 CSP
10:53:08 0174 10 1233 CSP
10:53:08 2797 10 1233 PDM
10:53:08 2800 10 1233 PDM
10:53:08 2819 10 1234 CSP
10:53:08 2858 10 1235 CSP
10:53:13 5783 10 1236 SCP
10:53:13 5786 10 1236 SCP
10:53:13 5789 10 1236 SCP
10:53:17 7079 10 1236 SCP
10:53:17 9790 10 1238 CSP
10:53:20 9282 10 1284 PDM
10:53:20 9286 10 1284 PDM
10:54:05 5294 24 0181 USER
10:54:05 5320 24 0181 CSP
10:54:21 1493 24 0183 SCP
10:54:21 1496 24 0183 SCP
10:54:21 1500 24 0183 SCP
10:54:23 8585 24 0184 CSP
10:54:29 7490 24 0186 SCP

```

```

.....
WELCOME TO THE NRL CRAY XMP
.....
There will be no CRAY off-line data set recalls on Tuesday or Wednesday
mornings between 2:00 AM and 7:00 AM in order for us to perform CLEANUP
runs on our CRAY archive tape library.
.....
CRAY X-MP SERIAL 415 65   NAVAL RESEARCH LABORATORY   04 20 88
CRAY OPERATING SYSTEM   COS 1.15   ASSEMBLY DATE 01 04 88

JOB:JN-CPJ3.MFL-511000.US-DEFER
ACCOUNT:AC-US-UPW-APW-
AC213 - ** TOTAL BUDGET WARNING LEVEL REACHED FOR THIS ACCOUNT NUMBER
AUDIT.
AU003 - 189 DATASETS, 206967 BLOCKS, 106927943 WORDS
AU003 - 39 DATASETS, 29076 BLOCKS, 14876808 WORDS ONLINE
AU003 - 150 DATASETS, 179891 BLOCKS, 92051135 WORDS OFFLINE
ACCESS, DN-DISLIB.ID-DISSPLA.OWN-LIBRARY.
PD000 - PDM - DISLIB ID - DISSPLA ED - 1 OWN - LIBRARY
PD000 - ACCESS COMPLETE
ACCESS, DN-INTLIB.ID-DISSPLA.OWN-LIBRARY.
PD000 - PDM - INTLIB ID - DISSPLA ED - 1 OWN - LIBRARY
PD000 - ACCESS COMPLETE
ACCESS, DN-DVSD.ID-DISSPLA.OWN-LIBRARY.
PD000 - PDM - DVSD ID - DISSPLA ED - 1 OWN - LIBRARY
PD000 - ACCESS COMPLETE
FETCH, DN-FJ3.TEXT-PRAM1CD.FOR.
VAX TO CRAY: %SYSTEM-S-NORMAL, normal successful completion
VAX TO CRAY: FILE-$1SDUAL07:[HILFER.FR2]PRAM1CD.FOR:9
VAX TO CRAY: 177688 BYTES TRANSFERRED
SS004 - DATASET RECEIVED FROM FRONT END
CFT, 1-FJ3.ON-INZ.
CF000 - CFT VERSION - 06 16 87 1.15BF2
CF001 - COMPILE TIME - 8.4859 SECONDS
CF002 - 3938 LINES, 2705 STATEMENTS
CF003 - 107850 WORDS, 14540 I/O BUFFERS USED
LDR, LIB-INTLIB:DISLIB.NX.AB-XPJ3.
SAVE, DN-XPJ3.
PD000 - PDM - XPJ3 ID - ED - 8 OWN - HILFER
PD000 - SAVE COMPLETE
RELEASE, DN-FJ3.
FETCH, DN-NPRAM1.TEXT-NPJ3.DAT.
VAX TO CRAY: %SYSTEM-S-NORMAL, normal successful completion
VAX TO CRAY: FILE-$1SDUAL07:[HILFER.FR2]NPJ3.DAT:9
VAX TO CRAY: 1056 BYTES TRANSFERRED
SS004 - DATASET RECEIVED FROM FRONT END
XPJ3.
PD000 - PDM - FJ3042088 ID - ED - 1 OWN - HILFER
PD000 - ACCESS COMPLETE
UT003 - EXIT CALLED BY PRAM1CD
DISPOSE, DN-META.DF-BB.WAIT.TEXT-PLT2.DAT.
CRAY TO VAX: %RMS-S-NORMAL, normal successful completion
CRAY TO VAX: FILE-$1SDUAL07:[HILFER.FR2]PLT2.DAT:1
CRAY TO VAX: 547920 BYTES TRANSFERRED
DISPOSE, DN-EPRM.DF-BB.WAIT.TEXT-CPJ3.MSG.
CRAY TO VAX: %RMS-S-NORMAL, normal successful completion

```


CPJ3.CPR

```

10:54:29 7493      24 0188  SCP      CRAY TO VAX: FILE-S1SDUA107-[HILFER FR2]CPJ3 MSG:1
10:54:29 7496      24 0188  SCP      CRAY TO VAX: 3404 BYTES TRANSFERRED
10:54:34 8297      24 0188  CSP      DISPOSE: DN-META.DF-BB.WAIT.TEXT- CPJ3.DSP
10:54:34 8315      24 0188  EXP      SY001 - RLS COULD NOT FIND A DMT FOR  META
10:54:35 1661      24 0192  USER     AUDIT
10:55:05 8981      24 2272  USER     AU003      190 DATASETS      209325 BLOCKS      107111217 WORDS
10:55:05 8988      24 2272  USER     AU003      40 DATASETS      29434 BLOCKS      15080082 WORDS ONLINE
10:55:05 8990      24 2273  USER     AU003      150 DATASETS      179891 BLOCKS      92051135 WORDS OFFLINE
10:55:05 8847      24 2274  CSP      EXIT
10:55:05 8868      24 2274  CSP      END OF JOB
10:55:05 8869      24 2274  CSP
10:55:05 8874      24 2274  CSP
10:55:06 0309      24 2276  USER     JOB NAME      CPJ3
10:55:06 0312      24 2276  USER     USER NUMBER -      HILFER
10:55:06 0318      24 2276  USER     JOB SEQUENCE NUMBER -      38928
10:55:06 0319      24 2276  USER
10:55:06 0323      24 2276  USER     TIME EXECUTING IN CPU      0000:00:24.2275
10:55:06 0327      24 2276  USER     TIME WAITING TO EXECUTE      0000:02:01.7699
10:55:06 0330      24 2276  USER     TIME WAITING FOR I O      0000:00:38.4403
10:55:06 0334      24 2276  USER     TIME WAITING IN INPUT QUEUE -      0000:00:00.0166
10:55:06 0338      24 2277  USER     MEMORY CPU TIME (HWDS*SEC) -      4.85757
10:55:06 0342      24 2277  USER     MEMORY I O WAIT TIME (HWDS*SEC) -      4.39400
10:55:06 0345      24 2277  USER     MINIMUM JOB SIZE (WORDS) -      43008
10:55:06 0349      24 2277  USER     MAXIMUM JOB SIZE (WORDS) -      311296
10:55:06 0651      24 2277  USER     MINIMUM FL (WORDS) -      38400
10:55:06 0654      24 2277  USER     MAXIMUM FL (WORDS) -      308178
10:55:06 0658      24 2277  USER     MINIMUM JTA (WORDS) -      3584
10:55:06 0661      24 2277  USER     MAXIMUM JTA (WORDS) -      5120
10:55:06 0665      24 2278  USER     DISK SECTORS MOVED -      4492
10:55:06 0669      24 2278  USER     FSS SECTORS MOVED -      0
10:55:06 0672      24 2278  USER     USER I O REQUESTS -      1553
10:55:06 0676      24 2278  USER     USER I O SUSPENSIONS -      1877
10:55:06 0679      24 2278  USER     OPEN CALLS -      51
10:55:06 0683      24 2278  USER     CLOSE CALLS -      49
10:55:06 0686      24 2278  USER     MEMORY RESIDENT DATASETS -      0
10:55:06 0690      24 2278  USER     TEMPORARY DATASET SECTORS USED -      183
10:55:06 0694      24 2278  USER     PERMANENT DATASET SECTORS ACCESSED -      2451
10:55:06 0697      24 2278  USER     PERMANENT DATASET SECTORS SAVED -      358
10:55:06 0701      24 2278  USER     SECTORS RECEIVED FROM FRONT END -      45
10:55:06 0704      24 2278  USER     SECTORS QUEUED TO FRONT END -      138
10:55:06 3479      24 2355  USER
10:55:06 3481      24 2355  USER
10:55:06 3485      24 2356  USER
10:55:06 3489      24 2357  USER
10:55:06 3493      24 2358  USER
10:55:06 3497      24 2359  USER
10:55:06 3500      24 2360  USER
10:55:06 3504      24 2361  USER
10:55:06 3508      24 2362  USER
10:55:06 3512      24 2363  USER
10:55:06 3517      24 2364  USER
10:55:06 3521      24 2366  USER
10:55:06 3525      24 2367  USER
10:55:06 3529      24 2368  USER
10:55:06 3531      24 2368  USER
10:55:06 3534      24 2368  USER

```

.....

*** COST TABLE FOR THIS JOB ***			
JOBNAME	USER IDENT	CPJ3	
24 234252	SECONDS OF CPU TIME	4.858011	4 24
4.858011	MEMORY CPU (MWRD-SEC)	4.397854	0 11
4.397854	MEMORY I O (MWRD-SEC)	0.004494	0 10
0.004494	I O MEGASECTORS MOVED	0.000000	0 38
0.000000	TAPE MOUNT(S)		0 00
*** TOTAL COST FOR THIS JOB ***			4 83

.....

PLT2.DAT (Example A2)

LIST OF INPUT PARAMETERS

ICOND = 1
 NMAX = 4000
 NPUMP = 2
 NT = 1024
 NT = 1
 GAIN = 0.0000
 PHST = 0.0000
 RALASH = 0.0000
 RAMASH = 0.0000
 RAST = 0.0000
 RKP = 0.0000
 RKS = 0.0000
 TDC = 0.0000
 TOST = 0.0000
 TTHO = 0.0000
 TWST = 0.0000
 TWNA = 0.0000
 TWKEP = 0.0000
 TWSTEP = 0.0000

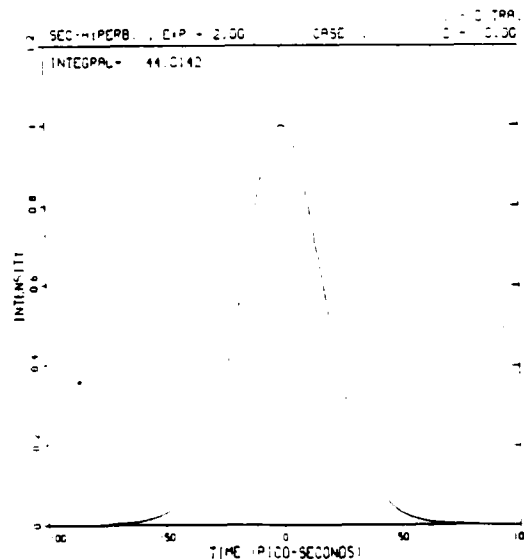
LIST OF INPUT PARAMETERS CONTD.

ITYPE = 1
 PML(1-10) = 0.0000 0.0000 0.2800 0.0000 0.0000
 RINT(1-10) = 0.0000 0.0000 0.0000 0.0000 0.0000
 RTYPE = 0.0000 0.0000 0.0000 0.0000 0.0000
 TML(1,2) = 0.0000 0.0000
 TOFF(1-10) = 0.0000 0.0000 0.0000 0.0000 0.0000
 TWIDTH = 0.0000 0.0000 0.0000 0.0000 0.0000

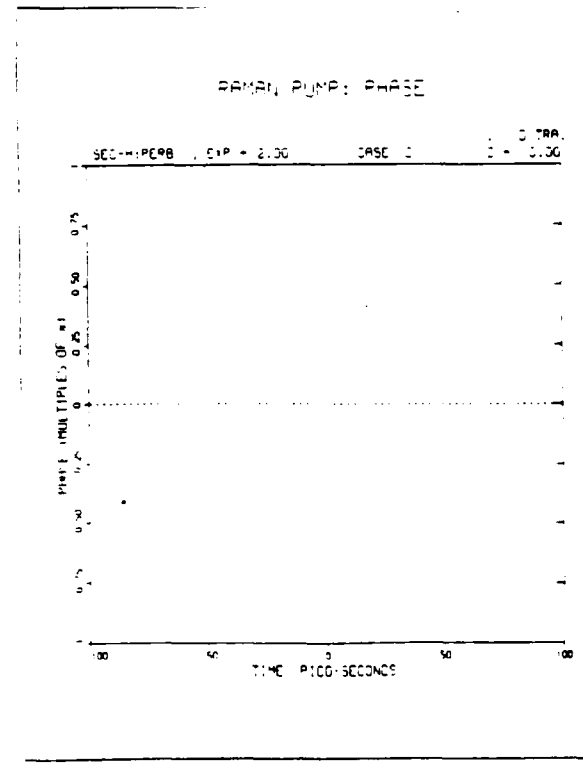
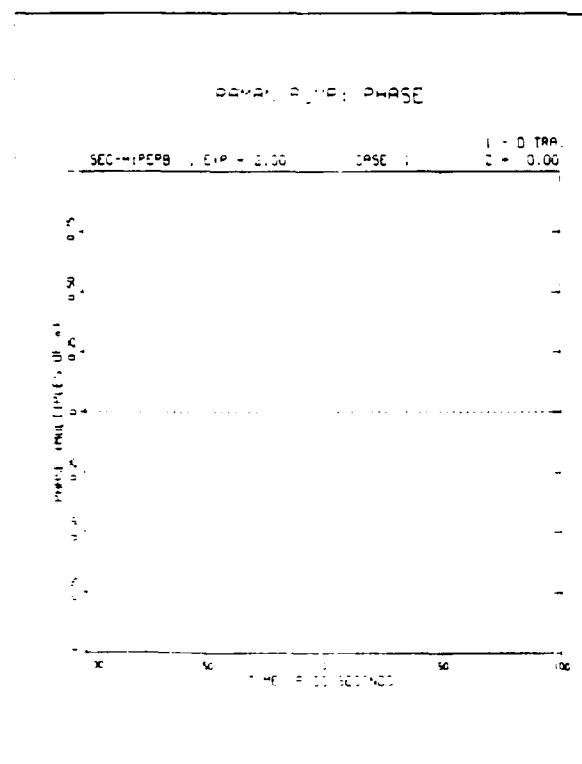
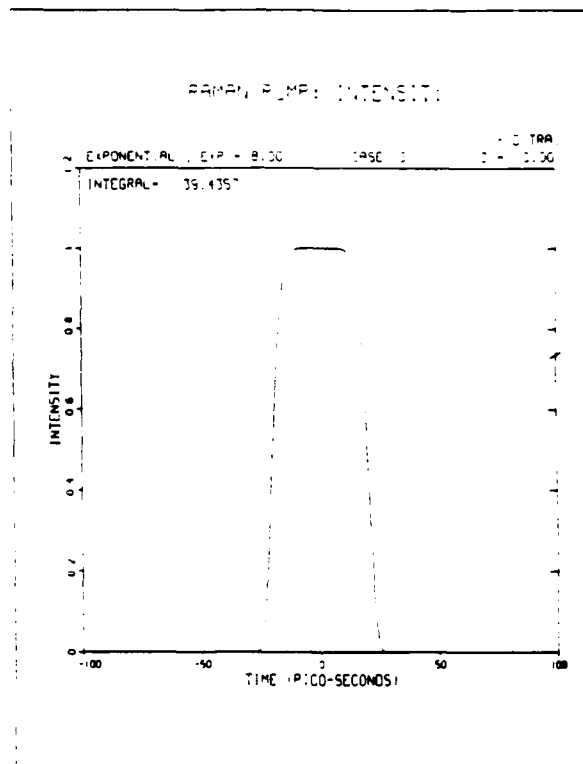
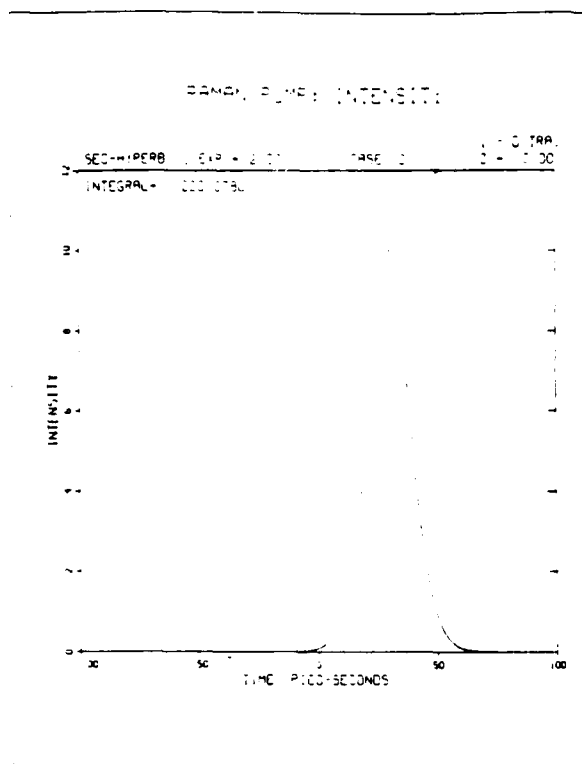
LIST OF INPUT PARAMETERS CONTD.

ICOND = 1
 NMAX = 4000
 NPUMP = 2
 NT = 1024
 NT = 1
 GAIN = 0.0000
 PHST = 0.0000
 RALASH = 0.0000
 RAMASH = 0.0000
 RAST = 0.0000
 RKP = 0.0000
 RKS = 0.0000
 TDC = 0.0000
 TOST = 0.0000
 TTHO = 0.0000
 TWST = 0.0000
 TWNA = 0.0000
 TWKEP = 0.0000
 TWSTEP = 0.0000

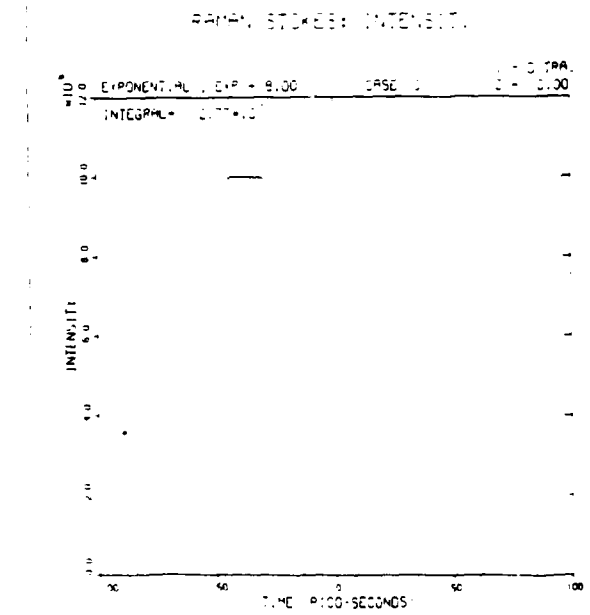
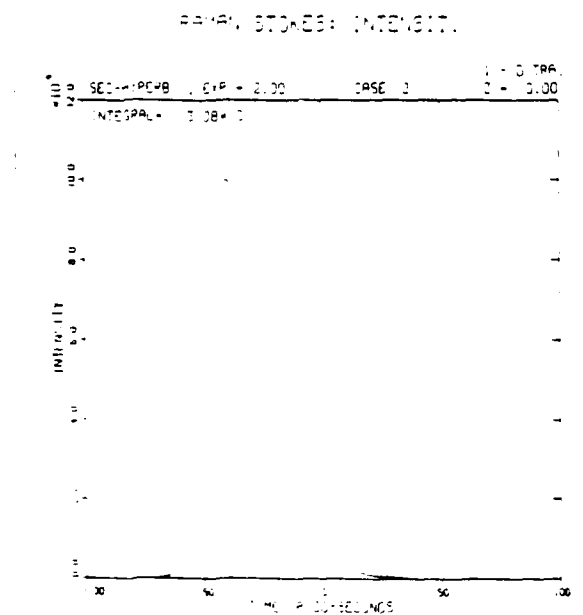
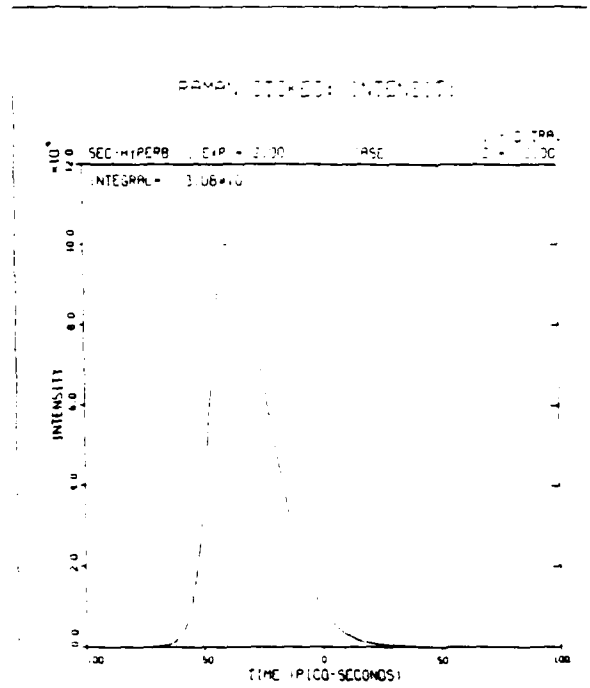
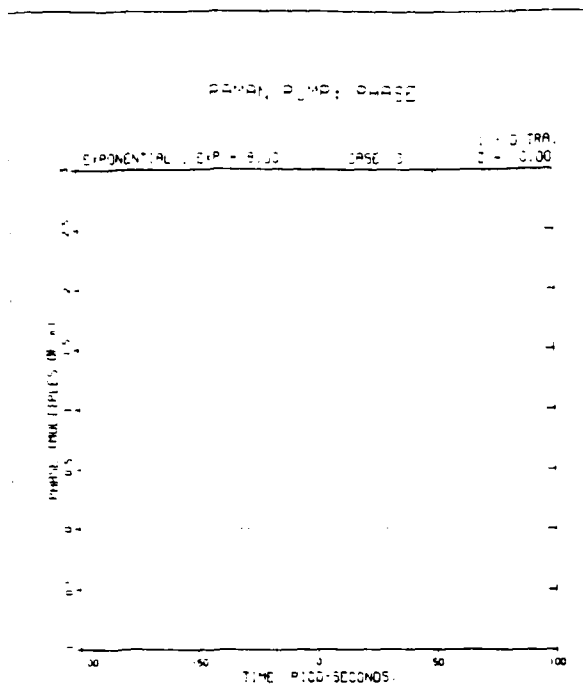
RAMAN PUMP: INTENSITY



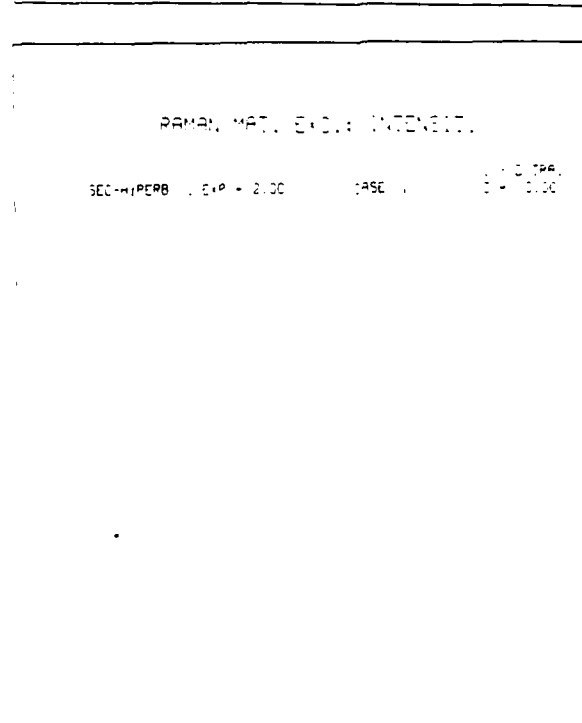
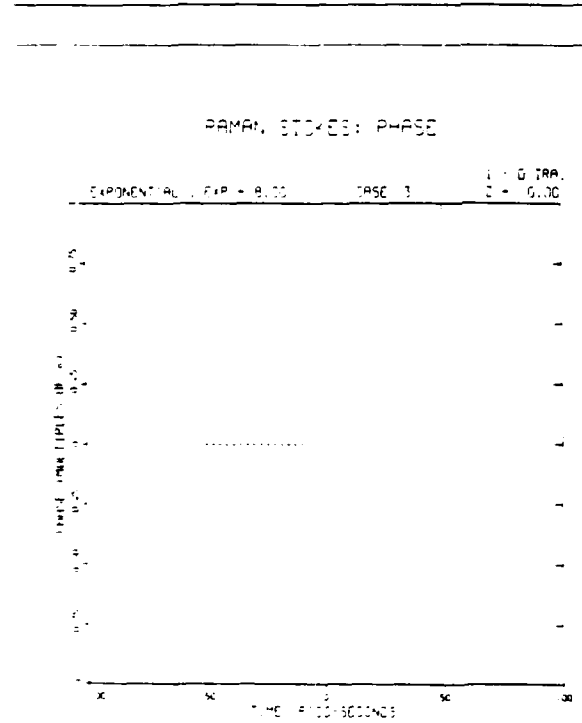
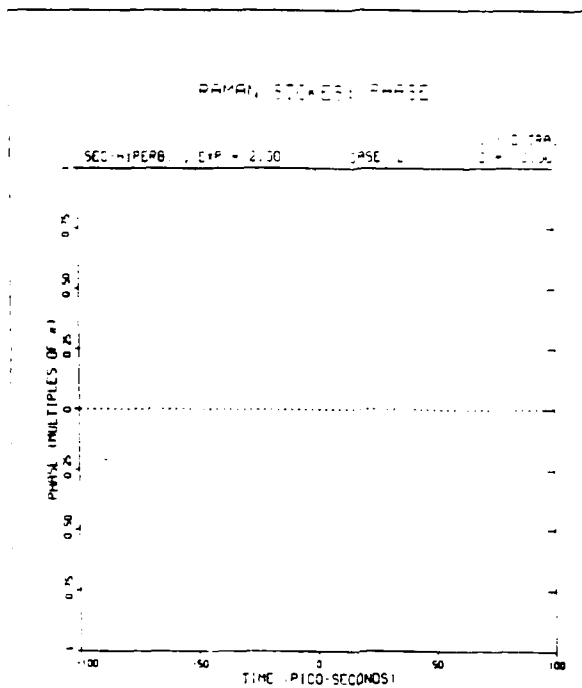
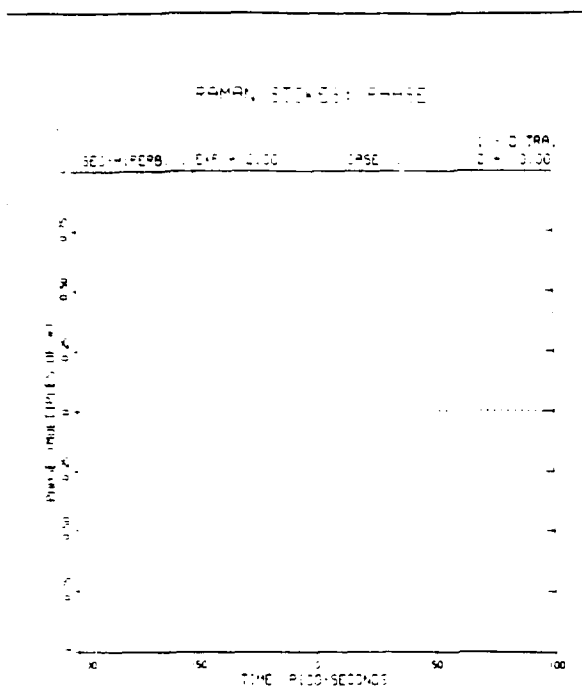
PLT2.DAT (Example A2)



PLT2.DAT (Example A2)



PLT2.DAT (Example A2)



PLT2.DAT (Example A2)

RAMAN MAT. EXC.: INTENSIT.

SEC-HYPERB. , EXP = 2.00 CASE 2 1 = 0.799
2 = 0.000

RAMAN MAT. EXC.: INTENSIT.

EXPONENTIAL , EXP = 8.00 CASE 3 1 = 0.799
2 = 0.000

RAMAN MAT. EXC.: PHASE

SEC-HYPERB. , EXP = 2.00 CASE 1 1 = 0.799
2 = 0.000

RAMAN MAT. EXC.: PHASE

SEC-HYPERB. , EXP = 2.00 CASE 2 1 = 0.799
2 = 0.000

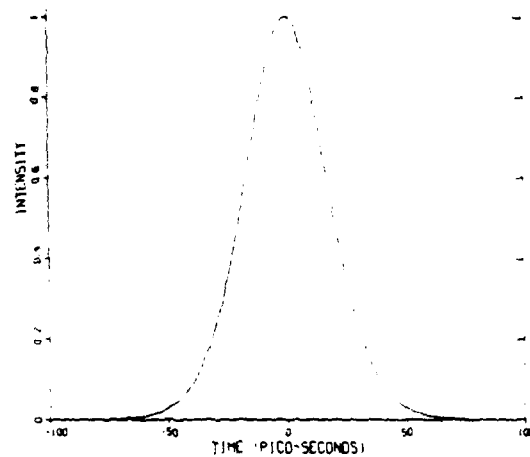
PLT2.DAT (Example A2)

RAMAN PUMP: INTENSIT

EXPONENTIAL, EXP = 8.00 CASE 1
1 = 0.00
2 = 0.00

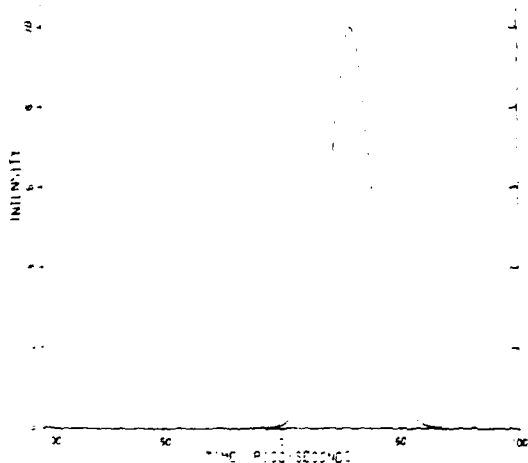
RAMAN PUMP: INTENSIT

SEC-HYPERB, EXP = 2.00 CASE 1
1 = 0.00
2 = 50.00
DEPLETION = 1.116



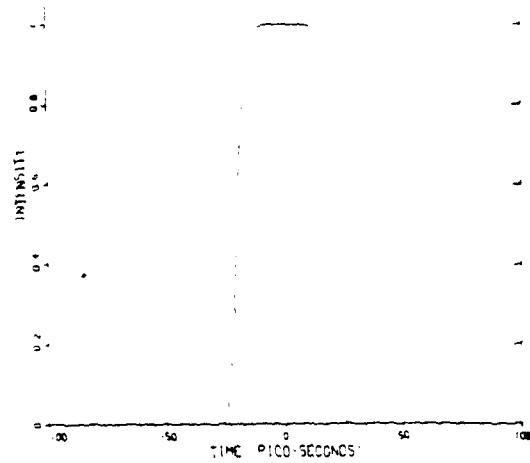
RAMAN PUMP: INTENSIT

SEC-HYPERB, EXP = 2.00 CASE 2
1 = 0.00
2 = 50.00
DEPLETION = 5.519

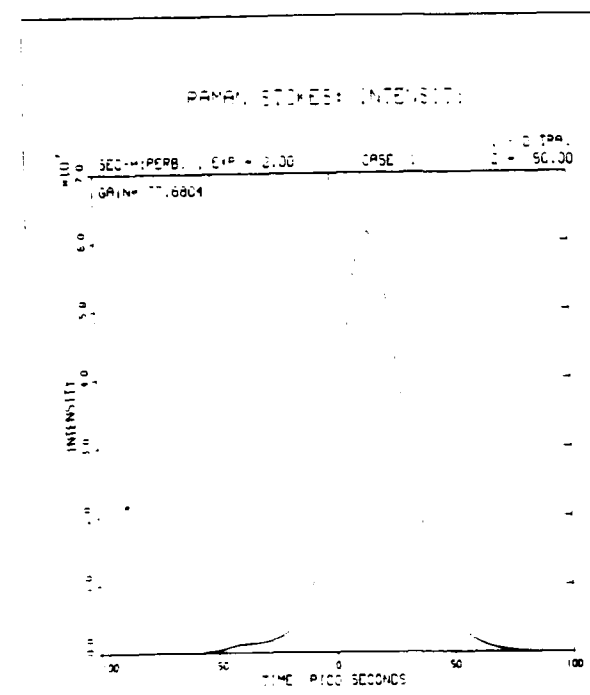
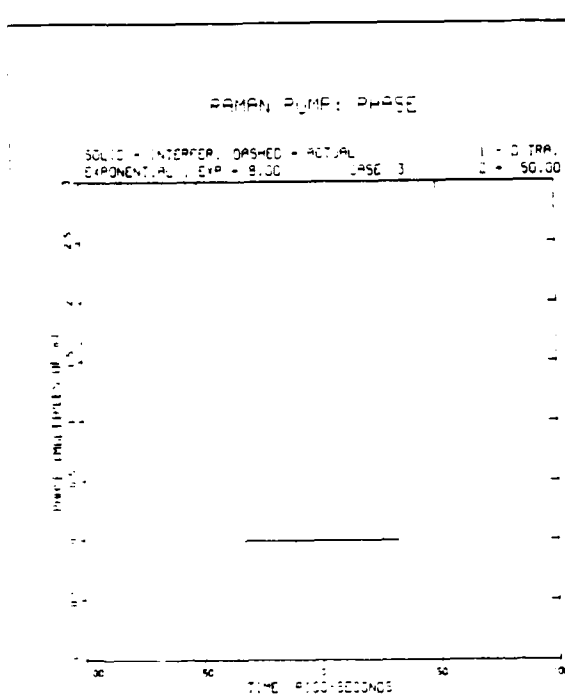
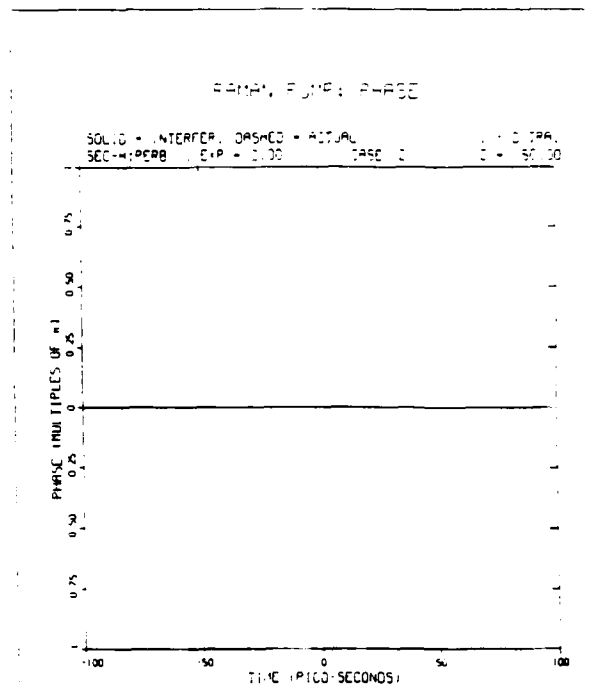
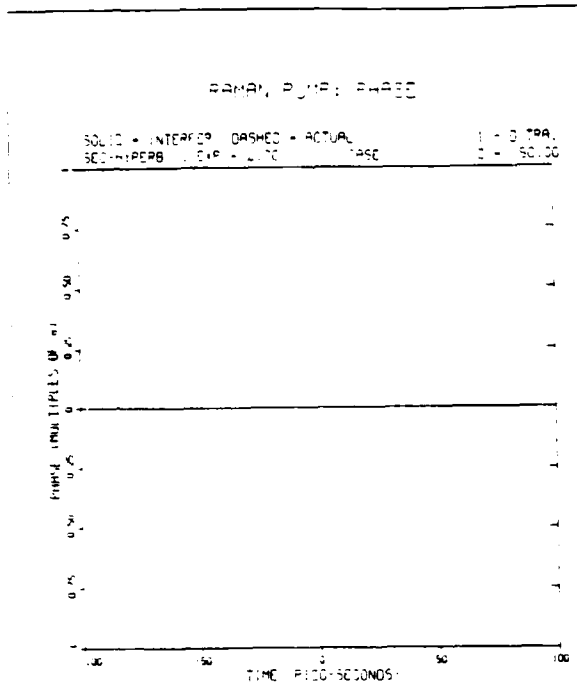


RAMAN PUMP: INTENSIT

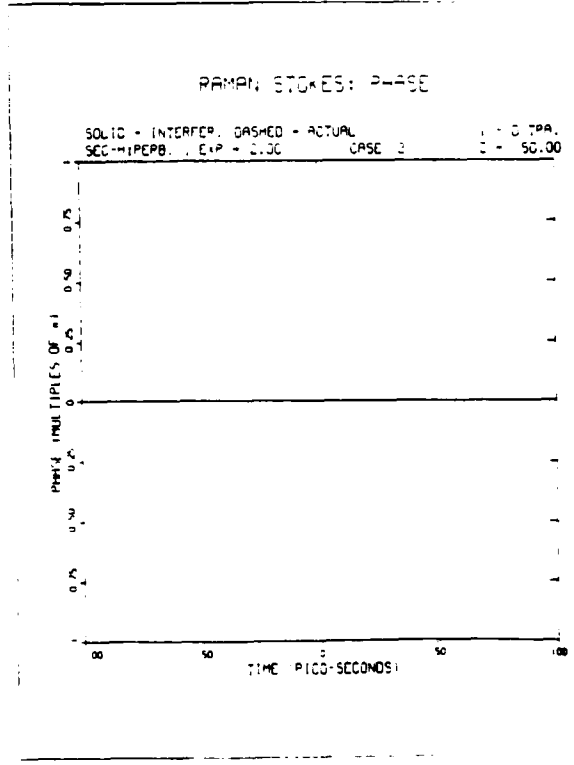
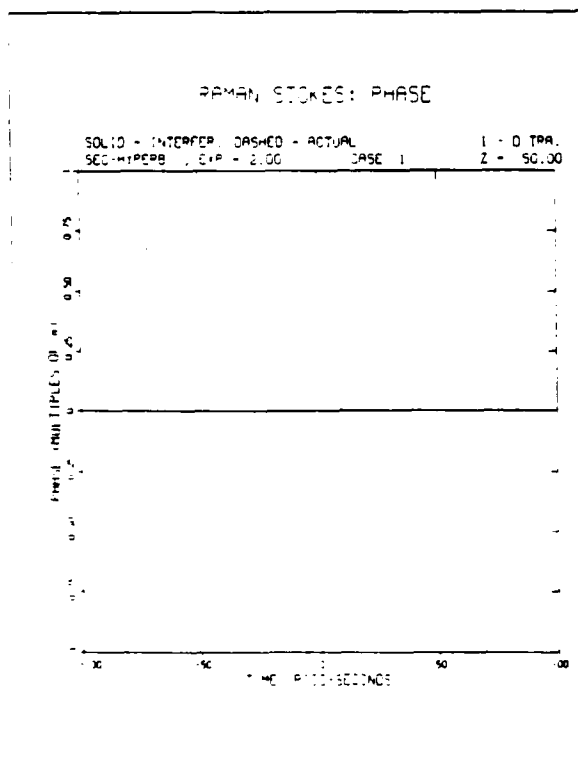
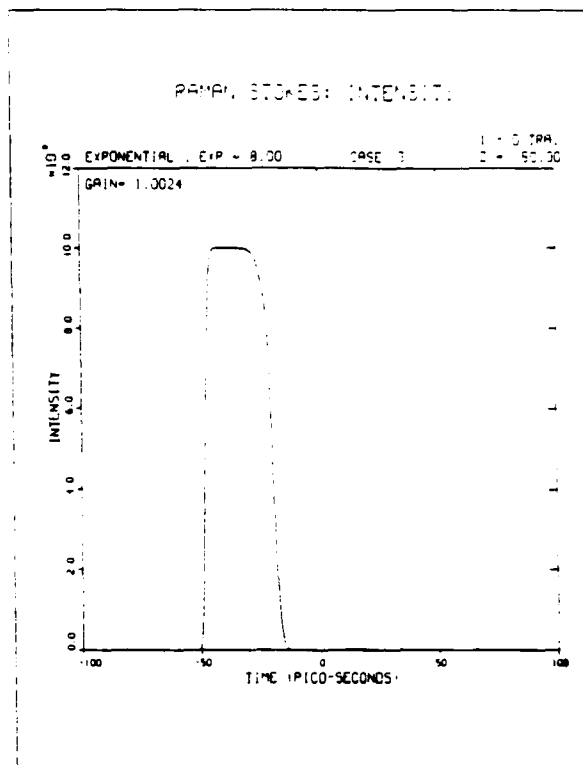
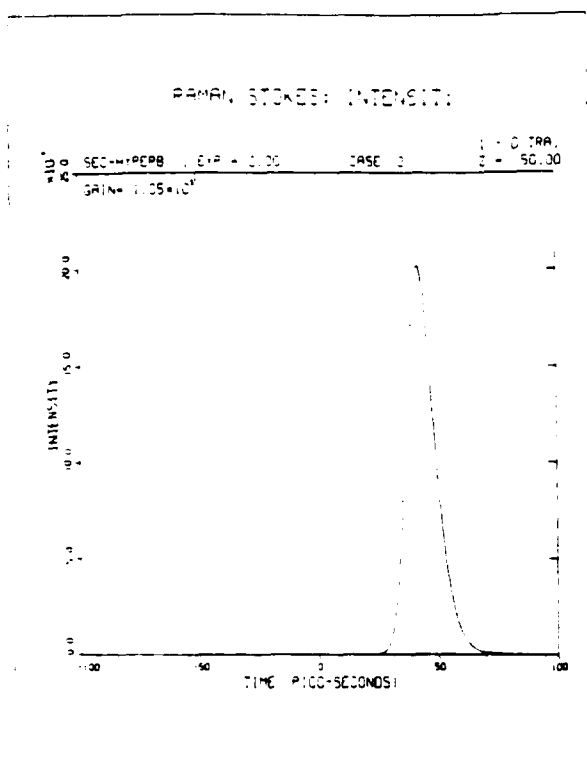
EXPONENTIAL, EXP = 8.00 CASE 1
1 = 0.00
2 = 50.00
DEPLETION = 1.000



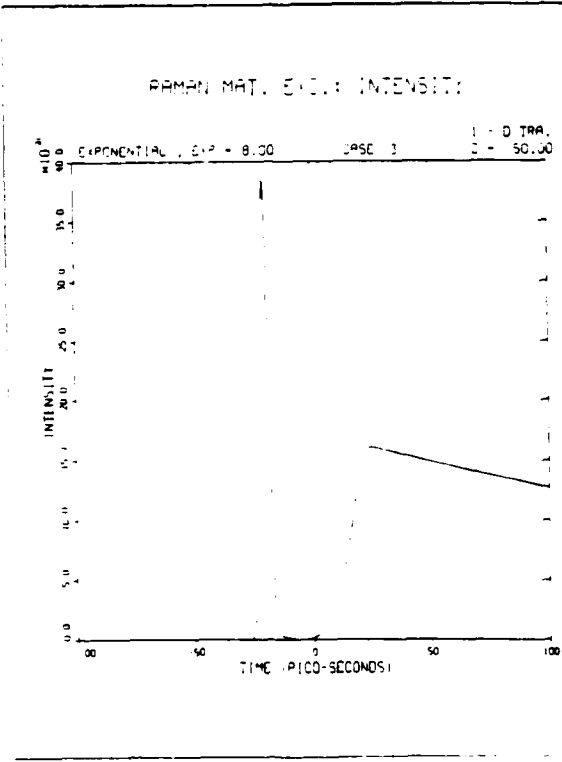
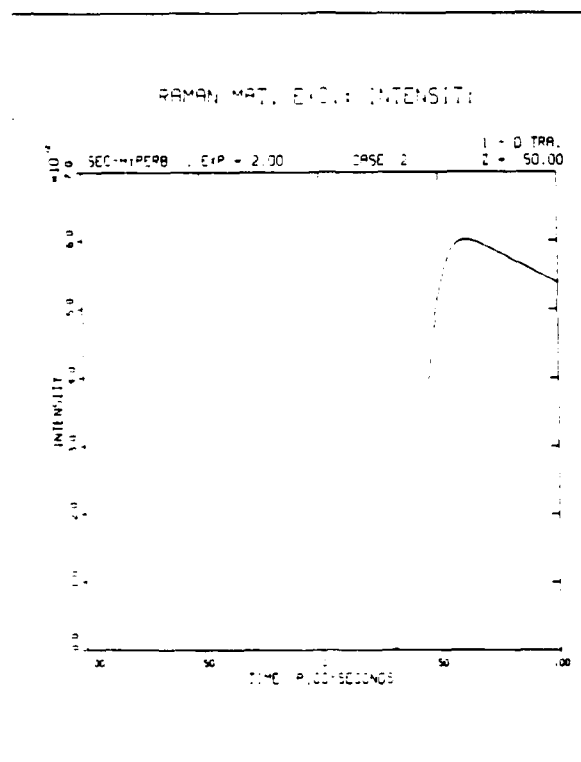
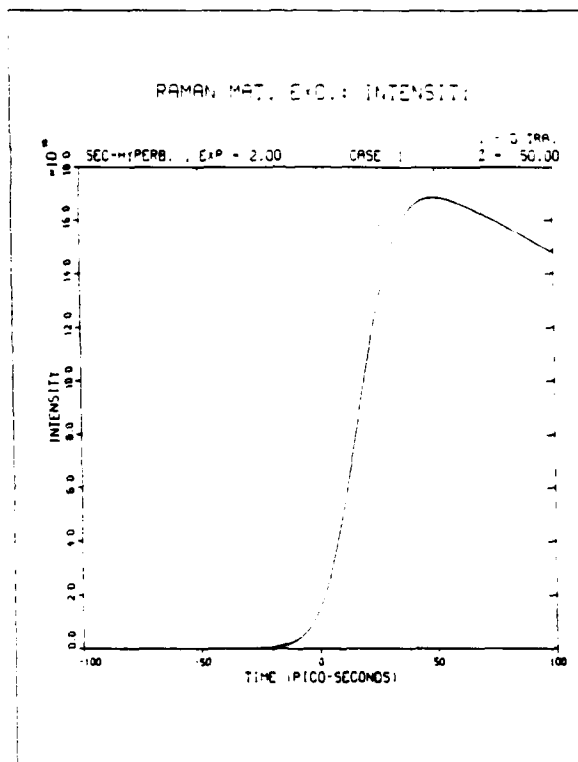
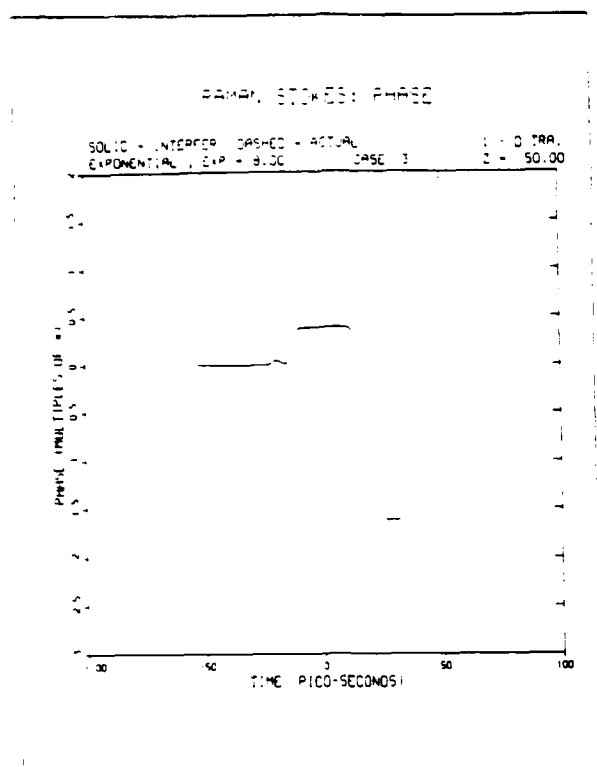
PLT2.DAT (Example A2)



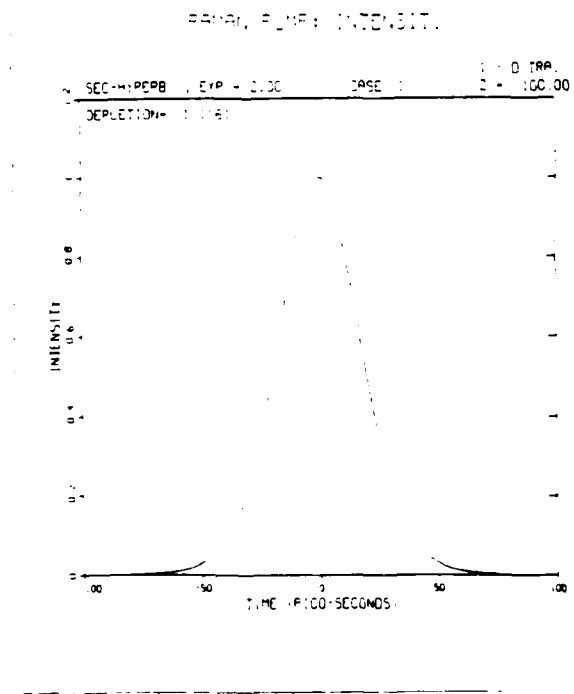
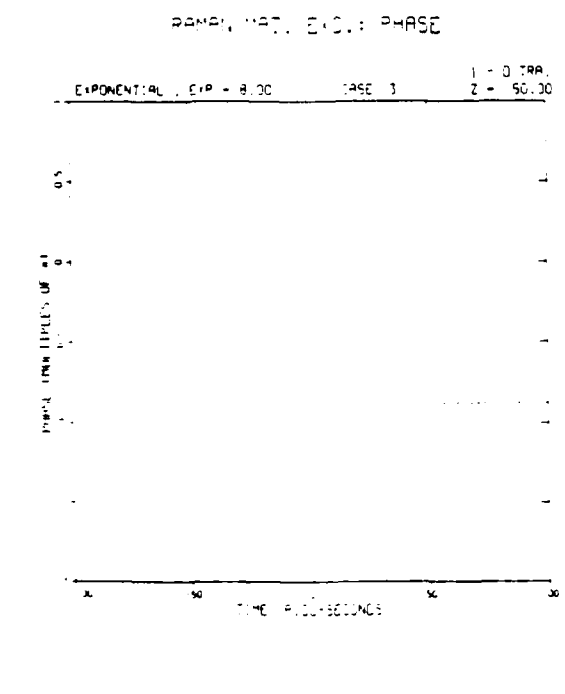
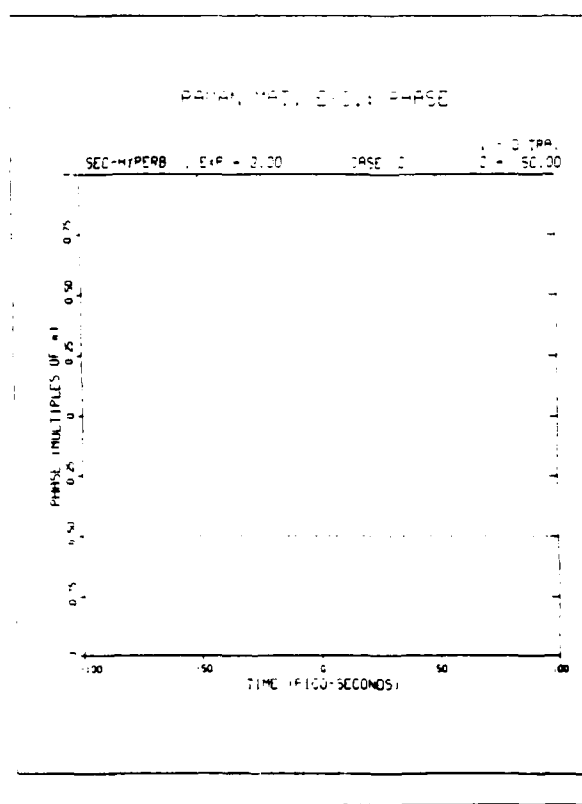
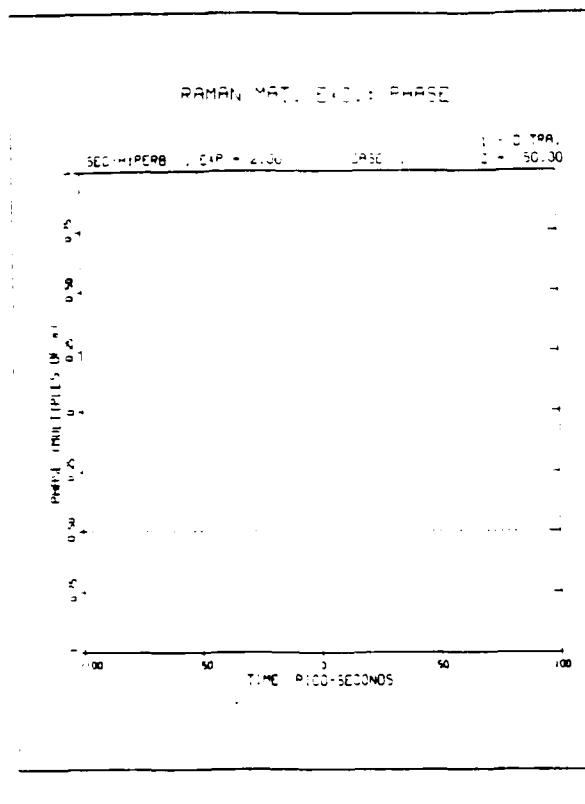
PLT2.DAT (Example A2)



PLT2.DAT (Example A2)

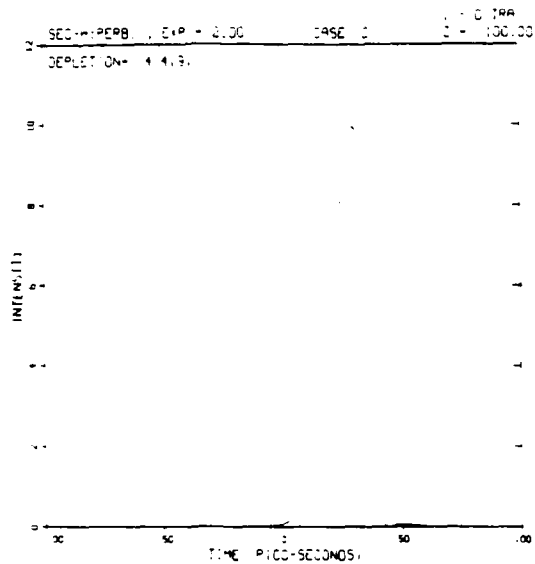


PLT2.DAT (Example A2)

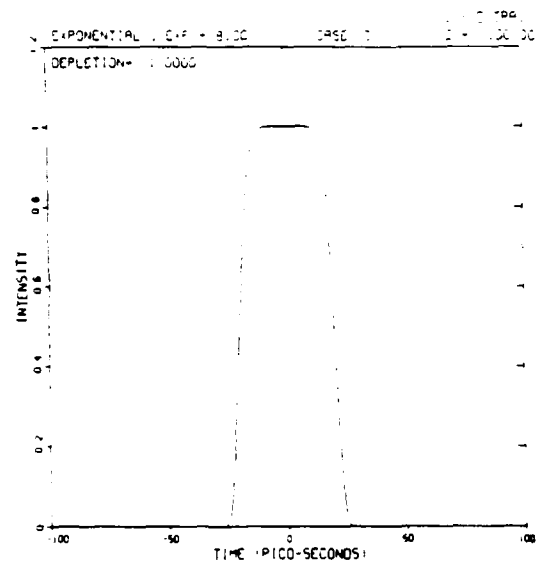


PLT2.DAT (Example A2)

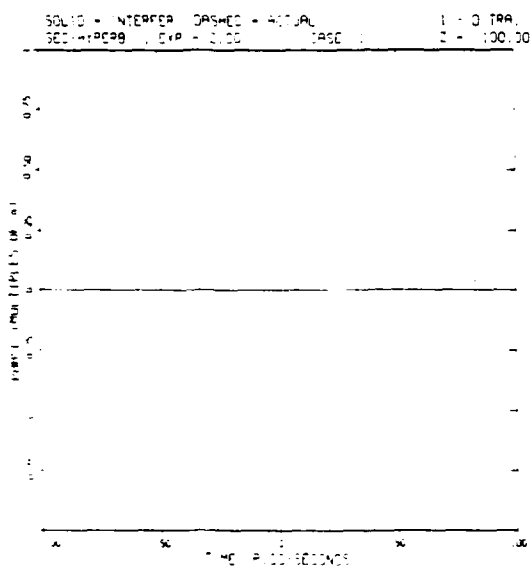
PARAM, PUMP: INTENSITY



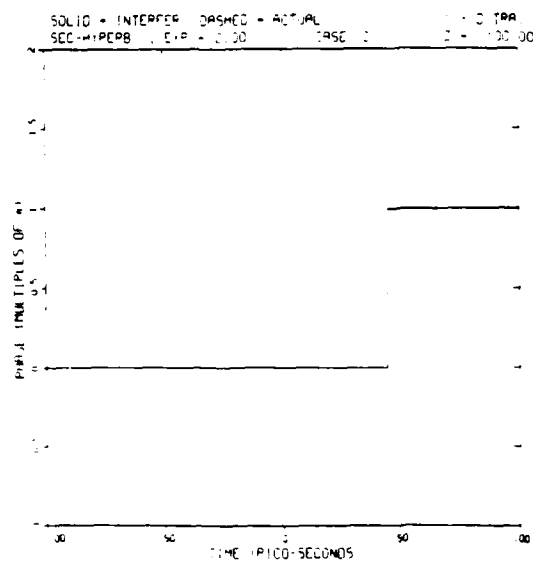
PARAM, PUMP: INTENSITY



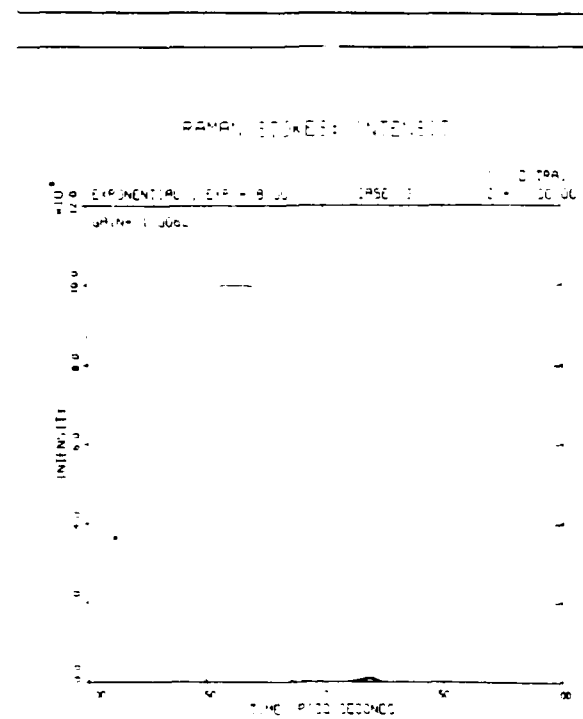
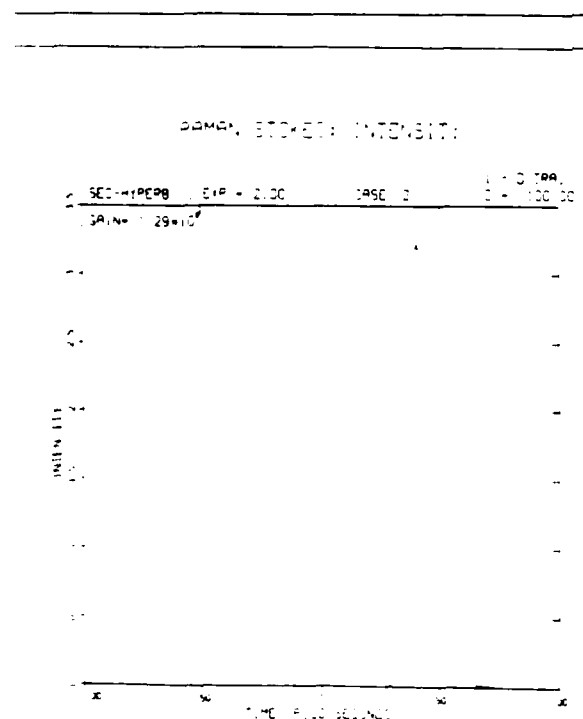
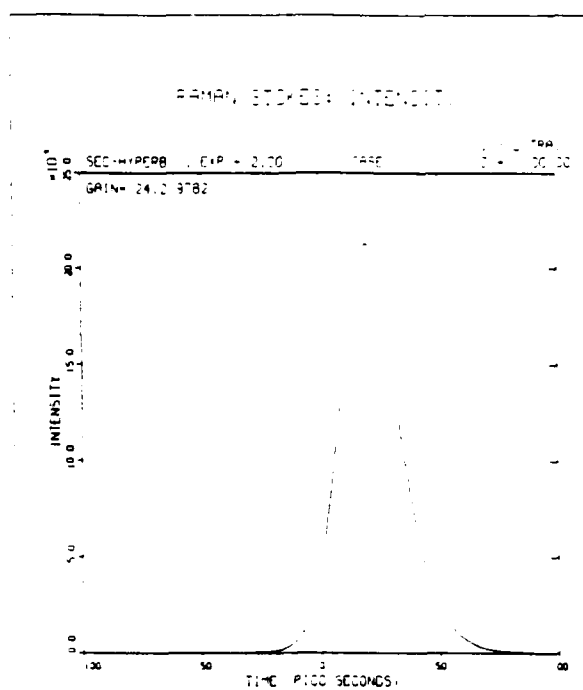
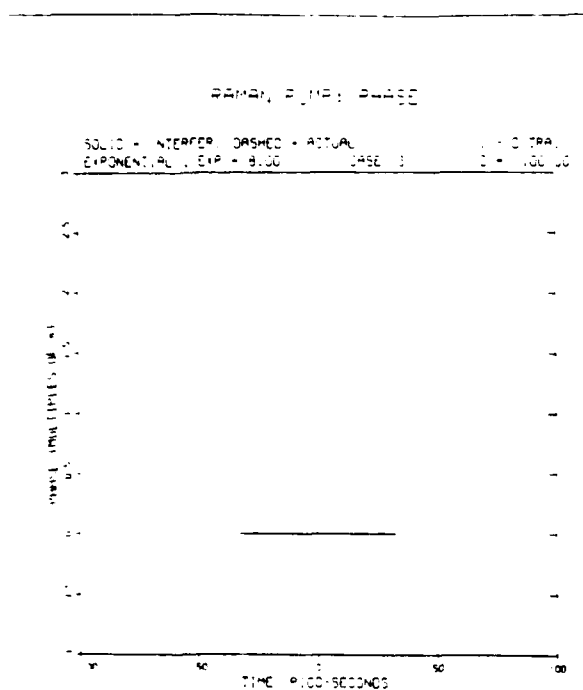
PARAM, PUMP: PHASE



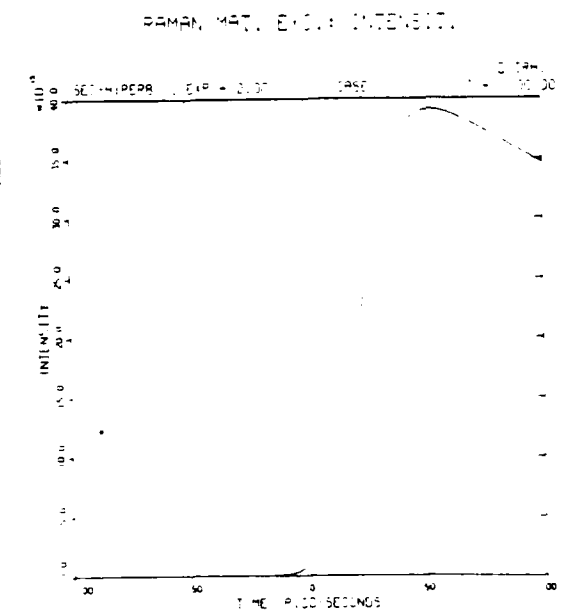
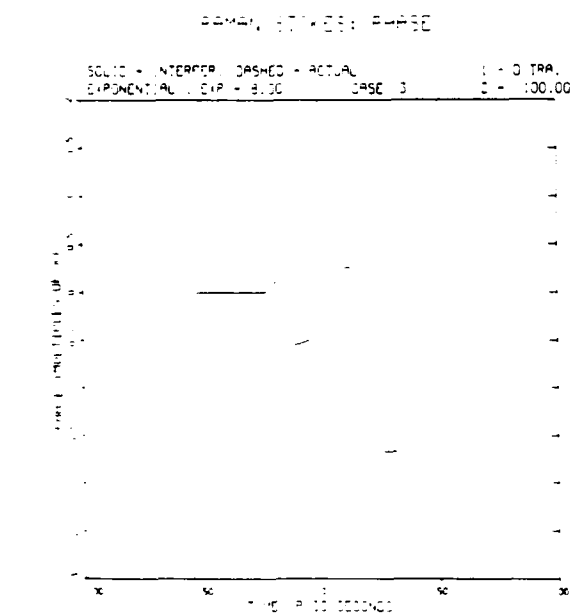
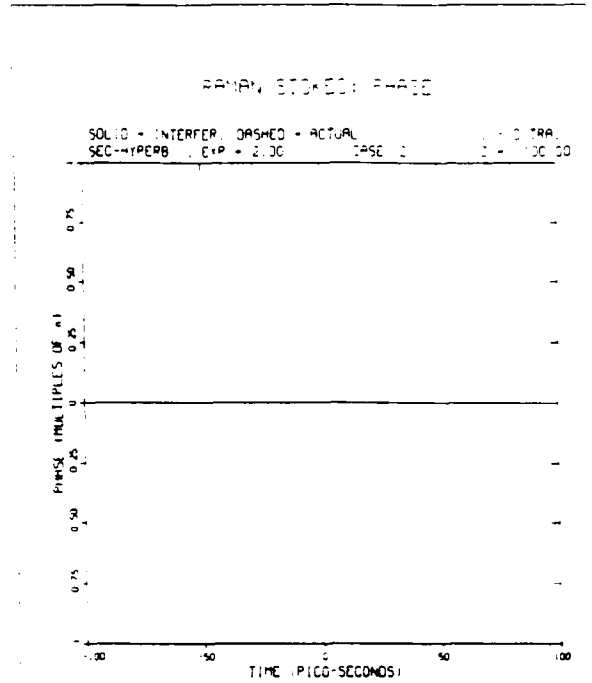
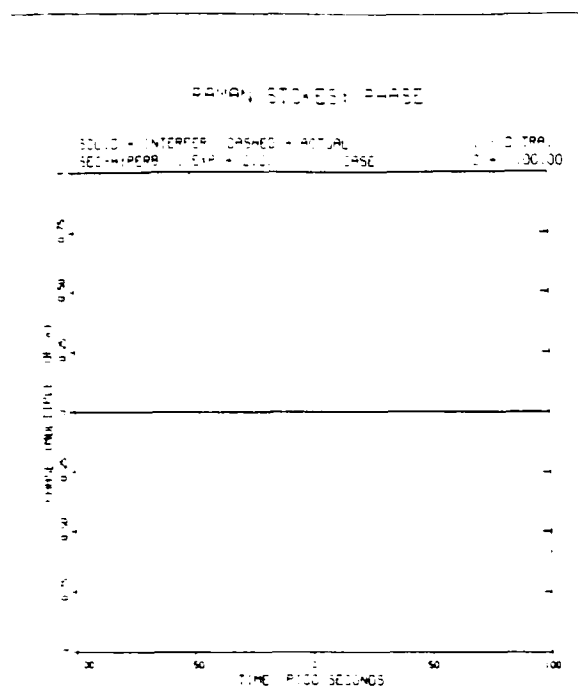
PARAM, PUMP: PHASE



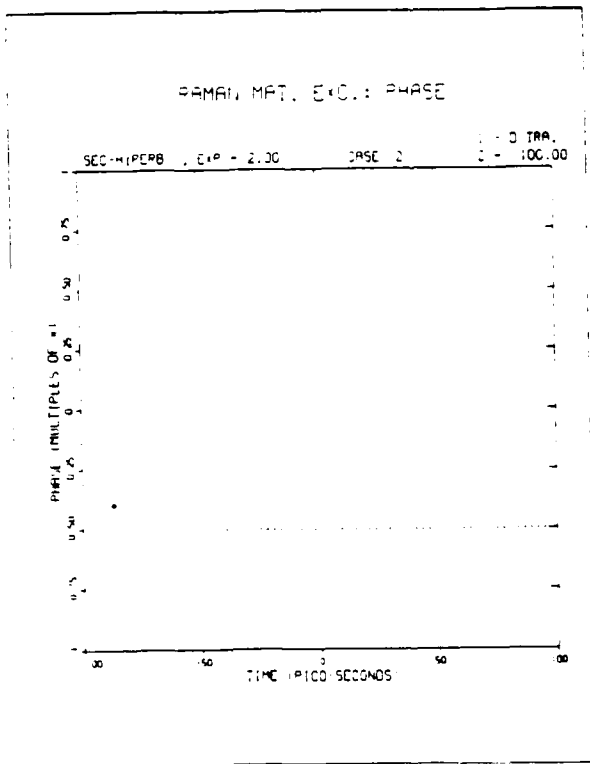
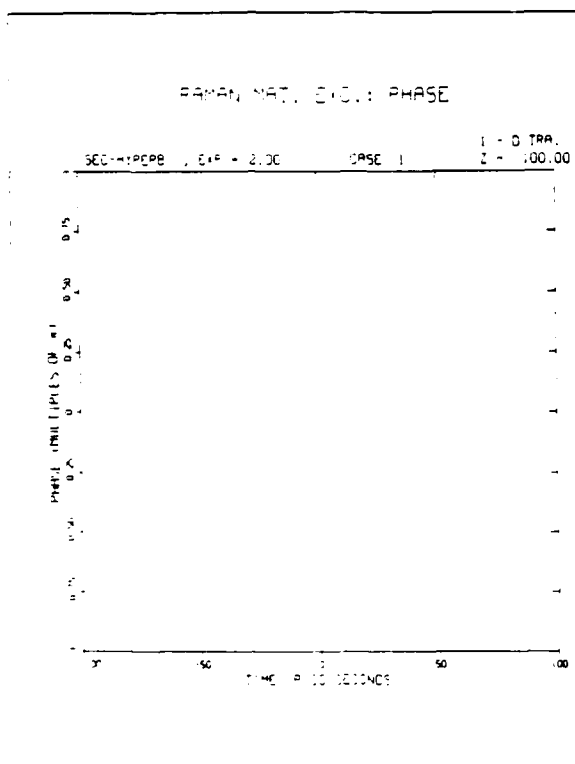
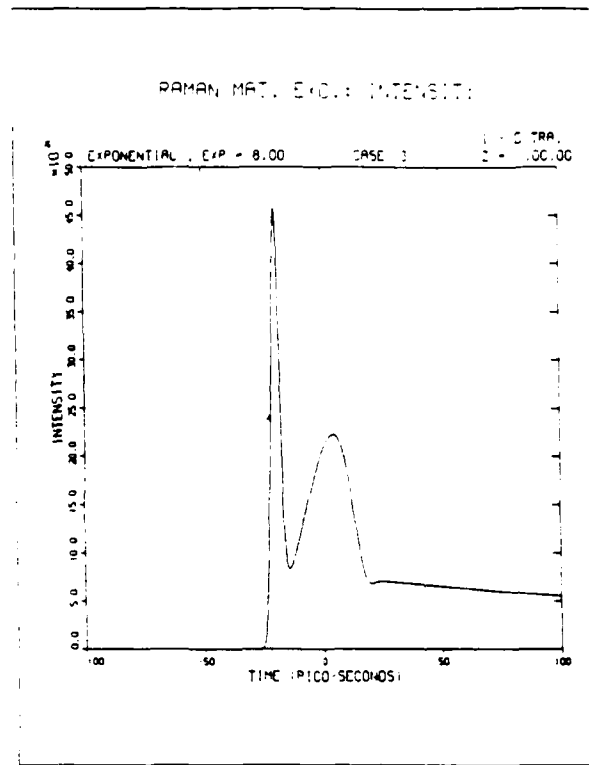
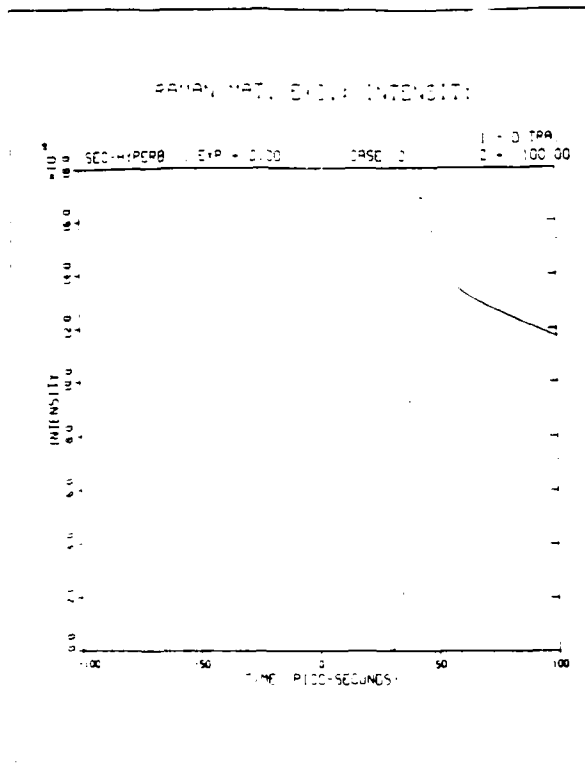
PLT2.DAT (Example A2)



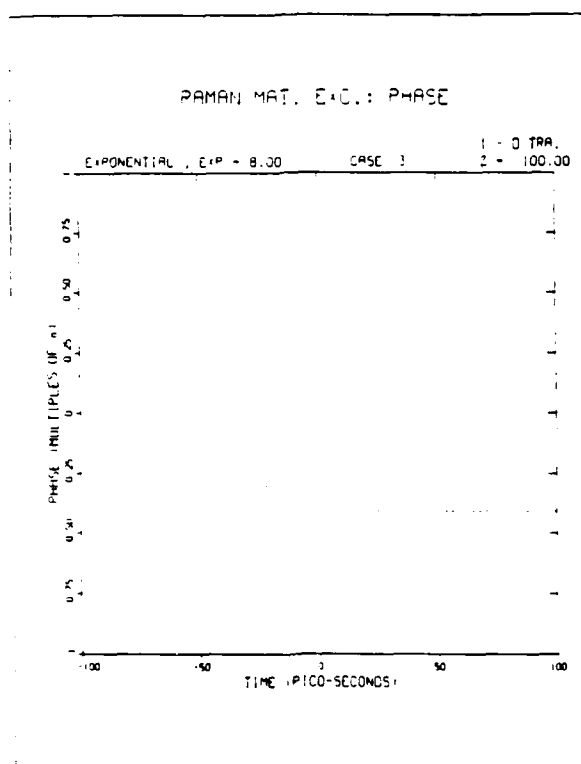
PLT2.DAT (Example A2)



PLT2.DAT (Example A2)



PLT2.DAT (Example A2)



APPENDIX B 1-D Stationary Limit; Examples

Two examples are appended to illustrate code operation in the stationary limit. The illustration features the batch job command files, the input data files, the output CPR-files and the resulting output. The first example is a simple run that features a chirped input signal and fast Fourier transforms of the fields. The second example illustrates several cases of multiple aberrated beam interaction in one run of the programs.

EXAMPLE B1

X1J.JOB

```
AUDIT.  
FETCH, DN=NRAM,TEXT='NR1J.DAT'.  
ACCESS, DN=XR1J.  
XR1J.  
DISPOSE, DN=ERRM,DF=BB,WAIT,TEXT='X1J.MSG.'.  
ACCESS, DN=DISLIB,ID=DISSPLA,OWN=LIBRARY.  
ACCESS, DN=INTLIB,ID=DISSPLA,OWN=LIBRARY.  
ACCESS, DN=DVSD,ID=DISSPLA,OWN=LIBRARY.  
ACCESS, DN=XP1J.  
FETCH, DN=NPRAM1,TEXT='NP1J.DAT'.  
XP1J.  
AUDIT.  
DISPOSE, DN=META,DF=BB,WAIT,TEXT='PLT2.DAT'.  
DISPOSE, DN=EPRM,DF=BB,WAIT,TEXT='X1J.MSG.'.  
DISPOSE, DN=DISOUT,DF=BB,WAIT,TEXT='X1J.DSP.'.  
EXIT.
```

NR1J.DAT

\$NAML

RIST=1.0E-12,
 PHL(1)=3.14,
 PHL(2)=3.14,
 PHST=3.14,
 ICOND=4,
 ZFINAL=40.0,
 ZKEEP=10.0,
 GAIN=0.4,

\$

NAMLIST/NAML/NPUMP, YM, TM, ZINT, RKP, RKS, YOFF, TOFF, YWIDTH, TWIDTH,
 1 YOST, TOST, YWST, TWST, RINT, RIST, RAMASM, RALASM, NHYP, PHL, PHST, TOC,
 2 ITYPE, RTYPE, RABAMP, RDSLIM, ICOND, ZSTEP, ZFINAL, ZKEEP, NMAX, TTWO, GAIN

NP1J.DAT

\$FLDATE

DONYET=1,
 MONTH=03,
 DAY=28,
 YEAR=88,
 IPART=2,
 NEDN=1,

\$

\$CONDAT

LPRMT(1)=1,
 LPRMT(2)=1,
 LPRMT(3)=1,
 LPRMT(4)=1,
 NSEC=1,
 CSEC(1,1)=(1.0,2.0),
 CSEC(2,1)=(1.0,2.0),
 CSEC(4,1)=(1.0,2.0),
 CSEC(5,1)=(1.0,2.0),
 CSEC(7,1)=(1.0,2.0),
 CSEC(8,1)=(1.0,2.0),
 CSEC(10,1)=(1.0,2.0),
 CSEC(11,1)=(1.0,2.0),
 CSEC(13,1)=(1.0,2.0),
 CSEC(14,1)=(1.0,2.0),
 CSEC(16,1)=(1.0,2.0),
 CSEC(17,1)=(1.0,2.0),

\$

\$ZPLOT

KZ(1)=1,
 KZ(2)=2,
 KZ(3)=3,
 KZ(4)=4,
 KZ(5)=5,

\$

X1J.CPR

```

16 30 55 8536 0 0000 - CSP
16 30 55 8539 0 0000 CSP
16 30 55 8541 0 0000 CSP
16 30 55 8544 0 0000 CSP
16 30 55 8548 0 0001 CSP
16 30 55 8549 0 0001 CSP
16 30 55 8551 0 0001 CSP
16 30 55 8554 0 0001 CSP
16 30 55 8557 0 0001 CSP
16 30 55 8559 0 0001 CSP
16 30 55 8562 0 0001 CSP
16 30 55 8564 0 0001 CSP
16 30 56 0923 0 0002 CSP
16 30 56 0928 0 0002 CSP
16 30 56 0950 0 0002 CSP
16 30 56 0954 0 0002 CSP
16 30 56 0959 0 0002 CSP
16 30 56 1127 0 0002 CSP
16 30 56 1482 0 0013 CSP
16 30 58 6049 0 1023 USER
16 30 59 0474 0 1051 USER
16 31 03 6942 0 1872 USER
16 31 03 6946 0 1873 USER
16 31 03 6950 0 1874 USER
16 31 03 7015 0 1875 CSP
16 31 09 1017 0 1876 SCP
16 31 09 1020 0 1876 SCP
16 31 09 1022 0 1876 SCP
16 31 12 8095 0 1876 SCP
16 31 12 9878 0 1878 CSP
16 31 13 0703 0 1878 PDM
16 31 13 0708 0 1878 PDM
16 31 13 1357 0 1880 CSP
16 31 19 9477 5 8062 PDM
16 31 19 9472 5 8062 PDM
16 31 19 9489 5 8062 USER
16 31 19 9540 5 8062 CSP
16 31 31 1975 5 8064 SCP
16 31 31 1978 5 8064 SCP
16 31 31 1982 5 8064 SCP
16 31 39 1423 5 8067 CSP
16 31 39 6518 5 8068 PDM
16 31 39 6520 5 8068 PDM
16 31 39 6541 5 8071 CSP
16 31 40 2745 5 8072 PDM
16 31 40 2748 5 8072 PDM
16 31 40 2768 5 8075 CSP
16 31 40 5113 5 8075 PDM
16 31 40 5116 5 8075 PDM
16 31 40 5132 5 8077 CSP
16 31 40 7760 5 8077 PDM
16 31 40 7763 5 8077 PDM
16 31 40 7778 5 8078 CSP
16 31 45 0963 5 8079 SCP
16 31 45 0966 5 8079 SCP
16 31 45 0988 5 8079 SCP
16 31 48 2089 5 8079 SCP
16 31 48 4865 5 8081 CSP
16 31 48 9837 5 8117 PDM
16 31 48 9840 5 8117 PDM

```

WELCOME TO THE NRL CRAY XMP

We are attempting to cleanup the library of CRAY archive tapes. These cleanup runs will be made on Tuesday and Wednesday mornings between 2:00 and 7:00 AM. During these times, there will be no recall of off line data sets. If you plan on running jobs during these hours, please insure that required files are on line.

CRAY X MP SERIAL 415 65 NAVAL RESEARCH LABORATORY 03 28 88

CRAY OPERATING SYSTEM COS 1 15 ASSEMBLY DATE 01 04 88

```

JOB JN=X1J.MFL-511000.US-DEFER.
ACCOUNT.AC-.US-.UPW-.APW-
AC213 ** TOTAL BUDGET WARNING LEVEL REACHED FOR THIS ACCOUNT NUMBER
AUDIT
AU003 63 DATASETS. 95112 BLOCKS. 48671617 WORDS
AU003 6 DATASETS. 1535 BLOCKS. 784380 WORDS ONLINE
AU003 57 DATASETS. 93577 BLOCKS. 47887237 WORDS OFFLINE
FETCH. DN-NRAM.TEXT- NR1J.DAT
VAX TO CRAY: %SYSTEM-S-NORMAL. normal successful completion
VAX TO CRAY: FILE-$1SDUAL07:[HILFER FR2]NR1J.DAT:18
VAX TO CRAY: 488 BYTES TRANSFERRED
SS004 - DATASET RECEIVED FROM FRONT END
ACCESS. DN-XR1J.
PD000 - PDM - XR1J ID - ED - 4 OWN - HILFER
PD000 - ACCESS COMPLETE
XR1J
PD000 - PDM - F1J032888 ID - ED - 1 OWN - HILFER
PD000 - SAVE COMPLETE
UT003 - EXIT CALLED BY RAM2DIC
DISPOSE. DN-ERRM.DF-BB.WAIT.TEXT- X1J.MSG.
CRAY TO VAX: %RMS-S-NORMAL. normal successful completion
CRAY TO VAX: FILE-$1SDUAL07:[HILFER FR2]X1J.MSG:3
CRAY TO VAX: 51004 BYTES TRANSFERRED
ACCESS. DN-DISLIB.ID-DISSPLA.OWN-LIBRARY.
PD000 - PDM - DISLIB ID - DISSPLA ED - 1 OWN - LIBRARY
PD000 - ACCESS COMPLETE
ACCESS. DN-INTLIB.ID-DISSPLA.OWN-LIBRARY.
PD000 - PDM - INTLIB ID - DISSPLA ED - 1 OWN - LIBRARY
PD000 - ACCESS COMPLETE
ACCESS. DN-DVSD.ID-DISSPLA.OWN-LIBRARY.
PD000 - PDM - DVSD ID - DISSPLA ED - 1 OWN - LIBRARY
PD000 - ACCESS COMPLETE
ACCESS. DN-XP1J.
PD000 - PDM - XP1J ID - ED - 5 OWN - HILFER
PD000 - ACCESS COMPLETE
FETCH. DN-NPRAM1.TEXT- NP1J.DAT
VAX TO CRAY: %SYSTEM-S-NORMAL. normal successful completion
VAX TO CRAY: FILE-$1SDUAL07:[HILFER FR2]NP1J.DAT:19
VAX TO CRAY: 912 BYTES TRANSFERRED
SS004 - DATASET RECEIVED FROM FRONT END
XP1J
PD000 - PDM - F1J032888 ID - ED - 1 OWN - HILFER
PD000 - LOCAL DATASET NAME ALREADY IS IN USE

```

X1J.CPR

16 32 00 8542	15 0458	USER	UT003	EXIT CALLED BY	PRAM1CD
16 32 01 0706	15 0462	USER	AUDIT		
16 32 05 8101	15 1204	USER	AU003	64 DATASETS	95257 BLOCKS
16 32 05 8125	15 1205	USER	AU003	7 DATASETS	1680 BLOCKS
16 32 05 8120	15 1206	USER	AU003	57 DATASETS	97577 BLOCKS
16 32 05 8198	15 1296	CSP	DISPOSE	DN-META.DF-BB.WAIT.TEXT	FLI2 DAT
16 32 05 8106	15 1298	SCP	CRAY TO VAX	4RMS-S-NORMAL	normal successful completion
16 32 05 7109	15 1298	SCP	CRAY TO VAX	FILE-\$1SDUAL07:[HILFER.FR2]	FLI2 DAT.3
16 32 05 7111	15 1298	SCP	CRAY TO VAX	391680 BYTES TRANSFERRED	
16 32 01 9142	15 1209	CSP	DISPOSE	DN-EPRM.DF-BB.WAIT.TEXT	X1J MSG
16 32 06 0567	15 1300	SCP	CRAY TO VAX	4RMS-S-NORMAL	normal successful completion
16 32 06 0560	15 1300	SCP	CRAY TO VAX	FILE-\$1SDUAL07:[HILFER.FR2]	X1J MSG.4
16 32 06 0562	15 1300	SCP	CRAY TO VAX	24879 BYTES TRANSFERRED	
16 32 07 6037	15 1301	CSP	DISPOSE	DN-DISOUT.DF-BB.WAIT.TEXT	X1J DSP
16 32 03 0014	15 1303	SCP	CRAY TO VAX	4RMS S-NORMAL	normal successful completion
16 32 03 0017	15 1303	SCP	CRAY TO VAX	FILE-\$1SDUAL07:[HILFER.FR2]	X1J DSP.2
16 32 03 0019	15 1303	SCP	CRAY TO VAX	1004 BYTES TRANSFERRED	
16 32 04 7496	15 1303	CSP	EXIT		
16 32 04 7510	15 1304	CSP	END OF JOB		
16 32 04 7513	15 1304	CSP			
16 32 04 7516	15 1304	CSP			
16 32 05 2206	15 1305	USER	JOB NAME	X1J	
16 32 05 2209	15 1305	USER	USER NUMBER	HILFER	
16 32 05 2413	15 1305	USER	JOB SEQUENCE NUMBER	34182	
16 32 05 2418	15 1305	USER			
16 32 05 2422	15 1305	USER	TIME EXECUTING IN CPU	0000:00:15	1305
16 32 05 2426	15 1306	USER	TIME WAITING TO EXECUTE	0000:01:10	4314
16 32 05 2429	15 1306	USER	TIME WAITING FOR I O	0000:00:13	2222
16 32 05 2432	15 1306	USER	TIME WAITING IN INPUT QUEUE	0000:00:00	0021
16 32 05 2435	15 1306	USER	MEMORY CPU TIME (MWDS*SEC)	2	74438
16 32 05 2438	15 1306	USER	MEMORY I O WAIT TIME (MWDS*SEC)	1	72518
16 32 05 2442	15 1306	USER	MINIMUM JOB SIZE (WORDS)	44544	
16 32 05 2445	15 1306	USER	MAXIMUM JOB SIZE (WORDS)	228864	
16 32 05 2448	15 1307	USER	MINIMUM FL (WORDS)	40960	
16 32 05 2451	15 1307	USER	MAXIMUM FL (WORDS)	224256	
16 32 05 2454	15 1307	USER	MINIMUM JTA (WORDS)	3584	
16 32 05 2457	15 1307	USER	MAXIMUM JTA (WORDS)	5632	
16 32 05 2460	15 1307	USER	DISK SECTORS MOVED	3305	
16 32 05 2463	15 1307	USER	FSS SECTORS MOVED	0	
16 32 05 2466	15 1307	USER	USER I O REQUESTS	915	
16 32 05 2469	15 1307	USER	USER I O SUSPENSIONS	1308	
16 32 05 2473	15 1307	USER	OPEN CALLS	35	
16 32 05 2476	15 1307	USER	CLOSE CALLS	34	
16 32 05 2479	15 1307	USER	MEMORY RESIDENT DATASETS	0	
16 32 05 2482	15 1307	USER	TEMPORARY DATASET SECTORS USED	145	
16 32 05 2485	15 1308	USER	PERMANENT DATASET SECTORS ACCESSED	2810	
16 32 05 2488	15 1308	USER	PERMANENT DATASET SECTORS SAVED	145	
16 32 05 2491	15 1308	USER	SECTORS RECEIVED FROM FRONT END	2	
16 32 05 2494	15 1308	USER	SECTORS QUEUED TO FRONT END	144	
16 32 05 5314	15 1380	USER			
16 32 05 5317	15 1380	USER			
16 32 05 5320	15 1380	USER			
16 32 05 5323	15 1381	USER			
16 32 05 5327	15 1382	USER			
16 32 05 5330	15 1383	USER			
16 32 05 5333	15 1384	USER			
16 32 05 5337	15 1385	USER			
16 32 05 5340	15 1386	USER			
16 32 05 5344	15 1387	USER			
16 32 05 5347	15 1388	USER			
16 32 05 5351	15 1389	USER			
16 32 05 5355	15 1390	USER			
16 32 05 5412	15 1391	USER			
16 32 05 5414	15 1391	USER			
16 32 05 5416	15 1391	USER			

*** COST TABLE FOR THIS JOB ***					
JOBNAME	USER IDENT	MON MAR 28, 1988	X1J	HILFER	
BEGAN EXECUTION			16:30:55	HOURS	3
AT A PRIORITY OF					
AND JOB CLASS OF					
15 136789	SECONDS OF CPU TIME	2 \$ 830 00	HR	\$	2 65
2 744818	MEMORY CPU (MWRD SEC)	2 \$ 84 00	HR	\$	0 06
1 728561	MEMORY I O (MWRD SEC)	2 \$ 84 00	HR	\$	0 04
0 003307	I O MEGASECTORS MOVED	2 \$ 84 00	EA	\$	0 28
0 000000	TAPE MOUNT(S)	2 \$ 5 00	EA	\$	0 00
TOTAL COST FOR THIS JOB				\$	3 03

PLT2.DAT (Example B1)

LIST OF INPUT PARAMETERS

```

ICOND - 4
NIMP - 8
NPAR - 1000
NPLUMP - 2
NT - 1
NT - 1024
GAIN - 0.4000
RIST - 1.00E+01
RKP - 1.18E+01
RKS - 1.9E+01
TTHO - 633.00
TOST - 0.0000
TWST - 0.1000
ZFINAL - 40.0000
ZINT - 20.0000
ZKEEP - 10.0000
ZSTEP - 0.03000

```

LIST OF INPUT PARAMETERS CONTD

RAAMP(1:8) -	0.0000	0.0000	0.0000	0.0000	0.0000
	0.0000	0.0000	0.0000		
ROSLIM(1:8) -	1.0000	1.0000	1.0000	1.0000	1.0000
	1.0000	1.0000	1.0000		
RINT(1:10) -	0.5500	0.5500	0.5500	0.5500	0.5500
	0.5500	0.5500	0.5500	0.5500	0.5500
YOFF(1:10) -	0.1400	-0.1400	0.3000	0.3000	0.0000
	0.0000	0.0000	0.0000	0.0000	0.0000
YM(1,2) -	-0.3000	0.3000			
YMODM -	0.1000	0.1000	0.1000	0.1000	0.1000
	0.1000	0.1000	0.1000	0.1000	0.1000

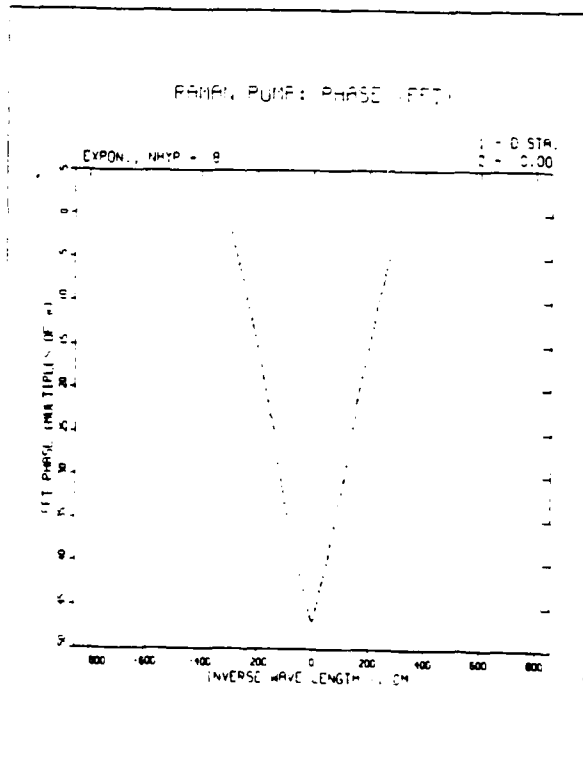
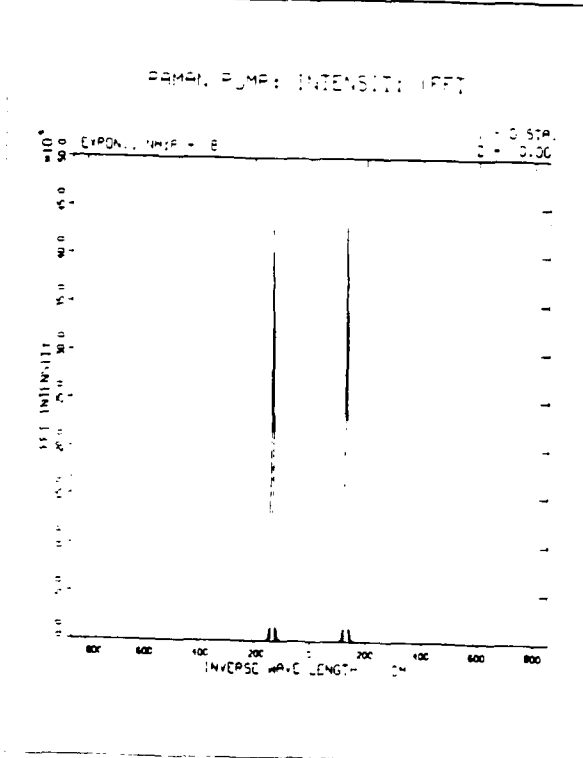
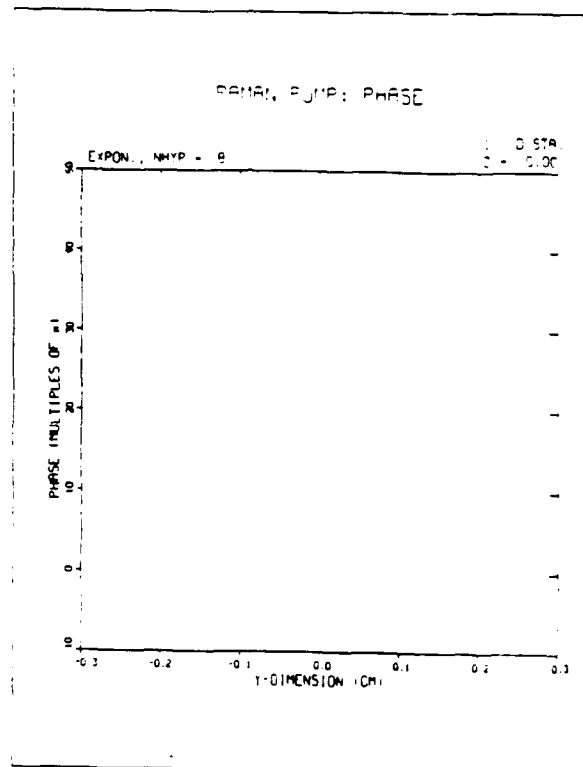
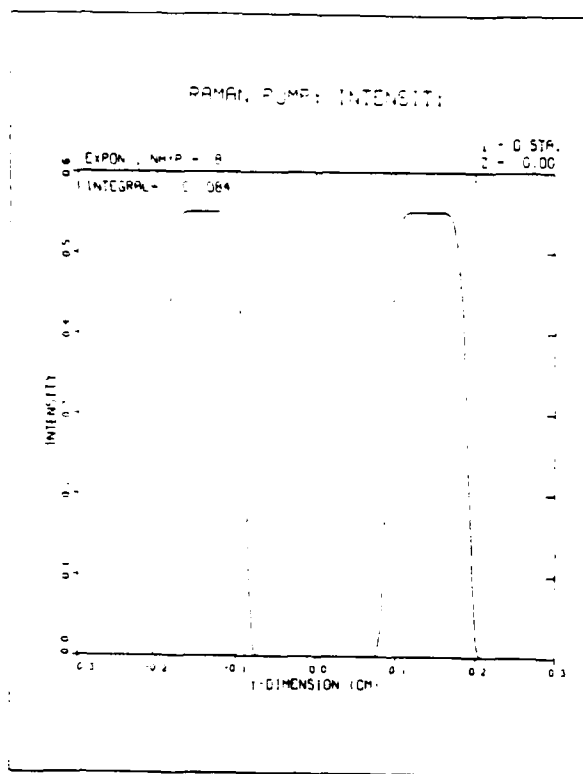
LIST OF INPUT PARAMETERS (CONT)

[illegible]

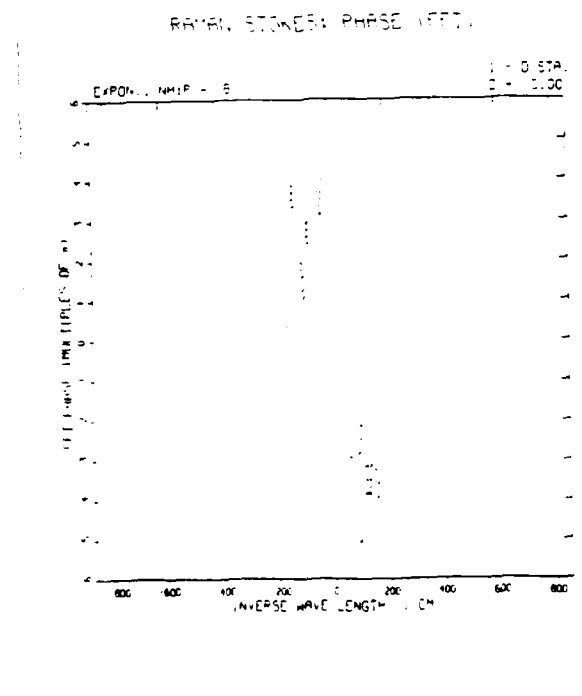
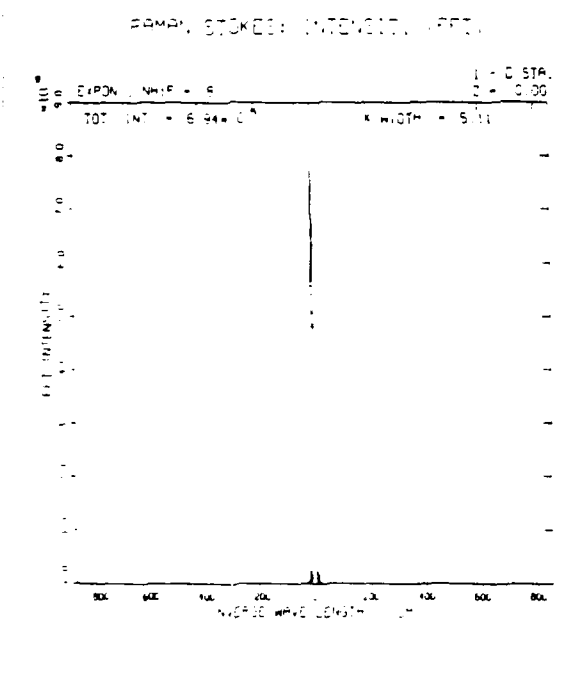
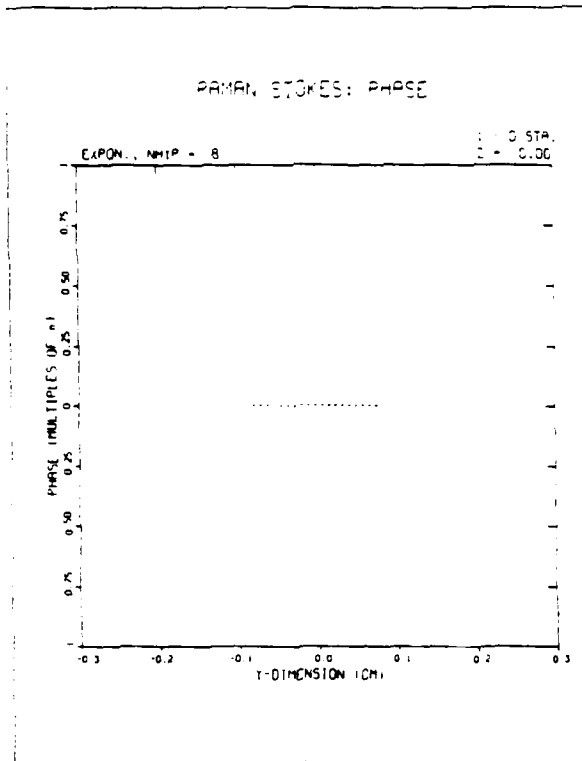
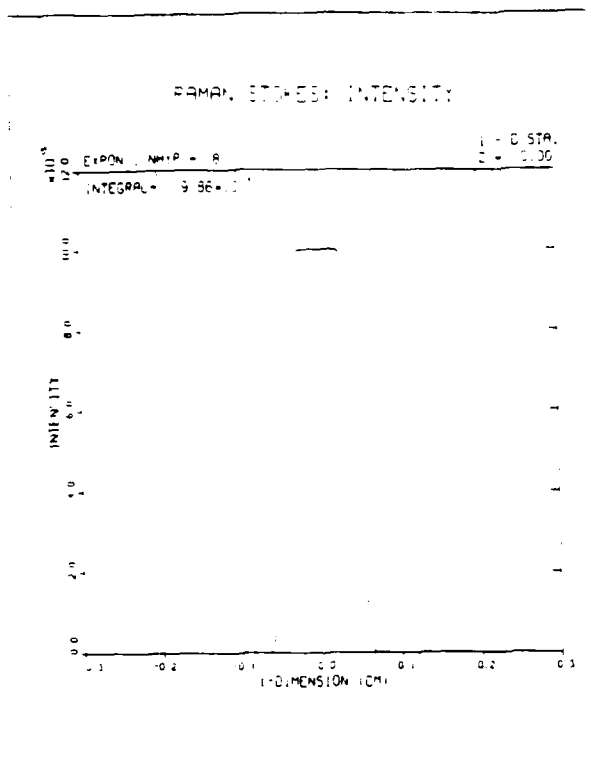
LIST OF OUTPUT PARAMETERS

PUMP	TOTAL INTENSITY	WIDTH
2	3.82×10^3	6.18
1	3.82×10^3	6.18

PLT2.DAT (Example B1)



PLT2.DAT (Example B1)



PLT2.DAT (Example B1)

RAMAN MAT. EXCL.: INTENSIT:

EXPON., NHYP = 8

1 - 0 STA.
2 - 0.00

RAMAN MAT. EXCL.: PHASE

EXPON., NHYP = 8

1 - 0 STA.
2 - 0.00

RAMAN MAT. EXCL.: INTENSIT: FFT

EXPON., NHYP = 8

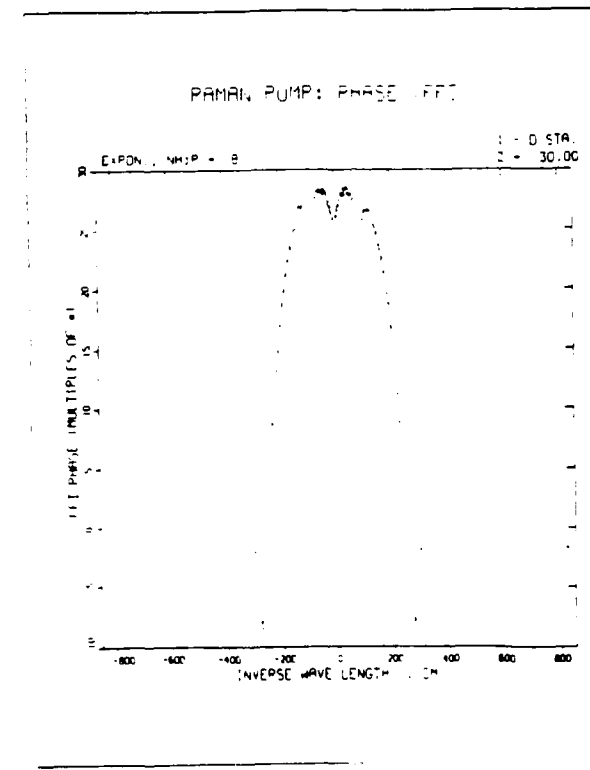
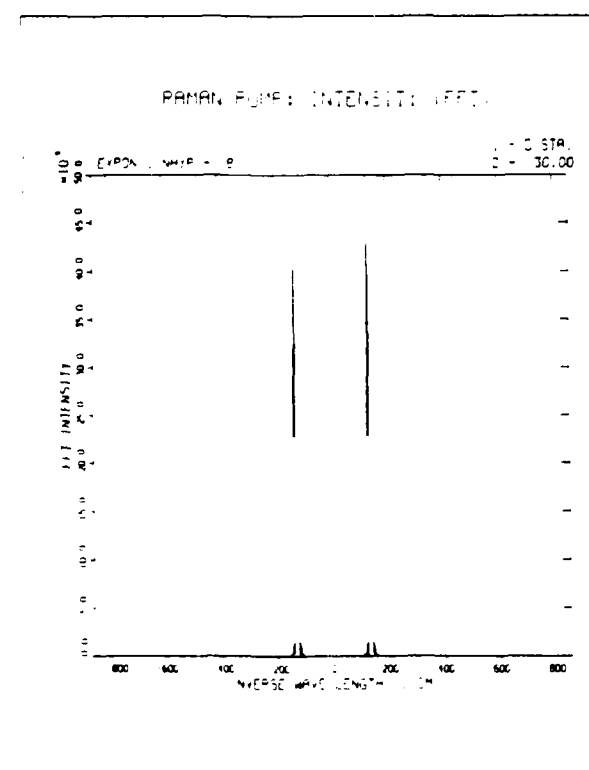
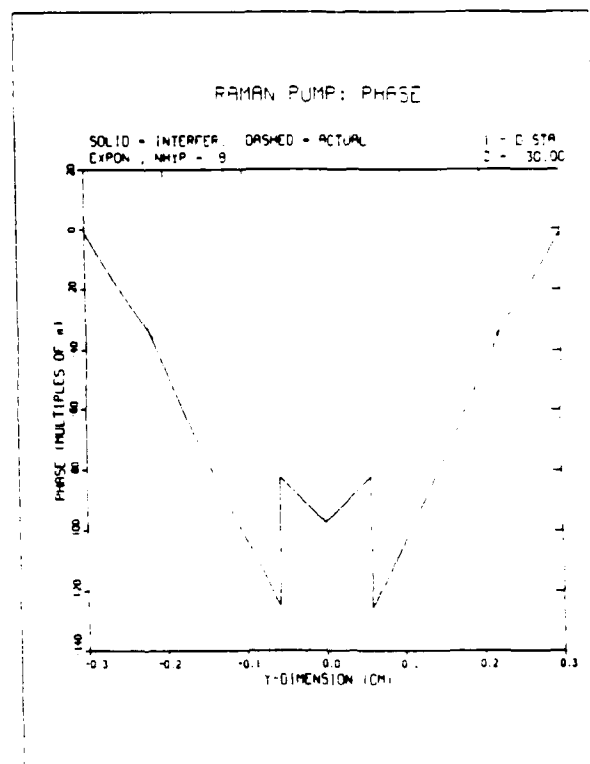
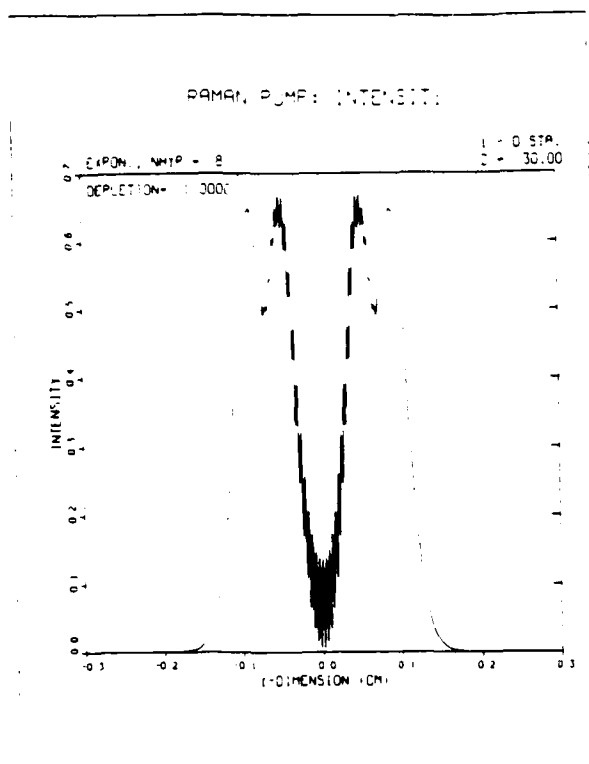
1 - 0 STA.
2 - 0.00

RAMAN MAT. EXCL.: PHASE: FFT

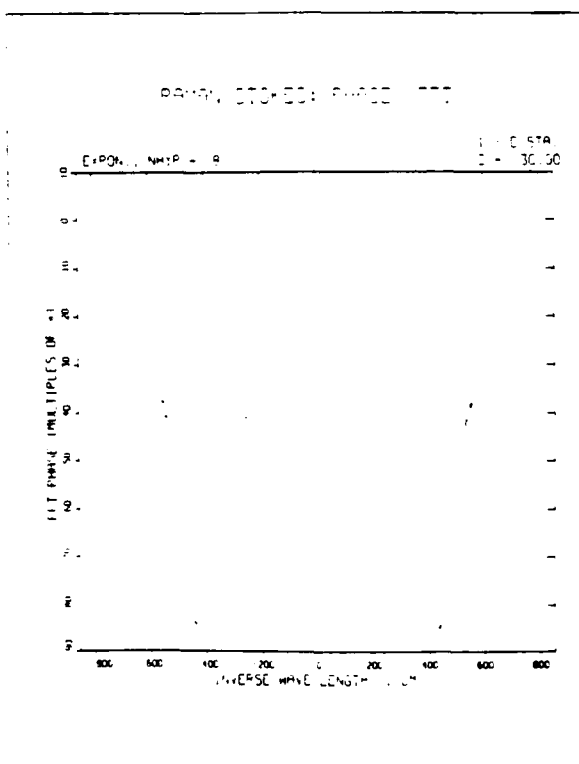
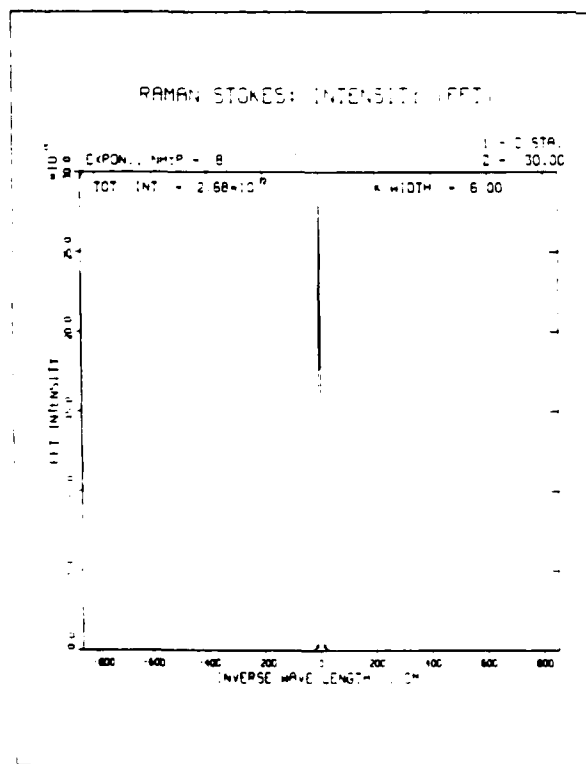
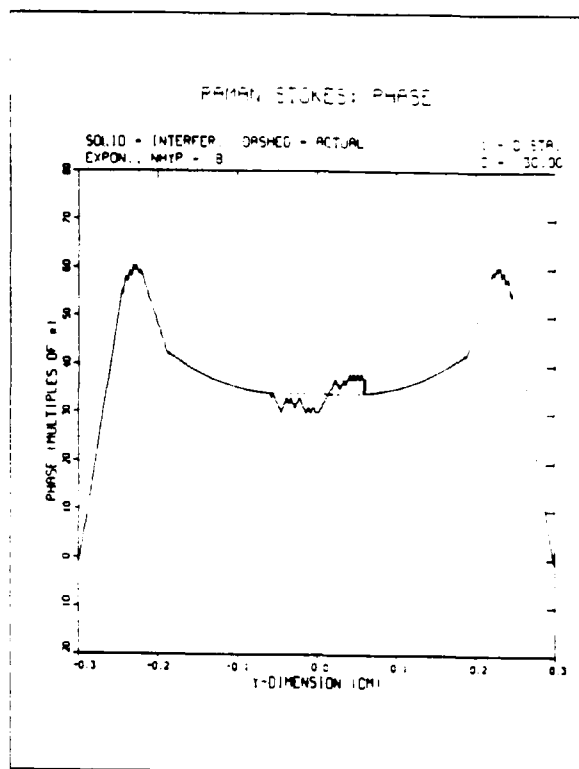
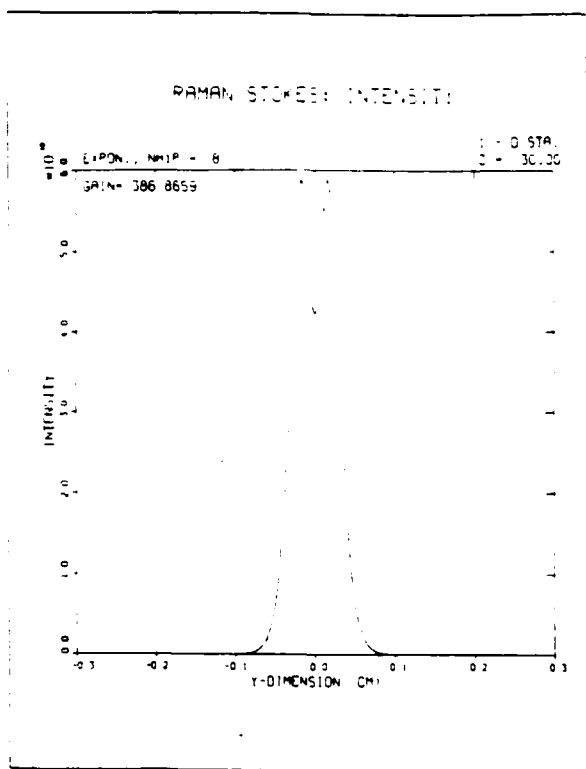
EXPON., NHYP = 8

1 - 0 STA.
2 - 0.00

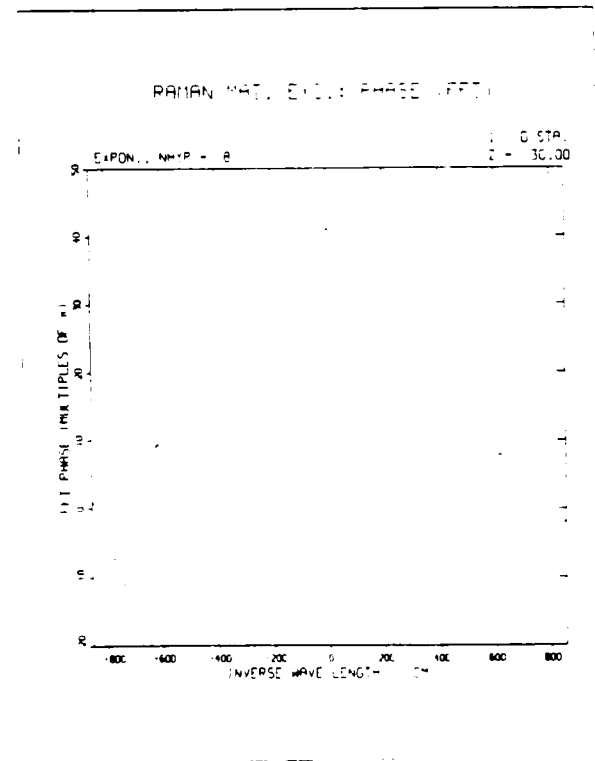
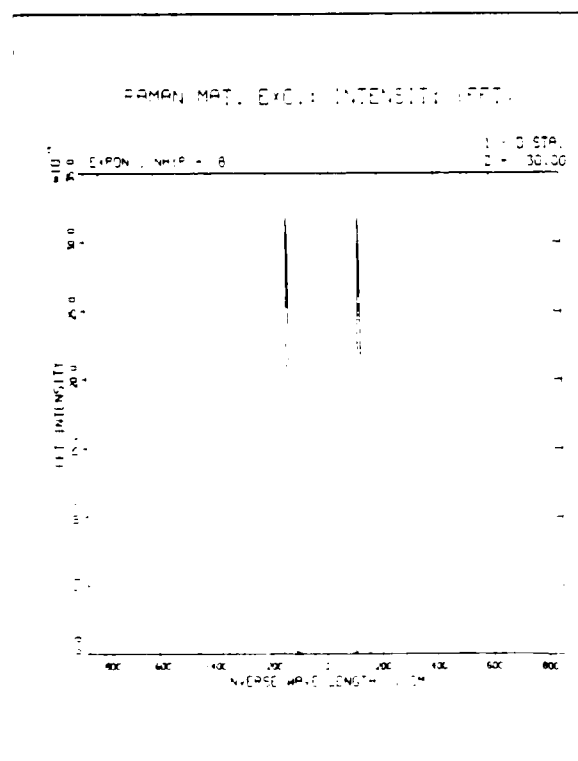
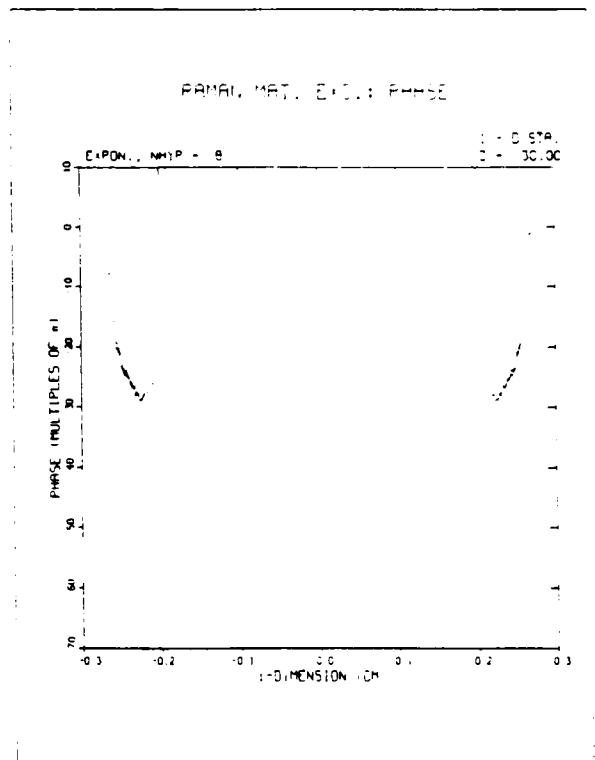
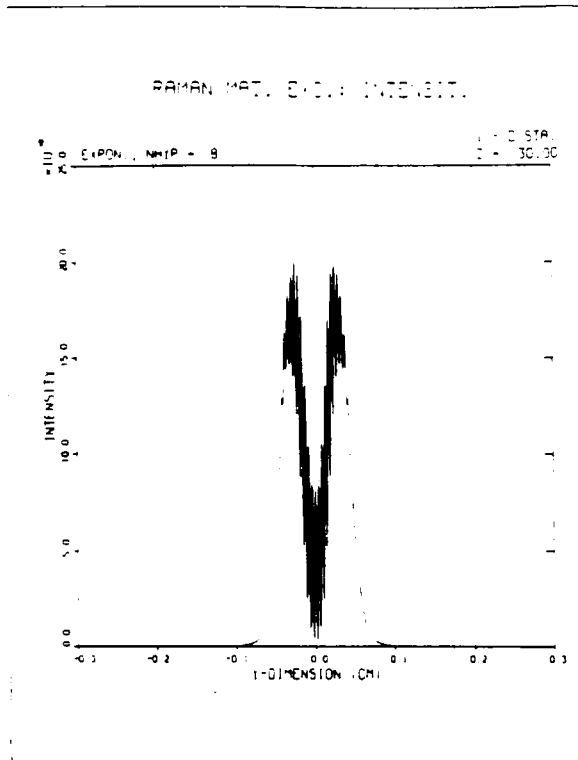
PLT2.DAT (Example B1)



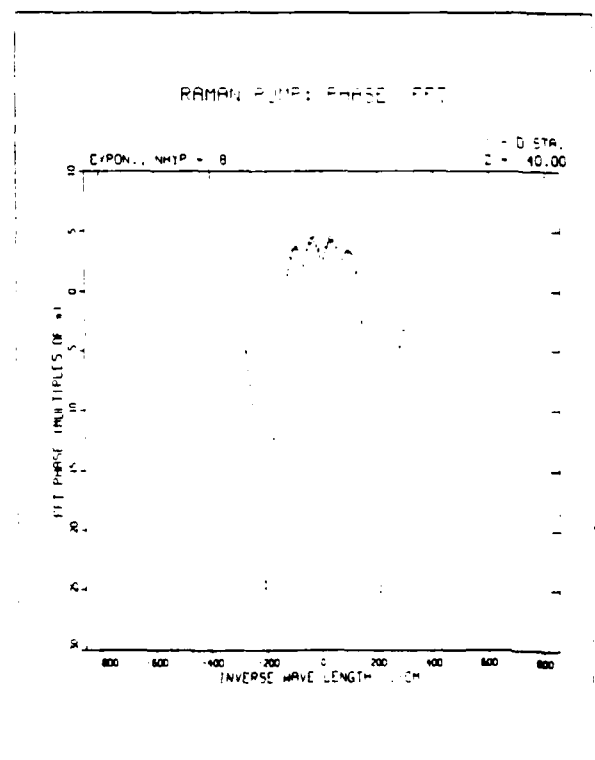
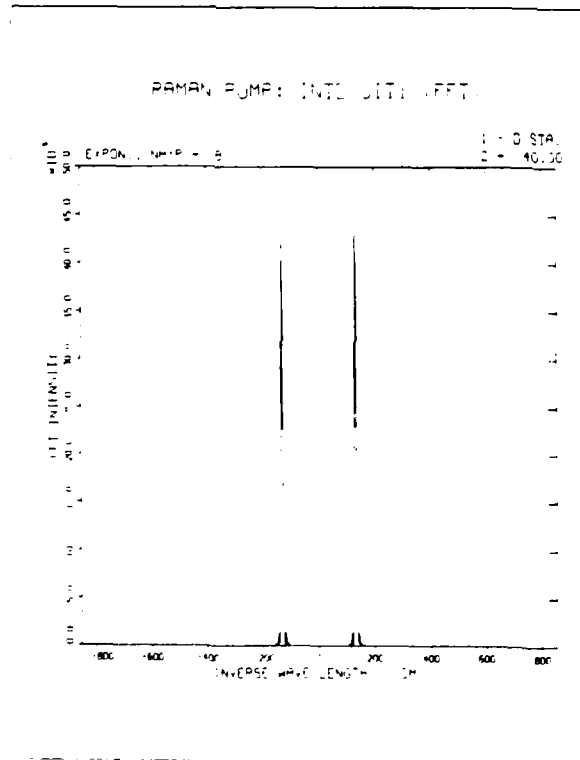
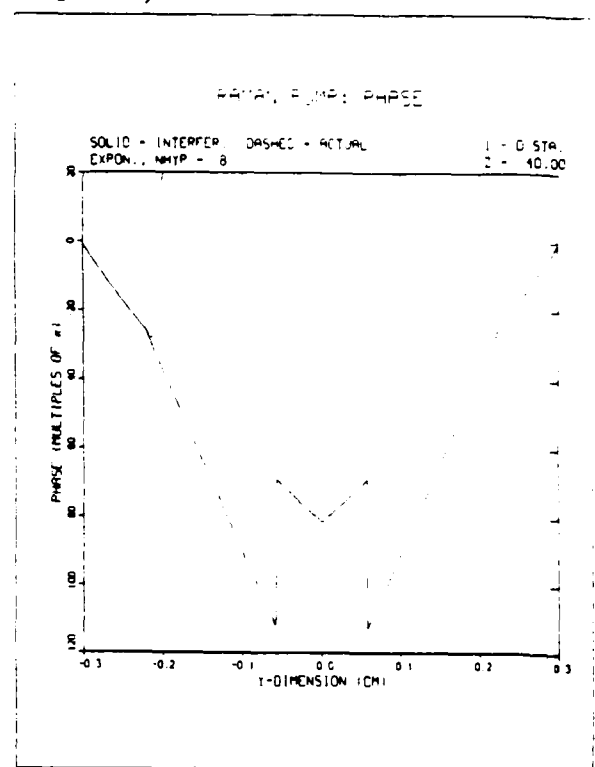
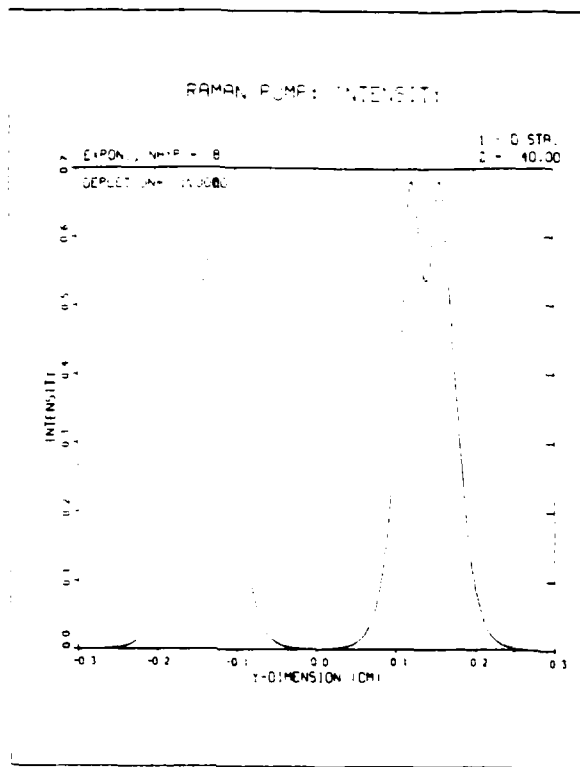
PLT2.DAT (Example B1)



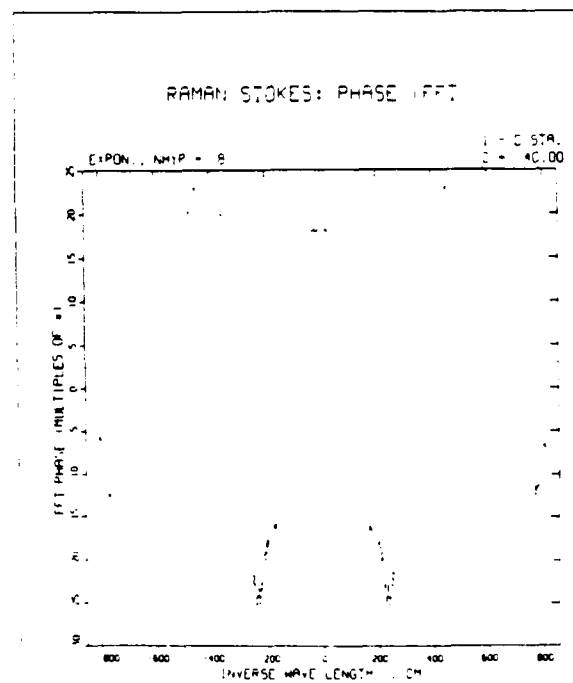
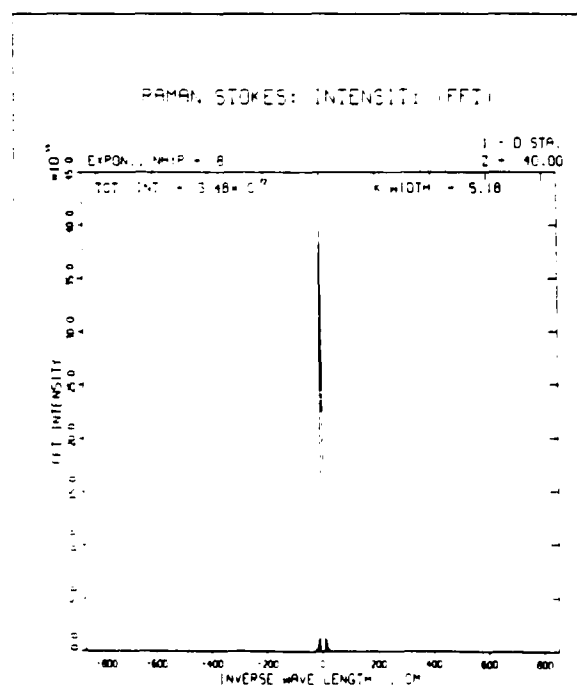
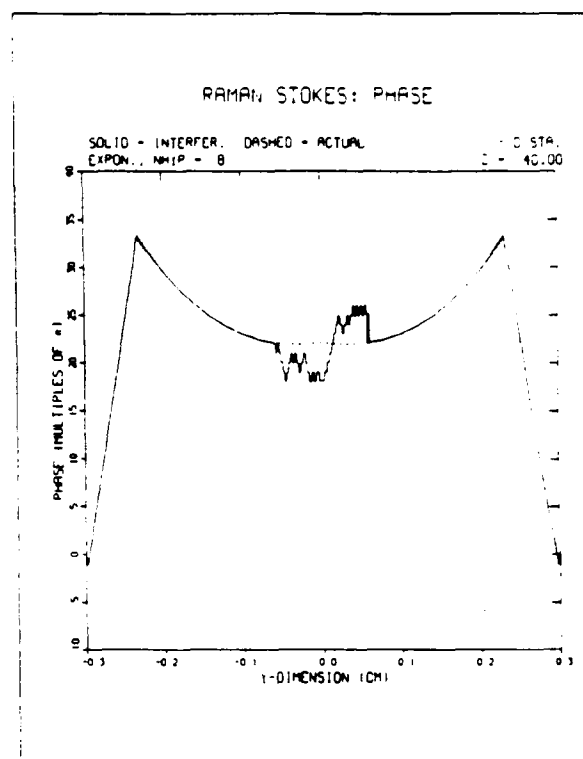
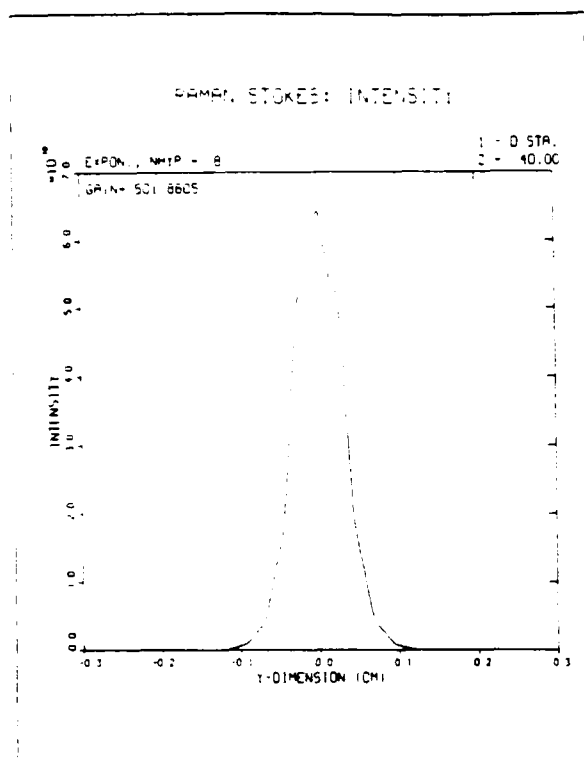
PLT2.DAT (Example B1)



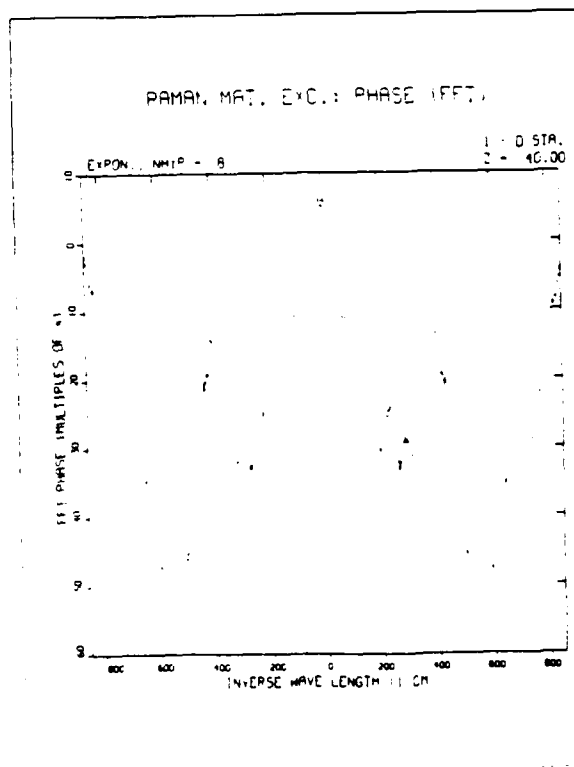
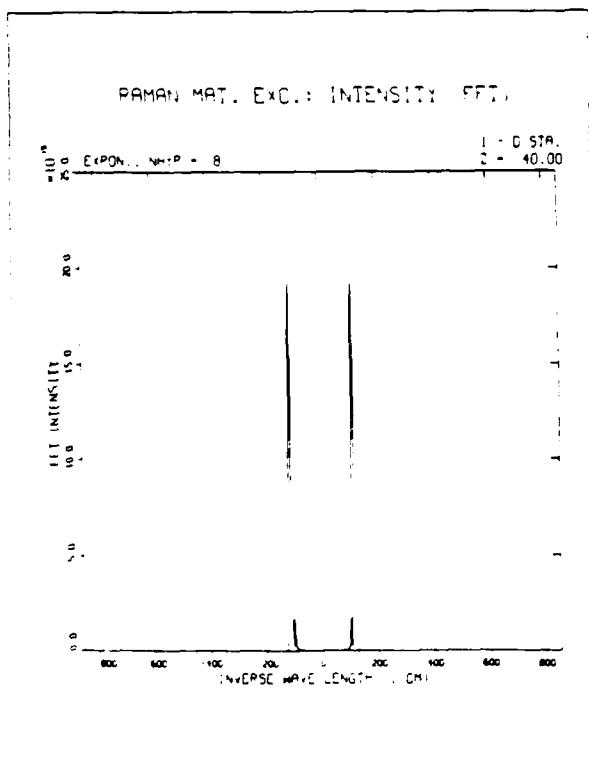
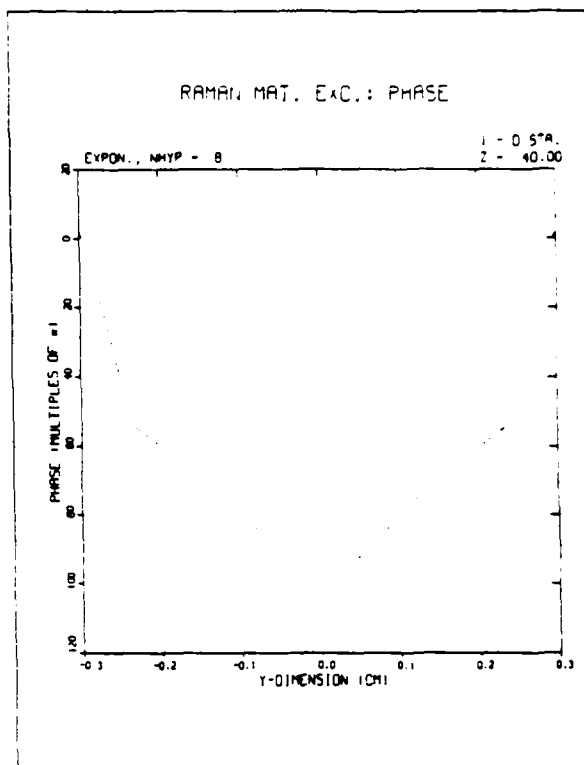
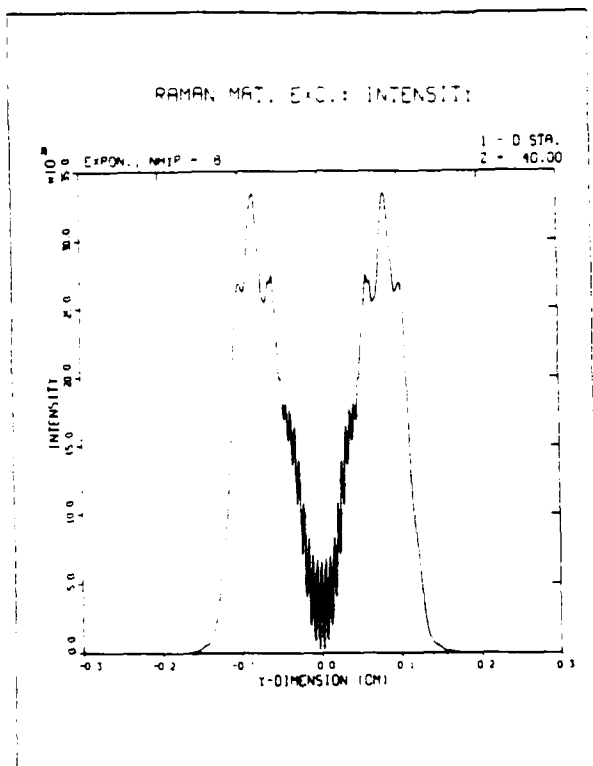
PLT2.DAT (Example B1)



PLT2.DAT (Example B1)



PLT2.DAT (Example B1)



EXAMPLE B2

PLT2.DAT (Example B2)

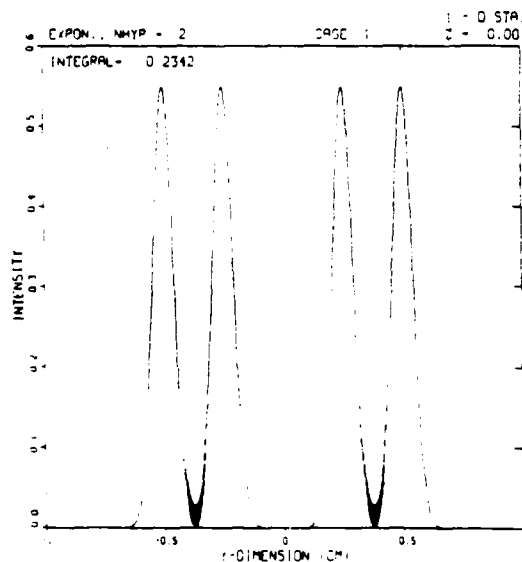
LIST OF INPUT PARAMETERS

ICOND = 1
 NHYP = 2
 NMAX = 4000
 NPUMP = 4
 NT = 3
 NY = 4096
 GAIN = 0.4000
 RIST = 1.00×10^{-11}
 RKP = 1.18×10^3
 RKS = 9.19×10^3
 TTWD = 633.00
 YOST = 0.0000
 THST = 0.1000
 CFINAL = 40.000
 CINT = 20.000
 CKEEP = 10.000
 CSTEP = 0.0500

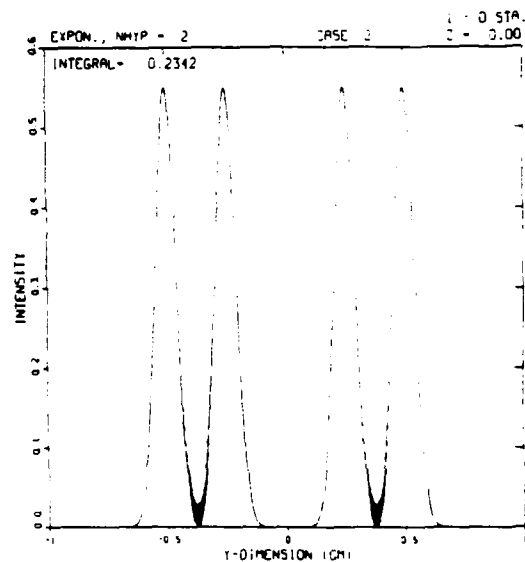
LIST OF INPUT PARAMETERS CONT

RADPMP(1-8) = 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
 ROSLIM(1-8) = 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
 RINT(1-10) = 0.5500 0.5500 0.5500 0.5500 0.5500 0.5500
 YOFF(1-10) = -0.5000 -0.2500 0.2500 0.5000 0.0000 0.0000
 TM(1,2) = -1.0000 1.0000
 YWIDTH = 0.1000 0.1000 0.1000 0.1000 0.1000 0.1000

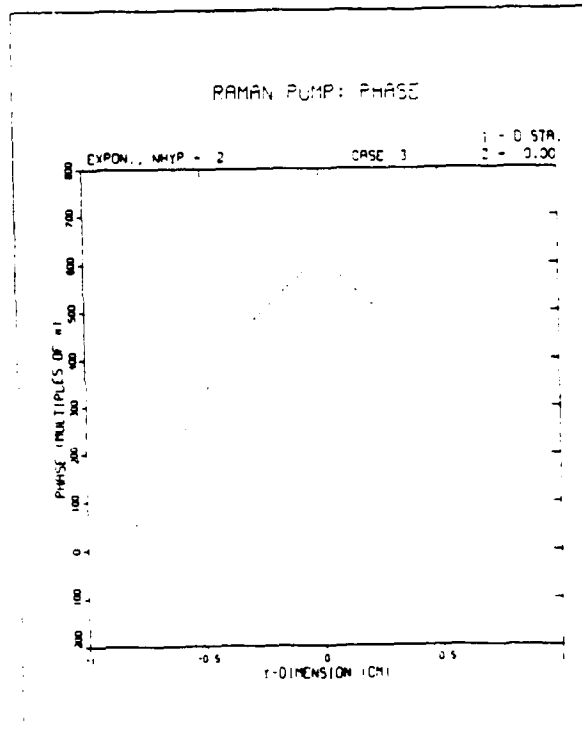
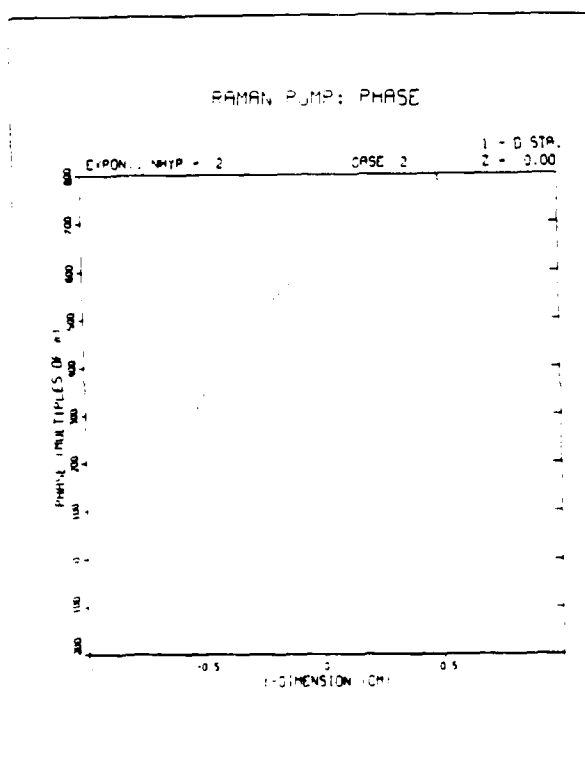
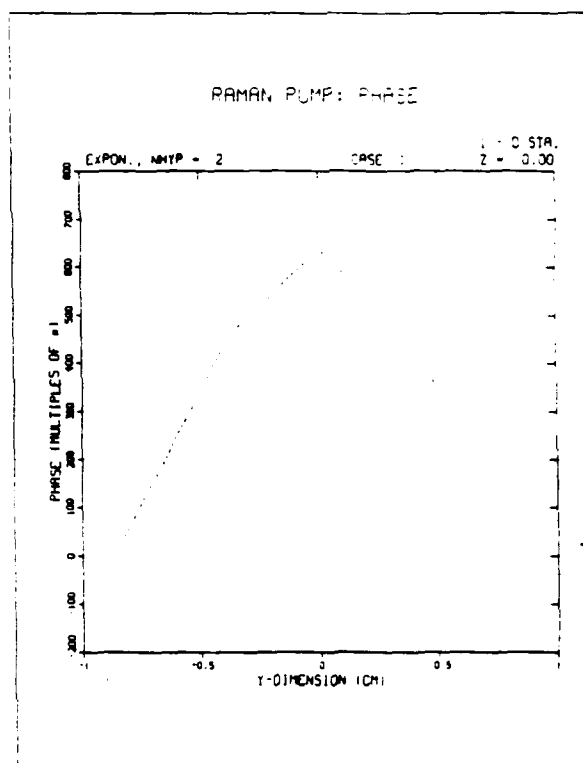
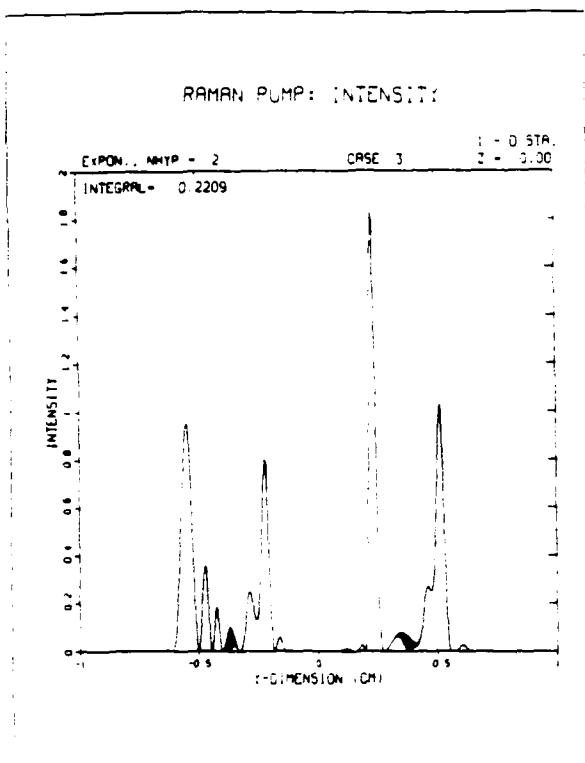
RAMAN PUMP: INTENSITY



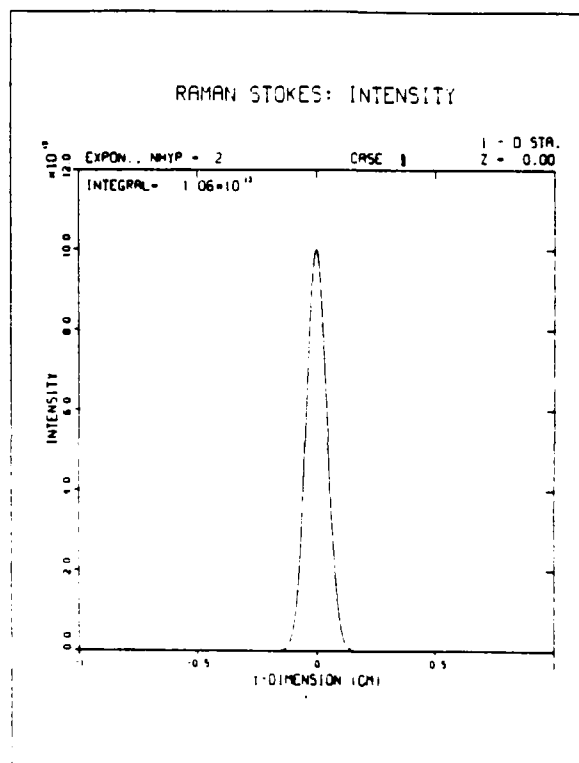
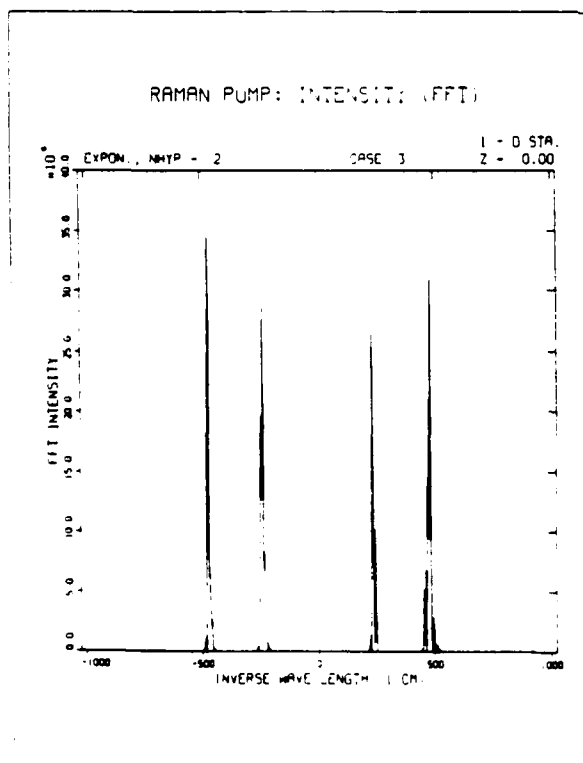
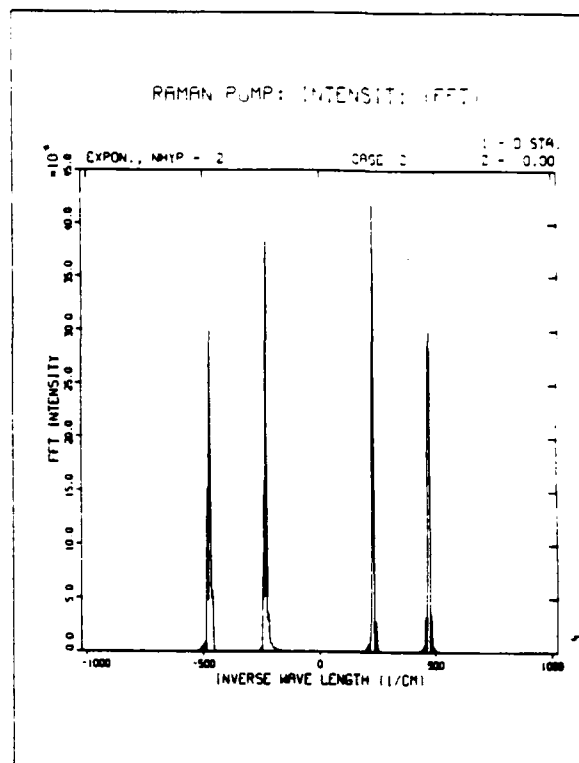
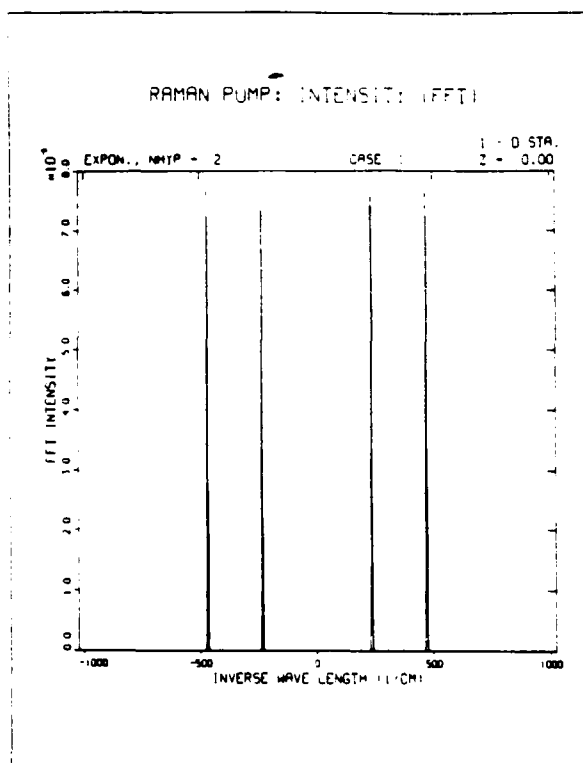
RAMAN PUMP: INTENSITY



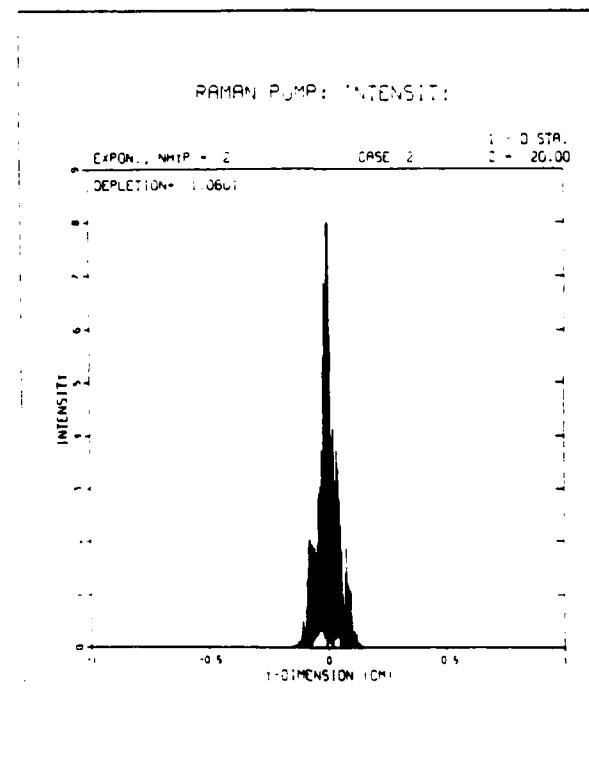
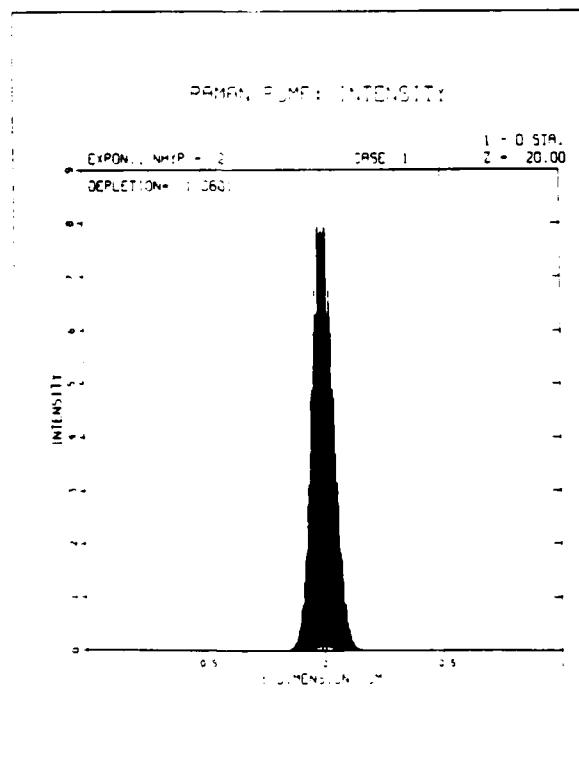
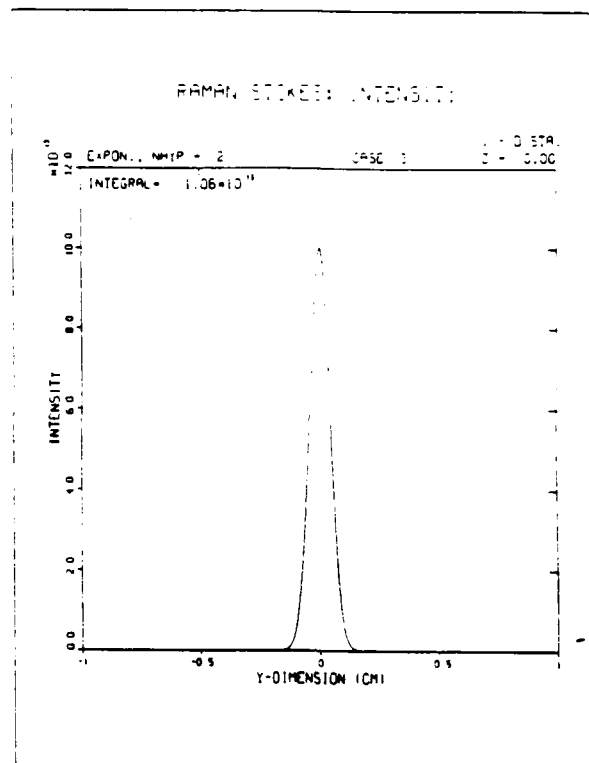
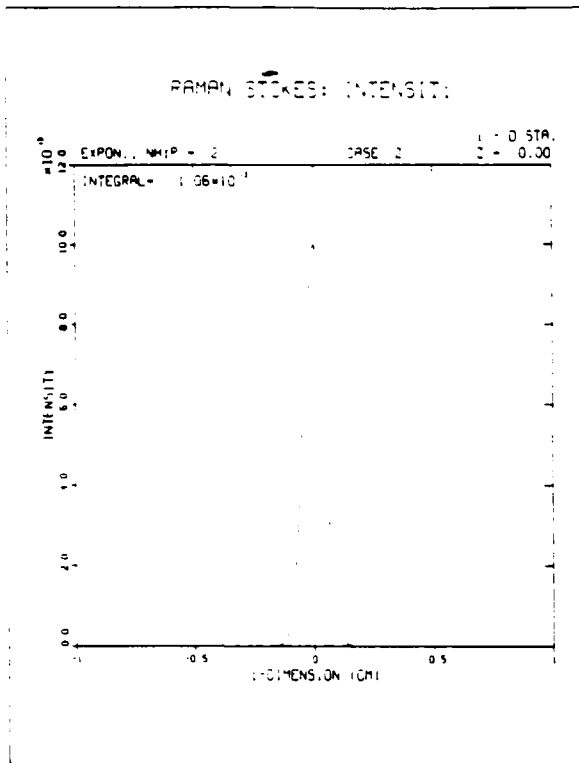
PLT2.DAT (Example B2)



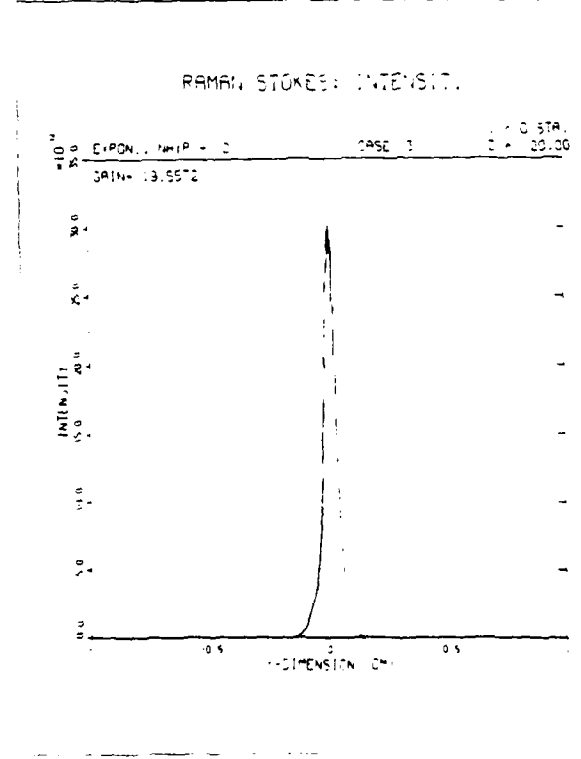
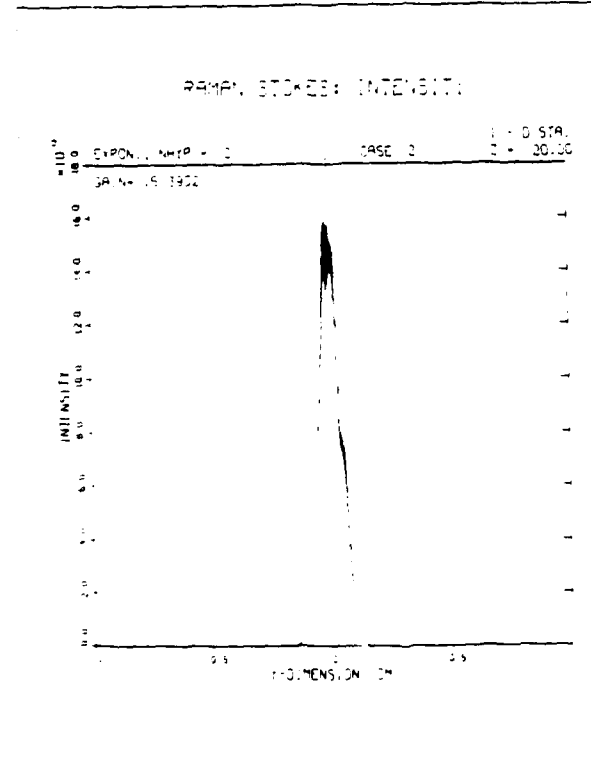
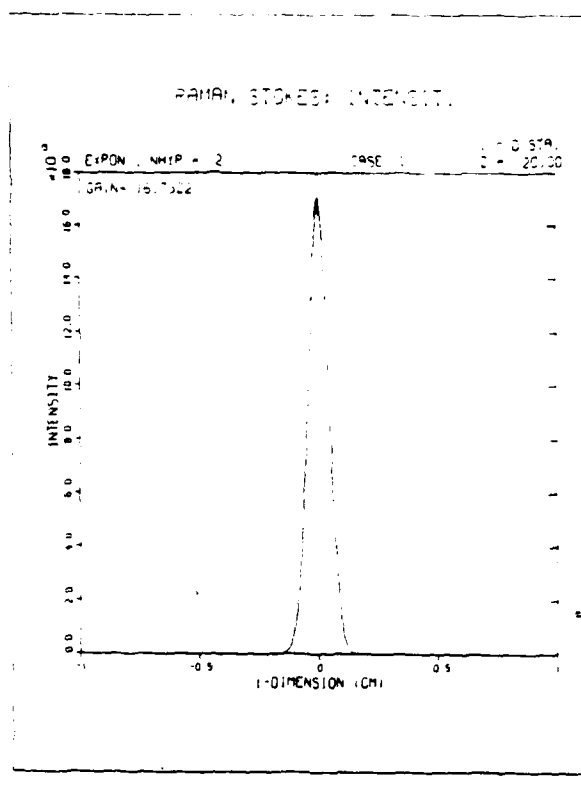
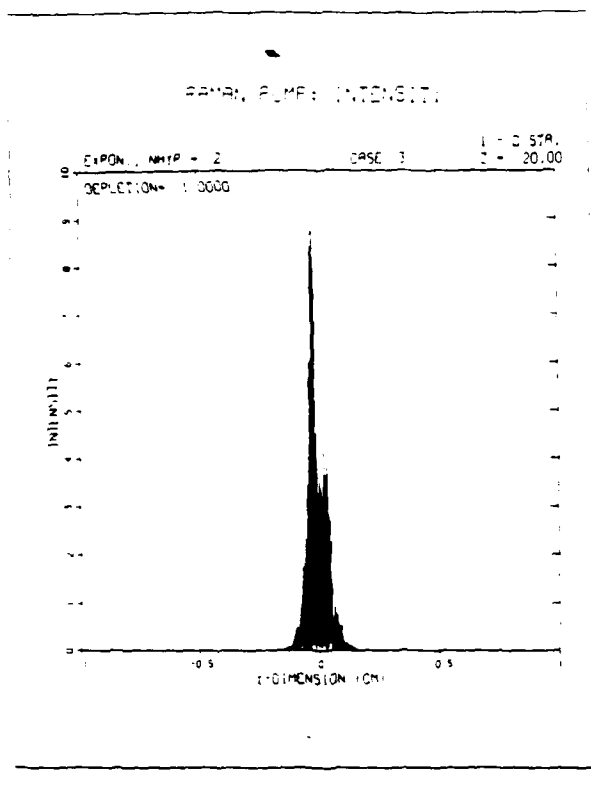
PLT2.DAT (Example B2)



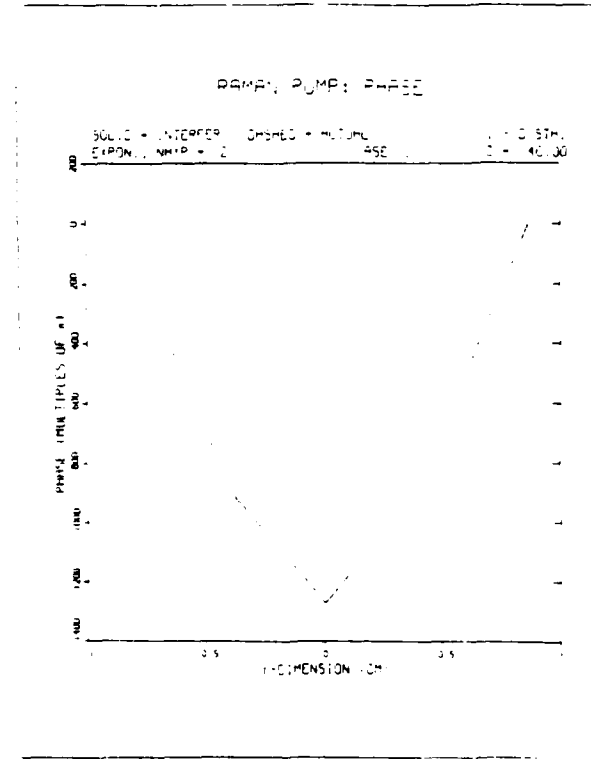
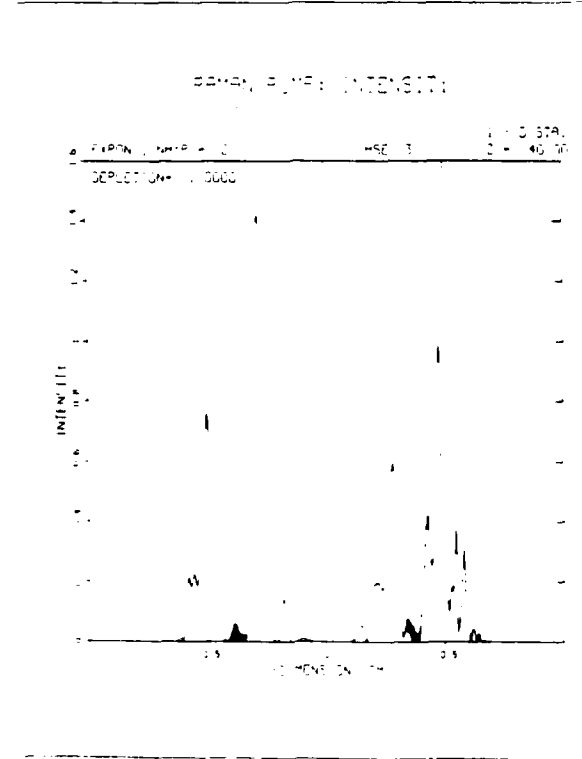
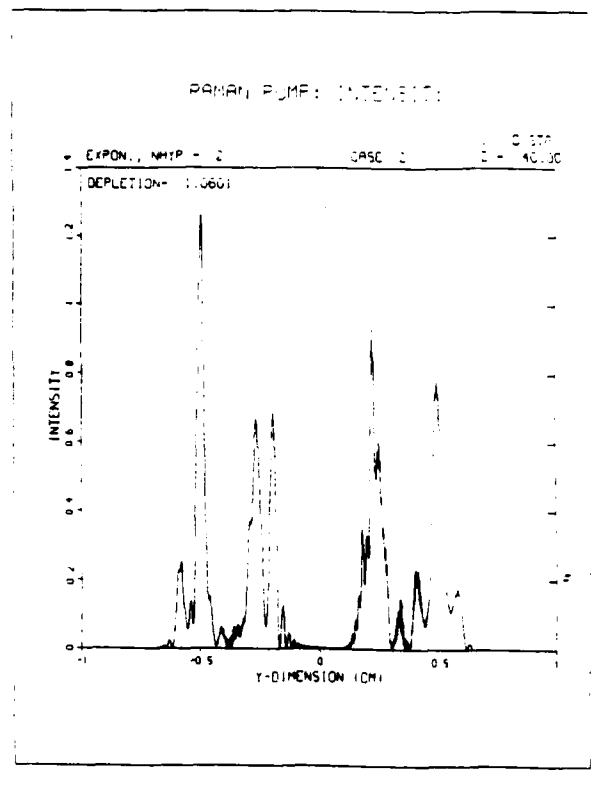
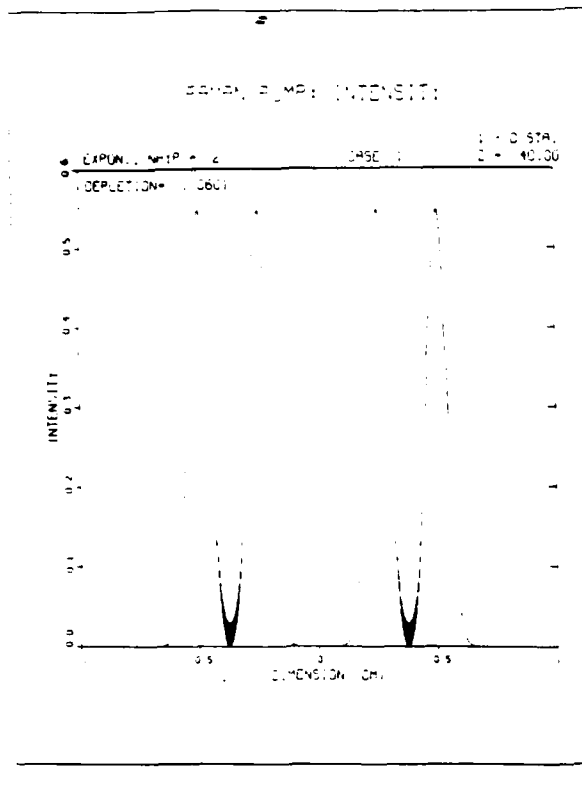
PLT2.DAT (Example B2)



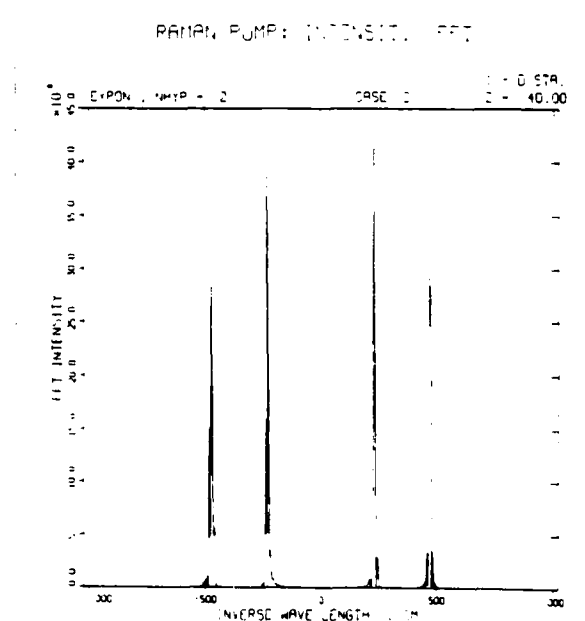
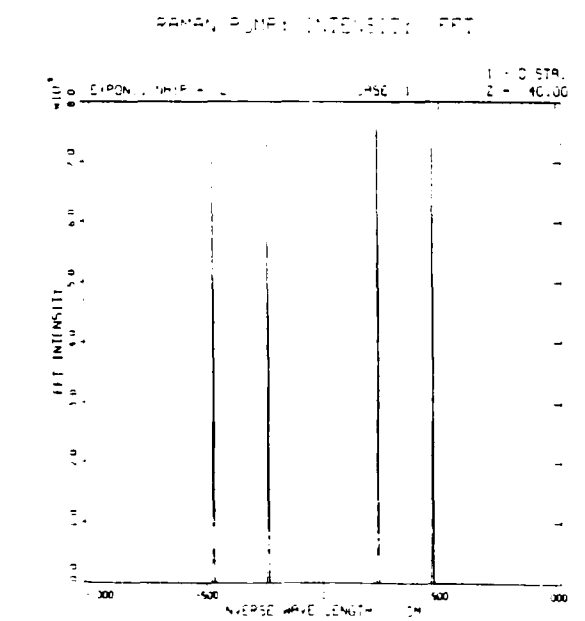
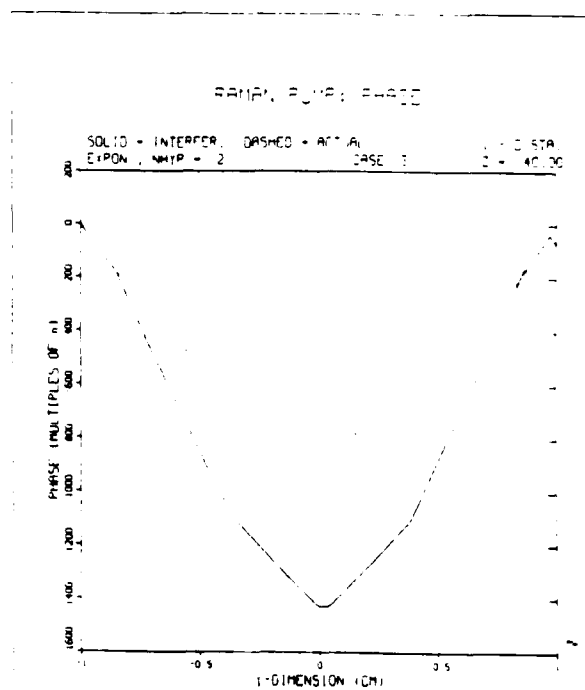
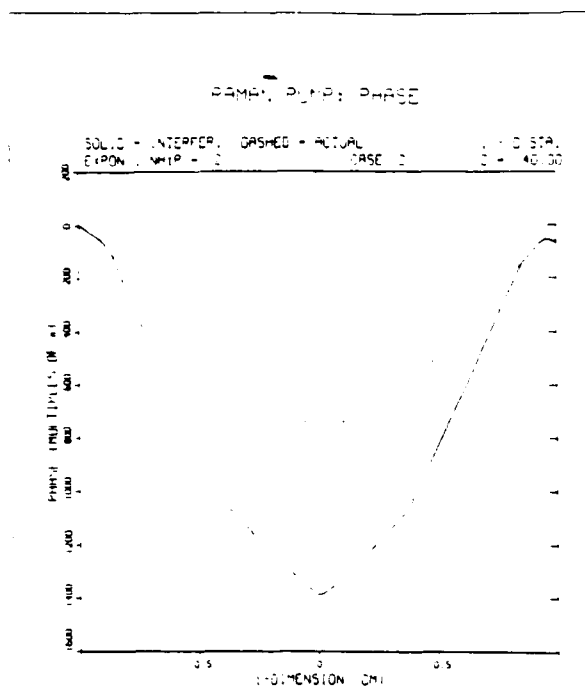
PLT2.DAT (Example B2)



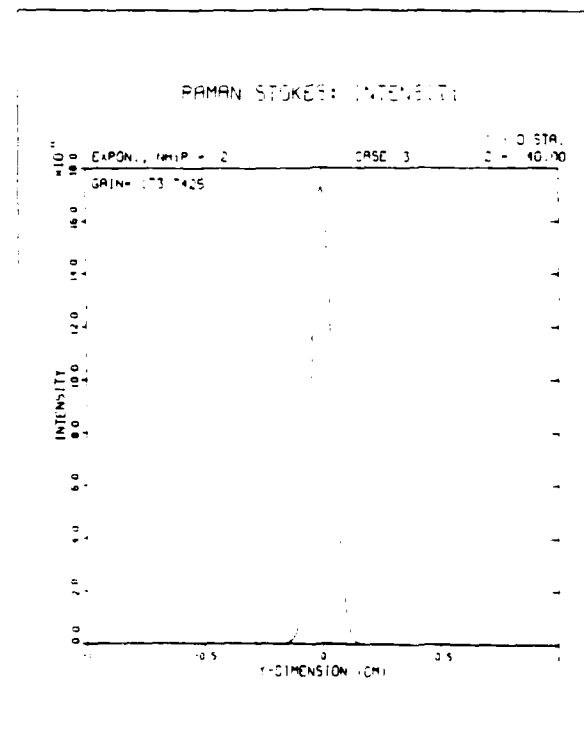
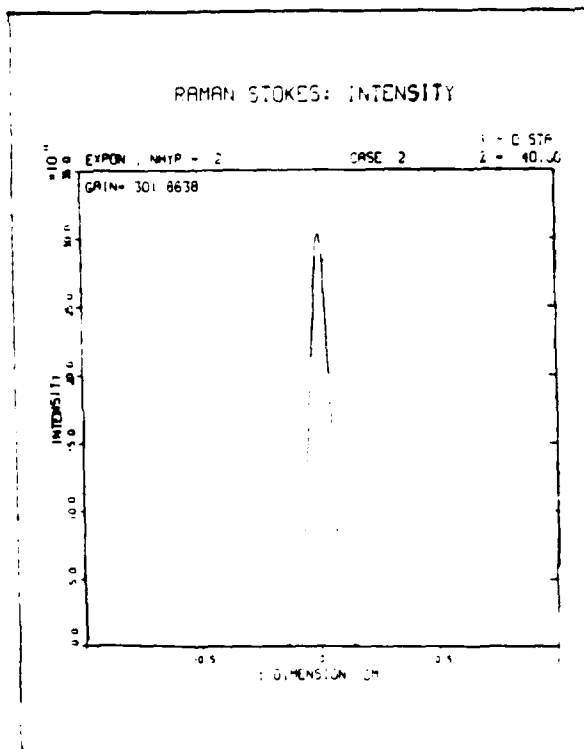
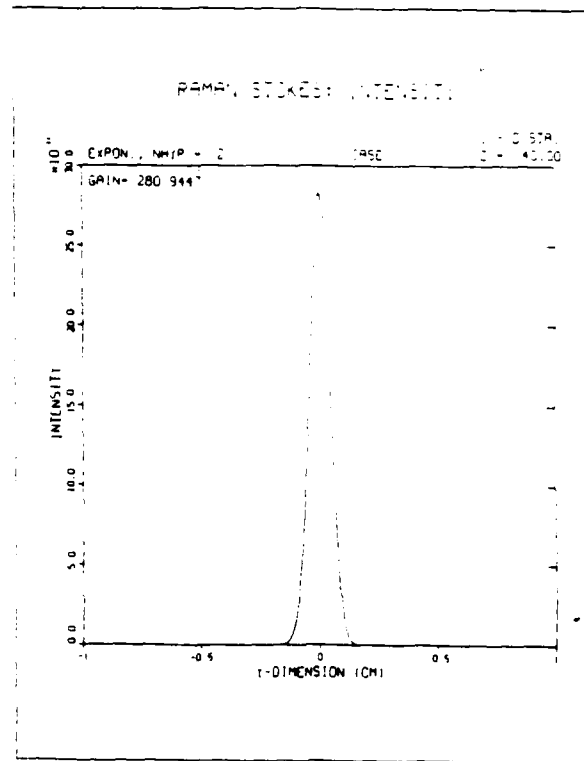
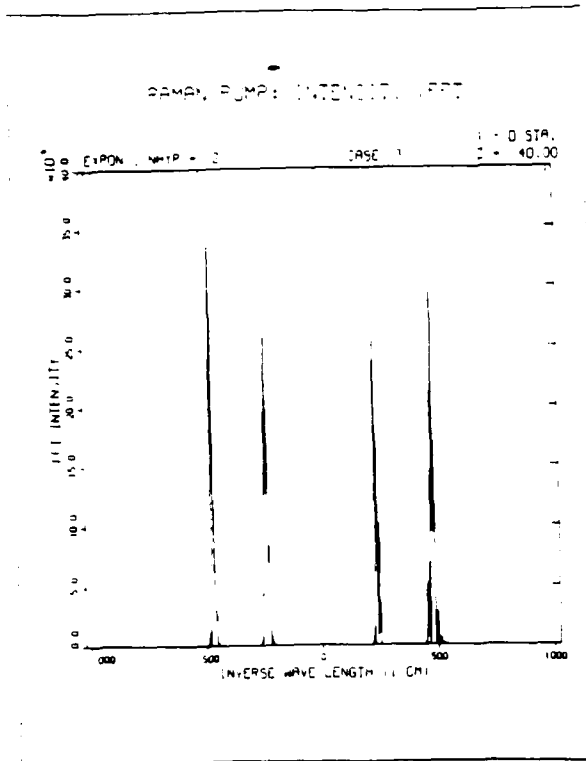
PLT2.DAT (Example B2)



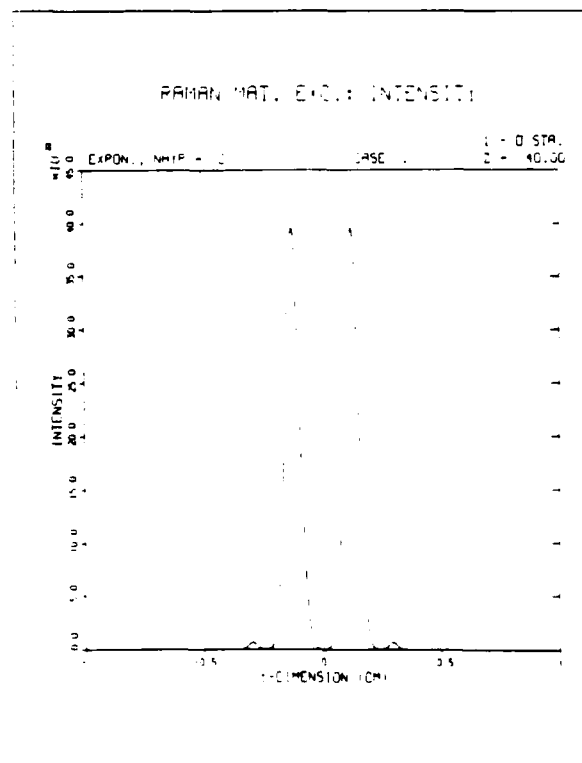
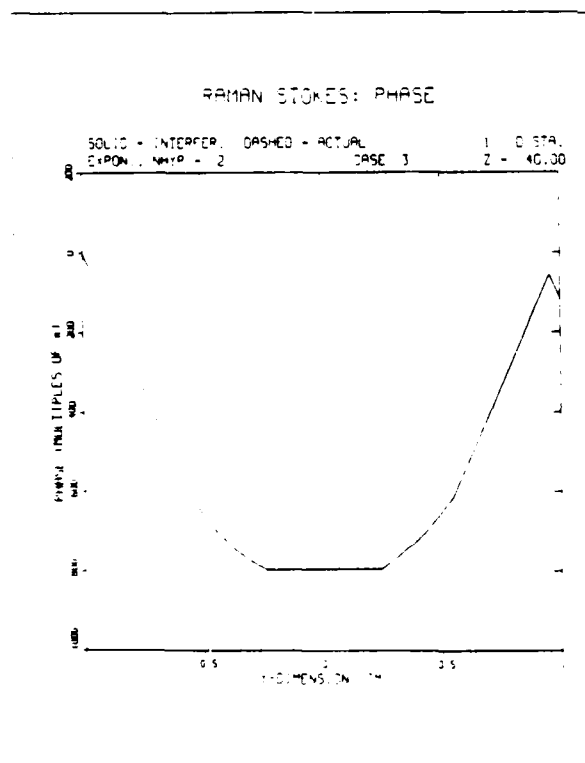
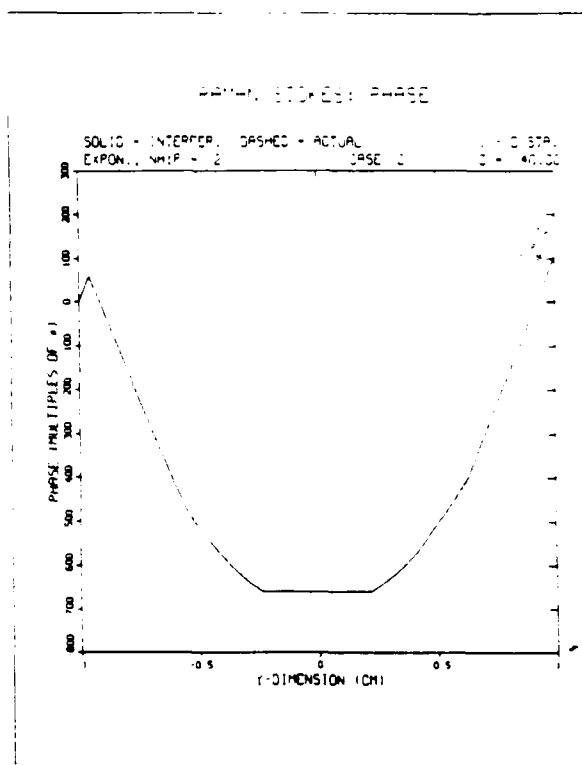
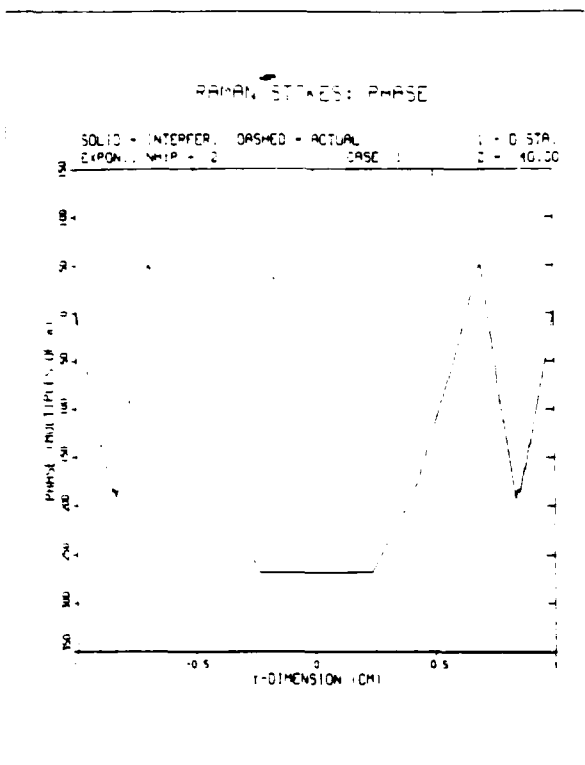
PLT2.DAT (Example B2)



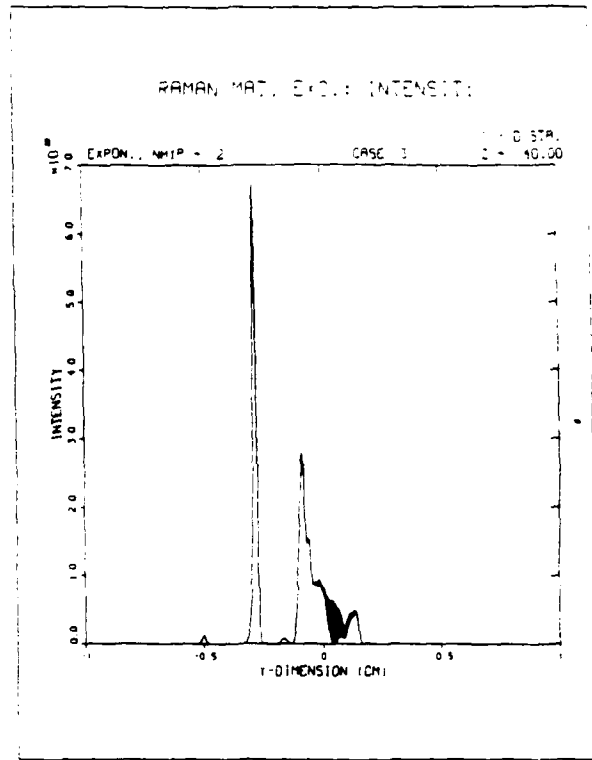
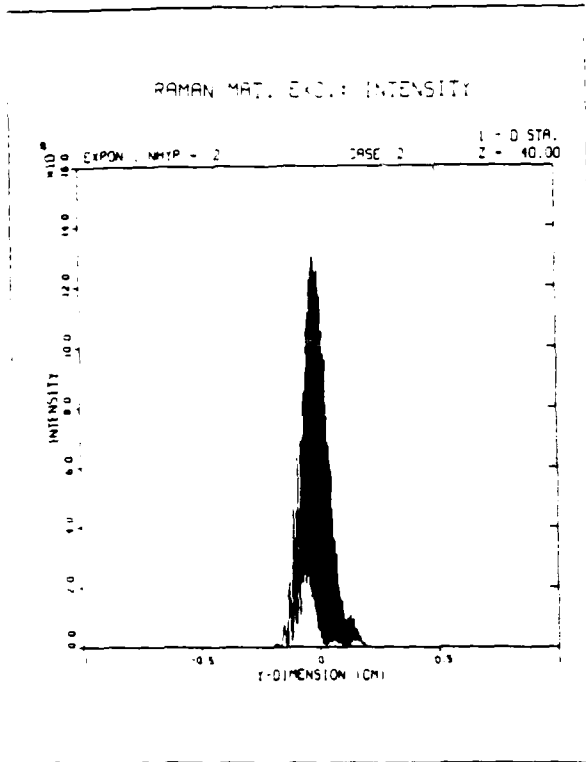
PLT2.DAT (Example B2)



PLT2.DAT (Example B2)



PLT2.DAT (Example B2)



XRL3.CPR

09 09 11 3658	60 3760	USER	MINIMUM JTA WORDS -	3584
09 09 11 3661	60 3760	USER	MAXIMUM JTA WORDS -	4808
09 09 11 3664	60 3760	USER	DISK SECTORS MOVED	2770
09 09 11 3667	60 3760	USER	FMS SECTORS MOVED -	0
09 09 11 3671	60 3760	USER	USER I/O REQUESTS -	747
09 09 11 3674	60 3760	USER	USER I/O SUSPENSIONS -	1163
09 09 11 3677	60 3760	USER	OPEN CALLS -	20
09 09 11 3681	60 3760	USER	CLOSE CALLS -	18
09 09 11 3684	60 3760	USER	MEMORY RESIDENT DATASETS -	0
09 09 11 3687	60 3760	USER	TEMPORARY DATASET SECTORS USED -	0
09 09 11 3691	60 3761	USER	PERMANENT DATASET SECTORS ACCESSED -	1594
09 09 11 3694	60 3761	USER	PERMANENT DATASET SECTORS SAVED -	0
09 09 11 3697	60 3761	USER	SECTORS RECEIVED FROM FRONT END -	1
09 09 11 3701	60 3761	USER	SECTORS QUEUED TO FRONT END -	0
09 09 11 6604	60 3837	USER		
09 09 11 6608	60 3837	USER		
09 09 11 6610	60 3838	USER		
09 09 11 6613	60 3839	USER		
09 09 11 6617	60 3840	USER		
09 09 11 6620	60 3841	USER		
09 09 11 6624	60 3842	USER		
09 09 11 6628	60 3843	USER		
09 09 11 6717	60 3844	USER		
09 09 11 6721	60 3845	USER		
09 09 11 6725	60 3846	USER		
09 09 11 6728	60 3848	USER		
09 09 11 6741	60 3849	USER		
09 09 11 6745	60 3850	USER		
09 09 11 6747	60 3850	USER		
09 09 11 6750	60 3850	USER		

```

***** COST TABLE FOR THIS JOB *****
JOBNAME -----
USER IDENT -----
BEGAN EXECUTION ---- THU APR 21, 1988
AT A PRIORITY OF --
AND JOB CLASS OF --
60 382510 SECONDS OF CPU TIME @ $ 830.00 HR -- $ 10 57
23.047894 MEMORY CPU (MWRD-SEC) @ $ 84.00 HR -- $ 0 54
1 847957 MEMORY I/O (MWRD-SEC) @ $ 84.00 HR -- $ 0 04
0 002771 I/O MEGASECTORS MOVED @ $ 84.00 EA -- $ 0 23
0 000000 TAPE MOUNT(S) @ $ 5.00 EA -- $ 0 00

***** TOTAL COST FOR THIS JOB ***** -- $ 11 38

```

XPL3.CPR

```

09:10:53 3247 0 3000 CSP
09:10:53 3250 0 3000 CSP
09:10:53 3253 0 3000 CSP
09:10:53 3255 0 3000 CSP
09:10:53 3258 0 3001 CSP
09:10:53 3260 0 3001 CSP
09:10:53 3263 0 3001 CSP
09:10:53 3266 0 3001 CSP
09:10:53 3268 0 3001 CSP
09:10:53 3271 0 3001 CSP
09:10:53 3273 0 3001 CSP
09:10:53 3276 0 3001 CSP
09:10:53 3303 0 3002 CSP
09:10:53 3306 0 3002 CSP
09:10:53 3309 0 3002 CSP
09:10:53 3312 0 3002 CSP
09:10:53 3314 0 3002 CSP
09:10:53 4910 0 3002 CSP
09:10:53 9945 0 3003 CSP
09:10:55 0825 0 1007 USER
09:10:55 8605 0 1127 USER
09:11:10 7238 0 3403 USER
09:11:10 7242 0 3404 USER
09:11:10 7246 0 3405 USER
09:11:10 7326 0 3409 CSP
09:11:11 0174 0 3409 PDM
09:11:11 0176 0 3409 PDM
09:11:11 0195 0 3412 CSP
09:11:11 2570 0 3413 PDM
09:11:11 2572 0 3413 PDM
09:11:11 2590 0 3416 CSP
09:11:11 4937 0 3417 PDM
09:11:11 4939 0 3417 PDM
09:11:11 4956 0 3418 CSP
09:11:11 7657 0 3419 PDM
09:11:11 7659 0 3419 PDM
09:11:11 7675 0 3419 CSP
09:11:13 9189 0 3421 SCP
09:11:13 9172 0 3421 SCP
09:11:13 9176 0 3421 SCP
09:11:18 3138 0 3421 SCP
09:11:18 5569 0 3422 CSP
09:11:20 1680 0 3489 PDM
09:11:20 1683 0 3489 PDM
09:11:20 1707 0 3491 USER
09:11:20 1712 0 3493 USER
09:11:20 1715 0 3493 USER
09:11:20 1718 0 3493 USER
09:11:20 1721 0 3493 USER
09:11:20 1725 0 3494 USER
09:11:20 1728 0 3494 USER
09:11:20 1731 0 3494 USER
09:11:20 1736 0 3494 ABORT
09:11:20 1739 0 3494 ABORT
09:11:20 1741 0 3494 ABORT
09:11:20 1746 0 3494 EXP
09:11:20 1755 0 3494 CSP
09:11:20 1768 0 3495 CSP
09:11:20 1770 0 3495 CSP
09:11:20 3507 0 3496 USER

.....
WELCOME TO THE NRL CRAY XMP
.....
The CRAY will be unavailable Sunday April 24 from 8 00 A M to 4 00 P M
for software testing
.....
There will be no CRAY off-line data set recalls on Tuesday or Wednesday
mornings between 2:00 AM and 7:00 AM in order for us to perform CLEANUP
runs on our CRAY archive tape library
.....
CRAY X MP SERIAL 415 65 NAVAL RESEARCH LABORATORY 04 21 88
CRAY OPERATING SYSTEM COS 1 15 ASSEMBLY DATE 01 04 88

JOB JN-XP3L.MFL-511000 US-DEFER
ACCOUNT.AC-US-UPW-APW-
AC213 - TOTAL BUDGET WARNING LEVEL REACHED FOR THIS ACCOUNT NUMBER
AUDIT
AU003 - 214 DATASETS. 226297 BLOCKS. 115795201 WORDS
AU003 - 64 DATASETS. 46408 BLOCKS. 23744068 WORDS ONLINE
AU003 - 150 DATASETS. 179891 BLOCKS. 92051135 WORDS OFFLINE
ACCESS. DN-DISLIB.ID-DISSPLA.OWN-LIBRARY
PD000 - PDN - DISLIB ID - DISSPLA ED - 1 OWN - LIBRARY
PD000 - ACCESS COMPLETE
ACCESS. DN-INTLIB.ID-DISSPLA.OWN-LIBRARY
PD000 - PDN - INTLIB ID - DISSPLA ED - 1 OWN - LIBRARY
PD000 - ACCESS COMPLETE
ACCESS. DN-DVSD.ID-DISSPLA.OWN-LIBRARY
PD000 - PDN - DVSD ID - DISSPLA ED - 1 OWN - LIBRARY
PD000 - ACCESS COMPLETE
ACCESS. DN-XP3L ID - ED - 1 OWN - HILFER
PD000 - PDN - XP3L ID - ED - 1 OWN - HILFER
PD000 - ACCESS COMPLETE
FETCH. DN-NPRAM1.TEXT- NP3L.DAT
VAX TO CRAY: %SYSTEM S-NORMAL. normal successful completion
VAX TO CRAY: FILE-113DUAL07:[HILFER FR2]NP3L.DAT:5
VAX TO CRAY: 1680 BYTES TRANSFERRED
SS004 - DATASET RECEIVED FROM FRONT END
XP3L
PD000 PDN - F3L042188 ID - ED - 1 OWN - HILFER
PD009 DATASET NOT FOUND
IO054 - ATTEMPT TO BACKUP FROM BOD
SLO10 - READ F3L421 READ PAST END OF DATA
TB001 - BEGINNING OF TRACEBACK
- STRBK WAS CALLED BY SLERP% AT 1137553a
SLERP% WAS CALLED BY SRWDP AT 1136510a
SRWDP WAS CALLED BY SRUV% AT 1100165a
SRUV% WAS CALLED BY PRAM1CD AT 102425a LINE NUMBER 144
TB002 - END OF TRACEBACK
AB025 - USER PROGRAM REQUESTED ABORT
AB000 - JOB STEP ABORTED. P - 01137560b
AB000 - BASE 07703000 LIMIT 11226000 CPU NUMBER 00
EXIT
END OF JOB

JOB NAME XP3L

```

XPL3.CPR

09 11 20 3510	0 3496	USER	USER NUMBER	HILFER
09 11 20 3513	0 3496	USER	JOB SEQUENCE NUMBER	40355
09 11 20 3516	0 3496	USER		
09 11 20 3519	0 3496	USER	TIME EXECUTING IN CPU -	0000:00:00.3496
09 11 20 3522	0 3496	USER	TIME WAITING TO EXECUTE -	0000:00:14.1034
09 11 20 3525	0 3496	USER	TIME WAITING FOR I O	0000:00:12.3816
09 11 20 3528	0 3497	USER	TIME WAITING IN INPUT QUEUE -	0000:00:00.0068
09 11 20 3532	0 3497	USER	MEMORY ' CPU TIME (MWD'S)SEC -	0 02972
09 11 20 3535	0 3497	USER	MEMORY ' I O WAIT TIME (MWD'S)SEC -	1 50167
09 11 20 3538	0 3497	USER	MINIMUM JOB SIZE (WORDS)	44544
09 11 20 3541	0 3497	USER	MAXIMUM JOB SIZE (WORDS) -	374784
09 11 20 3544	0 3497	USER	MINIMUM FL (WORDS) -	40960
09 11 20 3547	0 3497	USER	MAXIMUM FL (WORDS) -	370176
09 11 20 3550	0 3497	USER	MINIMUM JTA (WORDS) -	3584
09 11 20 3553	0 3498	USER	MAXIMUM JTA (WORDS) -	4608
09 11 20 3556	0 3498	USER	DISK SECTORS MOVED -	1888
09 11 20 3559	0 3498	USER	FSS SECTORS MOVED -	0
09 11 20 3562	0 3498	USER	USER I O REQUESTS -	733
09 11 20 3566	0 3498	USER	USER I O SUSPENSIONS -	956
09 11 20 3569	0 3498	USER	OPEN CALLS	20
09 11 20 3572	0 3498	USER	CLOSE CALLS -	18
09 11 20 3575	0 3498	USER	MEMORY RESIDENT DATASETS -	0
09 11 20 3580	0 3498	USER	TEMPORARY DATASET SECTORS USED -	0
09 11 20 3583	0 3498	USER	PERMANENT DATASET SECTORS ACCESSED -	2482
09 11 20 3586	0 3498	USER	PERMANENT DATASET SECTORS SAVED -	0
09 11 20 3589	0 3498	USER	SECTORS RECEIVED FROM FRONT END -	1
09 11 20 3592	0 3498	USER	SECTORS QUEUED TO FRONT END -	0

.....

*** COST TABLE FOR THIS JOB ***			
JOBNAME	-----	XP3L	
USER IDENT	-----	HILFER	
BEGAN EXECUTION	----	THU APR 21, 1988	09:10:52 HOURS
AT A PRIORITY OF	--		3
AND JOB CLASS OF	--	DSMALL	
0.355991	SECONDS OF CPU TIME	@ \$ 830.00	HR 0 06
0.030136	MEMORY CPU (MWRD-SEC)	@ \$ 84.00	HR 0 00
1.503896	MEMORY I O (MWRD SEC)	@ \$ 84.00	HR 0 04
0.001890	I O MEGASECTORS MOVED	@ \$ 84.00	EA 0 16
0.000000	TAPE MOUNT(S)	@ \$ 5.00	EA 0 00
*** TOTAL COST FOR THIS JOB ***			
		\$	0 26

.....

NPL3.DAT

\$FLDATE

DONYET=1,
MONTH=03,
DAY=30,
YEAR=88,
IPART=2,
NEDN=1,

\$

\$CONDAT

LPRMT(1)=1,
LPRMT(2)=1,
LPRMT(3)=1,
LPRMT(4)=0,
NSEC=3,
CSEC(1,1)=(1.0,2.0),
CSEC(1,2)=(2.0,2.0),
CSEC(1,3)=(3.0,2.0),
CSEC(2,1)=(1.0,2.0),
CSEC(2,2)=(2.0,2.0),
CSEC(2,3)=(3.0,2.0),
CSEC(4,1)=(1.0,2.0),
CSEC(4,2)=(2.0,2.0),
CSEC(4,3)=(3.0,2.0),
CSEC(5,1)=(1.0,2.0),
CSEC(5,2)=(2.0,2.0),
CSEC(5,3)=(3.0,2.0),
CSEC(7,1)=(1.0,2.0),
CSEC(7,2)=(2.0,2.0),
CSEC(7,3)=(3.0,2.0),
CSEC(8,1)=(1.0,2.0),
CSEC(8,2)=(2.0,2.0),
CSEC(8,3)=(3.0,2.0),
CSEC(10,1)=(1.0,2.0),
CSEC(10,2)=(2.0,2.0),
CSEC(10,3)=(3.0,2.0),
CSEC(11,1)=(1.0,2.0),
CSEC(11,2)=(2.0,2.0),
CSEC(11,3)=(3.0,2.0),
CSEC(13,1)=(1.0,2.0),
CSEC(13,2)=(2.0,2.0),
CSEC(13,3)=(3.0,2.0),
CSEC(14,1)=(1.0,2.0),
CSEC(14,2)=(2.0,2.0),
CSEC(14,3)=(3.0,2.0),
CSEC(16,1)=(1.0,2.0),
CSEC(16,2)=(2.0,2.0),
CSEC(16,3)=(3.0,2.0),
CSEC(17,1)=(1.0,2.0),
CSEC(17,2)=(2.0,2.0),
CSEC(17,3)=(3.0,2.0),

\$

\$ZPLOT

KZ(1)=1,
KZ(2)=2,
KZ(3)=3,
KZ(4)=4,
KZ(5)=5,

\$

NRL3.DAT

\$NAML

NPUMP=4,
YM(1)=-1.0,
YM(2)=1.0,
YOFF(1)=-0.5,
YOFF(2)=-0.25,
YOFF(3)=0.25,
YOFF(4)=0.5,
RIST=1.0E-12,
NHYP=2,
RABAMP(1)=0.0,
RABAMP(2)=0.0,
RABAMP(3)=1.0,
RDSLIM(1)=0.0,
RDSLIM(2)=2.0,
RDSLIM(3)=2.0,
ICOND=4,
ZFINAL=40.0,
ZKEEP=10.0,
GAIN=0.4,

\$

NAMelist/NAML/NPUMP,YM,TM,ZINT,RKP,RKS,YOFF,TOFF,YWIDTH,TWIDTH,
1 YOST,TOST,YWST,TWST,RINT,RIST,RAASM,RALASM,NHYP,PHL,PHST,TOC,
2 ITYPE,RTYPE,RABAMP,RDSLIM,ICOND,ZSTEP,ZFINAL,ZKEEP,NMAX,TTWO,GAIN

XRL3.JOB

AUDIT.
FETCH, DN=NRAM, TEXT='NR3L.DAT'.
ACCESS, DN=XR3L.
XR3L.
DISPOSE, DN=ERRM, DF=BB, WAIT, TEXT='XR3L.MSG.'.
AUDIT.
EXIT.

XPL3.JOB

AUDIT.
ACCESS, DN=DISLIB, ID=DISSPLA, OWN=LIBRARY.
ACCESS, DN=INTLIB, ID=DISSPLA, OWN=LIBRARY.
ACCESS, DN=DVSD, ID=DISSPLA, OWN=LIBRARY.
ACCESS, DN=XP3L.
FETCH, DN=NPRAM1, TEXT='NP3L.DAT'.
XP3L.
AUDIT.
DISPOSE, DN=META, DF=BB, WAIT, TEXT='PLT2.DAT'.
DISPOSE, DN=EPRM, DF=BB, WAIT, TEXT='XP3L.MSG.'.
DISPOSE, DN=DISOUT, DF=BB, WAIT, TEXT='XP3L.DSP.'.
EXIT.

APPENDIX C 2-D Operation; Examples

One example is presented to illustrate the output of the codes RAM2D1 and PRAM1 in two-dimensional simulations.

EXAMPLE C

PLT2.DAT (Example C)

LIST OF INPUT PARAMETERS

```

COND      = 2
LEN       = 1
ISHM      = 1
NDEC      = 3
NHYP      = 8
NMAX      = 4000
NPUMP     = 2
NT        = 128
NY        = 512
GAIN      = 3.0000
PHST      = 0.0000
PALASH    = 5.0000
PAMASH    = 1.5000
RIST      = 0.0002
RKP       = 1.1800*103
RKS       = 91893.
TDC       = 5.0000
TOST      = -40.000
TTWO      = 633.00
TWST      = 40.000
TOST      = 0.0000
TWST      = 0.1000
CFINAL    = 40.000
CINT      = 20.000
CKEEP     = 1.0000
CSTEP     = 0.0500
    
```

LIST OF INPUT PARAMETERS CONTD.

```

ISRF(1-6) = -1 -1 -1 -1 -1 -1
LEVEL      = 5 3 4 5 5 7 8 9
PHL(1-10) = 0.0000 0.0000 0.0000 0.0000 0.0000
RINT(1-10) = 0.0800 0.0800 0.5500 0.5500 0.5500
            = 0.5500 0.5500 0.5500 0.5500 0.5500
TM(1,2)    = -100.00 100.00
TOFF(1-10) = 0.0000 0.0000 0.0000 0.0000 0.0000
            = 0.0000 0.0000 0.0000 0.0000 0.0000
TWIDTH     = 40.000 40.000 40.000 40.000 40.000
            = 40.000 40.000 40.000 40.000 40.000
YOFF(1-10) = 0.1400 -0.1400 0.0000 0.0000 0.0000
            = 0.0000 0.0000 0.0000 0.0000 0.0000
YM(1,2)    = -0.5000 0.5000
YWIDTH     = 0.1000 0.1000 0.1000 0.1000 0.1000
            = 0.1000 0.1000 0.1000 0.1000 0.1000
    
```

LIST OF INPUT PARAMETERS CONTD.

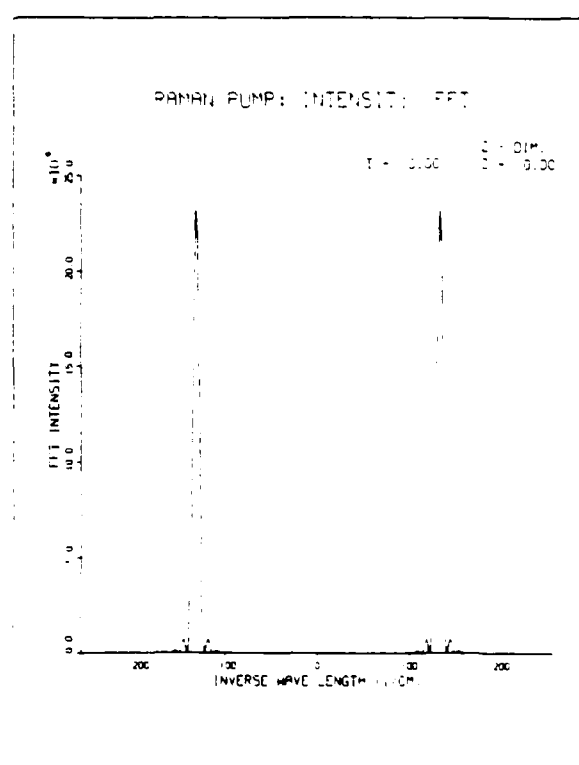
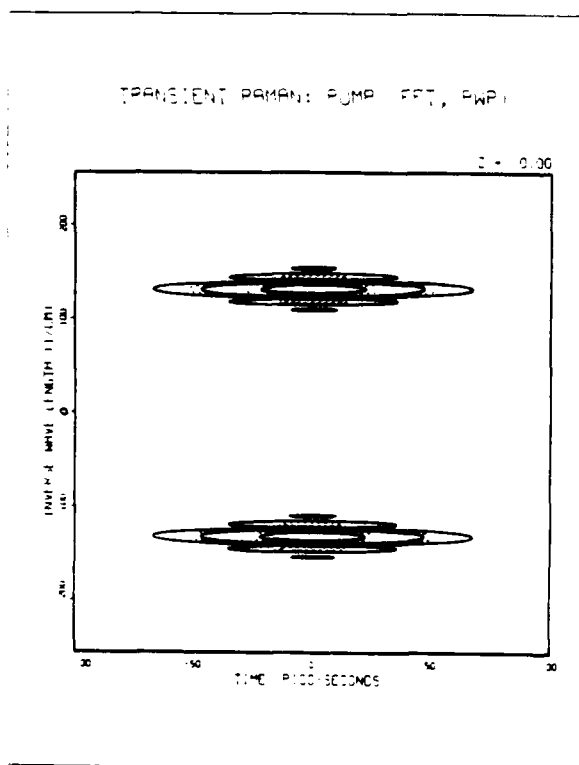
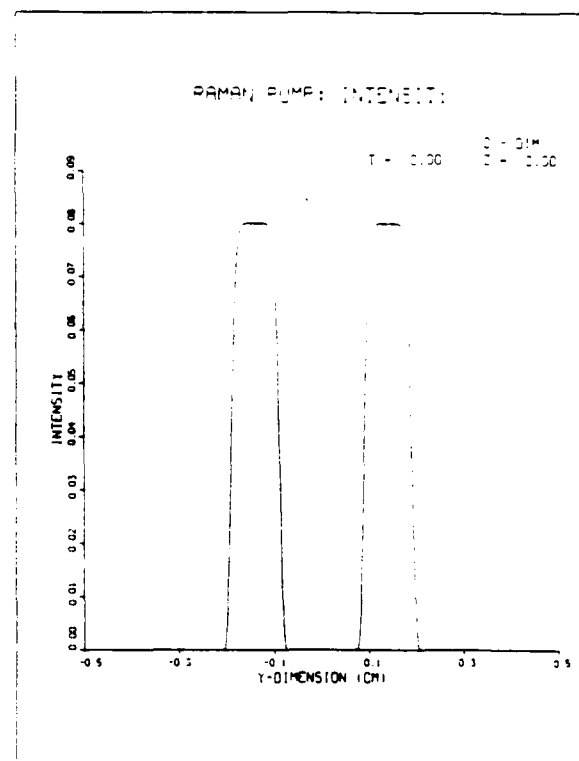
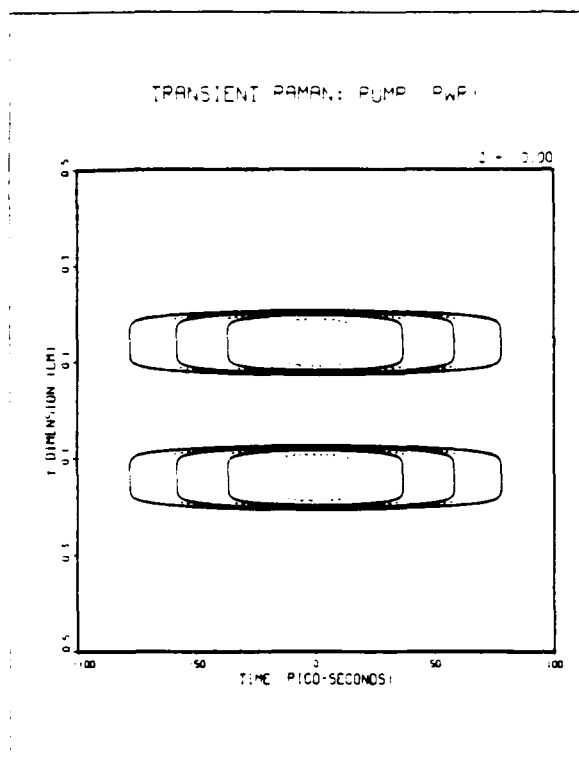
```

CSEC(1-19,1-8) =
0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
    
```

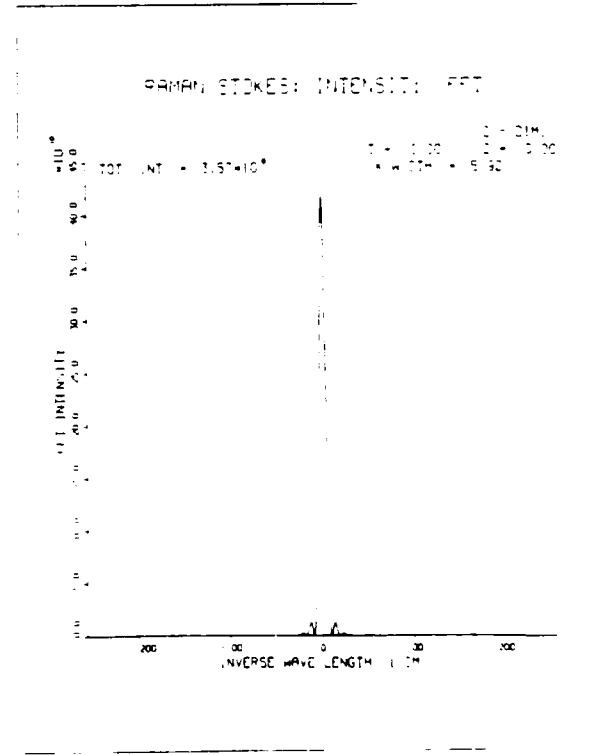
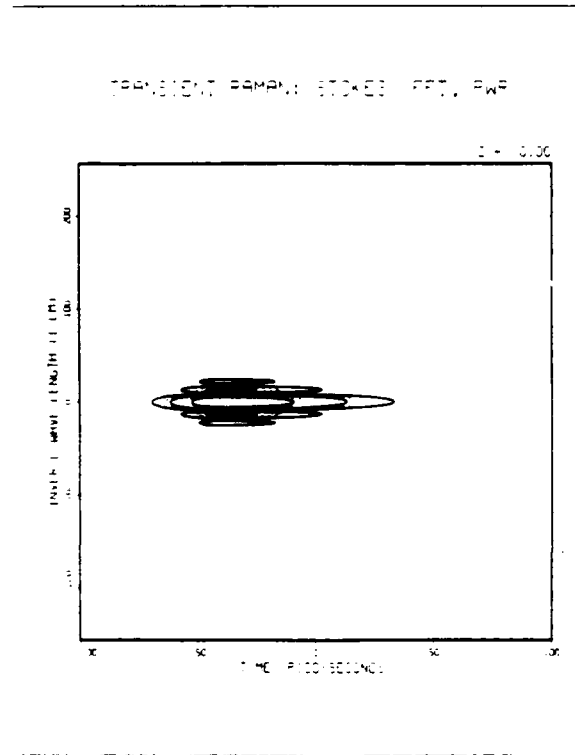
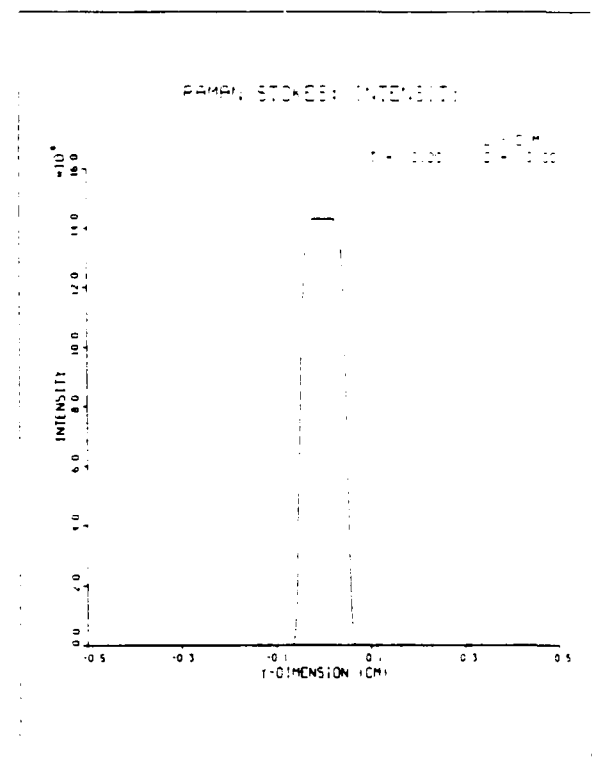
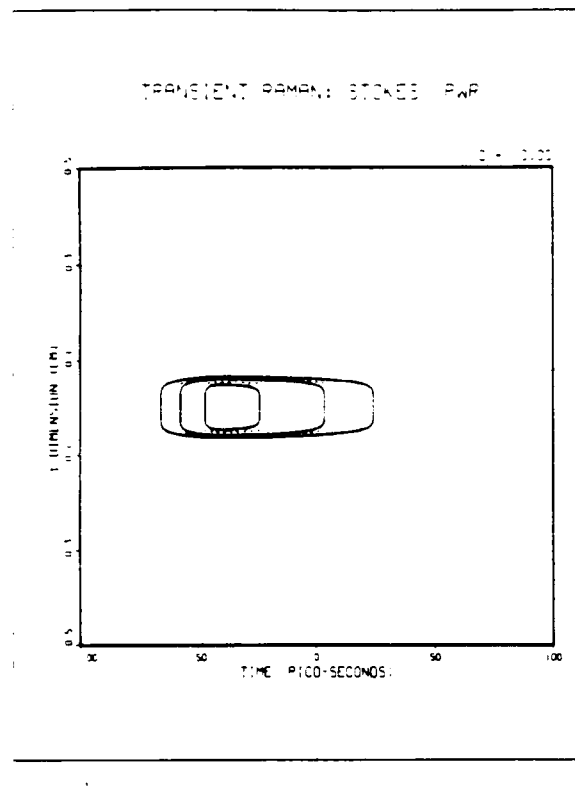
LIST OF OUTPUT PARAMETERS

PUMP	TOTAL INTENSITY	X-WIDTH
2	2.00*10 ⁴	6.07
1	2.00*10 ⁴	6.07

PLT2.DAT (Example C)



PLT2.DAT (Example C)



PLT2.DAT (Example C)

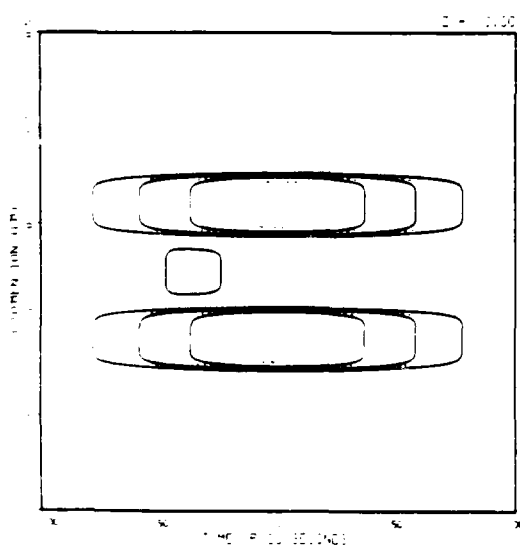
PARAM, MAT, EXCISE INTENSIT, PART

EXCISE INTENSIT, PART

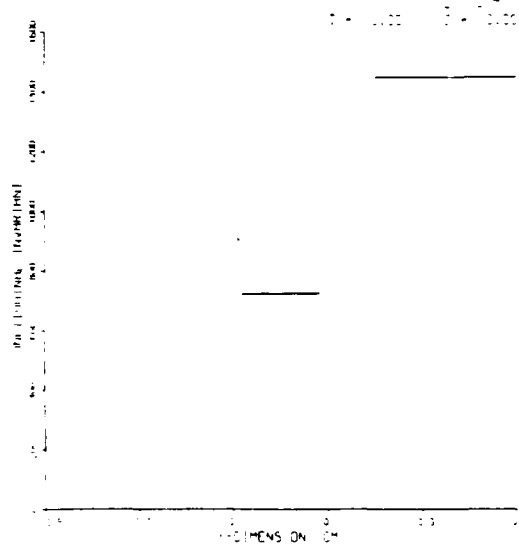
PARAM, MAT, EXCISE INTENSIT, PART

EXCISE INTENSIT, PART

PARAM, MAT, EXCISE INTENSIT, PART

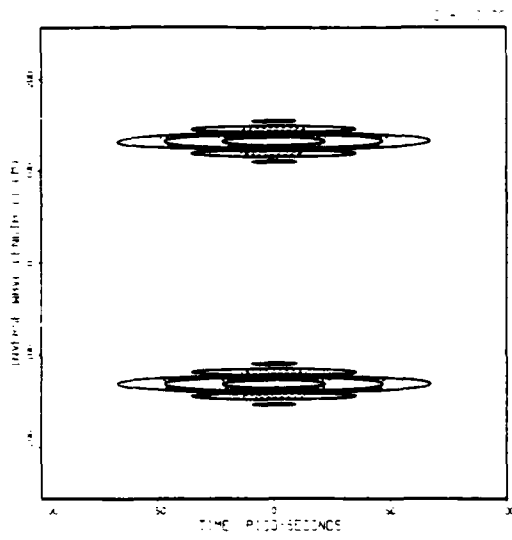


PARAM, MAT, EXCISE INTENSIT, PART

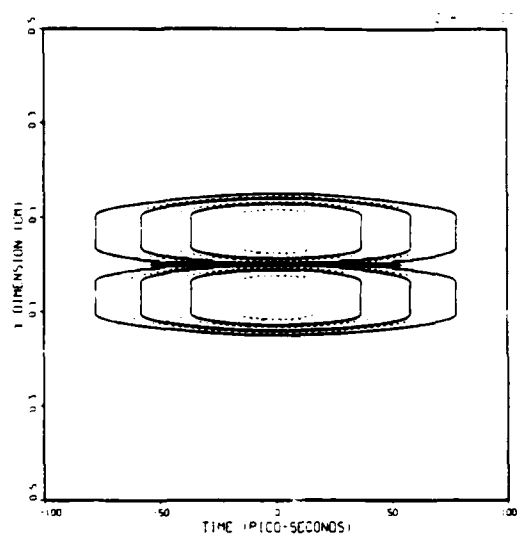


PLT2.DAT (Example C)

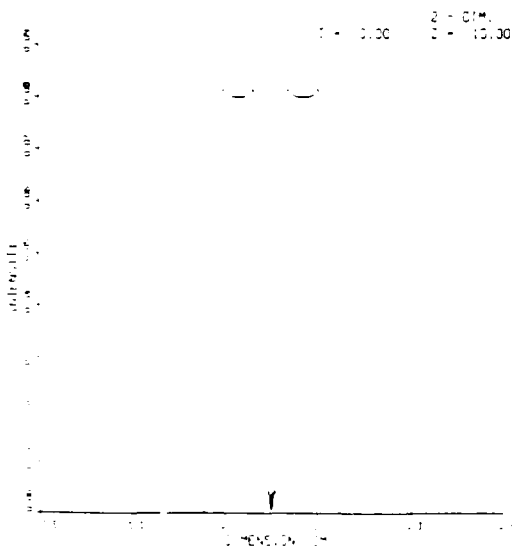
TRANSIENT Raman: PUMP AND Stokes FPD, FWP



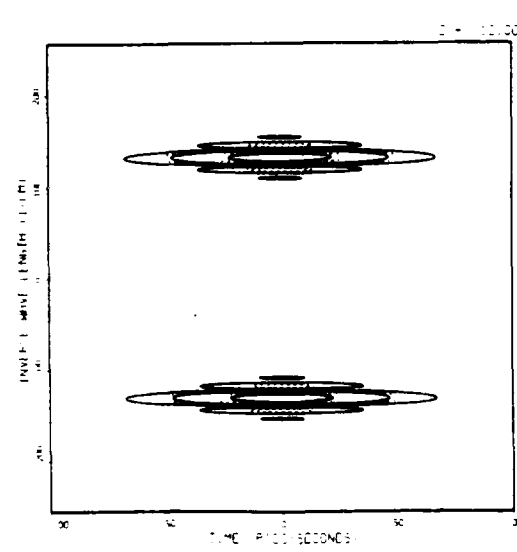
TRANSIENT Raman: PUMP PWP



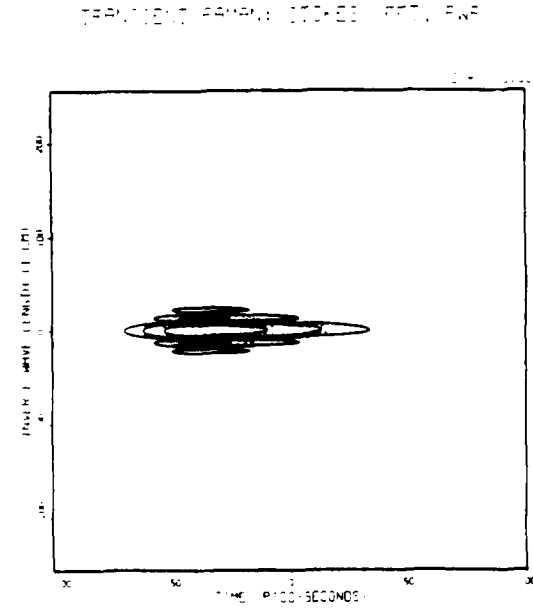
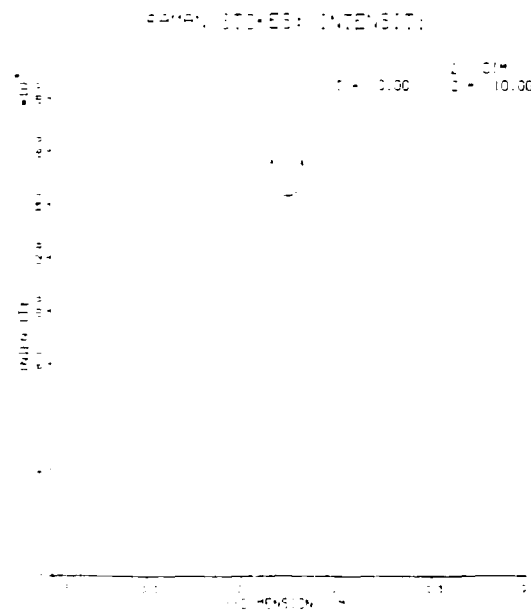
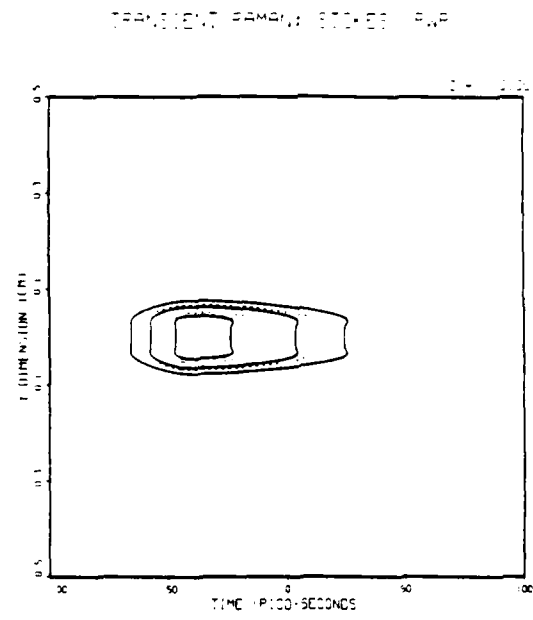
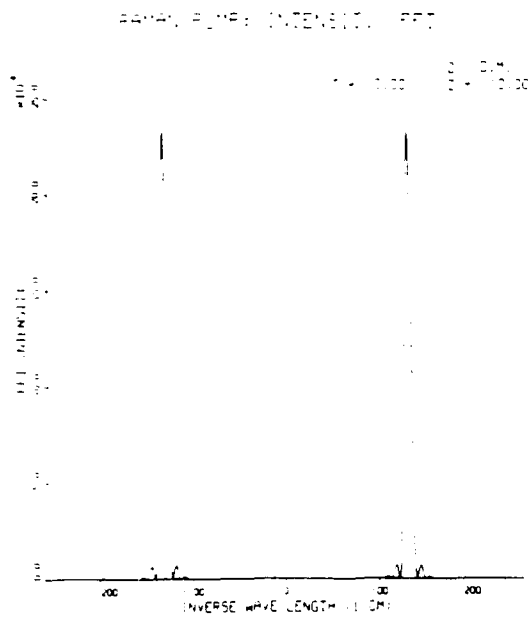
Raman: PUMP: INTENSITY



TRANSIENT Raman: PUMP FPD, FWP



PLT2.DAT (Example C)



6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

38

39

40

41

42

43

44

45

46

47

48

49

50

51

52

53

54

55

56

57

58

59

60

61

62

63

64

65

66

67

68

69

70

71

72

73

74

75

76

77

78

79

80

81

82

83

84

85

86

87

88

89

90

91

92

93

94

95

96

97

98

99

100

101

102

103

104

105

106

107

108

109

110

111

112

113

114

115

116

117

118

119

120

121

122

123

124

125

126

127

128

129

130

131

132

133

134

135

136

137

138

139

140

141

142

143

144

145

146

147

148

149

150

151

152

153

154

155

156

157

158

159

160

161

162

163

164

165

166

167

168

169

170

171

172

173

174

175

176

177

178

179

180

181

182

183

184

185

186

187

188

189

190

191

192

193

194

195

196

197

198

199

200

201

202

203

204

205

206

207

208

209

210

211

212

213

214

215

216

217

218

219

220

221

222

223

224

225

226

227

228

229

230

231

232

233

234

235

236

237

238

239

240

241

242

243

244

245

246

247

248

249

250

251

252

253

254

255

256

257

258

259

260

261

262

263

264

265

266

267

268

269

270

271

272

273

274

275

276

277

278

279

280

281

282

283

284

285

286

287

288

289

290

291

292

293

294

295

296

297

298

299

300

301

302

303

304

305

306

307

308

309

310

311

312

313

314

315

316

317

318

319

320

321

322

323

324

325

326

327

328

329

330

331

332

333

334

335

336

337

338

339

340

341

342

343

344

345

346

347

348

349

350

351

352

353

354

355

356

357

358

359

360

361

362

363

364

365

366

367

368

369

370

371

372

373

374

375

376

377

378

379

380

381

382

383

384

385

386

387

388

389

390

391

392

393

394

395

396

397

398

399

400

401

402

403

404

405

406

407

408

409

410

411

412

413

414

415

416

417

418

419

420

421

422

423

424

425

426

427

428

429

430

431

432

433

434

435

436

437

438

439

440

441

442

443

444

445

446

447

448

449

450

451

452

453

454

455

456

457

458

459

460

461

462

463

464

465

466

467

468

469

470

471

472

473

474

475

476

477

478

479

480

481

482

483

484

485

486

487

488

489

490

491

492

493

494

495

496

497

498

499

500

501

502

503

504

505

506

507

508

509

510

511

512

513

514

515

516

517

518

519

520

521

522

523

524

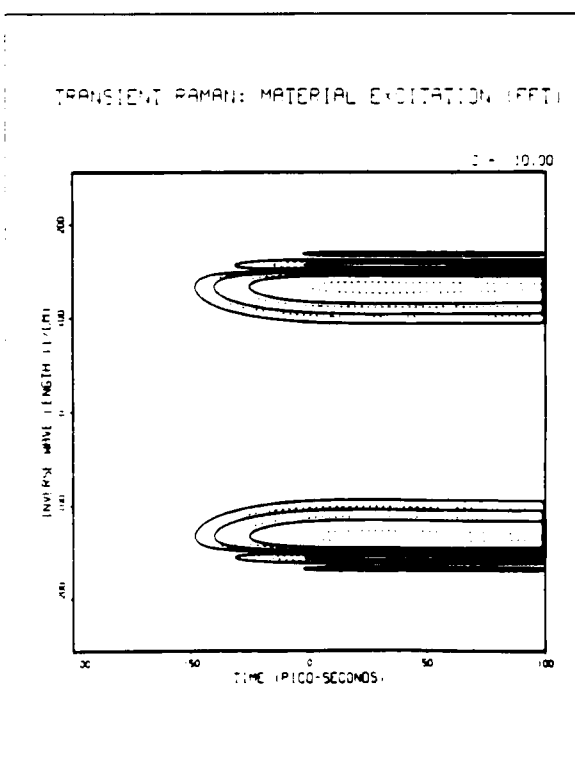
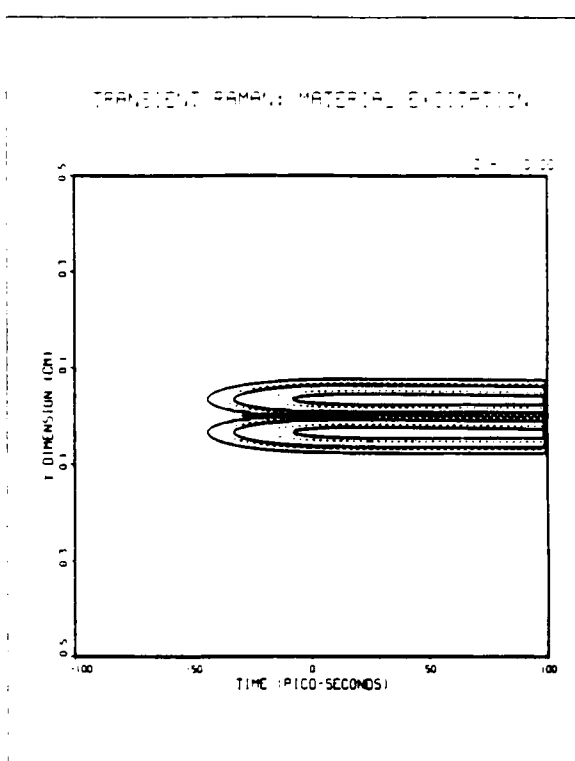
525

526

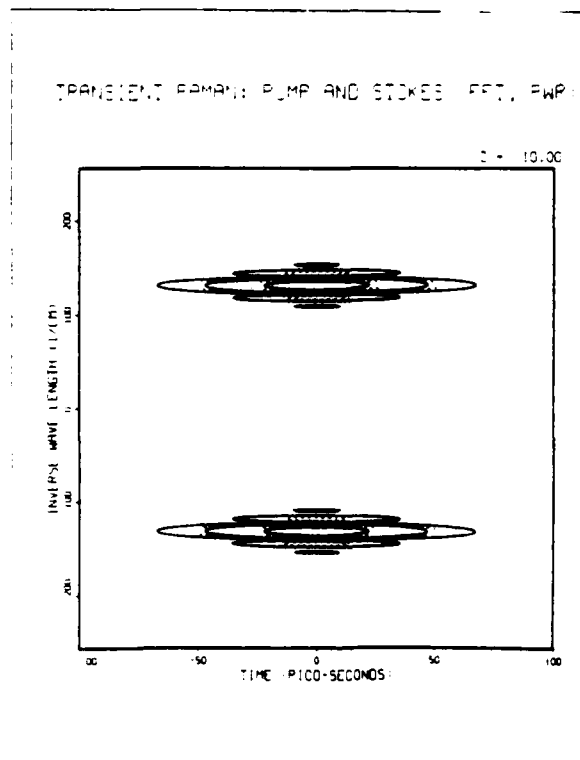
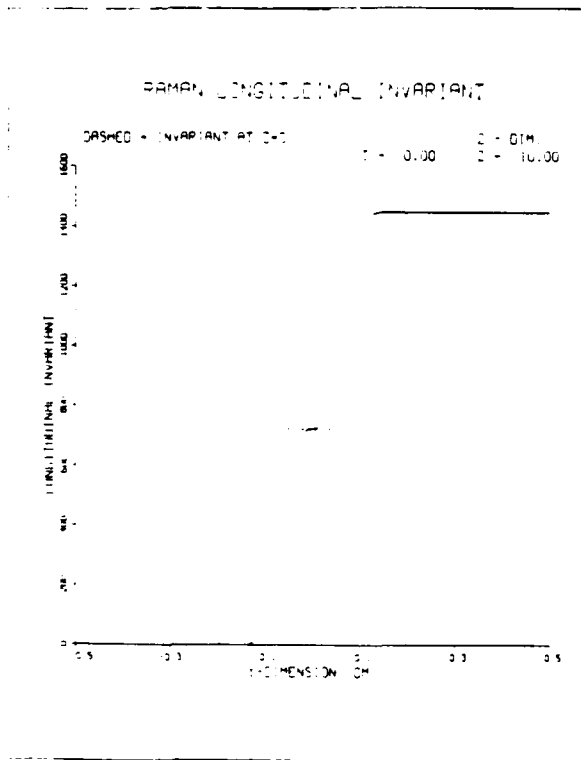
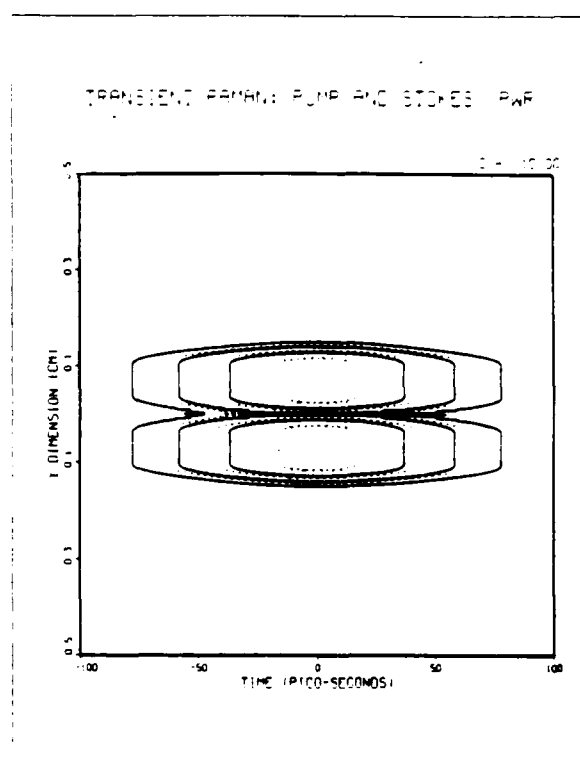
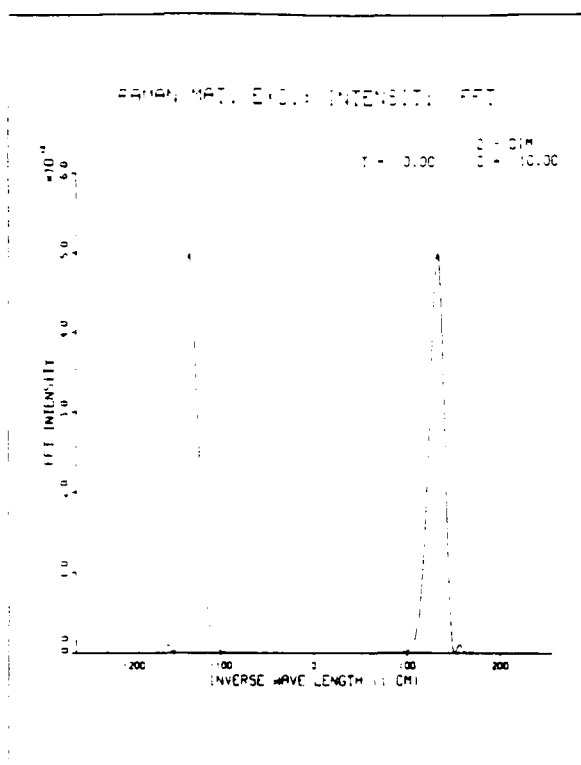
527

528

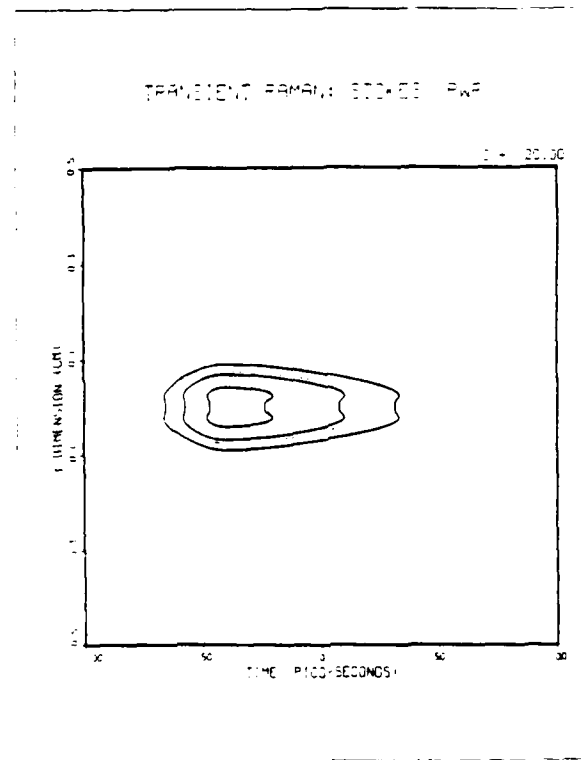
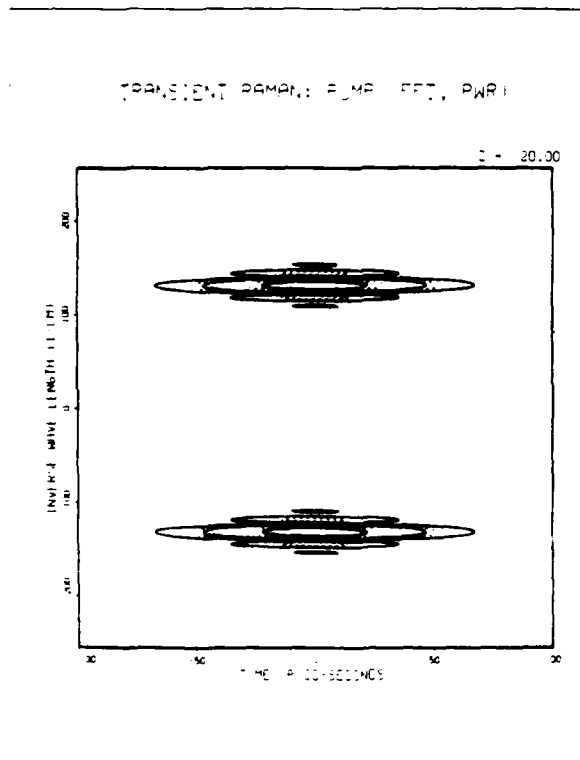
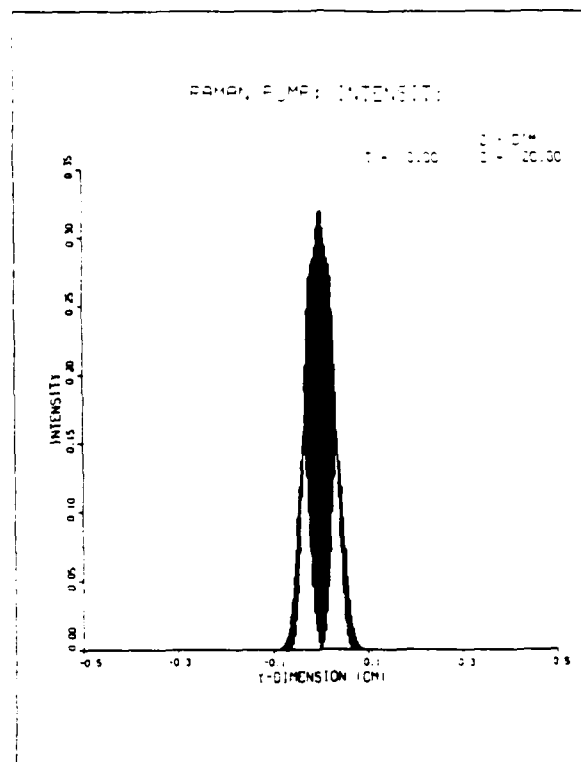
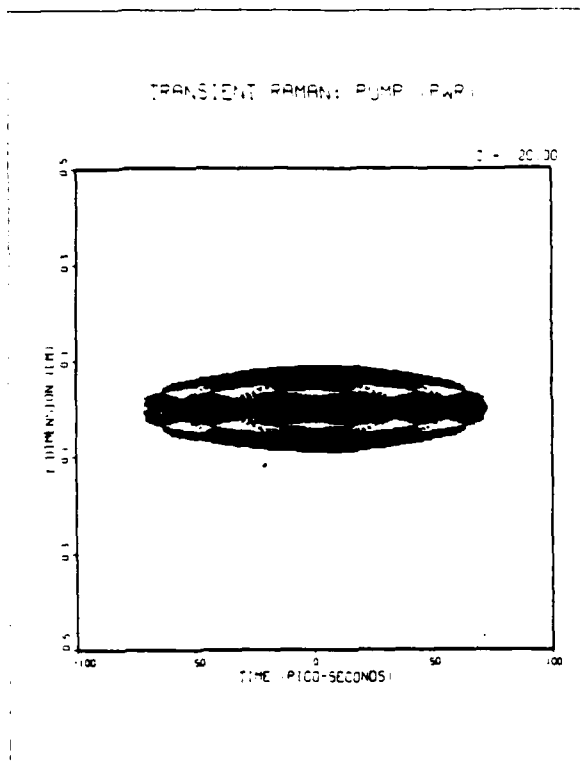
529



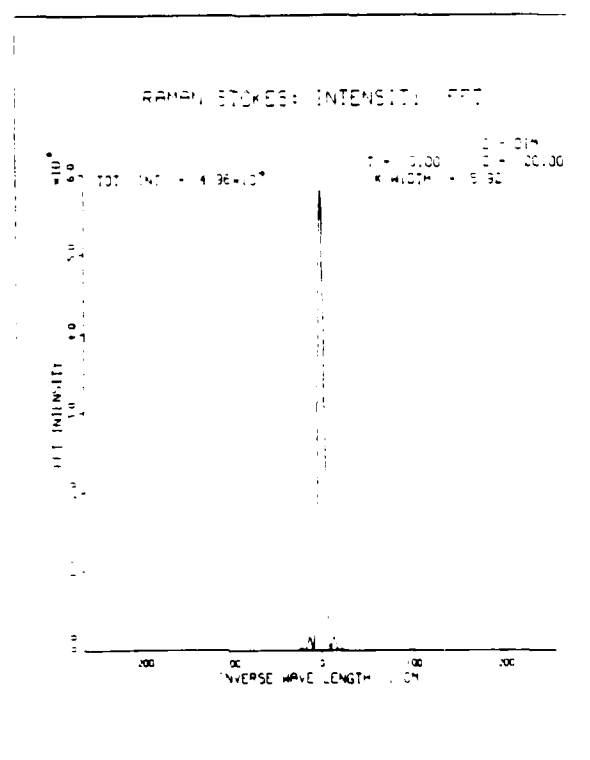
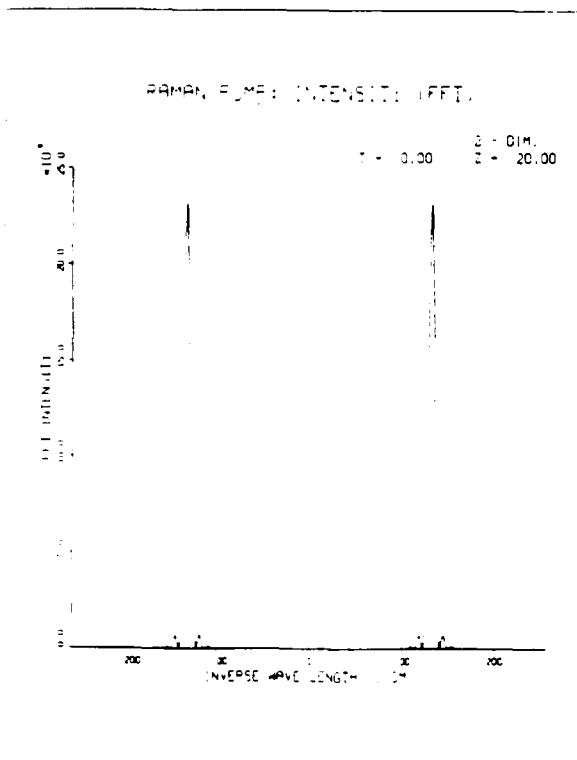
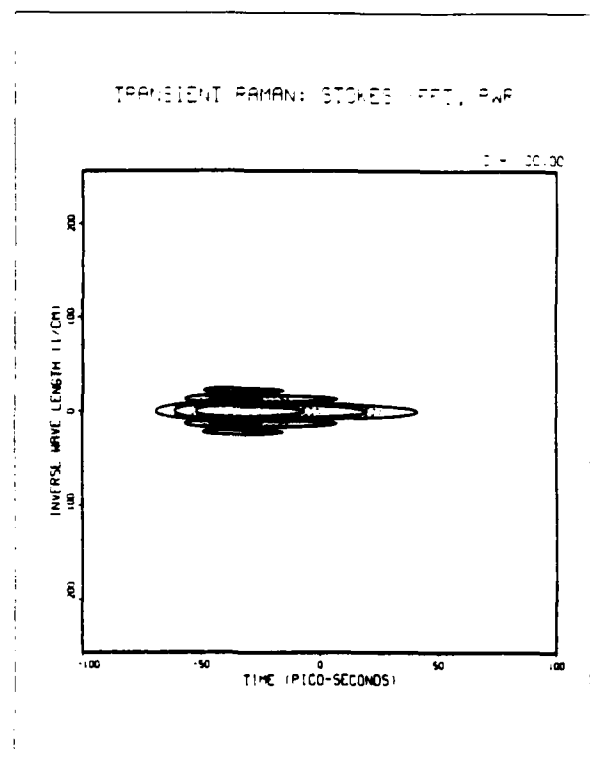
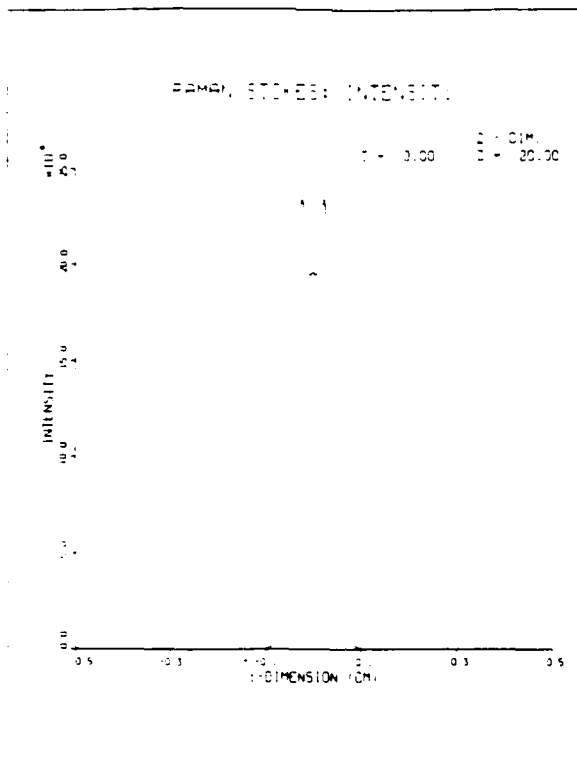
PLT2.DAT (Example C)



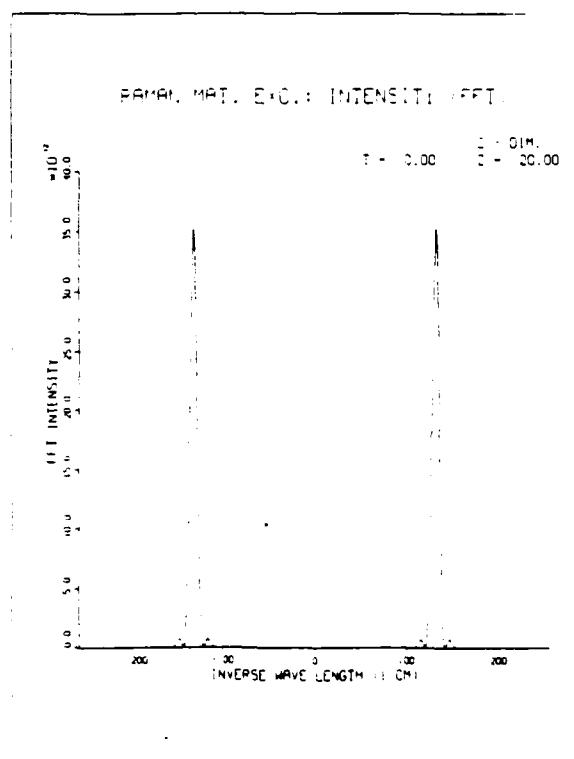
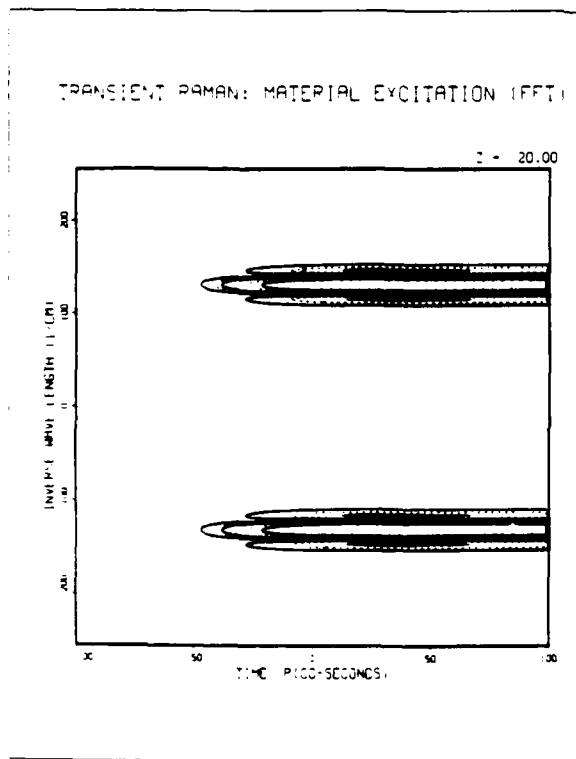
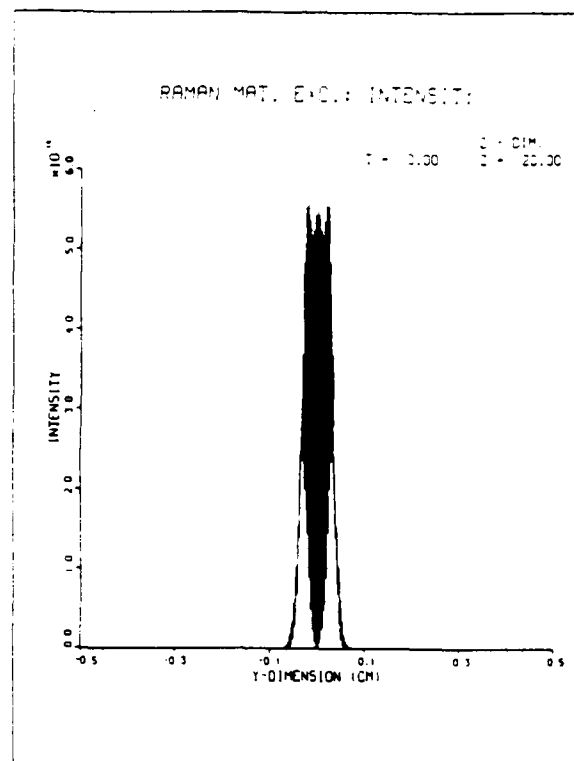
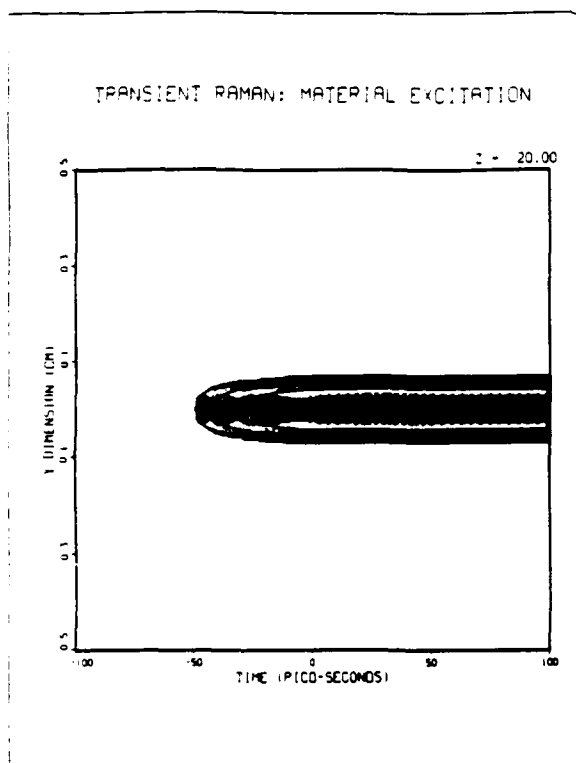
PLT2.DAT (Example C)



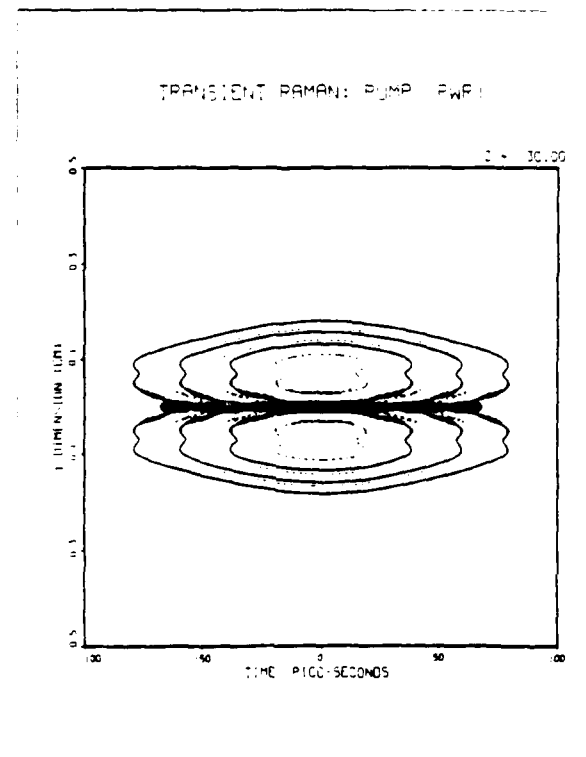
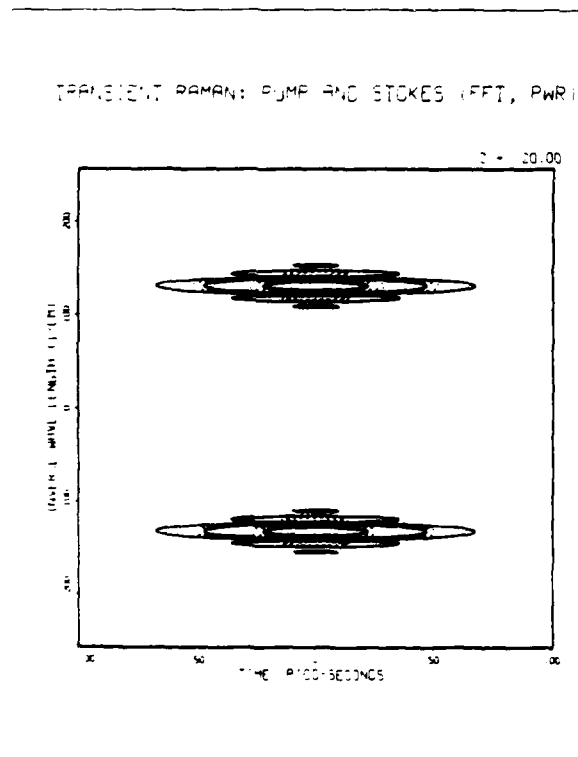
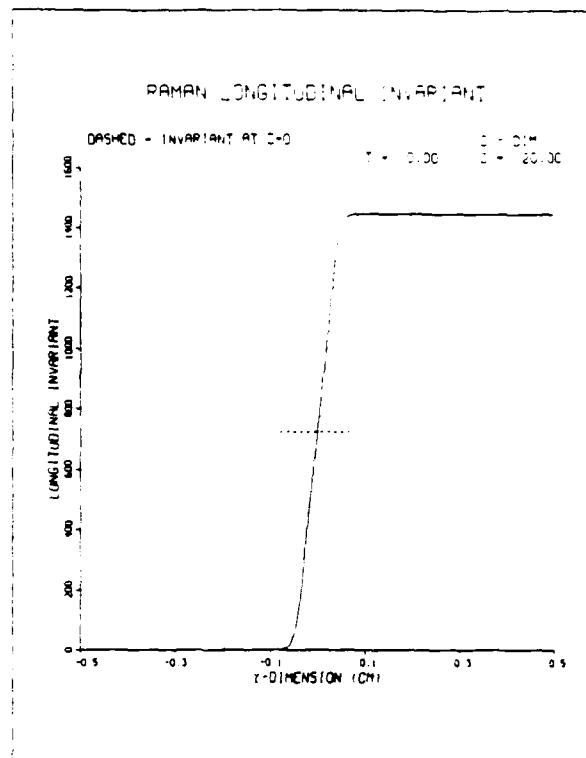
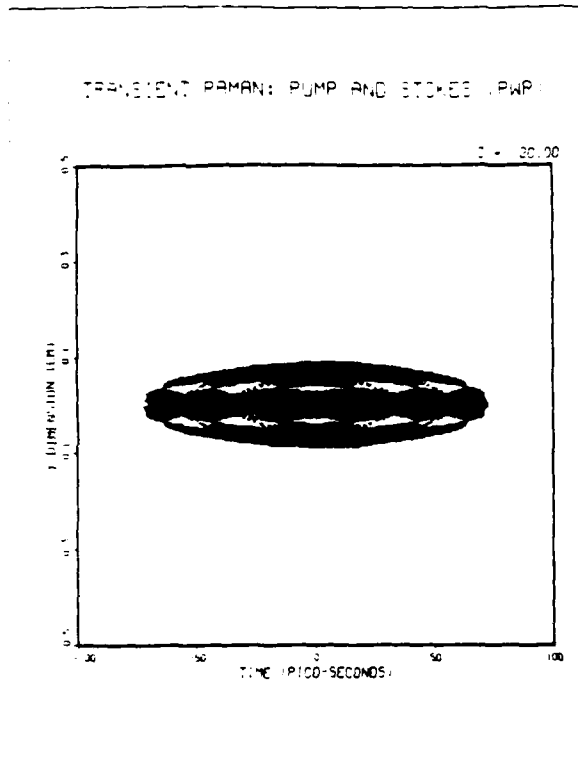
PLT2.DAT (Example C)



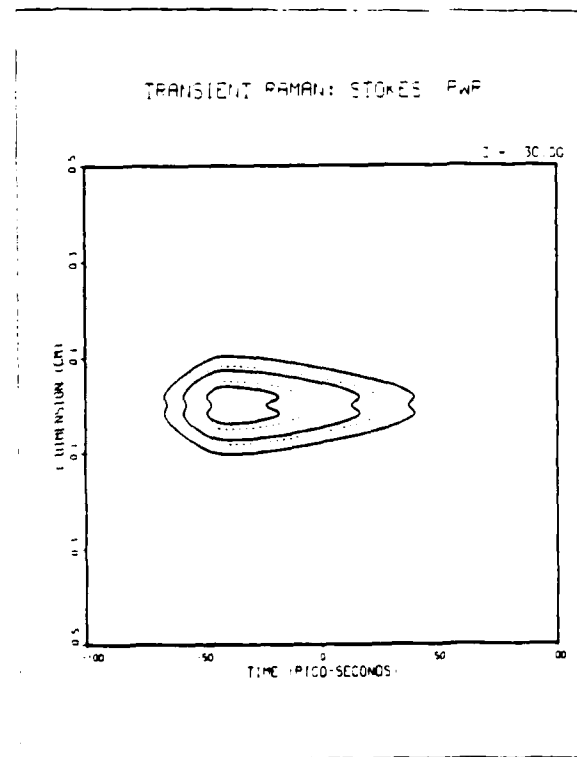
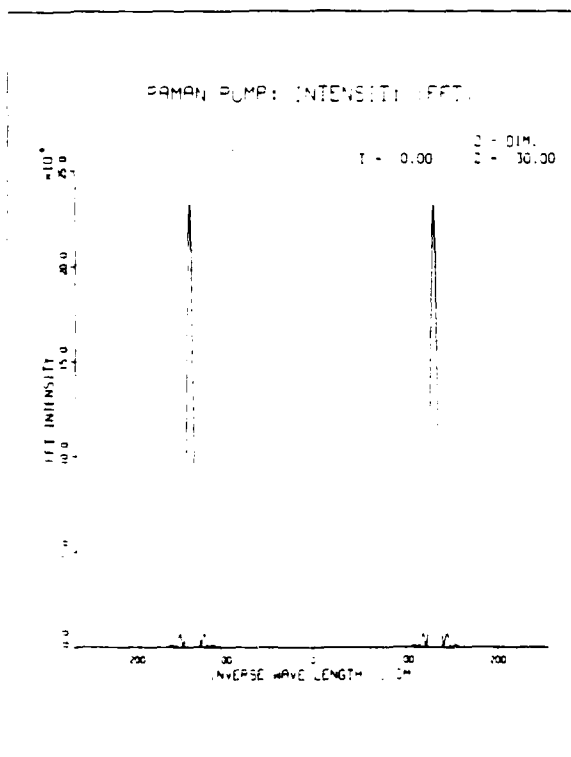
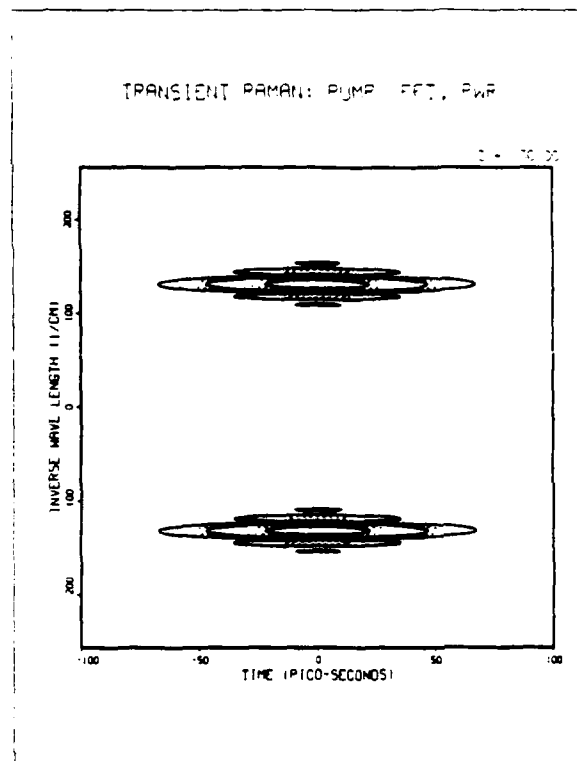
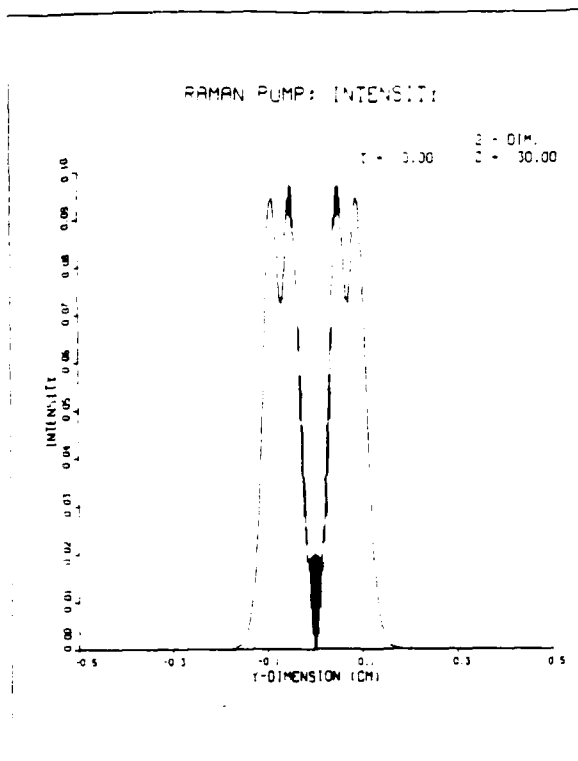
PLT2.DAT (Example C)



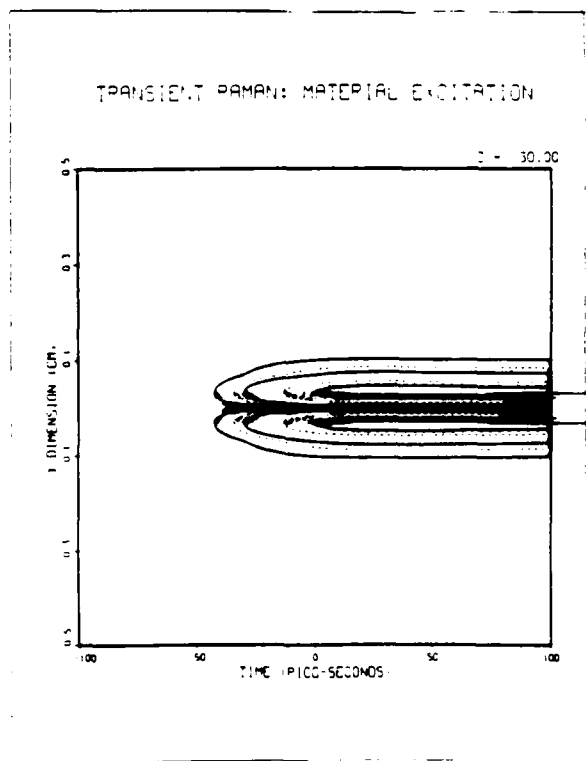
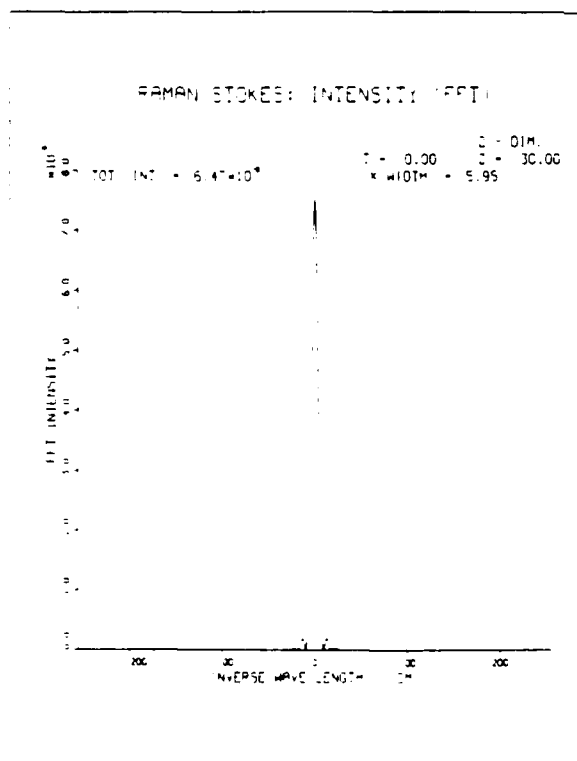
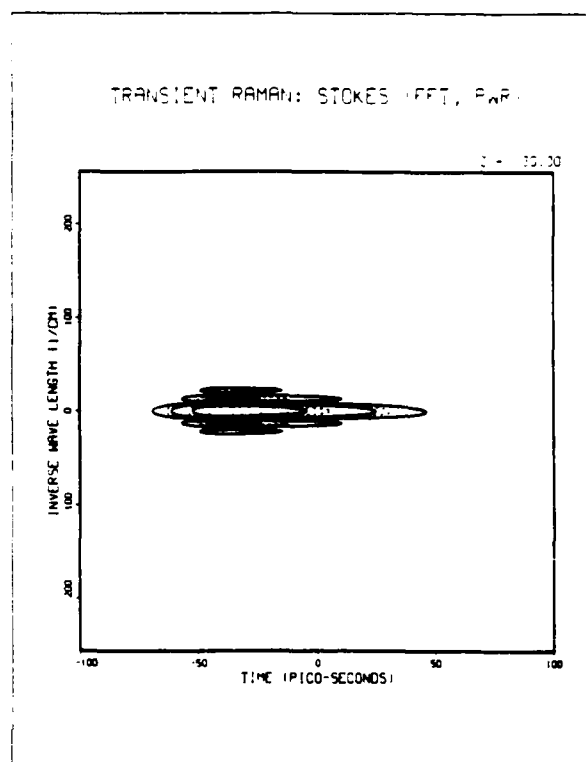
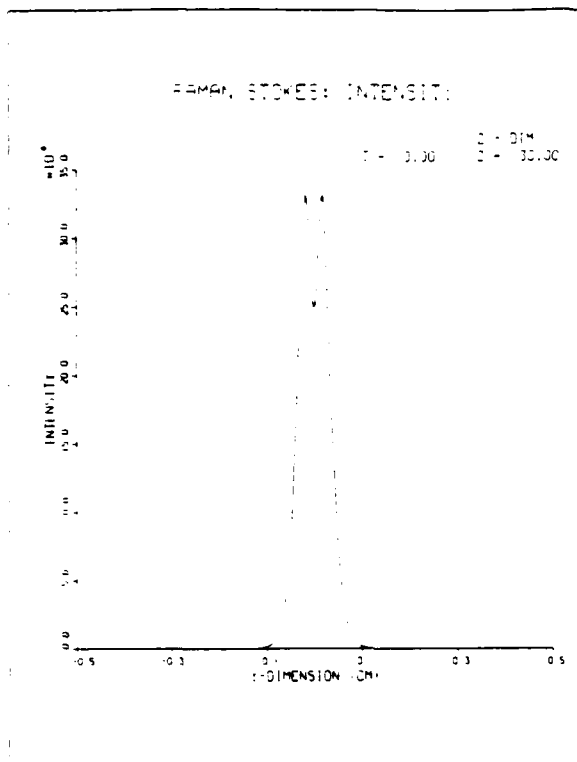
PLT2.DAT (Example C)



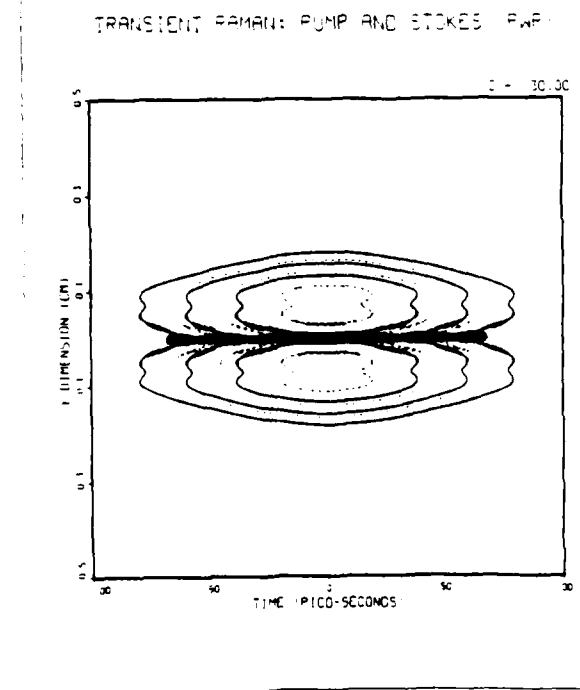
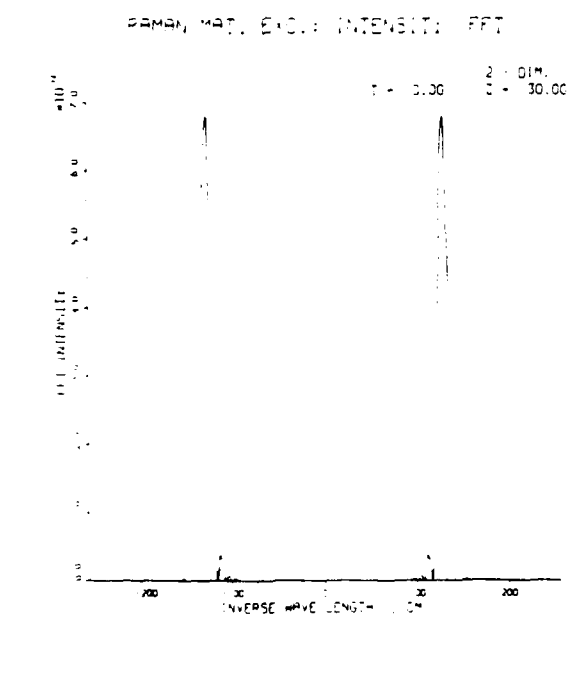
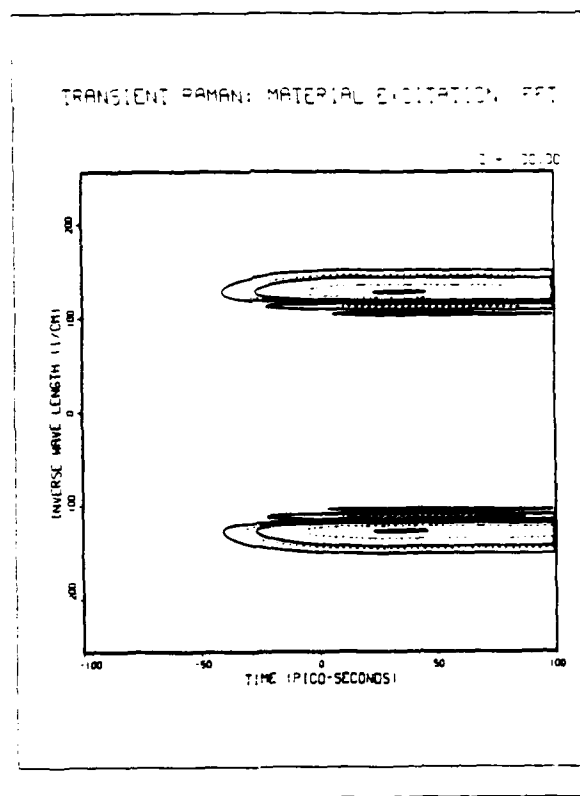
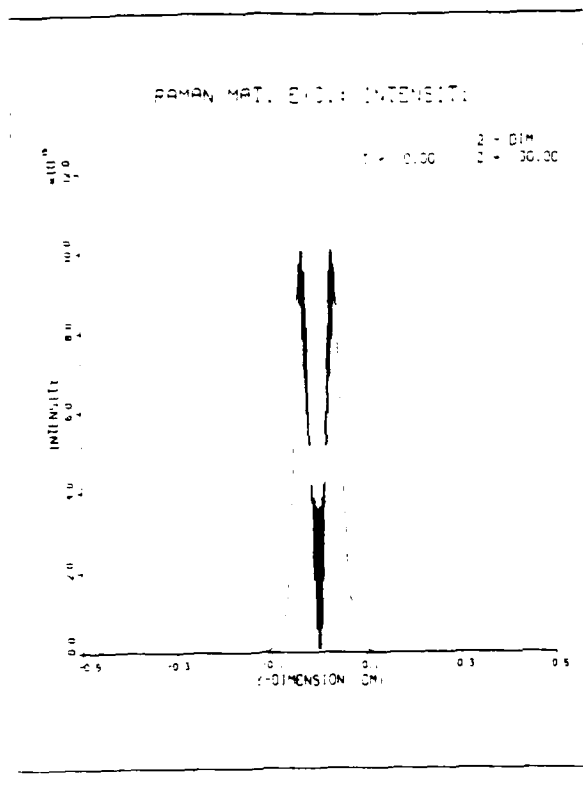
PLT2.DAT (Example C)



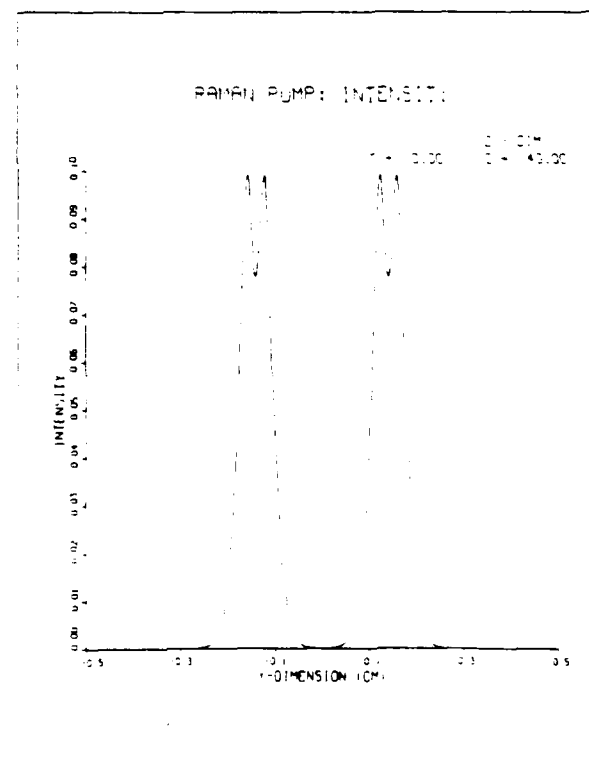
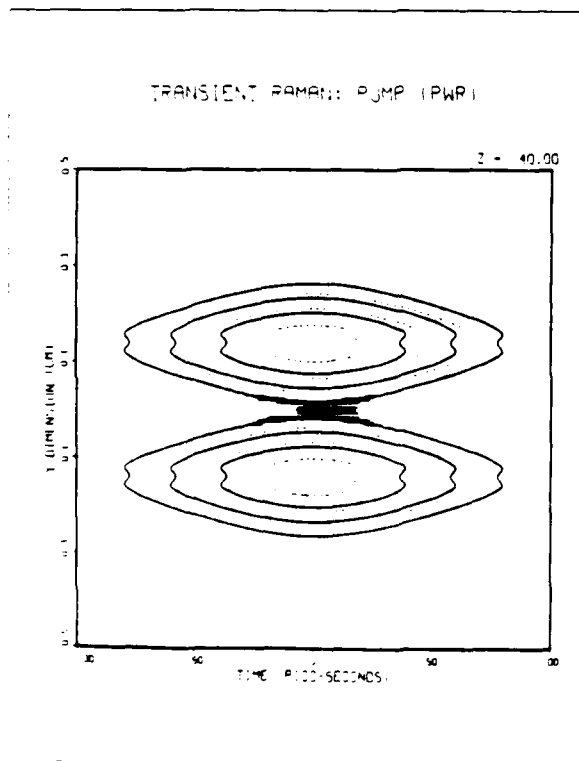
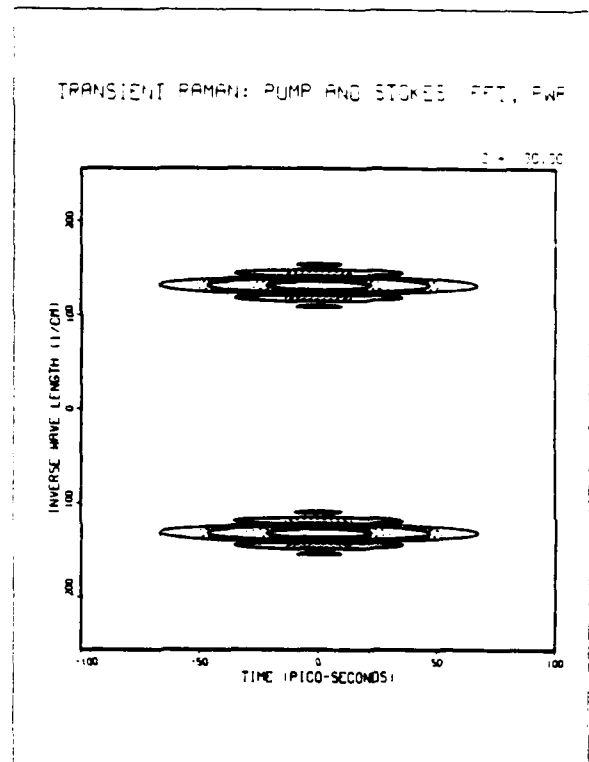
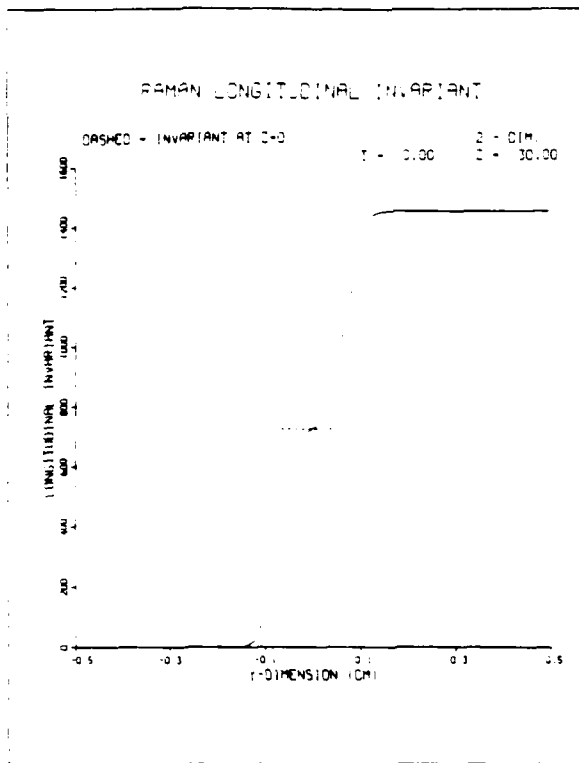
PLT2.DAT (Example C)



PLT2.DAT (Example C)

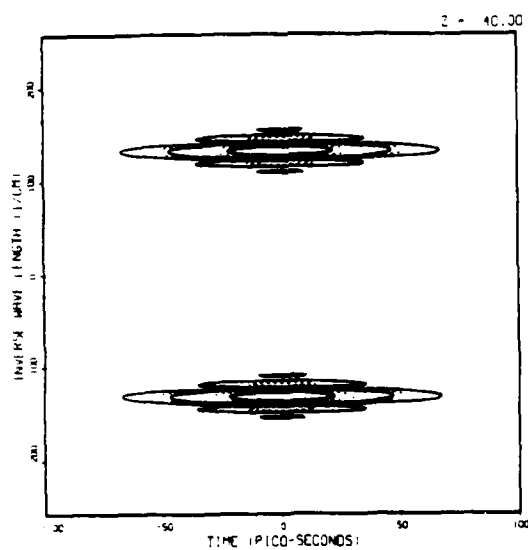


PLT2.DAT (Example C)

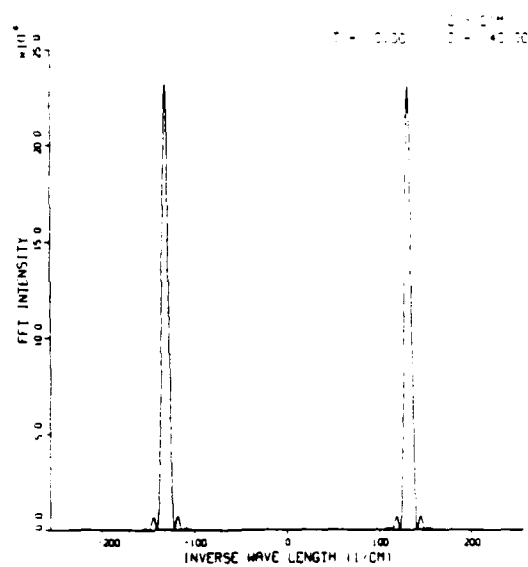


PLT2.DAT (Example C)

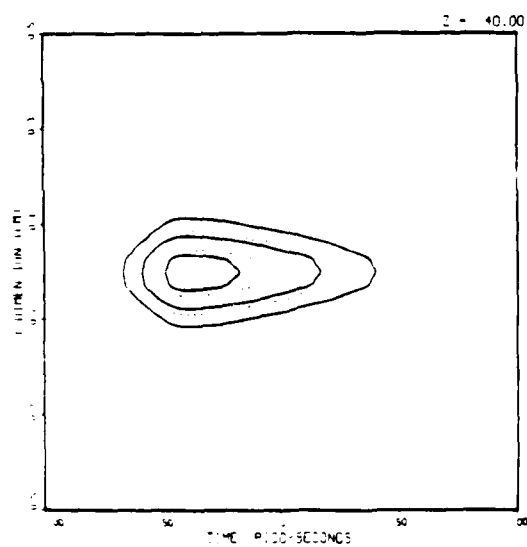
TRANSIENT RAMAN: PUMP (FFT, PWR)



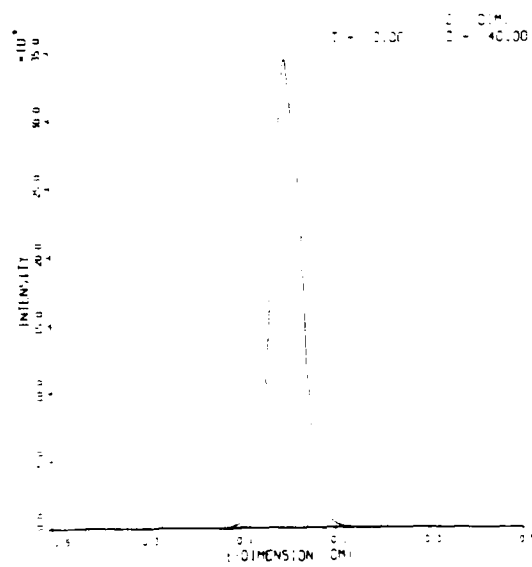
RAMAN PUMP: INTENSITY: FFT



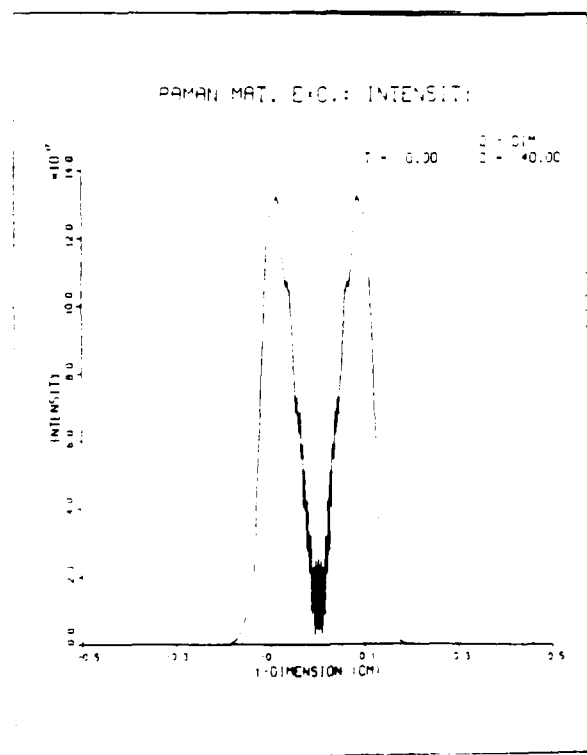
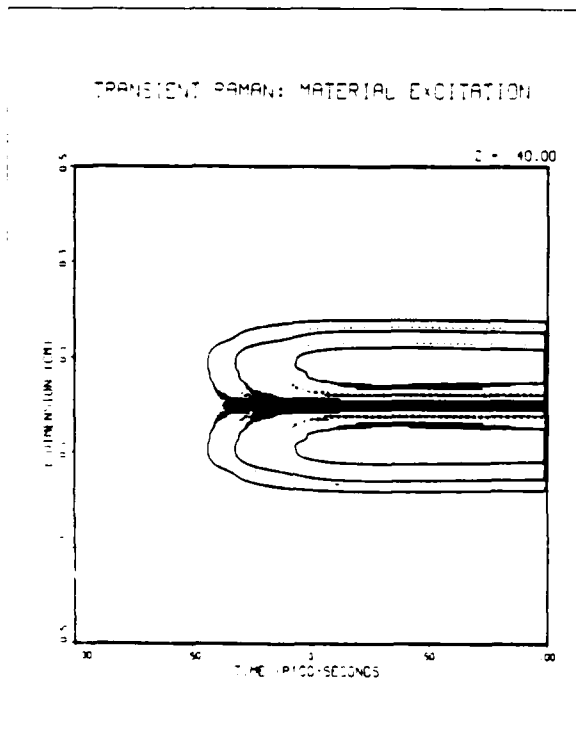
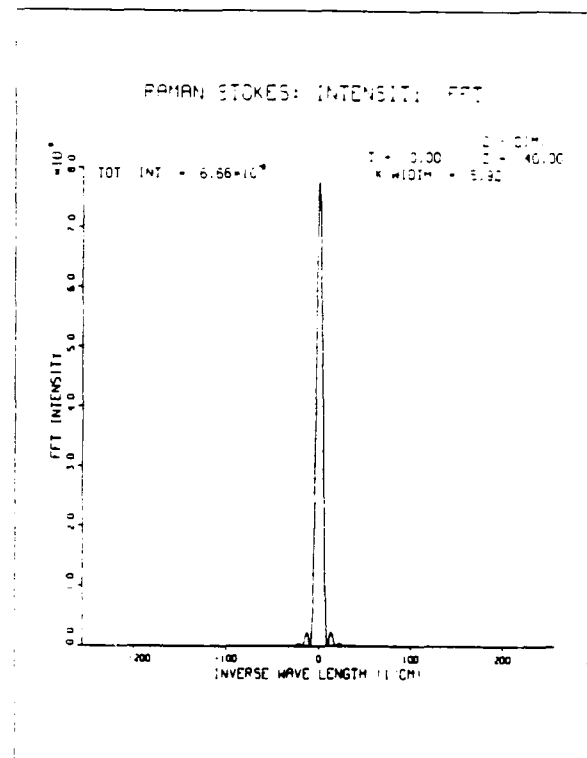
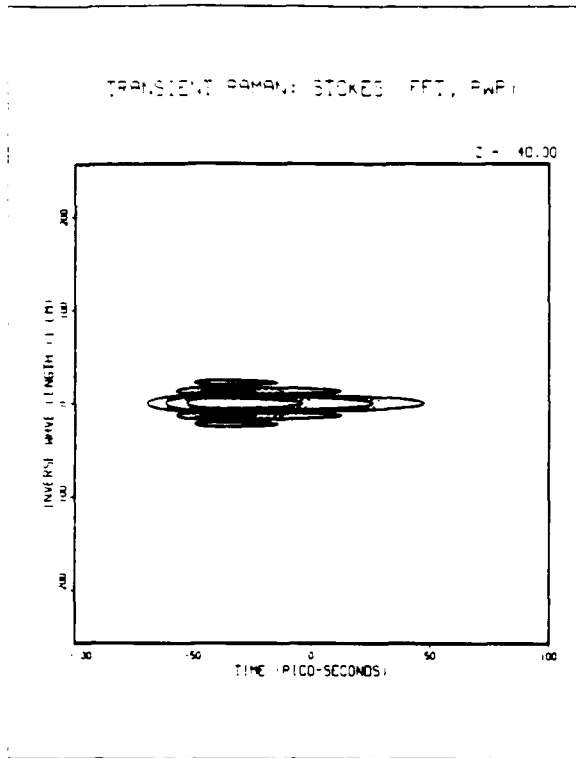
TRANSIENT RAMAN: STOKES (PWR)



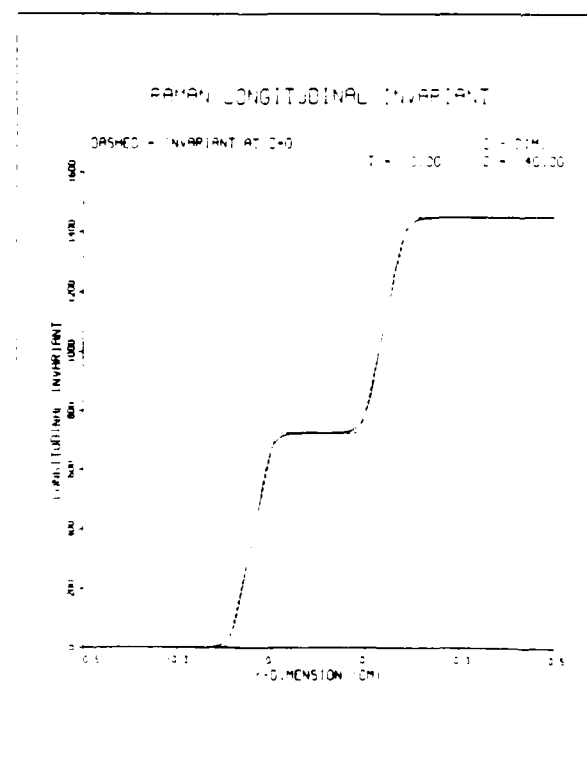
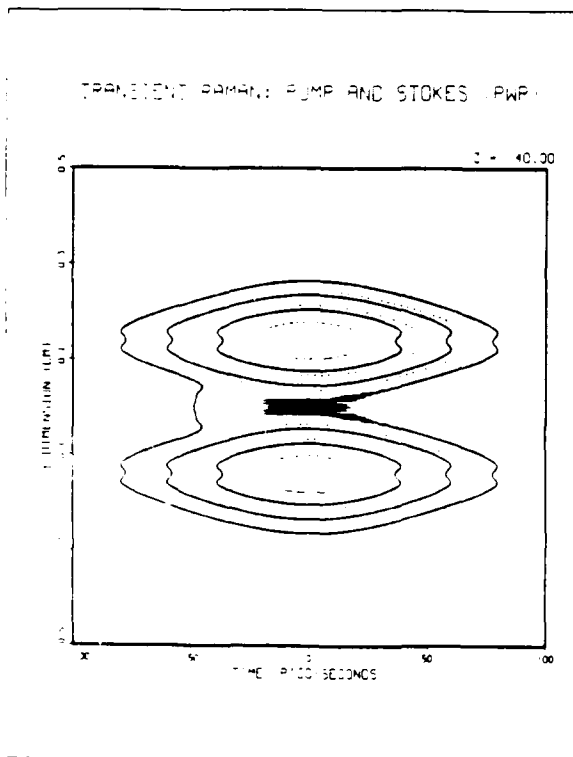
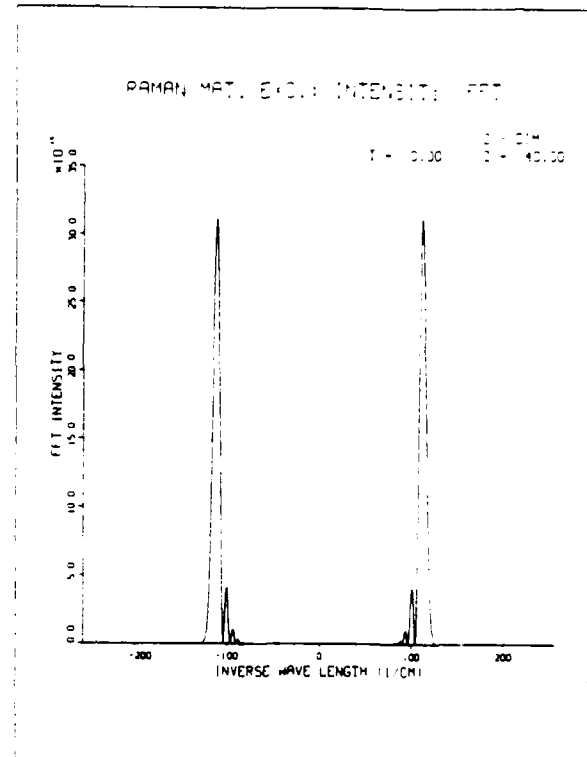
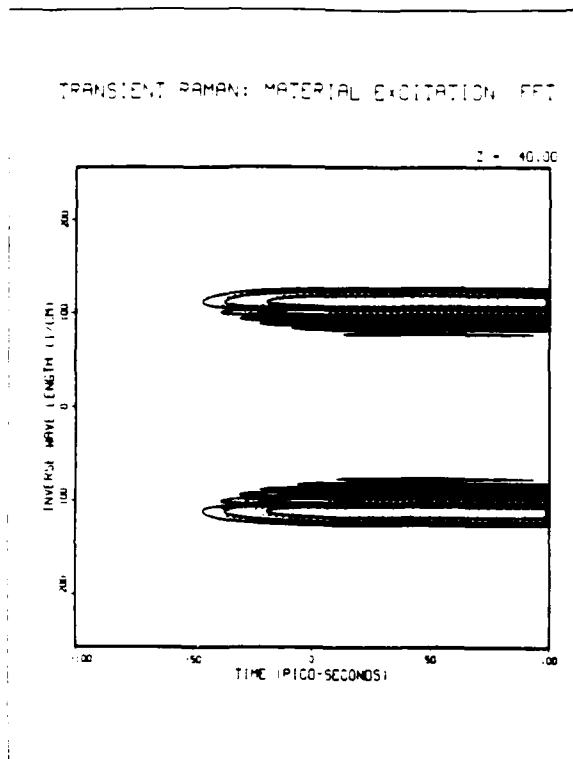
RAMAN STOKES: INTENSITY



PLT2.DAT (Example C)

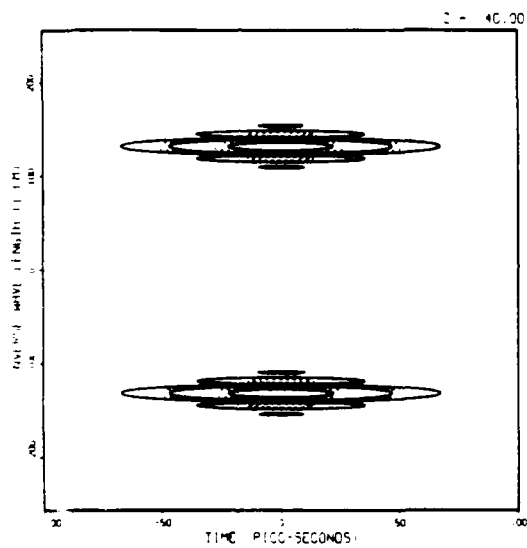


PLT2.DAT (Example C)



PLT2.DAT (Example C)

TRANSIENT RAMAN: PUMP AND STOKES FFT, PWP



APPENDIX D Special Versions of the Codes

It shall be mentioned that several special versions of the presented two programs RAM2D1 and PRAM1 exist. As was mentioned in section IV.B.5, the code RAM2D1C described in this manual is paralleled by the code RAM2D1D. The D version differs from the C version in that it does not keep the field arrays in memory during execution of the program, but ships them in from and out to storage disk as needed.

The versions A and B of RAM2D1 are the same as versions C and D in function, but different in form. They are adapted for use under the CTSS operating system as installed in the National Magnetic Fusion Energy Computing Center (NMFEECC) at the Lawrence Livermore National Laboratory (LLNL), CA, where the code was implemented first. The A and B version take the CIVIC FORTRAN compiler while the C and D versions ought to be compiled by the CFT compiler.

Since the data from either RAM2D1A or RAM2D1B have the same format, one version of the diagnostic program PRAM1 is sufficient. This is called PRAM1AB. In connection with the theoretical studies just mentioned, several special adaptations of PRAM1AB span off. These are PRAM1A, PRAM1B, PRAM1C, PRAM1D, and PRAM1E. The output data files from the NRL-based RAM2D1C and RAM2D1D are also identical in form and are diagnosed with PRAM1CD which is described in this manual.

A special version of RAM2D1A is called RMS1DT which is basically identical with RAM2D1A, but contains an additional block of code that is executed when the program runs in the one-dimensional transient limit. In this limit, an extra output file with time history data on the fields is created. This version was used for obtaining comparison with the results of a theory pertinent to strong pump depletion developed by the authors. The associate diagnostic program of RMS1DT is PRAM1E which plots the number of depletion holes in the pump and the ratio of initial to final pump energy. An expanded version of PRSE is PR1ENL which calculates and plots also other aspects of the analytical theory.

APPENDIX C

Publications

"Application of Lie methods to autonomous Hamiltonian perturbations of the Korteweg-de Vries equation: Second-order calculation," (C.R. Menyuk), in *Nonlinear Evolutions*, J.J.P. Léon, ed. (World Scientific Publ., Singapore, 1988), pp. 571-592.

APPLICATION OF LIE METHODS TO AUTONOMOUS HAMILTONIAN PERTURBATIONS OF THE KORTEWEG-DE VRIES EQUATION: SECOND ORDER CALCULATION

C.R. Menyuk
Department of Electrical Engineering
University of Maryland
Catonsville, MD 21228
and
College Park, MD 20742
USA

ABSTRACT

The Lie perturbation method of Hori and Deprit is a practical method for determining the evolution of nearly integrable finite-dimensional Hamiltonian systems. I show how to extend this approach to small, autonomous Hamiltonian perturbations of the Korteweg-de Vries equation. Explicit second order calculations are carried out in some simple cases where the initial data contains a single solitary wave and a small amount of radiation. We show explicitly that a solitary wave will emerge from the initial data through second order. This approach can be extended to arbitrarily high order.

I. INTRODUCTION

Nonlinear wave equations which can be integrated using spectral methods are quite special. Nonetheless, they play an important role in physics and have been used to model a wide variety of phenomena. Generally, these equations are derived by making a small parameter expansion of the underlying physical equations. At zeroth order, one obtains the linear equation. Moving into the wave frame, one obtains the integrable wave equation at first order. If this process is continued to higher order, one obtains corrections which in general destroy the equation's integrability^[1-4]. The two most experimentally important examples which lead to the Korteweg-de Vries equation are water waves in channels^[1,5,6] and ion acoustic waves in plasmas^[2,7,8]. In the first case, the underlying physical equation is Euler's

equation with appropriate boundary conditions, and small parameters include d/h , the height of the pulse divided by the height of the channel, and h/l , the height of the channel divided by the length of the pulse. In the second case, the underlying equations are the two fluid equations with inertia-less electrons and constant temperatures. Small parameters include $\delta n/n$, the size of the ion density pulse divided by the undisturbed ion density, and T_i/T_e , the ratio of the ion and electron temperatures.

Experimentally, one finds that these small parameters can be quite large, as large as 0.3–0.4 using standard normalizations, and solitons (or more precisely solitary waves) are seen to emerge from an initial pulse just as if the system was integrable; however, their widths and velocities are related to their heights somewhat differently than in the Korteweg-de Vries equation^[5–8]. By contrast, relatively small dissipative perturbations—or perturbations that vary in space and time—are sufficient to destroy the integrable-appearing behavior.

In the past I have tried to provide qualitative insight into this behavior by showing that the higher order corrections yield Hamiltonian perturbations that do not vary in space and time^[3,4], i.e. autonomous Hamiltonian perturbations, and that under certain conditions, which the experiments reproduce reasonably well, solitary waves emerge to all orders in the small parameters^[9–12]. There are important restrictions: First, an asymptotic theory in which secularities are removed order by order can only be carried out once the solitons corresponding to the poles of the spectral data are well-separated. Previous to this separation, the solitary waves interact and continuum radiation and even new solitary waves can be produced. By reducing the perturbation strength, the amplitudes of any new solitary waves produced and their number can be bounded. A second restriction is that the theory in its present form is non-uniform in x , the coordinate space. As a consequence, the possibility cannot be ruled out that a portion of the continuum “to all orders” might actually be a low, broad solitary wave “beyond all orders.” The converse also holds. A third restriction is that we consider initial data which falls off faster than some exponential as $x \rightarrow \pm\infty$ and which is analytic in some strip surrounding the real axis in complex x -space.

Despite these restrictions, it is my hope that this perturbative approach will prove quantitatively useful in the long run. While it is simpler to determine solitary wave solutions by looking for stationary solutions of the equations, such an approach does not allow one to determine the amplitude(s) of the solitary wave(s) which will

ultimately emerge from given initial data. The approach presented here does.

In this work, I will be concentrating on the experimentally important case where the initial data contains only a single solitary wave, or, more precisely, only a single pole in the transmission data. We thus avoid any problems related to solitary wave interactions. I will be using a Hamiltonian approach, specifically the Lie approach first developed by Hori^[13] and Deprit^[14]. I use a Hamiltonian approach, rather than a more general approach such as that of Karpman and Maslov^[15] or Kaup and Newell^[16], because it allows one to concentrate in a natural way on the autonomous Hamiltonian systems and obtain results which only apply to them. I use the Lie approach rather than the Poincaré-von Zeipel approach because the Lie approach is now generally considered the simpler of the two to use^[17].

I will be concentrating in this paper on perturbed Hamiltonians of the form

$$H[u] = H_0[u] + \epsilon H_1[u], \quad (1)$$

where

$$\begin{aligned} H_0[u] &= \int_{-\infty}^{\infty} dx \left(u^3 + \frac{u_x^2}{2} \right), \\ H_1[u] &= \int_{-\infty}^{\infty} dx u^p, \end{aligned} \quad (2)$$

with $p = 2, 3, 4$ and 5 . Using the Poisson bracket

$$[F, G] = \int_{-\infty}^{\infty} dx \left(\frac{\delta F}{\delta u} \frac{\partial}{\partial x} \frac{\delta G}{\delta u} \right), \quad (3)$$

we find

$$u_t = [u, H] = 6uu_x - u_{xxx} + \epsilon p(p-1)u^{p-2}u_x, \quad (4)$$

which is just the Korteweg-de Vries equation with a small perturbation. The case $p = 2$ corresponds to a Galilean transformation; the case $p = 3$ corresponds to a change in the nonlinear coefficient of the Korteweg-de Vries equation; and the case $p = 4$ corresponds to a Miura transformation. The case $p = 5$ produces a non-integrable system. For these relatively simple examples, I will calculate the perturbed Hamiltonian through second order, as well as the perturbed potential u through first order.

In previous work, I have studied other simple examples, but only through first order^[10]. My motivation for carrying out explicit second order calculations is that

some of my colleagues, notably Yuji Kodama, felt that this calculation would be very useful in clarifying the basic structure of the theory by showing in detail how to avoid secularities, as must be done in any Hamiltonian theory. One must divide the Hamiltonian between its coordinate-independent and coordinate-dependent pieces and group the former with the zero-order Hamiltonian at each order. The first non-trivial order at which this separation must be carried out is second order.

In previous work I have primarily employed Hamiltonian perturbation theory to show, with the limitations described earlier, that solitons emerge from arbitrary initial data to all orders in the small parameter for a large class of Hamiltonian perturbations^[11,12]. The goal was to obtain insight into the experimentally observed robustness of solitons^[5,8]. In this respect my work was motivated by Martin Kruskal's classic study of the theory of adiabatic invariants^[18], and, in my opinion, the approach and conclusions are conceptually similar. Nonetheless, it is my hope that this approach will ultimately prove useful in carrying out detailed quantitative comparisons between theory and experiment, much as it has proved useful in the study of satellite orbits about the Earth^[13,14], and particle motion in accelerators^[17]. In order for the theory to be useful in this regard, one must be comparing to experiments where the perturbations are quite small, the distances quite large, and the measurements quite precise. While experiments modelled by field equations do not seem to fulfill these conditions at present. It is my belief that they will do so within the next twenty years.

The remainder of this paper is organized as follows: In Section II, we specify the action-angle transformation from $u(x)$ to $[p(k), q(k), p_\alpha, q_\alpha]$, the canonical variables which evolve linearly in time. In Section III, we show how to write the Hamiltonian in terms of the canonical variables. In Section IV, we show how to obtain the lowest order Lie generator and discuss the problem of small denominators. Section V contains the explicit calculation of the second order perturbed Hamiltonian and includes the determination of the second order Lie generator. In Section VI, we calculate the first order potential. Section VII contains the conclusions and acknowledgments.

II. ACTION-ANGLE TRANSFORMATION

Before I can apply Hamiltonian methods to the perturbed system, I must determine action-angle variables for the underlying integrable system. Quite generally,

we may write the original coordinates as u , where $u = u_i$ in a system with a discrete number of independent variables and $u = u(x)$ in a system with an uncountably infinite number of degrees-of-freedom. We are interested here in the latter case. We are searching for an invertible transformation of the form

$$u \rightarrow (J, \theta), \quad (5)$$

where J and θ represent the ensemble of action and angle variables; in general, J and θ can have both uncountably infinite and discrete components. The set of variables J and θ should be canonically related to each other, and the Hamiltonian should only depend on the action variables J .

The equations of motion in terms of these new variables is trivially integrable. Writing Ω for the ensemble of frequency variables, i.e.

$$\Omega_i = \partial H[J] / \partial J_i \quad \text{and} \quad \Omega(k) = \partial H[J] / \partial J(k), \quad (6)$$

for the discrete and continuous components, it follows that

$$\begin{aligned} J &= J_0, \\ \theta &= \theta_0 + \Omega t, \end{aligned} \quad (7)$$

where J_0 and θ_0 represent the ensemble of initial conditions. At any time, we may determine $u(x)$ by inverting the action-angle transformation. This situation is shown schematically in Fig. 1. If we take the direct, left-hand path shown as a dashed arrow and integrate the equations of motion using a computer, it is generally necessary to take many small time steps. By contrast, if one integrates the equations of motion using the solid, three-sided path, one can carry out the time integration in one fell swoop. Hence, no matter how complicated the backward and forward transformations, one always "wins" using the three-sided approach over a sufficiently long time interval. I note that this notion of "winning," while useful, is not precise, something which can also be said of the notion of integrable systems. For linear, infinite-dimensional systems, the three-sided path represents Fourier integration; for nonlinear, infinite-dimensional systems, it represents an analogous nonlinear transformation.

The appropriate action-angle transformation when the underlying integrable system is the Korteweg-de Vries equation, was first found by Zakharov and Fadeev^[19]. They begin by making the spectral transformation^[19]

$$u(x) \rightarrow [r(k), \kappa_j, c_j], \quad (8)$$

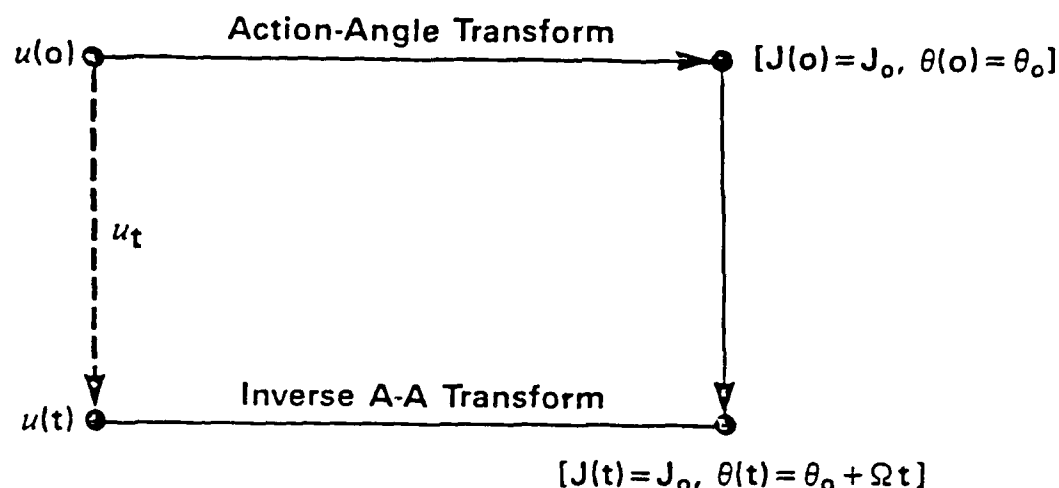


FIGURE 1. Schematic illustration of the way in which an action-angle transform and its inverse can be used to solve the equations of motion of a completely integrable system.

and then converting these variables into the appropriate action-angle form. I will make a slightly different choice of variables from theirs which proves to be useful in what follows. Concentrating on the case where the initial data contains a single soliton, my choice is

$$\begin{aligned} p(k) &= -\frac{2k}{\pi} \ln[1 - r(k)r(-k)], \\ q(k) &= -\frac{1}{2i} \ln[r(k)/r(-k)], \\ p_\alpha &= \frac{4}{3} \kappa_\alpha^3, \\ q_\alpha &= \frac{1}{\kappa_\alpha} \ln c_\alpha, \end{aligned} \tag{9}$$

where the subscript α indicates the single soliton. The Hamiltonian becomes

$$H_0 = \int_0^\infty dk \, 8k^3 p(k) - \frac{12 \cdot 3^{2/3}}{5 \cdot 2^{1/3}} p_\alpha^{5/3}, \tag{10}$$

which depends only the action variables, and the Poisson bracket becomes

$$\begin{aligned} [F, G] &= \int_0^\infty dk \left[\frac{\partial F}{\partial q(k)} \frac{\partial G}{\partial p(k)} - \frac{\partial F}{\partial p(k)} \frac{\partial G}{\partial q(k)} \right] \\ &\quad + \frac{\partial F}{\partial q_\alpha} \frac{\partial G}{\partial p_\alpha} - \frac{\partial F}{\partial p_\alpha} \frac{\partial G}{\partial q_\alpha}, \end{aligned} \tag{11}$$

which is canonical in form. I take the k integrals over the half interval $[0, \infty)$, rather than the full interval $(-\infty, \infty)$, which helps in keeping track of the cross terms which appear in the theory between $-k$ and k . These integrals must eventually be extended first over the full interval and then into the upper half plane. The soliton variables q_α and p_α are related to those used by Zakharov and Fadeev^[19] by a simple canonical transformation.

I close this section with an important aside. The canonical form of the Poisson bracket in Eq. (11) follows from the form of the Poisson bracket in Eq. (3). Once the Korteweg-de Vries equation is perturbed, it is not evident *a priori* that this symplectic structure will suffice at all orders. Recently, H. H. Chen and I^[3,4] have shown that it does indeed suffice in cases where the physically important systems of one-dimensional ion acoustic waves or shallow channel water waves are being considered.

III. DETERMINING THE HAMILTONIAN

My next task is to re-express the Hamiltonian, $H[u] = H_0[u] + \epsilon H_1[u]$ explicitly in terms of the canonical variables. Zakharov and Fadeev have showed us how to obtain an explicit expression for $H_0[u]$, and the result is given by Eq. (10). Their procedure, however, cannot be applied to the general case where the Hamiltonian $H_1[u]$ will depend on the canonical coordinates as well as the momenta. Instead, I directly calculate $H_1[u]$. I begin by determining u in terms of the canonical variables. In the case of interest here where u contains a single solitary wave (or more precisely the transmission coefficient has a single pole at the initial time), I write

$$u = u_\alpha + 2 \frac{d}{dx} K(x, x), \quad (12)$$

where

$$u_\alpha = -2\kappa_\alpha^2 \operatorname{sech}^2[\kappa_\alpha(x + q_\alpha/2)]$$

is the single soliton solution and $K(x, y)$ is given by the solution to the Marchenko equation

$$K(x, y) + F(x, y) + \int_{-\infty}^x dz K(x, z) F(z, y) = 0. \quad (13)$$

The kernel $F(x, y)$ is given by the relation

$$F(x, y) = \int_{-\infty}^{\infty} \frac{dk}{2\pi} r(k) g_\alpha(x, k) g_\alpha(y, k), \quad (14)$$

where

$$g_\alpha(x, k) = \exp(-ikx) \left(\frac{k - i\kappa_\alpha \tanh[\kappa_\alpha(x + q_\alpha/2)]}{k + i\kappa_\alpha} \right) \quad (15)$$

is the left Jost function corresponding to a single soliton potential. The Neumann expansion of Eq. (13)

$$K(x, y) = -F(x, y) + \int_{-\infty}^x dz F(x, z) F(z, y) - \dots, \quad (16)$$

is always convergent. This Neumann expansion is essentially an expression in powers of $r(k)$. Writing now,

$$u = u_\alpha + u_1 + u_2, \quad (17)$$

through second order in powers of $r(k)$, and letting

$$\xi = x + q_\alpha/2, \quad (18)$$

I find explicitly

$$u_1 = 4 \int_{-\infty}^{\infty} \frac{dk}{2\pi} r(k) \exp(ikq_\alpha) \frac{\exp(-2ik\xi)}{k + i\kappa_\alpha} \sum_{j=0}^1 \left[a_j \operatorname{sech}^{2j}(\kappa_\alpha \xi) + b_j \operatorname{sech}^{2j}(\kappa_\alpha \xi) \tanh(\kappa_\alpha \xi) \right], \quad (19a)$$

$$u_2 = -2 \int_{-\infty}^{\infty} \frac{dk_1}{2\pi} r(k_1) \int_{-\infty}^{\infty} \frac{dk_2}{2\pi} r(k_2) \frac{\exp[i(k_1 + k_2)q_\alpha] \exp[-2i(k_1 + k_2)\xi]}{(-i)(k_1 + i\kappa_\alpha)^2 (k_2 + i\kappa_\alpha)^2} \sum_{j=0}^2 \left[c_j \operatorname{sech}^{2j}(\kappa_\alpha \xi) + d_j \operatorname{sech}^{2j}(\kappa_\alpha \xi) \tanh(\kappa_\alpha \xi) \right], \quad (19b)$$

where

$$\begin{aligned} a_0 &= ik(k^2 - \kappa_\alpha^2), \\ b_0 &= 2k^2 \kappa_\alpha, \\ a_1 &= 2ik\kappa_\alpha^2, \\ b_1 &= \kappa_\alpha^3, \\ c_0 &= 2i(k_1 + k_2)[(k_1 k_2 - \kappa_\alpha^2)^2 - (k_1 + k_2)^2 \kappa_\alpha^2], \\ d_0 &= 4(k_1 + k_2)^2 (k_1 k_2 - \kappa_\alpha^2) \kappa_\alpha, \\ c_1 &= 2i(k_1 + k_2)[(k_1 + k_2)^2 + (2k_1 k_2 - 3\kappa_\alpha^2)] \kappa_\alpha^2, \\ d_1 &= 2[2(k_1 + k_2)^2 + (k_1 k_2 - \kappa_\alpha^2)] \kappa_\alpha^3, \\ c_2 &= 3i(k_1 + k_2) \kappa_\alpha^4, \\ d_2 &= 0. \end{aligned} \quad (20)$$

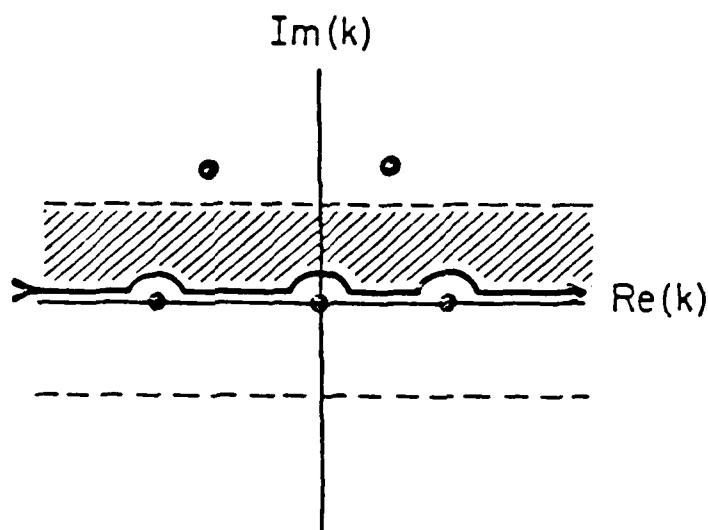


FIGURE 2. Schematic illustration of an integration path through the upper half Bargmann strip, represented by the hatched region. The dots represent poles.

In calculating Eq. (19b), I had to exchange an integral of the form $\int_{-\infty}^z dz \dots$ with two integrals of the form $\int_{-\infty}^{\infty} dk \dots$. The integral over z was then explicitly evaluated. This exchange is only valid when $\text{Im}(k) > 0$, i.e., when the integrals over k are performed in the upper half Bargmann strip shown schematically as the hatched area in Fig. 2. While quite simple conceptually, this result is important as it tells me how to integrate around the poles which appear in the theory on the real k -axis.

At this point, I determine

$$H_1[u] = \int_{-\infty}^{\infty} u^p d\xi \quad (21)$$

in terms of the canonical variables. To do so, I need to exchange the integral $\int_{-\infty}^{\infty} d\xi \dots$ with the integrals over k . This exchange is not permitted unless the integrand decreases exponentially as $\xi \rightarrow +\infty$ which is not always the case. When that is not the case, we may pick up a δ -function contribution. To show how this works, I first consider the case $p = 2$. In this case

$$u^2 = u_{\alpha}^2 + 2u_{\alpha}u_1 + u_1^2 + 2u_2u_{\alpha}; \quad (22)$$

so, writing

$$H_1 = h_0 + h_1 + h_2, \quad (23)$$

where I have expanded H_1 in powers of $r(k)$ through second order, I find

$$h_0 = \int_{-\infty}^{\infty} u_{\alpha}^2 d\xi = 4\kappa_{\alpha}^4 \int_{-\infty}^{\infty} \text{sech}^4(\kappa_{\alpha}\xi) d\xi = \frac{16}{3}\kappa_{\alpha}^3. \quad (24)$$

I also find

$$h_1 = \int_{-\infty}^{\infty} 2u_{\alpha}u_1 d\xi \quad (25)$$

which is non-singular and yields

$$\begin{aligned} h_1 = & -16 \int_{-\infty}^{\infty} \frac{dk}{2\pi} r(k) \exp(ikq_{\alpha}) \int_{-\infty}^{\infty} d\xi \frac{\exp(-2ik\xi)}{(k+i\kappa_{\alpha})^2} \\ & [ik(k^2 - \kappa_{\alpha}^2)\kappa_{\alpha}^2 \text{sech}^2(\kappa_{\alpha}\xi) + 2k^2\kappa_{\alpha}^3 \text{sech}^2(\kappa_{\alpha}\xi) \tanh(\kappa_{\alpha}\xi) \\ & + 2ik\kappa_{\alpha}^4 \text{sech}^4(\kappa_{\alpha}\xi) + \kappa_{\alpha}^5 \text{sech}^4(\kappa_{\alpha}\xi) \tanh(\kappa_{\alpha}\xi)] = 0. \end{aligned} \quad (26)$$

At next order,

$$h_2 = \int_{-\infty}^{\infty} (u_1^2 + 2u_{\alpha}u_2) d\xi = h_2^{(s)} + h_2^{(n)} \quad (27)$$

has both a singular part $h_2^{(s)}$ and a non-singular part $h_2^{(n)}$. The non-singular part can be shown to equal zero, and I concentrate on the singular part,

$$\begin{aligned} h_2^{(s)} = & \lim_{\xi \rightarrow \infty} 16 \int_{-\infty}^{\infty} \frac{dk_1}{2\pi} r(k_1) \int_{-\infty}^{\infty} \frac{dk_2}{2\pi} r(k_2) \exp[i(k_1 + k_2)q_{\alpha}] \\ & \int_{-\infty}^{\xi} d\xi_1 \exp[-2i(k_1 + k_2)\xi_1] \frac{4k_1^2k_2^2 - k_1k_2(k_1^2 - \kappa_{\alpha}^2)(k_2^2 - \kappa_{\alpha}^2)}{(k_1 + i\kappa_{\alpha})^2(k_2 + i\kappa_{\alpha})^2}. \end{aligned} \quad (28)$$

Since the limit operator is outside the integral over k_1 the exchange of the integrals over k_1 and ξ_1 is legitimate. To evaluate Eq. (28), I first explicitly carry out the integral over ξ_1 assuming both k_1 and k_2 are in the upper half Bargmann strip. We then lower the contour over k_1 , avoiding the pole as shown in Fig. 3. The continuous part of the integral vanishes, leaving only the pole contribution. I thus find

$$h_2^{(s)} = 8 \int_{-\infty}^{\infty} \frac{dk_2}{2\pi} |r(k_2)|^2 k_2^2, \quad (29)$$

where $|r(k)|$ indicates the usual absolute value on the real k -axis and its analytic extension elsewhere. I conclude

$$H_1 = \frac{16}{3}\kappa_{\alpha}^3 + 8 \int_{-\infty}^{\infty} \frac{dk}{2\pi} k^2 |r(k)|^2, \quad (30)$$

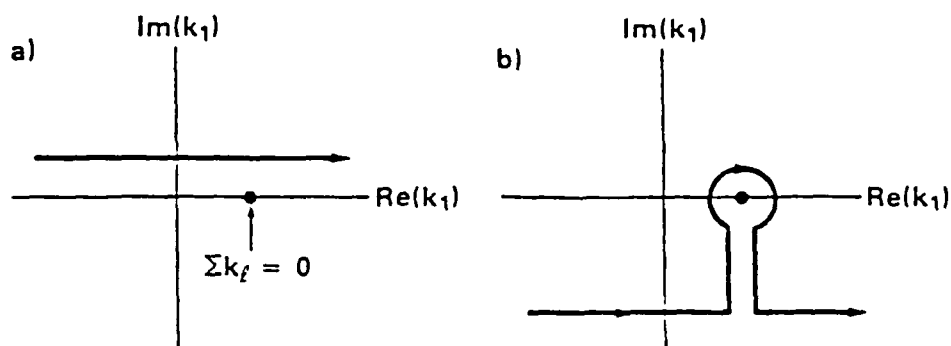


FIGURE 3. Illustration of the integration path as $Im(k_1)$ is decreased. The dot represents a pole. As $Im(k_1)$ is decreased, the pole yields a δ -function contribution.

which agrees through the order to which we are working with the exact result of Zakharov and Fadeev^[19].

$$H_1 = \frac{16}{3} \kappa_\alpha^3 - 8 \int_{-\infty}^{\infty} \frac{dk}{2\pi} k^2 \ln[1 - |r(k)|^2]. \quad (31)$$

Since H_1 only depends on the momenta, a soliton in the unperturbed equation remains a soliton, although its velocity of propagation changes.

When $p = 3$ or $p = 4$, the equations are still integrable, but the initial conditions for a soliton are changed, and, as a consequence, the h_n will depend on the coordinates just as in the non-integrable case where $p = 5$. Explicitly, I first obtain the result

$$h_0 = \alpha^{(p)} \kappa_\alpha^{2p-1}, \quad (32)$$

where

$$\alpha^{(3)} = -\frac{128}{15}, \quad \alpha^{(4)} = \frac{512}{35}, \quad \alpha^{(5)} = -\frac{8192}{315}. \quad (33)$$

Next, I find

$$h_1 = \beta^{(p)} \pi i \int_{-\infty}^{\infty} \frac{dk}{2\pi} r(k) \exp(ikq_\alpha) k^2 (k - i\kappa_\alpha)^2 P^{(p)}(k, \kappa_\alpha) \operatorname{csch}(k\pi/\kappa_\alpha), \quad (34)$$

where

$$\beta^{(3)} = \frac{64}{3}, \quad \beta^{(4)} = -\frac{256}{15}, \quad \beta^{(5)} = \frac{512}{105}, \quad (35)$$

and

$$P^{(3)} = 1, \quad P^{(4)} = (k^2 + 4\kappa_\alpha^2), \quad P^{(5)} = (k^2 + 4\kappa_\alpha^2)(k^2 + 9\kappa_\alpha^2). \quad (36)$$

Finally, I obtain

$$h_2 = \gamma^{(p)} \pi \int_{-\infty}^{\infty} \frac{dk_1}{2\pi} r(k_1) \int_{-\infty}^{\infty} \frac{dk_2}{2\pi} r(k_2) \frac{\exp[i(k_1 + k_2)q_\alpha]}{(k_1 + i\kappa_\alpha)^2 (k_2 + i\kappa_\alpha)^2} Q^{(p)}(k_1, k_2, \kappa_\alpha) R^{(p)}(k_1, k_2, \kappa_\alpha) (k_1 + k_2) \operatorname{csch}[\pi(k_1 + k_2)/\kappa_\alpha], \quad (37)$$

where

$$\gamma^{(3)} = -\frac{64}{3}, \quad \gamma^{(4)} = \frac{256}{15}, \quad \gamma^{(5)} = -\frac{512}{315}, \quad (38)$$

and

$$Q^{(3)} = 1, \quad Q^{(4)} = (k_1 + k_2)^2 + \kappa_\alpha^2, \quad Q^{(5)} = [(k_1 + k_2)^2 + \kappa_\alpha^2][(k_1 + k_2)^2 + 4\kappa_\alpha^2]. \quad (39)$$

The quantity $R^{(p)}$ is a polynomial of the form

$$R^{(p)} = a_1^{(p)} \kappa_\alpha^6 + a_2^{(p)} (k_1 + k_2)^2 \kappa_\alpha^4 + a_3^{(p)} k_1 k_2 \kappa_\alpha^4 + a_4^{(p)} (k_1 + k_2)^4 \kappa_\alpha^2 + a_5^{(p)} k_1 k_2 (k_1 + k_2)^2 \kappa_\alpha^2 + a_6^{(p)} k_1^2 k_2^2 \kappa_\alpha^2 + a_7^{(p)} (k_1 + k_2)^6 + a_8^{(p)} k_1 k_2 (k_1 + k_2)^4 + a_9^{(p)} k_1^2 k_2^2 (k_1 + k_2)^2 + a_{10}^{(p)} k_1^3 k_2^3, \quad (40)$$

where

$a_1^{(3)} = 1$	$a_2^{(3)} = 3$	$a_3^{(3)} = -7$	$a_4^{(3)} = 3$	$a_5^{(3)} = -14$
$a_1^{(4)} = 4$	$a_2^{(4)} = 9$	$a_3^{(4)} = -26$	$a_4^{(4)} = 6$	$a_5^{(4)} = -35$
$a_1^{(5)} = 27$	$a_2^{(5)} = 57$	$a_3^{(5)} = -192$	$a_4^{(5)} = 33$	$a_5^{(5)} = -216$
$a_6^{(3)} = 17$	$a_7^{(3)} = 1$	$a_8^{(3)} = -7$	$a_9^{(3)} = 17$	$a_{10}^{(3)} = -9$
$a_6^{(4)} = 52$	$a_7^{(4)} = 1$	$a_8^{(4)} = -9$	$a_9^{(4)} = 28$	$a_{10}^{(4)} = -30$
$a_6^{(5)} = 375$	$a_7^{(5)} = 3$	$a_8^{(5)} = -32$	$a_9^{(5)} = 135$	$a_{10}^{(5)} = -210$

There are no singular contributions in any of these cases.

Structurally, these results are simpler than they perhaps appear at first glance. For all possible perturbations of the sort we are interested in, polynomial in u , its derivatives, and its integrals, one finds that h_n has the general form^[11]

$$h_n = \int_{-\infty}^{\infty} \frac{dk_1}{2\pi} |r(k_1)| \cdots \int_{-\infty}^{\infty} \frac{dk_n}{2\pi} |r(k_n)| \exp[-iq(k_1) \cdots -iq(k_2) + i(k_1 + \cdots + k_n)q_\alpha] h_n(k_1, \dots, k_n; \kappa_\alpha). \quad (41)$$

Hence, the dependence on the canonical coordinates (as opposed to the canonical momenta) is entirely isolated inside the argument of imaginary exponentials. The quantity h_n depends only on κ_α and thus p_α . In general, it consists of a number of terms, each of which is a rational function of its arguments and may be multiplied by a δ -function factor of the form

$$\delta\left(\sum_j k_j\right)$$

due to a singular contribution containing two or more of the k_j . Those δ -functions which contain only two elements must be resolved explicitly since they have the effect of eliminating part of the coordinate dependence. Those δ -function factors containing three or more elements need not be resolved explicitly, although they can have an important effect on the behavior of the resonant denominators as I will describe shortly.

IV. LOWEST ORDER LIE GENERATOR AND RESONANT DENOMINATORS

The goal of Hamiltonian perturbation theory is to make a series of canonical transformations which eliminate the dependence of the Hamiltonian on the canonical coordinates through any given order. Through that order, the transformed action-angle variables evolve linearly in time,

$$\begin{aligned} J^{(n)} &= J_0^{(n)}, \\ \theta^{(n)} &= \theta_0^{(n)} + \Omega^{(n)}t, \end{aligned} \tag{42}$$

just as the original variables did in the unperturbed problem. Here the superscript n indicates the order of the transformation. The effect of these transformations is shown schematically in Fig. 4. Before the action-angle transformation, the original coordinates u evolve in a complicated way, shown as the dashed line to the left. However, after the action-angle transformation, the perturbed system *still* evolves in a complicated way. To obtain variables which evolve linearly through order n , we make further transformations using Hamiltonian perturbation theory. The evolution of these variables is shown schematically as the right-hand branch of Fig. 4. The three-sided path including this branch has through order n the same property that the original path of Fig. 1 has for the unperturbed system; it allows us to "win" over straightforward time integration when the time interval becomes sufficiently long.

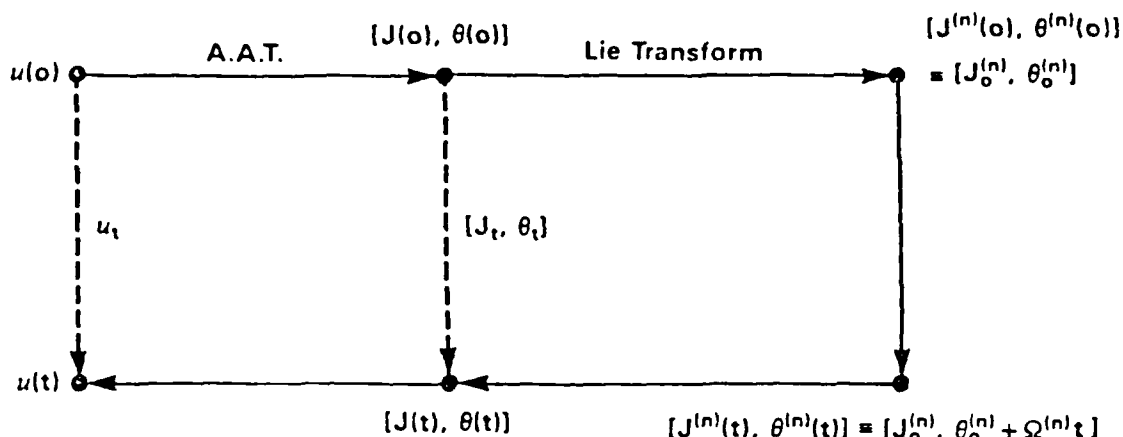


FIGURE 4. Schematic illustration of the way in which Hamiltonian perturbation theory can be used to solve perturbed equations. This figure should be compared with Fig. 1.

There is, however, a price to pay. These transformations generally diverge as one continues them to arbitrarily high order due to resonant or small denominators. The small denominators which appear in the theory equal zero in some cases on the real k -axis. To avoid this difficulty, I must extend integrals over k into the upper half Bargmann strip. To ensure that this extension is possible, we demand that $|u(x)| \rightarrow 0$ as $x \rightarrow \pm\infty$ faster than some exponential which implies that $r(k)$ is analytic in some strip surrounding the real k -axis. To ensure that all our integrals exist we impose the complementary constraint that $u(x)$ be analytic inside some strip around the real x -axis. Hence, $|r(k)|$ decreases faster than some exponential as $k \rightarrow \pm\infty$. If these conditions hold at any point in time, they hold for all times.

One of the beauties of the Hamiltonian approach is that it allows one to eliminate secularities in a simple, natural way. At any order n , we divide the Hamiltonian $H^{(n)}$ into two pieces

$$H^{(n)} = \hat{H}^{(n)} + \tilde{H}^{(n)}, \quad (43)$$

where the former is coordinate-independent and the latter is coordinate-dependent. We then eliminate $\tilde{H}^{(n)}$ to obtain $\tilde{H}^{(n+1)}$ which has a renormalized frequency $\Omega^{(n+1)}$ to which $\hat{H}^{(n)}$ contributes. At lowest order, for the examples which we are considering, this division is trivial. When $p = 2$, I find $\hat{H}_1 = H_1$ and $\tilde{H}_1 = 0$. As $\tilde{H}_1 = 0$, there is nothing to eliminate and no need to transform the Hamiltonian. When $p = 3, 4$, or 5 , I find $\hat{H}_1 = h_0$ and $\tilde{H}_1 = h_1 + h_2$ through second order in powers of

$r(k)$.

The Lie approach to Hamiltonian perturbation theory which I am using is based on two theorems^[13,14]. I let $F[u]$, $G[u]$, and $H[u]$ be arbitrary functionals of u . I also define the Lie operator $:F:$ corresponding to $F[u]$ as^[17]

$$:F:G \equiv [F, G], \quad (44)$$

where $[F, G]$ indicates the Poisson bracket of F and G . The two theorems are:

1) *The transformation*

$$\bar{u} = \exp(:F:)u = \sum_{i=0}^{\infty} \frac{1}{i!} (:F:)^i u \quad (45)$$

is symplectic

2) *The relation*

$$\exp(-:F:)H[\exp(:F:)u] = H(u) \quad (46)$$

holds.

At lowest order, our task is to find a functional F_1 such that $H^{(1)} = \exp(-:F:)H$ no longer includes $\tilde{H}_1$. Then, from the second theorem, it follows that

$$H^{(1)}[p^{(1)}(k), q^{(1)}(k), p_{\alpha}^{(1)}, q_{\alpha}^{(1)}] = H[p(k), q(k), p_{\alpha}, q_{\alpha}], \quad (47)$$

while from the first theorem

$$\begin{aligned} p^{(1)}(k) &= \exp(\epsilon :F_1:)p(k), & q^{(1)}(k) &= \exp(\epsilon :F_1:)q(k), \\ p_{\alpha}^{(1)} &= \exp(\epsilon :F_1:)p_{\alpha}, & q_{\alpha}^{(1)} &= \exp(\epsilon :F_1:)q_{\alpha}, \end{aligned} \quad (48)$$

is a symplectic transformation and is just the transformation we want! The procedure is then continued to arbitrarily high order. Explicitly, we find through second order in ϵ

$$\begin{aligned} \exp(-\epsilon :F_1:)H &= H_0 + \epsilon \hat{H}_1 + \epsilon \tilde{H}_1 - \epsilon [F_1, H_0] \\ &\quad - \epsilon^2 [F_1, \hat{H}_1] - \epsilon^2 [F_1, \tilde{H}_1] + \frac{1}{2} \epsilon^2 [F_1, [F_1, H_0]]. \end{aligned} \quad (49)$$

To eliminate $\tilde{H}_1$, we must set

$$\tilde{H}_1 = [F_1, H_0] = \left. \frac{dF_1}{dt} \right|_0. \quad (50)$$

In other words F_1 may be determined from $\tilde{H}_1$ by integration of the unperturbed orbits. Explicitly, I find

$$\int dt_0 \exp\left\{-i \sum_j [q(k_j) - k_j q_\alpha]\right\} = \frac{\exp\{-i \sum_j [q(k_j) - k_j q_\alpha]\}}{(-8i) \sum_j (k_j^3 + k_j \kappa_\alpha^2)}. \quad (51)$$

The zeroes in the small denominators

$$D = \sum_j (k_j^3 + k_j \kappa_\alpha^2), \quad (52)$$

may be avoided by integrating around them in the complex k -plane as needed.

For the cases of interest here, I find, expanding F_1 in powers of $r(k)$, that $F_1 = f_1 + f_2$, where f_1 and f_2 may be written

$$f_1 = -\frac{\beta^{(p)}\pi}{8} \int_{-\infty}^{\infty} \frac{dk}{2\pi} r(k) \exp(ikq_\alpha) k \frac{k - i\kappa_\alpha}{k + i\kappa_\alpha} P^{(p)}(k, \kappa_\alpha) \operatorname{csch}(k\pi/\kappa_\alpha), \quad (53)$$

and

$$f_2 = \frac{\gamma^{(p)}\pi i}{8} \int_{-\infty}^{\infty} \frac{dk_1}{2\pi} r(k_1) \int_{-\infty}^{\infty} \frac{dk_2}{2\pi} r(k_2) \frac{\exp[i(k_1 + k_2)q_\alpha]}{(k_1 + i\kappa_\alpha)^2 (k_2 + i\kappa_\alpha)^2 [(k_1 + k_2)^2 - 3k_1 k_2 + \kappa_\alpha^2]} Q^{(p)}(k_1, k_2, \kappa_\alpha) R^{(p)}(k_1, k_2, \kappa_\alpha) \operatorname{csch}[\pi(k_1 + k_2)/\kappa_\alpha]. \quad (54)$$

It is not difficult to show that the k -integrals in the Eqs. (53) and (54) are well-defined when both k_1 and k_2 are in the upper half Bargmann strip. More generally, if we consider the solution to the expression $D = 0$, I find that as long as at least two of the k_j are not tied together by a single δ -function factor and I assume that all the k_j except one which I designate k_l , are arbitrarily close to being purely real, then $D = 0$ is possible only if $\operatorname{Im}(k_l) = 0$ or $\operatorname{Im}(k_l) > \kappa_\alpha$. By choosing $\operatorname{Im}(k_l) = \kappa_\alpha/2$, it is possible to bound D away from zero. If all the k_j are tied together by a δ -function factor then it is no longer possible to bound D away from zero, although I can always avoid having it equal zero by an appropriate choice of the k -integration contour. As a consequence of these latter terms, the perturbation theory is expected to ultimately diverge, although it is finite order-by-order. Physically, these terms correspond to a radiation component which travels with the solitary wave and only

slowly disappears as $t \rightarrow +\infty$. More details on the resonant denominators may be found in reference 11.

V. CALCULATION OF THE HIGHER ORDER HAMILTONIAN

Having determined F_1 , I may now calculate $H^{(1)}$, the transformed Hamiltonian. From Eqs. (49) and (50), it follows that

$$H^{(1)} = \exp(-\epsilon; F_1;)H = H_0 + \epsilon \hat{H}_1 - \epsilon^2 [F_1, \hat{H}_1] - \frac{\epsilon^2}{2} [F_1, \tilde{H}_1]. \quad (55)$$

Noting that

$$\frac{\partial |r(k)|}{\partial p(k)} = \frac{\pi}{4k} \frac{1 - |r(k)|^2}{|r(k)|}, \quad (56)$$

I find that in order to keep terms through order ϵ^2 in the equations of motion, where I assume that $|r(k)|$ is of order ϵ , I must keep terms in the perturbed Hamiltonian of order $\epsilon^2 |r(k)|$ and $\epsilon |r(k)|^2$ as well as terms of order ϵ^2 and $\epsilon |r(k)|$.

The calculation of $H^{(1)}$ for the examples which I am considering here is straightforward, albeit somewhat lengthy. I will concentrate here on calculating in detail the term which contributes to $\hat{H}^{(1)}$ at order ϵ^2 in the case $p = 3$ and simply record the full results. The term in Eq. (55) on which we concentrate is $[f_1, h_1]$ which is part of $[F_1, H_1]$. Only the continuous portion of this Poisson bracket contributes since the soliton portion yields a term of order $\epsilon^2 [r(k)]^2$. We first find that when $p = 3$

$$\begin{aligned} \frac{\partial f_1}{\partial p(k)} = & -\frac{\pi}{3} \frac{1 - |r(k)|^2}{|r(k)|} \exp[-iq(k) + ikq_\alpha] \frac{k - i\kappa_\alpha}{k + i\kappa_\alpha} \text{csch}(\pi k/\kappa_\alpha) \\ & - \frac{\pi}{3} \frac{1 - |r(k)|^2}{|r(k)|} \exp[iq(k) - ikq_\alpha] \frac{k + i\kappa_\alpha}{k - i\kappa_\alpha} \text{csch}(\pi k/\kappa_\alpha), \end{aligned} \quad (57)$$

and

$$\begin{aligned} \frac{\partial f_1}{\partial q(k)} = & \frac{4i}{3} |r(k)| \exp[-iq(k) + ikq_\alpha] k \frac{k - i\kappa_\alpha}{k + i\kappa_\alpha} \text{csch}(\pi k/\kappa_\alpha) \\ & - \frac{4i}{3} |r(k)| \exp[iq(k) - ikq_\alpha] k \frac{k + i\kappa_\alpha}{k - i\kappa_\alpha} \text{csch}(\pi k/\kappa_\alpha), \end{aligned} \quad (58)$$

where k is real and I have used the relations $|r(-k)| = |r(k)|$, $q(-k) = -q(k)$. Similar results can be obtained for $\partial h_1/\partial p(k)$ and $\partial h_1/\partial q(k)$. The operators $\partial/\partial q(k)$ and

$\partial/\partial q(k)$ are anti-symmetric in k ; hence, the k -integrals can always be extended from the half interval $[0, \infty)$ to the full interval $(-\infty, \infty)$ and from there into the upper half Bargmann strip. In the case considered here I find through the order to which I am working,

$$\begin{aligned} [f_1, h_1] &= \int_0^\infty dk \left(\frac{\partial f_1}{\partial q(k)} \frac{\partial h_1}{\partial p(k)} - \frac{\partial f_1}{\partial p(k)} \frac{\partial h_1}{\partial q(k)} \right) \\ &= -\frac{128\pi^2}{9} \int_0^\infty \frac{dk}{2\pi} [1 - |r(k)|^2] k^2 (k^2 + \kappa_\alpha^2) \operatorname{csch}^2(\pi k/\kappa_\alpha) \\ &= -\frac{128\pi^2}{9} \int_{-\infty}^\infty \frac{dk}{2\pi} k^2 (k^2 + \kappa_\alpha^2) \operatorname{csch}^2(\pi k/\kappa_\alpha) \\ &= -\frac{128}{45} \kappa_\alpha^5. \end{aligned} \quad (59)$$

In general, I may write

$$H^{(1)} = H_0 + \epsilon \hat{H}_1 + \epsilon^2 \hat{H}_2 + \epsilon^2 \tilde{H}_2. \quad (60)$$

I now find

$$\hat{H}_2 = \bar{\alpha}^{(p)} \kappa_\alpha^{4p-7},$$

where

$$\bar{\alpha}^{(3)} = -\frac{128}{45}, \quad \bar{\alpha}^{(4)} = \frac{53,248}{1575}, \quad \bar{\alpha}^{(5)} = \frac{128,712,704}{525,525}, \quad (61)$$

and

$$\begin{aligned} \tilde{H}_2 &= \bar{\beta}^{(p)} \pi i \int_{-\infty}^\infty \frac{dk}{2\pi} r(k) \exp(ikq_\alpha) k^2 \kappa_\alpha^{p-2} \frac{k - i\kappa_\alpha}{k + i\kappa_\alpha} P^{(p)}(k, \kappa_\alpha) \operatorname{csch}(\pi k/\kappa_\alpha) \\ &\quad + \bar{\gamma}^{(p)} \pi^2 i \int_{-\infty}^\infty \frac{dk_1}{2\pi} \int_{-\infty}^\infty \frac{dk_2}{2\pi} r(k_2) \exp(ik_2q_\alpha) \frac{\bar{Q}^{(p)}(k_1, k_2, \kappa_\alpha)}{(k_2 + i\kappa_\alpha)^2} \\ &\quad R^{(p)}(k_1, k_2, \kappa_\alpha) \left[\frac{k_1}{(k_1 + k_2)^2 - 3k_1k_2 + \kappa_\alpha^2} + \frac{(k_1 + k_2)}{(k_1^2 + \kappa_\alpha^2)} \right] \\ &\quad \operatorname{csch}(\pi k_1/\kappa_\alpha) \operatorname{csch}[\pi(k_1 + k_2)/\kappa_\alpha]. \end{aligned} \quad (62)$$

The quantities $P^{(p)}$ and $R^{(p)}$ are defined in Eqs. (36) and (40) respectively. The factors $\bar{\beta}^{(p)}$ and $\bar{\gamma}^{(p)}$ equal

$$\begin{aligned} \bar{\beta}^{(3)} &= -\frac{128}{9}, & \bar{\beta}^{(4)} &= -\frac{2048}{75}, & \bar{\beta}^{(5)} &= -\frac{65,536}{3675}, \\ \bar{\gamma}^{(3)} &= -\frac{128}{9}, & \bar{\gamma}^{(4)} &= -\frac{2048}{225}, & \bar{\gamma}^{(5)} &= -\frac{8192}{33,075}, \end{aligned} \quad (63)$$

and the $\bar{Q}^{(p)}$ equal

$$\begin{aligned}\bar{Q}^{(3)} &= 1, \\ \bar{Q}^{(4)} &= [(k_1 + k_2)^2 + \kappa_\alpha^2](k_1^2 + 4\kappa_\alpha^2), \\ \bar{Q}^{(5)} &= [(k_1 + k_2)^2 + \kappa_\alpha^2][(k_1 + k_2)^2 + 4\kappa_\alpha^2](k_1^2 + 4\kappa_\alpha^2)(k_1^2 + 9\kappa_\alpha^2).\end{aligned}\quad (64)$$

Given the explicit form for $H^{(1)}$, we may now go on to calculate F_2 and $H^{(2)}$. We find explicitly

$$\begin{aligned}F_2 &= -\frac{\bar{\beta}^{(p)}\pi}{8} \int_{-\infty}^{\infty} \frac{dk}{2\pi} r(k) \exp(ikq_\alpha) k \kappa_\alpha^{2p-2} \frac{1}{(k + i\kappa_\alpha)^2} P^{(p)}(k, \kappa_\alpha) \operatorname{csch}(\pi k/ka) \\ &\quad - \frac{\bar{\gamma}^{(p)}\pi}{8} \int_{-\infty}^{\infty} \frac{dk_1}{2\pi} \int_{-\infty}^{\infty} \frac{dk_2}{2\pi} r(k_2) \exp(ik_2q_\alpha) \frac{\bar{Q}^{(p)}(k_1, k_2, \kappa_\alpha)}{k_2(k_1^2 + \kappa_\alpha^2)(k_2 + i\kappa_\alpha)^2} \\ &\quad R^{(p)}(k_1, k_2, \kappa_\alpha) \left[\frac{k_1}{(k_1 + k_2)^2 - 3k_1k_2 + \kappa_\alpha^2} + \frac{(k_1 + k_2)}{(k_1^2 + \kappa_\alpha^2)} \right],\end{aligned}\quad (65)$$

and

$$H^{(2)} = H_0 + \epsilon \hat{H}_1 + \epsilon^2 \hat{H}_2. \quad (66)$$

Through the order to which we are working the Hamiltonian depends only on the canonical momenta. One can directly verify that all the k -integrals in $\hat{H}_2$ and F_2 are well-defined when carried out in the upper half Bargmann strip.

VI. CALCULATION OF THE FIRST ORDER POTENTIAL

Having determined F_1 and F_2 , we can in principle, calculate the second order potential through the formula

$$u^{(2)} = \exp(-\epsilon^2 : F_2 :) \exp(-\epsilon : F_1 :) u; \quad (67)$$

however, I restrict myself now to calculating $u^{(1)} = \exp(-\epsilon : F_1 :) u$ in order to keep the algebra within reasonable bounds. Having determined $u^{(1)}$, I can check the Hamiltonian approach by finding the first order solitary wave solution and comparing the results to what is obtained using simpler methods which do not apply to arbitrary initial conditions. Such a check has already been carried out for other

simple examples^[10]. Calculating $u^{(1)}$ when $p = 3$ and setting $r^{(1)}(k) = 0$, I obtain for the solitary wave solution

$$u_s = -2\kappa_\alpha^2 \operatorname{sech}^2(\kappa_\alpha \xi) - \frac{2}{3}\epsilon\kappa_\alpha^2 \left[\operatorname{sech}^2(\kappa_\alpha \xi) - 2(1 + 2\kappa_\alpha \xi) \operatorname{sech}^2(\kappa_\alpha \xi) \tanh(\kappa_\alpha \xi) \right], \quad (68)$$

where I have left out the superscript 1 which should be on κ_α and ξ . I note that the result is non-uniform in ξ although it does fall off faster than some exponential as $|\xi| \rightarrow \pm\infty$. From the Hamiltonian, I find

$$\kappa_\alpha \dot{\xi} = \kappa_\alpha \dot{q}_\alpha / 2 = -4\kappa_\alpha^3 - \frac{16}{3}\epsilon\kappa_\alpha^2, \quad (69)$$

Combining Eqs. (68) and (69), I find that through the order to which I am working

$$\begin{aligned} u_s &= -2\kappa_\alpha^2 \left(1 + \frac{1}{3}\epsilon\right) \operatorname{sech}^2 \left[\kappa_\alpha \left(1 + \frac{2}{3}\epsilon\right) \xi + \frac{\epsilon}{3} \right] \\ &= -2\kappa_\alpha^2 \left(1 + \frac{1}{3}\epsilon\right) \operatorname{sech}^2 \left[\kappa_\alpha \left(1 + \frac{2}{3}\epsilon\right) \xi_0 - 4\kappa_\alpha^3 \left(1 + \frac{2\epsilon}{3}\right) \left(1 + \frac{4\epsilon}{3}\right) t + \frac{\epsilon}{3} \right] \end{aligned} \quad (70)$$

where ξ_0 is a constant of integration. Letting $\tilde{\kappa} = \kappa_\alpha(1 + 2\epsilon/3)$, we conclude

$$u_s = -2\tilde{\kappa}^2 (1 - \epsilon) \operatorname{sech}^2 \left[\tilde{\kappa} \xi_0 - 4\tilde{\kappa}^3 t + \frac{\epsilon}{3} \right], \quad (71)$$

which is the same as the exact solution

$$u_s = -\frac{2\tilde{\kappa}^2}{1 + \epsilon} \operatorname{sech}^2 \left[\tilde{\kappa} \xi_0 - 4\tilde{\kappa}^3 t + \frac{\epsilon}{3} \right], \quad (72)$$

through the order to which we are working.

I have obtained similar results for the cases $p = 4$ and $p = 5$. In the former case I compared the results of my theory to what is obtained from the exact solution. In the latter case, no exact solution exists, and I compared the results with what is obtained from the expansion procedure of Kodama and Taniuti^[20]. I have also explicitly verified that all the k -integrals which appear in $u^{(2)}$ are finite, although I have not carried them out in detail.

VII. CONCLUSIONS AND ACKNOWLEDGMENTS

In past work, I have used Hamiltonian perturbation methods to show that solitary waves emerge "to all orders" in a small parameter from arbitrary initial

data. In this work, I apply the results to a second order calculation of some simple examples, $H_1 = u^p$. I have shown explicitly how to eliminate secularities by splitting the Hamiltonian into its coordinate-dependent and coordinate-independent pieces. I have also calculated the first order potentials and, from that, extracted the solitary wave structure. The results agree with previous theoretical work.

This work was supported by the Naval Research Laboratory and Science Applications International Corporation.

REFERENCES

- [1] D.J. Korteweg and G. DeVries, "On the change of form of long waves advancing in a rectangular canal, and on a new type of long stationary waves," *Philos. Mag. Ser. 5* **39**, 422-443 (1895).
- [2] H. Washimi and T. Taniuti, "Propagation of ion acoustic solitary waves of small amplitude," *Phys. Rev. Lett.* **17**, 996-998 (1966).
- [3] C.R. Menyuk and H.H. Chen, "Hamiltonian Structure of the higher-order corrections to the Korteweg-de Vries equation," *Phys. Rev. Lett.* **55**, 1809-1811 (1985).
- [4] C.R. Menyuk and H.H. Chen, "On the Hamiltonian structure of ion-acoustic plasma waves and water waves in channels," *Phys. Fluids* **29**, 998-1003 (1986).
- [5] J.L. Hammack and H. Segur, "The Korteweg-de Vries equation and water waves, Part 2. Comparison with experiments," *J. Fluid Mech.* **65**, 289-314 (1974).
- [6] P.D. Weidman and T. Maxworthy, "Experiments on strong interaction between solitary waves," *J. Fluid Mech.* **85**, 417-431 (1978).
- [7] H. Ikezi, "Experiments on ion acoustic solitary waves," *Phys. Fluids* **16**, 1668-1675 (1973).
- [8] S. Watanabe, "Ion-acoustic solitons excited by a single grid," **14**, 353-364 (1975).
- [9] C.R. Menyuk, "Lie perturbation theory for an infinite-dimensional Hamiltonian system," in *Proceedings of the 19th International Colloquium on Group Theoretical Methods in Physics*, edited by W.W. Zachary (World Scientific Publ., Singapore, 1984), pp. 54-57.
- [10] C.R. Menyuk, "Origin of solitons in the 'real' world," *Phys. Rev. A* **33**, 4367-4374 (1986).
- [11] C.R. Menyuk, "Hamiltonian perturbations of the Korteweg-de Vries equation," (to appear).

- [12] C.R. Menyuk, "Finite time solution of the perturbed Korteweg-de Vries equation," (to appear).
- [13] G.-I. Hori, "Theory of general perturbations with unspecified canonical variables," Publ. Astron. Soc. Japan **18**, 287-296 (1966).
- [14] A. Deprit, "Canonical transformations depending on a small parameter," Cel. Mech. **1**, 12-30 (1969).
- [15] V.I. Karpman and E.M. Maslov, "Perturbation theory for solitons," Zh.E.T.F. **73**, 537-559 (1977) [Sov. Phys. JETP **46**, 281-291 (1977)].
- [16] D.J. Kaup and A.C. Newell, "Solitons as particles, oscillators, and in slowly changing media: A singular perturbation theory," Proc. Roy. Soc. London A, **361**, 413-446 (1978).
- [17] A.J. Dragt, "Lectures on nonlinear orbit dynamics," in *Physics of High Energy Particle Accelerators*, edited by R.A. Carrigan, F.R. Huson, and M. Month (A.I.P. Conf. Proc. no. 87, New York, 1982), pp. 147-313.
- [18] M. Kruskal, "Asymptotic theory of Hamiltonian and other systems with all solutions nearly periodic," J. Math. Phys. **3**, 806-828 (1962).
- [19] V.E. Zakharov and L.D. Fadeev, "Korteweg-de Vries equation: A completely integrable Hamiltonian system," Funct. Anal. Pril. [Funct. Anal. Appl. **5**, 280 (1971)].
- [20] Y. Kodama and T. Taniuti, "Higher order approximation in the reductive perturbation method, I, the weakly dispersive system," J. Phys. Soc. Japan **45**, 298-310 (1978).

"Solitons in a birefringent Kerr medium," (C.R. Menyuk), in *Solitons and Chaos in Optical Systems*, H.C. Morris and D. Heffernan, eds. (World Scientific Publ., Singapore), to appear.

Solitons in a Birefringent Kerr Medium

C.R. Menyuk*
Department of Electrical Engineering
University of Maryland
Baltimore, MD 21228
USA

ABSTRACT

Equations are derived which govern wave propagation in a birefringent Kerr medium. Painlevé analysis indicates that these equations are integrable when the two polarizations are uncoupled or when the Kerr coefficient for each polarization depends on the total intensity. In the latter case, the equation's integrability was first proved by Manakov who found single soliton solutions. Here, the single soliton solutions that he found are extended.

I. INTRODUCTION

Many optical media are birefringent, and, as a consequence of this birefringence, have two normal modes with preferred axes of propagation. If the modes are linearly polarized, then we may designate the axes $\hat{e}_x$ and $\hat{e}_y$ which correspond to two orthogonal, real directions; however, if the modes are circularly polarized, then we designate the axes $\hat{e}_r = (\hat{e}_x - i\hat{e}_y)/\sqrt{2}$ and $\hat{e}_l = (\hat{e}_x + i\hat{e}_y)/\sqrt{2}$ which are no longer real.

If the nonlinear dielectric medium can be considered isotropic, then the lowest order nonlinear interaction which will appear is the cubic or Kerr nonlinearity [1-4]. In an intermediate range of birefringence, to be defined more precisely later in this paper, we then find

$$\begin{aligned}iu_{\xi} + \frac{1}{2}u_{ss} + (|u|^2 + B|v|^2)u &= 0, \\iv_{\xi} + \frac{1}{2}v_{ss} + (B|u|^2 + |v|^2)v &= 0,\end{aligned}\tag{1}$$

where u and v represent the amplitude envelopes of two normal modes, ξ and s are normalized distance along the medium, and B is a parameter whose value depends on the details of the nonlinear dielectric response, although it is always $O(1)$. The most important single case is when the nonlinear dielectric response can be considered instantaneous, as is the case in optical fibers [5]. One then finds $B = 2/3$.

We note that while the coefficient of birefringence does not appear explicitly in Eq. (1), the transformation

$$\begin{aligned}u' &= u \cos \theta + v \sin \theta, \\v' &= v \cos \theta - u \sin \theta,\end{aligned}\tag{2}$$

does not leave Eq. (1) invariant unless $B = 1$. This invariance is a fundamental symmetry requirement if the normal modes are linearly polarized. Hence, the birefringence serves to break the azimuthal symmetry in this case.

Recently, Eq. (1) has been subjected to intensive study due to the interest in optical fiber applications [6-9]. Unfortunately, these equations appear to be non-integrable when $B = 2/3$. Still, as Manakov showed some time ago, Eq. (1) is integrable when $B = 1$. Eq. (1) is also integrable when $B = 0$ since Eq. (1) reduces to two uncoupled nonlinear Schrödinger equations. We have carried out a Painlevé analysis which indicates that these are the only integrable cases.

The case $B = 1$, aside from its intrinsic interest, is a useful starting point for studying more general B -values. In his paper, Manakov [10] showed how to solve the initial value problem and extracted those single soliton solutions where u and v are both proportional to $\text{sech}(\alpha s)$. We find more general single soliton solutions by a direct search for stationary solutions of Eq. (1). These solutions can be obtained by using a procedure first described by Darboux [11] and based on the original work of Bertrand [12] and Liouville [13].

In Sec. II of this paper, we give a brief derivation of Eq. (1). Our goal here is not rigor, but rather to elucidate what we consider to be the most important physical points. In Sec. III, we outline the Painlevé analysis and show how to obtain single soliton solutions of Eq. (1). The conclusions are in Sec. IV.

II. THE BASIC EQUATION

Recently, there have been several derivations of the nonlinear Schrödinger equation for applications to optical fibers and other optical systems. (See, e.g., [3, 4, 14-16]). We shall present a simple derivation which can easily be made more rigorous by following the approach of [16]. We consider one-dimensional propagation in a homogeneous medium and ignore transverse effects.

In the slowly varying envelope approximation, we may assume that the E -field has the form

$$\mathbf{E}(z, t) = \mathbf{E}^+(z, t) + \mathbf{E}^-(z, t),\tag{3}$$

where $\mathbf{E}$ is the real field, $\mathbf{E}^+$ is the contribution to $\mathbf{E}$ near the carrier frequency $\omega = \omega_0$, and $\mathbf{E}^-$ is the contribution to $\mathbf{E}$ near $\omega = -\omega_0$. The Fourier transform of $\mathbf{E}$ is zero outside a small range of frequencies surrounding $\omega = \omega_0$ and $\omega = -\omega_0$. Since $\mathbf{E}(z, t)$ is real, it immediately follows that $\tilde{\mathbf{E}}(z, \omega)$, the Fourier transform of $\mathbf{E}(z, t)$, satisfies the relation $\tilde{\mathbf{E}}(z, -\omega) = \tilde{\mathbf{E}}^*(z, \omega)$ from which we conclude $\tilde{\mathbf{E}}^-(z, -\omega) = \tilde{\mathbf{E}}^{+*}(z, \omega)$. For each normal mode of the medium, we may now write

$$\begin{aligned} \mathbf{P}_1(z, t) &= \frac{1}{2\pi} \int_{-\infty}^t \chi_1(t-t') \mathbf{E}_1(z, t') dt', \\ \mathbf{P}_2(z, t) &= \frac{1}{2\pi} \int_{-\infty}^t \chi_2(t-t') \mathbf{E}_2(z, t') dt', \end{aligned} \quad (4)$$

where $\mathbf{P}_1$ and $\mathbf{P}_2$ indicate the linear polarizabilities in each component of the wave. Writing the Fourier transform

$$\tilde{A}(\omega) = \frac{1}{2\pi} \int_{-\infty}^{\infty} A(t) \exp(i\omega t) dt, \quad (5)$$

we find

$$\tilde{\mathbf{P}}_{1,2}(z, \omega) = \tilde{\chi}_{1,2}(z, \omega) \tilde{\mathbf{E}}_{1,2}(z, \omega), \quad (6)$$

or, separating out the + and - contributions [2],

$$\begin{aligned} \tilde{\mathbf{P}}_{1,2}(z, \omega) &= \tilde{\mathbf{P}}_{1,2}^+(z, \omega) \hat{\mathbf{e}}_{1,2}(\omega) + \tilde{\mathbf{P}}_{1,2}^-(z, \omega) \hat{\mathbf{e}}_{1,2}^*(\omega), \\ \tilde{\mathbf{E}}_{1,2}(z, \omega) &= \tilde{\mathbf{E}}_{1,2}^+(z, \omega) \hat{\mathbf{e}}_{1,2}(\omega) + \tilde{\mathbf{E}}_{1,2}^-(z, \omega) \hat{\mathbf{e}}_{1,2}^*(\omega), \end{aligned} \quad (7)$$

where the unit vectors satisfy the relations

$$\hat{\mathbf{e}}_1 \cdot \hat{\mathbf{e}}_1^* = \hat{\mathbf{e}}_2 \cdot \hat{\mathbf{e}}_2^* = 1, \quad \hat{\mathbf{e}}_1 \cdot \hat{\mathbf{e}}_2^* = \hat{\mathbf{e}}_2 \cdot \hat{\mathbf{e}}_1^* = 0. \quad (8)$$

We concentrate on $\tilde{\mathbf{P}}_1^+$. Since the relation $\tilde{\mathbf{P}}_1^- = \tilde{\mathbf{P}}_1^{+*}$ holds, knowledge of $\tilde{\mathbf{P}}_1^+$ is sufficient to determine $\tilde{\mathbf{P}}_1^-$; $\tilde{\mathbf{P}}_2^+$ can then be determined by analogy with $\tilde{\mathbf{P}}_1^+$. It is useful to consider the slowly varying envelopes of the polarizability and the field,

$$\begin{aligned} \rho(z, t) &= P_1^+(z, t) \exp(-ik_0 z + i\omega_0 t), \\ U(z, t) &= E_1^+(z, t) \exp(-ik_0 z + i\omega_0 t), \end{aligned} \quad (9)$$

where we will specify k_0 shortly. From Eqs. (8) and (9), we find

$$\rho(z, t) = \int_{-\infty}^{\infty} \tilde{\chi}(\omega + \omega_0) \tilde{U}(z, \omega) \exp(-i\omega t) d\omega. \quad (10)$$

The quantity $\bar{U}(z, \omega)$ is peaked in a small region surrounding $\omega = 0$, and we assume that $\bar{\chi}$ is slowly varying throughout this region. Thus, we may write

$$\bar{\chi}_1(\omega + \omega_0) \simeq \bar{\chi}_1(\omega_0) + \bar{\chi}'_1(\omega_0)\omega + \frac{1}{2}\bar{\chi}''_1(\omega_0)\omega^2, \quad (11)$$

where $\bar{\chi}'_1(\omega_0)$ and $\bar{\chi}''_1(\omega_0)$ are the first and second derivatives of $\bar{\chi}_1(\omega + \omega_0)$ evaluated at $\omega = 0$, leading to the result

$$\rho = \bar{\chi}_1(\omega_0)u + i\bar{\chi}'_1(\omega_0)\frac{\partial u}{\partial t} - \frac{1}{2}\bar{\chi}''_1(\omega_0)\frac{\partial^2 u}{\partial t^2}. \quad (12)$$

We now recall

$$\bar{D}_1(z, \omega) = [1 + \bar{\chi}_1(\omega)]\bar{E}_1(z, \omega) = \bar{\epsilon}_1(\omega)\bar{E}_1(z, \omega), \quad (13)$$

where $\bar{\epsilon}_1(\omega)$ is the dielectric response. We further recall, from Maxwell's equations,

$$\nabla^2 \bar{E}_1 + \frac{\omega^2}{c^2} \bar{D}_1 = 0, \quad (14)$$

and we define

$$k(\omega) = \frac{\omega}{c} [\bar{\epsilon}_1(\omega)]^{1/2}, \quad (15)$$

corresponding to positive propagation. Letting $k_0 \equiv k(\omega_0)$, we now find from Eqs. (9, 13-15),

$$i\frac{\partial U}{\partial z} + ik'_0\frac{\partial U}{\partial t} - \frac{1}{2}k''_0\frac{\partial^2 U}{\partial t^2} = 0, \quad (16)$$

where k'_0 and k''_0 are the first and second derivatives of $k(\omega)$, evaluated at $\omega = \omega_0$. Similarly, we find

$$i\frac{\partial V}{\partial z} + il'_0\frac{\partial V}{\partial t} - \frac{1}{2}l''_0\frac{\partial^2 V}{\partial t^2} = 0, \quad (17)$$

where V is the envelope of E_2^+ ,

$$l(\omega) = \frac{\omega}{c} [\bar{\epsilon}_2(\omega)]^{1/2} = \frac{\omega}{c} [1 + \bar{\chi}_2(\omega)]^{1/2}, \quad (18)$$

and l_0 , l'_0 , and l''_0 are defined by analogy with k_0 , k'_0 , and k''_0 .

We now suppose that the polarizability has a cubic component and that this cubic component is isotropic. When both the anisotropy and nonlinearity are weak, the case of greatest practical interest, then anisotropy can be ignored in the nonlinear contribution at lowest order since the anisotropy is formally of higher order. The polarizability must have the form

$$\begin{aligned} P(z, t) = \frac{1}{(2\pi)^3} \int_{-\infty}^t dt_1 \int_{-\infty}^t dt_2 \int_{-\infty}^t dt_3 \chi(t - t_1, t - t_2; t - t_3) \\ [E(z, t_1) \cdot E(z, t_2)] E(z, t_3). \end{aligned} \quad (19)$$

Equation (19) is the only cubic combination of E_x and E_y which is invariant under rotations and mirror reflections. From the form of Eq. (19), it follows that the dielectric function $\chi(\tau_1, \tau_2; \tau_3)$ is invariant under the interchange $\tau_1 \leftrightarrow \tau_2$ but not under the interchanges $\tau_1 \leftrightarrow \tau_3$ or $\tau_2 \leftrightarrow \tau_3$. We thus obtain

$$\begin{aligned} \mathbf{P}^+(z, t) = \frac{1}{(2\pi)^3} \int_{-\infty}^t dt_1 \int_{-\infty}^t dt_2 \int_{-\infty}^t dt_3 \chi(t-t_1, t-t_2; t-t_3) \\ \{ [2\mathbf{E}^+(z, t_1) \cdot \mathbf{E}^-(z, t_2)] \mathbf{E}^+(z, t_3) \\ + [\mathbf{E}^+(z, t_1) \cdot \mathbf{E}^+(z, t_2)] \mathbf{E}^-(z, t_3) \}. \end{aligned} \quad (20)$$

and a similar result for $\mathbf{P}^-$. The decomposition of Eq. (20) depends on the nature of the normal modes. For linearly polarized waves,

$$\begin{aligned} P_1^+(z, t) = \frac{1}{(2\pi)^3} \int_{-\infty}^t dt_1 \int_{-\infty}^t dt_2 \int_{-\infty}^t dt_3 \chi(t-t_1, t-t_2; t-t_3) \\ \{ 2[E_1^+(z, t_1)E_1^-(z, t_2) + E_2^+(z, t_1)E_2^-(z, t_2)] E_1^+(z, t_3) \\ + [E_1^+(z, t_1)E_1^+(z, t_2) + E_2^+(z, t_1)E_2^+(z, t_2)] E_1^-(z, t_3) \}, \end{aligned} \quad (21)$$

with a similar result for P_2^+ , while for circularly polarized waves

$$\begin{aligned} P_1^+(z, t) = \frac{1}{(2\pi)^3} \int_{-\infty}^t dt_1 \int_{-\infty}^t dt_2 \int_{-\infty}^t dt_3 \chi(t-t_1, t-t_2; t-t_3) \\ \{ 2[E_1^+(z, t_1)E_1^-(z, t_1) + E_2^+(z, t_1)E_2^-(z, t_2)] E_1^+(z, t_3) \\ + 2[E_1^+(z, t_1)E_2^+(z, t_2)] E_2^-(z, t_3) \}. \end{aligned} \quad (22)$$

Making the slowly varying envelope approximation, just as in the linear case, and keeping only the lowest order terms in the expansion of $\tilde{\chi}$, we find in the case of linearly polarized waves that

$$\begin{aligned} \rho(z, t) = \alpha \{ 2(|U|^2 + |V|^2) \} U \\ + \beta \{ U^2 + V^2 \exp[-2i(k_0 - l_0)z] \} U^*, \end{aligned} \quad (23)$$

where $\alpha = \tilde{\chi}(\omega_0, -\omega_0; \omega_0)$ and $\beta = \tilde{\chi}(\omega_0, \omega_0; -\omega_0)$. For circularly polarized modes, we obtain

$$\rho(z, t) = \alpha \{ 2(|U|^2 + |V|^2) \} U + 2\beta |V|^2 U. \quad (24)$$

When the medium has an instantaneous response, $\tilde{\chi}(\omega_0, -\omega_0; \omega_0) = \tilde{\chi}(\omega_0, \omega_0; -\omega_0) = \tilde{\chi}(0, 0; 0)$, so that $\alpha = \beta$.

In many cases of practical interest, the birefringent beat length is short compared to the length scale of the pulse variation. Then, the term in Eq. (23) is rapidly oscillating

and can be dropped. We now combine the effects of the linear and nonlinear polarizability. After transforming to the intermediate group velocity frame and appropriate normalization [8, 9], we find for linearly polarized waves,

$$\begin{aligned} iu_\xi + i\delta u_s + \frac{1}{2}u_{ss} + (|u|^2 + B|v|^2)u &= 0, \\ iv_\xi - i\delta v_s + \frac{1}{2}v_{ss} + (B|u|^2 + |v|^2)v &= 0, \end{aligned} \quad (25)$$

in the anomalous dispersion regime where $B = 2\alpha/(2\alpha + \beta)$. For circularly polarized waves, Eq. (25) still holds with $B = (\alpha + \beta)/\alpha$, and no assumption concerning the birefringence strength is required. The first derivatives in s can be removed by the transformation

$$\begin{aligned} \bar{u} &= u \exp\left[i\delta\left(1 - \frac{\delta}{2}\right)\xi - i\delta s\right], \\ \bar{v} &= v \exp\left[-i\delta\left(1 + \frac{\delta}{2}\right)\xi + i\delta s\right]. \end{aligned} \quad (26)$$

Removing the bars yields Eq. (1). We see that the Manakov equation results when $\beta = 0$.

It is worthy of note that when the birefringence is so weak that the exponential term in Eq. (1) can be set equal to 1, we find

$$\begin{aligned} iu_\xi + \frac{1}{2}u_{ss} + (|u|^2 + |v|^2)u + (1 - B)(uv^* - vu^*)v &= 0, \\ iv_\xi + \frac{1}{2}v_{ss} + (|u|^2 + |v|^2)v - (1 - B)(uv^* - vu^*)v &= 0. \end{aligned} \quad (27)$$

The final terms in Eq. (27) lead to ellipse rotation [1].

II. INTEGRABILITY AND SOLITONS

We now look for stationary solutions of Eq. (1) which have the form

$$\begin{aligned} u(\xi, s) &= \exp(i\Omega_1 \xi) f(s), \\ v(\xi, s) &= \exp(i\Omega_2 \xi) g(s), \end{aligned} \quad (28)$$

where f and g are real functions and Ω_1 and Ω_2 are two real parameters. In the case $B = 0$ where the single soliton solutions are well-known, we find that this *ansatz* yields the general solution to within a Galilean transformation. Substitution of Eq. (28) into Eq. (1) yields

$$\begin{aligned} f_{ss} - 2\Omega_1 f + 2(f^2 + Bg^2)f &= 0, \\ g_{ss} - 2\Omega_2 g + 2(Bf^2 + g^2)g &= 0. \end{aligned} \quad (29)$$

In the remainder of this section, we study Eq. (29). We apply Painlevé analysis [17] to Eq. (29) which indicates that it is only integrable when $B = 0$ or $B = 1$. Then, setting $B = 1$, we determine the homoclinic orbits which correspond to single soliton solutions.

A. Painlevé Analysis

Following the procedure of Ablowitz, *et al.* [17], we search for a Laurent series solution of Eq. (29),

$$\begin{aligned} f &= \sum_{j=0}^{\infty} a_j (s - s_0)^{p+j}, \\ g &= \sum_{j=0}^{\infty} b_j (s - s_0)^{q+j}, \end{aligned} \quad (30)$$

valid in the neighborhood of any singular point $s = s_0$. The only choice of p and q which allows us to balance leading terms in Eq. (29) while leading to four arbitrary coefficients in Eq. (30) is $p = q = -1$. We then find

$$a_0^2 = b_0^2 = -\frac{1}{B+1}. \quad (31)$$

We next determine the values of j at which arbitrary coefficients in the Laurent expansion enter. Letting $j = r$ designate these resonant values, we find that r satisfies the equations,

$$\begin{aligned} \left[(r-1)(r-2) - \frac{6}{B+1} - \frac{2B}{B+1} \right] a_r &= \pm 4 \frac{B}{B+1} b_r, \\ \left[(r-1)(r-2) - \frac{6}{B+1} - \frac{2B}{B+1} \right] b_r &= \pm 4 \frac{B}{B+1} a_r, \end{aligned} \quad (32)$$

where we have made use of Eq. (31) to eliminate a_0 and b_0 . We now find

$$(r-1)(r-2) - \frac{6}{B+1} - \frac{2B}{B+1} \mp \frac{4B}{B+1} = 0, \quad (33)$$

from which, taking the $-$ and $+$ signs in turn, we conclude that Eq. (32) has the roots

$$r = -1, \quad 4, \quad \frac{3}{2} \pm \frac{1}{2} \left(9 - 16 \frac{B-1}{B+1} \right)^{1/2}. \quad (34)$$

The only values of B which yield real, integral roots are $B = 0$, in which case $r = -1$ and $r = 4$ are both double roots, or $B = 1$, in which case $r = -1, 0, 3$, and 4 . Hence, the only values of B for which Eq. (29) can have the Painlevé property are $B = 0$ and $B = 1$.

To complete the Painlevé analysis, we must substitute Eq. (30) into Eq. (29) and show through $j = 4$ that no logarithmic singularities develop when $B = 0$ and $B = 1$. We have done so, but do not describe the algebraic details.

B. Soliton Solutions When $B = 1$

When $B = 1$, Eq. (29) is generated by the Hamiltonian

$$\mathcal{H} = \frac{1}{2}F^2 + \frac{1}{2}G^2 + \frac{1}{2}[(f^2 + g^2)^2 - 2\Omega_1 f^2 - 2\Omega_2 g^2], \quad (35)$$

where $F = df/ds$ and $G = dg/ds$ are, respectively, the momenta canonical to f and g . The independent variable is s . A second, independent constant of the motion is

$$C = \frac{1}{2}(gF - fG)^2 + (\Omega_1 - \Omega_2)[F^2 - 2\Omega_1 f^2 + (f^2 + g^2)f^2]. \quad (36)$$

Equation (36) implies the integrability of Eq. (29) when $B = 1$. When $\Omega_1 > 0$ and $\Omega_2 > 0$, homoclinic orbits exist which correspond to solitons. If $\Omega_1 = \Omega_2 \equiv \Omega$, then the solution

$$\begin{aligned} f(s) &= (2\Omega)^{1/2} \cos \alpha \operatorname{sech}[(2\Omega)^{1/2}s], \\ g(s) &= (2\Omega)^{1/2} \sin \alpha \operatorname{sech}[(2\Omega)^{1/2}s], \end{aligned} \quad (37)$$

corresponds to the solitons found by Manakov [10]. If $\Omega_1 \neq \Omega_2$, then the homoclinic orbits are considerably more complicated.

Some time ago, Darboux [11] shown that a two degree-of-freedom Hamiltonian system with a second integral quadratic in the momenta has a generic form. Once this form is obtained by using Bertrand's method [12], (see also [18]) the equations of motion can be reduced to quadratures using a procedure due to Liouville. To reduce our equation to this form, we first note that the potential contribution to $\mathcal{H}$ is

$$V(f, g) = \frac{1}{2}(f^2 + g^2)^2 - \Omega_2(f^2 + g^2) - (\Omega_1 - \Omega_2)f^2. \quad (38)$$

We next define new variables x and y such that

$$\begin{aligned} x^2 + y^2 &= f^2 + g^2 + \gamma, \\ x^2 - y^2 &= [(f^2 + g^2 + \gamma)^2 - 4\gamma f^2]^{1/2}, \end{aligned} \quad (39)$$

where $\gamma = 2(\Omega_1 - \Omega_2)$. The potential $V(x, y)$ now has the appropriate generic form,

$$V(x, y) = \frac{X(x) - Y(y)}{x^2 - y^2}, \quad (40)$$

where

$$X(\alpha) = Y(\alpha) \equiv A(\alpha) = \frac{1}{2}\alpha^2(\alpha^2 - 2\Omega_1)(\alpha^2 - 2\Omega_1 - 2\Omega_2). \quad (41)$$

To reduce the equations of motion to quadratures, we first write the kinetic contribution to $\mathcal{H}$,

$$T(F, G) = \frac{1}{2}(F^2 + G^2) = \frac{1}{2}(x^2 - y^2) \left(\frac{x_s^2}{x^2 - \gamma} + \frac{y_s^2}{\gamma - y^2} \right). \quad (42)$$

Defining now,

$$\tilde{x} = \int \frac{dx}{(x^2 - \gamma)^{1/2}} \quad \text{and} \quad \tilde{y} = \int \frac{dy}{(\gamma - y^2)^{1/2}}, \quad (43)$$

we note that T and V have the forms

$$\begin{aligned} T &= \frac{1}{2} [c_1(\tilde{x}) + c_2(\tilde{y})] (\tilde{x}_s^2 + \tilde{y}_s^2), \\ V &= \frac{d_1(\tilde{x}) + d_2(\tilde{y})}{c_1(\tilde{x}) + c_2(\tilde{y})}. \end{aligned} \quad (44)$$

Defining further $c = c_1(\tilde{x}) + c_2(\tilde{y})$ and writing the Lagrangian

$$\frac{d}{ds} \left(\frac{\partial T}{\partial \tilde{x}_s} \right) - \frac{\partial T}{\partial \tilde{x}} = - \frac{\partial V}{\partial \tilde{x}}, \quad (45)$$

we obtain after some algebra

$$\frac{d}{ds} (c^2 \tilde{x}_s^2) - c \tilde{x}_s \frac{\partial c}{\partial \tilde{x}_s} (\tilde{x}_s^2 + \tilde{y}_s^2) = -2c \tilde{x}_s \frac{\partial V}{\partial \tilde{x}}. \quad (46)$$

From the Hamiltonian, we find

$$\frac{1}{2} c (\tilde{x}_s^2 + \tilde{y}_s^2) = h - V, \quad (47)$$

where h is some constant, and, after some more algebra, we arrive at the expression

$$\frac{d}{ds} (c^2 \tilde{x}_s^2) = 2 \frac{d}{ds} (hc_1 - d_1), \quad (48)$$

or

$$\frac{1}{2} c^2 \tilde{x}_s^2 = hc_1 - d_1 + \gamma_1, \quad (49)$$

where γ_1 is a constant of integration. Carrying out a similar operation for $\tilde{y}_s$, we finally conclude

$$(hc_1 - d_1 + \gamma_1)^{1/2} d\tilde{x} = (hc_2 - d_2 + \gamma_2)^{1/2} d\tilde{y}, \quad (50)$$

which reduces the problem to quadratures.

Closed form expressions can be found for the solitons and were recently reported by Cristodoulides and Joseph [19] with some generalization from the case considered here. We do not reproduce their analytic form since it is rather complicated; however, the physical structure of the solution is not difficult to determine. When $\Omega_1 > \Omega_2$, the f -component is sharper and the g -component dominates at large values of $|s|$. The self-similar structure retains its shape through a complex balance of the contributions of the two different components.

IV. CONCLUSIONS

In this paper, we have shown how the Kerr effect leads to the coupled nonlinear Schrödinger equation in a birefringent medium. Painlevé analysis indicates that these equations are only integrable in two special cases. In the first case, the two polarizations are uncoupled. In the second case, the nonlinear contribution of the two polarizations to the Kerr coefficient of each polarization is identical. This latter case was shown to be integrable by Manakov who found special single soliton solutions. We have extended his results by finding a more general class of single soliton solutions.

ACKNOWLEDGMENTS

The author is grateful to Dr. Alfred Ramani of the Ecole Polytechnique who first pointed out to him that Eq. (29) could be integrated using Darboux's approach. Early work on this subject was carried out at the Ecole Polytechnique with support from the Centre National d'Etudes en Télécommunications. Later work was supported by the Naval Research Laboratory.

REFERENCES

1. P. D. Maker and R. W. Terhune, "Study of optical effects due to an induced polarization third order in the electric field strength," *Phys. Rev.* **137**, A801-A818 (1965).
2. H. A. Haus, *Waves and Fields in Optoelectronics* (Prentice-Hall, Englewood Cliffs, NJ, 1984), pp. 358-359.
3. V. I. Talanov, "Self-focusing of wave beams in nonlinear media," *Zh.E.T.F. Pis. Red.* **2**, 218-223 (1965) [*Sov. Phys. JETP Lett.* **2**, 138-141 (1965)].
4. P. L. Kelley, "Self-focusing of optical beams," *Phys. Rev. Lett.* **15**, 1005-1008 (1965).
5. See, e.g., R. H. Stolen, J. Botineau, and A. Ashkin, "Intensity discrimination of optical pulses with birefringent fibers," *Opt. Lett.* **7**, 512-514 (1982).

6. A. Hasegawa, "Self-confinement of multimode optical pulse in a glass fiber," *Opt. Lett.* **5**, 416-417 (1980).
7. B. Crosignani and P. Di Porto, "Soliton propagation in multimode optical fibers," *Opt. Lett.* **6**, 329-330 (1981).
8. C. R. Menyuk, "Nonlinear pulse propagation in birefringent optical fibers," *IEEE J. Quantum Electron.* **QE-23**, 174-176 (1987).
9. C. R. Menyuk, "Stability of solitons in birefringent optical fibers I: Equal propagation amplitudes," *Opt. Lett.* **12**, 614-616 (1987).
10. S. V. Manakov, "On the theory of two-dimensional stationary self-focusing of electromagnetic waves," *Zh.E.T.F.* **65**, 505-516 (1973) [*Sov. Phys. JETP* **38**, 248-253 (1974)].
11. G. Darboux, "Sur un problème de mécanique," *Archives néerlandaises* **6** (Ser. 2), 371-376 (1901). See also, E. T. Whittaker, *A Treatise on the Analytical Dynamics of Particles and Rigid Bodies* (Dover, New York, 1944), pp. 332-335.
12. M. J. Bertrand, "Sur les intégrales communes à plusieurs problèmes de mécanique," *J. Math. Pures Appliquées* **17** (Ser. 1), 121-174 (1852). See also, E. T. Whittaker, *op. cit.*, pp. 331-332.
13. J. Liouville, "Mémoire sur quelques cas particuliers où les équations différentielles du mouvement d'un point matériel peuvent s'intégrer," *J. Math. Pures Appliquées* **14** (Ser. 1), 257 (1849). See also, E. T. Whittaker, *op. cit.*, pp. 67-69.
14. A. Hasegawa and F. Tappert, "Transmission of stationary nonlinear optical pulses in dispersive dielectric fibers I. Anomalous dispersion," *Appl. Phys. Lett.* **23**, 142 (1973).
15. P. Bendow, P. D. Gianino, N. Tzoar, and M. Jain, "Theory of nonlinear pulse propagation in optical waveguides," *J. Opt. Soc. Am.* **70**, 539-546 (1980).
16. Y. Kodama, "Optical solitons in a monomode fiber," *J. Stat. Phys.* **39**, 597-614 (1985).
17. M. J. Ablowitz, A. Ramani, and H. Segur, "A connection between nonlinear evolution equations and ordinary differential equations of P-type. I," *J. Math. Phys.* **21**, 715-721 (1980).
18. B. Dorizzi, B. Grammaticos, and A. Ramani, "A new class of integrable systems," *J. Math. Phys.* **24**, 2282-2288 (1983).
19. D. N. Christodoulides and R. I. Joseph, "Vector solitons in birefringent nonlinear dispersive media," *Opt. Lett.* **13**, 53-55 (1988).

*Also at: Department of Electrical Engineering, University of Maryland, College Park, MD 21228

"Pump replication in stimulated Raman scattering using a crossed-beam geometry,"
(C.R. Menyuk, G. Hilfer, and J. Reintjes), in *Nonlinear Optics and Beam Coupling*,
R.A. Fisher, ed. (SPIE Publ. no. 874, Bellingham, WA), to appear.

Pump replication in stimulated Raman scattering using a crossed-beam geometry

C. R. Menyuk* and G. Hilfer

Science Applications International Corporation, 1710 Goodridge Drive, McLean, Va. 22102

J. Reintjes

Laser Physics Branch, Naval Research Laboratory, Washington, DC 20375

ABSTRACT

A theory of side beam replication in a crossing-beam geometry is reported. It is shown that side beam replication is not expected to occur when the Fresnel number of the aberrations (FN_A) is large, while it is expected to occur when FN_A is small, in accord with experiments. An analytic threshold is derived for the value of FN_A at which side beam replication no longer occurs, and this threshold agrees well with the experiments. We propose a method for eliminating side beam replication at low values of FN_A .

1. INTRODUCTION

The theoretical and experimental work which has been carried out to date on Raman beam cleanup and beam combining of stationary waves has been strongly motivated by previous work on phase conjugation, mostly based on Brillouin scattering, rather than Raman scattering.^{1,2} In both cases, four wave mixing processes are involved. In the early experiments of Goldhar and Murray³ counter-propagating beams were considered and the effect of a finite pump beam correlation length was determined. They show that a large number of pump beams leads to averaging and a smoother Stokes output. Shortly thereafter, Chang and Djeu⁴ carried out experiments in a co-propagating beam geometry. They found, in keeping with the theoretical predictions of Bespalov, *et al.*⁵ that as the gain rose, the Stokes beam distortion increased, due to incomplete intensity averaging along the length of the amplifier. In later work, Goldhar, *et al.*⁶ showed that their approach could be made more efficient by using a double-pass amplifier, and Chang, *et al.*^{7,8} showed that far better output Stokes quality could be obtained if a multi-beam geometry with the central pump component removed, was used. More recently, Reintjes, *et al.*^{9,10} have explored in considerable detail the different parameter regimes which occur in a multi-beam geometry and the behavior observed in the different regimes.

In their experiments, Reintjes, *et al.*^{9,10} observed that the efficiency of beam cleanup is determined in large measure by the beam geometry and by the Fresnel number of the aberrations $FN_A = D_A^2/\lambda L$, where D_A is the transverse scale length of the aberrations, λ is the pump wavelength, and L is the interaction length. In a collinear beam geometry, with a large Fresnel number so that diffraction can be ignored, the same portions of the pump beam and Stokes beam continually interact. As a consequence, no intensity averaging can occur, and any amplitude structure in the pump will print through onto the Stokes, although no phase structure prints through. As the Fresnel number decreases, intensity averaging begins to occur, reducing the deleterious effect of amplitude aberrations. However, diffraction of phase structure into amplitude structure now occurs, so that some printing through of phase aberrations takes place. As the Fresnel number decreases yet further, the intensity averaging improves substantially, but it is always incomplete.

If we consider instead a multi-beam geometry where there is no on axis pump beam, shown schematically in Fig. 1, then intensity averaging is considerably enhanced, and at small Fresnel numbers the Stokes beam is essentially diffraction limited. However, when

the Fresnel number is not small, side beam replication can occur. That is to say, new Stokes beams can be created which propagate collinearly with the off-axis pump beams.

In this work, we theoretically examine the conditions under which side beam replication occurs. This replication is closely analogous to Brillouin phase conjugation due to four wave mixing, and we make heavy use of the approach which was first developed in theoretical studies of this effect.^{1,2} Flusberg and Korff¹¹ have already noted this analogy, and they have made excellent use of it in their recent study of Raman amplification in a collinear beam geometry. In the experiments of interest to us, however, this analogy is incomplete. The difference between k_L and k_S , the pump and Stokes wavenumbers, is quite large, amounting to 13% of k_L .^{9,10} As a consequence, important modifications must be made in the theory.

Our theory leads us to propose a novel method for eliminating side beam replication without degrading pump beam quality by adjusting the phases of the incoming pump beams.

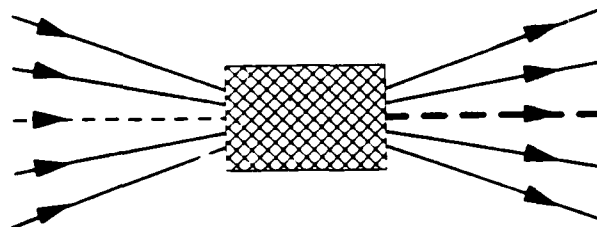


FIGURE 1 A schematic illustration of the crossing-beam geometry is shown. The pump beams are shown as solid lines, and the Stokes beam is shown as a dashed line. There is no pump beam propagating collinearly with the Stokes beam. When the Stokes beam emerges from the interaction region, shown as a hatched box, it is amplified.

2. THEORETICAL DEVELOPMENT

The basic equations which govern wave evolution in a Raman active medium are

$$\begin{aligned} \frac{\partial E_L}{\partial z} - \frac{1}{2k_L} \frac{\partial^2 E_L}{\partial y^2} &= -i \frac{k_L}{k_S} \kappa_2 Q E_S, \\ \frac{\partial E_S}{\partial z} - \frac{1}{2k_S} \frac{\partial^2 E_S}{\partial y^2} &= -i \kappa_2 Q^* E_L, \\ \frac{\partial Q}{\partial t} + \Gamma Q &= -i \kappa_1 E_S^* E_L, \end{aligned} \quad (1)$$

where E_L and E_S are the complex envelopes of the pump and Stokes waves, Q is the material excitation, κ_1 and κ_2 are the gain coefficients, and $\Gamma \equiv 1/T_2$ is the damping rate of the material excitation. Here, we consider only one transverse dimension for simplicity of presentation but note that the results which we will obtain hold without change for two transverse dimensions. To derive Eq. (1), we make a slowly varying envelope approximation, a paraxial wave approximation, and assume that the material excitation is far from saturation.

In the stationary limit of Eq. (1), where time dependent effects can be neglected, we find

$$Q = -i \kappa_1 \frac{E_S^* E_L}{\Gamma}, \quad (2)$$

and, as a consequence,

$$\begin{aligned} \frac{\partial E_L}{\partial z} - \frac{1}{2k_L} \frac{\partial^2 E_L}{\partial y^2} &= -\frac{k_L g}{k_S} |E_S|^2 E_L, \\ \frac{\partial E_S}{\partial z} - \frac{1}{2k_S} \frac{\partial^2 E_S}{\partial y^2} &= \frac{g}{2} |E_L|^2 E_S, \end{aligned} \quad (3)$$

where $g = 2\kappa_1\kappa_2/\Gamma$. The experiments^{9,10} indicate that the system's linear behavior (i.e., behavior when $|E_S| \ll |E_L|$) plays a crucial role in determining the quality of the emerging Stokes beam. We thus specialize to the linear limit where Eq. (3) becomes

$$\frac{\partial E_L}{\partial z} - \frac{1}{2k_L} \frac{\partial^2 E_L}{\partial y^2} = 0, \quad (4a)$$

$$\frac{\partial E_S}{\partial z} - \frac{1}{2k_S} \frac{\partial^2 E_S}{\partial y^2} = \frac{g}{2} |E_L|^2 E_S. \quad (4b)$$

In the multi-beam geometry that we are considering, it is always the case that in k_y -space, the Fourier transform space corresponding to the y -direction, the separation between the beams is much larger than the bandwidth of each individual beam. In other words, $(\Delta K)_{\text{sep}} \gg (\Delta K)_{\text{beam}}$, where $(\Delta K)_{\text{sep}}$ is the minimum k_y -separation between the beams and $(\Delta K)_{\text{beam}}$ is the maximum bandwidth of an individual beam. Given this condition, it is useful to decompose E_L and E_S into a sum of contributions from each beam in which the central wavenumber K_l of each beam is explicitly accounted for,

$$\begin{aligned} E_L(y, z) &= \sum_l E_L^{(l)}(y, z) \exp(iK_l y) \exp(-iK_l^2 z/2k_L), \\ E_S(y, z) &= \sum_l E_S^{(l)}(y, z) \exp(iK_l y) \exp(-iK_l^2 z/2k_S). \end{aligned} \quad (5)$$

Here, l refers to the beam number. The quantities $E_L^{(l)}$ and $E_S^{(l)}$ give the envelopes of the individual beams; these envelopes are slowly varying in the y -direction. For the case of interest to us here, it is appropriate to assume that $K_l = lK_1$, that $E_L^{(0)} = 0$, and that $E_S^{(l)}$ is very small except for $l = 0$. The $E_S^{(l)}$ for $l \neq 0$ correspond to the side beams, and it is their growth which we wish to determine.

It now follows that

$$\frac{\partial E_L^{(l)}}{\partial z} + \frac{K_l}{k_L} \frac{\partial E_L^{(l)}}{\partial y} - \frac{1}{2k_L} \frac{\partial^2 E_L^{(l)}}{\partial y^2} = 0, \quad (6)$$

and

$$\begin{aligned} \frac{\partial E_S^{(l)}}{\partial z} + \frac{K_l}{k_S} \frac{\partial E_S^{(l)}}{\partial y} - \frac{1}{2k_S} \frac{\partial^2 E_S^{(l)}}{\partial y^2} \\ = \frac{g}{2} \sum_{m,n,o} E_L^{(m)} E_L^{(n)*} E_S^{(o)} \exp[i(K_m - K_n + K_o - K_l)y] \\ \exp\left[\frac{-i}{2k_L}(K_m^2 - K_n^2 + \frac{k_L}{k_S}K_o^2 - \frac{k_L}{k_S}K_l^2)z\right]. \end{aligned} \quad (7)$$

In the previous sum, we only keep terms for which

$$K_m - K_n + K_o - K_l = 0, \quad (8)$$

or $m - n + o - l = 0$, in order to satisfy the condition that $E_S^{(l)}$ vary slowly compared with $\exp(iK_l y)$ which in turn comes from the condition $(\Delta K)_{\text{beam}} \ll (\Delta K)_{\text{sep}}$.

In general, the explicit variation in Eq. (7) can lead to rapidly oscillating terms; these terms make no contribution to the sum. Writing the e -folding growth length as $(g(I_L))^{-1}$, where (I_L) is the summed, average strength of the pump beams in the interaction region, our condition to have rapidly oscillating terms is

$$\frac{1}{2k_L} (\Delta K)_{\text{sep}}^2 \gg g(I_L), \quad (9)$$

a condition which is well-obeyed. A similar condition applies in the theory of Brillouin four wave mixing^{1,2} or Flusberg and Korff's theory¹¹ of collinear Raman interactions, although $(\Delta K)_{\text{sep}}$ is replaced by the total bandwidth. In these theories, one also assumes that a complementary condition

$$\frac{r}{2k_S} (\Delta K)_{\text{sep}}^2 \ll g(I_L), \quad (10)$$

holds, where $r = (k_L - k_S)/k_L$. As a consequence, k_L can be set equal to k_S , and we can avoid rapid oscillations when

$$K_m^2 - K_n^2 + K_o^2 - K_l^2 = 0 \quad (11)$$

which, combining with Eq. (8), implies either 1) $K_m = K_n$ and $K_o = K_l$ or 2) $K_m = K_l$ and $K_n = K_o$. The first case corresponds to terms in the equations which lead to intensity amplification of the Stokes wave; the second case corresponds to terms which lead to replication of the pump structure. There are as many terms of the second type as there are of the first; hence, Flusberg and Korff¹¹ conclude that the portion of the Stokes beam in phase with the pump grows at twice the rate of the rest of the Stokes structure.

In our experiments, Eq. (10) does not hold because of the large difference between k_L and k_S . Instead, we find

$$\frac{r}{2k_S} (\Delta K)_{\text{sep}}^2 > g(I_L) \quad (12)$$

and

$$r(\Delta K)_{\text{sep}} > (\Delta K)_{\text{beam}}. \quad (13)$$

As a consequence, rapid oscillations do not occur when either 1) $K_m = K_n$ and $K_o = K_l$, just as before, or 2) $K_m = K_l = -K_n = -K_o$, which strongly restricts the previous second case. The second condition, Eq. (13), ensures that the finite bandwidth of the $E_S^{(l)}$ does not lead to a non-zero contribution from one of the terms for which $K_m = K_l$ and $K_n = K_o$, but $K_m \neq -K_n$. We now find that

Eq. (7) becomes

$$\frac{\partial E_s^{(l)}}{\partial z} + \frac{K_l}{k_s} \frac{\partial E_s^{(l)}}{\partial y} - \frac{1}{2k_s} \frac{\partial^2 E_s^{(l)}}{\partial y^2} = \frac{g}{2} \sum_m E_L^{(m)} E_L^{(m)*} E_s^{(l)} + \frac{g}{2} E_L^{(l)} E_L^{(-l)*} E_s^{(-l)}. \quad (14)$$

In the experiments of interest to us, it is always the case that

$$\frac{1}{2k_L} (\Delta K)_{\text{beam}}^2 < g(I_L). \quad (15)$$

We may thus assume that over one growth length the effect of diffraction can be ignored. Letting $z_l = z$ and $y_l = y - (K_l/k_s)z$, we obtain

$$\frac{\partial E_s^{(l)}}{\partial z_l} = \frac{g}{2} \sum_m E_L^{(m)} E_L^{(m)*} E_s^{(l)} + \frac{g}{2} E_L^{(l)} E_L^{(-l)*} E_s^{(-l)}. \quad (16)$$

The quantity y_l measures transverse length from the center of the l th beam. The other beams' variation in z_l will in most cases be more rapid than that of the l th beam.

To analyze Eq. (16), we first consider a limiting case where FN_A is very long for the pump beams, and, at a given y_l , their amplitude variation as a function of z_l can be neglected over some long length in the interaction region. Equation (16) then has the solution for $l = 0$,

$$E_s^{(0)}(z_l, y_l) = E_s^{(0)}(0, y_l) \exp \left[\frac{g}{2} (I_L) z_l \right], \quad (17)$$

where $(I_L) = \sum_m E_L^{(m)} E_L^{(m)*}$, and we recall $E_L^{(0)} = 0$. When $l \neq 0$, the equations for l and $-l$ are coupled, and, assuming that

$$|E_L^{(l)} E_L^{(-l)*}| = (I_L)/N,$$

where N is the number of beams, we find

$$\begin{aligned} \begin{pmatrix} E_s^{(l)} \\ E_s^{(-l)} \end{pmatrix} &= \alpha \begin{pmatrix} \exp(i\phi_l) \\ 1 \end{pmatrix} \exp \left[\frac{g}{2} \frac{N+1}{N} (I_L) z_l \right] \\ &+ \beta \begin{pmatrix} \exp(i\phi_l) \\ -1 \end{pmatrix} \exp \left[\frac{g}{2} \frac{N-1}{N} (I_L) z_l \right], \end{aligned} \quad (18)$$

where $\exp(i\phi_l) = E_L^{(l)} E_L^{(-l)*} / E_L^{(l)} E_L^{(-l)*}$, and

$$\begin{aligned} \alpha &= \frac{1}{2} [E_s^{(l)}(0, y_l) \exp(-i\phi_l) + E_s^{(-l)}(0, y_l)], \\ \beta &= \frac{1}{2} [E_s^{(l)}(0, y_l) \exp(-i\phi_l) - E_s^{(-l)}(0, y_l)]. \end{aligned} \quad (19)$$

We find that the vector $(E_s^{(l)}, E_s^{(-l)})^t$ consists of two portions, a portion which is in phase with $(E_L^{(l)}, E_L^{(-l)})^t$ and grows somewhat faster than the central Stokes beam and a portion which is out of phase with $(E_L^{(l)}, E_L^{(-l)})^t$ and grows somewhat slower. On a length scale longer than $(g(I_L)/N)^{-1}$, the in phase component dominates over the out of phase component.

At this point, we can outline the condition for side beam replication to occur. If the pump beams satisfy the condition

$$\frac{1}{2k_L} (\Delta K)_{\text{beam}}^2 < g(I_L)/N, \quad (20)$$

then the pump beams $E_L^{(l)}$ and $E_L^{(-l)}$ are correlated over a length greater than $(g(I_L)/N)^{-1}$ and, as a result, the phase difference between $E_s^{(l)}$ and $E_s^{(-l)}$ is locked to the phase difference between the

pump beams. Thus, the gain of the side beams is higher than that of the central Stokes beam, and side beams will be observable if the overall gain is sufficiently large. By contrast, in the opposite limit of Eq. (20), the phases of $E_L^{(l)}$ and $E_L^{(-l)}$ change too rapidly for the phase difference of the Stokes beams to lock to them. In this case, the average growth rate of the side beams is only slightly higher than that of the central beam, and side beam replication is not expected to occur.

3. DISCUSSION

The experiments of interest to us here were carried out at the Naval Research Laboratory using a XeCl laser at 308 nm and a high pressure H₂ cell.^{9,10} The Stokes radiation emerges at 353 nm implying that $r = (k_L - k_s)/k_L = 0.13$. The angular separation between incoming pump beams is 5.6 mrad, so that $(\Delta K)_{\text{sep}} = 1.1 \times 10^3 \text{ cm}^{-1}$. Experiments were carried out with pump beams 20 or 120 times dispersion limited. In the former case, their angular spread was typically 0.03 mrad and in the latter case, it was typically 0.18 mrad, corresponding respectively to $(\Delta K)_{\text{beam}} = 6 \text{ cm}^{-1}$ and $(\Delta K)_{\text{beam}} = 37 \text{ cm}^{-1}$. The interaction length of the H₂ chamber is 500 cm and in all cases $4 < g(I_L)z < 20$, so that there is enough gain to achieve reasonable amplification without causing self-oscillation. We conclude $8 \times 10^{-3} < g(I_L) < 4 \times 10^{-2}$. The number of beams is given by $N = 24$.

We now examine our conditions to be sure that they are met. We first find

$$\frac{r}{2k_s} (\Delta K)_{\text{sep}}^2 = 0.5 \text{ cm}^{-1} > g(I_L), \quad (12')$$

$$r(\Delta K)_{\text{sep}} = 150 \text{ cm}^{-1} > (\Delta K)_{\text{beam}} = 6 - 37 \text{ cm}^{-1}. \quad (13')$$

We next examine $(1/2k_L)(\Delta K)_{\text{beam}}^2$. For the 20 times dispersion limited beam we find

$$\frac{1}{2k_L} (\Delta K)_{\text{beam}}^2 = 9 \times 10^{-5} \text{ cm}^{-1} < g(I_L), \quad (15')$$

and for the 120 times dispersion limited beam, we find

$$\frac{1}{2k_L} (\Delta K)_{\text{beam}}^2 = 3.4 \times 10^{-3} \text{ cm}^{-1} < g(I_L). \quad (15'')$$

Thus, all our basic conditions are met. We now recall $N = 24$, so that

$$g(I_L)/N = 3 \times 10^{-4} - 1.7 \times 10^{-3}.$$

When the pump beams are 20 times dispersion limited, we thus find

$$\frac{1}{2k_L} (\Delta K)_{\text{beam}}^2 < g(I_L)/N, \quad (20')$$

and we expect side beam replication to occur. By contrast, when the pump beams are 120 times dispersion limited, we find

$$\frac{1}{2k_L} (\Delta K)_{\text{beam}}^2 > g(I_L)/N, \quad (20'')$$

and no pump replication is expected. Both these results are in accord with the experiments.

We note that while the theory of Sec. 2 agrees well with the experiments, it is not sufficiently refined to lead to a precise determination of the boundaries between the different regimes. Here, numerical simulations are likely to be of assistance, and we intend to carry them out in the near future.

Finally, we turn to methods for eliminating side beam replication. These include: 1) Since $E_L^{(+)}$ and $E_L^{(-)}$ must both be non-zero for pump beam replication to occur, we can arrange the pump beams asymmetrically. Unfortunately, this approach leads to asymmetries in the Stokes amplification and degradation of the beam quality. 2) We can increase the number of pump beams. This approach does not appear to be practical. 3) We can phase $E_L^{(+)}$ and $E_L^{(-)}$ so that they are out of phase with each other. If, as seems likely, the Stokes beams are seeded almost symmetrically by scattering from the central beam, the pump and Stokes beams should be out of phase. This approach appears promising, and we intend to explore it.

4. ACKNOWLEDGMENT

The work of Drs. Menyuk and Hilfer was supported by the Naval Research Laboratory.

* Permanent address: Department of Electrical Engineering, University of Maryland, Baltimore, MD 21228.

5. REFERENCES

1. R. W. Hellwarth, "Theory of phase conjugation by stimulated scattering in a waveguide," *J. Opt. Soc. Am.* **68**(8), 1050-1056 (1978).
2. B. Ya. Zel'dovich, N. F. Pilipetskiĭ, and V. V. Shkunov, "Phase conjugation in stimulated scattering," *Usp. Fiz. Nauk* **138**(10), 249-288 (1982) [*Sov. Phys. Usp.* **25**(10), 713-737 (1982)].
3. J. Goldhar and J. R. Murray, "Intensity averaging and four-wave mixing in Raman amplifiers," *J. Quantum Electron.* **QE-18**(3), 399-409 (1982).
4. R. S. F. Chang and N. Djeu, "Amplification of a diffraction-limited Stokes beam by a severely distorted pump," *Opt. Lett.* **8**(3), 139-141 (1983).
5. V. I. Bespalov, A. A. Betin, and G. A. Pismanik, "Reproduction of the pumping wave in stimulated-scattering radiation," *Radiophys. Quantum Electron.* **21**(7), 675-688 (1978) [*Izv. Vyssh. Uchen. Zaved. Radiofiz.* **21**(7), 961-980 (1978)].
6. J. Goldhar, M. W. Taylor, and J. R. Murray, "An efficient double-pass Raman amplifier with pump intensity averaging in a light guide," *IEEE J. Quantum Electron.* **QE-20**(7), 772-785 (1984).
7. R. S. F. Chang, M. T. Duignan, R. H. Lehmberg, and N. Djeu, "Use of stimulated Raman scattering for reducing the divergence of severely aberrated laser beams," in *Excimer Lasers, Their Applications, and New Frontiers in Lasers*, Proc. SPIE **476**, 81-89 (1984).
8. R. S. F. Chang, R. H. Lehmberg, M. T. Duignan, and N. Djeu, "Raman beam cleanup of a severely aberrated pump laser," *J. Quantum Electron.* **QE-21**(5), 477-487 (1985).
9. J. Reintjes, R. H. Lehmberg, G. Calame, and M. T. Duignan, "Applications of Raman beam cleanup and phase conjugation to the reduction of laser beam aberrations," in *Laser Research and Development in the Northeast*, Proc. SPIE **709**, 20-27 (1986).
10. J. Reintjes, R. H. Lehmberg, R. S. F. Chang, M. T. Duignan, and G. Calame, "Beam cleanup with stimulated Raman scattering in the intensity-averaging regime," *J. Opt. Soc. Am. B* **3**(10), 1408-1427 (1986).
11. A. Flusberg and D. Korff, "Wave-front replication versus beam cleanup by stimulated scattering," *J. Opt. Soc. Am. B* **4**(5), 687-690 (1987).

"Asymptotic evolution of transient pulses undergoing stimulated Raman scattering,"
(C.R. Menyuk and G. Hilfer), to be submitted to Optics Lett.

ASYMPTOTIC EVOLUTION OF TRANSIENT PULSES UNDERGOING STIMULATED RAMAN SCATTERING

by

Curtis R. Menyuk* and Godehard Hilfer
Science Applications International Corporation
1710 Goodridge Drive
McLean, VA 22102

Abstract

Propagation of short, transient pulses undergoing stimulated Raman scattering over long length scales is considered. It is shown that under common experimental circumstances, the evolution has two different regimes: 1) The *I*-regime, at short lengths, where the pump changes little and the Stokes rapidly grows, and 2) the *J*-regime, at long lengths, where the Stokes intensity is close to saturation and the pump intensity decreases slowly as the square root of distance. The distance at which the *J*-regime is reached is determined numerically.

*Permanent Address: Department of Electrical Engineering
University of Maryland
Catonsville, MD 21228

Asymptotic Evolution of Transient Pulses Undergoing Stimulated Raman Scattering

Since the early work of Carmen, *et al.*¹ the evolution of pulses undergoing stimulated Raman scattering has been a subject of constant interest.²⁻¹² In the limit considered by Carmen, *et al.*² where diffraction, level saturation, interaction with anti-Stokes or higher order Stokes radiation, and quantum noise can all be ignored, the wave interaction is governed by the equations

$$\begin{aligned}\frac{\partial E_L}{\partial z} &= -i \frac{k_L}{k_S} \kappa_2 Q E_S, \\ \frac{\partial E_S}{\partial z} &= -i \kappa_2 Q^* E_L, \\ \frac{\partial Q}{\partial t} + \Gamma Q &= -i \kappa_1 E_S^* E_L.\end{aligned}\tag{1}$$

The purpose of this letter is to revisit Eq. (1) in the highly transient limit where $T_2/\tau \ll 1$. The quantity $T_2 = \Gamma^{-1}$ is the damping time of the material excitation, and τ is the full width at half maximum (FWHM) pulse intensity. This limit is relevant to recent experiments which have been carried out at the Naval Research Laboratory.^{11,12}

It has long been known that in the initial growth phase where $|E_S| \ll |E_L|$, Eq. (1) can be linearized in a simple way, allowing for a simple characterization of the solution. We shall show that in the limit of large z , a simple description is once again possible. In effect,

$$K(t) \equiv |E_L(z, t)|^2 + \frac{k_L}{k_S} |E_S(z, t)|^2\tag{2}$$

remains constant for all z . We thus find that as the pump intensity diminishes, the Stokes intensity grows, ultimately taking on the shape of the initial pump. One can then assume that the Stokes intensity is fixed and carry out a theory analogous to that of Carmen, *et al.*¹ One finds, however, that the I -Bessel functions are replaced by J -Bessel functions.

Recent experiments at the Naval Research Laboratory have studied transient pulses short compared to T_2 .^{11,12} These pulses typically have a slight chirp proportional to the pump

intensity, but no rapid phase shift. Under these circumstances, numerical results indicate that there is a rapid transition between the *I*-regime where the theory of Carmen, *et al.*¹ applies and the *J*-regime where the theory to be presented shortly applies. In other experimental settings, where a phase shift which is rapid compared to the pulse size is present, a soliton-like structure can form;^{7,10} however, its velocity is smaller than light, so that it must ultimately travel to the back end of the pulse and disappear if the pulse size is short compared to T_2 . Moreover, we will show that soliton-like structures cannot form when the pulse size is short compared to T_2 if two conditions are met: 1) the initial Stokes amplitude is small compared to the pump, and 2) there is no phase variation in the leading edge of the pulse.

We begin our theoretical development by recalling that in the *I*-regime, the solution to Eq. (1) is given by

$$E_S(z, t) = E_S(0, t) + (\kappa_1 \kappa_2 z)^{1/2} E_L(t) \int_{-\infty}^t \exp[-\Gamma(t - t')] E_L^*(t') E_S(0, t') [\tau(t) - \tau(t')]^{-1/2} I_1 \left(2 \{ \kappa_1 \kappa_2 z [\tau(t) - \tau(t')] \}^{1/2} \right) dt' , \quad (3a)$$

$$Q(z, t) = -i\kappa_1 \int_{-\infty}^t \exp[-\Gamma(t - t')] E_L(t') E_S^*(0, t') I_0 \left(2 \{ \kappa_1 \kappa_2 z [\tau(t) - \tau(t')] \}^{1/2} \right) dt' , \quad (3b)$$

where

$$\tau(t) = \int_{-\infty}^t K(t') dt' . \quad (4)$$

We now solve Eq. (3a) approximately using the method of steepest descent.¹³ In the regime which we are considering, where T_2 is much larger than the pulse width, most of the contribution to the integral comes from a restricted region in t' where the rapid increase in E_L and/or E_S at their leading edges balances the rapid decrease in the Bessel function as $\tau(t')$ approaches $\tau(t)$. The steepest descent path is along the real t -axis. The details of the solution depend on the rapidity with which E_L and E_S vary in the neighborhood of the steepest descent point.

We now assume that the initial Stokes pulse leads the pump pulse and is varying slowly at the steepest descent point; this assumption corresponds to maximum gain.^{11,12} We will further assume that leading edge of the pump varies exponentially. We define now

$$\begin{aligned}s &\equiv 4\kappa_1\kappa_2z[\tau(t) - \tau(t')] , \\ s_\infty &\equiv 4\kappa_1\kappa_2z\tau(t) ,\end{aligned}\tag{5}$$

and note that when s is large

$$I_1(s^{1/2}) \simeq \exp[s^{1/2} - \frac{1}{2}\ln(2\pi s^{1/2})] .\tag{6}$$

Physically, we are assuming that z is large enough so that the Stokes pulse has undergone substantial gain, but is not so large that pump depletion has begun. For these assumptions to be consistent, the initial Stokes amplitude must be small relative to the pump amplitude. At the leading edge of the pump pulse, we write by assumption

$$E_L(t') = A_L \exp(\Gamma_\omega t') \exp[i\phi_L(t')] ,\tag{7}$$

where A_L, Γ_ω and ϕ_L are all real. Equation (7) effectively defines all three quantities. It is useful to define another quantity $\tau(t)$ through the relationship

$$\tau(t) = r^2(t)A_L^2/\Gamma_\omega .\tag{8}$$

We stress that s , and thus the steepest descent point t_0 , is a function of t .

In the case being considered, we may write $E_S(t') = A_S \exp[i\phi_S(t')]$. Both phases ϕ_L and ϕ_S are assumed to be slowly varying. Gathering together all the rapidly varying terms and substituting the results into Eq. (3a), we find that the argument of the resulting exponent is given by

$$\psi = (\Gamma_\omega + \Gamma)t' + s^{1/2} - \frac{1}{2}\ln(2\pi s^{3/2}) .\tag{9}$$

The steepest descent point is the point at which $d\psi/dt' = 0$. This point satisfies the relation

$$\Gamma_w + \Gamma + \frac{3}{4} \frac{|E_L|^2(t')}{[r(t) - r(t')]} - \frac{(\kappa_1 \kappa_2 z)^{1/2} |E_L|^2(t')}{[r(t) - r(t')]^{1/2}} = 0 \quad (10)$$

At large z with t inside the main part of the pulse, t' is out on the leading edge of the pulse; hence, $r(t') \ll r(t)$ and may be neglected to lowest order in $s_\infty^{1/2}$. Using Eq. (7), we conclude

$$t_0 = \frac{1}{2\Gamma_w} \ln \left[\frac{2(1 + \Gamma/\Gamma_w) r^2(t)}{s_\infty^{1/2} - 3/2} \right] \quad (11)$$

Carrying out the remainder of the steepest descent calculation,¹³ we find

$$E_S(z, t) = 2\kappa_1 \kappa_2 z \left[\frac{\pi}{\Gamma_w^2 (1 + \Gamma/\Gamma_w)} \right]^{1/2} E_L(t) E_S(0, t_0) E_L^*(t_0) \exp[-\Gamma(t - t_0)] \exp(s^{1/2}) / (2\pi s^{3/2})^{1/2}, \quad (12)$$

where s is evaluated at $t = t_0$. We stress that this calculation is not asymptotic in z as the exponential rise of E_L is controlled by Γ_w , not z . It does, however, yield a useful approximation. We have compared Eq. (12) to numerically calculated exact solutions of Eq. (1) in several instances, and we have shown that they agree well to within factors of order unity in the appropriate parameter regime.

We can obtain a number of results directly from these calculations. First, the phase difference $\phi_L(t) - \phi_S(t)$ in the bulk of the Stokes pulse at large z is controlled by the phase difference $\phi_L[t_0(t)] - \phi_S[t_0(t)]$ at $z = 0$. At large z , the range of t_0 -values controlling the bulk phases reaches a constant value. Considering the half-widths at half maxima, we find

$$\Delta t_0 = \frac{1}{2\Gamma_w} \ln[r^2(\tau/2)/r^2(-\tau/2)] \quad (13)$$

Second, since the central t_0 -value decreases with increasing z , we conclude that if the initial phase difference approaches a constant value, soliton-like structures cannot form. Third, for any given z and t , it follows from Eq. (12) that the maximum growth is obtained by placing

$E_S(t_0)$ at the steepest descent point. It is not trivial to determine precisely the optimum offset for the Stokes pulse from this calculation as we must sum the contribution at all values of t ; however, we immediately conclude that the Stokes pulse should precede the pump pulse by an amount on the order of $1/\Gamma_\omega$. These conclusions all agree well with available experimental and computational results.^{11,12}

We turn now to the J -regime. In this regime, where the Stokes intensity is close to its asymptotic value, we find

$$E_L(z, t) = E_L(z_0, t) - [\kappa_1 \kappa_2 (z - z_0)]^{1/2} \frac{k_L}{k_S} E_S(t) \int_{-\infty}^t \exp[-\Gamma(t - t')] E_S^*(t') E_L(0, t') \quad (14a)$$

$$[\tau(t) - \tau(t')]^{-1/2} J_1 \left(2 \{ \kappa_1 \kappa_2 (z - z_0) [\tau(t) - \tau(t')] \}^{1/2} \right) dt' ,$$

$$Q(z, t) = -i \kappa_1 \int_{-\infty}^t \exp[-\Gamma(t - t')] E_S^*(t') E_L(z_0, t') \quad (14b)$$

$$J_0 \left(2 \{ \kappa_1 \kappa_2 (z - z_0) [\tau(t) - \tau(t')] \}^{1/2} \right) dt' ,$$

These equations can be derived using the approach described by Wang¹⁴ or verified by substitution into Eq. (1). Using the asymptotic expression

$$J_n(x) = \left(\frac{2}{\pi x} \right)^{1/2} \cos \left(x - \frac{1}{2} n \pi - \frac{1}{4} \pi \right) , \quad (15)$$

we reach the following conclusions: First, at large z , the amplitude E_L at any time scales like $z^{-1/4}$, multiplied by a periodic variation. The total integrated intensity must therefore scale as $z^{-1/2}$. Second, the number of zero-crossings of the real and imaginary parts of E_L and Q scale like $z^{1/2}$. Third, new zeros enter the E_L and Q pulses at large t and travel toward smaller t as z increases.

In order to verify these trends, we have considered a large number of numerical solutions which will be presented in detail in a later publication. A small fraction of these results are sum-

marized in Figs. 1 and 2. In these figures we display $R = [\int_{-\infty}^{\infty} dt |E_L|^2(0) / \int_{-\infty}^{\infty} dt |E_L|^2(\zeta)]^2$ vs. $\zeta \equiv \kappa_1 \kappa_2 z \tau(\infty)$ and N (the number of zero-crossings of E_L) vs. ζ , where N is plotted on a parabolic scale. We display three different cases, in all of which E_L and E_S are purely real and Q is purely imaginary. The initial Stokes intensity is 10^{-3} of the pump intensity at all points in time. The maximum intensity of the pump is the same in all three examples, and their FWHM values of the the initial pump profiles are chosen so that they all have nearly the same integrated intensity. In all cases, the FWHM is roughly 40 ps and $T_2 = 633$ ps. We find that the observed scaling agrees well with the analytical predictions. Moreover, for all the cases shown here, the linear scaling is obtained when $\zeta \simeq 120$ and $R \simeq 10$, corresponding to 70% pump depletion. In our examination of a large number of different cases, we have noted the following trends for pulses short compared to T_2 : 1) There is little dependence on the pulse shape. 2) When the Stokes offset is decreased, i.e., the Stokes pulse arrives at the Raman cell earlier than the pump pulse, the pump must deplete more before the J -regime is reached. With a negative offset equal to the FWHM, the pump must be 90% depleted before R scales linearly. The scaling of N is, however, only slightly affected. Moreover, the ζ -value at which the J -regime is reached only increases from $\zeta = 120$ to $\zeta = 180$. 3) When the Stokes offset is increased so that the Stokes pulses arrives after the pump, one finds that beyond 70% pump depletion R scales linearly. However, linear scaling of N^2 is delayed. With a positive offset equal to the FWHM, this scaling sets in at around $\zeta = 180$. 4) When a chirp proportional to the pump amplitude is added to the pump and/or Stokes, no effect is observed when the magnitude of the chirp is approximately π , the experimental value. When the magnitude increases to approximately 10π , the ζ -value at which the J -regime is reached increases slightly, by under 50, at all Stokes offsets, with no observed alteration in the basic trends.

In this letter, we have considered the effect of stimulated Raman scattering on short,

transient pulses. We have shown that under normal experimental circumstances where the phase difference between the pump and Stokes pulses at their leading edges reaches a constant and where the Stokes pulse precedes the pump, the Stokes pulse will phase lock to the pump and grow steadily. Once the Stokes has nearly saturated, the total pulse enters a new regime where the pump intensity decreases as the square root of distance and the pump amplitude oscillates with a frequency proportional to the square of the distance.

This work was supported by the Naval Research Laboratory, and we gratefully acknowledge useful conversations with Dr. J. Reintjes.

REFERENCES

1. R. L. Carmen, F. Shimizu, C. S. Wang, and N. Bloembergen, *Phys. Rev. A* **2**, 60 (1970).
2. N. Tan-no, T. Shirahata, K. Yokoto, and H. Inaba, *Phys. Rev. A* **12**, 159 (1975).
3. G. I. Kachen and W. H. Lowdermilk, *Phys. Rev. A* **14**, 1472 (1976).
4. J. N. Elgin and T. B. O'Hare, *J. Phys. B* **12**, 159 (1979).
5. M. G. Raymer, J. Mostowski, and J. L. Carlsten, *Phys. Rev. A* **19**, 2304 (1979).
6. M. G. Raymer and J. Mostowski, *Phys. Rev. A* **24**, 1980 (1981).
7. F. Y. F. Chu and A. C. Scott, *Phys. Rev. A* **12**, 2060 (1975).
8. K. Druhl, R. G. Wenzel, and J. L. Carlsten, *Phys. Rev. Lett.* **51**, 1171 (1983).
9. D. J. Kaup, *Physica* **6D**, 143 (1983).
10. H. Steudel, *Physica* **6D**, 155 (1983).
11. M. D. Duncan, R. Mahon, L. L. Tankersley, and J. Reintjes, *JOSA B* **5**, 37 (1988).
12. M. D. Duncan, R. Mahon, L. L. Tankersley, and J. Reintjes in *Beam Combining and Atmospheric Propagation* (Proc. SPIE 784, Bellingham, WA, to appear).
13. See, e.g. P. M. Morse and H. Feshbach, *Methods of Theoretical Physics* (McGraw, New York, 1953), pp. 434-443.
14. C. S. Wang, *Phys. Rev.* **182**, 482 (1969).

FIGURE CAPTIONS

1. Plots of R vs. ζ for different pulse shapes. a) sech-squared amplitude, FWHM = 40 ps; b) Lorentzian-squared amplitude, FWHM = 39 ps; c) Square pulse, FWHM = 43.8 ps.
2. Plots of N vs. ζ ; N is plotted on a parabolic axis. Shapes and parameters are the same as in Fig. 1.

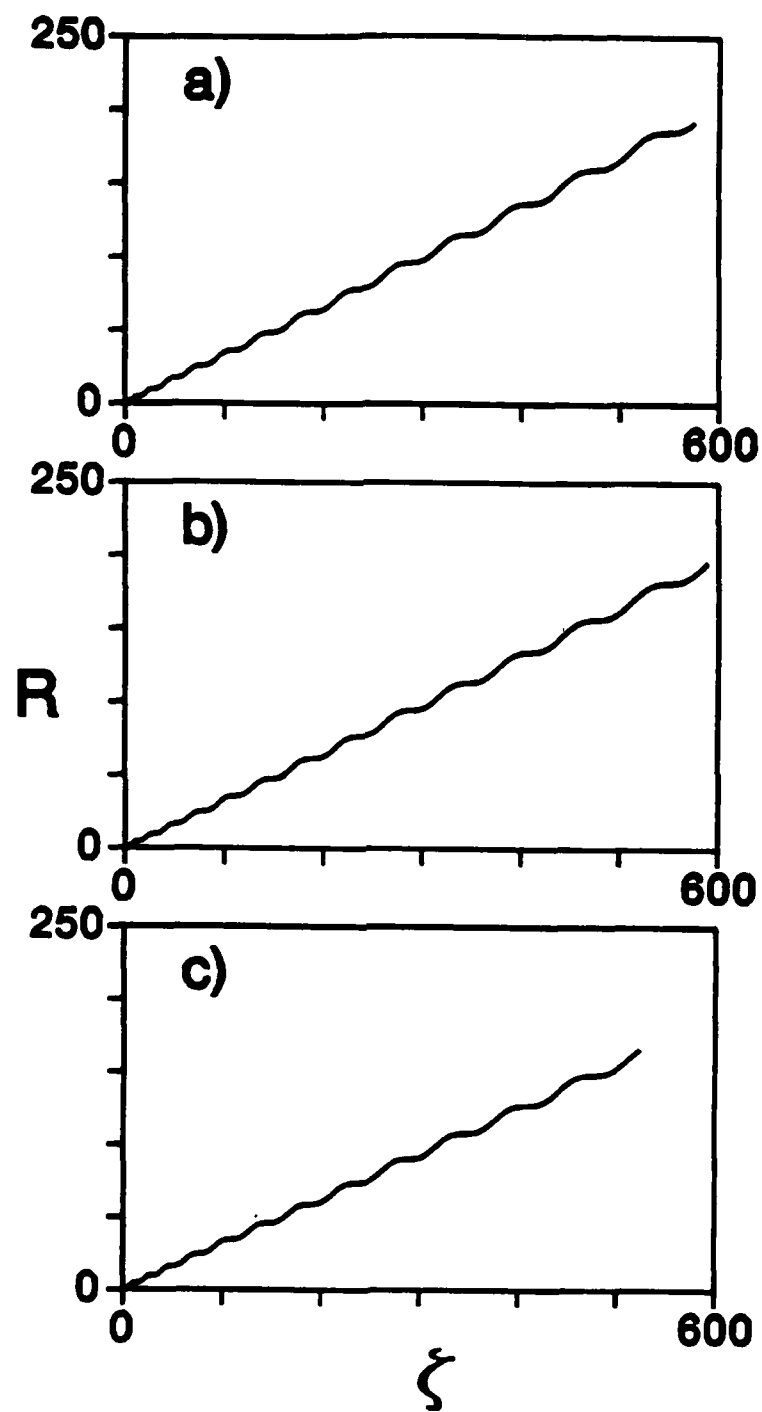


FIGURE 1.

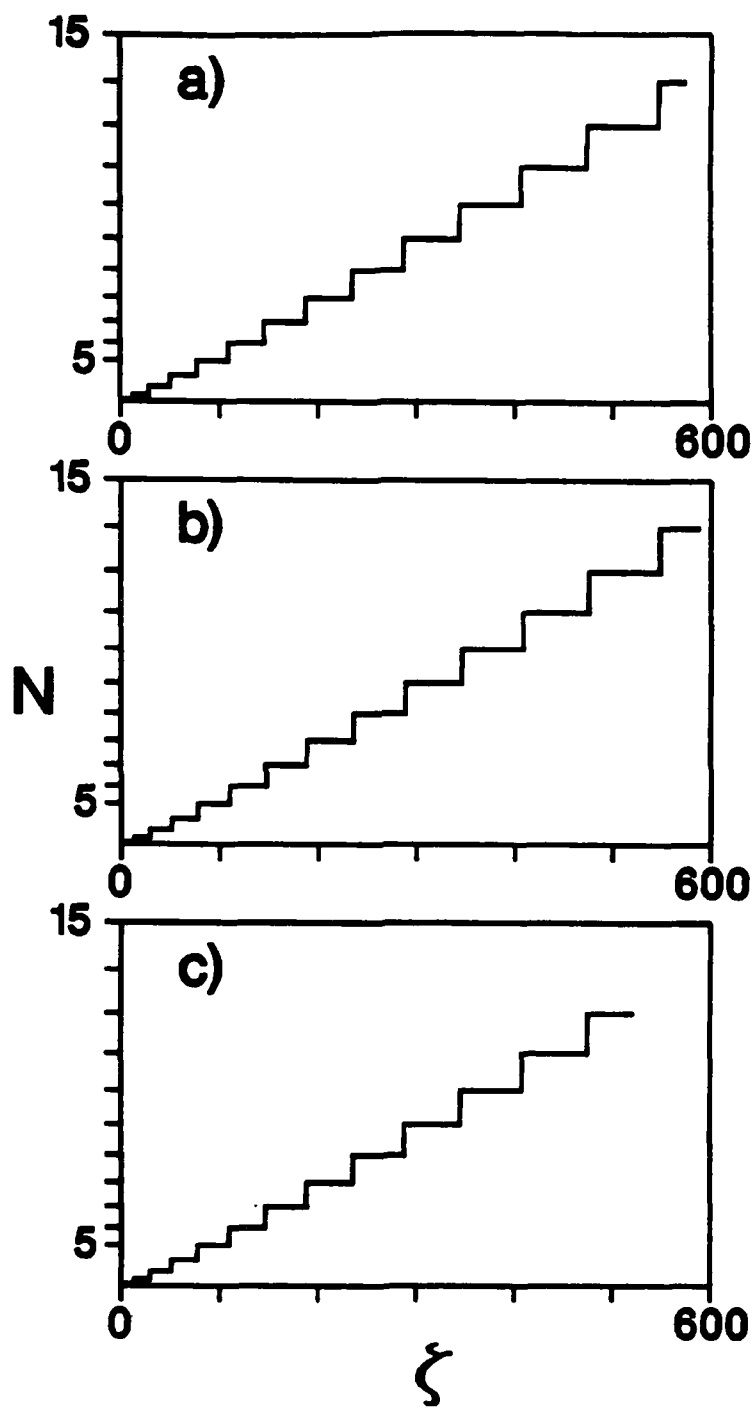


FIGURE 2 .

APPENDIX D

Presentations

IVth Workshop on Nonlinear Evolution Equations and Dynamical Systems,
(Balaruc-les-Bains, France, June 6-25, 1978).

LIE PERTURBATION METHODS AND THEIR APPLICATION
TO INFINITE-DIMENSIONAL, HAMILTONIAN SYSTEMS

Curtis R. MENYUK

University of Maryland -
Department of Electrical
Engineering -
COLLEGE PARK, MD 20742
U.S.A.

ABSTRACT

The Lie perturbation method of Hori and Deprit is a practical approach for determining the evolution of finite-dimensional, nearly integrable, Hamiltonian systems. It has been applied with notable success to problems including satellite motion around the Earth and particle motion in accelerators. We review this approach and describe how to extend it to infinite-dimensional systems. Explicit first and second order calculations are described in cases where the initial data contains a single solitary wave and a small amount of radiation. Implications for the emergence of solitary waves from arbitrary initial data are discussed.

REFERENCES

1. C.R. Menyuk and H.H. CHEN, On the Hamiltonian structure of ion acoustic waves and shallow channel water waves, *Phys. Fluids* 29, 998-1003 (1986).
2. C.R. MENYUK, Origin of solitons in the "real" world, *Phys. Rev. A* 33, 4367-4374 (1986).
3. C.R. MENYUK, Nonlinear pulse propagation in optical fibers, *I.E.E.E. J. Quantum Electron.* QE-23, 174-176 (1986).

Annual Meeting of the Optical Society of America (Rochester, NY, October 18-23, 1987), papers MI7 and MI12.

MI7 Numerical modeling of transient Raman amplification

GODEHARD HILFER, Science Applications International Corp., 1710 Goodridge Dr., McLean, VA 22102; CURTIS R. MENYUK, U. Maryland, College Park, MD 20742.

To model experiments performed by Reintjes *et al.*,¹ a numerical code has been developed that solves the Raman interaction equations:

$$\frac{\partial A_L}{\partial z} - \frac{i}{2k_L} \frac{\partial^2 A_L}{\partial y^2} = ik_L \frac{\omega_L}{\omega_S} A_S Q,$$

$$\frac{\partial A_S}{\partial z} - \frac{i}{2k_S} \frac{\partial^2 A_S}{\partial y^2} = ik_S A_L Q^*,$$

$$\frac{\partial Q}{\partial t} + \Gamma Q = ik_L A_L A_S.$$

The purpose is to study the influence of transience, transverse beam structure, pump depletion, and dispersive effects on the amplification and phase modulation of diverse forms of transient Stokes and pump beams in a cross-beam geometry.

Preliminary results of these simulations are reported for the beam parameters of the NRL experiments (12 min)

1. J. Reintjes, R. H. Lehmberg, R. S. F. Chang, M. T. Duignan, and G. Calame, "Beam Cleanup with Stimulated Raman Scattering in the Intensity-Averaging Regime," *J. Opt. Soc. Am. B* 3, 1408 (1986), see Table 1.

MI12 Linear theory of Raman beam cleanup and amplification in a crossing beam geometry

CURTIS R. MENYUK, U. Maryland, Department of Electrical Engineering, Catonsville, MD 21228

In experiments which have been performed to date at the Naval Research Laboratory,¹ it has been discovered that the results can be characterized by the Fresnel number of the aberrations $D_A^2/\lambda L$, where D_A is the transverse scale length of the aberrations, L is the interaction length of the pump and Stokes beams, and λ is the pump wavelength. The results are characterized by the geometry (collinear or crossing beam) and aberrations (phase, amplitude, or both). We show that many of the experimentally observed effects can be explained by a linear theory, although nonlinear effects due to pump depletion are in most cases important. The circumstances in which off-angle contributions, copropagating with the crossing beams, are seeded by the pump and can grow to important levels are elucidated. (12 min)

1. J. Reintjes, R. H. Lehmberg, R. S. F. Chang, M. T. Duignan, and G. Calame, "Beam Cleanup with Stimulated Raman Scattering in the Intensity-Averaging Regime," *J. Opt. Soc. Am. B* 3, 1408 (1986), see Table 1.

Workshop on Solitons and Chaos in Optical Systems (San Jose, CA, January 6-7, 1988).

SOLITONS IN A KERR MEDIUM

Curtis R. Menyuk
Department of Electrical Engineering
University of Maryland
Baltimore, MD 21228

ABSTRACT: In a weakly dispersive medium with a Kerr nonlinearity, the equations governing the electromagnetic wave evolution can be written as

$$\begin{aligned}u_{\xi} + \frac{1}{2}u_{ss} + (|u|^2 + B|v|^2)u &= 0, \\v_{\xi} + \frac{1}{2}v_{ss} + (B|u|^2 + |v|^2)v &= 0,\end{aligned}$$

where u and v are the envelopes of the two polarization amplitudes, ξ is the normalized distance, s is the normalized time, and B depends on the medium in question. For media such as optical fibers, where the Kerr response is essentially instantaneous, $B = 2/3$ [See, e.g. C. R. Menyuk, IEEE J. Quantum. Electron. **QE-23**, 174 (1986)]. Painlevé analysis indicates that this system is only integrable when $B = 0$ or $B = 1$. Both these cases merit profound study since they provide a useful starting point for studying the general case where $B \neq 0$ or 1 . In the former case, this system reduces to two uncoupled nonlinear Schrödinger equations. The nonlinear Schrödinger equation has been extensively studied. In the latter case, this system has been studied by Manakov [Sov. Phys. JETP **38**, 248 (1974)] who found the Lax pair for this system, showed how to solve the Cauchy problem using spectral transform methods, and explicitly derived single soliton solutions where both u and v are proportional to $\text{sech}(\alpha s)$. In this work it is shown by a direct search for stationary solutions that the class of single soliton solutions is substantially larger than the sech -like class found by Manakov. The soliton profiles can be obtained in the general case by using an approach originally described by Darboux [Archives néerlandaises, Ser. 2, 6, 371 (1901)].

SPIE: O-E LASE '88. Nonlinear Optics and Beam Combining (Los Angeles, CA, January 11-15, 1988), paper 874-50.

TUESDAY 12 JANUARY 1988

Registration and Information,

Hilton Pavilion	7:15 am to 6:00 pm
Central Message Desk, Hilton Pavilion	7:15 am to 6:00 pm
Information Desk,	
Marriott Ballroom Lobby	7:15 am to 6:00 pm
Breakfast Breads and Coffee,	
Marriott Ballroom Lobby	7:30 to 8:30 am
Speakers' Audiovisual Desk,	
Marriott Ballroom Lobby	7:30 am to 5:00 pm
Placement Service Center,	
Hilton International Ballroom	10:00 am to 4:00 pm
Exhibits, Hilton Pavilion	
Hilton International Ballroom	10:00 am to 6:00 pm

SESSION 3 **Tues. 8:15 am**

Nonlinear Optics and Beam Combining III

Chair: Matthew B. White, Office of Naval Research

Invited Paper: Raman beam combining using broadband XeCl laser radiation, M. N. Ediger, J. F. Reintjes, Naval Research Lab. [874-13]

Nonlinear hydrodynamic effects in gaseous SBS media, D. M. Walsh, B. S. Masson, U.S. Air Force Weapons Lab. [874-14]

Coherent beam processing concepts, P. Yeh, A. E. T. Chiou, I. C. McMichael, M. Khoshnevisan, Rockwell International Science Ctr. [874-15]

Characterization of asymmetric self-defocusing and centrosymmetric scattering in barium titanate, T. R. Moore, Lawrence Livermore National Lab.; D. L. Walters, U.S. Naval Post Graduate School [874-48]

***Coffee Break** **10:00 to 10:30 am**

Invited Paper: Beam combining in a gas via nonlinear, diffractive optics, J. S. Chivian, LTV Missiles and Electronic Group; C. D. Cantrell, Univ. of Texas at Dallas; W. D. Cotten, LTV Missiles and Electronics Group; C. A. Glosson, Univ. of Texas, Dallas [874-16]

High frequency stimulated Brillouin scattering experiments, M. E. Farey, C. G. Koop, TRW, Inc. [874-17]

Invited Paper: Stokes-anti-Stokes gain suppression in the transient regime, A. B. Hickman, W. K. Bischel, SRI International [874-18]

SPIE-Hosted Picnic-style Lunch, Hilton Plaza (Lower Level) Noon to 1:00 pm
Dessert in the Exhibit Halls, Hilton Pavilion and International Ballroom 1:00 to 2:00 pm

SESSION 4 **Tues. 2:00 pm**

Nonlinear Optics and Beam Combining IV

Chair: Pochi Yeh, Rockwell International Science Center

Invited Paper: Stimulated Brillouin scattering aberration control, M. J. LeFebvre, S. J. Pteiler, TRW, Inc. [874-19]

Coherent beam combination via microparticle plasma modes, D. N. Rogovin, T. P. Shen, Rockwell International Science Ctr. [874-20]

Pump replication in stimulated Raman scattering using a crossed beam geometry, C. R. Menyuk, G. Hilfer, Science Applications International Corp.; J. Reintjes, Naval Research Lab. [874-50]

***Coffee Break** **3:40 to 4:00 pm**

Invited Paper: Laser beam combining through the nonlinear response of a strongly driven atomic transition, K. R. MacDonald, M. T. Gruneisen, R. W. Boyd, Univ. of Rochester [874-22]

Orientational Kerr effect for millimeter wave applications, R. L. McGraw, Rockwell International Science Ctr. [874-23]

One-way transmission of images through a multimode optical fiber by degenerate four-wave mixing in a photorefractive BSO crystal, E.-S. Kim, California Institute of Technology. [874-24]

Frequency adding media for short wavelength gases and phase-insensitive beam combinations, J. A. Goldstone, J. P. Stone, Rockwell International Corp./Rocketdyne Div. [874-25]

WEDNESDAY 13 JANUARY 1988

Breakfast Breads and Coffee

Marriott Ballroom Lobby	7:30 to 8:30 am
Registration and Information,	
Hilton Pavilion	7:30 am to 5:00 pm
Central Message Desk, Hilton Pavilion	7:30 am to 5:00 pm
Information Desk,	
Marriott Ballroom Lobby	7:30 am to 5:00 pm
Speakers' Audiovisual Desk,	
Marriott Ballroom Lobby	7:30 am to 5:00 pm
Placement Service Center,	
Hilton International Ballroom	10:00 am to 4:00 pm
Exhibits, Hilton Pavilion,	
Hilton International Ballroom	10:00 am to 5:00 pm

SESSION 5 **Wed. 8:00 am**

Nonlinear Optics and Beam Combining V

Chair: Robert A. Fisher, R.A. Fisher Consulting

Invited Paper: Phase pulling in transient Raman amplifiers, M. D. Duncan, R. Mahon, L. L. Tankersley, J. F. Reintjes, Naval Research Lab. [874-26]

Four-wave mixing in cesium vapor, R. St. Pierre, A. Horwitz, J. Brock, TRW, Inc. [874-27]

Invited Paper: Adaptive optic phase compensation of an aperture combined Raman laser, J. R. Oldenettel, L. Cuellar, C. N. Howten, E. Newman, K. Roff, K. Y. Tang, Western Research Corp. [874-28]

Conditions for spontaneous generation of solitons in stimulated Raman scattering, C. M. Bowden, U.S. Army Missile Command, J. C. Englund, Southern Methodist Univ. [874-29]

***Coffee Break** **10:10 to 10:30 am**

New applications and designs for deformable mirrors, E. S. Bliss, J. R. Smith, R. L. Miller, Lawrence Livermore National Lab. [874-30]

Atmospheric effects on target detection with an imaging radiometer, T.-S. Chu, AT&T Bell Labs. [874-31]

Search techniques for wavefront estimation by phase retrieval, M. E. Dorros, AT&T Technologies; R. A. Gonsalves, Tufts Univ. [874-49]

**Coffee will be served in the Hilton Pavilion and in the Marriott Ballroom Lobby.*

1:00 PM-2:30 PM

WM13 Suppression of Feedback-Induced Noise in Semiconductor Lasers by a Combination of Optoelectronic Negative Feedback and High-Frequency Superimposition. Nonyuki Yoshikawa, Mitsuo Tamura, Ken Hamada, Masahiro Kume, Hirokazu Shimizu, Gota Kano, Iwao Teramoto, *Matsushita Electronics Corporation, Japan*. A high reduction of the optical feedback-induced intensity noise of semiconductor lasers has been successfully achieved by a combination of optoelectronic negative feedback and high-frequency superimposition, this being useful for optical disk systems.

WM14 Low-Frequency Fluctuations and Chaos in a Distributed Feedback Semiconductor Laser with Optical Feedback. J. Mork, *Technical U. Denmark*; K. Kikuchi, *U. Tokyo, Japan*. An experimental investigation of the route to chaos in a distributed feedback semiconductor laser with optical feedback is reported. A chaotic state may be reached through intermittent switching between high and low frequency fluctuations.

Nonlinear Optics, Phase Conjugation, and Spectroscopy

WM15 Measurement of Raman Gain Coefficients of Hydrogen, Deuterium, and Methane. John J. Ottusch, David A. Rockwell, *Hughes Research Laboratories*. Using a single Nd:YAG laser to pump a Raman oscillator and amplifier, we measured the steady-state gain coefficients of H_2 , D_2 , and CH_4 at 532 nm. The oscillator spectrum and the effects of oscillator amplifier pressure mismatch were also investigated.

WM16 Phase Conjugation in Liquid CS_2 Using a CO_2 Laser. P. E. Dyer, J. S. Leggett, *U. Hull, U.K.* Degenerate four-wave mixing in liquid CS_2 using a TEA CO_2 laser has resulted in a phase-conjugate reflectivity of 1%. Dramatic pulse reshaping and lengthening is observed and a detailed mathematical model proposed.

WM17 Nonlinear Optical Ranging Imager. Ian McMichael, Monte Khoshnevisan, Paul H. Beckwith, *Rockwell International Science Center*. A new method that can be used to image 3-D objects in two dimensions using nonlinear optical two-wave mixing techniques is described and demonstrated. Information about the third dimension of depth is represented as an intensity modulation in the image.

WM18 Laser Beam Combining Using Near-Resonance Nonlinear Dispersion. C. A. Giosson, C. D. Cantrell, *U. Texas at Dallas*; Jay S. Chivian, W. D. Cotten, *LTV Missiles & Electronics Group*. Near-resonance nonlinear dispersion is used to create a periodically modulated index of refraction in a collection of three-level systems, acting as a grating for beam addition.

WM19 Coherent Beam Coupling and Pulsations in Self-Pumped $BaTiO_3$. Putcha Venkateswarlu, P. Chandra Sekhar, H. Jagannath, M. C. George, M. Moghbel, *Alabama A&M U.* Beam couplings and coherent pulsations in $BaTiO_3$ using two coherent beams from Ar and He-Ne lasers are studied in three configurations. Relative strengths of self-pumped and cross-coupled Bragg reflected beams are obtained.

WM20 Four-Wave Mixing at Optical Frequencies in Collisional Plasmas. L. Zhang, *UC-Davis*; E. J. Beiting, *Aerospace Corporation*. Four-wave mixing in a collisional plasma was studied theoretically for plane waves input at two frequencies. Expressions for the intensities at the six sum and difference frequencies were obtained in terms of the electron density and plasma temperature.

WM21 Ab initio Theory of Stimulated Rotational Raman Scattering for Diatomic Molecules and Numerical Simulation. C. G. Parazzoli, D. M. Capps, *Hughes Aircraft Company*. The time-dependent semiclassical theory of stimulated rotational Raman scattering is presented. It includes multirotational lines, Stokes, anti-Stokes, multiphoton processes, pump-population depletion, spontaneous emission. Results from a numerical code with diffraction are reported.

WM22 Numerical Studies of Transient Raman Amplification. Godehard Hilfer, SAIC; Curtis R. Menyuk, *U. Maryland*; John Reintjes, *U.S. Naval Research Laboratory*. The (2 + 1)-dimensional Raman amplifier code RAM2D1 was used to study numerically the effects of the Raman interaction observed in the Naval Research Laboratory experiments. Both transient and diffractive effects are included in the code.

WM23 Transient Pulse Evolution in the Long Distance Limit of Stimulated Raman Scattering. Curtis R. Menyuk, *U. Maryland*; Godehard Hilfer, SAIC. We consider the evolution of transient SRS pulses over lengths sufficiently long to substantially deplete the pump. We show that both the pump and the material excitation develop rapid oscillations which can be described analytically.

WM24 Establishment of Phase in Stimulated Brillouin Scattering Beam Combiners. Joel Falk, Morton Kanefsky, Ronald Mehninger, Paul Suni, *U. Pittsburgh*. The phase difference between two stimulated Brillouin scattered beams generated in a single material must be described as a random process whose probability distribution depends on the overlap between the two pump beams, the Stokes pulse width, and the phonon lifetime.

WM25 Experimental Studies on the Second Harmonic Generation of Broadband High-Peak-Power Laser Radiation at 527 nm Using a Quadrature Crystal Array. M. S. Pronko, S. P. Obenshain, R. H. Lehmberg, *U.S. Naval Research Laboratory*. We present experimental results on the production of broadband laser radiation at a wavelength of 527 nm. A two-crystal quadrature configuration is shown to have higher second harmonic conversion efficiency than conventional single crystal systems.

WEDNESDAY, APRIL 27, 1988

"Formation of soliton-like structures in stimulated Raman scattering," (C.R. Menyuk), submitted to the International Quantum Electronics Conference '88 (Tokyo, Japan, July 18-22, 1988).

**Formation of Soliton-like Structures
in Stimulated Raman Scattering**

Curtis R. Menyuk

Department of Electrical Engineering

University of Maryland

Baltimore, MD 21228

tel. no. (301)-455-3488

ABSTRACT: Conditions for the formation of soliton-like pulses in stimulated Raman scattering are derived. Use of the spectral transform method to study arbitrary initial pump shapes is described with a concrete example.

Formation of Soliton-like Structures in Stimulated Raman Scattering

Curtis R. Menyuk

Department of Electrical Engineering

University of Maryland

Baltimore, MD 21228

tel. no. (301)-455-3488

The equations which describe transient stimulated Raman scattering are¹

$$\begin{aligned}\frac{\partial E_L}{\partial z} &= -i \frac{k_L}{k_S} \kappa_2 Q E_S, \\ \frac{\partial E_S}{\partial z} &= -i \kappa_2 Q^* E_L, \\ \frac{\partial Q}{\partial t} + \Gamma Q &= -i \kappa_1 E_S^* E_L.\end{aligned}\tag{1}$$

Here, E_L and E_S are the complex amplitude envelopes of the pump and Stokes waves, k_L and k_S are the corresponding wavenumbers, Q is the material excitation, κ_1 and κ_2 are gain coefficients, and $\Gamma = T_2^{-1}$ is the material damping rate. While these equations have been extensively examined in the past, their evolution depends strongly on the initial conditions, and there remains much that is of substantial interest to be examined.

If the initial pump pulse is larger in width than T_2 , it is possible for solitons to form when the phase of the Stokes pulse undergoes a rapid phase flip.²⁻⁴ How rapid must this flip be? Here, we address this question.

We first note that if $T_w \gg T_2$, where T_w is the width of initial pump wave, then Eq. (1) reduces to

$$\begin{aligned}\frac{\partial E_L}{\partial z} &= -\frac{k_L}{k_S} \frac{g}{2} |E_S|^2 E_L, \\ \frac{\partial E_S}{\partial z} &= \frac{g}{2} |E_L|^2 E_S,\end{aligned}\tag{2}$$

where $g = 2\kappa_1\kappa_2$. Equation (2) is easily solved. Letting $A_L = |E_L|$, $A_S = (k_L/k_S)^{1/2} |E_S|$, and noting that

$$K(z) = |E_L(z, t)|^2 + \frac{k_L}{k_S} |E_S(z, t)|^2\tag{3}$$

is constant in z , we find

$$\ln \frac{K^2 - A_S^2}{A_S^2} = \ln \frac{A_L^2}{K^2 - A_L^2} = C - 2gK^2 z\tag{4}$$

at each point in time, where C is a constant of integration. We now suppose that E_S has the form

$$E_S = K\Gamma_S t + iK_S \quad (5)$$

in the neighborhood of $t = 0$, where $K(t) = K$ is constant. We assume that $K_S \ll K$ but is non-zero, resulting in a small deviation from an exact π -phase shift for E_S . In the neighborhood of $t = 0$, we then find

$$C = \ln \left[\frac{K^2(1 - \Gamma_S^2 t^2) - K_S^2}{K^2 \Gamma_S^2 t^2 + K_S^2} \right] \simeq \ln \left[\frac{K^2}{K^2 \Gamma_S^2 t^2 + K_S^2} \right], \quad (6)$$

so that

$$\frac{A_L^2}{K^2 - A_L^2} \frac{K^2 \Gamma_S^2 t^2 + K_S^2}{K^2} = \exp(-2gK^2 z) \equiv F(z). \quad (7)$$

We conclude

$$[A_L(t = 0)]^2 = \frac{FK^2}{F + K_S^2/K^2}. \quad (8)$$

Defining τ as the t -value at which $[A_L(t)]^2 = \frac{1}{2}[A_L(0)]^2$, we find that

$$\tau = \frac{1}{\Gamma_S} \left[\frac{1}{2} \left(\frac{K_S^2}{K^2} + F \right) \right]^{1/2}. \quad (9)$$

For a soliton-like structure to form, it must be the case that $\tau < T_2$ at some z -value. Noting that $F \rightarrow 0$ as $z \rightarrow \infty$, we conclude that this will occur if

$$\frac{\Gamma}{\Gamma_S} \frac{K_S}{K} < 1. \quad (10)$$

In experiments where a Pockels cell was used to impose a phase reversal, soliton-like structures were only observed intermittently.⁴ Equation (10) and a careful reading of the experimental papers suggests that the ratio Γ/Γ_S may have been too small to lead to a reasonable expectation of satisfying Eq. (10).

We now turn to consideration of the case where the pulse size is small compared to T_2 . This limit is of substantial interest because in recent experiments at the Naval Research Laboratory, pulse sizes of about 40 picoseconds and T_2 -values of about 600 picoseconds are typical.^{5,6} In this limit, Γ may be set equal to 0 in Eq. (1). The equations are then integrable using spectral transform methods.^{7,8} However, the usual method of solution must be substantially modified, leading to substantial modifications in the behavior of the solutions.

In carrying out this theory, it is useful to first normalize our variables so that

$$\begin{aligned}\frac{\partial A_1}{\partial \chi} &= -X A_2, \\ \frac{\partial A_2}{\partial \chi} &= X^* A_1, \\ \frac{\partial X}{\partial \tau} &= A_1 A_2^*,\end{aligned}\tag{11}$$

where A_1 and A_2 are the normalized pump and Stokes amplitudes, X is the normalized material excitation, and χ and τ are normalized distance and time. We now define two new quantities, u_1 and u_2 which satisfy the equations

$$\begin{aligned}\frac{\partial u_1}{\partial \chi} - i\lambda u_1 &= X u_2, \\ \frac{\partial u_2}{\partial \chi} + i\lambda u_2 &= -X^* u_1,\end{aligned}\tag{12}$$

and

$$\begin{aligned}\frac{\partial u_1}{\partial \tau} &= -\frac{i}{\lambda} S_3 u_1 + \frac{1}{\lambda} S_+ u_2, \\ \frac{\partial u_2}{\partial \tau} &= \frac{i}{\lambda} S_3 u_2 - \frac{1}{\lambda} S_- u_1,\end{aligned}\tag{13}$$

where

$$\begin{aligned}S_3 &= \frac{1}{4}(A_1^* A_1 - A_2^* A_2), \\ S_+ &= \frac{i}{2} A_2^* A_1, \\ S_- &= S_+^*.\end{aligned}\tag{14}$$

Equations (12) and (13) are only *compatible*, i.e., their cross-derivatives are equal, only if Eq. (11) holds.

In the usual spectral transform approach, as applied for instance to the nonlinear Schrödinger equation, one would proceed by defining scattering data which relate (u_1, u_2) at $\tau = +\infty$ to (u_1, u_2) at $\tau = -\infty$. This scattering data has a one-to-one correspondence with the original variable set. Moreover, since u_1 and u_2 evolve simply in χ at $\tau = \pm\infty$, so do the scattering data, and one can then infer the evolution of the original variable set.⁹ In our case, a fundamental difficulty results from the fact that X does not in general tend toward zero as τ tends toward $+\infty$, and the χ -evolution at $+\infty$ is not simple. Kaup⁷ has resolved this issue in certain important cases by showing that our variable set only depends on the evolution of u_1 and u_2 at $\tau = -\infty$.

We apply his approach in detail to cases of practical interest to determine the full nonlinear evolution, notably the case where the initial pump and the initial Stokes have the same shape. We also show that in contrast to the usual case, where the initial pulse decomposes into a set of

enduring solitons and a dispersive continuum, all soliton-like structures must be transient. From a physical standpoint, this result is almost self-evident. These soliton-like structures are well-known to possess a velocity slower than light.² They must therefore ultimately disappear at the back end of the pulse.

This work was supported by the Naval Research Laboratory, and the author is grateful to Dr. J. Reintjes for numerous, useful discussions.

REFERENCES

1. R. L. Carmen, F. Shimuzu, C. S. Wang, and N. Bloembergen, *Phys. Rev. A* **2**, 60 (1970).
2. K. Druhl, R. G. Wenzel, and J. L. Carlsten, *Phys. Rev. Lett.* **51**, 1171 (1983).
3. K. J. Druhl, J. L. Carlsten, and R. G. Wenzel, *J. Stat. Phys.* **39**, 615 (1985).
4. R. G. Wenzel, J. L. Carlsten, and K. J. Druhl, *J. Stat. Phys.* **39**, 621 (1985).
5. M. D. Duncan, R. Mahon, L. L. Tankersley, and J. Reintjes, "A Study of Transient Stimulated Raman Scattering in Hydrogen," (preprint).
6. J. Reintjes, M. D. Duncan, G. Calame, L. L. Tankersley, and R. Mahon, *Optics News* **39**, 101 (1987).
7. D. Kaup, *Physica* **6D**, 143 (1983).
8. H. Steudel, *Physica* **6D**, 155 (1983).
9. See, *e.g.*, M. J. Ablowitz and H. Segur, *Solitons and the Inverse Scattering Transform*, (SIAM, Philadelphia, 1981).

END

DATE

9-88

DTIC