

NAGW-1715

HQ. GRANT

IN-61-CR

©OVERRIDE

32342

0119

# M.S.L.A.P. Modular Spectral Line Analysis Program Documentation

*Charles L. Joseph*  
Princeton University Observatory

Copyright (c) 1991

(NASA-CR-188713) M.S.L.A.P. MODULAR  
SPECTRAL LINE ANALYSIS PROGRAM DOCUMENTATION  
Final Report (Princeton Univ.) 119 p

CSCS 098

N91-30738

Unclass

63/61 0032342

# Modular Spectral Line Analysis Program (MSLAP) Documentation

*by Charles L. Joseph*

Copyright (C) 1991 by Charles L. Joseph (Revision 1991.2)

Permission is granted to copy and distribute this document, provided this copyright notice is retained unaltered.

MSLAP is copyrighted software. The astronomical version of MSLAP is distributed free of charge only to sites that are currently engaged in astronomical research for the sole purpose of conducting astronomical research. Possessing MSLAP in whole or part indicates that the user has accepted all of the terms of the licensing agreement, which serves to insure the integrity of the software. Basically, the terms of the license agreement are:

- 1) You are not permitted to distribute the standard source code to any other site. There are several distribution centers that supply standard source code.
- 2) You are permitted to distribute additional modules (subroutines) written for use with MSLAP to other sites, provided the new code meets documentation requirements.
- 3) You are permitted to alter the standard source code as you desire, but you must maintain the original copyright notice in the source file and you may not distribute the altered code without written authorization. Also, any alterations to the standard source code must meet documentation requirements.
- 4) If you represent a guest user facility, you are not permitted to incorporate modifications that remove the modularity of MSLAP, thus making it difficult for guest users to customize MSLAP.

---

The Modular Spectral Line Analysis Program (MSLAP) was written and copyrighted  
by Charles L. Joseph and Edward B. Jenkins

Partial Support for the development of Version 1.0 (the Astronomical Version) of  
MSLAP was provided by NASA grant NAGW-1915 to Princeton University

AND

by NASA grants NAS-5248 to Princeton University and NAS-5300 to the  
University of Colorado.

---

## Table of Contents

	Abbreviated Licensing Agreement .....	i
I.	Introduction to MSLAP .....	1
II.	Quick Start .....	1
	a. Getting Ready .....	1
	b. Running MSLAP .....	2
III.	Step-by-step Description .....	4
	1: Measure Profile Moments Only .....	5
	2: Measure Moments and Optical Depths .....	14
	3: Manipulate Tabled Data .....	15
	4: Edit Tabled Data .....	19
	5: Analyze Column Densities vs Velocity .....	20
	6: Compare Data to Curve of Growth .....	21
IV.	Comments on Correct Statistical Techniques .....	21
V.	Customizing MSLAP .....	23
	a. The mslap.pro File .....	23
	b. The dgets.pro File .....	24
	c. Look Up Tables .....	25
VI.	Installing MSLAP (for the system manager) .....	25
VII.	Trouble Shooting .....	17
VIII.	Appendix A - Listing of the Source Code .....	28
IX.	Appendix B - Data Structures in MSLAP .....	107
X.	Licensing Agreement .....	112

# I. Introduction to MSLAP

The Modular Spectral Line Analysis Program (MSLAP), as its name implies, forms a backbone of programs, designed so that customized subroutines can be inserted and implemented with minimal difficulty. MSLAP, a third generation package of software, also is a complete and powerful stand-alone program for analyzing spectra, providing the basic structure to identify spectral features, to make quantitative measurements of these features, and to store the measurements for convenient access. MSLAP can be used to measure not only the zeroth moment (equivalent width) of a profile, but also the first and second moments. Optical depths and the corresponding column densities across the profile can be measured as well for sufficiently high resolution data.

The software was developed for an interactive, graphical analysis where the computer carries most of the computational and data organizational burden and the investigator is responsible for all judgment decisions. Cursors are used not only to provide graphical input, but also for logical control branching. It employs sophisticated statistical techniques for determining the best polynomial fit to the continuum and for calculating the uncertainties. MSLAP, making use of data structures, provides substantially more capabilities in the handling, presentation, editing, and manipulation of intermediate results than do its predecessors.

MSLAP is written in the Interactive Data Language (IDL) and issues some commands to the UNIX operating system. While MSLAP has not been transported to a VMS environment, there is no apparent reason why it could not be. MSLAP is specifically designed for workstations, running in either an X Window or Sunview environment.

The basic structure of this document is as follows. A method for getting started quickly is given in §II, where it is assumed that MSLAP has already been installed in some convenient library directory and the reader is anxious to try running MSLAP for the first time. Section III provides a detailed, step-by-step description of the six major portions of the program. Some common mistakes in data handling are listed as cautions in §IV. The documentation for modifying and customizing MSLAP is given in §V and for installing MSLAP on system is in §VI. Finally, for convenient access, Appendix A contains a listing of the source code, including a table of contents, while the three data structures used by MSLAP provided in Appendix B.

## II. Quick Start

It has been said that a fundamental difference between men and women is that men do not like to read directions, preferring to get started quickly, while women do read the directions, opting to do the job only once. Whatever your persuasion, this section is designed to get the novice going with the minimal amount of reading. Once program has been started, the novice can rely on the instructions provided by MSLAP as it runs or rely on §III, which provides a more detailed description.

### a. Getting Ready

MSLAP runs in either an X Window or Sunview window environment. Sunview is perhaps the more sophisticated system, but is specific only to Sun Workstations, while X Windows is an industry standard that is available on most machines. X Windows has the added advantage that a user can run software on a remote machine, displaying the graphics and printed matter on the local console. It is not possible, however, to run MSLAP on a remote machine without

a local workstation or terminal that supports windows.

Both Sunview and X Windows can be configured in many different ways. If you are at a Guest Facility, the chances are that your machine automatically starts one of these systems when you log onto the workstation. MSLAP needs a large window in the lower left corner for printing instructions. If you do not have a window in the lower left corner and are not an experienced user of one of these environments, it is recommended that you copy the appropriate files from the MSLAP library to your login directory. Your system manager has placed the MSLAP library in the directory: \_\_\_\_\_ . For X Windows, the files are .x11defaults, .twmrc, and .xinitrc. For Sunview, only .sunview is needed. If necessary, make sure that your path includes: \_\_\_\_\_ . (Your path is found either in your .login or .cshrc file and can be changed using your favorite editor.) Then type: "source .login" to reset your path name. Start the window system by typing either:

```
"sunview"
```

or

```
"xinit ; kbd_mode -a ; clear"
```

for Sunview or X windows, respectively. The latter could be aliased to something simpler or either could be placed in your .login file. Now each window that appears, effectively represents a separate terminal. Each time the cursor, associated with the mouse, is placed into a window, that window becomes the "active terminal". (In some cases, it is also necessary to click one of the mouse buttons to activate the new window.) All other windows, however, can still continue processes that were previously started, including output printing.

Place the mouse cursor into the large window in the lower left of the screen. Change to the directory where you want to run MSLAP and copy from the MSLAP library the files: mslap.pro and dgets.pro You are now ready to proceed to run MSLAP.

## b. Running MSLAP

MSLAP is written in the Interactive Data Language (IDL). Use the mouse to place the cursor in the large window in the lower left corner of the screen and start IDL by typing: "idl". Once you see the prompt "IDL>", type: ".run mslap.pro". A menu of options will then appear. The first two options allow the researcher to scan quickly through the spectra, searching for features, identifying the species in look up tables, making measurements of the profiles, and storing all of this information in a data base for latter use. The second option also produces optical depths as a function of velocity across the profiles.

In options 1: and 2: of the FIRST MENU, a series of questions appear to configure MSLAP. The default values for all Yes/No questions is assumed to be NO except for answers of either a "Y" or a "y".

The main body of MSLAP begins after these questions are answered. In this portion of the software, the mouse is used to input graphical information to the program as well as to control the data reduction. The basic philosophy of the mouse operation is as follows. The left button should be used in response to the general flow of the program while the middle button is used to signal the computer that the researcher wishes to move on to the next task. For

example when the main spectrum is showing, the left button causes the computer to attempt to identify features of interest (the first task MSLAP expects to perform) and the middle button causes the computer to go fetch the next portion of the spectrum. Another example would be in the continuum-fitting portion of MSLAP. There, the left mouse is used to select the portions of the spectrum to be fit, while the middle mouse button indicates all portions are selected - now go on and do the fits.

Finally, all other branching is accomplished by using the right mouse button, which causes a Menu of additional options to appear. The researcher is invited to examine this Menu from time to time. There is no harm in striking the right mouse button since "Take No Action" is always one of the options. Note: once the Menu appears, the left mouse is used to make the selection. This Menu can be modified easily and customized routines inserted into it by editing the mslap.pro file that was transferred to your working directory. (See §V for details on modifying this file.) The researcher is encouraged to examine this file once he or she has become familiar with the operation of MSLAP.

At this point put down this manual and concentrate on running MSLAP. The program provides self-contained directions that appear in the lower left window. However, if you run into trouble, §III of this manual has a detailed, step-by-step set of instructions.

### III. Step-by-step Description

This section is organized into 6 subheading that represent each of the 6 options from the first menu that MSLAP displays when started. This menu appear as follows:

May 17, 1990

```

;
;
; *****
;           Modular Spectral Line Analysis Program
;           M.S.L.A.P.  version 1.0
;
; copyright (c) 1991  by Charles L. Joseph and Edward B. Jenkins
;   All rights reserved.  A license may be obtained from the
;   first author or from an authorized distribution center.
;
;   This software distributed through: Princeton University
; *****
;
;
;           THE FOLLOWING OPTIONS ARE AVAILABLE:
;
;
;   1:   Measure Profile Moments Only
;         (Equivalent Width, Profile Centroid, etc.)
;
;   2:   Measure Moments and Optical Depths
;
;   3:   Manipulate Tabled Data
;
;   4:   Edit Tabled Data
;
;   5:   Analyze Column Densities vs Velocity
;         on Option #2 Data
;
;   6:   Compare Data to Curve of Growth
;
;   10:  TO EXIT FROM THIS PROGRAM

```

Which option would you like ?

## 1: Measure Profile Moments Only

Option 1: of the first menu is for measuring the zeroth (equivalent width), first (velocity), and second moments of the spectral profiles. Both this and the second option ask the same set of questions, which will be discussed below. The computer I/O will be displayed in small style type like this, while the descriptive narration describing these questions will be displayed in ordinary style type. Answers to the questions will be provided as an example.

The first two questions deal with establishing an output data file. In the case below, the output file name is called "test" and a file called "test.DTL" will be created or opened, depending whether the file already exists. If a file named test.DTL is found, the program examines the contents to see how many measurements have been made previously and reports this information to the user. The investigator then has the option of starting with a number of measurements already made or initializing (erasing) the contents of the file. Note: if you wish to save the previous work, but not combine new data to the file, answer the "Append new data to the old?" question with a "y" and exit the program later. For the purposes of this example, a "n" was used.

What is the output file name ?test

File Already Exists

4 measurements have already been made.

Append new data to the old?

(Note: a NO will erase old data.)n

Next the program seeks to determine the radial velocity expected for the profiles. This information is used only to help select the correct entries in the Look Up Tables in order to identify the various spectral features. The information can be input in the form of Delta-Lambda/Lambda or as a Radial Velocity in km/s. If you are uncertain of the real value, enter a "0". The computer then mirrors the value that has been entered.

The radial velocity is used only to help identify species.

Enter 0 if you are uncertain of the real value.

Enter Delta-Lambda/Lambda or Radial Velocity for the source: 0

The Delta-Lambda/Lambda is: 0.00000

There are several Look Up Tables which contain the identification of various species along with important information such as the oscillator strength and rest wavelength. (See §V for details of the use and implementation of these tables.) MSLAP has the capability to search more than one of these tables automatically. In addition, a personal User Look Up Table, which is capable of holding up to 100 entries, is scanned.

Below is a list of Look Up Table Options. In the example, the investigator has requested both the Morton and Smith table of interstellar lines as well as a compilation of molecular bands be searched each time an identification is requested.



-----  
 Available lookup tables are:

- 1) Interstellar Lines (Morton and Smith 1973 plus updates)
  - 2) Molecular Hydrogen, HD, and CO
  - 3) Options 1 & 2 Combined
  - 4) Hot-Star Lines (not implemented)
- 

Which one would you like ? 3

Next the programs queries as to which dget (i.e. data-getting) routine is to be used and the name of the file holding the spectra. There are a number of data formats used by different observatories so it is appropriate for many users to keep several dget routines on hand. While only 5 dget routines are supported at a time, each researcher gets to incorporate the 5 that best meets his needs. (See §V for more details on the implementation of the dgets.pro file.) In the example, the investigator has requested a very specialized dget routine which reads data taken with an echelle spectrograph on a sounding rocket. This particular dget differs from the dget loaded into slot 3) because it contains information about the background level and associated errors. The file to be used is in another subdirectory: ../rocket and is named: imaps.

-----  
 DATAGET Options:

- 1) IUE Standard GO Files for High-Res. Spectra (DISKGET)
  - 2) NOT Being Used
  - 3) ASCII Format of Wavelength-Spectra-Quality-BG
  - 4) 1024-Element Stand-Alone Data
  - 5) IMAPS pseudo standard (ASCII)
- 

Which one would you like ? 5

Enter complete INPUT Data Filename  
 Including the path if necessary ../rocket/imaps

The final question before the researcher is off and running is the issue of coherence length. The message is pretty much self explanatory, but its importance should not be slighted (see §IV). Note that the coherence length need not be an integer value. For example, if the value of a pixel is influenced only slightly by its adjacent neighbors, the coherence length could be 1.2 for instance. A non-integer value might also be expected if the data have been smoothed by a running-gaussian smoothing routine.

**IMPORTANT:** The coherence length is used to calculate ALL uncertainties. It is the number of pixels influencing the value in a given pixel. Thus, the coherence length must be 1 or greater. For example, data smoothed by a 3-point Running Box Car has a coherence length of 3.

What is the coherence length of the present data? 3

The main program then begins and the screen resembles Figure 1. Generally speaking, instructions appear in the text (lower left) window, indicating how the graphical input is to be made in the plot window. As this portion of the software starts, the following information is displayed in the text window. Particularly important pieces of information are always boxed.

New Data Have Been Read -- Data Getting Option: 5

```

-----
| The background uncertainty frequently has a major impact on the |
| uncertainty of the various measurements.                         |
| Current background is:      12.0000 with an error:      5.00000 |
| Use the RIGHT Mouse Button if these are unsatisfactory.        |
| Note: all graphical input is performed by placing the cross-hairs |
|       at the point of interest and pressing a mouse button.     |
-----

```

Left Mouse to Locate Feature to be Measured -- Need NOT be centered  
 Middle Mouse to GO ON, get new data  
 Right Mouse to bring up UserMenu of other options.

Notice that the last 3 lines contain the instructions for the use of the Mouse. These instructions are constantly being updated as you proceed through the program. Once you become familiar with MSLAP, your focus should move almost exclusively to the plot in the upper right window.

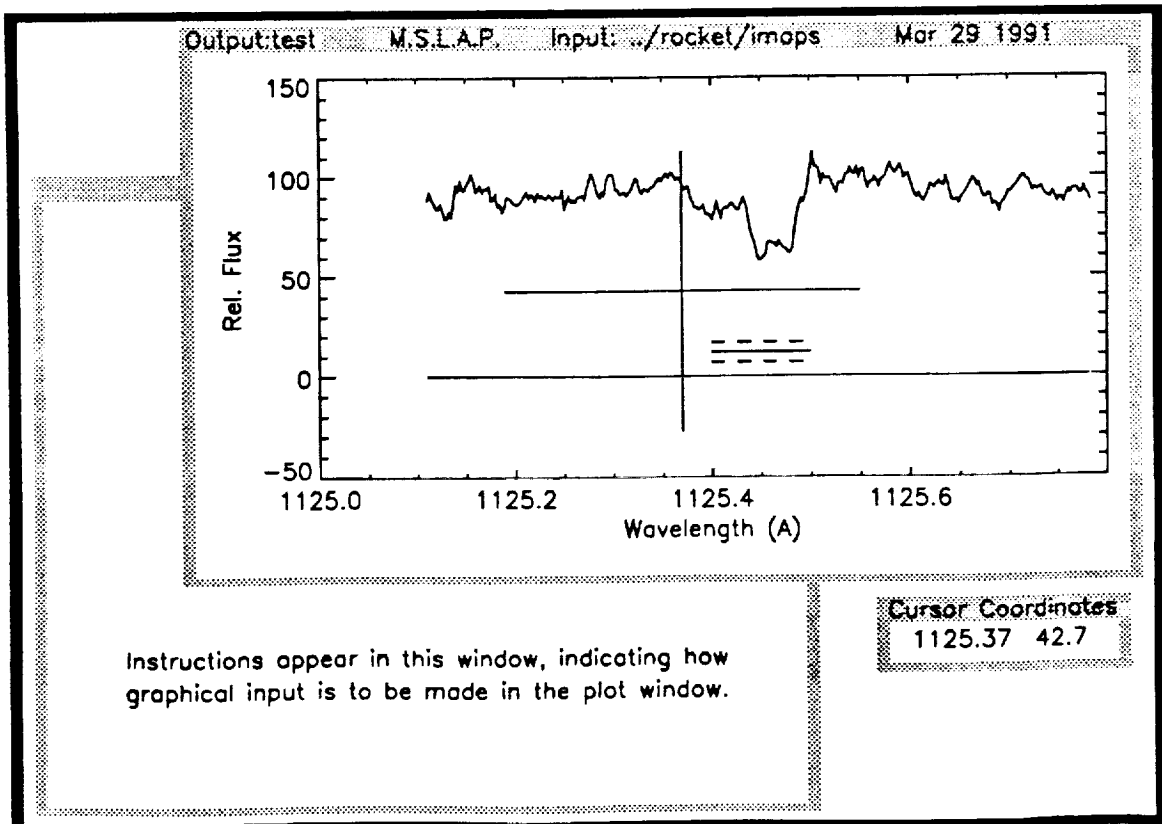


Fig. 1 showing the typical screen layout.

The plot in figure 1 has several features worth highlighting. In addition to the spectrum, there

is a data quality vector plotted. Not all data sets have such a vector, in which case it is set to zero as shown in figure 1. The three horizontal lines plotted between about 1125.4 and 1125.5 Å indicate the assumed background level and its uncertainty. This information is also printed in the text window, surrounded by a box, every time a new data is read. The graphical cross hairs or cursors are also depicted and as the Mouse is moved across the pad, the cursor moves accordingly. A small window, labeled Cursor Coordinates, provides continuous readout of the cursor location.

At this point, the branching possibilities become large and it is nearly impossible to provide a detailed description of each path on a step-by-step basis. The general philosophy, however, is to scan through the spectra, looking for spectral features of interest. The Center Mouse Button is reserved to indicate that the user is finished with the present task. Since the present task is to locate features to be measured, the Center Button indicates a request for new spectra. If your dget is configured for multiple portions of the spectrum in a single data file, striking the Center Button repeatedly displays pieces of the spectrum sequentially.

The Left Mouse Button is used to identify species by comparing the observed wavelength obtained from the cursor location at the time the button was struck to a set of laboratory wavelengths located in various Look Up Tables. Immediately after the Left Button is depressed, a small menu of options is provided as shown in Figure 2. The three closest matches from the requested Look Up Table(s) are always provided plus any close matches from the researcher's personal User Look Up table. There were no entries found from the User's Table in the example shown in Figure 2. The investigator must make a selection by moving the Mouse up or down and striking the Left Button. The current position is noted with an arrow and the potential selection is highlighted. Notice that "NONE - Return to Spectra" and "Input Identity" are also options. If the latter is chosen, the program will present a number of questions to obtain information, including species name, its rest wavelength, and its oscillator strength. This complete species identification can be used as a temporary set of variables or can be permanently stored in the researcher's personal table for subsequent access.

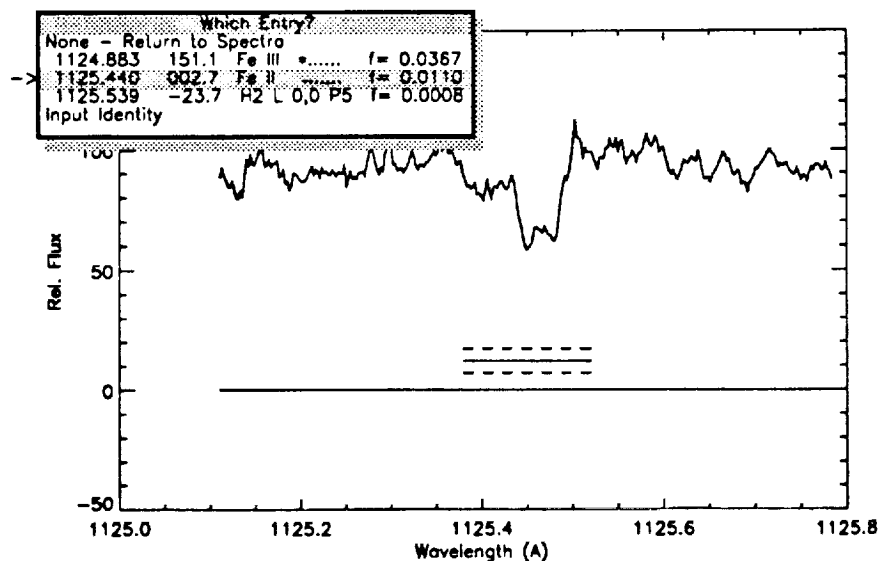


Fig. 2 showing the entries from the Look Up Tables

The Right Mouse Button provides all of the remaining flexibility. Striking it brings up the UserMenu of options shown in Figure 3. The novice is strongly encouraged to press this button, if for no other reason than to see the selection. This is a relatively safe operation since one of the options is to "Take No Action", just in case the Mouse Button was used inadvertently.

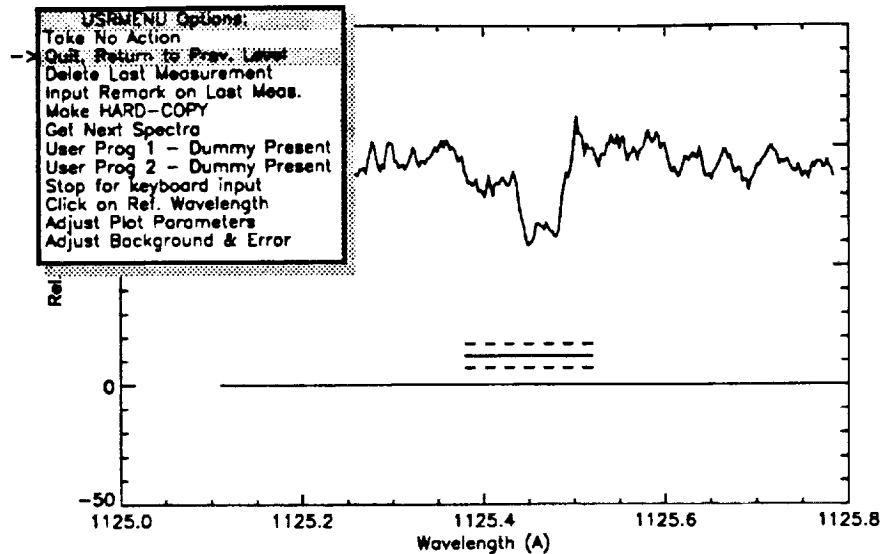


Fig. 3 showing the entries from the UserMenu

The UserMenu shown in Figure 3 works like all menus appearing above the plot window. A selection is made by using the mouse to place the arrow over the desired entry and depressing the Left Mouse Button. While the functionality of most of the entries are straight forward, the "Click on Ref. Wavelength" deserves a few comments. If this option is invoked, the user is solicited to move the cursor to any wavelength which he wishes "to declare" to be the rest wavelength and to click the Left Mouse Button. Now the abscissa coordinate, which is continually printed, reads in terms of velocity until the next graphical input. This capability is particularly useful when searching for a weak (1-2 sigma) feature after its doppler velocity has been determined from a strong line.

Once a species has been identified using the Left Mouse Button and making a selection as in Figure 2, the program enters the continuum fitting subroutine. MSLAP provides an expanded plot with the individual data points highlighted. In this portion of the software, the Left Mouse Button is used to isolate regions of the spectrum that are believed to be featureless, regions that will be used to perform a polynomial fit. The user may specify 15 or less regions, which are marked on the plot with numbers from 1 to 2, 3 to 4, 5 to 6, and so forth (see Fig. 4). As before, the Right Mouse brings up various MENUs, while the Center Mouse Button signifies all regions of interest have been identified (go on to the next task). Menu options that are available in the continuum fitting routine include: 1) defining discrete continuum points, 2) identifying additional profiles for measurement and corresponding species identifications, 3) adjusting the markers of the profile centers, and 4) bringing up the previous UserMenu of options.

Once the featureless portions have been identified, the program calculates polynomials fits of order 1 through 7. Then, MSLAP, starting with a polynomial of order 1, sequentially tests the polynomials of increasing order, searching for the case where no statistically significant

improvement of the fit is realized by polynomials of higher orders. Specifically, the next higher and next, next higher orders are tested for polynomials up to order 5. The program uses F Distribution Tests with a 5% significance to make this choice.

Then, a menu of options appears over the plot as shown in Figure 4. This menu is presented with MSLAP's choice of polynomial being indicated. The user can over ride the automatic selection of the order number, 4 in this particular case. He can in fact over plot various continuum fits until he is satisfied. Simultaneously, the following instructions plus a reminder of MSLAP's choice appear in the text window:

The recommended Order is 4 (highlighted), but you may select another. To assist in other choices: X is the observational difference in the reduced chi squares divided by the reduced chi square, which if larger than the theoretical  $F(1,n)$  indicates that going to the next higher order polynomial is justified statistically. Y is similar to X except it is for comparison to  $F(2,n)$ , an order that is 2 higher. The F Distributions are at the 5% confidence level for: 180 points.

Select the Order of the Polynomial

- Other Polynomials may be examined before deciding.
- Selecting order 4 implies use that polynomial.

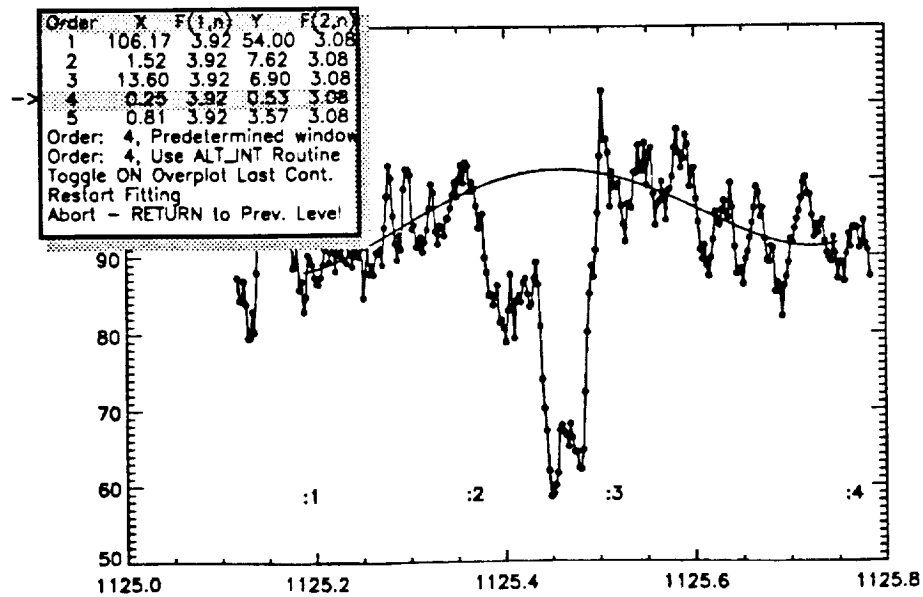


Fig. 4 showing the selection for the polynomials

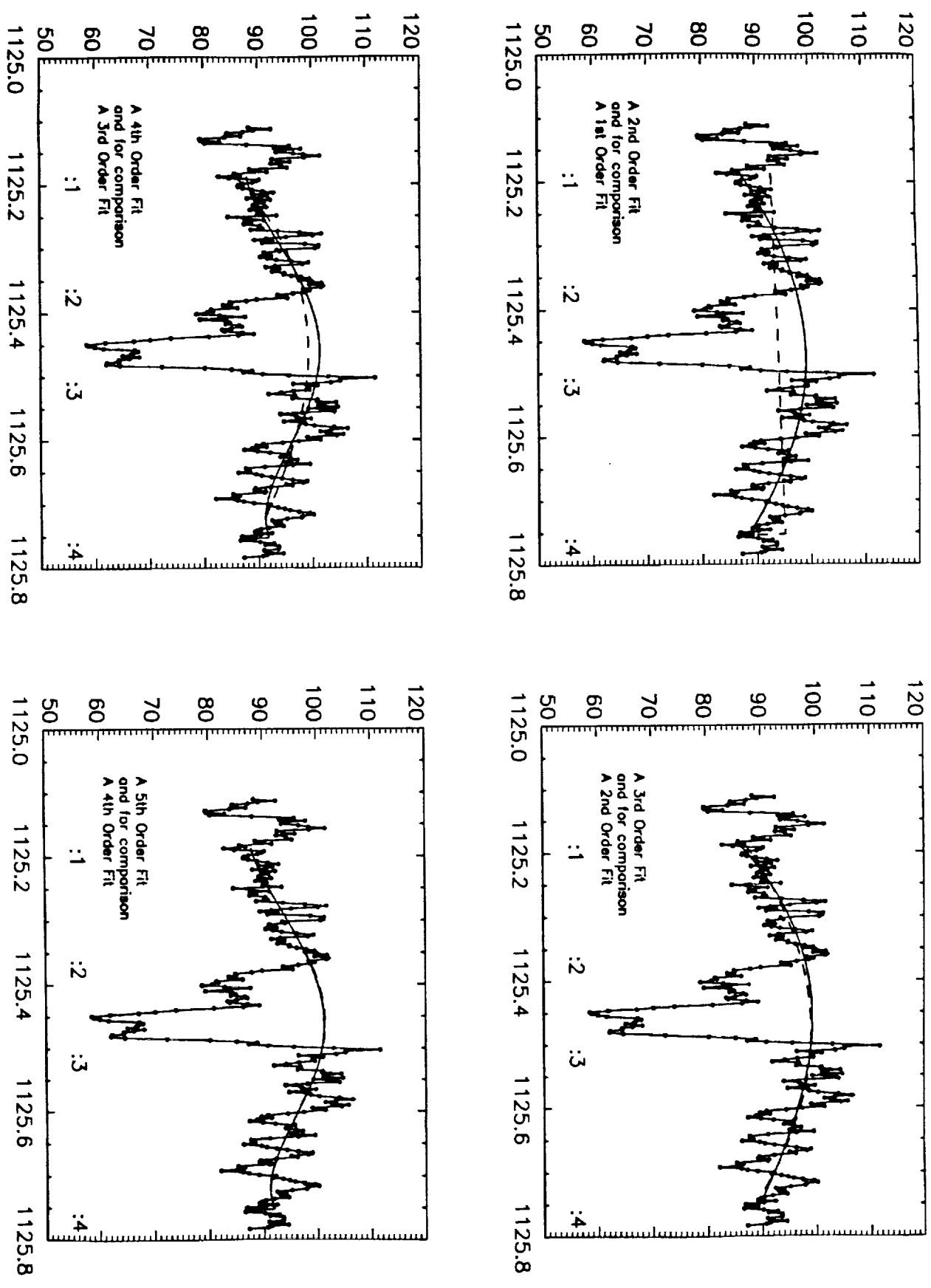
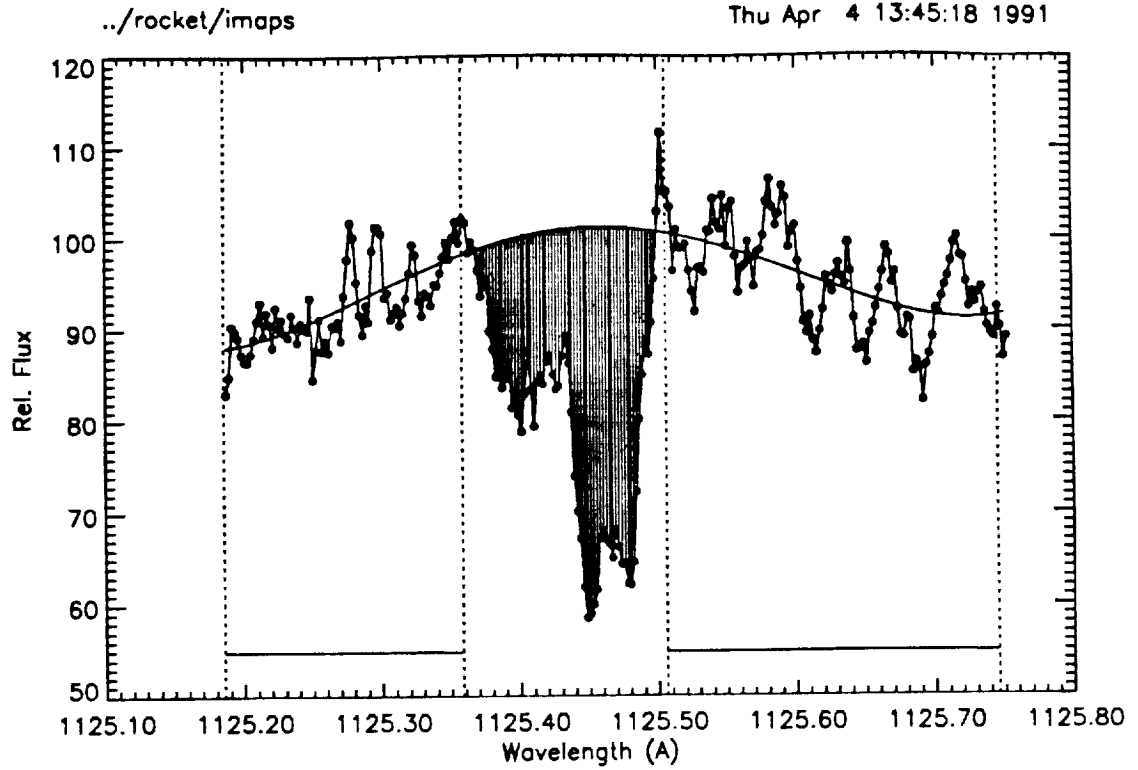


Figure 5

Figure 5 shows several polynomial fits. For comparison, a polynomial with an order number that is one less is also plotted as a dashed line. (These were made using one of the options shown in Figure 4.) In this example, there is a significant difference between a polynomial of order 1 and 2, but not much difference between orders 2 and 3. There is also significant changes between orders 3 and 4, but not between 4 and 5 or between 5 and 6 (not pictured).

A polynomial is finally determined by selecting the order number that is currently plotted. Notice that there are two entries that always indicate the order number currently plotted, but request either the use of the alternate integrating routine be used or a fixed sized integration window. The latter allows the user to specify some predetermined velocity interval over which the measurements are to be made. If this window is undefined, MSLAP will prompt the user for input.

Finally, the cursors and Left Mouse Button are used to set the end points for the integrating region. (See §IV for cautions on setting this range.) The program then calculates the various moments and produces a plot such as the ones shown in Figures 6 and 7. The latter depicts a case where two profiles were identified as CN from the User's customized table and were simultaneously measured. Up to 4 profiles can be measured at one time. The solid, horizontal lines near the bottom of the plot indicate those regions between the dotted vertical lines that were used in the continuum fit. The area of integration is shaded. It is possible, for example to integrate only part of the profile.

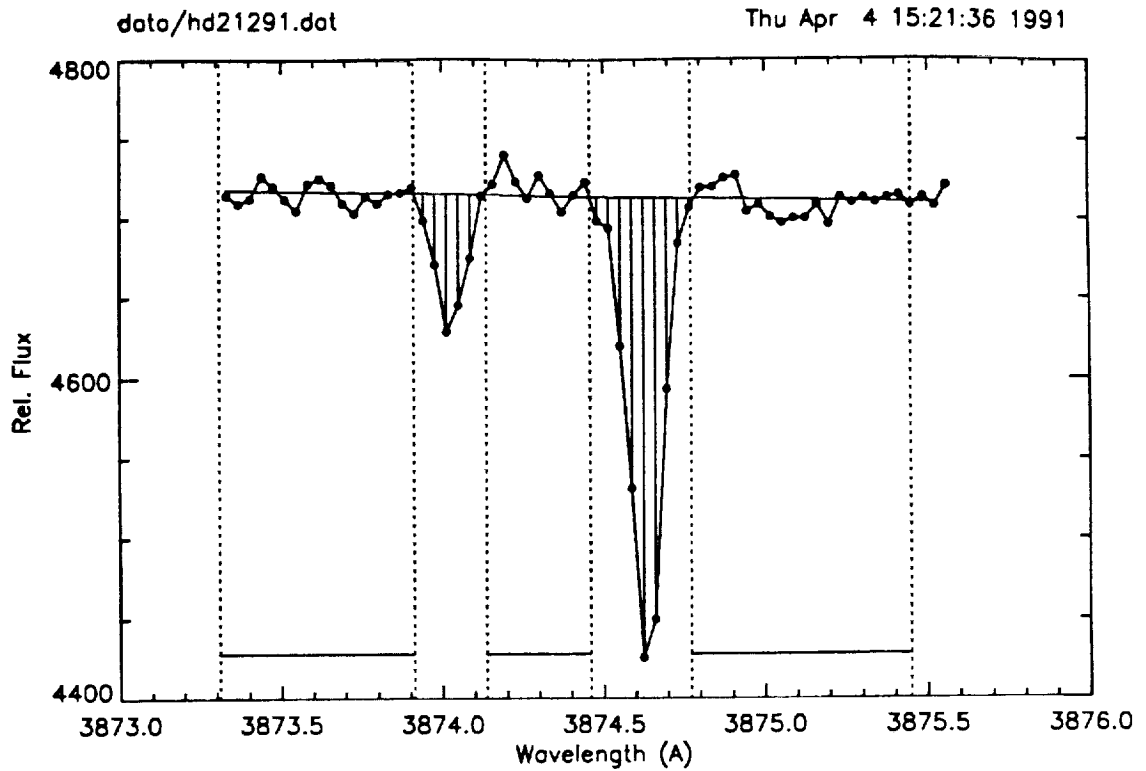


Measurements in file: test.DTL Real S/N = 13.3  
 Apparent S/N Ratio: 23.0 based on 185 points with a Noise Coherence Length of 3.0  
 Continuum was fit with a polynomial of order: 4 0.25 0.53  
 BG (background) was taken to be: 12.0 +/- 5.0  
 For Fe II ..... 1125.440 error contributions from BG: 0.0018 0.002 0.024  
 Errors below are the Addition in Quadrature of the Background and RMS-Noise Errors

Species	Lab. Wave.	f	Obs. Wave	EQW (Å)	1st (km/s)	2nd (km/s/s)
Fe II	1125.440	0.0110	1125.445	0.0323	1.245	72.094
			Errors:	0.0023	0.433	4.905

Fig. 6 showing the results for a single profile





Measurements in file: test.DTL Real S/N = 548.6  
 Apparent S/N Ratio: 548.6 based on 47 points with a Noise Coherence Length of 1.0  
 Continuum was fit with a polynomial of order: 1 1.42 2.11  
 BG (background) was taken to be: 0.0 +/- 0.0  
 For CN R(0) ..... 3874.608 error contributions from BG: 0.0000 0.000 0.000  
 Errors below are the Addition in Quadrature of the Background and RMS-Noise Errors

Species	Lab. Wave.	f	Obs. Wave	EQW (Å)	1st (km/s)	2nd (km/s/s)
CN R(0) .....	3874.608	0.0338	3874.628	0.0077	1.586	15.469
			Errors:	0.0002	0.550	16.462
CN R(1) .....	3874.000	0.0228	3874.025	0.0020	1.915	9.835
			Errors:	0.0002	1.174	22.954

Fig. 7 showing the results for multiple profiles

## 2: Measure Moments and Optical Depths

This option is identical to the previous one, except in that example it creates an additional file test.TAU as well as test.DTL. Also, it plots the optical depths immediately after it displays the results of the moments (i.e. Figures 6 or 7). The optical depths associated with Figure 6 are shown in Figure 8.

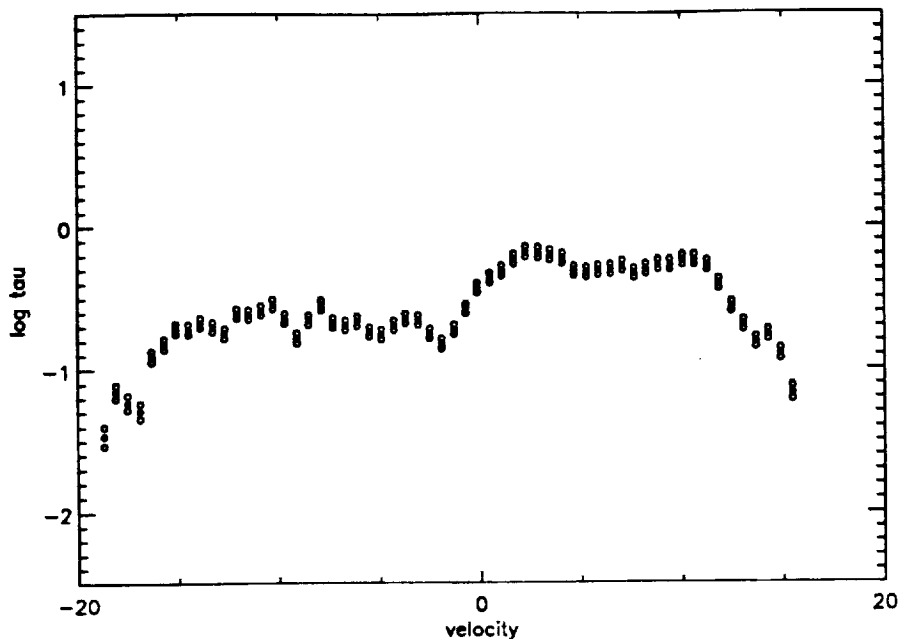


Fig. 8 showing the optical depths across the profile

The uncertainties, pictured in Figure 8 by open circles, only represent the systematic errors due to the uncertainties in the background and continuum placements. These data can be accessed later in Option 5: as profiles of the column density.

### 3: Manipulate Tabled Data

This portion of MSLAP enables the user to make customized tables of the measurements that were made in options 1: and 2: of the FIRST MENU. The investigator loads the "dtl" data structure by reading the .DTL file that was defined in option 1: or 2: of the FIRST MENU. He then can print an abbreviated form of this dtl structure, create a ASCII format table of only the entries of interest, or has several sorting options, including reordering by Laboratory Wavelengths, by Observed Wavelengths, or by ION.

The working text window should be expanded to include the entire screen in this portion of MSLAP. The table creation is designed to handle large tables and the full-screen window prevents text from rapping around, making it difficult to read.

The routine is very generalized. Values supplied by standard MSLAP as well as those in the UserParameter (up) can be accessed and displayed in any order. Data from more than one file can be combined into one master table. Measurements that are absent in one file, but present in another, are supplied with epsilons in the master table. However, all files that are to be combined must be sorted in the same fashion.

Option 5 (to create a customized table in ASCII format) is particularly powerful. It allows the user to build the table, taking any number of pieces in any order and add these to an existing 2-dimensional character string. Parts of several data files can be included as well. Figure 9 shows the initial table at the start of Option 5. A small (the first 6 lines) portion of the table is shown at the top, while a list of options for adding to this 2-dimensional character

field is provided at the bottom. Initially only the species identifications, the wavelengths, and the oscillator strengths are included. A scale is provided immediately below the partial table to assist with the addition and deletion of columns of individual characters. The researcher then adds to, or subtracts from this 2-dimensional character field, using options 1 through 11.





Figure 10 shows the development as the result of sequentially invoking options 4, 5, 6, 7, 8, and 9. Notice that an underline starting with the file name "test" extends to the right over all of this data. Once data from a new file is added the name changes and a new under line will continue from that point. Two data sets that have been combined from files: test2 and test3 as shown below. Notice that some minor editing of the file (such as the creation of true ellipses) is required.

			test2_-----	test3_-----			
Species	Wavelength	f	EQW (A)	ME (A)	EQW (A)	ME (A)	
Fe II	1125.44	0.0110	0.0275	0.0013	0.0275	0.0013	
Fe II	1125.44	0.0110	0.0281	0.0012	0.0281	0.0012	
Fe II	1125.44	0.0110	.....		0.0262	0.0012	
CN R(0)	3874.61	0.0338	0.0076	0.0002	0.0076	0.0002	
CN R(0)	3874.61	0.0338	0.0169	0.0009	0.0076	0.0002	
CN R(0)	3874.61	0.0338	.....		0.0077	0.0002	
CN R(0)	3874.61	0.0338	.....		0.0077	0.0002	
CN R(0)	3874.61	0.0338	.....		0.0169	0.0009	
CN R(0)	3874.61	0.0338	.....		0.0879	0.0227	
CN R(0)	3874.61	0.0338	.....		0.1160	0.0413	
CN R(0)	3874.61	0.0338	.....		0.0076	0.0002	
CN R(1)	3874.00	0.0228	0.0021	0.0002	0.0020	0.0002	
CN R(1)	3874.00	0.0228	0.0020	0.0003	0.0020	0.0002	
CN R(1)	3874.00	0.0228	0.0020	0.0002	0.0020	0.0002	

#### 4: Edit Tabled Data

This portion of MSLAP allows the user to perform basic editing functions on the dtl data structure. The routine is modeled after primitive line editors that were common in the days of the PDP-11 computers. Single key strokes control the editing functions. The user can move up or down by one line, move to the top or bottom of the data structure, insert or delete a line, or change a line. A line is defined currently as a complete set of measurements for a single profile, including any comments and any values in the user parameter. A small window is opened on the right side of the screen and a complete list of the functions is printed as a reminder to the user.

The changes become permanent if the users updates the file on disk. Otherwise, the researcher may Kill/Quit the session at any time without making any additional changes since the last time the disk file was updated.

When the option to changes the values in a single "line" is used (i.e. pressing a "c"), all values are defaulted to the existing values before entering this mode. In this manner, the investigator

only has to strike the `return` key repeatedly until he reaches the value(s) to be changed. This feature reduces the probability that typographical errors will be introduced while attempting to make minor changes.

## 5: Analyze Column Densities vs Velocity

In this portion of MSLAP, the investigator cycles through a series of MENUs over the plot window. The MENUs are ordered in the following hierarchical sequence to determine: 1) what to do next, 2) the species, 3) which profile (identified by wavelength), and 4) plotting symbol or style. After each profile has been plotted the researcher can then get additional profiles or make adjustments to the one just plotted. Figure 11 shows an example of the Fe II column density as a function of velocity for several profiles.

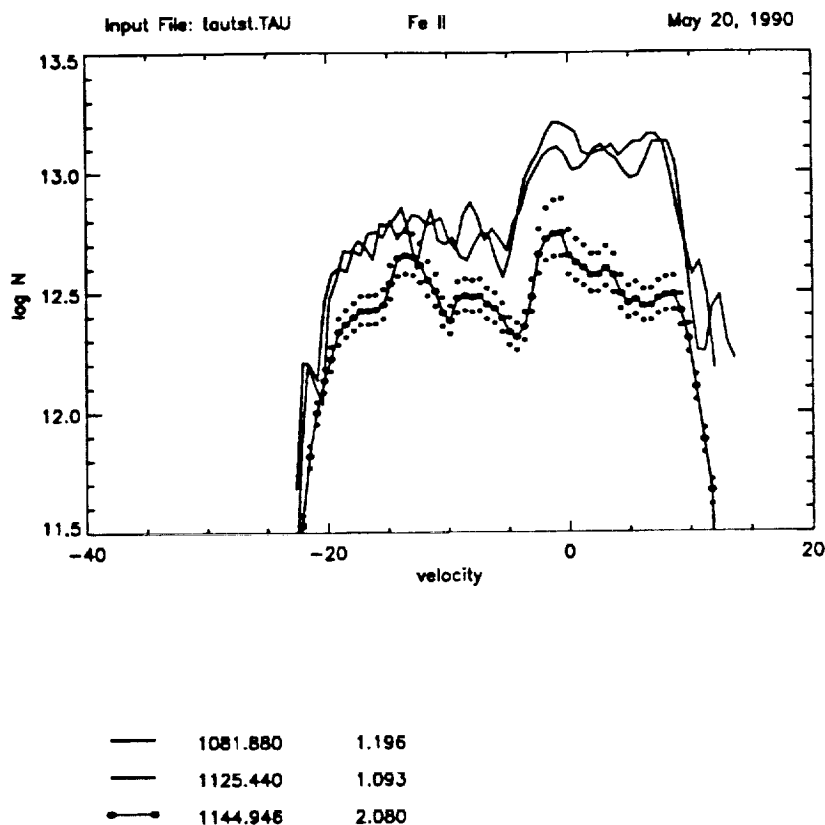


Fig. 11 showing the output from option 5:

Figure 11 shows the column-density results of two weaker profiles of Fe II, which are in good agreement with each other, plotted as continuous lines. The other over plot (solid line with dots) is for an intrinsically stronger absorption line. The disagreement indicates the presence of narrow, unresolved, saturated structure. Figure 11, also showing two of the different plot options, has error dots above and below for the strong line. A key of the plot symbols is provided at the bottom of Figure 11, showing the wavelength and  $\log(f\lambda)$  values.

### 6: Compare Data to Curve of Growth

The Curve of Growth routine is very similar to the one to analyze column densities as a function of velocity. The investigator configures the plotting through a series of MENUs that appear over the plot window. The first species to be plotted is selected, its plotting symbol, and the number of theoretical curves. All of these can be changed dynamically at any time to assemble a figure as shown in Figure 12. The software then presents the data and the investigator moves these data points horizontally by using the cursor to note starting and ending positions. Once satisfied with the fit, the researcher can then get additional data points to fit, or has a host of other options by using the right mouse button.

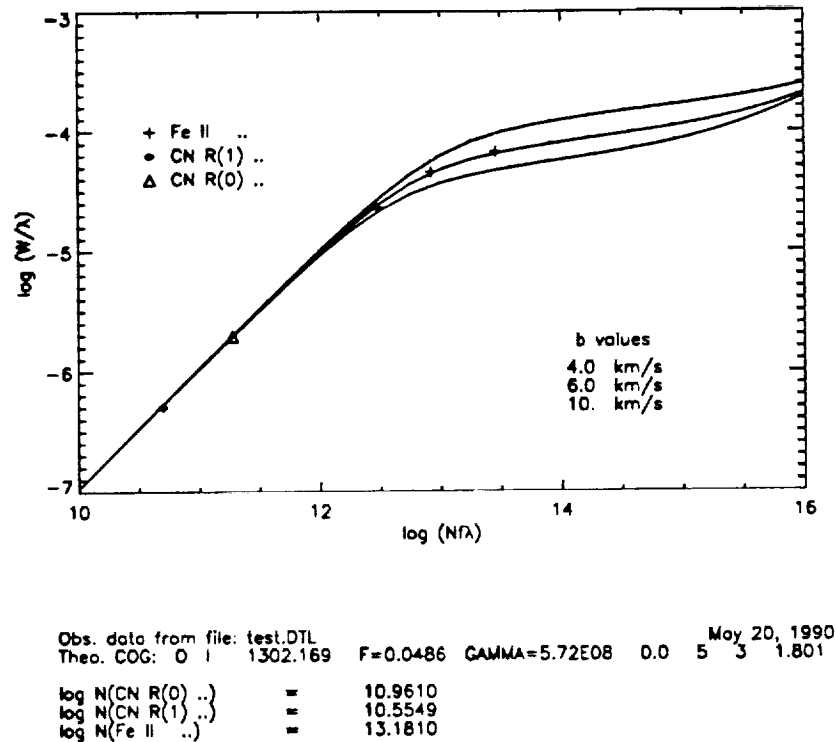


Fig. 12 showing the output of the Curve of Growth

## IV. Comments on Correct Statistical Techniques

The subject of Correct Statistical Techniques is one that requires numerous pages to be complete. Unfortunately, time pressures on the author do not permit an adequate treatment in the current volume. It is hoped that future updates of the MSLAP Documentation will contain substantially more topics and details on the various aspects of using good statistical techniques as they apply to the MSLAP analysis package. There are several issues, however, that must be stated, even if only briefly.

1) Getting a good fit of the continuum is particularly important. The uncertainties in all of the measurements are determined from the residuals of the real data minus the fitted continuum for each point in fitting region. If there is a systematic error produced over some



portion of the spectra, this may manifest itself in the form of unrealistically large values of the random uncertainties. To avoid this difficulty, one might be tempted always to fit with a polynomial of the highest order available on the assumption that such a fit should be equal to, or better than that from a lower order polynomial. However, such action is not without risk. If a polynomial with too high of an order number is selected, the continuum fit occasionally may be ill behaved, especially over the regions where it is not constrained. In other words, the fitted continuum may not represent the real continuum at precisely those wavelengths where it is needed most, over the absorption profile. The author did not have to search very far through his own data to see several examples of this problem. Figure 13 shows a first (solid line) and fifth (dashed line) order fit to the same Fe II profile. The first order fit is the appropriate choice based on the F Distribution Tests for selecting polynomials of various order. The first, second, third, and fourth order fits are almost indistinguishable from each other. If one were blindly to use the highest (5th) order available, the equivalent width would be over estimated by more than 1 sigma of the best estimate. Other profiles exhibited far less dramatic continuum shapes, but just as severe differences in the measured values.

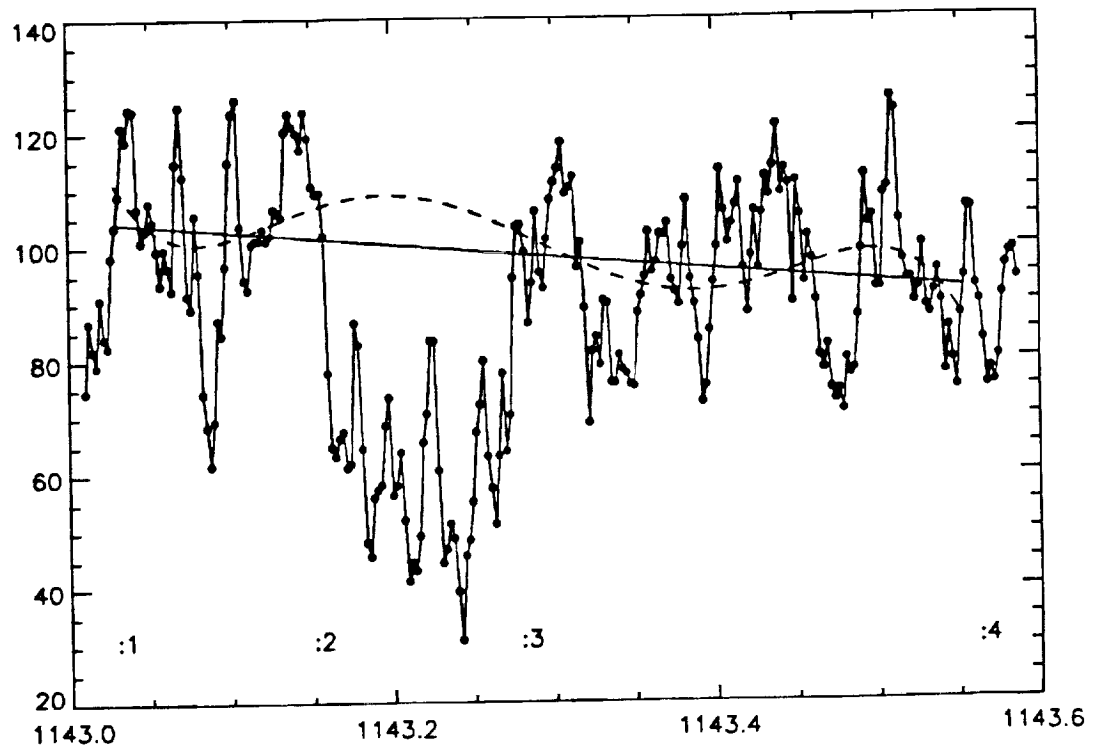


Fig. 13 showing different polynomial fits.

2) In MSLAP, the user specifies which regions of the spectra that he believes to be free of any spectral feature for the purposes of performing a continuum fit. The subjective responses of the researcher can make it very easy for him to avoid preferentially any 2-, or 3-sigma, deviations in choosing these regions. This action, of course, skews the uncertainty calculations towards smaller errors. While 3-sigma events are uncommon, these still have an impact on the total uncertainty since the contribution to the sum from each data point is quadratic in nature.

3) In the standard integrating program, the user specifies the starting and stopping points. As with issue 2, there is a human temptation capable of introducing systematic errors. In the

case of absorption features, many researchers choose the end points of the integration to be the places where the spectrum crosses the fitted continuum closest to the center of the profile. This action leads to a predictable, systematic over estimation of the equivalent widths and second moments (Joseph 1989, PASP, 101, 623).

4) There are a number of common oversights. For example, the background uncertainty in many applications is the dominant source of error, but it is often completely ignored by the investigator. Likewise, the researcher needs to pay attention to the coherence length of the noise. The easiest way to visualize this coherence length is to consider first a spectra produced by an instrument in which each pixel is totally linearly independent of all the others. These data would then have a coherence length of unity. If, instead, the value in each pixel in the instrument was equally dependent on the values of its adjacent pixels on either side, or if the previous data have been smoothed by a 3-point running boxcar, then the coherence length would be 3. For various reasons of data handling and instrumental affects, most data to be analyzed usually have a coherence factor greater than unity.

## V. Customizing MSLAP

MSLAP makes use of a IDL feature where subsequent calls to compile a routine with an identical name overlays the previous. The routines have been organized into several files in part according to the frequency in which each is expected to be modified. This organization is an attempt to minimize the amount of code the typical researcher has to wade through in order to make his desired changes. For example, 90% of all alterations are expected to occur in the file `mslap.pro`, containing less than 350 lines of code compared to the more than 4,000 lines for the entire MSLAP package. All source code is listed in Appendix A, including a table of contents with brief description of each routine.

As already stated, most modifications to MSLAP are expected to be implemented inside `mslap.pro`. Before running MSLAP, each user copies the `mslap.pro` file to his own directory. This file contains a number of dummy programs that have been commented out, but with instructions showing how to install altered forms of these routines. Thus, many investigators can have their own customized version of MSLAP without having to keep a complete copy of all of the source code. In fact, occasionally a researcher will have two or more customized forms of MSLAP in his account.

The rest of this section is divided into several specific applications. First, a detailed description of the `mslap.pro` file is given. Next, the data-getting routines and the file `dgets.pro` is presented. Finally, the structure and use of the Look Up Tables is discussed.

### a. The `mslap.pro` File

The file `mslap.pro` contains many useful dummy programs that have been broken out of the main software package. Some of these are real dummy programs, allowing the user to customize MSLAP merely by inserting a few lines of code, while others are actually comment fields for the purpose of being able to overlay different individualize routines. In addition, there is a routine at the beginning of `mslap.pro` called: `usermenu`, which provides the character strings used to print the primary MENU of options found above the graphic plot when

the right mouse button is pressed. This MENU is the principle means of branching once the main portion of MSLAP is running. Occasionally, a researcher will develop a mental block regarding the meaning of a question or menu option, which can be especially problematic if the investigator uses the software infrequently. In these cases, the user is invited to modify these character strings to make them more meaningful. The functionality, however, is defined elsewhere and remains unchanged.

There are 3 dummy routines, `userprog1`, `userprog2`, and `userprog5`, reserved for the user to create specialize programs. These routines are accessed during the execution of MSLAP by first pressing the right mouse button and then selecting the appropriate menu option. `Userprog1` and `2`, either of which can be executed from Options 1: or 2: in the FIRST MENU, are particularly useful in cases where the set of calculations are not always performed. If a set of specialized calculations is to be performed each time a profile is measured then `auto_int_sav` may be the more appropriate dummy routine to use. The latter is called every time the various moments of the profile are calculated. In a similar fashion, `userprog5` is used for inserting customized calculations during the portion of MSLAP that analyzes the Optical Depths/Column Densities (Option 5: of the FIRST MENU).

Furthermore, there are 2 routines, `plotlab1` and `plotlab2`, that are automatically called during the plotting of spectra and expanded spectra, respectively. These routines can be used, for example, to place additional labels on the plots.

## b. The `dgets.pro` File

There are 5 slots available in MSLAP for the data-getting routines used to read the files containing the input spectra. These routines, called `dget1`, `dget2`, ..., `dget5`, are formally outside of MSLAP, and therefore, are the sole responsibility of the user to write or obtain from some other source. DGET routines for various data sets, however, are being written all the time and these contributions will be continually added to the library. While MSLAP only supports up to 5 `dget` routines at any one time, each investigator chooses the 5 that best meets his needs.

These files are "linked" into MSLAP by issuing indirect compile statements in the `mslap.pro` file. For instance, the syntax to compile the `dget2.pro` file is: `"@dget2.pro"`. The user need only include as many (5 max.) `dget` routines as he needs. For convenience, a file called `dgets.pro`, containing 5 sample `dget` routines in a single file, has been included in the library. An indirect compile statement of this file avoids including 5 similar statements in `mslap.pro` and minimizes the number of file names in the researcher's directory.

To install a `dget` routine, the user must place the indirect compile statement in `mslap.pro` as already mentioned and must include the following statements at the beginning of the program:

```
if mp.dget eq 0 then begin
  mp.dget = 1
  return
end
```

These statements form the mechanism that MSLAP uses to sense that a valid `dget` routine is present and supercedes the dummy one that is otherwise supplied. As an added feature, the string variable called "ID" can be set to a brief, descriptive text, indicating the nature of the routine. This variable will also appear as part of the print-out of available data-getting

routines. For example, adding the line:

```
ID = '**** This is my favorite routine ****'
```

to dget5.pro just after the procedure definition statement will produce the following entry during the set up phase of MSLAP:

-----  
DATAGET Options:

- 1) IUE Standard GO Files for High-Res. Spectra (DISKGET)
  - 2) NOT Being Used
  - 3) ASCII Format of Wavelength-Spectra-Quality-BG
  - 4) 1024-Element Stand-Alone Data
  - 5) \*\*\*\* This is my favorite routine \*\*\*\*
- 

Which one would you like ?

Finally, the user should place in the parameter mp.FNAM, the file name being opened by the dget routine. MSLAP displays this parameter as the input data source above some plots. Other features include: mp.order, mp.CAM, mp.bg, and mp.bgerr, which indicate echelle order number, the camera number, the background level, and the error in the background level, respectively. Note: all parameters in the three main data structures are global in nature and thus, can be accessed from any routine. See Appendix B for a complete listing along with descriptions of functionality.

### c. Look Up Tables

MSLAP creates a file called ulut.tab in the working directory of the investigator when he first requests to store an species that he has identified. If this file exists, MSLAP searches it for close matches every time an identification is requested. MSLAP defines a close match as being within 10 pixels or 0.1 Å which ever is smallest. The value of the search agreement can be selected by setting: mcntrl.wtol. Note: removing the file ulut.tab destroys all user-ID entries.

Software exists to convert any look up table that may exist in ASCII format into one that can be used by MSLAP. For the time being, interested parties should contact the first author regarding this utility.

## VI. Installing MSLAP (for the system manager)

Obviously, the location of the software and tabled information can be organized in a number of ways. The present discussion, however, will only describe the simplest method of installing MSLAP. Note that MSLAP is written in the IDL language and must run on a workstation

supporting IDL.

First select or create a directory to hold the MSLAP programs and change to that directory. The entire package requires about 500 kilobytes of space. Next create a subdirectory called "tabdata" to hold the tabled information. Standard MSLAP expects the look-up tables to be located in a subdirectory of the directory holding the programs. All files containing tabled data have a ".tab" extension. The rest of the files should be placed in the parent directory. Make sure that all of the files have read-only permission. IDL programs are always compiled in real time from the source code. Thus, it is unnecessary for individual users to have either execute or write permission.

Edit the files: main.mslap, master.aux, and mslap.pro so that all occurrences of the string sequence: /u/clj/mslapdir are replaced with a string sequence appropriate for the path to the parent directory holding the MSLAP software. The new string should be sufficiently complete that users can access this directory from any directory where they store their programs and data. Note: be sure to keep the "@" symbol since this is a linking or call to compile command.

Turn to the section on Getting Ready (§II of this Manual) and write in the directory path where you have just installed the MSLAP software. Users will want to copy the mslap.pro file from this directory.

All plot commands that are hardware specific are located in a single file called plotconfig.pro. In standard MSLAP, this file is designed to handle automatically either sunview or X windows and to open one of several types of graphical or text windows, each with a specific location and size. Some minor adjustments to the values specifying size and location, therefore, may be necessary from one machine to another. The plotconfig.pro program also configures the hardware so that IDL plot commands produce PostScript plot files, which serve as hard copies of graphical output appearing at the console. The user of MSLAP can request that these "plots" be sent to the laser printer from inside MSLAP. If the default laser printer is not a PostScript printer, the line of code with the command:

```
spawn,'lpr temp.ps'
```

must be changed so that "lpr" is changed to "lpr -P[name]", where name is the device supporting PostScript. If your computer supports another device such as Hewlett-Packard Graphics Language (HP-GL) which IDL is equipped to handle, then you will need to consult the IDL manual for the appropriate commands to substitute in plotconfig.pro. Only the code inside plotconfig.pro needs to be altered, however. If you do not have a hard copy unit that IDL recognizes then you should comment out those portions of plotconfig.pro. Note: the appearance of a semicolon causes IDL to consider the remainder of that line to be a comment field.

The file dgets.pro contains a set of 5 data-getting routines, all of which are formally outside of MSLAP. Actually, it is the responsibility of the researcher using MSLAP to write or substitute as necessary various dget routines so that his data can be read into MSLAP. However, if you represent a guest-user facility, you may wish to substitute several dget routines with ones that are suitable for your specific data formats. The new combination can either replace the dgets.pro file or an alternative file can be generated. Then a guest user can simply copy one of these files to his own directory and run MSLAP with minimal amount of effort to get started.

Finally a word of caution. MSLAP is designed so that customized routines can be inserted with minimal difficulty. The way that this is accomplished is to make use of a feature of IDL

where a subsequent compilation of a routine with the same name overlays the previous. A number of dummy routines exist as place holders in the event that no user-written programs are supplied. One may be tempted to consolidate the use of indirect compilation calls (e.g. the use of `@program_name`), but this may eliminate the modularity of MSLAP. The sequence of compilations is important and you should be careful not to change this order inadvertently.

## VII. Trouble Shooting

The most common problem faced by the user of MSLAP appears to be either the failure to register a cursor reading or the registering of multiple readings from a single stroke of the mouse. For this reason, MSLAP normally attempts to reward the investigator with some type of output every time that he inputs a cursor location. The continuum fitting routine can be the most problematic since many cursor locations are read to set the intervals to be used for the polynomial fit. During this activity, the routine marks each selection with a colon ":" and a number beside it. The regions to be fit, thus, are from 1 to 2, 3 to 4, 5 to 6, and so on. The researcher is encouraged to monitor these numbers while selecting the ranges. If two numbers are superimposed or if one is missed, the continuum fitting routine should be started again. If improper registering of cursor readings becomes severe, then IDL "WAIT" statements will have to be included.

The data-getting routines (`dget1`, `dget2`, ..., `dget5`) are formally outside of MSLAP. Some unexpected behavior in MSLAP can occur if these routines do not make use of all of the required variables in the MSLAP-Parameter (`mp`) structure. For example, when the user requests new data in MSLAP, the software goes to one of the `dgets` provided by the researcher. That `dget` routine could solicit a new file name and read new spectra, but not place the updated file name in the `mp.FNAM` parameter. Then subsequent plots will not reflect the true input file name. Note that in some applications such as echelle data, it is desirable to configure the `dget` routine so that subsequent calls to `dget` return spectra from the next order and not open a new file. For other types of data, it is more natural to have a single spectrum for each file. Hence, the responsibility to return the necessary parameters to MSLAP has been left to the user. (See the section on the `dgets.pro` file.)

## APPENDIX A - The Source Code

Routine Name	Description	Page
----- In the file: main.mslap -----		
details.pro	- The MAIN Program for Options #1 & #2 from the First Menu.....	30
flagset.pro	- A set-up routine used by details.pro .....	33
graphs.pro	- Does the plotting/labeling for details.pro, but does not plot expanded portions.....	34
setup.pro	- A set-up routine used by details.pro .....	36
contin.pro	- The continuum fitting routine .....	38
store.pro	- Stores the result from details.pro in the dtl data structure and on disk (see fstore).....	47
fstore.pro	- Stores optical depths to disk (see store) .....	48
expand.pro	- Extracts a suitable portion of the spectra for an expanded plot for contin.pro .....	50
the main prog.	- Provides the FIRST MENU and calls, details.pro, posto.pro, edatdtl.pro mantau.pro, or cog.pro .....	51
----- In the file: master.aux -----		
dget(1-5).pro	- The Dummy Versions as place holders .....	52
curfit.pro	- Calculates a single polynomial fit .....	53
eqsol.pro	- Special routine called only by curfit.pro .....	54
ucursor.pro	- Provides cursors with special features. Also, provides Continuum Menu Options .....	55
dataget.pro	- An interface routine between MSLAP and dget.pro ...	58
----- In the file: compare.pro -----		
compare.pro	- To find Look Up Table (LUT) entries, main driver ....	60
userid.pro	- To find User LUT entries or make new ones .....	62
----- In the file: posto.pro -----		
posto.pro	- Organizes the tabling of the data base .....	64
srt.pro	- Sorts the data for posto.pro .....	65
publ.pro	- Lists (Publication Form) the data for posto.pro ....	66
mkcusttab.pro	- Makes Customized Tables in ASCII format .....	67
appnd.pro	- Called by mkcusttab.ro to append entries .....	71
nextstring.pro	- Called by APPND & MKCUSTTAB to get the size of an entry and to convert to string .....	73
----- In the file: edatdtl.pro -----		
edatdtl.pro	- Main program for editing the "dtl" data structure ...	74
getsdata.pro	- Get single set of measurements .....	76
----- In the file: mantau.pro -----		

.pro	- Manipulate (tau) optical depths .....	79
getrdy.pro	- A set up routine (GetReady) .....	86
----- In the file: intgrt.pro -----		
intgrt.pro	- The primary integrating routine .....	88
	(does significant plotting)	
equivw.pro	- Calculates the moments plus optical depths .....	92
	(originally just an Equivalent Width routine)	
----- In the file: plotconfig.pro -----		
plotconfig.pro	- Sets up the hardware for plotting functions .....	95
----- In the file: cog.pro -----		
cog.pro	- Curve-Of-Growth primary routine .....	97
fireup.pro	- A set up routine .....	100
lines.pro	- Gets and plots the theoretical curves .....	102
grab.pro	- Grabs the observed data points to be fit .....	103
plotem.pro	- Plots the data from GRAB.PRO with various symb.....	104
whichcog.pro	- Produces MENUS to input # of theo. COGS .....	105



\*\*\*\*\*

Modular Spectral Line Analysis Program  
M.S.L.A.P. version 1.0

copyright (c) 1991 by Charles L. Joseph and Edward B. Jenkins  
All rights reserved. A license may be obtained from the  
first author or from an authorized distribution center.

This software has been distributed through the GHRs Data Center.

\*\*\*\*\*

THE FOLLOWING OPTIONS ARE AVAILABLE:

- 1: Measure Profile Moments Only  
(Equivalent Width, Profile Centroid, etc.)
- 2: Measure Moments and Optical Depths
- 3: Manipulate Tabled Data
- 4: Edit Tabled Data
- 5: Analyze Column Densities vs Velocity  
on Option #2 Data
- 6: Compare Data to Curve of Growth
- 10: TO EXIT FROM THIS PROGRAM

-----  
COPYRIGHT (c) 1984, by Charles L. Joseph  
COPYRIGHT (c) 1991, by Charles L. Joseph & Edward B. Jenkins

Released: 01-February-1991  
Alteration History: None since release date.

-----  
PRO DETAILS,sff,timedata,libr,up

\*\*\*\*\* DETAILS.PRO \*\*\*\*\*

To control the measurement of equivalent widths or integrated fluxes  
BY CHARLES L. JOSEPH 09-SEPT-80 11-MAY-90

sff 0 => Do not do Optical Depths 1 => Do Optical Depths (TAU)  
timedata Variable containing just the date.  
up UserParameter - 30 element floating vector, which is global.





```

!x.range = [0.,0.] ; Reset to auto scaling
!y.range = [0.,0.] ; for the plots.
goto, IT
endif
IF mp.cntrl gt 2 THEN GOTO,NEXT ; Flag for next spectrum.
TAU = fltarr(6,8)+100. ; Make dummy just in case
INTGRT,ESP,EWA,CNT,TAU,JE,IF,XI,XLIMIT,mdata,mp,mcntrl,up ; Calls EQUIV.PRO &
; plots results.
!x.range = [0.,0.] ; Reset to auto scaling
!y.range = [0.,0.] ; for the plots.
;
;
Storit: IF mp.cntrl eq 1 THEN STORE,WV,ML,dtl,mdata,mp,mcntrl,up ; Store .DTL
if (mp.cntrl eq 1) and (sff eq 1) then begin ; Optical depths too?
fstore,TAU,sdata,mp,mcntrl,up ; If so, plot & store.
endif
IF (opt ne 2) AND (mp.cntrl ne -99) THEN GOTO, IT ; Do some more or go
; back to MSLAP
;
FINISH: wdelete,0 ; Remove plot windows.
wdelete,1 ; Remove plot windows.
if sff eq 1 then close,5 ; Close .TAU file.
close,1 ; Close .DTL file.
openu,1,mp.STAR+'.DTL' ; Open again to catch
writeu,1,dtl ; any last minute changes
close,1 ; Close .DTL file again.
;
;
RETURN ;
END ; DETAILS
;
;
;
PRO FLAGSET,FF,mp,up
;***** FLAGSET.PRO *****
;
; by Charles L. Joseph June 29, 1990
;
; This procedure determines which look up table and data-getting
; routine are to be used.
;
; FF flag for which DGET routine to use
; mp MSLAPparameter - a structure
; up UserParameter - 30 pt floating array
;
;*****
DTY=1
DETR: for k = 0,3 do print,' ' ; print lookup tables
print,'-----'
print,'Available lookup tables are:'
print,' '
print,' 1) Interstellar Lines (Morton and Smith 1973 plus updates)'
print,' 2) Molecular Hydrogen, HD, and CO'
print,' 3) Options 1 & 2 Combined'
print,' 4) Hot-Star Lines'

```

```

print,'-----'
for k=1,2 do print,' '
READ,'Which one would you like ?',DTY           ; find out which one
IF(DTY LT 1) OR ( DTY GT 4) THEN GOTO,DETR       ; no such table
mp.DTY = DTY
RFL: up = fltarr(30)                             ; define User-Param.
iok = intarr(6)
for k = 0,3 do print,' '
print,'-----'
print,'DATAGET Options:'
print,' '
mp.dget = 0                                       ; Sense which data-
dget1,FNAM,w,spect,q,ihdr,ID,mp,up              ; get routines are
if mp.dget eq 1 then iok(1) = 1                  ; present and show
print,'    1) ',ID                               ; options.
mp.dget = 0
dget2,FNAM,w,spect,q,ihdr,ID,mp,up              ;
if mp.dget eq 1 then iok(2) = 1
print,'    2) ',ID
mp.dget = 0
dget3,FNAM,w,spect,q,ihdr,ID,mp,up
if mp.dget eq 1 then iok(3) = 1
print,'    3) ',ID
mp.dget = 0
dget4,FNAM,w,spect,q,ihdr,ID,mp,up
if mp.dget eq 1 then iok(4) = 1
print,'    4) ',ID
mp.dget = 0
dget5,FNAM,w,spect,q,ihdr,ID,mp,up
if mp.dget eq 1 then iok(5) = 1
print,'    5) ',ID
print,'-----'
FF = total(iok)                                  ; How many valid progs?
if FF eq 0 then begin                             ; If none, complain.
print,string(7B),'There are no valid data-getting routines loaded.'
print,'At least 1 valid routine must be put in the file: dgets.pro'
FF = ' '
read,'Press <Return> to exit',FF                 ; Pause so user realizes
retall                                           ; the trouble & abort.
endif
for k=1,2 do print,' '
READ,'Which one would you like ?',FF           ; Find out which one.
if (FF lt 1) OR (FF gt 5) then goto,RFL        ; Valid Choice?
if iok(FF) ne 1 then goto,RFL                  ; Do this one exist?
if iok(FF) eq 1 then mp.dget = 1              ; Insure flag is set.
RETURN
END
;
;
;
PRO GRAPHS,SPECT,WA,IHDR,JJ,mp,mcntrl,up
;***** GRAPHES.PRO *****

```

```

;
;   TO PLOT THE SPECTRUM.      By Charles L. Joseph   Sept. 1980  Nov, 1990
;
;   SPECT    flux vector
;   WA       wavelength vector
;   IHDR     header vector
;   INDEX    plot or replot status
;   JJ       data quality vector
;   FNAM     data file name
;   mcntrl   structure (see DETAILS.PRO)
;   mp       MSLAP structured variable: mp.cntrl = control flag
;                                       mp.order = order number
;                                       mp.cam   = camera number
;   up       UserParameter - 30 pt floating array, exclusive for user.
;
;*****
;
!y.margin(0) = 4.
FNAM = mp.FNAM
star = mp.STAR
kwtd = 0 ; What-to-do HardCopy Flag.
if mp.cntrl eq -10 then begin
  kwtd = wmenu(['Copy?','Yes','No'],title=0, init=2)
  if kwtd eq 1 then plotconfig,1,' ',' ',-1,kdev,' ; Config for Hardcopy.
endif
t10 = 'Output: '+star+ ' M.S.L.A.P. Input: '+FNAM
t10 = t10+ ' '+mcntrl.date
xxxs = fltarr(2)
yyys = fltarr(2)
sz = size(WA)
jcnt = 1
if sz(1) gt 16 then jcnt = fix(sz(1)/16)
if mp.cntrl eq -1 then plotconfig,0,t10,' ',1,kdev,' ; Set up graphics device.
ymn = min(SPECT,J5, MAX=ymx) ; Find min and max flux.
if ymn gt 0 then ytst = ymn ELSE ytst = ymx ; For bottom of plot use 0
if ytst lt 0.75*ymx then ymn = 0. ; unless featureless.
xmx = max(WA,jmx, MIN=xmn) ; Min & max plus max(index)
vmn = mp.bg - mp.bgerr - 0.1*(ymx-ymn) ; Get min. needed for BG
if ymn eq 0. and vmn gt 0 then ytst=1 ELSE ytst=0 ; Prevent double jeopardy?
if ymn le vmn then vmn = ymn - 0.1*(ymx-ymn) ; Less than min. spect?
if ytst eq 1 then vmn = 0. ; Correct double jeopardy.
ymx = 1.1*ymx ; Adjust for more space.
xxxs(0) = xmn & xxxs(1) = xmx ; Put limits into small
yyys(0) = vmn & yyys(1) = ymx ; over plot array &
if !x.range(0) OR !x.range(1) OR !y.range(0) OR !y.range(1) then begin
  if !x.range(0) OR !x.range(1) then begin ; Use these values instead?
xmn = !x.range(0) ; If one is nonzero => user
xmx = !x.range(1) ; wants to over ride.
  endif
  if !y.range(0) OR !y.range(1) then begin ; Use these values instead?
vmn = !y.range(0)
ymx = !y.range(1)

```

```

endif
endif
plot,xxxs,yyys,/NODATA, xtitle='Wavelength (A)', ytitle='Rel. Flux', $
  xrange=[xmn,xxm],yrange=[ymn,ymx],xstyle=1,ystyle=1
if mp.SMO then begin
  OPLOT,WA,SMOOTH(SPECT,3), $ ; Plot smoothed flux.
  Xtitle='Wavelength Angstroms',Ytitle='Rel. Flux'
endif else begin
  OPLOT,WA,SPECT, $ ; Plot unsmoothed flux.
  Xtitle='Wavelength Angstroms',Ytitle='Rel. Flux'
endif
OPLOT,WA,JJ ; Plot data quality.
if (J5 gt jcnt) then xxxs(0) = WA(J5-jcnt) ELSE xxxs(0)=WA(0) ; BG range.
if (J5 lt jmx-jcnt) then xxxs(1)=WA(J5+jcnt) ELSE xxxs(0)=WA(jmx)
yyys(0:1) = mp.bg
oplot,xxxs,yyys ; Over plot BG.
yyys(0:1) = mp.bg - mp.bgerr
oplot,xxxs,yyys,linestyle=2 ; Over plot BG-BGerror.
yyys(0:1) = mp.bg + mp.bgerr
oplot,xxxs,yyys,linestyle=2 ; Over plot BG+BGerror.
plotlab1,WA,SPECT,JJ,mp,mcntrl,up ; Customized labels.
if mp.order gt 0 then PRINT,' Order' ; Print order # .
if mp.order gt 0 then PRINT,mp.order ;
if kwtd then begin
  xyouts,0.5,1.1,FNAM,/NORMAL ; Put file name on plot.
  plotconfig,-1,' ',' ',-1,kdev,'' ; Send plot & -> term.
endif
IF mp.cam NE 0 THEN BEGIN ; Valid IUE CAM #?
  IF mp.cam LE 2 THEN PRINT,' long' ELSE PRINT,' short'
  END ;
mp.cntrl=1 ; Set to norm.
RETURN ; Go back
END ;
;
;
;
;
PRO SETUP,ML,RS,FF,dtl,nullld,nside,mp,mcntrl,up
;***** SETUP.PRO *****
;
; Gets or creates a new .DTL file as well as other set up information
; for DETAILS. by Charles L. Joseph June 1981 May 1990.
;
; ML maximum number of measurements that can be made.
; RS Red Shift (Doppler velocity) used in LUT.
; dtl Structure that holds contents of .DTL file
; nullld A nulled or zero version of the variable dtl
; nside # of points on each side of the expanded plots - set to 60
; mp MSLAPparameter - a structure, used to pass many arguments.
; mcntrl structure (see DETAILS.PRO)
; up UserParmeter - 30 pt floating array, exclusive for user.
;

```

```

;
;*****
ML=200 ; max # of lines
I=0 ; set master index
nside = 150 ; # of expansion points.
START: STAR=' ' ; name loop
print,' '
READ,'What is the output file name ?',STAR
IF STAR EQ 'H' THEN BEGIN ; need help ?
    SPAWN,'ls *.DTL' ; list old files
    GOTO,START ; start again
END ;
PRINT,' ' ;
close,1
openr,1,STAR+'.DTL', ERROR = errtst ; Does file exists?
close,1
app = ' ' ; Set append flag to "NO"
if errtst eq 0 then begin ; File Found -----
    CLOSE,1
    OPENU,1,STAR+'.DTL' ; Get the old file.
    ML=200 ; MAX meas. that can be made
    readu,1,dtl ; Read the file.
    sz = where(dtl.iaf ne 0.) ; Get # of measurements.
    sz = size(sz)
    if (sz(0) eq 0) then I = 0 else I = sz(1) ; If some made, set counter.
    close,1 ; Close file for now.
    print,string(7B),'File Already Exists' ; Flash Screen and print.
    print,I,' measurements have already been made.' ; Indicate # already made.
    print,' '
    print,'Append new data to the old?' ; Append or erase & start?
    read,'(Note: a NO will erase old data.)',app ;
endif

if (app ne 'Y') AND (app ne 'y') then begin ; If APPEND flag NOT set in
    close,1 ; "file exists" section.
    OPENW,1,STAR+'.DTL' ; Make new file.
    dtl = replicate({ noda }, 200) ; Empty array for storage.
    writeu,1,dtl ; Write it to file.
    CLOSE,1 ; Close until latter.
    I = 0
endif
print,' '
print,'The radial velocity is used only to help identify species.'
print,'Enter 0 if you are uncertain of the real value.'
READ,'Enter Delta-Lambda/Lambda or Radial Velocity for the source: ',RS
IF ABS(RS) LT 1.0E-02 THEN BEGIN ; assume D-1/l
    PRINT,'The Delta-Lambda/Lambda is: ',RS ; tell assumption
END ELSE BEGIN ; else rad-vel
    PRINT,'The Radial Velocity is: ',RS,' km/s' ; say so
    RS=RS/299792.5 ; convert to D-1/l
END ;

```



```

FLAGSET,FF,mp,up ; get data type
;
CLOSE,1 ;
mp.STAR = STAR ;
mcntrl.I = I ;
EXIT: RETURN ; go back to DETAILS
END ;
;
;
;
PRO CONTIM,EXSP,EXWA,CNT,WA,EPS,XF,XI,JE,XLIMIT,mp,mcntrl,up,sdata,mdata, $
nulld,RS,nside
;***** CONTIM.PRO *****
;
; TO DETERMINE A CONTINUUM AND END POINTS FOR THE INTERGRATION
; By Charles L. Joseph June, 1982
; By Charles L. Joseph and Edward B. Jenkins May 12, 1990
;
; EXSP expanded flux vector
; EXWA expanded wavelength vector
; CNT continuum vector
; WA wavelength vector
; EPS data quality vector
; XF right stop X value for integration
; XI left stop X value for integration
; JE index of the line center
; XLIMIT array containing start/stop to fit regions.
; mp MSLAPparameter - a structure, used to pass many arguments.
; mcntrl structure (see DETAILS.PRO)
; up UserParmeter - 30 pt floating array, exclusive for user.
; sdata structure containing a single dtl(k) element
; mdata same as sdata, but for 4 elements.
; nulld same as sdata, but all elements are set to 0
; RS red shift (Doppler velocity) for LUT
; nside the number of data points on each side of the expansion.
;
;*****
;
mdata(0) = sdata
m = 0 ; Set Null Polynom. Fit
wmx = max(EXWA, jwmx, MIN=wmn) ; Find max & min lambdas.
START: imes = 0 ; # of discrete pts to fit.
csav = 0 ; Do Not Overplot last CNT.
cmes = fltarr(100,2) ; Holds cursor readings.
kget = 0 ; Index and # for XLIMIT.
iopt = 2 ; Cntrl option, UCURSOR.PRO
AC = dblarr(9,10) ; For CURFIT.PRO
BC = dblarr(16) ; For CURFIT.PRO, fit para.
bcsav = dblarr(10,16) ; To save various BC arrays
IWRK = fltarr(602) ; Working wavelength vector
YWRK = IWRK ; Working flux vector.
VSAV = IWRK ; Temporary holding vector.

```

```

CNT = 0.0*EXWA ; Make empty continuum.
XLIMIT = fltarr(30) ; Fitting end points.
MM = intarr(9) ; For CURFIT.PRO.
COHFAC = 1.0
sz = size(JE) ; Find the # of line cntrs.
if sz(0) eq 0 then lcs = 1 else lcs = sz(1)
xxx = fltarr(lcs,2) ; Array for overplot cntrs.
yyy = fltarr(2) ; Array for overplot cntrs.
yyy(1) = !y.crange(1) ; Set values for top to
yyy(0) = !y.crange(0) ; bottom over plots.
yhalf = 0.5*(yyy(1)-yyy(0)) + yyy(0) ; For cursor positioning.
fdsav = fltarr(2,mcntrl.mtot+1)

kdev = getenv('TERM') ; X Window or Sunview?
if kdev ne 'sun' then kdev = 'xterm' ;

; ----- Plot the spectrum, either smoothed or not -----

IF mp.SMO ne 0 then PLOT,EXWA,SMOOTH(EXSP,7),/YNOZ
if mp.SMO eq 0 then begin
  ymn = min(EXSP,J5, MAX=ymx) ; Find min and max flux.
  xmx = max(EXWA,jmx, MIN=xmn) ; Min & max plus max(index)
  ymn = ymn - 0.1*(ymx-ymn) ; Adjust for plotting.
  ymx = 1.1*ymx
  if !x.range(0) OR !x.range(1) OR !y.range(0) OR !y.range(1) then begin
    if !x.range(0) OR !x.range(1) then begin ; Use these values instead?
      xmn = !x.range(0) ; If one is nonzero => user
      xmx = !x.range(1) ; wants to over ride.
    endif
    if !y.range(0) OR !y.range(1) then begin ; Use these values instead?
      ymn = !y.range(0)
      ymx = !y.range(1)
    endif
  endif
  plot,exwa,exsp,/YNOZ,xrange=[xmn,xmx],yrange=[ymn,ymx],xstyle=1,ystyle=1
  asy = findgen(16)*(!PI*2/16.)
  usersym,0.5*cos(asy),0.5*sin(asy),/FILL
  oplot,exwa,exsp,psym=8 ; Show individual points.
endif
OPLLOT,WA,EPS<0 ; Plot data quality.
for k=0,lcs-1 do begin ; If mult. profiles, then
  if lcs eq 1 then xxx(k,0:1) = EXWA(JE) ; flag them on the plot.
  if lcs gt 1 then xxx(k,0:1) = EXWA(JE(k))
  oplot,xxx(k,0:1),yyy,linestyle=1 ; Use dotted line.
endfor
for k=0,3 do print,' ' ; Let user know in CONTIM.
print,' >>>>>> START of Continuum Fitting Routine <<<<<<<< '

for kk=0,1000 do begin ; Do a large # of attempts.

  if (kk lt 1) then tvcrs,0.5,0.5,/NORMAL ; 1st time? Place cursors.
  if (iopt eq 2) then begin ; 1st time? Instructions.

```

```

print, ' '
print, 'Left Mouse to locate END POINTS of featureless continuum segments'
print, '      - Only horizontal position will be used.'
print, '      - Up to 15 segments are allowed.'
print, 'Right Mouse button provides additional MENU options.'
print, 'Click Center Mouse when finished.'
endif
CHANGE: ucursor,IX,IY,opt,iopt,1,mp ; Graphical input & contrl.
if IX gt wmx then IX = EXWA(jwmx-1) ; Don't except out of range
if IX lt wmn then IX = wmn ; Don't except out of range

; ----- Branching in Continuum Routine -----

if !ERR eq 2 then goto, calc ; Middle Mouse used in
; UCURSOR.PRO.
if !ERR eq 1 then begin ; Left Mouse used.
XLIMIT(kget) = IX ; Store Fitting Limits.
kget = kget + 1 ; Update counter.
yout = 0.07*(!y.crange(1)-!y.crange(0)) + !y.crange(0)
xyouts,ix,yout,':' +strtrim(kget,2) ; Show the fitting limit.
endif

if iopt eq 2 then begin ; Define discrete
; continuum points.
imes = 0
plotconfig,0,' ',' ',3,kdev,' ; Open small instr. window.
xout3 = !x.crange(0)
yout = 0.5*(!y.crange(1)-!y.crange(0)) + !y.crange(0)
xyouts,xout3,yout,'Use Left Mouse - Continue until Middle Mouse'
wset,0 ; Control back to plot.
!ERR = 0
while (!ERR ne 2) AND (imes lt 100) do begin
ucursor,ix,iy,opt,kchoice,0,mp
; if kdev eq 'xterm' then wait,1
if !ERR ne 2 then xyouts,ix,iy,'x' ; Show discrete points.
if !ERR ne 2 then cmes(imes,0) = ix ; Update discrete param.
if !ERR ne 2 then cmes(imes,1) = iy
if !ERR ne 2 then imes = imes + 1 ; Update total number.
endwhile
wdelete,3 ; Remove instruct. window.
wset,0 ; Make sure back to graph.
print, ' ',string(7B) ; Bell and give instruct.
print, '>>>> Resume finding regions of featureless continuum <<<<'
endif

if iopt eq 3 then begin ; ----- ; Redefine line center(s).
print, ' '
print,string(7B),'>>>> Use LEFT Mouse to locate New Line Center <<<<'
print, ' NOTE: If the RIGHT Mouse is used first - then up to 3'
print, ' additional line centers can be defined.'
yout = 0.5*(!y.crange(1)-!y.crange(0)) + !y.crange(0)
if lcs gt 1 then JE = JE(0)

```

```

      TVCRS,EXWA(JE),yout,/DATA
cursor,ix,iy
      if kdev eq 'xterm' then wait,1
xtst = !err ; Temp. store key stroke.
!err = 0
vtmp = where(exva ge ix) ; Get index of exva cor-
JE = vtmp(0) ; responding to input.
kk = 0
if xtst eq 4 then begin ; If RT. Mouse, do more.
  xxx(0:1) = ix ;
  oplot,xxx,yyy,linestyle=1 ; Dotted line of new center
  vtmp = JE ; Save JE to make JE(k).
  JE = intarr(4) ; Make JE into an array.
  JE(0) = vtmp ; Put 1st line center in.
  for k=1,3 do begin
redo: vtmp = strtrim(string(4-k)) ; String of remaining #.
      print,' '
      print,'Use LEFT Mouse to find up to ',vtmp,' more line centers.'
      print,'Use MIDDLE Mouse if finished.'
      cursor,ix,iy ; Simple graphical input.
          if kdev eq 'xterm' then wait,1 ; X window? Slow down.
      if !err eq 2 then goto,lcfin ; Finished? Get out.
      vtmp = where(exva ge ix) ; Get index of exva cor-
      JE(k) = vtmp(0) ; responding to input.
          COMPARE,ix,RS,sdata,mp,mcntrl ; Findout which LUT entry.
      if mp.cntrl eq 0 then goto, redo ; If Not a good ID.
      xxx(0:1) = ix ; Good ID, get ready to
      oplot,xxx,yyy,linestyle=1 ; plot it, dotted line.
      mdata(k) = sdata ; Update Multiple Meas.
      kk = kk + 1 ; Increment counter.
      endfor
endif ; ----- ; END Redefine Line Cntrs.
lcfin: if (kk gt 0) then JE = JE(0:kk) ; How many line cntrs?
goto, START ; Now go fit continuum.
endif

      if iopt eq 4 then begin ; Adjust # of points in the expanded plots?
for k=0,2 do print,' '
print,'MOVE CURSOR into the COMMAND WINDOW to input the number of points'
print,'in the Expanded Plots. MSLAP starts with 150 points on either'
print,'side for 301 total. NOTE: PROGRAM RETURNS TO CALLING ROUTINE'
nside = FIX(nside)
print,'Current setting is:',nside,' for a total of ',2*nside+1,' points'
print,' '
read,'Enter the number of points on each side ',ntmp
if ntmp gt 0 then nside = ntmp
nside = FIX(nside) ; Make sure an integer.
goto, DONE
endif
if iopt eq 5 then goto,START ; Restart CONTIM.PRO.
IF opt eq 9 THEN begin & STOP & goto, CHANGE & end ; Stop for kbd input.
IF opt eq 6 THEN GOTO, DONE ; Get out of CONTIM.PRO.

```

```

IF opt eq 3 THEN BEGIN                                ; Delete last .DTL measure.
dtl(mcntrl.I) = nulld
PRINT, ' Line ',mcntrl.I, ' has been DELETED! ',STRING(7B) ; Print notice
mcntrl.I=mcntrl.I-1                                   ; Set I to over write.
IF mcntrl.I LT 0 THEN mcntrl.I=0                     ; Can't go negative.
ENDIF                                                 ;
IF opt eq 4 then begin                                ; Comment on .DTL measure.?
read,COMS
sdata.com = byte(string(COMS,format='(a10)'))
endif
IF opt eq 2 THEN GOTO, DONE                            ; Return to Main Prog
if opt le 4 then goto, CHANGE                          ; Rem.,Del.,Nothing
if opt eq 7 then usrprog1,WA,SPECT,JJ,mp,up
if opt eq 8 then usrprog2,WA,SPECT,JJ,mp,up
if (opt eq 7) OR (opt eq 8) then goto, CHANGE          ; Usrprog called.
if opt eq 5 then begin                                ; Hardcopy requested,
print, ' ',string(7B)                                  ; but is not allowed
print, '>>> HARD COPY is NOT available HERE <<<' ; Give Warning
print, 'One can be made after each measurement' ;
print, ' '
endif
if opt eq 11 then goto,START                            ; New plot parameters.
endifor

```

```

; ----- End Branching & Selecting Continuum Constraints -----

```

```

calc: print, ' '
if imes ge 1 then XWRK(0) = cmes(0:imes-1,0)
if imes ge 1 then YWRK(0) = cmes(0:imes-1,1)
if (kget lt 2) AND (imes lt 2) then begin ; >>> NOT ENOUGH END POINTS? <<<
kchoice = wmenu(['Exit Continuum Fitting?','YES','NO'],title=0,init=1)
if kchoice ne 1 then goto, CHANGE ; Go get more option OR tell
mp.cntrl = 0 ; calling routine ABORTED &
goto, OUT ; get out.
endif
print, ' Calculating various polynomial fits and the statistics to perform'
print, ' F TESTS for the validity of using higher order polynomials.'
print, ' -- Please Wait --'

```

```

; ----- Strip out those portions that are featureless and put compactly ---
; ----- into vector XWRK. Start at XWRK(jjj) each time through. -----

```

```

k = 0
if imes gt 0 THEN jjj=imes-1 ELSE jjj=0 ; Start index after discrete pts
for ii=0,kget/2-1 do begin ; Search for beginning and end-
xtst = XLIMIT(2*ii) ; ing indices for featureless
while EXWA(k) lt xtst do k=k+1 ; portions and stuff into XWRK.
jj1 = k
xtst = XLIMIT(2*ii+1)
while EXWA(k) lt xtst do k=k+1
jj2 = k

```

```

        XWRK(jjj) = EXWA(jj1:jj2)           ; Put into working array.
        YWRK(jjj) = EXSP(jj1:jj2)         ; Put into working array.
        jjj = jjj + jj2 - jj1             ; # of elements in working array
    endfor

    if (imes lt 3) AND (kget eq 0) then begin ; ----- Straight line fit?
VSAV = where(EXWA ge cmes(0,0) AND EXWA le cmes(1,0))
        DYDI = (cmes(1,1)-cmes(0,1))/(cmes(1,0)-cmes(0,0))
        XWRK = EXWA(VSAV)
        YWRK = cmes(0,1) + DYDI*(XWRK-cmes(0,0)) ; Linear fit.
    sz = size(XWRK)
    jjj = sz(1) - 1
    endif

        xmx = max(XWRK(0:jjj),min=xmn)     ; Find range of abscissa.
VSAV = where(EXWA ge xmn and EXWA le xmx) ; VSAV is temporary array.
    J1 = VSAV(0)                          ; Determine starting and ending
        sz = size(VSAV)                   ; points for continuum vector
    J2 = VSAV(sz(1)-1)                     ; and store in J1 and J2.
    X0 = (xmx + xmn) / 2.0                 ; Adjust range to center on 0
    XWRK = XWRK(0:jjj) - X0               ; Also adjust size of vectors
    YWRK = YWRK(0:jjj)                   ; to remove trailing zeros.
    VSAV = YWRK                           ; Make extra copy of YWRK
    jjj = jjj ; - 1
    if (jjj lt 6) then begin               ; Not enough points to fit?
print,string(7B)
print,'Warning: Poorly Defined Continuum'
print,'Returning to Previous Level'
return
    endif

; ---- Find the best chi-squared fit -----

        EE = dblarr(16)                    ; To hold residuals of var. fits
    mtot = mcntrl.mtot+2
    if mtot gt 9 then mtot = 9
    for m = 2,mtot+1 do begin               ; m-1 order polynomials.
        MP1 = m + 1
        M2 = 2*m - 2
    BC(*) = 0.0
        CURFIT,XWRK,YWRK,AC,BC,MM,m,MP1,M2,1, jjj,2,0.0,0
    bcsav(m-1,*) = BC                      ; Save the fit parameters.
    EE(m-1) = total(YWRK(1:jjj-3)^2)      ; Sum of square of residuals.
        YWRK = VSAV                       ; Replace destroyed spectra.
    endfor
    ENE = float(jjj-3)/COHFAC              ; # of independent samples.
    EE = EE/COHFAC
    mcntrl.NNE = ENE                       ; Save for posterity.

    c_choice = string(bytarr(31,mcntrl.mtot+6))
    fdist = mcntrl.fdist
    m = 0

```

```

      mstp = 0
      f1w = where(ene ge fdist(0,1:17),fcnt)
      if ene eq fdist(0,fcnt) then begin
f1n = fdist(1,fcnt)
f2n = fdist(2,fcnt)
      endif ELSE begin
f1n = (ene-fdist(0,fcnt))/(fdist(0,fcnt+1)-fdist(0,fcnt))
f2n = f1n*(fdist(2,fcnt+1)-fdist(2,fcnt))+fdist(2,fcnt)
f1n = f1n*(fdist(1,fcnt+1)-fdist(1,fcnt))+fdist(1,fcnt)
      endelse
      f1p = string(format='(f6.2)',f1n)
      f2p = string(format='(f6.2)',f2n)
      for k=1,mcntrl.mtot do begin
fahd = (ENE-k-3)*(EE(k)-EE(k+1))/EE(k+1)
fahd2 = (ENE-k-4)*(EE(k)-EE(k+2))/(2.*EE(k+2))
fdsav(0,k) = fahd
fdsav(1,k) = fahd2
if mstp eq 0 then m = k
if (fahd le f1n) AND (fahd2 le f2n) then mstp = 1
fahd = string(format='(f6.2)',fahd)
fahd2 = string(format='(f6.2)',fahd2)
c_choice(k) = ' '+strtrim(string(k),2)+' '+fahd+f1p+fahd2+f2p
      endfor
      mstp = string(format='(i2)',m)

final:BC = bcsav(m,0:15) ; Again with the best poly.
MP1 = m + 1
M2 = 2*m - 2
XWRK = EXWA(J1:J2) - X0 ; Take only valid part of
YWRK = EXSP(J1:J2) ; wavelength and spectra.
jjj = J2 - J1 ; Total # of points in the fit.
CURFIT,XWRK,YWRK,AC,BC,MM,m+1,MP1,M2,1,jjj,3,0.0,0 ; Fit & make continuum.
CNT = float(YWRK(1:jjj-3)) ; YWRK replaced with continuum.
plot,EXWA,EXSP,/YNOZ ; Plot original spectrum.
oplot,exwa,exsp,psym=8 ; Overplot with large dots.
sz = size(CNT) ; How many elements are there?
xc = EXWA(J1:J1+sz(1)) ; Size wavelength accordingly.
oplot,xc,CNT ; Over plot fit continuum.
if csav eq 1 then oplot,xc,cntsv,linestyle=2 ; Overplot prev. continuum.
vtmp = where(XLIMIT gt 0,nxs)
for k=1,nxs do begin ; Show range of continuum fit.
ix = XLIMIT(k-1)
xyouts,ix,yout,':'+strtrim(k,2)
endfor

; --- Get the choice of Polynomical and Type of Integration -----

getch: print, ' ' & print, ' '
      mstr = string(format='(i2)',m)
if mstr eq mstp then begin
print,'The recommended Order is',mstp,' (highlighted), but you may select another.'
endif ELSE begin

```

```

        oplot,xxx(k,0:1),yyy,linestyle=1      ; Flag the plot too.
TVCRS,xxx(k,0),yhalf,/DATA                  ; Repeat Cursor command, insure.
print,' '
        print,'Cross-Hairs INPUT the left edge of the integration range.'
print,'          - Only horizontal position will be used.'
        CURSOR,IX,IY                          ; Read left (beginning) point.
        if kdev eq 'xterm' then wait,1        ; X windows too fast, slow it.
        PRINT,STRING(7B)                       ; Ring the bell.
        if lcs gt 1 then XI(k)=IX else XI=IX   ; Store start pt as array?
print,' '
        print,'Cross-Hairs INPUT the right edge of the integration range.'
print,'          - Only horizontal position will be used.'
        CURSOR,JX,JY                          ; Read right (final) point.
        if kdev eq 'xterm' then wait,1        ; X windows too fast, slow it.
        PRINT,STRING(7B)                       ; Ring the bell.
        if lcs gt 1 then XF(k)=JX else XF=JX   ; Store stop pt as array?

if (xxx(k,0) lt IX) OR (xxx(k,0) gt JX) then begin
    print,string(7B)
    print,'Warning: Integration Window does not contain'
    print,'          Center Wavelength defined before.'
    print,' '
    goto, setrng
    endif
    endfor
endif                                          ; End kchoice < 6.

        if kchoice eq 6 then begin
if mp.window eq 0 then begin                  ; kchoice=> Use Fixed Window
        vttmp = 0.0                            ; Has one already been selected?
        read,'Enter Full Width of window in km/s ',vttmp ; If not, read window & store.
        mp.window = vttmp
        endif
print,'Will use standard window of ',mp.window
vttmp = mp.window/2.9979e5                    ; Convert to unitless value.
        XF=EXWA(JE)+EXWA(JE)*vttmp/2.         ; Wavelength integration
        XI=EXWA(JE)-EXWA(JE)*vttmp/2.         ; window around line center.
endif

if kchoice eq 7 then mcntrl.intopt=0 else mcntrl.intopt=1 ; Use ALT_INT.

        JE = JE - J1                            ; Adjust line center index.
        sz = size(CNT)
        EXWA = EXWA(J1:J1+sz(1))                ; Stip out only necessary parts.
        EXSP = EXSP(J1:J1+sz(1))                ; Stip out only necessary parts.
        mcntrl.ESAV = sqrt(EE(m)/(ENE - float(m))) ; Save error of the fit.
;
;
DONE: PRINT,' '                               ; Let's get out of here.
        mp.cntrl = 1                            ; Set flag to OK
        IF (opt eq 2) THEN mp.cntrl = 0         ; or EXIT, No Measurement
        IF (iopt eq 4) THEN mp.cntrl = 0       ; or EXIT, No Measurement

```



```

print,'REMINDER: the recommended Order is',mstp
endelse
print,'To assist in other choices: X is the observational difference in the'
print,'reduced chi squares divided by the reduced chi square, which if larger'
print,'than the theoretical F(1,n) indicates that going to the next higher order'
print,'polynomial is justified statistically. Y is similar to X except it is'
print,'for comparison to F(2,n), an order that is 2 higher. The F Distributions'
print,'are at the 5% confidence level for:',fix(ene),' points.'

print,' '
print,'Select the Order of the Polynomial'
print,'      - Other Polynomials may be examined before deciding.'
print,'      - Selecting order',mstr,' implies use that polynomial.'
c_choice(0) = 'Order  X  F(1,n) Y  F(2,n)'
c_choice(6) = 'Order: '+mstr+', Predetermined window'
c_choice(7) = 'Order: '+mstr+', Use ALT_INT Routine'
if csav eq 0 then c_choice(8) = 'Toggle ON Overplot Last Cont.'
if csav eq 1 then c_choice(8) = 'Toggle OFF Overplot Last Cont.'
c_choice(9) = 'Restart Fitting'
c_choice(10) = 'Abort - RETURN to Prev. Level'
kchoice = wmenu(c_choice,title=0, init=m)
if kchoice lt 1 then goto, getch          ; Not valid - do it again.

; --- Implement the choice -----
if kchoice eq 10 then opt = 2              ; Abort - pretend set by USRMENU
if kchoice eq 10 then goto, DONE          ; then go to the bottom.
if csav eq 1 then cntsv = CNT
if kchoice eq 8 then begin                ; Toggle Overplot last continuum
vtmp = csav
if vtmp eq 1 then csav = 0
if vtmp eq 0 then csav = 1
if csav eq 1 then cntsv = CNT
goto, getch
endif
TVCRS,0.5,0.5,/NORMAL                    ; Put Cross-hairs on screen.
XI = min(XLIMIT)                          ; Set integration range to the
XF = max(XLIMIT)                          ; maximum allowed by spectrum.
if kchoice eq 9 then goto, START          ; kchoice => restart
if (kchoice ne m) and (kchoice lt 6) then begin ; Different order?
m = kchoice                               ; Set order to user choice.
goto, final                               ; Go display different order.
endif
if kchoice lt 6 then begin                ; kchoice => Settled on order #.
TVCRS,0.5,0.5,/NORMAL                    ; Put Cross hairs on screen.
if lcs gt 1 then XI=fltarr(lcs) else XI=0. ; Multiple features? If
if lcs gt 1 then XF=fltarr(lcs) else XF=0. ; so, get ready.
for k=0,lcs-1 do begin
setrng:  if lcs eq 1 then xxx(k,0:1) = EXWA(JE) ; Get feature center.
if lcs gt 1 then xxx(k,0:1) = EXWA(JE(k)) ; Do same if multiple ones
TVCRS,0.5,0.5,/NORMAL                    ; Put Cursors on the graph.
TVCRS,xxx(k,0),yhalf,/DATA              ; Put Cursors on the feature.

```

```

IF opt eq 6 THEN mp.cntrl = 2 ; or EXIT to NEXT spect.

OUT: print, ' '
    mp.poly = m ; Store in MSLAP structure.
    mcntrl.fin = fdsav(0,m)
    mcntrl.f2n = fdsav(1,m)

RETURN ; go back to DETAILS
END ; CONTIM ;
;
;
PRO STORE,WW,ML,dtl,mdata,mp,mcntrl,up
;***** STORE.PRO *****
;
; TO WRITE OUT THE FILES FOR DETAILS
; by Charles L. Joseph June 1982 May 1990
;
; ML maximum number of measurements that can be made.
;
;*****

iafs = mdata.iaf ; Get the Ion Codes.
iafs = where(iafs ne 0,sz) ; How much incoming data?
if sz eq 0 then goto, RET
IF (mcntrl.I+SZ) GE ML THEN BEGIN ; forced exit if too
    ERROR = 66 ; many lines. set flag
    PRINT,'Too many lines measured!',STRING(7B) ; tell them
    YN=''
    READ,'use <RETURN> to exit',YN
    GOTO,RET ; exit
END ; go back to DETAILS
IF mcntrl.I+10 GE ML THEN BEGIN
    PRINT,STRING(7B)
    PRINT,'Within 10 measurements of the maximum allowed'
    WAIT,4
END
IF mcntrl.I LT 0 THEN mcntrl.I = 0 ; can't have I < 0
openu,1,mp.STAR+'.DTL'
readu,1,dtl
close,1
for k=0,sz-1 do begin ; Add all new data to old
    sdata = mdata(k)
    dtl(mcntrl.I) = sdata
    print,FORMAT='("Storing measurement:",i3,3x,a10)', $
    mcntrl.I,string(sdata.el)
    mcntrl.I=mcntrl.I+1 ;increment masterindex
    IF mcntrl.I GE ML THEN mcntrl.I=ML-1 ; catch over run
endfor
;
; or write file & exit
; store data on disk
openu,1,mp.STAR+'.DTL'
writeu,1,dtl

```

```

CLOSE,1                                ; close file
;                                       ;or write file & exit
RET:                                     ;
RETURN                                  ; go back to DETAILS
END                                      ; to exit normally
;
;
;
pro fstore,tau,sdata,mp,mcntrl,up
;***** FSTORE.PRO *****
;
; To store the flux, wavelength, continuum, and optical-depth vectors.
; by Charles L. Joseph                    May 1990
;
;
;*****
; ----- Plot Optical Depths before storing them -----

aax = fltarr(5)                          ; Make usersymbol of an
aay = aax                                 ; arrow for L.L.
aax = [0.,0.,-0.25,0.,0.25]              ; Values for an arrow
aay = [0.,3.0,2.6,3.0,2.6]               ; (Lower Limit).

!y.margin(0) = 4.
v = 2.9979e5*(tau(0,*)- sdata.wl)/sdata.wl
asy = findgen(16)*(!PI*2/16.)
usersym,0.5*cos(asy),0.5*sin(asy),/FILL
plot,v,tau(2,*),xtitle='velocity',ytitle='log tau',psym=8, $
    yrange=[-2.5,1.5],ystyle=1
usersym,0.5*cos(asy),0.5*sin(asy)
oplot,v,tau(1,*),psym=8
oplot,v,tau(3,*),psym=8
usersym,aax,aay
ttmp = where(tau(2,*) gt 1.0,tcnt)        ; Which are limits?
if tcnt gt 1 then oplot,v(ttmp),tau(2,ttmp),psym=8 ; Over plot lower limits
if tcnt eq 1 then oplot,[v(ttmp),v(ttmp)],[tau(2,ttmp),tau(2,ttmp)],psym=8
ttmp = where(tau(3,*) gt 1.0,tcnt)        ; Which are limits?
if tcnt gt 1 then oplot,v(ttmp),tau(3,ttmp),psym=8 ; Error's Lower Limits.
if tcnt eq 1 then oplot,[v(ttmp),v(ttmp)],[tau(3,ttmp),tau(3,ttmp)],psym=8
fval = string(format='(f6.4)',sdata.f)
fval = '    f = '+fval
print,' '
print,'Log Tau Plot    Rest Wavelength:',sdata.wl,'    ',fval

kwtd = wmenu(['Copy?','Yes','No'],title=0, init=2)
if kwtd then begin
    plotconfig,1,' ',' ',-1,kdev,'      ; Config. for Hardcopy.
    asy = findgen(16)*(!PI*2/16.)
    usersym,0.8*cos(asy),0.8*sin(asy),/FILL
    plot,v,tau(2,*), xtitle='velocity',ytitle='log tau',psym=8, $
    yrange=[-2.5,1.5],ystyle=1

```

```

usersym,0.5*cos(asy),0.5*sin(asy)
oplot,v,tau(1,*),psym=8
oplot,v,tau(3,*),psym=8
usersym,aax,aay
ttmp = where(tau(2,*) gt 1.0,tcnt) ; Which are limits?
if tcnt gt 1 then oplot,v(ttmp),tau(2,ttmp),psym=8 ; Over plot lower limits
if tcnt eq 1 then oplot,[v(ttmp),v(ttmp)],[tau(2,ttmp),tau(2,ttmp)],psym=8
ttmp = where(tau(3,*) gt 1.0,tcnt) ; Which are limits?
if tcnt gt 1 then oplot,v(ttmp),tau(3,ttmp),psym=8 ; Error's Lower Limits.
if tcnt eq 1 then oplot,[v(ttmp),v(ttmp)],[tau(3,ttmp),tau(3,ttmp)],psym=8
xxo=0.0*(!x.crange(1)-!x.crange(0)) + !x.crange(0)
yyo=1.06*(!y.crange(1)-!y.crange(0)) + !y.crange(0)
xyouts,xxo,yyo,mp.FNAM
xxo=0.4*(!x.crange(1)-!x.crange(0)) + !x.crange(0)
xyouts,xxo,yyo,string(sdata.e1)+strtrim(string(sdata.w1),2)
xxo=0.7*(!x.crange(1)-!x.crange(0)) + !x.crange(0)
xyouts,xxo,yyo,mcntrl.date
plotconfig,-1,' ',' ',-1,kdev,' ; Send plot & -> term.
endif

; ----- First Retrieve & Update Header Information -----

as = assoc(5,fltarr(200)) ; Association variables for
tas = assoc(5,fltarr(2,600)) ; random disk access.
close,5
openr,5,mp.STAR+'.TAU', ERROR = errtst ; Test if file exists.
close,5
if errtst ne 0 then begin ; No file so create one.
  openw,5,mp.STAR+'.TAU' ; Open a new file.
  for k=0,5 do as(k) = fltarr(200) ; Make 6 dummy records.
  close,5 ; Now close it.
endif
openu,5,mp.STAR+'.TAU' ; Open file for Update & get
iafs = as(0) ; 1st record - species codes
rwav = as(1) ; Get prev. rest wavelengths
bgs = as(2) ; Get previous BG levels.
bgers = as(3) ; Get previous BG errors.
dates = as(4) ; Get previous date codes.
lwfs = as(5) ; Get prev. log f-lambdas.

itau = TOTAL(iafs ne 0) ; Total number before?

iafs(itau) = long(sdata.iaf) ; Add new species code.
print,iafs(itau)
if iaifs(itau) lt 0 then stop
rwav(itau) = sdata.wl ; Add current lab wavelength
bgs(itau) = mp.bg ; Add current BG level.
bgers(itau) = mp.bgerr ; Add current BG error.
dates(itau) = 910112 ; Add current date.
lwfs(itau) = alog10(sdata.wl*sdata.f) ; Add new log f-lambda.

; ----- Next Store, Header, Optical Depths + Spectra -----

```

```

tautmp = fltarr(2,600)
sz = size(tau)
if sz(0) le 0 then begin
  print,string(7B),'
  print,'WARNING: No Optical Depth Data"
  wait,2
  close,5
  return
endif
sz = sz(2)

as(0) = iafs
as(1) = rwav
as(2) = bgs
as(3) = bgers
as(4) = dates
as(5) = lwfs

tautmp(0:1,0:sz-1) = tau(0:1,*)
tas(3*itau+1)      = tautmp
tautmp(0:1,0:sz-1) = tau(2:3,*)
tas(3*itau+2)      = tautmp
tautmp(0:1,0:sz-1) = tau(4:5,*)
tas(3*itau+3)      = tautmp

close,5
return
end
;
;
;
PRO EXPAND,JO,JS,WA,WS,JT,JE,J,nside
;***** EXPAND.PRO *****
;
;   TO EXPAND THE FLUX AND WAVELENGTH VECTORS.
;   by Charles L. Joseph                               Sept 1980.
;
;   JO      flux vector
;   JS      expanded flux vector
;   WA      wavelength vector
;   WS      expanded wavelength vector
;   JT      number of points in expanded vectors
;   JE      central index of vectors ( expanded )
;   J       index of cursor position
;
;
;*****
;
nsave = nside
NPTS=TOTAL(WA GT WA(0))
JT = 2*nside + 1

```

```

; Make temporary array.
; Find size of incoming data
; If no incoming, get out.
; Ring the alarm BELL.

; 2 seconds to see it.
; Close the file.
; No saving it, return.

; Change info. to a scalar.

; Store updates of record-
; keeping and header info.
; back to the disk file.

; Stuff Wavelength & TAUs(-)
; and write to disk.
; Stuff TAU's and TAUs(+)
; and write to disk.
; Stuff Spectrum & Continuum
; and write to disk.

; Close file and return.

; Save the request.
; # of points in vector
; # of expansion points

```

```

if (NPTS lt JT) then nside = FIX(NPTS/2) - 1 ; Adequate # of points?
if J lt nside then jfact = nside - J else jfact = 0 ; Too close to lft edge?
if (NPTS-J) lt nside then jfact = NPTS - J - nside ; Too close to rt edge?
JS = JO(J+jfact-nside:J+jfact+nside)
WS = WA(J+jfact-nside:J+jfact+nside)
JE = nside + 1 - jfact ; Index of line center.
nside = nsave ; Restore the request.
RETURN
END ;
;
;
; *****
;
; MAIN PROGRAM
;
; by Charles L. Joseph May 12, 1990
;
; *****
;
; date='Fri Apr 13 00:00:01 1990' ;
date = systime(0) ; Get time and date
date = strmid(date,4,7)+strmid(date,20,4) ; Strip out date
up = fltarr(30)
ST: for k=1,13 do print, ' '
print,date ;
NUM = 0 ; reset NUM
libr = '/u/clj/mslapdir' ; Defines the MSLAP lib.
spawn,'head -26 '+libr+'/main.mslap' ; print 1st 31 lines.
for k=1,3 do print, ' '
READ,'Which option would you like ? ',NUM ; what to do
IF NUM EQ 10 THEN GOTO,OUT ; to get out
;
;
; DETAILS OR POSTO
;
;
IF NUM EQ 1 THEN DETAILS,0,date,libr,up ; run DETAILS no tau's
IF NUM EQ 2 THEN DETAILS,1,date,libr,up ; run DETAILS with tau's
IF (NUM ge 3) AND (NUM le 6) THEN BEGIN ; POSTO, EDATDTL, COGS
;
;
GSTAR: STAR=' ' ; set up loop
READ,'What is the file name, created in Option #1 or #2 ?',STAR
IF STAR EQ 'H' THEN BEGIN ; Want help?
SPAWN,'ls *.DTL' ; Give a listing and
GOTO,GSTAR ; go back try again.
END ;
if NUM ne 5 then openr,1,STAR+'.DTL', ERROR = errtst ; See if file exist.
if NUM eq 5 then openr,1,STAR+'.TAU', ERROR = errtst ; See if file exist.
close,1 ; Close again.
if errtst ne 0 then begin ; Oops no file
YNO=' ' ;

```

```

if NUM eq 4 then read,'Create Dummy File?',YNO ; EDATDTL? Use dummy?
  if (YNO eq 'Y') OR (YNO eq 'y') then goto,CHOICES
  YNO='' ; Try another file?
  READ,'Would you like to try another FILE?',YNO
  if (YNO eq 'Y') OR (YNO eq 'y') then goto,GSTAR
  GOTO,ST ; else goto start
END ;
END ;
CHOICES: ;
  IF NUM EQ 3 THEN POSTO,STAR ; run POSTO
  IF NUM EQ 4 THEN edatdtl,STAR ; run EDATDTL
; ;
  IF NUM EQ 5 THEN mantau,STAR,date ;
; ;
  IF NUM EQ 6 THEN cog,STAR,date,libr ;
GOTO,ST ; go back to start
OUT: ; ; tty mode
END ;

```

```

; *****
; This file contains many procedures or indirect calls to compile procedures
; used by M.S.L.A.P. These contain some of the standard as well as the
; dummy routines that may be over layed by editing the mslap.pro routine
; in the user's local directory.
;
; Charles L. Joseph December 24, 1990
;
; *****
;
@/u/clj/mslapdir/mantau
@/u/clj/mslapdir/intgrt.pro
@/u/clj/mslapdir/edatdtl
@/u/clj/mslapdir/posto
@/u/clj/mslapdir/cog
@/u/clj/mslapdir/plotconfig
@/u/clj/mslapdir/compare.pro
pro dget1,FNAME,w,spect,q,ihdr,ID,mp,up
  mp.dget = 0 ; These are dummy routines
  ID = 'NOT Being Used' ; which reserve slots for
return ; real ones that will be
end ; plugged in.
pro dget2,FNAME,w,spect,q,ihdr,ID,mp,up
  mp.dget = 0
  ID = 'NOT Being Used'
return
end
pro dget3,FNAME,w,spect,q,ihdr,ID,mp,up
  mp.dget = 0
  ID = 'NOT Being Used'
return

```

```

end
pro dget4,FNAME,w,spect,q,ihdr,ID,mp,up
  mp.dget = 0
  ID = 'NOT Being Used'
return
end
pro dget5,FNAME,w,spect,q,ihdr,ID,mp,up
  mp.dget = 0
  ID = 'NOT Being Used'
return
end

```

```

;
;
; *****
; PROGRAM CURFIT -calculates ploynomial fits for CONTIM.
; By Edward B. Jenkins
; Translated to IDL and Edited by Charles L. Joseph May 1990
;
; ORDER OF POLYNOM. = M ; MP1 = M + 1 AND M2 = 2*M - 2
; ARRAY WILL BE PROCESSED FROM INDEX "NS" TO INDEX "N"
; IF SOME POSITIONS IN THE ARRAY BETWEEN INDICES NS AND N ARE TO BE IGNORED,
; JUST SET THE APPROPRIATE Y VALUES GREATER THAN 1.D30
; MODE = 0 JUST EVALUATES ANSWER VECTOR B
; MODE = 1 EVALUATES B AND REPLACES Y BY BEST FIT CURVE
; MODE = 2 EVALUATES B AND REPLACES Y BY RESIDUALS
; MODE = 3 COMPUTES BEST FIT CURVE FROM EXISTING X AND B ARRAYS
; XMUL = 0.DO TELLS SUBROUTINE TO REFERENCE X ARRAY; OTHERWISE X IS IGNORED
; LOGIND = 0 DOES ORDINARY FIT; IF LOGIND .GT. 0, COEFF'S ARE IN TERMS OF LOG Y
; IMPLICIT REAL*8(A-H,O-Z)
; DIMENSION Y(N),A(M,MP1),B(M2),MM(M),X(N)
; *****

```

```

PRO CURFIT,X,Y,A,B,MM,M,MP1,M2,NS,N,MODE,XMUL,LOGIND
  IF MODE EQ 3 then GOTO,COMP ; Go compute polynom.
  A = 0.0*A ; Zero A and B arrays
  B = 0.0*B
  A(0,0) = N - NS + 1 ; A(0,0)= # of points
  FOR I = NS,N DO BEGIN
    IF Y(I-1) GE 1.e70 THEN A(0,0) = A(0,0) - 1. ELSE BEGIN
      IF XMUL EQ 0 THEN XP = X(I-1) ELSE XP = double(I)*XMUL
      XX = XP
      B(0) = B(0) + XP
      IF LOGIND GT 0 THEN Y(I-1) = double(alog(Y(I-1)))
      A(0,MP1-1) = A(0,MP1-1) + Y(I-1)
      FOR J = 2,M DO BEGIN
        A(J-1,MP1-1) = A(J-1,MP1-1) + Y(I-1)*XP
        XP = XP*XX
        B(J-1) = B(J-1) + XP
      ENDFOR
      IF LOGIND GT 0 THEN Y(I-1) = double(EXP(Y(I-1)))
      FOR J= MP1,M2 DO BEGIN

```



```

        XP = XP*XX
        B(J-1) = B(J-1) + XP
    ENDFOR
  ENDElse
ENDFOR ; 6 CONTINUE
MS = 2
FOR I=1,M DO BEGIN
  FOR J = MS,M DO BEGIN
    IPJ = I + J - 2
    A(I-1,J-1) = B(IPJ-1)
  ENDFOR
MS = 1
ENDFOR
EQSOL,A,M,MP1,B,MM ; CALL EQSOL
IF MODE EQ 0 THEN RETURN
COMP: FOR I = MS,N-1 DO BEGIN
  IF XMUL EQ 0 THEN XP = X(I-1) ELSE XP = double(I-1)*XMUL
  XX = XP
  YCOMP = B(0)
  FOR J = 2,M DO BEGIN
    YCOMP = YCOMP + B(J-1)*XP
    XP = XP*XX
  ENDFOR
  IF LOGIND GT 0 THEN YCOMP = double(EXP(YCOMP))
  IF (MODE EQ 2) AND (Y(I-1) LT 1.D70) THEN Y(I-1) = Y(I-1) - YCOMP
  IF MODE NE 2 THEN Y(I-1) = YCOMP
ENDFOR
RETURN
END

```

```

; *****
;
; PROGRAM EQSOL - To solve Differential Equation for CURFIT.
; By Edward B. Jenkins
; Translated to IDL and Edited by Charles L. Joseph May 1990
;
; *****

```

```

pro eqsol,a,n,nn,x,m
; a = fltarr(n,nn)
; x = fltarr(n)
; m = intarr(n)
for i = 1,n do begin
  m(i-1) = 1
  amax = a(i-1,0)
  for j = 2,n do begin
if ((abs(a(i-1,j-1))-abs(amax)) gt 0) then begin
  amax = a(i-1,j-1)
  m(i-1) = j
  endif
  endfor
  for j = 1,nn do begin

```

```

        if(amax eq 0) then print,'about to divide by amax=0 in EQSOL'
        a(i-1,j-1) = a(i-1,j-1)/amax
    endfor
    for ip = 1,n do begin
    if (ip - 1) ne 0 then begin
        mmm = m(i-1)
        zmult = a(ip-1,mmm-1)
        for j=1,nn do if (j-mmm) eq 0 then a(ip-1,j-1) = 0 $
            else a(ip-1,j-1) = a(ip-1,j-1)-zmult*a(i-1,j-1)
        endif
    endfor
    endfor
for i = 1,n do begin
    mmm = m(i-1)
    x(mmm-1) = a(i-1,nn-1)
endfor

return
end
; -----
; PROGRAM UCURSOR to provide continuous readout of cursor position until
; the mouse is clicked. The left and middle mouse buttons return to calling
; routine, while the right button brings up a menu of options. If the
; graphics device is not a SUN, control information is can be passed by the
; key used to input the cursor location.
;
; cflg = 0 use Main UserMENU. Selection is returned in opt parameter.
; cflg = 1 use Continuum MENU first. Selection is returned in iopt param.
; cflg < 0 NO Menu, reserve right mouse button for other purposes.
; Coordinates are returned in IX and IY.
; See main.mslap for definitions of the mp structure.
;
; By Charles L. Joseph          6/4/90          Latest Revision: 1/7/91
; -----

pro ucursor,IX,IY,opt,iopt,cflg,mp

    vflg = 0

START: ;

    opt = 9999
    iopt = 9999
    kdev = getenv('TERM')
    if kdev eq 'vt300' then kdev = 'xterm'
    if (kdev eq 'xterm') OR (kdev eq 'sun') then begin
        xout1 = 0.0*(!x.crange(1)-!x.crange(0)) + !x.crange(0)
        xout2 = 0.5*(!x.crange(1)-!x.crange(0)) + !x.crange(0)
        youta = 0.5*(!y.crange(1)-!y.crange(0)) + !y.crange(0)
        tst = 0
        while (tst ne 1) do begin
            cursor,IX,IY,/CHANGE
        endwhile
    endif
; Continuous readout of cursor
; coordintes.

```

```

        if !ERR eq 1 then tst = 1           ; if true - left mouse
        if !ERR eq 2 then tst = 1           ; if true - middle mouse
        if !ERR eq 4 then tst = 1           ; if true - right mouse
        wset,1                               ; Set to cursor printing window
        erase                                 ; Clear and print new coords.
if vflg then xcrd=2.9979e5*(IX-wavrest)/wavrest ELSE xcrd = IX
        xyouts,xout1,youta,string(format='(f9.3)',xcrd)
        if (youta lt 600) then xyouts,xout2,youta,string(format='(f9.3)',IY)
        if (youta ge 600) then xyouts,xout2,youta,string(format='(f9.0)',IY)
        wset,0                               ; Return to plotting window.
    endwhile
    if kdev eq 'xterm' then wait,1
    if !ERR eq 2 then return                 ; Exit this routine if middle
    if !ERR eq 1 then return                 ; or left mouse used, otherwise
; a MENU is wanted.

; ----- OLD Tektronics type Cursors -----

endif else begin
    CURSOR,IX,IY                             ; what to do?
    PRINT,STRING(7B)                         ; 4025
endif

; ----- Get Various UserMenu Options -----

if cflg lt 0 then return                     ; No Menu Options Wanted
; under any circumstances.

; If cursor reading is out side of plot area, or the right mouse button used,
; then set menuflg=1 to get a menu. If calling routine is the Continuum
; fitting routine (cflg=1), then call special menu defined below. Otherwise
; (cflg=0), call Main UserMenu.

menuflg = 0
if((IX lt !x.crange(0)) or (IX gt !x.crange(1))) then menuflg = 1
if((IY lt !y.crange(0)) or (IY gt !y.crange(1))) then menuflg = 1
if (!ERR eq 4) and (cflg eq 0) then usrmenu,opt ; Not in Continuum Routine.
if (!ERR eq 4) and (cflg eq 1) then menuflg = 1 ; In Continuum Routine.

if menuflg then begin                       ; First MENU for Continuum
    c_opts = string(bytarr(22,7))           ; routine. Option 4 calls
    c_opts(0) = 'Continuum Options'         ; Main UserMenu.
    c_opts(1) = 'Take No Action'
    c_opts(2) = 'Descrete Cont. Points'
    c_opts(3) = 'Define Profile Cntr(s)'
    c_opts(4) = 'Change EXPANSION size'
    c_opts(5) = 'Restart Continuum Fit'
    c_opts(6) = 'Call MAIN userMENU'
    iopt = wmenu(c_opts, title=0, init=1)   ; Get iopt option.
    if iopt eq 6 then usrmenu,opt
endif

```

```

if (opt eq 10) OR (opt eq 12) then begin           ; Set up to display velocity
  print,string(7B),' '
  if opt eq 10 then begin
    print,'Find Wavelength of Interest and Click any Mouse'
    print,'X-Coordinte will then read in velocity'
  endif
  if opt eq 12 then begin
print,'Locate Background Level and Click LEFT Mouse -- then'
print,'offset vertically by 1 standard deviation and Click again.'
print,'If you wish to enter the numbers manually, Click RIGHT Mouse.'
  endif
  tst = 0
  while (tst ne 1) do begin                       ; Continuously read wave-
    cursor,IX,IY,/CHANGE                          ; length position of cursor.
    if !ERR eq 1 then tst = 1
    if !ERR eq 2 then tst = 1                     ; Cursor reading provides
    if !ERR eq 4 then tst = 1                     ; reference wavelength.
    wset,1
    erase
    xyouts,xout1,youta,string(format='(f9.3)',IX)
    xyouts,xout2,youta,string(format='(f9.3)',IY)
    wset,0
  endwhile
  wavrest = IX                                   ; Start this routine over,
  vflg = 1                                       ; using velocity instead of
  if opt eq 10 then goto, START                  ; wavelength.
  if opt eq 12 then begin
if !ERR eq 4 then begin
  print,' ',string(7B)
  Print,'Move cursor into COMMAND window for input.'
  Read,'Enter background: ',bg
  mp.bg = bg
  read,'Enter uncertainty: ',bg
  mp.bgerr = bg
  endif ELSE begin
print,''
print,'Now locate a 1 sigma deviation.'
mp.bg = IY
cursor,IX,IY
if IY gt mp.bg THEN mp.bgerr=IY-mp.bg ELSE mp.bgerr=mp.bg-IY
endelse
  endif
endif
if opt eq 11 then begin
  print,' ',string(7B)
  Print,'Move cursor into COMMAND window for input of New Plot Parameters.'
  print,' '
  read,'Enter min and max values for the X axis: ',x1,x2
  read,'Enter min and max values for the Y axis: ',y1,y2
  !x.range = [ x1, x2 ]
  !y.range = [ y1, y2 ]
endif

```

```

RETURN
END ; UCURSOR
;
;
;
PRO dataget,WA,SPECT,JJ,IHDR,FF,mp,mcntrl,up
;
;***** dataget.pro *****
;
;   To get the Spectrum Data.  -- formerly inside GRAPHS.PRO
;   This routine interfaces the data-getting routines of DGETS to DETAILS.
;   By Charles L. Joseph           May 1990
;
;   SPECT      flux vector
;   WA         wavelength vector
;   IHDR       header vector
;   FF         data type flag
;   JJ         data quality vector
;
;*****
COMMON BLOCK1,DTY,SMO ; get common

FNAME = mp.FNAM
IF mp.cntrl EQ -1 THEN begin ; 1st time through
START: FNAME = ' ' ; create blank name
for k=0,1 do print, ' '
print,'Enter complete INPUT Data Filename' ;
read,'Including the path if necessary ',FNAME; read file name
if (FNAME eq 'end') OR (FNAME eq 'END') then begin
mp.cntrl = -99
return
endif
if FNAME eq 'H' then begin
for k=0,2 do print, ' '
spawn,'ls'
for k=0,2 do print, ' '
print,'>>>> Note: entering the word "end" exits <<<<'
goto,START
endif
close,1 ; close logical unit
openr,1,FNAME, ERROR = errtst ; check that it exists
close,1
if errtst ne 0 then begin ; a problem if nonzero
print, ' ',string(7B) ; send beep
print,'Warning: Unable to read file' ; print warning
yno = ' '
read,'Try to input again? ',yno
if (yno eq '1') OR (yno eq 'y') then goto,START
mp.cntrl = -99
return ; return to MSLAP
endif
endif

```

```

COHFIND: cohfac = 0
  print,string(7B),' '
  print,' '
  print,'IMPORTANT: The coherence length is used to calculate ALL uncertainties.'
  print,'It is the number of pixels influencing the value in a given pixel.'
  print,'Thus, the coherence length must be 1 or greater. For example, data
  print,'smoothed by a 3-point Running Box Car has a coherence length of 3.'
  print,string(7B),' '
  read,'What is the coherence length of the present data?',cohfac
  mp.cohfac = cohfac
  if mp.cohfac lt 1 then goto, COHFIND
  mp.bg = 0.0 ; Assume background &
  mp.bgerr = 0.0 ; error initially.
endif
;
mp.FNAM = FNAME
IF FF EQ 1 THEN dget1,FNAME,WA,SPECT,JJ,IHDR,ID,mp,up
IF FF EQ 2 THEN dget2,FNAME,WA,SPECT,JJ,IHDR,ID,mp,up
IF FF EQ 3 THEN dget3,FNAME,WA,SPECT,JJ,IHDR,ID,mp,up
IF FF EQ 4 THEN dget4,FNAME,WA,SPECT,JJ,IHDR,ID,mp,up
IF FF EQ 5 THEN dget5,FNAME,WA,SPECT,JJ,IHDR,ID,mp,up
if (FF lt 1) OR (FF gt 5) then print,'Invalid File Option'
print,' '
print,'New Data Have Been Read -- Data Getting Option:',fix(FF)

if mp.cntrl eq -1 then begin ; Determine tolerance for
  sz = size(WA) ; finding entry in ULUT.
  if sz(0) eq 0 then mp.cntrl = -99 ; Just in case problem with DGET,
  if mp.cntrl eq -99 then begin
    print,string(7B),'WARNING: Data-Getting Routine returned a vector'
    print,' that has ZERO elements! -- Returning to MENU'
    wait,3
  endif
  if sz(0) eq 0 then return ; indicate an abort & exit.
  wavetol = 10*(WA(sz(1)-1)-WA(0))/sz(1) ; Tolerance = +/- 10 pixels, but
  if wavetol lt 0.5 then wavetol = 0.5 ; never less than 0.5 Angstrom.
  mcntrl.wtol = wavetol ; Save for COMPARE.
endif

print,' ',string(7B)
print,' -----'
print,' | The background uncertainty frequently has a major impact on the |'
print,' | uncertainty of the various measurements. |'
print,' | Current background is:',mp.bg,' with an error:',mp.bgerr,' |'
print,' | Use the RIGHT Mouse Button if these are unsatisfactory. |'
print,' | Note: all graphical input is performed by placing the cross-hairs |'
print,' | at the point of interest and pressing a mouse button. |'
print,' -----'
;
return
end
; ##### END of MASTER.AUX Package #####

```

```

PRO COMPARE,WV,RS,sdata,mp,mcntrl
;***** COMPARE.PRO *****
;
;   TO FIND INTERSTELLAR LINES IN VARIOUS LOOK UP TABLES
;
;       WV      wavelength of feature to be identified
;       RS      red shift (Doppler Velocity Features)
;       sdata   data structure for single measurement
;       mp      MSLAPparameter structure
;
;   Charles L. Joseph           3/28/79           Latest Rev.: 4/5/90
;
;*****
;
WF=WV/(RS+1.) ; apply the red shift
nullid = { notdat, el: bytarr(15), iaf: long(0), wl: 0.0, f: 0.0 }
uid1 = nullid ; Make a "single" entry.
mids = replicate({ notdat }, 2000) ; Make MSLAP LUT array.
uids = replicate({ notdat }, 200) ; Do the same for user's.
wrk = replicate({ notdat }, 10) ; Make small working
wrk2 = replicate({ notdat }, 10) ; structures.
wrk2.wl = -32000. ; Unused ones to odd val.
close,3
if (mp.DTY eq 1) OR (mp.DTY eq 3) then $
  openr,3,mcntrl.lib+'tabdata/ilut.tab' ; Open LUT file.
if mp.DTY eq 2 then openr,3,mcntrl.lib+'tabdata/mlut.tab' ; Open LUT file.
if mp.DTY eq 4 then openr,3,mcntrl.lib+'tabdata/qlut.tab' ; Open LUT file.
readu,3,mids ; Read LUT structure.
CLOSE,3
OPENR,3,'ulut.tab', ERROR = errtst ; Get User LookUpTable
if errtst eq 0 then readu,3,uids ; if one exists.
wtst = where(mids.wl le WF, N) ; Which is best entry?
if N le 2 then N = 2
wrk2(0:4) = mids(N-2:N+2) ; Into working structure.

if mp.DTY eq 3 then begin ; If two tables are to be
  close,3 ; used, replace strange
  openr,3,mcntrl.lib+'tabdata/mlut.tab' ; values with real ones.
  readu,3,mids
  wtst = where(mids.wl le WF, N) ; Which is best entry?
  wrk2(5:9) = mids(N-2:N+2) ; Stuff into working str.
endif

W = wrk2.wl
for k=0,9 do begin ; Order combined small
wtst = min(W,ibest) ; structure according to
  wrk(k) = wrk2(ibest) ; wavelength.
W(ibest) = 32000.
endfor

W = wrk.wl
W = abs(W-WF) ; Find absolute diffs.

```

```

      wtst = min(W,ibest) ; ibest is array index.
      if ibest lt 1 then ibest=1
OK: print, ' '
;
; ----- Set up to show choices found in the tables -----

nuids = 0
USERIDS,WV,WC,WF,nuids,n_names,ns,uid1,uids,mcntrl ; Any User ID's?
n_names(0) = 'Which Entry?'
n_names(1) = 'None - Return to Spectra'
; found in user Lookup.

for k = 0,2 do begin
  uid1 = wrk(ibest-1+k) ; Get next entry.
  DD = (WV-uid1.wl)/uid1.wl*299792.5 ; Convert to velocity.
  AION = string(uid1.el) + ' .....'
  AION = strmid(AION,0,15)
  wk = string(format='(f9.3)',uid1.wl)
  fk = string(format='(f7.4)',uid1.f)
  n_names(k+2) = wk+ ' '+string(format='(f6.1)',DD)+' '+AION+' f='+fk
endfor
n_names(nuids+5) = 'Input Identity' ; nuids = # of lines
print,'Select Look-Up Table Entry -- Note: None is an option.'
print,' -- 2nd number is the Doppler velocity (km/s).'
WHICH = wmenu(n_names, title=0,init=3)

NNN = WHICH - 3
mp.cntrl = 1
IF mp.cntrl EQ 0 THEN NNN=0 ;
if (WHICH le 1) then mp.cntrl = 0

; ----- Three sources of input -----

if(WHICH EQ (nuids+5)) then begin ; ===== ; Input new ULUT entry.
  NNN = 999
  print,'>>>> Requesting ID <<<<'
  USERIDS,WV,WC,WF,NNN,n_names,ns,uid1,uids,mcntrl
  sdata.el = uid1.el ; Put codes into struct-
  sdata.iaf = uid1.iaf ; for MSLAP measurements.
  sdata.wl = uid1.wl
  sdata.f = uid1.f
  WL=uid1.wl
  AION = string(uid1.el(0:1))
  MUL = 0
  NNN = 999
endif
IF ((NNN GE 2) AND (NNN le 100)) THEN BEGIN ; =====; Get ID from user table
  N = ns(WHICH)
  print,'Using ULUT entry #:',N
  uid1 = uids(N)
  sdata.el = uid1.el
  sdata.iaf = uid1.iaf
  sdata.f = uid1.f

```



```

sdata.wl = uid1.wl
AION = string(uid1.el(0:1))
MUL=0
CLOSE,3
END
if (NWN lt 2) then begin ; ===== ; Get from stndrd LUT.
  ibest = ibest + WHICH - 3 ; Get requested index.
  if ibest lt 0 then ibest = 0 ; Make sure range is OK.
  uid1 = wrk(ibest)
  sdata.el = uid1.el
  sdata.iaf = uid1.iaf ; Get ionization state.
  sdata.wl = uid1.wl
  sdata.f = uid1.f
END ;
if sdata.iaf lt 0 then print,'sdata.iaf:',sdata.iaf
if sdata.iaf lt 0 then stop
CLOSE,2 ; Close look up table.

RETURN ; Go back to DETAILS.
END ;
;
;
;
; *****
;
; PROCEDURE USERIDS to allow the user to identify lines
;
; by Charles L Joseph 13-Aug-84
;
; *****
;
;
PRO USERIDS,WW,NC,WF,nuids,n_names,ns,uid1,uids,mcntrl
  ncnt = where(uids.wl ne 0., NC) ; get # of lines
  W = uids.wl
  if (nuids gt 100) then goto, IDIT ; Input identity?
; ----- This part searches for existing entries -----
  i_names = string(bytarr(26,26)) ; Big enough to hold max
  ns = intarr(26) ; # of Menu selections.
  for K=0,99 do begin
    IF ABS(W(K)-WF) LT mcntrl.wtol THEN BEGIN ; Close to that observed?
  uid1 = uids(K)
    DB=(WW-W(K))/W(K)*299792.5 ; Convert to velocity.
  AION = string(uid1.el)+' ..... ' ; Start building string
  AION = strmid(AION,0,15) ; for the menu selection.
    wk = string(format='(f9.3)',W(K))
    i_names(nuids+5) = wk+' '+string(format='(f6.1)',DB)
    fk = string(format='(f7.4)',uid1.f)
  i_names(nuids+5) = i_names(nuids+5)+' '+AION+' f='+fk
    ns(nuids+5) = K ; Store ULUT address.
    nuids = nuids + 1 ; Increment the counter.

```

```

        if (nuids gt 20) then goto, OUT ; Can only hold 20 entries
    END
endfor
OUT: n_names = i_names(0:nuids+5) ; Reduce Menu Selections
close,3 ; to the correct size.
RETURN

; ----- New entries to the User LookUpTable (LUT) are made here -----

IDIT: print, ' '
print, 'Enter Complete Species Name - include ionization and fine structure.'
print, 'Do not leave any preceding blanks. Enter for example: "Fe II*" for'
print, 'iron singly ionized and excited Fine Structure. This entry will only'
print, 'be used as an element identifier and print label. The ionization will'
print, 'have to be entered later.'
vdum = ''
read, vdum
elm = vdum
uid1.el = 0*uid1.el
elm = strtrim(elm,1)+' ..... '
uid1.el = byte(strmid(elm,0,15))
vdum = ''
READ, 'What is the IONIZATION state [ 2 = II 4 = IV ] ?', vdum
IA = fix(vdum)
vdum = ''
if IA le 0 then begin ; Encode as molecule or
    print, ' '
    print, 'Species is assumed to be a molecule'
    read, 'What is the vibrational level?', vdum
    iaf = 10*fix(vdum)
    vdum = ''
    read, 'What is the rotation level? (9 Max)', vdum
    iaf = iaf + fix(vdum)
    vdum = strmid(elm,0,2)
    if (vdum eq 'H2') OR (vdum eq 'HD') then begin
read, 'Enter 1000 if this is part of the Werner system', ly
if ly eq 1000 then iaf = iaf + 1000
endif
endif ELSE begin ; else encode as ion.
    READ, 'What is the FINE STRUCTURE state [ 3 = *** ] ?', vdum
    FS = fix(vdum)
    iaf = FIX(IA*10 + FS) ; Ionization & Fine Str.
endif
READ, 'What is the Rest WAVELENGTH of the line ?', vdum
uid1.wl = float(vdum) ; Rest Wavelength.
READ, 'What is the OSCILLATOR strength of the line ?', vdum
uid1.f = float(vdum) ; Oscillator Strength.

tst = uid1.el(0:1) ; ----- Encode as Periodic Table & Molecules
els = 'H HeLiBeB C N O F NeNaMgAlSiP S ClArK CaScTiV CrMnFeCoNiCuZn'
els = els+'H2H2HDHDCOHCNC2OHU U ' ; Elements + Molecules
els = byte(els)

```

```

j = 0
atst = 0
while atst eq 0 do begin
atst = ((tst(0) eq els(j)) AND (tst(1) eq els(j+1)))
j = j + 2
if j gt 78 then goto,exitst
endwhile
exitst: uid1.iaf = long(iaf) + long(1000)*(j/2) ; Include ionization.
; -----
CLOSE,2 ; close look up table

ID = ''
READ,'Enter element permanently in table?',ID
IF (ID EQ 'y') OR (ID eq 'Y') THEN begin
uids(NC) = uid1
close,3
openr,3,'ulut.tab', ERROR = errtst ; See if file exists and
close,3 ; make one if necessary.
if errtst eq 0 then openu,3,'ulut.tab' ELSE openw,3,'ulut.tab'
writeu,3,uids ; Update ULUT.
endif
CLOSE,3
RETURN
END

PRO POSTO,STAR
;***** POSTO.PRO *****
;
; To list the measured lines from DETAILS and create ASCII files of
; this data.
; by Charles L. Joseph December 19, 1980
; Latest Revision: January 7, 1991
; STAR file name
;
;*****
LPF=0 ; flag to pause for LP
STRT:close,1
OPENU,1,STAR+'.DTL' ; Open the .DTL file.
nulld = { noda, el: bytarr(15), iaf: long(0), wl: 0., f: 0., owl: 0., eqw: 0., $
me: 0., fm: 0., fme: 0., sm: 0., sme: 0., com: bytarr(10), up: fltarr(30) }
dtl = replicate({ noda }, 200) ; Make array for storage
sdata = nulld ; Make working copy
readu,1,dtl ; Read the data.
close,1
sz = where(dtl.iaf ne 0.) ; Get # of measurements
sz = size(sz)
if (sz(0) eq 0) then NTL = 0 else NTL = sz(1) ; & store in NTL.
IF NTL EQ 0 THEN GOTO,EXIT ; return if none
LOOP:WTD=0 ; set up loop
spawn,'clear'
for k=0,3 do print,' '
print,' ===== Program POSTO.PRO ====='

```

```

print, ' '
print, 'It is best to resize this window to the full screen while in POSTO.'
print, 'This can be done now and a reminder to unzoom the window will appear'
print, 'at the appropriate time.'
for k=0,4 do print, ' '
PRINT, ' Options to MANIPULATE TABLED DATA:'
PRINT, ' 1: Reorder .DTL file by Laboratory Wavelength'
PRINT, ' 2: Reorder .DTL file by Observed Wavelength'
PRINT, ' 3: Reorder .DTL file by Ion (Use Opt 1 or 2 first)'
PRINT, ' 4: List existing .DTL file'
PRINT, ' 5: Create Customized Table in ASCII format'
PRINT, '       Note: all .DTL files should be reordered'
PRINT, '       in the same fashion first.'
PRINT, ' 10: EXIT - Back to MSLAP'
print, ' '
READ, ' Which do you want ?',WTD ; which one

IF WTD EQ 10 THEN GOTO,EXIT
LPF=0 ; lots of time for LP
IF WTD EQ 3 THEN PRINT, 'NOTE: It may be necessary to use Option 1 or 2, first'
IF (WTD GE 1) AND (WTD LE 3) THEN SRT,dtl,WTD,STAR
IF WTD EQ 4 THEN BEGIN
  PUBL,sdata,dtl,STAR,NTL
  YNO=' ' ; want a listing?
  READ,'Press <ENTER> when ready to proceed',YNO
END
if WTD eq 5 then begin
  mkcusttab,STAR,dtl ; Make Custom Table.
  goto, STRT ; Go re-read orig. data.
endif
if WTD lt 6 then goto, LOOP
EXIT: print,string(7B) ; Flash screen.
print,'Unzoom the this window IF it has been expanded to full screen.'
yno = ' '
read,'Press the <RETURN> key when ready to continue.',yno
RETURN
END ;
;
;
PRO SRT,dtl,WTD,STAR
;***** SORT.PRO *****
;
; TO SORT .DTL DATA FILES by Charles L. Joseph 22-Jan-1980
;
; dtl the primary structure holding the measurements of DETAIL.
; WTD What-To-Do Flag 1 => sort by Wavelength
; 3 => sort by Periodic table then molecules.
; STAR name of the file
;
;*****
if WTD eq 1 then begin ; Sort by Lab. Wavelength
  nslot = dtl.wl

```

```

        nslot = 10000. - nslot                ; Ordering is in NSLOT
    endif
    if WTD eq 3 then nslot = dtl.iaf          ; Sort by Ion => NSLOT
    nslot = (nslot > 0) + 99999.*(nslot eq 0) ; Flag to ignore null data.
    tdtl = dtl                               ; Make copy of data.
    kk = 0
    for k = 0,199 do begin                   ; Perform sort from min to
        vy = min(nslot,ky)                  ; max.
        if vy lt 99999. then begin
            tdtl(kk) = dtl(ky)
        nslot(ky) = 99999.
            kk = kk + 1
        endif else goto, out
    endfor
out: dtl = tdtl
    openu,1,STAR+'.DTL'                       ; Make changes permanent by
    writeu,1,dtl                             ; writing them to file.
    close,1
RETURN                                       ; back to POSTO
END                                         ;
;
;
PRO PUBL,sdata,dtl,STAR,NTL
;***** PUBL.PRO *****
;
;       To print the results for POSTO.
;       By Charles L. Joseph                November 1980
;
;       sdata      a structure holding the measurements of a single profile.
;       dtl        a structure holding the primary data >> many sdata's.
;       STAR       name of the file
;       NTL        number of measured lines
;*****
;
;
; PHDR='Ion          Lab      f      D-lam    EQW      error    comments'
; N=-1                                           ; counter for pages
; PG=0                                           ; page #
; NT=0                                           ; total counter
; for k=0,4 do print,' '
PAGE: if !d.name eq 'TEK' then erase           ; start loop 4010
; PG=PG+1                                       ; increment page #
; starp = STAR+'-----'
; starp = strmid(starp,0,15)
; PGP = string(format='(i2)',PG)
; PRINT,starp,'                               Page:',PGP
; PRINT,' '
; PRINT,PHDR                                   ; page header
; PRINT,FORMAT='(14x,a4,5x,"Value mA",5x," Eqv Width")'
; PRINT,' '                                   ; and 2nd line
; NT=NT+44                                     ; # on page
; IF NT GT NTL-1 THEN NT=NTL-1                ; how many lines ?

```

```

LOOP: N=N+1 ; printout data
  dt = dtl(N)
  AION = string(dt.el)
  COM = string(dt.com)
  IF N GT NTL-2 THEN NT=45*PG ; for pages
  PLMN = ' +/- '
  PRINT,FORMAT='(A15,1x,F7.2,1x,F7.3,1x,I5,1x,f8.3,1x,f8.3,2x,A10)', $
    AION,dt.wl,dt.f,dt.owl,dt.eqw,dt.me,COM
  IF NT LT 45*PG THEN GOTO, LOOP ; don't page yet
  FOR I=0,11 DO PRINT,' ' ; move out of the way
  IF N LT NTL-1 THEN GOTO, PAGE ; set up new page
  CLOSE,3 ; close .LST up
RETURN ; back to POSTO
END ;
;
; *****
; MKCUSTTAB - to make a customized table (ASCII format).
; by Charles L. Joseph November 29, 1990
; *****
pro mkcusttab,STAR,dtl
  frstim = 1 ; First time flag.
  sdots = ''
  for k=1,6 do sdots=sdots+'.....:.....|' ; Make an axis marker.
  sdots = '|'+sdots ; Index starts at 0.
  dash = '-----'
  uline = ''
  for k=1,5 do uline = uline+'-----' ; Make long underline.
  mstr = string(bytarr(120,308)) ; 308 el. string array
;   each el: 120 char.
  miaf = long(intarr(301))+long(99999) ; To hold the ID codes.
  vtmp = where(dtl.iaf gt 0,tmax) ; Table size -> tmax.
  nstrt = 0 ; Starting line-display.
  if tmax ge 6 then nstp = 5 ELSE nstp = tmax - 1 ; Stopping line-display.
  NTOT = 36 ; # needed - Species ID.
  L = 0 ; Index for copt.
  copt = intarr(20) ; Column options, a .DTL
  uopt = intarr(20) ; For options of UP.
  upc = string(bytarr(10,20)) ; Holds format strings.
  prvopt = ' ' ; Holds previous opts.
  spe = ' ' ;
  mstr(306) = spe+STAR ; Last few lines of mstr
  mstr(304) = 'Species Wavelength f ' ; hold header info.
  mstr(303) = '-----'
  for k=0,tmax-1 do begin
sdata = dtl(k)
spe = string(sdata.el) ; Build a string of
spe = spe+string(format='(f9.2)',sdata.wl) ; ID's and assoc. info
spe = spe+' '+string(format='(f9.4)',sdata.f)+' '
mstr(k) = spe ; Put into mstr.
miaf(k) = sdata.iaf ; Save ID codes.
  endfor

```



```

uopt(L) = wup ; Save choice.
formatin: print, ' ' ; Input format.
print, 'Supported Formats are: f, e, & i having a max field of 10.'
print, 'Examples: (f8.3), (e10.3), (f10.0), (i6)'
wptc = ''
read, 'Enter format:', wptc
print, ' '
tst = strpos(wptc, '(') ; Parenthesis used?
if tst eq -1 then print, 'Parenthesis must be included', string(7B)
if tst eq -1 then goto, formatin
upc(L) = wptc
endif
if iopt ne 11 then L = L + 1 ; Incr. later if UP.
endif
if (iopt EQ 1) then begin ; Same cols for new?
print, string(7B), 'Old file had options:', prvopt ; Show previous ones.
yno = ''
read, 'Add all of these columns in new .DTL data?', yno
if (yno ne 'Y') AND (yno ne 'y') then begin ; If not the same,
tst = 0 ; find out which
kst = 0 ; ones.
icnt = 0
print, 'Sequentially enter each option for next .DTL file'
print, 'Enter -1 when finished'
while tst ge 0 do begin ; Get options desired
read, 'Enter next option', tst
if tst gt 2 then copt(icnt) = tst
if tst gt 2 then icnt = icnt + 1
wup = 0
if iopt eq 11 then begin ; To use a UP option.
read, 'Which element of the UP is to be added?', wup
uopt(icnt) = wup ; Save choice.
print, 'Formats are: f, e, & i having a max field of 10.'
print, 'Examples: (f8.3), (e10.3), (f10.0), (i6)'
wptc = ''
read, 'Enter format:', wptc
upc(icnt) = wptc
endif
if icnt gt 0 then L = icnt ; New master count.
endwhile
if icnt gt 0 then begin
copt(icnt:19) = 0 ; Catch any stray
uopt(icnt:19) = 0 ; left overs from
upc(icnt:19) = '' ; from before.
prvopt = ''
for k=0, icnt-1 do prvopt=prvopt+string(copt(k))
endif
if icnt eq 0 then goto, MORE ; No choices - abort.
endif
vtmp = mstr(306)
ssz = strlen(vtmp)
if (ssz gt NTOT+4) then vtmp = strmid(vtmp, 0, NTOT+3) + '___'

```



```

mstr(306) = vtmp
appnd,dtl,mstr,miaf,copt,uopt,upc,NTOT,tmax           ; Get new .DTL info.
endif
if iopt eq 2 then begin                               ; Delete some columns
print,'Which character columns are to be deleted?' ; of characters.
read,'Enter starting and ending columns',c1,c2
for k=0,307 do begin
  tst = mstr(k)
  tst = strmid(tst,0,c1-1)+strmid(tst,c2,120-c2)
  mstr(k) = tst
endifor
NTOT = FIX(NTOT - (c2-c1) - 1)                       ; Adjust counter.
endif
if (iopt GE 3) AND (iopt LE 11) then begin           ; Add an entry.
if iopt eq 11 then begin                             ; UP option.
  wup = uopt(L)
  upf = upc(L)
  L = L + 1                                         ; Still Needs incr.
endif
for k=0,199 do begin
  sdata = dtl(k)
  nextstring,iopt+2,nstrg,stitl,strsz,sdata,wup,upf
  mstr(k) = mstr(k)+' '+nstrg
endifor
ssz = 0
if frstim then begin                                ; If first time, then
  ssz = strlen(STAR)                                ; adj. character size
  if (ssz gt strsz+2) then ssz=strsz+2             ; accordingly.
  frstim = 0                                         ; No longer first.
endif
mstr(306) = mstr(306)+strmid(uline,0,strsz+2-ssz) ; Adj. to file name
mstr(304) = mstr(304)+stitl                          ; + new column names
mstr(303) = mstr(303)+strmid(dash,0,strsz+2)        ; + more dashed line.
NTOT = NTOT + strsz + 2                              ; Adjust counter.
endif
if (iopt lt 12) then goto, MORE
Outfile: vtmp = ''
read,'Enter file name of the table: (type: "none" for no file) ',vtmp
if (vtmp ne 'none') AND (vtmp ne 'NONE') then begin
  close,1
  openr,1,vtmp, ERROR = errtst
  close,1
  if errtst eq 0 then begin
yno = ''
read,'File already exists. Try another? ',yno
if (yno eq 'Y') OR (yno eq 'y') then goto, Outfile
endif
openw,1,vtmp
wrk = mstr(306)
mstr(306) = strmid(wrk,0,NTOT)
for k=0,3 do printf,1,mstr(306-k)                   ; Top of table.
for k=0,tmax-1 do printf,1,mstr(k)                  ; Table entries.

```

```

printf,1,mstr(303)
  close,1
endif

```

```
; 2nd dash line.
```

```
RETURN
```

```
END ; mkcusttab
```

```
;
;
```

```
; *****
; APPND is used by MKCUSTTAB to append additional data files to the
; customized table.
```

```
; by Charles L. Joseph Dec. 1, 1990
```

```
; dtl structure holding the contents of the .DTL file
; mstr an array of character strings holding the master data
; miaf master iaf used to slot the appended file
; jcnt the counter pointing to entry in the dtl structure.
; for example, 3 =>
```

```
; *****
pro appnd,dtl,mstr,miaf,copt,uopt,upc,NTOT,tmax
```

```

STRT: ndtl = '' ; Next .DTL filename.
nnew = 0 ; No-New Counter.
nold = 0 ; No-Old Counter.
read,'Enter new .DTL file name (leave off .DTL extension): ',ndtl
close,1
openr,1,ndtl+'.DTL', ERROR = errtst ; Does file exist?
close,1
if errtst ne 0 then begin ; If file NOT pres.
  yno = ''
  read,'Error opening file -- try another?',yno ; Try again?
  if (yno eq 'Y') OR (yno eq 'y') then goto, STRT ; Yes, get another or
  return ; else return.
endif
openu,1,ndtl+'.DTL' ; Open the .DTL file.
readu,1,dtl ; Read the data.
close,1 ; Close the .DTL file.
niaf = dtl.iaf ; Get species codes.
vtmp = where(niaf ne 0,ktst) ; How many --> ktst.
niaf = 99999*(niaf eq 0) + niaf ; Set all 0's -> 99999
dtl.iaf = niaf
sdots = '.....' ; Make a dummy filler
sdots = sdots+'.....' ; string.
sdots = sdots+'.....' ;
dash = '-----' ; To continue dash.
vtmp = where(copt gt 0,L) ; Find L=# of options.
k = 0 ; Index for new data.
kk = 0 ; Index for old data.

```

```
Next: if k gt 199 then return
```

```
sdata = dtl(k)
```

```
iafnxt = sdata.iaf
```

```
; Get next line of new
; Next species code.
```

```

vtmp = 0
tmpstr = ''
tmptit = ''
for lcnt=0,L-1 do begin
    jcnt = copt(lcnt)+2
    wup = uopt(lcnt)
    upf = upc(lcnt)
    nextstring,jcnt,nstrg,stitl,strsz,sdata,wup,upf
    tmpstr = tmpstr+' '+nstrg
    tmptit = tmptit+stitl
    vtmp = vtmp + strsz + 2
endfor
nstrg = tmpstr
stitl = tmptit
strsz = vtmp
Loop: iaftst = miaf(kk)
if iafnxt eq iaftst then begin
    mstr(kk) = mstr(kk)+nstrg
    kk = kk + 1
    k = k + 1
    if (k lt ktst) OR (kk lt tmax) then goto, Next
endif
if iafnxt gt iaftst then begin
    nnew = nnew + 1
    mstr(kk) = mstr(kk)+' '+strmid(sdots,0,strsz-2)
    kk = kk + 1
    goto, Loop
endif
if (iafnxt ne 0) AND (iafnxt lt iaftst) then begin
    nold = nold + 1
    for n=kk,299 do miaf(300+kk-n) = miaf(299+kk-n)
    for n=kk,299 do mstr(300+kk-n) = mstr(299+kk-n)
    miaf(kk) = sdata.iaf
    mstr(kk) = ''
    spe = string(sdata.el)
    spe = spe+string(format='(f9.2)',sdata.wl)
    spe = spe+' '+string(format='(f9.4)',sdata.f)+' '
    spe = spe+strmid(sdots,0,NTOT-36)
    mstr(kk) = spe
    tmax = tmax + 1
    goto, Loop
endif

Fin: uline = ''
for k=1,5 do uline=uline+'-----'
ndtl = ndtl+uline
mstr(306) = mstr(306)+strmid(ndtl,0,strsz)
mstr(304) = mstr(304)+stitl
mstr(303) = mstr(303)+strmid(dash,0,strsz)
NTOT = NTOT + strsz
print,'# of blank lines in OLD:',nold,' in NEW:',nnew

```

; Get all entries for  
; next species.  
; In case UP get info-  
; mation & format.  
; Entry --> string.  
; Build up all entries

; Put accumulation  
; variables into  
; permanent ones.  
; Get next line of old  
; If species agree:  
; - add on  
; - increment each  
; counter &  
; - Any more?

; Missing line of new.  
; Fill entry with dots  
; Set index to get  
; next line of old.

; New entry not in old  
; Shift to make room.  
; Shift to make room.  
; Add species code.  
; In case new << old.  
; Build a string of  
; ID's and assoc.  
; information plus  
; fill in dots.  
; Add new entry.  
; Table has 1 more.  
; Should = next time.

; 2s added to strsz  
; before.

; Incr. column count.

RETURN  
END

```

;
;
; *****
; NEXTSTRING is used to get the size and the conversion to string of the
; next set of entries to be put in the table. Called by APPND or MKCUSTTAB.
; by Charles L. Joseph November 1990
;
;      jcnt      the index of the requested portion of the sdata structure.
;                (see the manual for the sdata or dtl structures.)
;
;      nstrg     the returned Next String
;
;      stitl     the title of the entry (e.g. EQW, ERROR, 1st MOM., etc.)
;
;      strsz     string size, the number of columns required for the next entry.
;
;      sdata     a structure holding the measurements for a single profile.
;
;      wup       Which User Parameter requested <-- only if jcnt = 13
;
;      upf       Holds the format if a UserParameter is requested.
; *****
pro nextstring,jcnt,nstrg,stitl,strsz,sdata,wup,upf

nstrg = ''
stitl = ''
strsz = 0
if jcnt eq 5 then nstrg = string(format='(f8.2)',sdata.owl)
if jcnt eq 5 then stitl = 'Obs. Wave.'
if jcnt eq 6 then nstrg = string(format='(f8.4)',sdata.eqw)
if jcnt eq 6 then stitl = ' EQW (A) '
if jcnt eq 7 then nstrg = string(format='(f8.4)',sdata.me )
if jcnt eq 7 then stitl = ' ME (A) '
if jcnt eq 8 then nstrg = string(format='(f8.2)',sdata.fm )
if jcnt eq 8 then stitl = ' 1st MOM.'
if jcnt eq 9 then nstrg = string(format='(f8.2)',sdata.fme)
if jcnt eq 9 then stitl = ' ERROR '
if jcnt eq 10 then nstrg = string(format='(f8.3)',sdata.sm )
if jcnt eq 10 then stitl = ' 2nd MOM.'
if jcnt eq 11 then nstrg = string(format='(f8.3)',sdata.sme)
if jcnt eq 11 then stitl = ' ERROR '
if jcnt eq 12 then nstrg = string(sdata.com)+'.....'
if jcnt eq 12 then nstrg = strmid(nstrg,0,10)
if jcnt eq 12 then stitl = ' Comments '
if jcnt eq 13 then begin
val = strpos(upf,'f')
if val eq -1 then val = strpos(upf,'F')
if val eq -1 then val = strpos(upf,'i')
if val eq -1 then val = strpos(upf,'I')
if val eq -1 then val = strpos(upf,'e')
if val eq -1 then val = strpos(upf,'E')
if val eq -1 then return
strsz = strpos(upf, '.')
if strsz eq -1 then strsz = strpos(upf, ')')
strsz = fix(strmid(upf,val+1,strsz-1))
val = sdata.up
; From UP array.
; Floating format?
; Floating format?
; Integer format?
; Integer format?
; Exponential format?
; Exponential format?
; Can't find format.
; Find end of format.
; If no dot.
; String size.
; UP array -> val.

```

```

        val = val(wup) ; Get array element.
        nstrg = string(format=upf,val)+' ..' ; Use format in upf.
        if strsz lt 6 then nstrg = ' ' + nstrg ; Pad with 4 blanks?
        nstrg = strmid(nstrg,0,10) ; Make 10 characters.
        stitl = ' < UP'+string(format='(i2)',wup)+' > ' ; Make column title.
    endif
    if jcnt ge 12 then strsz = 10 ELSE strsz = 8 ; # of characters.

RETURN
END
; ***** END of POSTO PACKAGE *****

;***** EDATDTL.PRO *****
;
; To edit the .DTL files created by the Modular Spectral Line Analysis Program
;
; by Charles L. Joseph 6/12/84
; Latest Revision: 8/10/90
;
;*****
pro edatdtl,star

nulld = { noda, el: bytarr(15), iaf: long(0), wl: 0., f: 0., owl: 0., eqw: 0., $
me: 0., fm: 0., fme: 0., sm: 0., sme: 0., com: bytarr(10), up: fltarr(30) }
dtl = replicate({ noda }, 200) ; Make array for storage
sdata = nulld ; Make working copy.
close,1
OPENR,1,STAR+'.DTL', ERROR = errtst ; Check if file exists.
close,1
if errtst ne 0 then begin ; If no file, make one.
    close,1
    openw,1,STAR+'.DTL'
    writen,1,dtl
    close,1
endif
openu,1,STAR+'.DTL' ; Open file for updating.
readu,1,dtl ; Read data file.

plotconfig,0,' ',' ',-2,kdev,' ' ; Set Hardware defaults.
; kdev = getenv('TERM')
if kdev eq 'xterm' then set_plot,'X'
window,0,color=2,title=' ',xpos=760,ypos=300,xsize=380,ysize=300
!y.margin(0) = 2 ; This window for print-
plot,findgen(10) ; ing Reminders.
if kdev ne 'xterm' then x0 = -0.25*(!x.crange(1)-!x.crange(0)) + !x.crange(0)
if kdev eq 'xterm' then x0 = 0.0*(!x.crange(1)-!x.crange(0)) + !x.crange(0)
y0 = 0.5*(!y.crange(1)-!y.crange(0))
y0 = !y.crange(0)
erase
for k=0,2 do print,' '
print,' >>>> EDITING the .DTL File <<<<' ; Send Reminders to
for k=0,2 do print,' ' ; extra window.

```

```

xyouts,x0,2.0*yo+y0,'-----Editing Reminders-----'
xyouts,x0,1.8*yo+y0,'<CR> or n - goto NEXT entry'
xyouts,x0,1.6*yo+y0,' u - go UP to prev. entry'
xyouts,x0,1.4*yo+y0,' c - change current entry'
xyouts,x0,1.2*yo+y0,' i - insert new data'
xyouts,x0,1.0*yo+y0,' d - delete current entry'
xyouts,x0,0.8*yo+y0,' t - goto TOP of data'
xyouts,x0,0.6*yo+y0,' b - goto BOTTOM of data'
xyouts,x0,0.4*yo+y0,' k or q - Kill/Quit'
xyouts,x0,0.2*yo+y0,' w - Write (Update) File'
xyouts,x0,0.0*yo+y0,' e - Exit and write changes'

```

```

START: ML= 200 ; Maximum # of measures.
A = where(dtl.wl ne 0.0,NTL)
if (NTL le 0) then NTL = 1 ; Null data set ?
s1 = ' IAF Code:' & s2 = ' Rest Wave:' & s3 = ' f:'
s4 = ' Obs:' & s5 = 'EQW:' & s6 = ' Error:'
s7 = ' 1st Mom:' & s8 = ' 2nd Mom:'

TOP: N = -1
LOOP: N=N+1 ; Each time through,
dt = dtl(N) ; print new line of data.
AION = string(dt.e1)
print,FORMAT='(a15,a11,i6,a12,f9.3,a4,f6.4,a6,f9.3)', AION,s1, $
dt.iaf,s2,dt.wl,s3,dt.f,s4,dt.owl
print,FORMAT='(15x,3(a12,f9.4))', s5,dt.eqw,s7,dt.fm,s8,dt.sm
print,FORMAT='(15x,3(a12,f9.4))', s6,dt.me,s6,dt.fme,s6,dt.sme
print,' Comment: ',string(dt.com)
print,' '
print,'UserParameters'
print,dt.up
A=' '
READ,A ; Read keyboard option.
A=BYTE(A)
A=A(0)
if (A gt 90) then A = A - 32 ; To handle lower case.
IF ((A LE 13) OR (A EQ 78)) THEN GOTO,LOOP ; <CR> N Next line
IF (A EQ 67) THEN BEGIN ; C - correction
sdata = dtl(N) ; Set defaults
getsdata,sdata,N,A ; Get Updates
dtl(N) = sdata ; Insert in dtl
N = N - 1 ; Adj. Pointer
END
IF (A EQ 69) OR (A EQ 87) THEN GOTO,DONE ; E W Exit/Write
IF A EQ 83 THEN STOP ; S Stop
IF (A EQ 72) OR (A EQ 83) THEN N=N-1
IF A EQ 85 THEN BEGIN ; U Up one line,
N=N-2 ; Adj. Pointer
IF N LT -1 THEN BEGIN ; Top of File?
N=-1
PRINT,'TOP OF FILE',STRING(7B)
END

```

```

END
IF A EQ 84 THEN GOTO, TOP ; T Goto Top of file
IF A EQ 66 THEN N=N-1 ; B Goto Bottom
IF (A EQ 75) OR (A EQ 81) THEN GOTO, KILL ; K Q Kill/Quit
IF (A EQ 68) then begin ; D Delete Line
dtl(N) = dtl(N+1:ML-1)
dtl(ML-1) = nulld
NTL = NTL - 1 ; Adj. Number tot.
endif
if (A EQ 73) THEN BEGIN ; I Insert Line.
dtl(N+1) = dtl(N:ML-2) ; Make room in dtl
getsdata, sdata, N, A ; Get new line.
dtl(N) = sdata ; Insert in dtl.
N = N - 1 ; Adj. Pointer
NTL = NTL + 1 ; Adj. Number tot.
END
GOTO, LOOP
DONE: close, 1
OPENU, 1, STAR+'.DTL'
writeu, 1, dtl ; Write changes to .DTL
IF A EQ 87 THEN GOTO, TOP ; Just a write "W" go on.
KILL: ; ; Abort - Make no changes.
EXIT: CLOSE, 1 ; Close .DTL file.
wdelete, 0 ; Remove extra window.
END
;
;
;
pro getsdata, sdata, N, wtd
;***** getsdata.pro *****
;
; Gets new single-line of data (sdata) for the .DTL file from the keyboard.
; If "wtd" eq 67 => make a correction. Then previous data is taken as the
; default values.
;
; By Charles L. Joseph 8/10/90
; Latest Revision: 8/10/90
;
;*****

if wtd eq 67 then begin ; C - corrections
print, string(7B)
print, 'Existing data will be taken as the Default Values in the'
print, 'following questions.'
endif
print, ' '
print, 'Enter Complete Species Name - include ionization and fine structure.'
print, '15 characters maximum and DO NOT leave any preceeding blanks.'
print, 'Enter for example:'
print, ' Fe II* --- for iron singly ionized and excited Fine Structure'
print, ' H2 L 7,0 P(5) -- Lyman system of molecular Hydrogen'
print, ' '

```

```

if wtd eq 67 then print,'Default:',string(sdata.el)
vdum = ''
read,vdum
if (vdum eq '') AND (wtd eq 67) then elm=string(sdata.el) else elm=vdum
sdata.el = 0*sdata.el
elm = strtrim(elm,1)+' ..... '
sdata.el = byte(strmid(elm,0,15))

iafsv = long(sdata.iaf)
iaf = long(sdata.iaf) - long(1000)*FIX(sdata.iaf/1000)
iaf = FIX(iaf)
vdum = ''
if wtd eq 67 then print,'Default:',iaf/10
print,'What is the IONIZATION state [ 2 = II 4 = IV ] ?'
read,'If a molecule, then what is the VIBRATION number? ',vdum
if (vdum eq '') AND (wtd eq 67) then IA=iaf/10 else IA=fix(vdum)
vdum = ''
if wtd eq 67 then print,'Default:',iaf-10*(iaf/10)
print,'What is the FINE STRUCTURE state [ 3 = *** ] ?'
read,'If a molecule, then what is the ROTATION number? ',vdum
if (vdum eq '') AND (wtd eq 67) then FS=iaf-10*(iaf/10) else FS=fix(vdum)
iaf=FIX(IA*10 + FS)
vdum = strmid(elm,0,2)
if (vdum eq 'H2') OR (vdum eq 'HD') then begin
  ly = ''
  read,'Is this part of the Werner system',ly
  if (ly eq 'y') OR (ly eq 'Y') then iaf = iaf + 1000
endif

if wtd eq 67 then print,format='(a8,f10.3)', 'Default:',sdata.wl
READ,'What is the Rest WAVELENGTH of the line (Angstroms)?',vdum
if (vdum ne '') OR (wtd ne 67) then sdata.wl = float(vdum)

if wtd eq 67 then print,'Default:',sdata.f
READ,'What is the OSCILLATOR strength of the line?',vdum
if (vdum ne '') OR (wtd ne 67) then sdata.f = float(vdum)

if wtd eq 67 then print,format='(a8,f10.3)', 'Default:',sdata.owl
READ,'What is the OBSERVED WAVELENGTH of the line (Angstroms)?',vdum
if (vdum ne '') OR (wtd ne 67) then sdata.owl = float(vdum)

if wtd eq 67 then print,'Default:',sdata.eqw
READ,'What is the EQUIVALENT WIDTH (Angstroms) of the line?',vdum
if (vdum ne '') OR (wtd ne 67) then sdata.eqw = float(vdum)

if wtd eq 67 then print,'Default:',sdata.me
READ,'What is the ERROR of the measurement?',vdum
if (vdum ne '') OR (wtd ne 67) then sdata.me = float(vdum)

if wtd eq 67 then print,'Default:',sdata.fm
read,'What is the First Moment (Velocity km/s)?',vdum
if (vdum ne '') OR (wtd ne 67) then sdata.fm = float(vdum)

```



```

if wtd eq 67 then print,'Default:',sdata.fme
read,'What is the ERROR in the First Moment?',vdum
if (vdum ne '') OR (wtd ne 67) then sdata.fme = float(vdum)

if wtd eq 67 then print,'Default:',sdata.sm
read,'What is the Second Moment (km/s/s)?',vdum
if (vdum ne '') OR (wtd ne 67) then sdata.sm = float(vdum)

if wtd eq 67 then print,'Default:',sdata.sme
read,'What is the ERROR in the Second Moment?',vdum
if (vdum ne '') OR (wtd ne 67) then sdata.sme = float(vdum)

if wtd eq 67 then print,'Default:',string(sdata.com)
read,'Comments on Measurement (Up to 10 characters)?',vdum
if (vdum eq '') AND (wtd eq 67) then com=string(sdata.com) else com=vdum
sdata.com = 0*sdata.com
com = strtrim(com,1)+'          )'
sdata.com = byte(strmid(com,0,10))

vdum = ''
print,' '
if wtd ne 67 then sdata.up = 0.*sdata.up
read,'Stop to change UserParameters? (Y/N Default is N)',vdum
print,' '
if (vdum eq 'y') OR (vdum eq 'Y') then begin
  up = sdata.up
  print,'The UserParameter is a 30 point floating array.'
  print,'The program has stopped and values may be entered interactively.'
  print,'For example: the statement "up(2) = 31.", puts 31 in the'
  print,'          third element of the array.'
  print,'Note: the addresses range from 0 to 29.'
  print,'The command: ".con <Return>" must be entered when finished.'
  stop
  sdata.up = up
endif

; ----- Encode as Periodic Table/Molecules -----

tst = byte(strmid(elm,0,2))
els = 'H HeLiBeB C N O F NeNaMgAlSiP S ClArK CaScTiV CrMnFeCoNiCuZn'
els = els+'H2H2HDHDCOCHCNC2OHUiUiUi'          ; Elements + Molecules.
els = byte(els)
j = 0
atst = 0
while atst eq 0 do begin                                ; Search for its order.
atst = ((tst(0) eq els(j)) AND (tst(1) eq els(j+1)))
j = j + 2
if j gt 78 then goto,exitst
endwhile
exitst: sdata.iaf = long(iaf) + long(1000)*(j/2)      ; Include ionization.
RETURN

```



```

        if ebtst eq 0 then c_opts(7) = 'Toggle Error Bars ON'
goto, START                                ; Go start again.
    endif
if wtd eq 9 then begin
    print,string(7B),'                      ; Ring the BELL.
    print,'Move cursor to this work area'
    read,'and enter velocity offset (km/s):',vshft
    goto, START                              ; Go start again.
    endif
if wtd eq 11 then begin
    userprog5,STAR,niaf,rwav,lwfs,bgs,bge,tau,kcnt ; Call user's program
    goto, START
endif
if (wtd lt 1) OR (wtd ge 12) then goto,EXIT ; Exit Options.
if wtd gt 2 then wtd = 2                    ; Temp. void options.
    symnum = intarr(19)                     ; Zero plot symbol table
snum = intarr(19)                           ; Zero species table.
lnum = intarr(19)                           ; Zero profile table.
ksav = intarr(19)                           ; To save record #'s.
pleb = intarr(19)                           ; Plot which Error Bars?
cplt = intarr(19)                           ; For Connect-dots plot.
vs = fltarr(19)                             ; For velocity shifts.
spei = string(80)                           ; For window label.

START: ; ----- Starting place for 1st species -----
L = 0 & H = 0                                ; Auto Plot ranges.
B = 0 & T = 0
    INUM = 0                                  ; Initialize Image Number
if (wtd eq 1) OR (wtd eq 2) then begin       ; Start with 1st data.
    which = wmenu(n_opt,title=0,init=1)      ; Get species choice.
    if which eq knone then goto, EXIT        ; Choice: "No More".
    snum(INUM) = which                       ; Save Species Number
    iaf = iafb4(which)
    spei = '...' +n_opt(which)
    plotconfig,0,spei,' ',0,kdev,'large'    ; Open type 0 window.
    endif
; ----- Next Get, Header, Optical Depths + Spectra -----

LOOP: ;

    lch = 1                                  ; Just in case no choices
    ncnt = where(niaf eq iaf)                 ; Find all matches.
    sz = size(ncnt)                          ; Total # in sz(1).
    if sz(0) eq 0 then goto, BRNCH           ; Problem, No Match.
; if sz(1) eq 1 then kcnt = fix(ncnt(0))    ; Only one profile.
    if sz(1) ge 1 then begin                ; Several profiles of the
        l_choice = string(bytarr(22,sz(1)+2)) ; same species setup to
        l_choice(0) = 'Which Line? Log fW'  ; make a selection.
        l_choice(sz(1)+1) = 'None'
        for k=1,sz(1) do begin              ; For each profile:
            swav = string(format='(f8.3)',rwav(ncnt(k-1))) ; -list rest wavelen.

```

```

slwf = string(format='(f5.3)',lwfs(ncnt(k-1))) ; -list log f-lambda.
l_choice(k) = swav+' '+slwf ; -put into MENU.
endfor
lch = wmenu(l_choice,title=0,init=1) ; Get MENU profile choice
if (lch le 0) OR (lch ge sz(1)+1) then goto, BRNCH ; Not valid selection.
kcnt = ncnt(lch-1) ; Put choice into variab.
endif
lnum(INUM) = lch ; Store profile number.
ksav(INUM) = kcnt ; Store for later plots.
if cncnt ne 2 then symnum(INUM) = wmenu(s_opt,title=0,init=1) ; Symbol choice?
if symnum(INUM) eq 9 then begin ; Line ONLY:
cncnt = 2 ; => Set to Line.
ebtst = 0 ; => No errors.
if ebtst eq 0 then c_opts(7) = 'Toggle Error Bars ON' ; => Reset Menu opt.
endif
if ebtst then pleb(INUM) = 1 ELSE pleb(INUM) = 0
cplt(INUM) = cncnt
vs(INUM) = vshft

; ----- Read & Plot Optical Depths -----

PLT: asy = findgen(16)*(!PI*2/16.) ; Used for plot symbol.
if HC then plotconfig,1,' ',' ',-1,kdev,' ; Config. for Hardcopy?

for k=0,INUM do begin
kcnt = ksav(k)
close,5
openr,5,STAR+'.TAU'
tautmp = fltarr(2,600) ; Make temporary array.
tautmp = tas(3*kcnt+1) ; Get wavelength & TAUs(-).
sz = TOTAL(tautmp(0,*) ne 0) ; Find the length.
tau = fltarr(6,sz) ; Make suitable TAU array &
tau(0:1,0:sz-1) = tautmp(0:1,0:sz-1) ; stuff Wavelength & TAUs(-)
tautmp = 0.*tautmp ; Zero temporary array.
tautmp = tas(3*kcnt+2) ; Get TAUs and TAUs(+) &
tau(2:3,0:sz-1) = tautmp(0:1,0:sz-1) ; put into TAU array.
tautmp = 0.*tautmp ; Zero temporary array.
tautmp = tas(3*kcnt+3) ; Get Spectrum & Continuum
tau(4:5,0:sz-1) = tautmp(0:1,0:sz-1) ; and put into TAU.
ttmp = where(tau(2,*) gt 1.0,tcnt) ; Which are limits?
mtmp = where(tau(3,*) gt 1.0,mcnt) ; Which are limits?
close,5

lwf_fact = 14.567 - lwfs(kcnt) ; Convert to Column Density.
tau(1:3,*) = tau(1:3,*) + lwf_fact
swav = string(format='(f8.3)',rwav(kcnt)) ; Rest wavelength string.

if MKA eq 1 then begin ; IF Make ASCII file.
slwf = string(format='(f6.3)',lwfs(kcnt)) ; <-- log f-lambda.
print,swav+' log f-lambda: '+slwf+' '+n_opt(snum(k))
printf,1,swav+' log f-lambda: '+slwf+' '+n_opt(snum(k))
sbg = string(format='(f9.1)',bgs(kcnt)) ; <-- Background.

```

```

sbge = string(format='(f9.1)',bge(kcnt)) ; <-- BG Error.
printf,1,'Background: '+sbg+ ' BG Error: '+sbge
printf,1,' '
printf,1,'Wavelength log N(min) log N log N(max) Spect. Cont.'
for kt=0,sz-1 do printf,1,format='(f9.3,5f11.2)', $
    tau(0,kt),tau(1,kt),tau(2,kt),tau(3,kt),tau(4,kt),tau(5,kt)
    printf,1,' '
endif

if vshft ne 0. then begin
print,string(7B),' '
print,'Caution: profile shifting should seldom be done'
print,' and then only with extreme care.'
print,'The '+swav+ 'profile has been shifted by:',vshft,' km/s.'
endif
v0 = 2.9979e5/rwav(kcnt)
v = v0*(tau(0,*)- rwav(kcnt)) + vs(k) ; Calculate velocity.

if symnum(k) eq 3 then begin ; Make special symbols?
usersym,0.7*cos(asy),0.5*sin(asy),/FILL ; Convert dots to filled
tsym = 8 ; circles.
end else begin
usersym,0.7*cos(asy),0.5*sin(asy) ; Option for open circles.
tsym = symnum(k) ; Most cases, take IDL's.
if tsym eq 9 then tsym = 0 ; Solid line option.
endelse
ymsav = !y.margin(0) ; Just in case HC = 1, be
yht = 0.22 - 0.04*k ; ready to make a key.
; If cplt(k)=2 -> LINE
; =1 -> LINE+DOTS.
; =0 -> DOTS.
if (k eq 0) AND (cplt(k) lt 2) then begin ;
if (H ne 0) OR (B ne 0) OR (L ne 0) OR (T ne 0) then begin
plot,v,tau(2,*),xtitle='velocity',ytitle='log N',psym=tsym,/YNOZ, $
xrange=[L,H],yrange=[B,T],xstyle=1,ystyle=1
endif ELSE begin
plot,v,tau(2,*),xtitle='velocity',ytitle='log N',psym=tsym,/YNOZ
endelse
if HC then !y.margin(0) = 4
if HC then plots,[0.15,0.20],[yht,yht],psym=tsym,/NORMAL
if HC then xyouts,0.25,yht-0.01,l_choice(lnum(k)),/NORMAL
!y.margin(0) = ymsav
endif
if (k ne 0) AND (cplt(k) lt 2) then begin
oplot,v,tau(2,*),psym=tsym
if HC then !y.margin(0) = 4
if HC then plots,[0.15,0.20],[yht,yht],psym=tsym,/NORMAL
if HC then xyouts,0.25,yht-0.01,l_choice(lnum(k)),/NORMAL
!y.margin(0) = ymsav
endif
if (k eq 0) AND (cplt(k) eq 2) then begin
if (H ne 0) OR (B ne 0) OR (L ne 0) OR (T ne 0) then begin
plot,v,tau(2,*),xtitle='velocity',ytitle='log N',/YNOZ, $

```

```

      range=[L,H],yrange=[B,T],xstyle=1,ystyle=1
endif ELSE begin
      plot,v,tau(2,*),xtitle='velocity',ytitle='log N',/YNOZ
endif
endelse
if HC then !y.margin(0) = 4
if HC then plots,[0.15,0.20],[yht,yht],/NORMAL
if HC then xyouts,0.25,yht-0.01,l_choice(lnum(k)),/NORMAL
!y.margin(0) = ymsav
endif
if (k ne 0) AND (cplt(k) eq 2) then begin
      oplot,v,tau(2,*)
if HC then !y.margin(0) = 4
if HC then plots,[0.15,0.20],[yht,yht],psym=tsym,/NORMAL
if HC then xyouts,0.25,yht-0.01,l_choice(lnum(k)),/NORMAL
!y.margin(0) = ymsav
endif
if cplt(k) eq 1 then begin
      oplot,v,tau(2,*) ; Connect the points opt.
if HC then !y.margin(0) = 4
if HC then plots,[0.15,0.20],[yht,yht],/NORMAL
if HC then xyouts,0.25,yht-0.01,l_choice(lnum(k)),/NORMAL
!y.margin(0) = ymsav
endif
if pleb(k) then oplot,v,tau(1,*),psym=tsym, symsize=0.6
if pleb(k) then oplot,v,tau(3,*),psym=tsym, symsize=0.6

usersym,aax,aay ; Over plot any Lower Limits
if tcnt gt 1 then oplot,v(ttmp),tau(2,ttmp),psym=8
if tcnt eq 1 then oplot,[v(ttmp),v(ttmp)],[tau(2,ttmp),tau(2,ttmp)],psym=8
if pleb(k) then begin
      if mcnt gt 1 then oplot,v(mtmp),tau(3,mtmp),psym=8
      if mcnt eq 1 then oplot,[v(mtmp),v(mtmp)],[tau(3,mtmp),tau(3,mtmp)], $
psym=8
endif

endif

endifor

if HC then begin ; If in HardCopy Mode:
      spe = 'Input File: '+STAR+'.TAU' ; Put File Name on header.
      xyouts,0.15,0.99,spe,/NORMAL
      spe = n_opt(1)
      xyouts,0.50,0.99,n_opt(snum(k)+1),/NORMAL ; Species Name on header.
      xyouts,0.80,0.99,date,/NORMAL ; Put Date on header.
      plotconfig,-1,spei,'',-1,kdev,'large' ; send to plotter & config
      HC = 0 ; no longer HardCopy.
endif

if MKA eq 1 then begin ; If Making ASCII file:
      close,1 ; close the file.
      MKA = 0 ; no longer Making ASCII.
endif

```

; ----- Master Branching Section inside LOOP -----

```

BRNCH: ;
  wtd = wmenu(c_opts,title=0,init=2) ; call COG Menu.
  if wtd eq 4 then stop ; Stop - Temporary Halt.
  if wtd eq 4 then goto, BRNCH ; - Start again.
  if wtd eq 7 then begin ; Toggle plot error bars.
    vtmp = ebtst
    if vtmp eq 0 then ebtst = 1 ; Plots to have errors.
    if vtmp eq 1 then ebtst = 0 ; Stop plotting errors.
    if ebtst eq 1 then c_opts(7) = 'Toggle Error Bars OFF'
    if ebtst eq 0 then c_opts(7) = 'Toggle Error Bars ON'
    if ebtst eq 1 then pleb(INUM) = 1 ELSE pleb(INUM) = 0
    goto, PLT ; Replot & Bring up Menu.
  endif
  if wtd eq 8 then begin ; Diff. Connect-Dots Status?
    tcnct = wmenu(ctd_opt,title=0,init=1) ; Get choice.
    if (tcnct le 0) OR (tcnct gt 3) then goto, PLT ; Silly choice - ignore.
    if (cnct eq 2) AND (tcnct lt 3) then $ ; From line to with "dots"?
    symnum(INUM) = wmenu(s_opt,title=0,init=1) ; Symbol choice?
    cnct = tcnct-1 ; Set current status & keep
    cplt(INUM) = cnct ; record for the future.
    if cnct eq 2 then ebtst = 0 ; Line only => No errors.
    if ebtst eq 0 then c_opts(7) = 'Toggle Error Bars DN'
    if ebtst eq 1 then pleb(INUM) = 1 ELSE pleb(INUM) = 0
    goto, PLT ; Replot & Bring up Menu.
  endif
  if wtd eq 9 then begin
    print,string(7B),' ; Ring the BELL.
    print,'Move cursor to this work area'
    read,'and enter velocity offset (km/s):',vshft
    vs(INUM) = vshft ; Save for later plots.
    goto, PLT ; Go start again.
  endif
  if wtd eq 10 then begin ; Adjust f value?
    print,string(7B),' ; Ring the BELL.
    print,'Move cursor to this work area.'
    print,'The current value is:',lwfs(kcnt)
    read,'Enter new value of log (f-lambda)',lwftmp
    lwfs(kcnt) = lwftmp ; Save for later plots.
    goto, PLT
  endif
  if wtd eq 11 then begin
    userprog5,STAR,niaf,rwav,lwfs,bgs,bge,tau,kcnt ; Call user's program
    goto, BRNCH ; - Start again.
  endif
  if wtd le 0 then goto, BRNCH
  if (wtd lt 1) OR (wtd ge 12) then goto,EXIT ; Exit Options.
  if (wtd eq 1) then begin ; Get NEW 1st data set.
    vshft = 0. ; Velocity shift default.
    goto, START
  endif
endif

```

```

      (wtd eq 2) then begin
        INUM = INUM + 1
        if INUM gt 18 then begin
          print,string(7B)
          print,'WARNING: Program can only handle 19 profiles.'
          INUM = 18
        endif
        vahft = 0.
        snum(INUM) = which
        iaf = iafb4(which)
        goto, LOOP
      endif
      if wtd eq 3 then begin
        HC = 1
        goto, PLT
      endif
      if wtd eq 5 then begin
        READ,'What are Xmin,Xmax,Ymin,Ymax?',L,H,B,T
        goto, LOOP
      endif
      if wtd eq 6 then begin
        ASC: afile = ''
        print,string(7B),'
        print,'Move cursor into this Workspace and'
        read,'Enter OUTPUT file name: ',afile
        close,1
        openr,1,afile, ERROR = errtst
        close,1
        if errtst eq 0 then begin
          print,string(7B),'Warning there is a file with that name.'
          yno = ''
          read,'Would you like to choose another name? (no => destroy old)',yno
          if (yno eq 'Y') OR (yno eq 'y') then goto, ASC
        endif
        openw,1,afile
        MKA = 1
        goto, PLT
      endif
      if (wtd eq 5) OR (wtd eq 6) then goto,LOOP
; ----- END Branching Section -----

EXIT: close,5
; !x.margin(0) = 10.
wdelete,0

RETURN
END
;
;
;

```

; Add more sets of data.

; Incr. Image Number.

; Too many profiles!

; Ring warning BELL.

; Set to back to max.

; Velocity shift default.

; Go get new set of obs.

; Want HardCopy?

; Adjust plot limits?

; Make ASCII File of data?

; Ring the BELL.

; Give directions.

; Get File Name.

; Is there one already?

; If file exists....

; Open the file.

; Set flag to print.

; GO cycle through data.

; Close file and return.

; Left plot margin-&gt;default.

; Delete Existing window.



```

; *****
;   GETRDY is a setup Program for MANTAU   by Charles L. Joseph 6/1/90
;
;   STAR      file name of the .TAU file.
;   niaf      an array holding the list of species codes.
;   rwav      an array holding the list of rest wavelengths.
;   lwfs      an array holding the log f-lambdas.
;   bgs       an array holding the background levels.
;   bge       an array holding the BG errors.
;   n_opt     string array holding the different species.
;   c_opts    string array holding the program Control options.
;   s_opt     string array holding the Symbol options.
;   ctd_opt   string array holding the Connect-The-Dots? options.
;   iaFB4     an array holding the distinctly unique species codes.
;             differs from niaf in that niaf may have multiple
;             entries of the same species.
;   knone     is the comparison value of the choice indicating none.
;
; *****

```

```

pro getrdy,STAR,niaf,rwav,lwfs,bgs,bge,n_opt,c_opts,s_opt,ctd_opt,iafb4,knone

```

```

as = assoc(5,fltarr(200))           ; Association variables for
close,5
openr,5,STAR+'.TAU'                 ; Open file for Update & get
niaf = as(0)                         ; 1st record - species codes
rwav = as(1)                         ; Get addresses in .DTL file
bgs = as(2)                          ; Get previous BG levels.
bge = as(3)                          ; Get previous BG errors.
dates = as(4)                        ; Get previous date codes.
lwfs = as(5)                         ; Get prev. log f-lambdas.
close,5

els = 'H HeLiBeB C N O F NeNaMgAlSiP S ClArK CaScTiV CrMnFeCoNiCuZn'
els = els+'H2H2HDHDCOCHCNC2OHU U U' ; Elements + Molecules
sfs = '***** X'
sii = 'I II III IV V VI VII VIIIIX X ..... '

plotconfig,0,'Optical Depths',' ',0,kdev,' ; Set up for type 0 window.
i_opt = string(bytarr(26,99))           ; Can handle 99 species.
iafs = where(niaf ne 0.)
sz = size(iafs)
if (sz(0) eq 0) then ntl = 0 ELSE ntl = sz(1)
iafb4 = fltarr(99)
kk = 1
for k = 1, ntl do begin ; Search through obs. data.
nxt = FIX(iafs(k-1)) ; Get address of next value.
iaftst = niaf(nxt) ; Put in test variable and
tst = where(iaftst eq iaFB4) ; see if there are other
sz = size(tst) ; identical entries.
if (sz(0) eq 0) then begin ; If not, add this species
iafb4(kk) = iaftst ; to the list.

```

```

icnt = fix(2.*(niaf(nxt)/1000.-1))
ii = niaf(nxt) - 1000.*FIX(niaf(nxt)/1000)
fs = ii mod 10.
ii = fix(ii/10)
siaf = ''
if icnt lt 60 then begin
  if ii le 0 then ii = 11
  if (fs gt 0) AND (fs le 9) then fs=strmid(sfs,0,fs) ELSE fs = ''
  siaf = strmid(sii,4*(ii-1),4)+fs
endif
if icnt ge 60 then begin
  siaf = 'vib:'+string(format='(i2)',ii)
  siaf = siaf+ ' rot:'+string(format='(i2)',fs)
endif
i_opt(kk) = strmid(els,icnt,2)+' '+siaf
      kk      = kk + 1 ; Incr. # of valid entries.
endif
endfor
i_opt(0) = 'Species Options'
i_opt(kk) = 'No more'
knone = kk
n_opt = i_opt(0:kk) ; Take only sub-array.
iafb4 = iaib4(0:kk)

;
; ----- Load the Control Menu Options -----
;
c_opts = string(bytarr(26,13))
c_opts(0) = 'Options:'
c_opts(1) = 'Get New Species'
c_opts(2) = 'Add Another Profile'
c_opts(3) = 'Send Plot to Laser Printer'
c_opts(4) = 'Stop - Temporary Halt'
c_opts(5) = 'Adjust Plotting Limits'
c_opts(6) = 'Make ASCII file of Data'
c_opts(7) = 'Toggle Error Bars ON'
c_opts(8) = 'Change PLOT Format'
c_opts(9) = 'Shift Rest Wavelength'
c_opts(10) = 'Change log (f-lambda)'
c_opts(11) = 'Run User Program 5'
c_opts(12) = 'Quit and Return'

;
; ----- Load Plotting Symbol Options -----
;
s_opt = string(bytarr(16,10))
s_opt(0) = 'Plotting Symbols'
s_opt(1) = 'Plus sign'
s_opt(2) = 'Asterisk'
s_opt(3) = 'Filled Circle'
s_opt(4) = 'Diamond'
s_opt(5) = 'Triangle'
s_opt(6) = 'Square'

```

```

s_opt(7) = 'X'
s_opt(8) = 'Open Circle'
s_opt(9) = 'Line - no pts'

```

```

; ----- Load Connect the Dots Options -----

```

```

ctd_opt = string(bytarr(24,4))
ctd_opt(0) = 'Connect the DATA points?'
ctd_opt(1) = 'Plot only as DOTS'
ctd_opt(2) = 'Plot as DOTS and CONNECT'
ctd_opt(3) = 'Plot data only as a LINE'

```

```

RETURN
END

```

```

PRO INTGRT,Y,X,CNT,TAU,JE,XF,XI,XLIMIT,mdata,mp,mcntrl,up

```

```

; ***** INTGRT *****

```

```

; By Charles L. Joseph Jan. 12, 1984 June, 1990

```

```

; DETERMINES EQUIVALENT WIDTHS OR INTEGRATED FLUXES
; OF ABSORPTION OR EMISSION FEATURES

```

```

; A TRAPEZOIDAL METHOD IS USED TO COMPUTE THE INTEGRATION

```

```

; Y flux vector
; X wavelength vector
; CNT continuum vector
; EPS epsilon vector (data quality)
; JE index of cursor
; XF final wavelength value allowed for in fits or integration
; XI initial or first wavelength value ( XI=left, XF=right)
; mp structure containing mp.wtd -1 => exit

```

```

; *****

```

```

; HC = 0 ; HardCopy Flag -> NO.
; xxxs = fltarr(2) ; For Overplot markers.
; yyys = fltarr(2) ; For Overplot markers.
; sz = size(JE) ; Determine # of profiles
; if sz(0) eq 0 then lcs = 1 else lcs = sz(1) ; to be measured.
; xmx = max(XLIMIT,J3) ; Get longest wavelength.
; xmx = max(x,J2) ;
; jmx = fix(J2/2) ; Temporary use of jmx.
; mp.SNR = (CNT(jmx)-mp.bg)/mcntrl.ESAV ; Signal-to-Noise Ratio.
; jmx = J2
; xmn = min(x)
; while (X(J2) gt xmx) do J2=J2-1 ; Find & adjust XLIMIT's
; XLIMIT(J3) = X(J2-3) ; Exclude last 3 points.
; kcmt = 0 ; # of pts in integration

```



```

PLTSTRT: if !d.name eq 'PS' then device,/CLOSE ;
  tle = '..... Measurement Results' ; Title for window.
  plotconfig,HC,'',tle,2,kdev,' ; Configure graphics.
; if HC eq 1 then device,/ENCAPSULATED,FILENAME='intgrt2.ps'
; TAUs => profile areas:
  ymn = min(tau(4,0:kcnt-1),J5) ; Find min and max flux
  ymax = max(Y) ; to see if plot BG.
  if (ymn lt 0.2*ymax) then pbg = 1 ELSE pbg = 0 ; Set plot BG flag.
if pbg then begin ; Going to show BG+error
  vmn = mp.bg - mp.bgerr ; Get min. needed for BG
  if vmn lt 0 then vmn = 1.1*vmn ; Adjust for more space.
  if ymn lt vmn then vmn = ymn ; Less than min. spect?
  ymax = 1.1*ymax ; Adjust for more space.
  xxxs(0) = xmn & xxxs(1) = xmx ; Put limits into small
  yyys(0) = vmn & yyys(1) = ymax ; over plot array &
  ; let IDL make axis.

  plot,xxxs,yyys,/NODATA, xtitle='Wavelength (A)', ytitle='Rel. Flux'
  if mp.SM0 then oplot,X,SMOOTH(Y,9) else oplot,X,Y ; Now really plot data.
  vmn = where(x lt tau(0,J5),jbg) ; Get ready to overplot
  if (jbg gt jcnt) then xxxs(0) = x(jbg-jcnt) ELSE xxxs(0)=x(0) ; BG range.
  if (jbg lt jmx-jcnt) then xxxs(1)=x(jbg+jcnt) ELSE xxxs(0)=x(jmx)
  yyys(0:1) = mp.bg
  oplot,xxxs,yyys ; Over plot BG.
  yyys(0:1) = mp.bg - mp.bgerr ; Over plot BG-BGerror.
  oplot,xxxs,yyys,linestyle=2
  yyys(0:1) = mp.bg + mp.bgerr ; Over plot BG+BGerror.
  oplot,xxxs,yyys,linestyle=2
endif
if pbg ne 1 then begin ; NOT going to show BG.
  if mp.SM0 then plot,X,SMOOTH(Y,9),/YNOZ, xtitle='Wavelength (A)', $
  ytitle='Flux' else plot,X,Y,/YNOZ,xtitle='Wavelength (A)',ytitle='Rel. Flux'
endif
if mp.SM0 ne 1 then oplot,X,Y,psym=8
oplot,X,CNT ; Put on continuum.
plotlab2,X,Y,CNT,mp,mcntrl,up ; Customized labels.
xxo=0.0*(!x.crange(1)-!x.crange(0)) + !x.crange(0)
yyo=1.05*(!y.crange(1)-!y.crange(0)) + !y.crange(0)
xyouts,xxo,yyo,mp.FNAM ; Put on File name.
xxo=0.65*(!x.crange(1)-!x.crange(0)) + !x.crange(0)
xyouts,xxo,yyo,systeme(0)
if kcnt gt 0 then begin
  for kk=0,kcnt-1 do begin ; Show areas that were
    xxxs(0:1) = tau(0,kk) ; integrated.
    yyys(0) = tau(4,kk)
    yyys(1) = tau(5,kk)
  oplot,xxxs,yyys
  endif
  xdum = where(XLIMIT gt 0,nxs)
  yout = 0.07*(!y.crange(1)-!y.crange(0)) + !y.crange(0)
  for kk=0,nxs-1,2 do begin ; Show the areas that

```

```

yyys(0) = !y.crange(0)
yyys(1) = !y.crange(1)
xxxs(0:1) = XLIMIT(kk)
oplot,xxxs,yyys,linestyle=1
xxxs(0:1) = XLIMIT(kk+1)
oplot,xxxs,yyys,linestyle=1
xxxs(0) = XLIMIT(kk)
yyys(0:1) = yout
oplot,xxxs,yyys
endifor

!y.margin(0) = 1

spe = 'Measurements in file: '+mp.STAR+'.DTL'
xyouts,0.05,0.32,spe,/NORMAL
spe = 'Real S/N = '+string(format='(f6.1)',mp.SNR/(mp.cohfac^0.5))
xyouts,0.765,0.32,spe,/NORMAL
spe = string(format='(f6.1)',mp.SNR)
spe = 'Apparent S/N Ratio:'+spe+' based on '
spe = spe+string(format='(i3)',mcntrl.NNE)+' points with a Noise '
spe = spe+'Coherence Length of'+string(format='(f4.1)',mp.cohfac)
xyouts,0.05,0.30,spe,/NORMAL
spe = string(format='(i2)',mp.poly)
spe = 'Continuum was fit with a polynomial of order:'+spe+'
spe = spe+string(format='(f9.2)',mcntrl.fin)+string(format='(f9.2)',mcntrl.f2n)
xyouts,0.05,0.28,spe,/NORMAL
spe = 'BG (background) was taken to be:'
spe = spe+string(format='(f12.1)',mp.bg)+' +/-'
spe = spe+string(format='(f12.1)',mp.bgerr)
xyouts,0.05,0.26,spe,/NORMAL
sdata = mdata(0)
spe = 'For '+string(sdata.e1)
xyouts,0.05,0.24,spe,/NORMAL
spe = string(format='(f8.3)',sdata.w1)
xyouts,0.24,0.24,spe,/NORMAL
spe = ' error contributions from BG:'
spe = spe+string(format='(f9.4)',mcntrl.e0)
spe = spe+string(format='(f9.3)',mcntrl.e1)+string(format='(f9.3)',mcntrl.e2)
xyouts,0.34,0.24,spe,/NORMAL
spe = 'Errors below are the Addition in Quadrature of'
spe = spe+' the Background and RMS-Noise Errors'
xyouts,0.05,0.22,spe,/NORMAL
spe = 'Species'
xyouts,0.05,0.18,spe,/NORMAL
spe = 'Lab. Wave.'
xyouts,0.21,0.18,spe,/NORMAL
spe = 'f'
xyouts,0.35,0.18,spe,/NORMAL
spe = 'Obs. Wave'
xyouts,0.43,0.18,spe,/NORMAL
spe = ' EQW (A) 1st (km/s) 2nd (km/s/s)'
xyouts,0.57,0.18,spe,/NORMAL

```

```

; Set to plot top to
; bottom.
; were used to calculate
; polynomail fit of the
; continuum.

```

```

; Connect solid line for
; continuum areas.

```

```

; Enable text area.

```

```

; Create string.
; Print File name.

```

```

; Start building string.

```

```

; Print S/N results.

```

```

; Start building string.

```

```

; Start building string.

```

```

; Print BG & uncertainty

```

```

; BG Error contribs.

```

```

; Start building

```

```

; a string.

```

```

; BG Error contribs.

```

```

; Build string for a

```

```

; note on errors.

```

```

; BG Error contribs.

```

```

; For each measurement:
; - Get Measurements.
; - Adj. print locat.
for k=0,lcs-1 do begin
  sdata = mdata(k)
  yht = 0.16 - 0.04*k
  spe = string(sdata.el)
  spe = strtrim(spe,2)+'.....'
  spe = strmid(spe,0,15)
  xyouts,0.05,yht,spe,/NORMAL ; - Print moments.
  spe = string(format='(f8.3)',sdata.wl)
  xyouts,0.21,yht,spe,/NORMAL ; - Print moments.
  spe = string(format='(f6.4)',sdata.f)
  xyouts,0.33,yht,spe,/NORMAL
  spe = string(format='(f8.3)',sdata.owl)
  xyouts,0.43,yht,spe,/NORMAL
  spe = string(format='(f10.4)',sdata.eqw) ; - Build a string.
  xyouts,0.55,yht,spe,/NORMAL ; - Print moments.
  spe = string(format='(f10.3)',sdata.fm)
  xyouts,0.67,yht,spe,/NORMAL ; - Print moments.
  spe = string(format='(f10.3)',sdata.sm)
  xyouts,0.82,yht,spe,/NORMAL ; - Print moments.
  yht = 0.14 - 0.04*k
  xyouts,0.47,yht,'Errors:',/NORMAL ; - Adj. print locat.
  spe = string(format='(f10.4)',sdata.me) ; values.
  xyouts,0.55,yht,spe,/NORMAL ; - Print errors.
  spe = string(format='(f10.3)',sdata.fme)
  xyouts,0.67,yht,spe,/NORMAL ; - Print errors.
  spe = string(format='(f10.3)',sdata.sme)
  xyouts,0.82,yht,spe,/NORMAL ; - Print errors.
endifor
if HC eq 1 then plotconfig,-1,' ',' ',-1,kdev,' ; Send plot & -> term.
if HC eq 0 then begin ; If not already making
  HC = wmenu(['Copy?','Yes','No'],title=0, init=1) ; a hardcopy, want to?
  if HC eq 1 then goto, PLTSTRT ; If yes, go do it.
endif
EXIT: CLOSE,4 ; close it up
  wdelete,2 ; Delete extra window.
  RETURN ; go back to DETAILS
END ; INTGRT ;
;
;
;
;
PRO EQUIVW,X,Y,C,TAU,JE,XF,XI,kcnt,XLIMIT,sdata,mp,mcntrl,up
;
; ***** EQUIVW *****
;
; COMPUTES INTEGRATED FLUX - ABSORPTION OR EMISSION LINES
;
;
; TRAPOZOIDAL METHOD
;
; X wavelength vector
; Y flux vector
; C continuum vector

```

```

;      JE      index of cursor position
;      XF      right def. and stop point
;      XI      left def. and stop point
;      kcnt    number of points used in the integration
;      XLIMIT  cursor positions used to define continuum regions
;      sdata   structure containing the ID plus moments and errors
;      mp      structure MSLAPparameter
;      mcntrl  structure for some control variables
;      up      UserParameter, a 30 point floating vector which is
;              reserved exclusively for the user.

```

```

;
;
;      Translated to IDL by C. JOSEPH 7/1/89 from code by E. Jenkins
;

```

```

; *****

```

```

J1 = 0
while (X(J1) lt XI) do J1=J1+1      ; Find left edge of integrat-
xmx = max(x,J2)                     ; ing region.
while (X(J2) gt XF) do J2=J2-1      ; Find right edge of integr-
kcnt = J2 - J1 + 1                   ; ating region.
X0 = 0.0
BKG = mp.bg
CF2 = 9.0
COHFAC = mp.cohfac                   ; Set Coherence Factor.
EBKG = mp.bgerr                      ; Set background error est.
V = fltarr(5,3)                      ; create array: V
EM = fltarr(5)                       ; create array: EM
ER = fltarr(5)                       ; create array: ER
EE = EM                              ; create array: EE
F = EM                               ; create array: F
A = fltarr(6,600)                    ; create array: A

XSAV = X
velfact = 2.9979e05/sdata.wl
X = velfact*(X-sdata.wl)
xje = X(JE)
X = X - X(JE)

QDELTA = (X(J2) - X(J1))/FLOAT(J2 - J1) ;
for II = J1,J2 do begin               ; For each integration point:
    D = (C(II) - Y(II))                ; Calculate Continuum - Flux
    B = C(II) - BKG                    ; and Continuum - Background.
    A(0,II) = D/(B + EBKG)              ; Use these to find the fract-
    A(1,II) = D/B                      ; ion of missing flux, taking
    A(2,II) = D/(B - EBKG)             ; into account BG errors.
    X1 = X(II) - X0                    ; Calculate weighting for
    X2 = X1^2                          ; 1st and 2nd moments.

    for J = 0,2 do begin
        V(0,J) = V(0,J) + A(J,II)      ; Update 0th, (equiv. width)
        V(1,J) = V(1,J) + A(J,II)*X1  ;      1st, and
        V(2,J) = V(2,J) + A(J,II)*X2  ;      2nd moments for the

```



```

                                ; three BG cases.
    endfor
  endfor
  for J = 1,3 do begin
    V(1,J-1) = V(1,J-1)/V(0,J-1) ; Normalize 1st & 2nd mom.
    V(2,J-1) = V(2,J-1)/V(0,J-1) - V(1,J-1)^2
  endfor
  EM(0) = FLOAT(J2 - J1 + 1)/COHFAC
  for j=1,4 do EM(j) = EM(j-1)*EM(0)
  for J = 0,4 do F(J) = ABS(V(J,2) - V(J,0))/2.
  F(0) = F(0)*QDELTA
  JCENT = (J1 + J2)/2
  SIOC2 = 1./(mp.SNR^2)
  SCOC2 = SIOC2/FLOAT(mcntrl.MNE)
  T1 = SIOC2*EM(0)
  T2 = SCOC2*(EM(0) - V(0,1)/COHFAC)^2
  EE(0) = COHFAC*SQRT(T1 + T2)
  VB2 = ((V(1,1) - X(JCENT) + X0)/QDELTA)^2
  V3Q = V(2,1)/QDELTA^2
  EE(1) = COHFAC*SQRT(SIOC2*(EM(2)*CF2/12. + EM(0)*VB2) $
    + SCOC2*EM(1)*VB2)/V(0,1)
  Q = EM(2)*CF2/12. + EM(0)*VB2
  EE(2) = COHFAC*SQRT(SIOC2*(EM(4)*CF2^2/80. + EM(2)*CF2*VB2/2. $
    + EM(0)*VB2^2 - 2.*V3Q*Q $
    + EM(0)*V3Q^2) + SCOC2*(Q - EM(0)*V3Q)^2)/V(0,1)
  EE(2) = EE(2)*QDELTA
  for J = 0,4 do begin
    EE(J) = EE(J)*QDELTA
    ER(J) = SQRT(EE(J)^2 + F(J)^2)
  endfor
  V(0,1) = V(0,1)*QDELTA
  sdata.eqw = V(0,1)/velfact ; Save Equivalent Width (A).
  sdata.me = ER(0)/velfact ; Save its error.
  sdata.owl = (V(1,1)+xje)/velfact + sdata.wl ; Save observed wavelength.
  sdata.fm = V(1,1)+xje ; Save first moment.
  sdata.fme = ER(1) ; Save its error.
  sdata.sm = V(2,1) ; Save 2nd moment.
  sdata.sme = ER(2) ; Save its error.
  mcntrl.e0 = F(0)/velfact ; BG contrib. to EQW error.
  mcntrl.e1 = F(1) ; BG contrib. to 1st error.
  mcntrl.e2 = F(2) ; BG contrib. to 2nd error.

; ----- Calculate the Optical Depths -----

  X = X + X(JE) ; Change origin of vel. back.
  SCOC = SQRT(SCOC2)
  B = C(J1:J2) - BKG
  D = C(J1:J2) - Y(J1:J2)
  A(2,0:J2-J1) = D / B ; TAU's for no BG error.
  CCOR = C*SCOC
  D = D - CCOR
  B = B - CCOR
  A(1,0:J2-J1) = D / (B + EBKG) ; Minimum TAU's.

```

```

D = D + 2.*CCOR
B = B + 2.*CCOR
A(3,0:J2-J1) = D/(B - EBKG) ; Maximum TAU's.
A = 1.0 - A(*,0:J2-J1)
A = -1.0*ALOG(A > 4.e-5) ; Limit the range.
A = ALOG10(A > 0.005)
X = XSAV ; Return X to wavelength.
A(0,0:J2-J1) = X(J1:J2) ; Wavelengths for TAU's.
A(4,0:J2-J1) = Y(J1:J2) ; Store Spectrum.
A(5,0:J2-J1) = C(J1:J2) ; Store Fit Continuum.
TAU = A ; All info. -> TAU array.
RETURN ; back to INTGRT
END ; EQUIVW ;
;

```

```

-----
; This routine sets plotting configuration throughout M.S.L.A.P. It sizes
; the plot windows, correctly adjusting for the environment. It also directs
; the plots to various terminals or hard copy devices.
;
; HC HardCopy flag -1 Close Hardcopy device & set to terminal.
; 0 Terminal or console.
; 1 HardCopy Unit - currently PostScript.
;
; t10 Title for window 0.
; t12 Title for window 2.
; wtype open new window of various types.
; -2 => No New window - reset to default plot parameters.
; - ALSO: Do not produce CURSORS.
; -1 => No New window - reset to default plot parameters.
; 0 => window 0 ----- the large, main plotting window.
; 1 => windows 0 & 1 - MAIN plus ONE for the cursor coords.
; 2 => window 2 ----- the expanded results window.
; 3 => window 3 ----- small instructions window, then do
; an immediate return.
;
; kdev Terminal Device Type either a 'sun' or 'xterm' is returned.
; xmarg Sets !x.margin(0), the left edge of the plot.
; xmarg = 'large' => more space to WMENU functions.
; Note: xmarg = 'large' overrides the wtype variable.
;
-----
pro plotconfig,HC,t10,t12,wtype,kdev,xmarg

```

```

if wtype eq 3 then begin ; Only open window 3
    window,3,title='',xpos=450,ypos=830,xsize=450,ysize=40 ; for intructions.
    return ; RETURN.
endif

kdev = getenv('TERM') ; Get terminal type.
if kdev ne 'sun' then kdev='xterm' ; Assume X window or SUN

if HC lt 0 then begin ; Finish HC plotting.
    device,/CLOSE ; Close device/PS-file.
endif

```



```

device,/INCHES,YOFFSET=2.8,YSIZE=7.5 ; Adjust plot area.
!y.margin(0) = 24. ; Reserve text area.
!x.margin(0) = 10. ; Use all hor. space.
endif

if wtype eq -2 then return ; Avoid making cursors.

if HC eq 0 then device,/CURSOR_CROSSHAIR ; Graphical cursors &
if HC eq 0 then TVCRS,0.5,0.5,/NORMAL ; place on screen.

RETURN
END ; plotconfig

;***** COG.PRO *****
;
; CURVE OF GROWTH - for workstations
; - Input input files are from MSLAP
;
; By Charles L. Joseph: 5/22/79
; Latest Modification: 10/19/90
;
;
;*****
PRO COG,STAR,date,libr
nulld = { noda, el: bytarr(15), iaf: long(0), wl: 0., f: 0., vel: 0., eqw: 0., $
me: 0., fm: 0., fme: 0., sm: 0., sme: 0., com: bytarr(10), up: fltarr(30) }
dtl = replicate({ noda }, 200) ; Make array for storage
kdev = getenv('TERM')
FIREUP,IAF,DV,STAR,dtl,n_opt,c_opts,s_opt,nns,iafb4,cch,knone
wtd = wmenu(c_opts,title=0,init=1) ; Input What-To-Do Flag
if wtd eq 4 then stop ; Stop - Temporary Halt.
if (wtd lt 1) OR (wtd ge 7) then goto,EXIT ; Exit Options.
if wtd gt 2 then wtd = 2 ; Temp. void options.
symnum = intarr(9) ; Zero plot symbol table
snum = intarr(9) ; Zero species table.
cdsav = fltarr(9) ; Column Density mstr tab
melsv = fltarr(9,2,20) ; Mean Error mstr table.
cogi = string(80)
STRT: ; ----- Starting place for 1st species -----
INUM = 0 ; Initialize Image Number
eqslv = fltarr(9,20) ; log(EQW) master table.
wfsav = fltarr(9,20) ; log(Wf) master table.
if (wtd eq 1) OR (wtd eq 2) then begin ; Start with 1st data.
which = wmenu(n_opt,title=0,init=1) ; Get species choice.
if which eq knone then goto, EXIT ; Choice: "No More".
snum(INUM) = which ; Save Species Number
iaf = iafb4(which)
tst = where(cch eq iaf) ; Computer's Choice for
sz = size(tst) ; the theo. CogNumber
if (sz(0) eq 0) then cnum=3 else cnum=tst(0)
openr,1,libr+'/coginfo.tab'
for k=0,cnum do readf,1,cogi

```

```

close,1
  cwf = float(strmid(cogi,71,7))
  print,cwf
  plotconfig,0,cogi,' ',0,kdev,'large' ; Open type 0 window.
;      !x.margin(0) = 30. ; Make plot more square.
;      if kdev eq 'xterm' then !x.margin(0)=50. ; X window a little diff.
  symnum(INUM) = wmenu(s_opt,title=0,init=1) ; Get symbol choice.
endif
whichcogs,nns,NS,NT ; Which C.O.G.'s to use?
T = 0. & B = 0. & L = 12. ; Set plot for autoscale.
START: ; ----- Starting place for Additional Species -----
  iaf = long(0) ; iaf code id's species.
print,iafb4(which),long(iafb4(which))
  if (which gt 0) then iaf = long(iafb4(which))
  GRAB,STAR,C,IAF,WF,EQL,MEL,WAF,F,W,EQW,NL,ML,ME,L,dtl,cnum
wfsav(INUM,0:NL-1) = WF ; Update master log(Wf).
eqlsv(INUM,0:NL-1) = EQL ; Update master log(EQW).
melsv(INUM,0:1,0:NL-1) = MEL
  IF NL LT 0 THEN GOTO, EXIT

LOOP: ; ----- Master LOOP to shift Obs. relative to theo. C.O.G. -----

  LINES,A,NT,NS,M,H,L,T,B,WO,EQL,libr,which ; DRAWS C.O.G.
plotem,wfsav,eqlsv,symnum,melsv,n_opt,snum,INUM ; Overplot Obs. data.
PRINT,' '
print,'Use left mouse to locate a starting data position for translation'
print,'Use right mouse to bring up a MENU of options.'
  CURSOR,X1,IY
if kdev eq 'xterm' then wait,1 ; X window? Slow down.

; ----- Master Branching Section inside LOOP -----

if !ERR eq 4 then begin ; If right mouse, then
  wtd = wmenu(c_opts,title=0,init=2) ; call COG Menu.
  if wtd eq 4 then stop ; Stop - Temporary Halt.
  if wtd le 0 then goto, LOOP
  if (wtd lt 1) OR (wtd ge 7) then goto,EXIT ; Exit Options.
  if (wtd eq 1) then begin ; Get NEW 1st data set.
    tst = 1 ; OK, if not much data.
    if INUM gt 1 then begin ; Check before data loss.
print,string(7B)
print,'Warning: Previous Column densities will be erased'
tst = wmenu(['Confirm','Yes','No'],title=0,init=2)
    endif
    if tst eq 1 then goto,STRT ; Confirmed, start over.
    goto, LOOP ; Continue on.
  endif
  if (wtd eq 2) then begin ; Add more sets of data.
    which = wmenu(n_opt,title=0,init=1) ; Get species choice.
    if which eq knone then goto, LOOP ; Choice: "No More".
    INUM = INUM + 1 ; Incr. Image Number.
    snum(INUM) = which

```

```

        symnum(INUM) = wmenu(s_opt,title=0,init=1) ; Get symbol choice.
        goto, START                               ; Go get new set of obs.
    endif
    if wtd eq 3 then goto, HC                       ; Go make hardcopy?
        if wtd eq 5 then $                         ; Adjust plot limits?
            READ,'What are Xmin,Xmax,Ymin,Ymax?',L,H,B,T
    if wtd eq 6 then whichcogs,nns,NS,NT          ; Change # of curves?
    if (wtd eq 5) OR (wtd eq 6) then goto,LOOP
        endif ; ----- END Branching Section -----

print, ' '
print,'Use left mouse to indicate new location for the translation.'
    CURSOR,IX,IY
    if kdev eq 'xterm' then wait,1                ; X window? Slow down.
        WF=WF+IX-I1
    wfsav(INUM,0:NL-1) = WF
        LINES,A,NT,NS,M,H,L,T,B,WO,EQL,libr,which ; DRAWS C.O.G.
    plotem,wfsav,eqlsv,symnum,melsv,n_opt,snum,INUM ; Overplot Obs. data.
    print,'Total SHIFT: ',wf(0)-waf(0)
    cdl = wf(0) - waf(0) + cwf
    cdsav(INUM) = cdl
    print,'Log Column Density: ',cdl,' for current set of observations'
    goto, LOOP

HC:    PRINT, ' '
;        !x.margin(0) = 10.                        ; Set left margin back.
    plotconfig,1,' ',' ',-1,kdev,' '            ; Set up for hardcopy.
        LINES,A,NT,NS,M,H,L,T,B,WO,EQL,libr,which ; DRAWS C.O.G.
    if nt gt 5 then xyouts,0.7,0.66,'b Values',/NORMAL $ ; Position & write
    else xyouts,0.7,0.6,'b values',/NORMAL      ; Doppler broadening vals.
    for k=NS,NS+NT-1 do begin
        if nt gt 3 then yht = 0.63-0.02*(k-NS)
        if nt le 3 then yht = 0.57-0.02*(k-NS)
        xyouts,0.69,yht,DV(k)+' km/s',/NORMAL
    endfor
    plotem,wfsav,eqlsv,symnum,melsv,n_opt,snum,INUM ; Overplot Obs. data.
    !y.margin(0) = 4                             ; Enable text area.
    spe = 'Obs. data from file: '+STAR+'.DTL'    ; Create string.
    xyouts,0.1,0.28,spe,/NORMAL                 ; Print Data file name.
    xyouts,0.85,0.28,date,/NORMAL              ; Print date of analysis.
    xyouts,0.1,0.26,cogi,/NORMAL               ; Print Theo. COG info.
    for k=0,INUM do begin                      ; For each species, write
        yht = 0.22 - 0.02*k                    ; log N, column density.
        spe = strtrim(n_opt(snum(k)))          ; Get species name.
        spe = 'log N('+spe+')'
        xyouts,0.1,yht,spe,/NORMAL            ; Print species.
        spe = ' = '+strtrim(string(cdsav(k)))
        xyouts,0.35,yht,spe,/NORMAL          ; Print column density.
    endfor
        IF NL LE 7 THEN STP=NL ELSE STP=7
    plotconfig,-1,' ',' ',-1,kdev,'large'      ; Send plot & -> terminal.
;        !x.margin(0) = 30.                    ; Make plot more square.

```

```

;      if kdev eq 'xterm' then !x.margin(0)=50.      ; X window a little diff.
GOTO,LOOP
EXIT:  wdelete,0
;      !x.margin(0) = 10.      ; Set back to default.
RETURN
END
;
;
;
;
PRO FIREUP,IAF,DV,STAR,dtl,n_opt,c_opts,s_opt,nns,iafb4,cch,known
;***** FIREUP.PRO *****
;
;      TO DETERMINE WHICH C.O.G. IS TO BE USED.
;      AND TO SET VARIOUS PARAMATERS.
;
;      by Charles L. Joseph      5/19/79
;*****
;*****      OPEN THE INFO FILE
;
;
close,1
openr,1,STAR+'.DTL', ERROR = errtst      ; Does file exists?
close,1
app = ' '
if errtst eq 0 then begin
  CLOSE,1
  OPENU,1,STAR+'.DTL'      ; Get old file.
  ML=200
  readu,1,dtl
  iafts = where(dtl.iaf ne 0.)      ; Get # of lines.
  sz = size(iafts)
  if (sz(0) eq 0) then ntl = 0 else ntl = sz(1)
  close,1
endif
plotconfig,0,'Curve of Growth',' ',0,kdev,' ' ; Set up for type 0 window.
i_opt = string(bytarr(26,99))      ; Can handle 99 species.
iafdat = dtl.iaf
iafs = where(iafdat ne 0)
iafb4 = long(intarr(99))
iondat = dtl.el
kk = 1
for k = 1, ntl do begin      ; Search through obs. data.
  nxt = iafts(k-1)      ; Get address of next value.
  iaftst = iafdat(nxt)      ; Put in test variable and
  tst = where(iaftst eq iafb4)      ; see if there are other
  sz = size(tst)      ; identical entries.
  if (sz(0) eq 0) then begin      ; If not, add this species
    iafb4(kk) = iaftst      ; to the list.
    i_opt(kk) = string(iondat(0:9,nxt))
    kk = kk + 1      ; Incr. # of valid entries.
  endif
endfor

```

```

i_opt(0) = 'C.O.G. Choices'
i_opt(kk) = 'No more'
knone    = kk
n_opt    = i_opt(0:kk)           ; Take only sub-array.
iafb4    = iaib4(0:kk)

```

```

;
; ----- Load Table of Available Theo. C.O.G. -----
;

```

```

cch = intarr(5)
cch(0) = 26020           ; Fe II COG
cch(1) = 12020           ; Mg II COG
cch(2) = 25020           ; Mn II COG
cch(3) = 8010            ; O I COG, default
cch(4) = 14020          ; Si II COG

```

```

;
; ----- LOAD THE DOPP VEL VALUES -----
;

```

```

DV = string(BYTARR(3,11))
DV(1) = '1.0'
DV(2) = '1.5'
DV(3) = '2.0'
DV(4) = '3.0'
DV(5) = '4.0'
DV(6) = '6.0'
DV(7) = '10.'
DV(8) = '15.'
DV(9) = '20.'
DV(10) = '30.'

```

```

;
; ----- Load the COG Menu Options -----
;

```

```

c_opts = string(bytarr(26,8))
c_opts(0) = 'Options:'
c_opts(1) = 'Compare 1st Species to COG'
c_opts(2) = 'Add another species to COG'
c_opts(3) = 'Send Plot to Laser Printer'
c_opts(4) = 'Stop - Temporary Halt'
c_opts(5) = 'Adjust Plotting Limits'
c_opts(6) = 'Adjust # of Theo. Curves'
c_opts(7) = 'Quit and Return'

```

```

;
; ----- Load Plotting Symbol Options -----
;

```

```

s_opt = string(bytarr(16,9))
s_opt(0) = 'Plotting Symbols'
s_opt(1) = 'Plus sign'
s_opt(2) = 'Asterisk'
s_opt(3) = 'Filled Circle'
s_opt(4) = 'Diamond'
s_opt(5) = 'Triangle'
s_opt(6) = 'Square'
s_opt(7) = 'X'

```



```

-Opt(8) = 'Open Circle'

;
; ----- To get number of curves options -----
;
nns      = string(bytarr(11,11))
nns(1)   = '< 1 >'
nns(2)   = '< 2 >'
nns(3)   = '< 3 >'
nns(4)   = '< 4 >'
nns(5)   = '< 5 >'
nns(6)   = '< 6 >'
nns(7)   = '< 7 >'
nns(8)   = '< 8 >'
nns(9)   = '< 9 >'
nns(10)  = '< 10 >'
;*****      END OF FIREUP
RETURN
END
;
;
;
;
;
PRO LINES,A,NT,NS,M,H,L,T,B,X,EQL,libr,which
;*****      LINES.PRO      *****
;
;      TO GET AND PLOT THE CURVES FOR C.O.G.
;
;      by Charles L. Joseph                      5/17/79
;*****
      if (T eq 0.) AND (B eq 0.) then begin
tst = max(EQL)                                ; If NOT Scaling Override
if (tst lt -5) then begin                      ; Test for only weak &
  T = -3.0                                     ; Set plot limits,
  B = -7.0                                     ; accordingly.
  L = 10.
  H = 16.
      end else begin
  T = -2.                                       ; Only strong-lines-case
  B = -6.                                       ; plotting limits.
  L = 11.5
  H = 18.
endelse
endif
ERASE
A = libr+'/tabdata/feii2382.tab'
A = libr+'/tabdata/siii1260.tab'

OPENR,1,A
y = fltarr(10)
yy = fltarr(10,41)

```

```

xx = fltarr(41)
  for k=0,40 do begin
    readf,1,format='(f9.3,10(f13.4))',x,y
  yy(0:9,k) = y
  xx(k)      = x
  endfor
  close,1
  NA=NS+NT-1
  FOR M=NS,NA DO BEGIN
    y = yy(M-1,0:40)
    IF M EQ NS THEN PLOT,xx,y,xrange=[L,H],yrange=[B,T], $
      XTITLE='!3log (Nf!4k!3)',YTITLE='!3log (W!4k!3)'
    IF M GT NS THEN OPLOTT,xx,y
  END
;   XYOUT,400,30,'log N ( cm -2 )'
;   YXOUT,0,600,'log eqw/ lambda'
  CLOSE,1
;   !PSYM=1
  M=0
RETURN
END
;
;
;
;
;
PRO GRAB,STAR,C,IAF,WF,EQL,MEL,WAF,F,W,EQW,NL,ML,ME,L,dtl,cnum
;
;***** GRAB.PRO *****
;
;   TO GET THE EQW DATA FROM THE .DTL FILE
;
;   WRITTEN BY C. JOSEPH      1980
;
;   MODIFIED BY C. JOSEPH AND T. ARMITAGE ON 19 MAY 1983
;   TO HANDLE A SINGLE OBSERVED EQUIVALENT WIDTH
;
;*****
;
;   OPENR,1,STAR+'.DTL'           ; Open data file and read
readu,1,dtl                     ; observed data.
close,1
iafs = where(dtl.iaf eq iaf)     ; Get addresses of desired data.
NL = size(iafs)                 ; Get Number of Lines and put
NL = NL(1)                      ; value in NL.

W = dtl.wl                      ; Get those portions of the
F = dtl.f                       ; .DTL file to be used.
EQW = dtl.eqw
ME = dtl.me

```

```

IF NL gt 1 then begin
  W = W(iafs)           ; Strip out only those measure-
  F = F(iafs)           ; ments for the give species.
  EQW = EQW(iafs)
  ME = ME(iafs)

  and else begin
    tmpw = W(iafs)
    tmpf = F(iafs)
    tmpe = EQW(iafs)
    tmpm = ME(iafs)
    W = fltarr(2)
    F = fltarr(2)
    EQW = fltarr(2)
    ME = fltarr(2)
    W(0:1) = tmpw
    F(0:1) = tmpf
    EQW(0:1) = tmpe
    ME(0:1) = tmpm
    NL = 2
  endelse

  WAF = ALOG10(W+F)      ; Calculate log f-lambda's.
  EQL = ALOG10(EQW/W)   ; Take log of equivalent widths

  MEL=FLTARR(2,NL)
  FOR N=0,NL-1 DO BEGIN
    IF ME(N) NE -1000 THEN MEL(0,N) = (EQW(N)+ME(N))/W(N)
    IF ME(N) NE -1000 THEN MEL(1,N) = (EQW(N)-ME(N))/W(N)
    IF ME(N) EQ -1000 THEN MEL(0,N) = EQW(N)/W(N)
    IF ME(N) EQ -1000 THEN MEL(1,N) = (EQW(N)-17)/W(N)
  END
  MEL=ALOG10(MEL)
  SS = L + 1 - WAF*(F GT 0) ; Shift points so they are on
  WF=WAF+MIN(SS)           ; the COG plot.

FIN: RETURN
END
;
;
;
;
; ***** PLOTEM.PRO *****
; To overplot the various species equivalent widths plus error bars.
; By Charles L. Joseph 5/15/90
; *****

pro plotam,wfsav,eqlsv,symnum,melsv,n_opt,snum,INUM
  asy = findgen(16)*(!PI*2/16.) ; Used for plot symbol.
  DX = fltarr(2)
  for k=0,INUM do begin ; For each species, over
    teql = eqlsv(k,0:19) ; plot with correct symb.
    tst = where(teql ne 0,NL) ; Get addresses of valid
    teql = teql(tst) ; data and strip out.

```

```

twf = wfsav(k,0:19) ;
twf = twf(tst) ; Strip out log(fw).
tme = melsv(k,0:1,0:19) ;
tme = tme(0,0:1,tst) ; Strip out Mean Errors.

if symnum(k) eq 3 then begin ; Make special symbols?
usersym,0.7*cos(asy),0.5*sin(asy),/FILL ; Convert dots to open
tsym = 8 ; circles.
end else begin
usersym,0.7*cos(asy),0.5*sin(asy) ; Option for open circles.
tsym = symnum(k) ; Most cases, take IDL's.
endelse
OXPLOT,twf,teql,PSYM=tsym ; Over plot obs. DATA
for N=0,NL-1 do begin ; Over plot Error Bars.
DX(0:1) = twf(N)
DY = tme(0,0:1,N)
OXPLOT,DX,DY
endfor ; END Error Bars part.
endfor

if INUM gt 0 then begin ; If more than one species,
Y2 = fltarr(2) ; make symbol key at left.
yrng = !y.crange(1) - !y.crange(0) ; Get range of plot so that
xrng = !x.crange(1) - !x.crange(0) ; the KEY can be positioned
xout1 = 0.10*xrng + !x.crange(0)
xout2 = 0.13*xrng + !x.crange(0)
if INUM lt 5 then tymx = 0.75 else tymx = 0.85 ; Lots of obs.? Adjust TOP
DX(0:1) = xout1 ; Stuff 2-elm. X array.
for k=0,INUM do begin ;
if symnum(k) eq 3 then begin ; Make special symbols?
usersym,0.7*cos(asy),0.5*sin(asy),/FILL ; Convert dots to open
tsym = 8 ; circles.
end else begin
usersym,0.7*cos(asy),0.5*sin(asy) ; Option for open circles.
tsym = symnum(k) ; Most cases, take IDL's.
endelse
tdy = (tymx-0.05*(INUM-k))*yrng + !y.crange(0) ; Y-position for next.
Y2(0:1) = tdy ; Stuff 2-elm. Y array.
oplot,DX,Y2,PSYM=tsym ; Over plot the symbol.
spe = n_opt(snum(k))
tdy = tdy - 0.01*yrng
xyouts,xout2,tdy,spe ; Print species name.
endfor
endif ; END KEY-making part.

return
end
;
;
; ***** WHICHCOGS.PRO *****
;
; To determine starting and how many theoretical C.O.G. are to be used.
; By Charles L. Joseph 5/15/90

```

```

; *****
pro whichcogs,nns,NS,NT
print,string(7B) ; Which C.O.G.'s to use?
print,'Indicate the total number of theoretical Curves of Growth
print,'that are to be plotted. (1 - 10)'
print,'The b values are: 1.0, 1.5, 2, 3, 4, 6, 10, 15, 20, and 30'
print,'km/s, respectively'
nns(0) = '# of Curves?'
wait,1
NT = wmenu(nns,title=0,init=10)
if NT lt 1 then NT = 1
print,' '
print,'Indicate the curve number for the C.O.G. with the lowest'
print,'b value to be plotted. (e.g. use 2 to get the 1.5 km/s.)'
nns(0) = 'Starting?'
NS = wmenu(nns,title=0,init=1)
if NS lt 1 then NS = 1 ; Make sure range range
if NT gt (11-NS) then NT = 11 - NS ; makes sense.
return
end
;
;
; ***** END OF COG PACKAGE *****

```

# MSLAP SITE LICENSE AGREEMENT

## TERMS AND CONDITIONS

1. The title and full ownership rights to each and every part of the Modular Spectral Line Analysis Program (hereinafter MSLAP) shall remain the sole property of Charles L. Joseph and Edward B. Jenkins (hereinafter the authors). You shall acquire no rights in the Program other than as expressly granted in this Agreement. Each portion of MSLAP constitutes valuable proprietary assets of the authors, embodying substantial creative efforts and significant expenditure of time.

YOU MAY NOT USE, COPY, TRANSFER, OR MODIFY THE PROGRAM OR ANY COPY, MODIFICATION, OR MERGED PORTION THEREOF, IN WHOLE OR IN PART, EXCEPT AS EXPRESSLY PROVIDED IN THIS AGREEMENT. YOUR LICENSE TO USE MSLAP IS AUTOMATICALLY TERMINATED IF YOU TRANSFER POSSESSION OF ANY COPY, MODIFICATION OR MERGED PORTION OF THE PROGRAM TO ANOTHER PARTY.

Version 1.0 of MSLAP is licensed free of charge only to sites that are actively engaged in astronomical research for the sole purpose of pursuing astronomical research and that are in compliance with the terms of this Agreement. Possessing MSLAP in whole or in part indicates that the user has accepted all of the terms of this agreement. Violation of any terms of this Agreement shall make you liable for full payment for any damages incurred to the authors.

The provisions of this section 1 shall survive any termination of this Agreement.

2. The authors provide MSLAP and license its use on a single site bases. A site is defined for the purposes of this Agreement as a single Department, Center, or Institute that is located on a single campus of a university or government agency. Permission is granted to you to use or to copy MSLAP to any machine on your site that is controlled by your Department, Center, or Institute, provided such action is in full compliance with all of the terms of this Agreement.

3. MSLAP IS PROVIDED 'AS IS' WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED. NO WARRANTY OF ANY KIND IS IMPLIED OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR APPLICATION. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU AND NOT THE AUTHORS NOR THEIR AGENTS.

In no event will the authors or their agents be liable to you for any damages arising out of use or inability to use MSLAP, even if the authors or their agents have been advised of the possibility of such damages, or for any claim by any other party. If any third-party claims arise against you, you shall bare all such liabilities.

4. Standard version 1.0 of MSLAP consists of the 34 routines licensed to you that initially are contained in the files: main.mslap, master.aux, compare.pro, posto.pro, edatdl.pro, mantau.pro, intgrt.pro, plotconfig.pro, and cog.pro as well as all of the associated tables found in files with the extension ".tab". You are permitted to alter this standard source code as you desire for your personal or site's use, but you must maintain the original copyright notice in both the source file and in the initial start up of MSLAP, and you may NOT distribute the altered code off site without written authorization from Charles L. Joseph. If your site has 2 or more visitors in a given year who use MSLAP, you are considered a "guest user facility", and you must comply with the conditions spelled out in section 8 of this Agreement. Any alterations to the standard source code must also meet the documentation requirements spelled out in section 6 of this Agreement.

set of routines, contained in the file mslap.pro, are designed for individualized customization of MSLAP. You are permitted to create new modules (subroutines) of the USERPROG's or to make adaptations of the routines originally found in the file mslap.pro, and to distribute these customized modules to other sites, provided the new code meets the requirement in section 6.

6. Any and all modifications must comply with the following:

- i. All modified code must describe all parameters in each procedure definition statement.
- ii. All modified code must indicate the original author, author's affiliated institution, and a modification history including each and every major participant involved in the modification.
- iii. If the altered code is designed to override a particular behavior of standard MSLAP, a concise description of the former and new behavior must be output in a conspicuous manner and also included in the source code documentation.
- iv. The ALT\_INT routine has special rules. This routine is used to provide an alternate type of integration. The following disclaimer must appear in a conspicuous manner whenever the routine is called. Standard MSLAP with a working instruction window at the lower left of the screen automatically produces this disclaimer. The following lines from that disclaimer are listed below and may NOT be removed from the file intgrt.pro.

```
print,' ',string(7B)
print,string(7B),'No Alternate Integration Routine is Provided'
print,'in standard M.S.L.A.P. - version 1.0'
print,' '
print,'Origin of the Alt. Integration Routine is as follows:'
```

7. A library of user-generated data-getting routines are made available to the astronomical community and users are encouraged to add to this library. MSLAP supports any combination of 5 or less of these routines at one time. (See the MSLAP Documentation Manual for the general implementation of dget1, dget2, dget3, dget4, and dget5 procedures and their appropriate protocols.) All "dget" routines are considered to be formally outside of MSLAP and not subject to the terms of this Agreement. However, it is strongly recommended that the documentation rules listed above be implemented to maximize the overall utility of any user-contributed routines for data retrieval.

8. If you represent a guest user facility, defined as any site where more than 2 visitors use MSLAP in a given year, you are not permitted to incorporate modifications that remove the modularity of MSLAP, thus making it difficult for guest users to customize MSLAP. For example, you must avoid reserving more than 10 of the 30-element vector called "up" (UserParameter) since such action would severely inhibit the ability of a user to store and manipulate customized calculations.

Some version of the original file mslap.pro must be retained and be made available to individual users for their personal customization of MSLAP.

If you modify any portion of MSLAP including the original file mslap.pro to override a particular behavior of standard MSLAP, a concise description of the former and new behavior must be output in a conspicuous manner when MSLAP is running and also must be included in the source code documentation.

9. Permission is granted to incorporate MSLAP into a large driver program, provided the name MSLAP and the original copyright notice is maintained in any menu or other solicitation to activate MSLAP. Any incorporation into a driver program is also subject to sections 4, 5, 6, and 8.

9. Your site is considered to be a Distribution Center and you may transfer copies of the STANDARD, version 1.0 of MSLAP to other astronomical sites only if this section of this agreement is complete with all necessary signatures.

\_\_\_\_\_  
Site Name and Address of the Department, Center, or Institute that is to be a a Distribution Center.

\_\_\_\_\_  
Date and Signature of authorized representative of the above site.

\_\_\_\_\_  
Name and Position of the person signing above. (Please print or type)

\_\_\_\_\_  
Date and Signature of Charles L. Joseph, the first author.

11. The license granted under this Agreement is effective until terminated. You may terminate this license by destroying all copies, modifications and merged portions of MSLAP in your possession and notifying Charles L. Joseph in writing of such destruction and termination.

The license granted under this Agreement will terminate if you violate any of the terms and conditions of this Agreement. The first author, Charles L. Joseph, reserves the right to terminate this Agreement, if he believes that any of the terms and conditions of this Agreement have been violated. Notice of such termination shall be made in writing to you. You agree upon such termination to destroy promptly all copies, modifications, and merged portions of MSLAP in your possession and to certify to the first author that such action has been taken.

12. None of your rights, duties or obligations under this Agreement may be sold, sublicensed, or otherwise assigned without prior written consent of Charles L. Joseph.

13. YOU ACKNOWLEDGE THAT (a) YOU HAVE READ THIS ENTIRE AGREEMENT AND AGREE TO BE BOUND BY ITS TERMS AND CONDITIONS; (b) THIS AGREEMENT IS THE COMPLETE AND AND EXCLUSIVE STATEMENT OF THE UNDERSTANDING AND CONTRACT BETWEEN US AND SUPERSEDES ANY AND ALL PRIOR ORAL OR WRITTEN COMMUNICATIONS RELATING TO THE SUBJECT MATTER HEREOF; AND (c) THIS AGREEMENT MAY NOT BE MODIFIED, AMENDED OR IN ANY WAY ALTERED EXCEPT IN WRITING AND SIGNED BY BOTH YOU AND THE FIRST AUTHOR.



## APPENDIX B

## Data Structures in MSLAP

There are 3 structures used by MSLAP, "mp", "mcntrl", and "dtl". All three are global in nature, meaning they can be accessed from any subroutine. The first, "mp" is the primary structure and the one most frequently used to modify a particular behavior of MSLAP. "mcntrl" is a second structure used primarily for record keeping and some control. Greater care should be exercised in make adjustments to "mcntrl" since MSLAP could become "confused" and may lead to spurious results that may not be obvious. This structure ("mcntrl") is intended for the advanced user of MSLAP. The final structure is "dtl", which contains the measurements that are written to the output data file. This "dtl" structure can be edited using option 4 in the beginning Main MSLAP Menu. Below is a list of the parameters along with a description of each.

-----  
 The primary (non-data storage) structure used by MSLAP is "mp", MslapParameter.  
 -----

mp.dget - a flag used to indicate the presence of a dget routine. During the setup stage, MSLAP calls each dget routine (i.e. dget1, dget2, ..., dget5). At this time mp.cntrl equals -1, a flag to indicate the initial call. If the user-installed dget routine is to be recognized by MSLAP, it must return mp.dget not equal to zero on this initial call. The dummy dget routines leave mp.dget=0.

mp.cntrl - The control flag for MSLAP operations. Some care should be exercised when changing the value of mp.cntrl in any USERPROG, but generally speaking mp.cntrl can be safely set in any DGET.

mp.cntrl = -1 => This is the first time the dget routine is being called. Hardware graphics are configured.

mp.cntrl = 1 => No problems have been encountered in the subroutine. OK flag for storing the measurement.

mp.cntrl = -99 => Abort Condition, MSLAP will exit.

mp.cntrl = -10 => Make HardCopy Flag.

mp.cntrl = 0 => Subroutine Returned as if NEVER CALLED. MSLAP will just replot spectra and start over.

mp.cntrl = 2 => Get NEXT Spectra.

mp.order - is used for echelle spectra. It can be used to select a portion of the spectra when more than one record is placed in a given data file. (Standard MSLAP does not make use of this parameter; it is supplied solely for the user's convenience [e.g. it can be displayed using plotlab1 or 2.] )

mp.CAM - Specifies a camera number. Many satellites have more than one

camera and it is often useful to record this information. (Standard MSLAP does not make use of this parameter; it is supplied solely for the user's convenience [e.g. it can be displayed using plotlab1 or 2.]

**mp.SMO** - a flag that causes the graphics to smooth the data by a 3 point running box car. (This feature is becoming obsolete and may not be available in future versions of MSLAP.) It is best to perform any smoothing inside the user's DGET or USERPROG routine and then adjust the mp.cohfac value accordingly.

**mp.cohfac** - the coherence factor of the noise in the data. (See MSLAP the Manuel for details.) Basically, this is a measure of the linear independence of the one pixel to other nearby pixels. For example, data that have been smoothed by a 5-point running box car would have a coherence factor of 5.

**mp.bg** - the background level, if not zero.

**mp.bgerr** - the 1-sigma uncertainty in the background level. This value is VERY IMPORTANT since in many applications this error is the dominant source. (See the MSLAP Manuel for details.)

**mp.window** - this parameter is used to define a fixed-sized integration window. In data with poor signal-to-noise ratios or in cases where the intrinsic strength of a spectral feature is expected to be weak compared to the noise, this feature of MSLAP allow the user to make unbiased measurements of the signal strength based on info. obtain from other spectral features. This parameter is set and/or used in response to the MENU option which selects the order of the polynomial to fit the continuum. (See the MSLAP Manuel for information regarding the "Predetermined Window" option.)

**mp.DTY** - a parameter with integer values in the range [1, ...,5] which selects which [DGET1, ..., DGET5] routine will be called.

**mp.STAR** - Contains the Output File name. This string plus the ".DTL" extension form the primary output data file that MSLAP creates.

**mp.FNAM** - Contains the Input File name from which the spectral data will be retrieved. This string may include a directory path as well.

**mp.poly** - Holds the order number of the polynomial that was used to fit the continuum.

**mp.SNR** - is the APPARENT Signal-to-Noise Ratio. It is the RMS value of the residuals of the real data minus the polynomial fit. The Real SNR is obtained by dividing mp.SNR by the square-root of the the coherence factor [i.e. Real SNR = mp.SNR/(mp.cohfac<sup>0.5</sup>)]. This

RMS value is added in quadrature with the mp.bgerr to estimate the total uncertainties.

-----  
 A structure called "mcntrl" is primarily used for record keeping while MSLAP is running. It is recommended that only advance users of MSLAP adjust these parameters.  
 -----

mcntrl.I - is the pointer of the number of measurements that have been made. If a "output.DTL" file exists at start up, this file is opened and mcntrl.I is set to the number of non-zero data entries. The user is then prompted to accept or reinitialize this pointer.

mcntrl.date a string holding the date obtained from the system clock.

mcntrl.intopt an integer flag indicating which integrating routine is to be used. Normally, mcntrl.intopt=1. If mcntrl.intopt=0, the ALT\_INT routine is called.

mcntrl.WNE the number of continuum points used in the polynomial fit and in the uncertainty calculations.

mcntrl.ESAV the RMS uncertainty, calculated from the residual differences between the real continuum and the polynomial fit.

mcntrl.e0 Background's contribution to the zeroth moment (Equivalent Width). If multiple measurements are made simultaneously, only the first measurement is stored in this parameter. The same is true for mcntrl.e1 and mcntrl.e2

mcntrl.e1 same as mcntrl.e0, except it is for the first moment.

mcntrl.e2 same as mcntrl.e0, except it is for the second moment.

mcntrl.wtol wavelength tolerance used in testing for a match between the selected observed wavelength and comparison to the USER Look Up Table of rest wavelengths. Normally, this is taken to be 10 spectral elements wide or 0.5 Angstroms, which ever is largest.

mcntrl.lib a string holding the main "library" directory of routines and tabled data.

mcntrl.mtot the largest polynomial that will be fit by MSLAP. Standard MSLAP uses mcntrl.mtot=5. A larger value will automatically fit corresp-

~~dtl.fm~~ Holds all of the First Moments of the profiles.

dtl.fme Holds all of the Errors associated with the First Moments.

dtl.sm Holds all of the Second Moments of the profiles.

dtl.sme Holds all of the Errors associated with the Second Moments.

dtl.com Holds all of the byte arrays containing the information that was entered as comments. This part is similar to dtl.el above.

dtl.up Holds all of the UserParameter Arrays. Each dtl.up holds a 30 element floating-point vector that is reserved for the user's exclusive use.

# MSLAP SITE LICENSE AGREEMENT

## TERMS AND CONDITIONS

1. The title and full ownership rights to each and every part of the Modular Spectral Line Analysis Program (hereinafter MSLAP) shall remain the sole property of Charles L. Joseph and Edward B. Jenkins (hereinafter the authors). You shall acquire no rights in the Program other than as expressly granted in this Agreement. Each portion of MSLAP constitutes valuable proprietary assets of the authors, embodying substantial creative efforts and significant expenditure of time.

YOU MAY NOT USE, COPY, TRANSFER, OR MODIFY THE PROGRAM OR ANY COPY, MODIFICATION, OR MERGED PORTION THEREOF, IN WHOLE OR IN PART, EXCEPT AS EXPRESSLY PROVIDED IN THIS AGREEMENT. YOUR LICENSE TO USE MSLAP IS AUTOMATICALLY TERMINATED IF YOU TRANSFER POSSESSION OF ANY COPY, MODIFICATION OR MERGED PORTION OF THE PROGRAM TO ANOTHER PARTY.

Version 1.0 of MSLAP is licensed free of charge only to sites that are actively engaged in astronomical research for the sole purpose of pursuing astronomical research and that are in compliance with the terms of this Agreement. Possessing MSLAP in whole or in part indicates that the user has accepted all of the terms of this agreement. Violation of any terms of this Agreement shall make you liable for full payment for any damages incurred to the authors.

The provisions of this section 1 shall survive any termination of this Agreement.

2. The authors provide MSLAP and license its use on a single site bases. A site is defined for the purposes of this Agreement as a single Department, Center, or Institute that is located on a single campus of a university or government agency. Permission is granted to you to use or to copy MSLAP to any machine on your site that is controlled by your Department, Center, or Institute, provided such action is in full compliance with all of the terms of this Agreement.

3. MSLAP IS PROVIDED 'AS IS' WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED. NO WARRANTY OF ANY KIND IS IMPLIED OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR APPLICATION. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU AND NOT THE AUTHORS NOR THEIR AGENTS.

In no event will the authors or their agents be liable to you for any damages arising out of use or inability to use MSLAP, even if the authors or their agents have been advised of the possibility of such damages, or for any claim by any other party. If any third-party claims arise against you, you shall bare all such liabilities.

4. Standard version 1.0 of MSLAP consists of the 34 routines licensed to you that initially are contained in the files: main.mslap, master.aux, compare.pro, postp.pro, edatdl.pro, mantau.pro, intgrt.pro, plotconfig.pro, and cog.pro as well as all of the associated tables found in files with the extension ".tab". You are permitted to alter this standard source code as you desire for your personal or site's use, but you must maintain the original copyright notice in both the source file and in the initial start up of MSLAP, and you may NOT distribute the altered code off site without written authorization from Charles L. Joseph. If your site has 2 or more visitors in a given year who use MSLAP, you are considered a "guest user facility", and you must comply with the conditions spelled out in section 8 of this Agreement. Any alterations to the standard source code must also meet the documentation requirements spelled out in section 6 of this Agreement.

5. A set of routines, contained in the file mslap.pro, are designed for individualized customization of MSLAP. You are permitted to create new modules (subroutines) of the USERPROG's or to make adaptations of the routines originally found in the file mslap.pro, and to distribute these customized modules to other sites, provided the new code meets the requirement in section 6.

6. Any and all modifications must comply with the following:

- i. All modified code must describe all parameters in each procedure definition statement.
- ii. All modified code must indicate the original author, author's affiliated institution, and a modification history including each and every major participant involved in the modification.
- iii. If the altered code is designed to override a particular behavior of standard MSLAP, a concise description of the former and new behavior must be output in a conspicuous manner and also included in the source code documentation.
- iv. The ALT\_INT routine has special rules. This routine is used to provide an alternate type of integration. The following disclaimer must appear in a conspicuous manner whenever the routine is called. Standard MSLAP with a working instruction window at the lower left of the screen automatically produces this disclaimer. The following lines from that disclaimer are listed below and may NOT be removed from the file intgrt.pro.

```
print, ' ,string(7B)
print,string(7B),'No Alternate Integration Routine is Provided'
print,'in standard M.S.L.A.P. - version 1.0'
print, '
print,'Origin of the Alt. Integration Routine is as follows:'
```

7. A library of user-generated data-getting routines are made available to the astronomical community and users are encouraged to add to this library. MSLAP supports any combination of 5 or less of these routines at one time. (See the MSLAP Documentation Manuel for the general implementation of dget1, dget2, dget3, dget4, and dget5 procedures and their appropriate protocols.) All "dget" routines are considered to be formally outside of MSLAP and not subject to the terms of this Agreement. However, it is strongly recommended that the documentation rules listed above be implemented to maximize the overall utility of any user-contributed routines for data retrieval.

8. If you represent a guest user facility, defined as any site where more than 2 visitors use MSLAP in a given year, you are not permitted to incorporate modifications that remove the modularity of MSLAP, thus making it difficult for guest users to customize MSLAP. For example, you must avoid reserving more than 10 of the 30-element vector called "up" (UserParameter) since such action would severely inhibit the ability of a user to store and manipulate customized calculations.

Some version of the original file mslap.pro must be retained and be made available to individual users for their personal customization of MSLAP.

If you modify any portion of MSLAP including the original file mslap.pro to override a particular behavior of standard MSLAP, a concise description of the former and new behavior must be output in a conspicuous manner when MSLAP is running and also must be included in the source code documentation.

9. Permission is granted to incorporate MSLAP into a large driver program, provided the name MSLAP and the original copyright notice is maintained in any menu or other solicitation to activate MSLAP. Any incorporation into a driver program is also subject to sections 4, 5, 6, and 8.

10. Your site is considered to be a Distribution Center and you may transfer copies of the STANDARD, version 1.0 of MSLAP to other astronomical sites only if this section of this agreement is complete with all necessary signatures.

---

Site Name and Address of the Department, Center, or Institute that is to be a Distribution Center.

---

Date and Signature of authorized representative of the above site.

---

Name and Position of the person signing above. (Please print or type)

---

Date and Signature of Charles L. Joseph, the first author.

11. The license granted under this Agreement is effective until terminated. You may terminate this license by destroying all copies, modifications and merged portions of MSLAP in your possession and notifying Charles L. Joseph in writing of such destruction and termination.

The license granted under this Agreement will terminate if you violate any of the terms and conditions of this Agreement. The first author, Charles L. Joseph, reserves the right to terminate this Agreement, if he believes that any of the terms and conditions of this Agreement have been violated. Notice of such termination shall be made in writing to you. You agree upon such termination to destroy promptly all copies, modifications, and merged portions of MSLAP in your possession and to certify to the first author that such action has been taken.

12. None of your rights, duties or obligations under this Agreement may be sold, sublicensed, or otherwise assigned without prior written consent of Charles L. Joseph.

13. YOU ACKNOWLEDGE THAT (a) YOU HAVE READ THIS ENTIRE AGREEMENT AND AGREE TO BE BOUND BY ITS TERMS AND CONDITIONS; (b) THIS AGREEMENT IS THE COMPLETE AND AND EXCLUSIVE STATEMENT OF THE UNDERSTANDING AND CONTRACT BETWEEN US AND SUPERSEDES ANY AND ALL PRIOR ORAL OR WRITTEN COMMUNICATIONS RELATING TO THE SUBJECT MATTER HEREOF; AND (c) THIS AGREEMENT MAY NOT BE MODIFIED, AMENDED OR IN ANY WAY ALTERED EXCEPT IN WRITING AND SIGNED BY BOTH YOU AND THE FIRST AUTHOR.