MJSANL

The Positive-Ion Least-Squares Fitting Routine for the Voyager PLS Instrument

September 1, 2023

(Equations are created using *MathType*: *https://www.wiris.com/en/mathtype/*)

Contents

Introduction 4
History of the Port 6
GETDATA 6
FULL UP MODE 6
Examples of running *mjsanl* 7
Example 1: Fitting an *M-mode* cruise spectra to a single proton using *ICPLT* = 4 7
Example 2: Fitting an *L-mode* cruise spectra to a single proton using *ICPLT* = 3 11
Example 3: Fitting an *L-mode* cruise spectra to a proton and alpha using *ICPLT* = 3 13
Example 4: Fitting an *M-mode* cruise spectra to a proton and alpha using *ICPLT* = 3/4 15
Example 5: Fitting an *M-mode* spectra to two protons and alpha full using *ICPLT* = 4 17
Example 6: Fitting an *M-mode* spectra to a proton late in mission using *ICPLT* = 4 19
Example 7: Fitting an *L-mode* heliosheath spectra to a proton using *ICPLT* = 4 20
Example 8: Fitting an *M-mode* from Voyager 1 at Jupiter using *ICPLT* = 5 21
Example 9: Fitting an *M-mode* from Voyager 2 at Jupiter using *ICPLT* = 5 22
Example 10: Fitting an *L-mode* from Voyager 1 at Saturn using *ICPLT* = 5 22
Example 11: Fitting an *M-mode* from Voyager 2 at Saturn using *ICPLT* = 5 23
Example 12: Fitting an *L-mode* from Voyager 2 at Saturn using *ICPLT* = 4 24
Example 13: Fitting an *L-mode* from Voyager 2 at Uranus using *ICPLT* = 4 25
Example 14: Fitting an *L-mode* from Voyager 2 at Neptune using *ICPLT* = 4 26
Example 15: Fitting an *M-mode* from Voyager 1 Cruise post-Jupiter using *ICPLT* = 4 27
Example 16: Fitting an *M-mode* during Barnett's Cruise Maneuver using *ICPLT* = 4 27
List of Routines 29
Routines Associated with VMV's PMODEL 30
cpresp.f 30
cuprsp.f 30
pmodel.f 30
xmax.f 30
Routines Associated with Barnett's Response Function 30
abccur.f 30
armain.f 31
arside.f 31
conrd.f 31
cupint.f 31
dcur.f 31

displd.f 31
displn.f 31
dphi.f 31
fun.f 32
labbi.f 32
labcur.f 32
ldcur.f 32
redart.f 32
transd.f 32
transn.f 32
xbarp.f 32
xcur.f 32
xecurnt.f 32
xcurnt.f 33
Routines Associated with Non-Linear Least Squares Fit 33
eigsrt.f 33
resolv.f 33
tderf.f 33
tqli.f 33
tred2.f 33
xunpck.f 33
Unpacking fit parameters. See the routine xpack.f. 33
xanal.f 33
xfull.f 33
xfncdv.f 33
xinv.f 33
xjc.f 33
xparm.f 33
xmjsft.f 33
xpack.f 34
xsimul.f 34
xtest.f 34
Routines Associated with Data 34
iandh.f 34
kntcur.f 34
kntflt.f 34
kntpk.f 34
limits.f 34
mjsanl.f 34
nxtmod.f 34
nxtvgrf.f 34
plsbeg.f 34
setjtl.f 35
vgrtrd.f 35
writedat.f 35
Routines Associated with Trajectory/Rotation Matrices 35

epsiln.f 35
phase2.f 35
phase3.f 35
onaxr.f 35
readdata.f 35
rotmtx.f 36
treadbatf.f 36
vsccp.f 36
vtrns.f 36
vxyzsp.f 36
Miscellaneous Routines 36
calanl.f 36
cave.f 36
dummy.f 36
dati.f 36
gi.f 36
l1x.f 36
mjstem.f 36
nchar.f 36
numstr.f 36
plcl.f 36
pltst.f 37
sclr.f 37
setdata.f 37
so31286.f 37
sodint.f 37
ss.f 37
stncur.f 37
strequ.f 37
teqab.f 37
teqatr.f 37
timer.f 37
vctr.f 37
xtimer.f 37
xtyme.f 37
mvc.c 37
mvc5.c 37

Introduction

MJSANL is a set of code put together by Dr. Ralph McNutt, Jr, with extensive help from the senior programmer in the space plasma group, Dr. George Gordon, Jr, research scientists, post-doctoral fellows, and graduate students. This document is an explanation of the *MJSANL* code, used to do non-linear fits of the positive ion data in the interplanetary medium and at the four magnetospheres that Voyager encountered. The acronym *MJS* comes from *Mariner Jupiter Saturn*, as the mission was known until launch. The two spacecraft were only expected to reach Jupiter and Saturn. We have not gotten the *FULL UP* data analysis mode of *MJSANL* working, but we have gotten the "standard" mode working. We have made every attempt not to disrupt those parts of the code that do the *FULL UP* analysis (see the Section *FULL UP MODE* below).

The non-linear fits of the positive ion data are straightforward but complicated: given a measured set of ion spectra, fit those currents to a model (usually a convected Maxwellian), and rotate the velocity vector from spacecraft coordinates into the appropriate inertial set of coordinates, properly accounting for the velocity of the spacecraft. The basic ingredients we need are a file containing the spectra and a file containing the trajectory and orientation of the spacecraft, and programs to analyze that data. The complexity comes because the data format and the trajectory format changes depending on whether we are in cruise or at a planetary encounter, and because we use a variety of methods for fitting the spectra depending on the response function we assume for the instrument.

The execution in *MJSANL* is driven by interactive keyboard input. It was originally developed on SUN workstations running *Solaris*. The present code is a port of the *Solaris* code to *CentOS* begun in July of 2017, after Voyager had crossed the termination shock (in 2007) but before it crossed into the interstellar medium (in 2018). This was in conjunction with an effort to write a read routine for the Experiment Data Records (*EDR*'s) on a more modern Linux system, *CentOS 7.9*.

The details of the instrument itself are discussed in a separate document, *Voyager Interstellar: A Description of the Voyager Plasma Science Instrument and Data* (referred to as *VI-PLS* below). The method used to perform our least-squares fit is described in Section 3.2.4 of that document.

The documentation, including this document, software and data are contained in the directory *Voyager_PLS_v_01,* currently at MIT in the directory

/nfs/faraday/data/jwb/Voyager_PLS_v_01

Flash drives containing this code and zip files of the code and data with an extraction script are also available. The structure of the *Voyager_PLS_v_01* directory is explained in Section 1.4 of *VI-PLS.* The *MJSANL* code and *Makefile* are contained in the directory *Voyager_PLS_v_01/software_v_01/mjsanl/code*.

The subroutines in *MJSANL* were copied from the SUN workstations, for example, from */nfs/carrington/h3/vgr*, and other places, to a modern workstation *Faraday* running *CentOS 7.9.* They were recompiled with modern compilers (*gfortran* and *gcc*), with the help of a student,

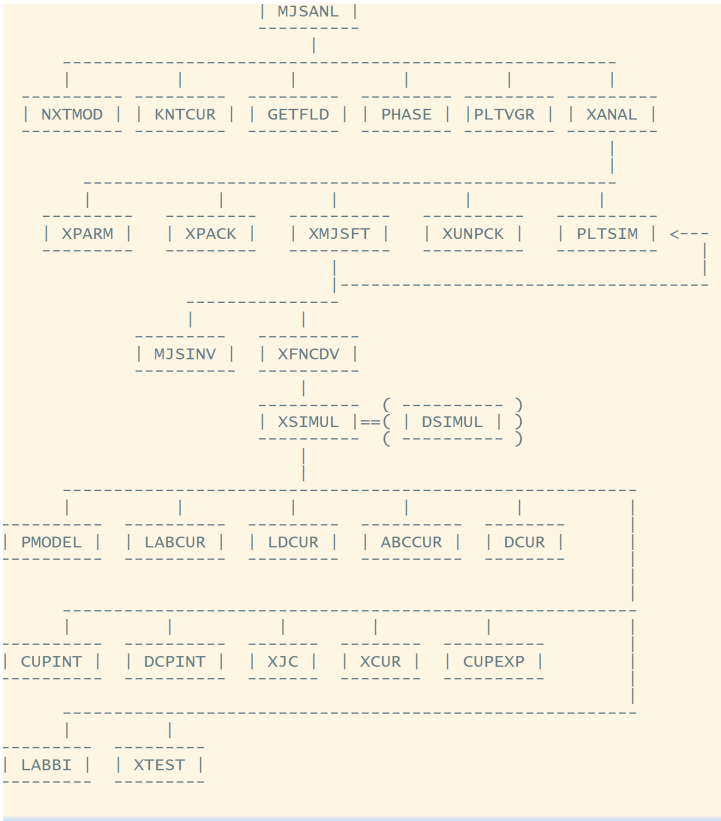
Paula Garcia-Galindo. After many iterations, the current version of *MJSANL* processes in the "standard" mode both cruise data and encounter *data* (but not in the *FULL UP* mode)

There are eighty-eight *Fortran* routines in *MJSANL*, and two *c* routines. We give a cursory summary of what each routine does in the *List of Routines* section below. The *Makefile "make"* in *software_v_01/mjsanl/code* compiles all of these and puts them in an executable module *test*. As part of the *PDART* program, most of our binary data files on *Solaris* have been converted to *ascii* files, both data and trajectory, for the encounters and the 45 years of cruise and interstellar medium data, and these are the files used in the *Examples* below (see Voyage Memos 193, 200, 202-207, 208, and 214 in *Voyager_PLS_v_01/documentation_v_01/voyager_memos/pdf*).

To make sure that the *MJSANL* programs do not contain non-portable features, these materials have been transferred to a laptop running *Fedora 38*, recompiled, and run successfully in that environment.

These *CentOS* programs have been tested by doing fits to cruise data, and to magnetospheric data at all the planets, ***comparing with historic and current fits done on the Solaris system***, as explained in the *Examples* below. Note that *Solaris* programs are reading binary data and trajectory files, whereas the corresponding *CentOS 7.9* programs are reading our *ascii* versions of those binary files. The generally good agreement we find between the two means that the binary files have been converted successfully and that the port of the program is successful.

The schematic below, from Ralph McNutt, gives an overall view of the structure of the program.



History of the Port

The code for *mjsanl* was assembled using code in the various Voyager code libraries. For completeness, I have included "historic" code from the *vgr/src/mjs*, *vgr/src/pls, vgr/src/sedr, vgr/src/vec*, and *vgr/src/vgr* libraries in the directory *Voyager_PLS_v_01/software_v_01/historic_libraries*. Using these libraries, we assembled a complete set of *mjsanl* routines and ported them to *CentOS*. The code for our first successful attempt to run the code on *CentOS* is contained in the directory *Voyager_PLS_v_01/software_v_01/mjsanl/initial_port_code*. Note that this initial code had data and trajectory information input from Fortran files; we were not reading real data and trajectory information. Routines to read real data and trajectory information were added later. We subsequently eliminated a number of the original *Solaris* subroutines (mostly *CalComp* plotting routines), but the subroutines we eliminated are contained in */software_v_01/mjsanl/deprecated_code*. We have kept copies of those deprecated routines because they may be useful in resurrecting the "*FULL UP MODE*" data analysis (see below).

Note that a major factor in the port was to move the plotting from *CalComp* routines to *gnuplot*. On a given *gnuplot* plot of a measured spectrum and the fit to the measured spectrum, I have displayed the same quantities we have historically displayed, e.g., the measured spectrum, the total fit, and the fit to each component. But I have not used the same color coding for this, and I have not written out the fit parameters on the plot.

Note also that I have made no attempt to provide a means of preserving a hard copy of the fits. All the fits I show in the examples are from using the *Sniping Tool* on *Windows*.

Finally, there was code in the Solaris version of *mjsanl* to redo fits, progress to the next spectrum, and so on. I have *disabled* all those features, and only have put in the capability to fit one spectrum at a time.

GETDATA

When running *mjsanl* on *Solaris*, we initially only specify the year and the day that we want to fit data on (see line 7 of *script_01_solaris* in Example 01 below). The *perl* script *GETDATA* on *Solaris* then looks for the mission (*vgr*) or encounter *spl* data file file that contains that year and day and for the appropriate *ssedr* file, and the user never has to specify the files. We have ***not*** converted *GETDATA* over to *CentOS*, so on *CentOS* we must specify those files in the *mjs.cru* bash script, for example. We find the names of the appropriate files by looking in *Voyager_PLS_v_01/v1/vgr* and *Voyager_PLS_v_01/v1/cruise_ssedr*, similarly for Voyager 2. When dealing with magnetospheric data, we look in *Voyager_PLS_v_01/v1/encounter_spl* and *Voyager_PLS_v_01/v1/encounter_ssedr*, similarly for Voyager 2.

FULL UP MODE

The FULL UP MODE was written by Ralph McNutt, Jr. to study the extent of the contamination of the measured ion currents by feed-through from the hot electrons. In McNutt's words, his intent was as follows:

1. Get the ion currents correct when there is a significant, hot electron background.
2. Use VMV's PMODEL for simultaneously simulating ion and electron currents in L-mode (ion), E1 and E2 (electron).
3. Assess the significance of cross ion/electron feedthrough.

One example of the plots and analysis in the FULL UP mode is given in the *pdf* document *Positive ion fits near Europa* which can be found in *Voyager_PLS_v_01/documentation_v_01/general*.

To assist in the resurrection of the FULL UP MODE (if someone wants to attempt it), I have kept all the relevant routines in *Voyager_PLS_v_01/software_v_01/mjsanl/code* with as few structural changes as I could manage, and moved the deprecated subroutines into *Voyager_PLS_v_01/software_v_01/mjsanl/deprecated_code*. For example, in this deprecated code directory you can find the subroutine *pltele.f*, which was used to "PLOT OUT PLS ELECTRON SPECTRAL DATA FOR USE WITH "EANAL" MAGNETOSPHERIC ANALYSIS PACKAGE". This routine was only called when doing the FULL UP MODE analysis, and I have commented out that call in */software_v_01/mjsanl/code/mjsanl.f* . However, I have kept a copy of that code in the *deprecated_code* directory if someone wants to resurrect that.

In resurrecting the FULL UP MODE it may also be useful to refer to the code in */software_v_01/mjsanl/initial_port_code*. This has the earliest successful effort to make the port to *CentOS* and therefore the least modified versions of the *Solaris* code.

Examples of running *mjsanl*

Example 1: Fitting an *M-mode* cruise spectra to a single proton using *ICPLT* = 4

Below we give a set of examples of running *mjsanl* for cruise and for each planetary encounter. Code and bash scripts for running these examples can be found in, for example:

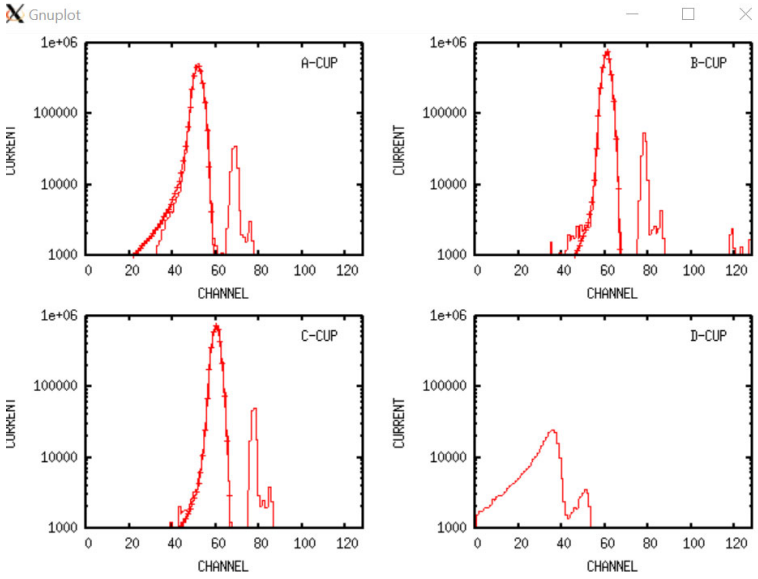
Voyager_PLS_v_01/software/mjsanl/example_01

Below is the list of fitting options the user is given to choose from when running *mjsanl*. The user's response to this is stored in the variable *ICPLT*. I give the values of *ICPLT* chosen in the examples below.

```
SELECT FITTING OPTION: if negative calculate current
      0 ==> DO NOT DO FIT
      1 ==> FIT USING PMODEL
      3 ==> FIT USING XJC (UNITY RESPONSE)
      4 ==> FIT USING ABSO (L OR M MODES - LABCUR,LDCUR)
      5 ==> FIT USING ABSOM (M-MODE ONLY - ABCCUR,DCUR)
      6 ==> FIT USING ABSO (BIMAXWELLIAN VERSION)
      7 ==> FIT USING COLD BEAM (CUPINT,DCPINT)
      9 ==> FIT USING VERY COLD BEAM (CUPEXP)
    GT 10 ==> TERMINATE EXECUTION
```

We describe the procedure that we use to fit the *M* mode spectrum below to a single proton component, a Voyager 2 spectrum taken at 1977 304 4 59 12 412 (see the discussion in *VI-*

PLS Section 1.5.2.1 about this time in milliseconds actually being 414 originally). The histogram-like curve below gives the measured currents and the smooth/'+' curve shows the fit. In this case we choose to use the full response for the fit, because we want to illustrate how close the *CentOS* and *Solaris* results are in the most advanced case. In examples that follow this one, we will do cases with unit response.



To get the fit above, go to *Voyager_PLS_v_01/software/mjsanl/example_01* and start the process by executing the bash script *mjs_cru*. All the steps in the procedure below are recorded in a session capture file script_*01*. The file *mjs.cru* in its entirety is given below:

```
#!/bin/bash
# script to run example01
# see Example 1:  Fitting an M-mode cruise spectra to a single
# proton using  ICPLT = 4
# in "MJSANL The Positive-Ion Least-Squares Fitting Routine
# for the Voyager PLS Instrument"
#
# copy executable from /software_v_01/mjsanl/code
cp ../code/test .
# copy utility files from common
cp ../common/gnutest_L .
cp ../common/gnutest_M .
cp ../common/reM .
cp ../common/remove .
# define data and sedr files using relative addressing
export ft32="../../../v2/vgr/v2_vgr_1977.233.03_1977.365.23"
export ft33="../../../v2/cruise_ssedr/v2_shortsedr_1977.232.00_1979.186.00"
# execute the executable file
./test
```

The initial copy statement copies over the executable *test* from the *code* directory. The *code* directory is where all the *mjsanl* code and the *Makefile* are stored, and we assume *make* has been run before we start any of the examples. All the examples that we give here use the *code* in the code directory. The script *mjs.cru* also copies some *gnu* plotting routines from the *common* directory, and two bash scripts, *reM*, a bash script to remove *"^M"* from the end of lines in a

file, and *remove* which cleans up the directory after running *mjs.cru*. The *gnu* plotting routines are referenced in the subroutine *writedat.f*. Other files in the *common* directory are explained in the *readme.txt* file in that directory.

When you run the above bash script it prompts for the spacecraft *id*. **Providing this is a crucial step since** this is used to determine the constant *V0*, an overall constant used to determine the conversion from digital units to *fA*, which differs between the two spacecraft. **There is no other check on the identity of the spacecraft in this version of the code.**

The "`export ft32`" statement gives the relative path to get to the mission data (*vgr*) file we use in this fit (see 7.3 of *VI-PLS*). The "`export ft33`" gives the relative path to get to the *ssedr* trajectory file we use in this fit (see 5.4 of *VI-PLS*). The last statement begins the execution of *mjsanl*, via the *test* executable.

Once in execution (*see script_01*), *mjsanl* first asks the user to input *IPLNT*, which is 0 for cruise, and 1, 2, 3, 4 for Jupiter, Saturn, Uranus, and Neptune. The value of *IPLNT* is passed via the common block *mid* to the routine *NXTMOD*, which uses this value to decide how to read in the data from an *ascii* file. There are only two types of data files read by *mjsanl*, *vgr* for cruise (see 7.3 of *VI-PLS*) and *spl* for the encounters (see 7.5 of VI-PLS). Depending on what value of *IPLNT* is entered, we use the read format for either the *vgr* or the *spl* file. In this example, when *ft32* is defined in *mjs.cru*, it referenced a *vgr* file.

MJSANL then asks whether there is trajectory information to be read in. Select option 1, "*Use short cruise file (via PHASE3)*". It then asks several questions about what we are going to do, and we refer to the script *script_01* for the proper answers to these questions. *MJSANL* then asks for a time. It then finds the appropriate data for that time and converts the *DN* values to currents in *fA* by calling *KNTCUR*. It then asks for the fitting method for predicting the currents: here we will choose that parameter *ICPLT* to be *4 ==> FIT USING ABSO (L OR M MODES - LABCUR,LDCUR)*. *XANAL* is then called.

XANAL asks for information about the fit itself: how many ion species to be fit, what parameters of a given species will be fit (in this example, we chose only one component, protons, with a velocity vector, a density, and an isotropic thermal speed). We also specify the channels in the various cups will be used in the fit, which we take to be:

```
FIT OF SPECTRA AT TIME 1977. 304.   4.  59.  12. 412.
  CUP CHANNELS USED IN FIT BY ABSO
   A-CUP:  CHANNELS  32 - 64
   B-CUP:  CHANNELS  32 - 73
   C-CUP:  CHANNELS  32 - 72
   D-CUP:  *** NO FIT PERFORMED ***
```

Then the program determines what the initial values of our parameters will be. In the case of cruise, it obtains these initial values by looking at the answer array (see 1.5.2 and 7.4 of *VI-PLS*) on the mission tape (in this case, (328.950,8.922,24.319) in *km/s* for the vector velocity and 10.764 and 17.180 for the density and thermal speed, in *km/s* and */cm3*).

XANAL then calls *XMJSFT*. This is the routine that carries out the non-linear least squares fit, as described in Section 3.2.4 of *VI-PLS.* That is, from a given starting guess about the plasma parameters, it evaluates the matrices $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ in equation (3.2.4.5) and (3.2.4.6) of that document:

$$\beta_k = -\frac{1}{2}\frac{\partial \chi_o^2}{\partial a_k} \quad \alpha_{jk} = \frac{1}{2}\frac{\partial^2 \chi_o^2}{\partial a_k \partial a_j} \qquad \boldsymbol{\beta} = \boldsymbol{\delta a \alpha} \qquad \boldsymbol{\varepsilon} = \boldsymbol{\alpha}^{-1} \qquad \boldsymbol{\delta a} = \boldsymbol{\varepsilon \beta}$$

XMJSFT does that by calling the subroutine *XFNDCV* which returns information that we use to correct our initial parameter guess to a set that more closely minimizes the value of χ^2. The inversion of the $\boldsymbol{\alpha}$ is done in *XMJSFT* (see the statement '*CALL XINV (ARRAY, NTERMS, DET)*', which returns the inverse of the matrix *ARRAY* in *ARRAY*). The correction of our initial parameter set after a given iteration to an improved one is also done in *XMFSFT* in the statement:

*B(J) = B(J) + BETA(K)*array(j,k)/sqrt(alpha(j,j)*alpha(k,k))*

As you can see in the script, the user can choose to make a plot of the data itself before any fits are attempted. As the iterations of the fit proceed, the user can at any iteration plot the data with the current iteration, including iteration "0", which just gives the currents predicted by the initial guess and the measured currents. Iteration "0" for this example is almost indistinguishable from our final fit given above. This means that our initial guess as provided from the mission file is pretty good. As we proceed through the fit we get information on each iteration. We can also chose to just skip all the iterations and go to the final fit. The scheme decides it is "finished" when the fractional change in chi square is less that *EPS* = 0.10. This value can be changed in the routine *plsbeg.f* (see Example 8 below). The plot shown above at the beginning of this section shows the final fit after three iterations.

We can also run this fit on Solaris using the original version of *mjsanl* (see the file *script_01_solaris*), and below we give the comparison between the CentOS and the Solaris fits:

CentOS: (*script_01*)

MASS	NQ	DEN	XDEN	Vr	XVr	Vt	XVt	Vn	XVn	W	XW
1	1	10.7860	0.0599	330.7221	0.0903	12.4047	0.3206	24.1117	0.2397	15.1686	0.0945

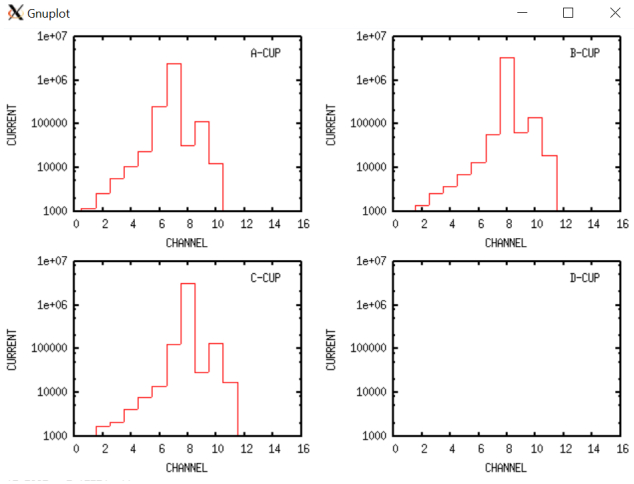
Solaris: (*script_01_solaris*)

MASS	NQ	DEN	XDEN	Vr	XVr	Vt	XVt	Vn	XVn	W	XW
1	1	10.7860	0. 0599	330.7218	0.0902	12.4055	0.3199	24.1142	0.2393	15.1683	0.0945

Note how good this agreement is. Any differences are probably within the numerical accuracy of the two machines. Also note that the *Solaris* fit is reading binary *vgr* tapes and binary *ssedr* tapes on *Solaris*, whereas the *CentOS* fit is reading *ascii* versions of those files on *CentOS* converted from the binary versions on *Solaris*. The fact that we get such good agreement above means both that we have ported the *mjsanl* code correctly, and that the conversion from binary to *ascii* was also done correctly for both kinds of files.

Example 2: Fitting an *L-mode* cruise spectra to a single proton using *ICPLT* = 3

We describe the procedure that we use to fit the *L* mode spectrum below, a Voyager 2 spectrum taken at 1977 304 4 57 50 852, just before the *M* mode above, using unit response. The histogram-like curve below gives the measured currents.



If we assume unit response of the instrument, equation (3.2.3.8) of *VI-PLS* gives a theoretical expression for the current measured in each energy channel for a given cup as a function of the density, the bulk velocity, and the thermal speed along that cup normal:

$$I_k = qA_{eff}\frac{n}{2w_n}\left\{-\frac{1}{\sqrt{\pi}}e^{-\xi^2}\Big|_{(v_k-V_n)/w_n}^{(v_{k+1}-V_n)/w_n} + \left(erf\left((v_{k+1}-V_n)/w_n\right)-erf\left((v_k-V_n)/w_n\right)\right)\right\}$$

This expression is evaluated numerically in the subroutine *xcurnt.f*. We want to fit the data shown above to this theoretical expression. Go to

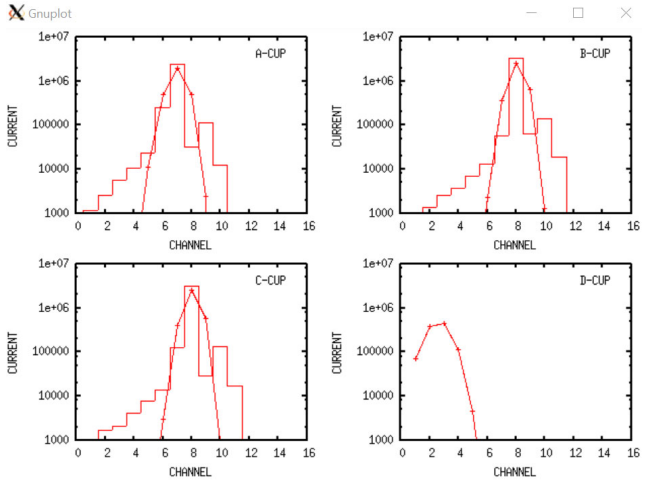
Voyager_PLS_v_01/software/mjsanl/example_02

and start the process by executing the bash script *mjs_cru*. See the file *script_02*. Here we will choose the parameter *ICPLT* to be *3 ==> FIT USING XJC (UNITY RESPONSE)* which is fitting using unit response. *XANAL* is then called.

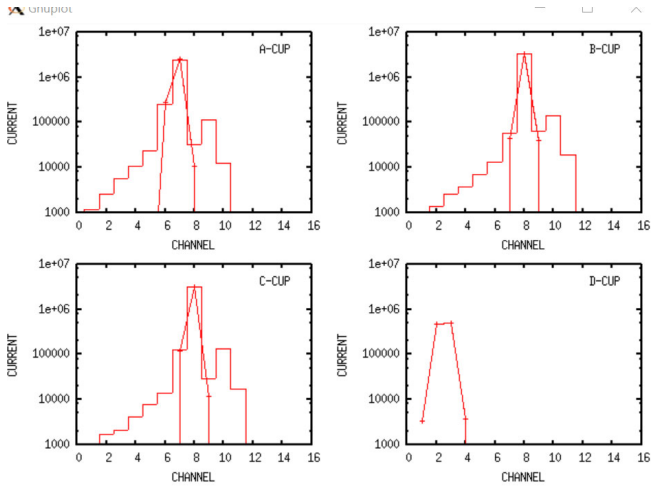
XANAL again asks for information about the fit itself: how many ion species to be fit, what parameters of a given species will be fit (in this example, we chose only one component, protons, with a velocity vector, a density, and an isotropic thermal speed), what channels in the various cups will be used in the fit (we use channels 6-8 in cups A, B, and C, and no channels in the D cup), and determines what the initial values of our parameters will be. In the case of cruise, it obtains these initial values by looking at the answer array (see 7.1 of *VI-PLS*) on the mission tape (in this case, (332.810,11.834,20.550) in *km/s* for the vector velocity and 11.094 and 31.063 for the density and thermal speed, in *km/s* and */cm3*).

XANAL then calls *XMJSFT*. As you can see in the script, the user can choose to make a plot of the data itself before any fits are attempted. As the iterations of the fit proceed, the user can at

any iteration plot the data with the current iteration, including iteration "0", which just gives the currents predicted by the initial guess and the measured currents. Iteration "0" is below.



From the above plot, we can see that the initial guess for the thermal speed is too large, although the density and velocity vector look ok. As we proceed through the fit we get information on each iteration. We can also chose to just skip all the iterations and go to the final fit. The scheme decides it is "finished" when the fractional change in chi square is less that *EPS* = 0.10. The plot below is after seven iterations. The fit has decreased the thermal speed, as we expected.



We have also recently run this fit on *Solaris* (see the file *script_02_solaris*), and below we give the comparison between the *CentOS* and the *Solaris* fits:

CentOS:

MASS	NQ	DEN	XDEN	Vr	XVr	Vt	XVt	Vn	XVn	W	XW
1	1	10.5180	0.0841	327.0838	2.6133	8.6812	2.8115	24.3328	4.6972	14.7680	1.3666

Solaris:

MASS	NQ	DEN	XDEN	Vr	XVr	Vt	XVt	Vn	XVn	W	XW
1	1	10.5265	0.0759	327.7507	1.8282	8.0366	2.0842	25.3386	3.6446	15.0842	1.0217

Example 3: Fitting an *L-mode* cruise spectra to a proton and alpha using *ICPLT* = 3

We consider the same spectrum as in Example 2 above, but we now we fit both a proton and an alpha component. We chose to fit the temperatures and densities separately but use the same velocity for both species. This determination is made in the following section of *script_03* (see the bold face highlights below):

```
1     ** PARAMETERS FOR USE IN ANALYSIS **
 INPUT NUMBER OF COMPONENTS TO FIT TO SPECTRA
2
 DO YOU WISH TO HAVE DENSITY AS A FIT PARAMETER?   (T OR F)
t
 INDICATE WHICH DENSITIES TO FIT (T OR F FOR EACH OF THE 2 SPECIES)
t,t
    Use ECL50 coordinates            ==> 1
    Use S/C coordinates              ==> 2
    Use RTN coordinates              ==> 3
    Use RNP Interstellar Coordinates ==> 4
3
 INPUT LOGICAL VARIABLES (T OR F ) TO SELECT FIT PARAMETERS

 FIT VR, VPHI, VZ FOR COMPONENT  1 ?
t,t,t
 FIT THERMAL SPEED FOR THIS COMPONENT ?
t
 INPUT MASS AND CHARGE STATE ( INTEGERS ) OF COMPONENT  1
1,1
 INPUT LOGICAL VARIABLES (T OR F ) TO SELECT FIT PARAMETERS

 FIT VR, VPHI, VZ FOR COMPONENT  2 ?
f,f,f
 FIT THERMAL SPEED FOR THIS COMPONENT ?
t
 INPUT MASS AND CHARGE STATE ( INTEGERS ) OF COMPONENT  2
4,2
 USE ISOTROPIC MAXWELLIANS IN FIT ?
t
 NCOMP CHARGE  MASS  FIT N FIT VR FIT VPHI FIT VZ FIT W
     1      1     1     T      T        T      T     T
     2      2     4     T      F        F      F     T
 FLAM =      0.100    ALAM =      0.001
 ISOTROPIC MAXWELLIAN     T
 SAME TEMPERATURE         T
 CHANGE FIT OPTIONS? 1=YES, 0=NO
0
```

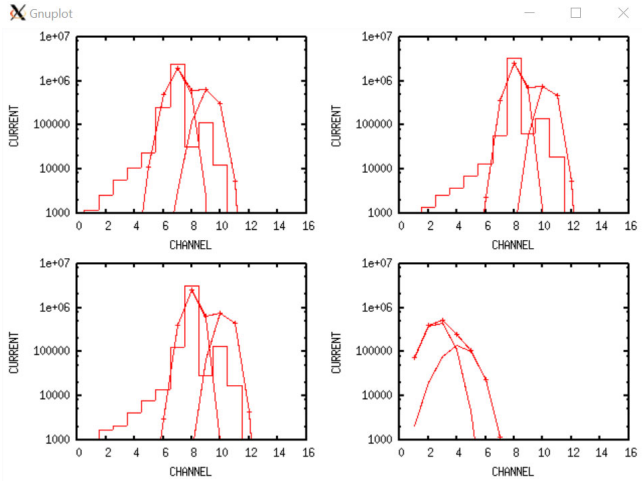
We now fit using channels 6 to 10, to include the alphas in the fit. When we get to the part about choosing our first guess, the script looks like this (see *script_03*):

```
VELOCITY AND 1ST COMPONENT INITIALIZED FROM
 PROTON FIT PARAMETERS FROM MISSION TAPE
 NCOMP        VR       VPHI        VZ      W      DEN
     1   332.810    11.834    20.550    31.063    11.094
     2   332.810    11.834    20.550    31.063*********
 HOW MANY SPECIES DO YOU WISH TO MODIFY FIT PARAMETERS OF?   0=NONE   9=ALL  9
 INPUT LOGICAL VARIABLES (T OR F) FOR PARAMETERS YOU WISH CHANGED
 CHANGE VR,VPHI,VZ,W,DEN FOR COMPONENT 1
f,f,f,f,f
 INPUT LOGICAL VARIABLES (T OR F) FOR PARAMETERS YOU WISH CHANGED
 CHANGE VR,VPHI,VZ,W,DEN FOR COMPONENT 2
```

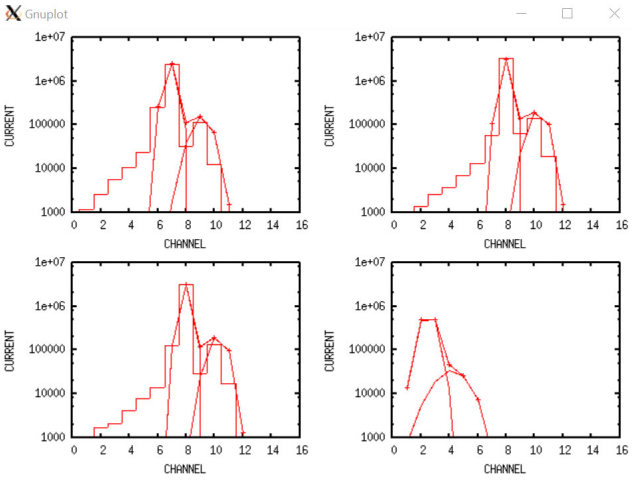
```
f,f,f,f,t
 INPUT NEW VALUE OF DEN FOR COMPONENT 2
2
 BELOW ARE THE CURRENT VALUES OF THE FIT PARAMETERS

 NCOMP         VR        VPHI         VZ       W       DEN
     1    332.810     11.834     20.550     31.063     11.094
     2    332.810     11.834     20.550     31.063      2.000
 DO YOU WISH TO MAKE ANY FURTHER CHANGES?   1=YES   0=NO 0
```

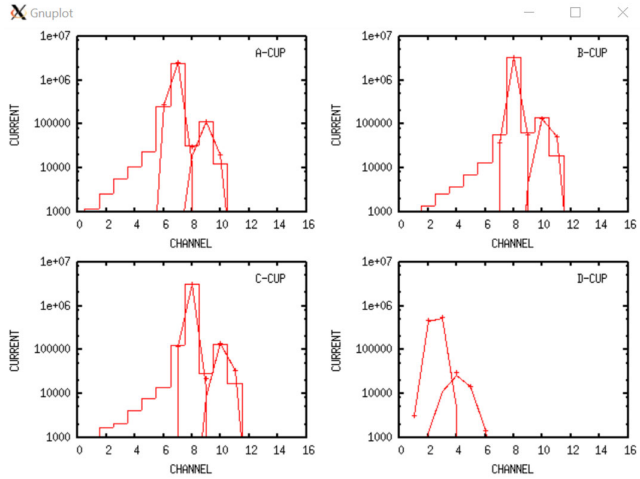
What is happening above is that iniitally the program does not know what to choose for the alpha density, since there was no alpha fit on the *vgr* file for this *L* mode. We have an option to change the initial guess, which we take, as indicated above, and choose a density of 2.0 for the alphas, and then proceed. This will turn out to be an overestimate of the alpha density by an order of magnitude, but we proceed with this estimate for illustration, and choose to plot the zeroth and the first iteration. The zeroth iteration is no interation at all, we just use the initial estimate of the parameters with no attempt to get better values using the least squares routine. The zeroth iterations is plotted below and we see indeed that we have overestimated the alpha density:



The first iteration (after one attempt to correct the initial values using the least squares routines is given below, and we see the code immediately attempting to reduce the alpha density.



The final fit is given below, where we see that we have reduced the alpha density even more.

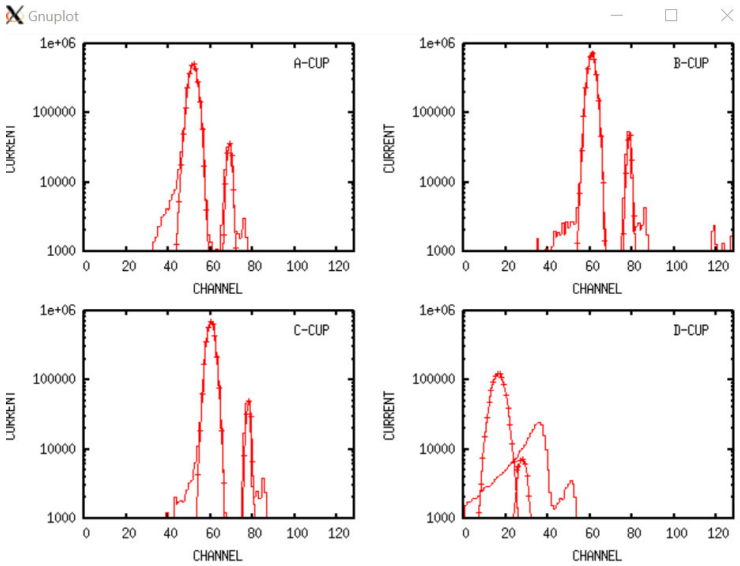


Example 4: Fitting an *M-mode* cruise spectra to a proton and alpha using *ICPLT* = 3/4

We now return to the *M* mode measurement in Example 1 and include an alpha fit and *chose unit response* initially. The results are plotted below, and the script is in *script_04_unit.* In this fit, we have chosen two ranges of channels to fit in the *A*, *B*, and *C* cup, with no channels used in the *D* cup.

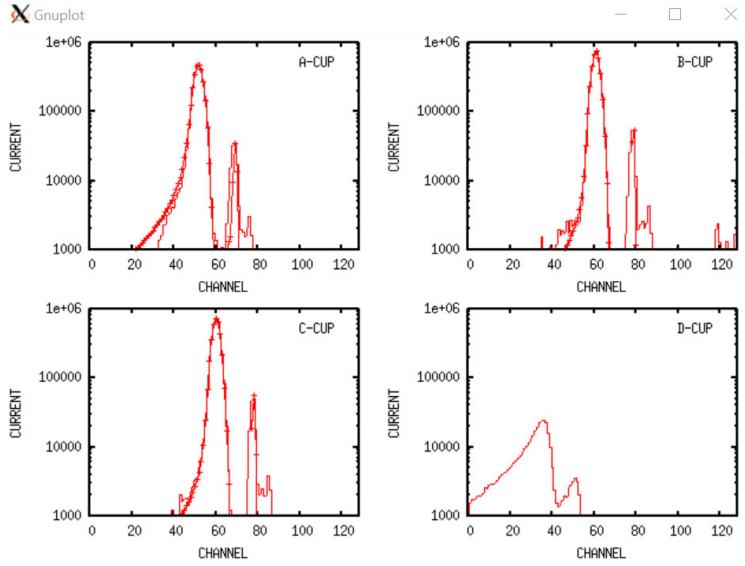
```
CUP CHANNELS USED IN FIT BY ABSO
 A-CUP:  CHANNELS  35 - 58
         CHANNELS  66 - 76
 B-CUP:  CHANNELS  54 - 65
         CHANNELS  76 - 81
 C-CUP:  CHANNELS  55 - 65
         CHANNELS  76 - 81
 D-CUP:  *** NO FIT PERFORMED ***
```

In this case the mission file answer array had a value for the alpha density, so we did not have to change the first guess case. The final fits are shown in the plot below.



Note that the program plots the predicted currents in the *D* cup even though we are using no channels in the *D* cup in the fit, as well as the measured currents. You may be surprised to see measured currents in the *D* cup above. In this period the spacecraft was on the omni antenna and as a result was looking way off the spacecraft-sun line, which is why we see the signals in the *D* cup.

We repeat the above procedure (*see script_04_full*), except we now *ICPLT* = 4 instead of the third option. The final fits are shown below.



Note that in using the full response we reproduce the lower energy wings of the measured spectra. To quote Alan Barnett on page 47 of his thesis, "The broadening and skewness of the peak are both caused by the "feedthrough" effect; the contibution to the signal [to channel *k*] by modulation of particles with $v_z > v_{k+1}$. This effect tends to widen the left side of the peak; there are very few particles which satiasfy this condition on the right side of the peak!". We do not see the lower energy wings when we use unit response, because there is no feed-through in that model.

The *D* cup plots are interesting in that the absence of any fit curves in that cup means that our full response function does not predict any currents in these channels, which is contrary to the fact that we do see measured current there. This implies that our response function is incorrect at these large angles. That is possible, given that the *D* cup currents are a factor of 30 below the C-cup currents, and it may be that Barnett's response function cuts off to zero below that level. It would be worth reviewing the cruise manuever measurements used by Barnett to test his response function, to see if those measurments are consistent with the observatons above. Barnett's thesis is at *Voyager_PLS_v_01/documentation_v_01/general/BarnettThesis.pdf* and the published paper is at *Voyager_PLS_v_01/documentation_v_01/publications/ BarnettOlbert1986.pdf.* One other interesting thing is that in the unit response plot of the *D* cup we see peaks in fits in the *D* cup, but they are not at the energies of the measured currents. They are well below those peaks. That just seems odd, that is we expect a reduction of the current when the ions are coming in at high angles, but not an *increase* in the arrival energy above what

pure geometry would suggest, especially by the amount suggested in the *D* cup plot for unity response. We return to a further consideration of these issues in Example 16 below.

We compare the results with the unit and the full response below:

CentOS Unit Response: *(script_04_unit)*

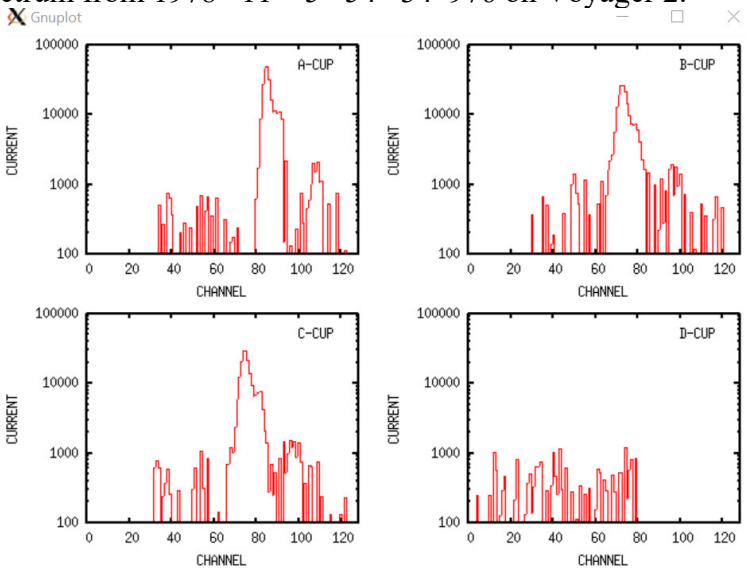
MASS	NQ	DE	XDEN	Vr	XVr	Vt	XVt	Vn	XVn	W	XW
1	1	10.4158	0.0546	329.2622	0.0910	10.0178	0.2965	24.0938	0.2523	15.3640	0.0953
4	2	0.2026	0.0177	329.2622	0.0910	10.0178	0.2965	24.0938	0.2523	7.8272	0.7742

CentOS Full Response: *(script_04_full)*

MASS	NQ	DE	XDEN	Vr	XVr	Vt	XVt	Vn	XVn	W	XW
1	1	10.7873	0.0782	330.6938	0.1210	12.3414	0.434	24.2089	0.3291	15.1699	0.1321
4	2	0.1311	0.0185	330.6938	0.1210	12.3414	0.4354	24.2089	0.3291	3.3059	0.5920

Example 5: Fitting an *M-mode* spectra to two protons and alpha full using *ICPLT* = 4

We now turn to fitting an *M* mode spectrum with clear double peaking in the protons. Below is the spectrum from 1978 11 5 54 34 976 on Voyager 2.



Our steps in fitting this spectrum are given in *script_05*. We choose to fit two protons and one alpha to this spectra, three components in all. We allow the velocities of each of the three components to be independent variables. We do this by selecting the following **in bold** (see script_05)

```
INPUT LOGICAL VARIABLES (T OR F ) TO SELECT FIT PARAMETERS

 FIT VR, VPHI, VZ FOR COMPONENT  1 ?
t,t,t
 FIT THERMAL SPEED FOR THIS COMPONENT ?
t
 INPUT MASS AND CHARGE STATE ( INTEGERS ) OF COMPONENT  1
1,1
 INPUT LOGICAL VARIABLES (T OR F ) TO SELECT FIT PARAMETERS

 FIT VR, VPHI, VZ FOR COMPONENT  2 ?
t,t,t
 FIT THERMAL SPEED FOR THIS COMPONENT ?
t
```

```
 INPUT MASS AND CHARGE STATE ( INTEGERS ) OF COMPONENT  2
1,1
 INPUT LOGICAL VARIABLES (T OR F ) TO SELECT FIT PARAMETERS

 FIT VR, VPHI, VZ FOR COMPONENT  3 ?
t,t,t
 FIT THERMAL SPEED FOR THIS COMPONENT ?
t
 INPUT MASS AND CHARGE STATE ( INTEGERS ) OF COMPONENT  3
4,2
 USE ISOTROPIC MAXWELLIANS IN FIT ?
T
```

By examining the spectrum in detail, we manually pick the following channels for each cup, two ranges per cup:

```
    CUP CHANNELS USED IN FIT BY ABSO
     A-CUP:  CHANNELS  82 - 92
             CHANNELS 105 -111
     B-CUP:  CHANNELS  69 - 81
             CHANNELS  96 -100
     C-CUP:  CHANNELS  69 - 83
             CHANNELS  95 -100
     D-CUP:  *** NO FIT PERFORMED ***
```

When we come to the first guess, the program suggests the following parameters:

```
VELOCITY AND 1ST COMPONENT INITIALIZED FROM
 PROTON FIT PARAMETERS FROM MISSION TAPE
 NCOMP        VR       VPHI        VZ      W      DEN
     1   504.430     7.455     7.766    35.302     0.445
     2   504.430     7.455     7.766    35.302     0.006
     3   504.430     7.455     7.766    35.302     0.000
 HOW MANY SPECIES DO YOU WISH TO MODIFY FIT PARAMETERS OF?   0=NONE   9=ALL  9
```

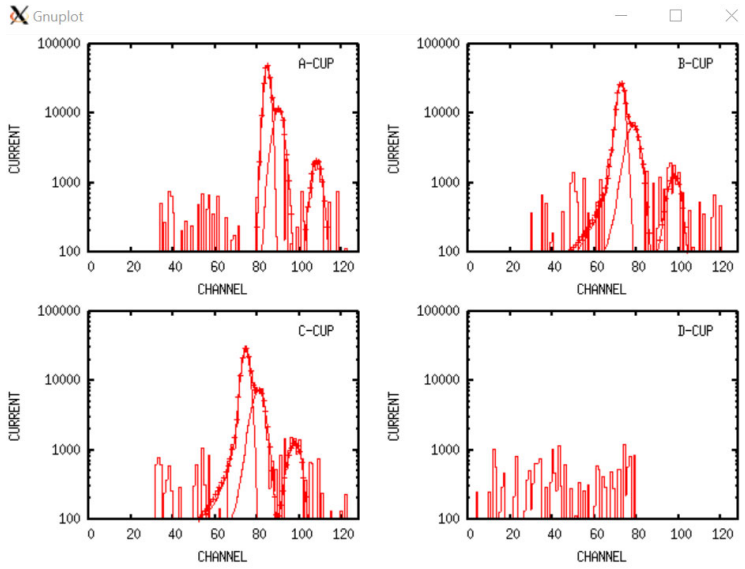
By choosing “9”, we change all these first guesses, to the following set:

```
BELOW ARE THE CURRENT VALUES OF THE FIT PARAMETERS

 NCOMP        VR       VPHI        VZ      W      DEN
     1   490.000     0.000     8.000    15.000     0.400
     2   540.000    -9.000     7.000    30.000     0.100
     3   540.000     4.000     9.000    30.000     0.010
```

We have chosen the first of our protons to have a lower velocity and a higher density than the second, to match what we see in the measured data. We have added an alpha density. We then let the fit proceed, and after three iterations we get the fit below.

We run the exact same fit on *Carrington*, a *Solaris* machine, with the results given below (see *script_05_solaris*).



CentOS fit: (*script_05*)

MASS	NQ	DEN	XDEN	Vr	XVr	Vt	XVt	Vn	XVn	W	XW
1	1	0.3403	0.0099	494.2411	0.2602	4.4223	0.5130	9.2654	0.6912	17.7764	0.3033
1	1	0.1268	0.0130	542.8430	2.0221	-8.7689	3.5727	7.6971	4.6683	33.3942	3.9481
4	2	0.0112	0.0027	535.5438	6.9657	-9.5738	20.8094	39.1037	30.4354	32.0117	10.2953

Solaris fit: (*script_05_solaris*)

MASS	NQ	DEN	XDEN	Vr	XVr	Vt	XVt	Vn	XVn	W	XW
1	1	0.3404	0.0098	494.2432	0.2593	4.4114	0.5034	9.2473	0.6994	17.7790	0.3020
1	1	0.1267	0.0130	542.8585	2.0175	-8.9170	3.5652	7.2726	4.6536	33.3619	3.9406
4	2	0.0112	0.0027	535.5873	6.8982	-9.0043	20.5208	39.7098	30.1791	32.0565	10.1905

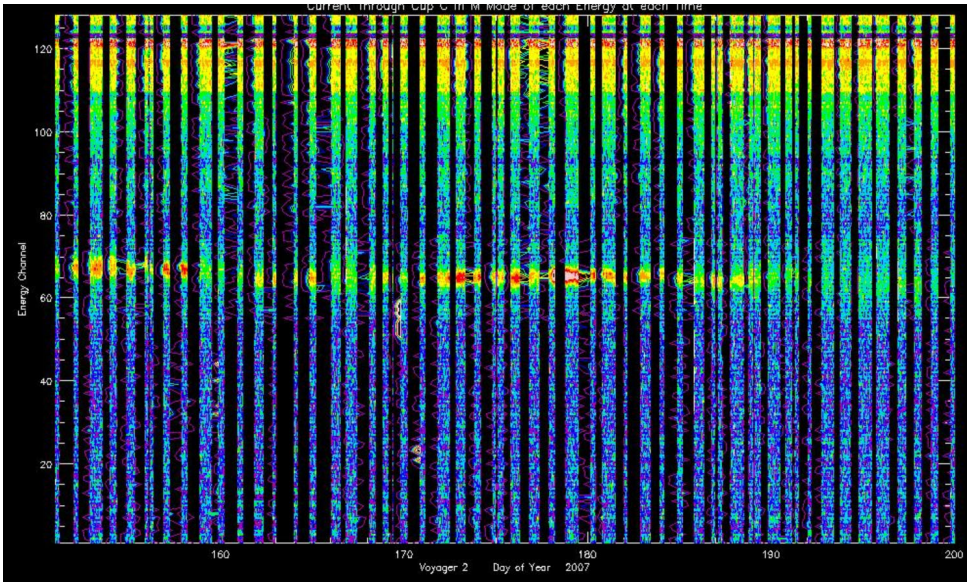
Note that since we have chosen to fit the full vector velocity for each component, as well as individual densities and temperatures for each, we have a total of 3x5=15 dependent variables to the fit.

Also note again the close agreement between the *Solaris* and the *CentOS* fit, even in the complicated situation of fitting three components with 15 independent variables. Once again this argues for the correct porting of the code and the data sets (the *Solaris* data sets are binary, the *CentOS* data sets are ascii files made from those original binary files).

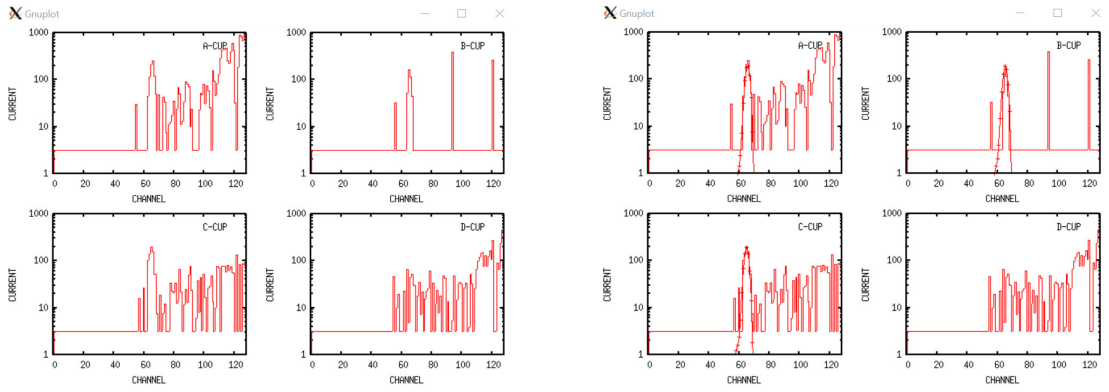
There are better examples of double peaked proton distributions in the Voyager data, see *Amador_2_peaks_senior_thesis.pdf* in *documentation_v_01/general*.

Example 6: Fitting an *M-mode* spectra to a proton late in mission using *ICPLT* = 4

The examples above have been near the beginning of the mission. We now turn to an *M* mode taken not too long before crossing the termination shock. Voyager 2 crossed the termination shock of the solar wind at least five times on DOY 242 to 244, 2007. We choose an *M* mode at 2007 179 12 2 9 196, well before the crossing. Below, we show a color spectragram of the data in the *M* mode C cup which includes this spectrum (see Section 7.4 of *VI-PLS* for information on where to find the color spectrograms).

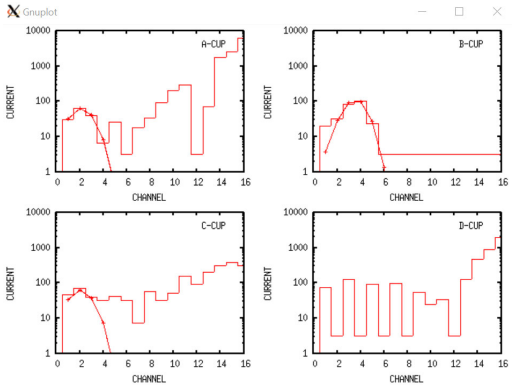


This is the initial data with no fit and the final fit after 4 iterations (see *script_06*).



Example 7: Fitting an *L-mode* heliosheath spectra to a proton using *ICPLT* = 4

We now move to one of John Richardson's heliosheath spectra fits, an *L* mode at 2007 267 19 16 26 136. The spectrum and the fit are shown below:



CentOS fit: (script_07)

MASS	NQ	DEN	XDEN	Vr	XVr	Vt	XVt	Vn	XVn	W	XW
1	1	0.0019	0.0001	122.6006	2.4853	43.8798	7.7169	-74.8281	7.1593	36.1469	2.9503

Solaris fit: (script_07_solaris)

MASS	NQ	DEN	XDEN	Vr	XVr	Vt	XVt	Vn	XVn	W	XW
1	1	0.0019	0.0001	122.5981	2.4858	43.9101	7.7182	-74.8093	7.1560	36.1539	2.9487

Note again the excellent agreement between the *CentOS* fit and the *Solaris* fit. All of John Richardson's heliosheath fits can be found in

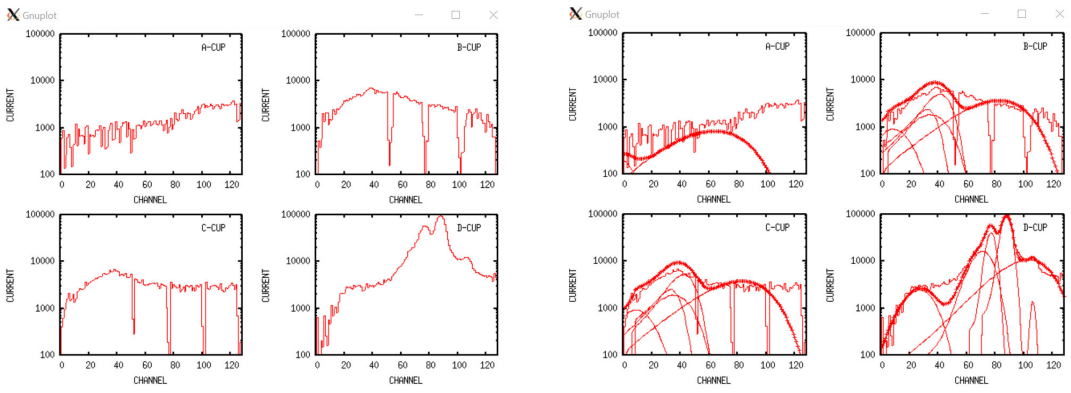
Voyager_PLS_v_01/v2/daily_hourly_fine/fine/after_shock

Example 8: Fitting an *M-mode* from Voyager 1 at Jupiter using *ICPLT* = 5

We now move to magnetospheric fits, an M mode at 1979 63 23 42 23 6. In *Voyager_PLS_v_01/software/mjsanl/example_08* we have a new bash script *mjs.jup*. Here we switch from using the mission (*vgr*) files we use for cruise to the *spl* files we use for encounters (see Section 7.3 of *VI-PLS*). That is, we now define

```
export ft32="../../../v1/encounter_spl/spl.rmaster.v1.jup.txt"
export ft33="../../../v1/encounter_ssedr/ssedr.v1.jup.txt"
```

The plot on the left below is the data, and the plot on the right the data and final fit. See *script_08*.



We give the results of the *CentOS* fit and also the *Solaris* fit from *v1jupiter.mag.M.testcase*, below. There are significant differences that I do not understand. To get the *CentOS* fit to agree as well as it does, I had to reduce the value of *EPS* in the data statement in *plsbeg.f* to 0.03, instead of the usual 0.1 (see line 327). I did this by editing that data statement in *plsbeg.f* located in *Voyager_PLS_v_01/software_v_01/mjsanl/code* , saving, and doing a make. I have to remember to *undo* this change in *EPS* after I am finished doing the fit below.

CentOS fit: (see *script_08*)

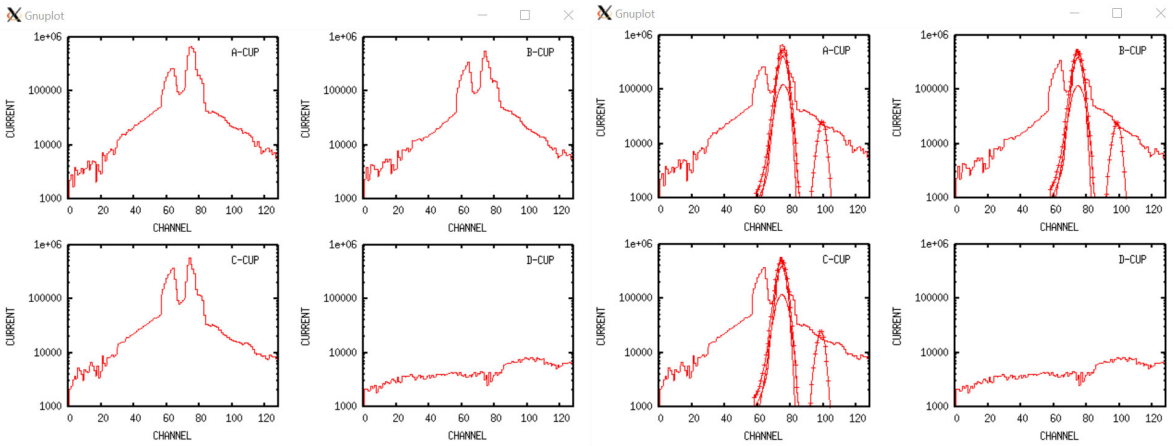
MASS	NQ	DEN	XDEN	VR	XVR	VPHI	XVPHI	VZ	XVZ	W	XW
1	1	0.6759	0.1039	-0.2019	0.0244	141.1373	0.1337	12.0649	0.9809	55.1679	8.1570
16	2	1.5221	0.0996	-0.2019	0.0244	141.1373	0.1337	12.0649	0.9809	25.6009	1.3529
32	3	1.0433	0.0447	-0.2019	0.0244	141.1373	0.1337	12.0649	0.9809	10.0963	0.3010
16	1	7.5993	0.1386	-0.2019	0.0244	141.1373	0.1337	12.0649	0.9809	10.8346	0.1669
32	1	0.0699	0.0741	-0.2019	0.0244	141.1373	0.1337	12.0649	0.9809	4.9141	4.0070
16	1	5.6031	0.2125	-0.2019	0.0244	141.1373	0.1337	12.0649	0.9809	79.1309	2.0013

Solaris fit: (see *v1jupiter.mag.M.testcase*)

MASS	NQ	DEN	XDEN	VR	XVR	VPHI	XVPHI	VZ	XVZ	W	XW
1	1	0.6561	0.1076	-2.2076	0.2140	141.3109	0.1623	13.0458	0.8784	50.8388	2.0239
16	2	0.9004	0.0389	-2.2076	0.2140	141.3109	0.1623	13.0458	0.8784	12.7097	0.5060
32	3	1.1099	0.0212	-2.2076	0.2140	141.3109	0.1623	13.0458	0.8784	8.9871	0.3578
16	1	8.5227	0.1161	-2.2076	0.2140	141.3109	0.1623	13.0458	0.8784	12.7097	0.5060
32	1	0.1348	0.0702	-2.2076	0.2140	141.3109	0.1623	13.0458	0.8784	8.9871	0.3578
16	1	5.3320	0.2181	-2.2076	0.2140	141.3109	0.1623	13.0458	0.8784	77.3918	2.0836

Example 9: Fitting an *M-mode* from Voyager 2 at Jupiter using *ICPLT* = 5

We now move to magnetospheric fits from Voyager 2, an *M* mode at 1979 190 20 57 46 43. We change the data files read in the appropriate Voyager 2 files by making changes in *mjs.jup*. The plot on the left below is the data, and the plot on the right the data and final fit. See script_09.



We give the results of the CentOS fit and also the Solaris fit from John Richardson below. There are significant differences that I do not understand.

CentOS fit: (script_09)

MASS	NQ	DEN	XDEN	VR	XVR	VPHI	XVPHI	VZ	XVZ	W	XW
16	1	10.4372	8.7699	-3.9282	1.1419	125.6470	0.3169	-1.3694	1.1677	9.3578	0.8542
32	2	12.3571	4.4332	-3.9282	1.1419	125.6470	0.3169	-1.3694	1.1677	6.6169	0.6040
39	1	1.4501	1.2658	-3.9282	1.1419	125.6470	0.3169	-1.3694	1.1677	5.9938	0.5471

Solaris fit: (see *v2jupiter.mag.M.testcase*)

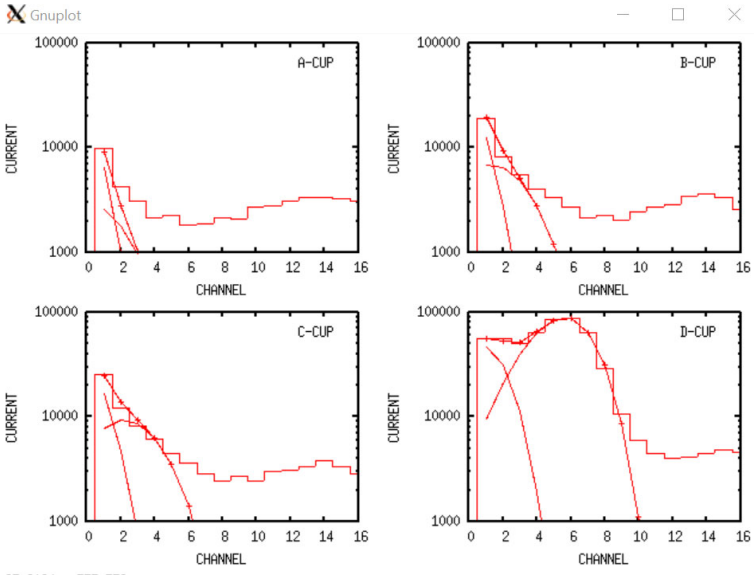
MASS	NQ	DEN	XDEN	VR	XVR	VPHI	XVPHI	VZ	XVZ	W	XW
16	1	16.7024	9.3625	-3.9360	1.1375	125.6788	0.3182	-1.6590	1.1455	8.4111	0.933
32	2	9.1092	4.7765	-3.9360	1.1375	125.6788	0.3182	-1.6590	1.1455	5.9475	0.659
39	1	1.4249	1.2929	-3.9360	1.1375	125.6788	0.3182	-1.6590	1.1455	5.3874	0.597

Example 10: Fitting an *L-mode* from Voyager 1 at Saturn using *ICPLT* = 5

In *Voyager_PLS_v_01/software/mjsanl/example_10* we have a new bash script *mjs.sat*. In that script, we now define

```
export ft32="../../../v1/encounter_spl/spl.rmaster.v1.sat.txt"
export ft33="../../../v1/encounter_ssedr/ssedr.v1.sat.txt"
```

The script script_10 gives the fit and below is the final plot.



Here are the final fit results from *CentOS*.

MASS	NQ	DEN	XDEN	VR	XVR	VPHI	XVPHI	VZ	XVZ	WPAR	XWPAR	WPER	XWPER
1	1	4.9781	13.6928	10.1282	1.9257	47.6988	3.3354	-9.8288	4.3880	50.2935	139.1514	50.7000	1.9552
16	1	16.1285	16.7191	10.1282	1.9257	47.6988	3.3354	-9.8288	4.3880	23.2269	23.7297	25.8127	0.9795

Here are the the fit results from *v1saturn.mag.bimax.L.testcase*, a *Solaris* run on October 2, 1992 from John Richardson.

MASS	NQ	DEN	XDEN	VR	XVR	VPHI	XVPHI	VZ	XVZ	WPAR	XWPAR	WPER	XWPER
1	1	5.3097	0.3758	10.8794	2.3352	48.2434	5.3631	-12.0439	5.0419	64.8720	11.7703	46.9323	4.2208
16	1	15.2825	2.0166	10.8794	2.3352	48.2434	5.3631	-12.0439	5.0419	12.4342	11.8688	28.5273	2.5326

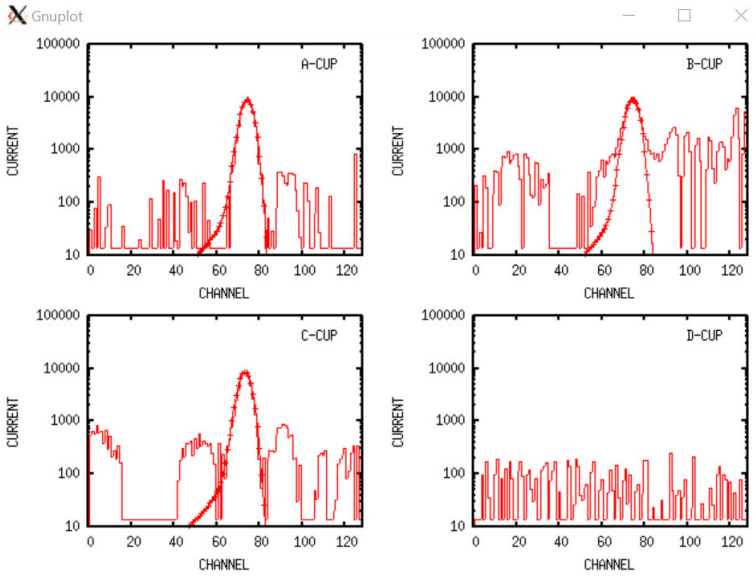
The fits are reasonably close, although there is a large value of *XDEN* in the *CentOS* fits, for reasons I do not understand.

Example 11: Fitting an *M-mode* from Voyager 2 at Saturn using *ICPLT* = 5

In */nfs/faraday/data/jwb/Voyager_PLS_v_01/software/mjsanl/example_11* we have the same bash script *mjs.sat*. except in that script, we now define

```
export ft32="../../../v2/encounter_spl/spl.rmaster.v2.sat.txt"
export ft33="../../../v2/encounter_ssedr/ssedr.v2.sat.txt"
```

We fit an M mode *in the solar wind* at time 1981 237 0 24 10 589. We do the fit in the solar wind because we do not have a good *M* mode in the magnetosphere. We do this in cylindrical coordinates, so that the good agreement with the *Solaris* fits means that all the rotations etc. to get to those coordinate are correct in *CentOS*. The final fit and the currents are given below.



CentOS fit: (see *script_11*)

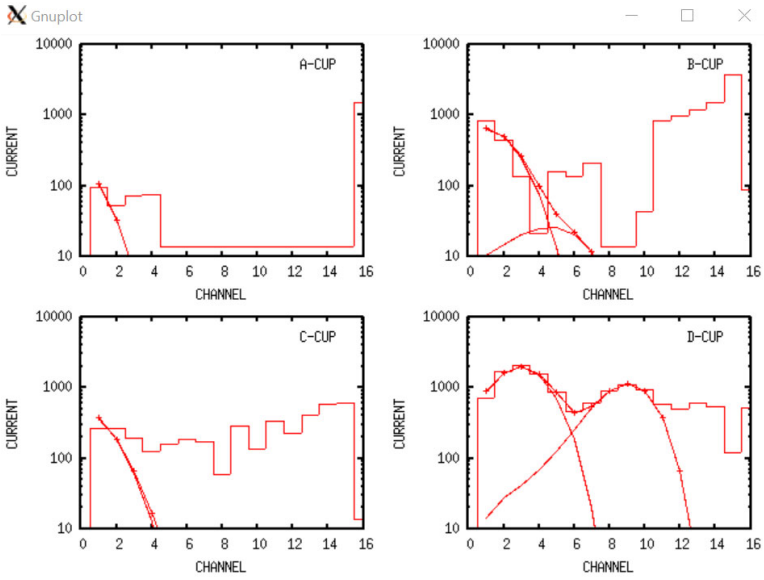
MASS	NQ	DEN	XDEN	VR	XVR	VPHI	XVPHI	VZ	XVZ	W	XW
1	1	0.1527	0.0034	-404.1609	0.9097	135.2944	1.9844	-38.4049	2.0299	28.6564	0.7630

Solaris fit: (see *v2saturn.mag.M.testcase*)

MASS	NQ	DEN	XDEN	VR	XVR	VPHI	XVPHI	VZ	XVZ	W	XW
1	1	0.1556	0.0035	-404.1593	0.9123	135.3142	1.9811	-38.3596	2.0661	28.6653	0.7613

Example 12: Fitting an *L-mode* from Voyager 2 at Saturn using *ICPLT* = 4

We fit an L mode in the magnetosphere at time 1981 237 12 9 11 986. The script script_12 gives the fit and below is the final plot.



CentOS fit: (see *script_12)*

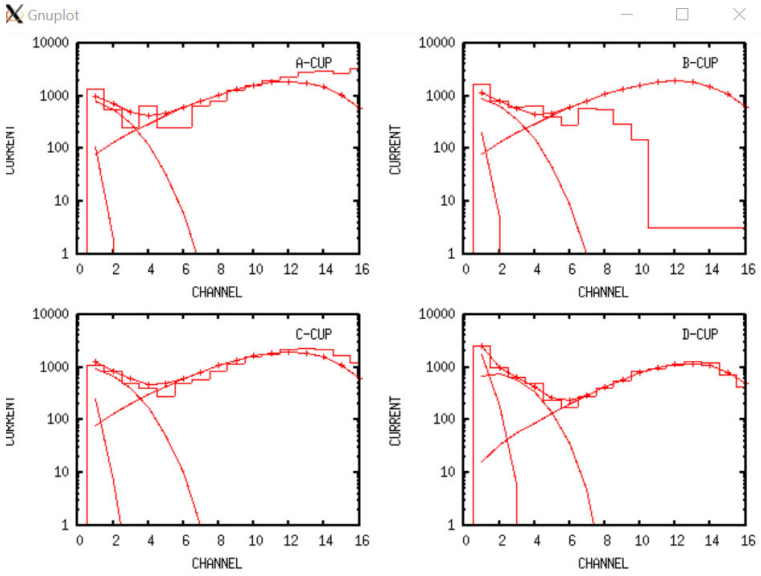
MASS	NQ	DEN	XDEN	VR	XVR	VPHI	XVPHI	VZ	XVZ	W	XW
1	1	0.1840	0.0368	-9.7758	6.3399	96.5045	4.7159	58.0640	18.4063	67.6883	2.9752
14	1	0.1199	0.0511	-9.7758	6.3399	96.5045	4.7159	58.0640	18.4063	31.8104	5.9487

Solaris fit: (see *v2saturn.mag.L.testcase*)

MASS	NQ	DEN	XDEN	VR	XVR	VPHI	XVPHI	VZ	XVZ	W	XW
1	1	0.1819	0.0267	-9.7315	6.3866	96.2230	4.2880	56.5567	17.0357	67.9139	2.8505
14	1	0.1160	0.0325	-9.7315	6.3866	96.2230	4.2880	56.5567	17.0357	32.3309	5.9545

Example 13: Fitting an *L-mode* from Voyager 2 at Uranus using *ICPLT* = 4

We now move to magnetospheric fits at Uranus using an L mode at 1979 63 23 42 23 6. In *Voyager_PLS_v_01/software/mjsanl/example_13* we have a new bash script *mjs.ura.* The script script_13 gives the fit and below is the final plot.



CentOS fit: (see *script_13*)

MASS	NQ	DEN	XDEN	VR	XVR	VPHI	XVPHI	VZ	XVZ	W	XW
1	1	0.5322	0.0810	0.0000	0.0000	12.3240	0.0000	0.0000	0.0000	35.5004	4.8889
1	1	0.2084	0.0398	0.0000	0.0000	12.3240	0.0000	0.0000	0.0000	86.1362	6.9342
1	1	0.1419	0.0046	0.0000	0.0000	12.3240	0.0000	0.0000	0.0000	539.1489	9.7828

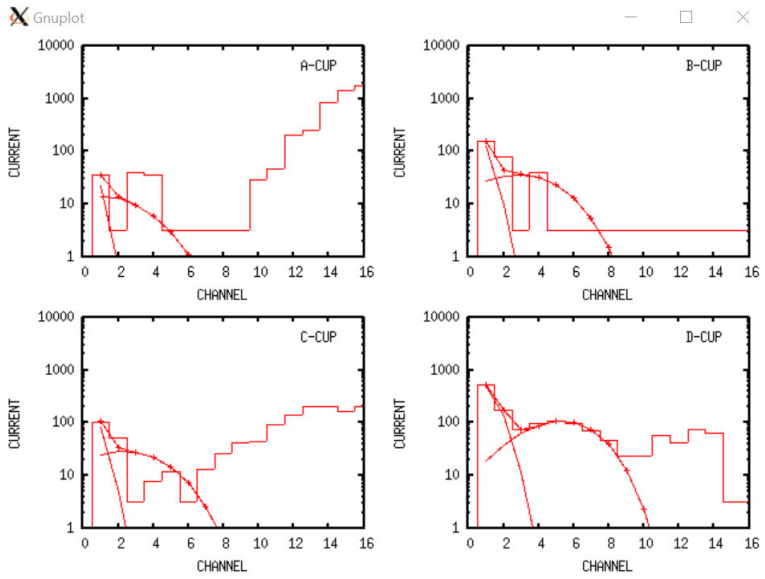
Solaris fit: (see *v2uranus.mag.L.testcase*)

MASS	NQ	DEN	XDEN	VR	XVR	VPHI	XVPHI	VZ	XVZ	W	XW
1	1	0.5330	0.0809	0.0000	0.0000	12.3239	0.0000	0.0000	0.0000	35.5088	4.8909
1	1	0.2087	0.0399	0.0000	0.0000	12.3239	0.0000	0.0000	0.0000	86.1450	6.9429
1	1	0.1421	0.0047	0.0000	0.0000	12.3239	0.0000	0.0000	0.0000	539.1536	9.7851

The agreement is excellent.

Example 14: Fitting an *L-mode* from Voyager 2 at Neptune using *ICPLT* = 4

We now move to magnetospheric fits at Neptune using an *L* mode at 1979 63 23 42 23 6. In *Voyager_PLS_v_01/software/mjsanl/example_14* we have a new bash script *mjs.nep.* The script *script_14* gives the fit and below is the final plot.



CentOS fit: (see *script_14*)

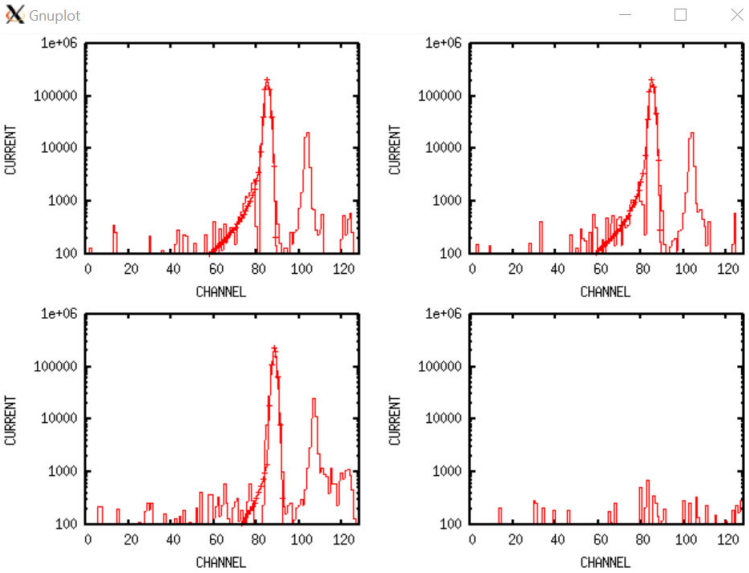
MASS	NQ	DEN	XDEN	VR	XVR	VPHI	XVPHI	VZ	XVZ	W	XW
1	1	0.0689	0.0073	0.0000	0.0000	22.5494	2.7496	0.0000	0.0000	39.3401	0.8552
14	1	0.0417	0.0056	0.0000	0.0000	22.5494	2.7496	0.0000	0.0000	35.3759	1.9787

Solaris fit: (see *neptune.mag.testcase*)

MASS	NQ	DEN	XDEN	VR	XVR	VPHI	XVPHI	VZ	XVZ	W	XW
1	1	0.0700	0.0076	0.0000	0.0000	22.0869	2.7971	0.0000	0.0000	39.4669	0.8791
14	1	0.0424	0.0058	0.0000	0.0000	22.0869	2.7971	0.0000	0.0000	35.5725	2.0162

The agreement is excellent.

Example 15: Fitting an *M-mode* from Voyager 1 Cruise post-Jupiter using *ICPLT* = 4



We now return to cruise and fit a Voyager 1 spectrum after the Jupiter encounter in the *M* mode at 1979 93 12 0 7 572. In *Voyager_PLS_v_01/software/mjsanl/example_15* we have a bash script *mjs.cru.* The script *script_15* gives the fit and above is the final plot. The initial variables suggested for the fit from the Mission file are (VR,VT,VN,W,Den) :

528.9 -10.3 -3.6 18.9 1.047

The final *CentOS* fit yields (VR,VT,VN,W,Den):

532.5 -11.7 -1.4 13.6 1.158

Example 16: Fitting an *M-mode* during Barnett's Cruise Maneuver using *ICPLT* = 4

Let us return to the discussion in Example 04 above. From Barnett's thesis, page 96, I have the following table listing the times of the cruise maneuver he used to test his response function. The 14th of September in a non-leap year is day 157.

Table 3.1
Plasma Parameters From Cruise Maneuver on 14 September 1978

Time	Number	V_x	V_y	V_z	V_R	V_T	V_N	N_o	w
2345	1	305.	-8.	225.	378.±1.0	18.2±1.2	-23.8±1.4	.284±.008	19.8±.5
2313	2	221.	-8.	297.	370.±0.4	11.8±0.8	-19.1±0.9	.300±.007	16.5±.3
2315	3	118.	-5.	353.	372.±0.3	9.1±0.8	-12.9±1.0	.294±.009	17.2±.2
2316	4	11.4	-6.4	370.	370.±0.2	10.0±0.8	-16.2±0.8	.293±.009	16.4±.2
2316	4a	10.9	-6.5	368.	368.±0.3	9.5±1.1	-16.0±1.1	.276±.011	17.7±.3
2243	5	17.8	-6.6	370.	371.±0.2	7.0±0.9	-14.1±0.9	.237±.008	17.0±.3
2243	5a	17.5	-7.0	369.	369.±0.3	6.5±1.1	-14.3±1.2	.258±.010	18.3±.3
2211	6	-81.5	-7.8	363.	372.±0.4	8.1±1.0	-15.3±1.0	.304±.012	15.9±.3
2212	7	-184.	-4.7	323.	371.±0.7	10.8±0.9	-13.8±1.0	.291±.009	16.4±.3
2248	8	-276.	-2.1	248.	371.±1.0	13.8±0.6	-13.9±1.0	.258±.006	14.3±.4
2216	9	-341.	-1.6	161.	377.±1.3	9.0±0.7	- 9.8±1.3	.278±.007	18.1±.6

I had wanted to look at the spectra below (Figure 3.1 of the thesis), which is spectrum #1 in the Table above.

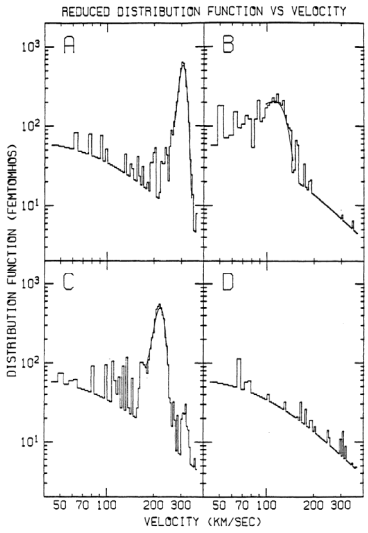
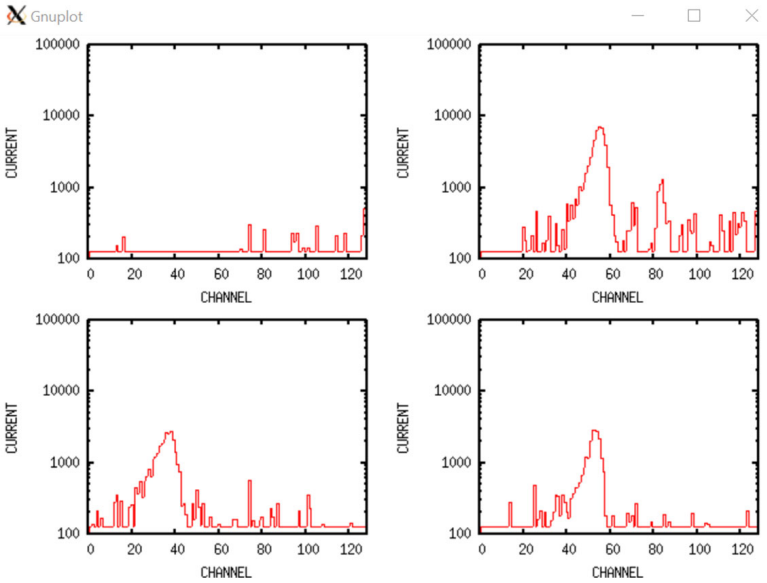


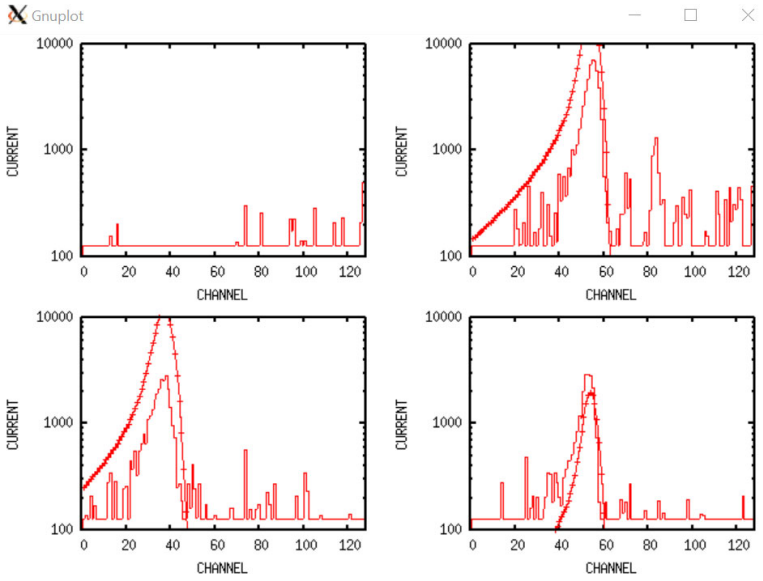
Figure 3.1

However I have been unable to find this spectrum in the Mission (*vgr*) files for this day, as the time occurs in a data gap! I have no idea why this is. Perhaps Alan was using an in-house *edr*. There is one spectrum on the following day, 1978 258 3 16 59 836, see below, that I have fit, with odd results.

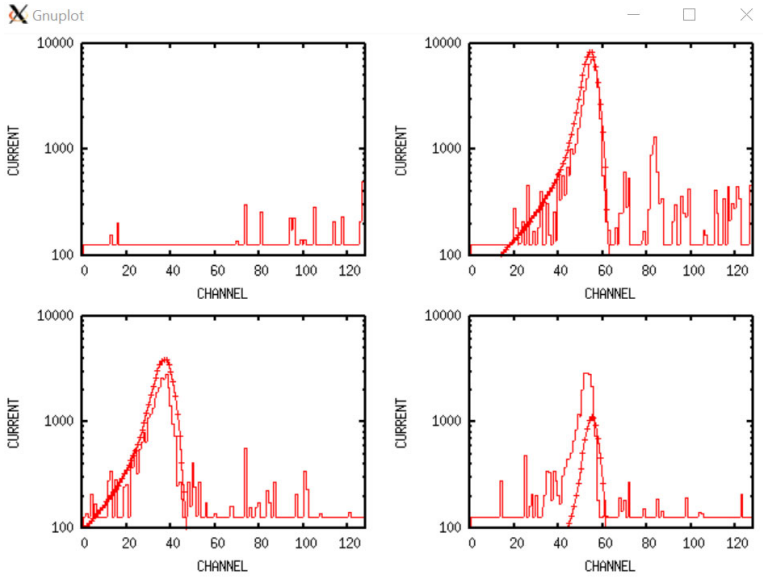
The script *script_16* in *software_v_01/mjsanl/example_16* is produces the plot below. From the first plot below, we can see that the solar wind has the following components of speed into the sensors: *Vc* is -186.3, *Vb* is -268.5, and *Vd* is -257.7, because the peaks are at channels 38, 55, and 53 (see Section 3.1.1.2 of *VI-PLS*). Solving for *Vx*,*Vy*, and *Vz* (see Section 2.2 of *VI-PLS*) gives -326.3, -28.1 and 188. We then try to fit the spectrum below, chosing 2 when we have a choice of coordinates (*Use S/C coordinates ==> 2*).



Below is the zeroth iteraton for the fit, this is just a plot of the data and the initial guess of parameters, with no attempt to improve them. We are spot on with our velocity guess, as would be expected, but the predicted currents are vastly in excess in *B* and *C* as compared to *D*.



Below is the first iteration, which has knocked down the density so that the currents in the *A* and *B* cup are close to the measured ones, but the predicted currents in the *D* cup are far below the measured ones. I don't know why the response function for the *D* sensor should be so far off.



I would like to find another cruise maneuver (*CRM*) to do fits similar to Barnett's fits. The only way I know to locate one is by looking at color spectrograms.

List of Routines

There is also information about these routines contained in the document *16.2softwaredocs _Software library documentation_* in *documentation_v_01/general*.

Routines Associated with VMV's PMODEL

Voyager_PLS_v_01/documentation_v_01/generalPMODEL.pdf provides documentation for V. M. Vasyliunas's *PMODEL* program.

cpresp.f

A function in VMV's *PMODEL* code that computes the response function. Called by *pmodel.f* and *cuprsp.f*.

cuprsp.f

This may belong in Barnett's section. Routine to calculate the normalized response of the main and side sensors of the Voyager instrument. Normalized response is defined as the cup transparency times sensitive area for the direction defined by V divided by the cup transparency times sensitive area at normal incidence. Called by *phase2.f* and *phase3.f*. Mentioned in *xparm.f*.

pmodel.f

Computes currents in the Voyager plasma experiment for a given species, given the species bulk velocity V, the field direction B, and any gyro-tropic distribution function. Called by *xsimul.f*.

xmax.f

Bi-maxwellian distribution for low mach number flow (M<5). Used in *xsimul.f* in the call to *pmodel.f*.

Routines Associated with Barnett's Response Function

Alan Barnett's thesis on the instrument response function is available at *Voyager_PLS_v_01/documentation_v_01/generalBarnettThesis.pdf*. The paper based on this thesis is at *Voyager_PLS_v_01/documentation_v_01/publications/BarnettOlbert1986.pdf*.

abccur.f

SUBROUTINE ABCCUR(DN,W,U1,A,Z,NCHN,ncup,LDIST,DISTF):
This subroutine computes the main sensor response of the Voyager PLS experiment operating in the M-mode to a plasma whose distribution function can be modeled as a convected Maxwellian. It uses an algorithm in which the integrations over the components of velocity perpendicular to the cup normal are performed analytically, while the integration over the component of velocity along the cup normal is done numerically. The algorithm is described in detail in Alan Barnett's doctoral thesis. The integration over *vz* is divided into two regions which are treated separately. For channel *k*, the integration over velocities up to the threshold velocity for channel *k+2* is evaluated by a different loop from the remainder of the integral. The rest of the range of *vz* is called the tail. The contribution for each value of *vz* is computed using a Taylor expansion of the current as a function of the response function parameters. The tail is further divided into two halves, the near tail and the far tail. The currents in the tail and the derivatives of those currents with respect to the response function parameters are computed by subroutine *SETCUR*, a subroutine contained in *abccur.f*. Called by *xsimul.f*.

armain.f

FUNCTION ARMAIN(X,Y):
This function returns the sensitive area of the Voyager main sensors (not normalized) as a function of *X=S*Vx/Vz* and *Y=S*Vy/Vz* in cup coordinates by linear interpolation in a two-dimensional look-up table. Here *S* is the shift function (see thesis of Alan Barnett). Called by *cpresp.f.*

arside.f

FUNCTION ARSIDE(DAS,DSC):
This function computes the area in square centimeters of the Voyager side sensor.
DAS=relative displacement of aperture and shield circle centers, cm.
DSC=relative displacement of shield and collector circle centers, cm.
Called by *cpresp.f.*

conrd.f

This subroutine fills the common blocks which are used in Alan Barnett's simulation routines. On the SUN system the routine originally read a binary file *conrd.data*, which can be found now in */nfs/faraday/data/plas7/d3/vgr/const/conrd.data*. That binary file has been converted to an *ascii* file, *conrdascii*. That file is stored in *Voyager_PLS_v_01/software_v_01/mjsanl/common*, and *conrd.f* expects that to be the location (see code in *conrd.f*: *open(39, file='../common/conrdascii'. form='formatted')*). Called by *mjsanl.f* and *xsimul.f.*

cupint.f

This subroutine simulates the response of the Voyager experiment main sensor to a cold ($M > 5$) convected Maxwellian plasma. The common blocks are filled by the subroutine *conrd.f*, which must be called before the first call to *cupint.f*. Called in *xsimul.f.*

dcur.f

Routine to calculate side sensor response. Called in *xsimul.f.*

displd.f

Computes the relative displacement of aperture, shield, and collector projected along a beam in Voyager side sensor. Called by *cpresp.f.*

displn.f

A function that computes the beam displacement in the Voyager main sensor, as factor multiplying *vx/vz*, *vy/vz*. Called by *cpresp.f.*

dphi.f

A function required for the evaluation of the integrals over *vx* and *vy*. Called by *pmodel.f* and *xbarp.f*. *xbarp.f* is in turn called in *abccur.f* and *labcur.f*.

fun.f

This subroutine evaluates a function required to calculate XBAR. Not sure where it is used or where it belongs.

labbi.f

Part of Alan Barnett's code.

```
C       THIS SUBROUTINE COMPUTES THE MAIN SENSOR RESPONSE OF THE
C     VOYAGER PLS EXPERIMENT OPERATING IN EITHER THE M-MODE OR THE
C     L-MODE TO A PLASMA WHOSE DISTRIBUTION FUNCTION CAN BE MODELED
AS
C     CONVECTED MAXWELLIAN.
```

labcur.f

Part of Alan Barnett's code.

ldcur.f

D sensor response, either from Alan Barnett or VMV's *pmodel.f*. Called in *xsimul.f* and *xfull.f* and *xanal.f.*

redart.f

Routine to fill the MNTABL and SDTABL common blocks with the sensitive area information of the main and side sensors. On the SUN system the routine originally read a binary file *redart.data* which can be found now in */nfs/faraday/data/plas7/d3/vgr/const/redart.data*. That file has been converted to an *ascii* file *redartascii* on the SUN system which is readable on any machine. That file is stored in *Voyager_PLS_v_01/software_v_01/mjsanl/common*, and *redart.f* expects that to be the location (see code in *redart.f*: *open(38, file= '../common/redartascii', form='formatted')*).

transd.f

Something to do with transparency, called from *cpresp.f*

transn.f

Something to do with transparency, called from *cpresp.f*

xbarp.f

Not sure what this is. Called from *abccur.f* and *labcur.f*, so must belong to Alan Barnett.

xcur.f

Something to do with Barnett's thesis, written by RLM. The "Rectangular Aperture" Approximation". Called by *xsimul.f.*

xecurnt.f

Not sure what this is.

xcurnt.f

Current calculation assuming unit response. Calls the routine *tderf.f* described below. See Section 3.2.3 of *VI-PLS.*

Routines Associated with Non-Linear Least Squares Fit

eigsrt.f

Called in *resolv.f*, which in turn is called in *xmjsft.f.*

resolv.f

Something to do with the fit precision.

tderf.f

Evaluates the *erf* function. This function is used in evaluating the unit response of the instrument (see equation (3.2.3.8) of *VI-PLS.*

tqli.f

Something to do with non-linear least squares fits. Called in *resolv.f*

tred2.f

Something to do with non-linear least squares fits. Called in *resolv.f*

xunpck.f

Unpacking fit parameters. See the routine *xpack.f.*

xanal.f

Analysis routine, called by *mjsanl.f.*

xfull.f

For computation of effect of feed through from electrons, part of the FULL UP MODE analysis. Needs to extensively rewritten. Never called in the current version of the code.

xfncdv.f

Code to calculate function and derivatives for non-linear least squares fits.

xinv.f

Matrix inversion

xjc.f

Computes currents using *xcurnt.f.*

xparm.f

Setting up the logical flags and the variables to be fit.

xmjsft.f

Non-linear least squares.

xpack.f

Packing up the fit parameter array. See the routine *xunpack.f*.

xsimul.f

Not sure what this is doing, it calls PMODEL.

xtest.f

"FAST" APPROXIMATION TO XCUR (SEE BELOW). 3/6/85 RLM

Routines Associated with Data

iandh.f

Sometime to do with decoding the status word. Called by *setjtl.f* (which in the present code is never called). Although I include it, I use a different method to unpack the status word. See lines 728 through 741 of *mjsanl.f*.

kntcur.f

DN to femtoamps current conversion.

kntflt.f

Something to do with bad currents in currents due to interference. " If the is no signal in cup A, (at high energies) then any in B is suspect. This is a phase problem in the amplifiers we think.".

kntpk.f

Peak finding routine.

limits.f

Setting limits on matrix elements.

mjsanl.f

Basic analysis routine.

nxtmod.f

My routine to decide whether to read *vgr* (cruise) files or *spl* (encounter) files. See Section 7.3 and 7.5 of *VI-PLS.* The decision is based on the user's response to the question
INPUT IPLNT(0,1,2,3,4 CRU JUP SAT URA NEPT)

nxtvgrf.f

My routine to read *vgr* (cruise) files. See Section 7.3 of *VI-PLS.* A stand-alone example of using this program is available in *Voyager_PLS_v_01/software_v_01/vgr_read*.

plsbeg.f

This routine sets up lots of tables and constants. The condition for a fit to be finished, based on comparing successive values of the chi squared, is *EPS*, which is set here in the data statement on line 327.

setjtl.f

I don't understand what this routine is supposed to do, and I do not call it. Called in *kntcur.f* and *mjsanl.f*.

vgrtrd.f

This routine reads the *ascii* version of the mission (*vgr*) files, and is called by *nxtvgrf.f*.

writedat.f

This routine is the interface between the data and the *gnuplot* routines, *gnutest_L* and *gnutest_M*. It is called first in *mjsanl.f*, where the user is given the option of plotting only the data with no fits. It is then called multiple times in *xmjsft.f*. The user is first given the option of plotting the data plus the initial guess (no iterations), and then for each iteration plotting the data with the new estimate of the fit parameters. It is called one last time in *xanal.f* when the program has decided the fit is good and has come out of *xmjsft.f*. The user can then plot the data and the final fit parameters.

The scheme is the following: for each requested plot, *writedat.f* writes out a data file *data.txt* with the data and the fits, if requested. Then *writedat.f* calls *gnuplot* to plot that requested plot. Note that even though mjsanl can handle up to 15 fit components, the routines *gnutest_L* and *gnutest_M* will only plot up to **seven** fit components. **If you want to plot more than seven you must add lines in the gnuplot routines**, as indicated in comments in *gnutest_L* and *gnutest_M*.

Routines Associated with Trajectory/Rotation Matrices

epsiln.f

Parity of a permutation on NARG integers. Called in *onaxr.f*

phase2.f

This subroutine provides trajectory information for the encounters. It is called from *mjsanl.f*, if the user inputs one of the planetary options. See line 1038. It also evidently changes the period of rotation of Uranus, as per the following comments in the file:

```
c  uranus period is changed from 15.56 to 17.24 hours
c  jupiter, saturn and neptune use rotation rates on the sedr
```

phase3.f

This subroutine provides trajectory information for cruise. It is called from *mjsanl.f*, if the user inputs chooses the cruise option.

onaxr.f

Called by *rotmtx.f*.

readdata.f

Read sedr information from an ascii file

rotmtx.f

Construct a rotation matrix given two vectors in the primed and unprimed system. Called in the *sedr* handling routines *phase2* and *phase3*.

treadbatf.f

Reads an sedr file.

vsccp.f

Something to do with coordinate transformations.

vtrns.f

Coordinate transformation plus added constant. Called by *pmodel.f*, *vsccp.f*, *xanal.f*, *xfncdv.f*, *xparm.f*.

vxyzsp.f

Gets the spherical coordinates of a Cartesian vector.

Miscellaneous Routines

calanl.f

Analysis of the calibration modes of the instrument. Not implemented. Called by *mjsanl.f*.

cave.f

This routine provides an option for averaging over multiple spectra before doing the analysis. Called from *kntcur.f*, but there appears to be similar code in *mjsanl.f*.

dummy.f

No idea what this routine does. Defines three entry points. Does not appear to be called.

dati.f

Gets date and time for the completion of the fit. Called by *xanal.f* and *xfull.f*.

gi.f

Compute Bessel function of order 0 or 1.

l1x.f

Bit manipulation. Used in unpacking status word.

mjstem.f

SUBROUTINE MJSTEM CALCULATES RESISTANCE FROM DATA NUMBER AND THEN LOOKS UP TEMPERATURE IN TEMP/RES TABLE

nchar.f

Computes NUMBER OF CHARACTERS IN NUMBER.

numstr.f

plcl.f

Dummy routine? Never called.

pltst.f

Plotting. Includes *clrtek* routine. Not used.

sclr.f

Vector handling routines, dot and cross products

setdata.f

CalComp lotting. Not used.

so31286.f

Have no idea what this does, but I think it is important to keep. Includes *so31286.fc*.

sodint.f

Have no idea what this does.

ss.f

CalComp plotting. Not used.

stncur.f

Have no idea what this does.

strequ.f

Have no idea what this does.

teqab.f

Matrix multiplication?

teqatr.f

Matrix manipulation?

timer.f

Cpu time. Not sure whether this is used or not.

vctr.f

vector manipulation routines. Never called as far as I can see.

xtimer.f

Gets *cpu* time since last *xcall*. Not sure whether this is used or not.

xtyme.f

RETURNS THE NUMBER OF MILLISECONDS SINCE THE BEGINNING OF 1977.

mvc.c

Moves bytes.

mvc5.c

Moves bytes.