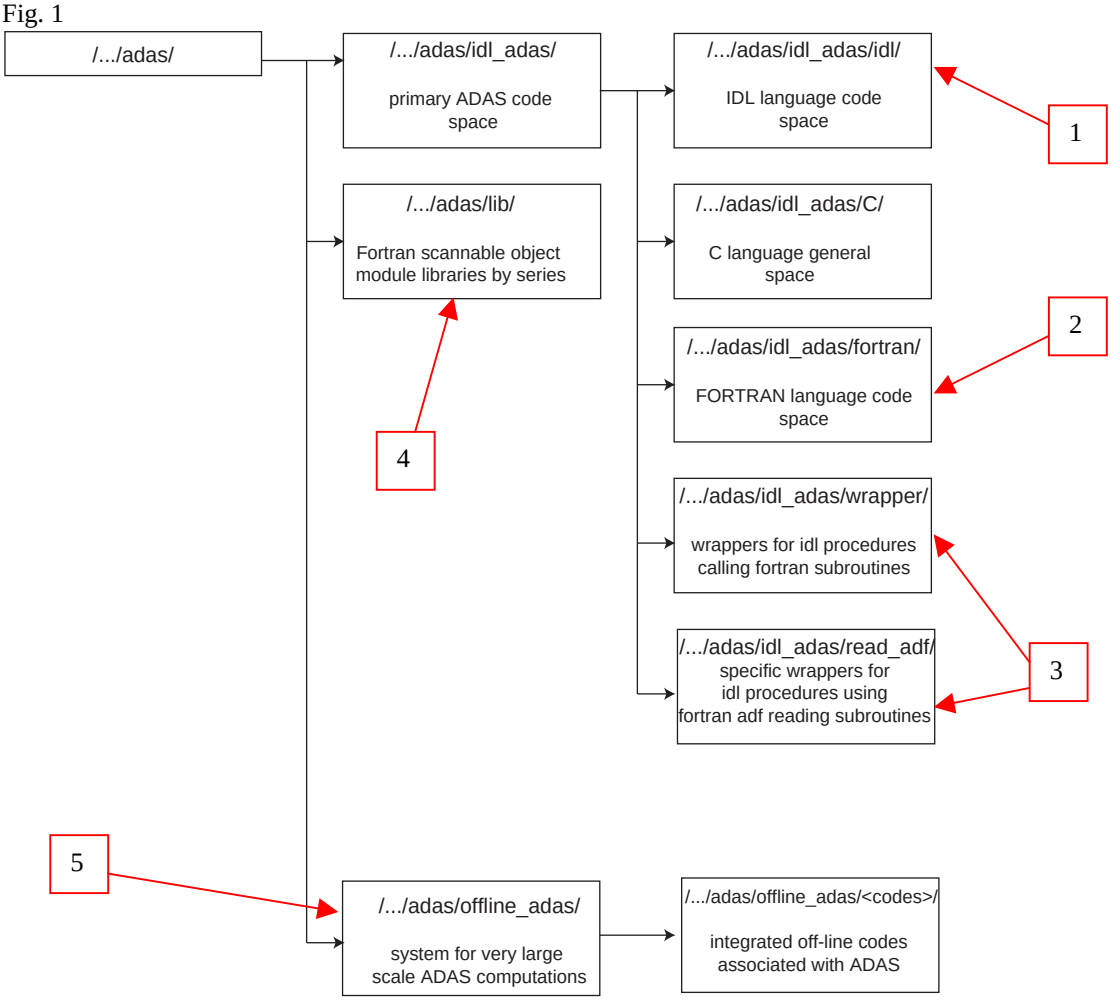
There are rather a lot of structural changes and overhauls of key codes as well as the usual bug fixes for this release together with a large amount of new data. Incidentally this release is version 2.7 since we include a new main program – ADAS8#1, the first of the *offline_adas/* programs anticipated in the user manual (appendix B). The heavy species development has caused the principal trials and tribulations and has had a major knock-on effect in that it has prompted us to rationalise a lot of code – especially *adas2xx* and *adas4xx* – and to organise and improve the subroutines giving access to ADAS data formats. The latter problem is highlighted by the adf04 specific ion file class. This format has evolved over the years and we have many separate subroutines which access adf04 (*bxdata*, *badata b8data* etc.) but mostly accessing some part of the data and not adf04's full glory. We are seeking to get rid of some of these, now unnnecessary, variants. Finally we have upgraded ADAS408 to handle power filtering properly – that means including arbitrary filter files (adf35), such as those produced by ADAS414 using Henke data, which we set up the machinery for some time ago.

So I wish to split the bulletin up into five parts as follows:

1. Review and update on the disposition and use of access routines to ADAS data formats and codes
2. An introduction to ADAS8#1
3. The detailed corrections and additions to codes.
4. New and replaced data - note that data <u>in error we replace</u>, upgraded data is given a new name
5. The upgraded ADAS408.

Fig. 1



1. Access to ADAS data and codes from IDL and fortran

I draw your attention to a chart (figure 1) from the ADAS user manual appendix B showing the disposition of ADAS code. Passing down into the IDL code region at (1), we have figure 2. At (2.1), the directory *read_adf/* contains the IDL procedures such as *read_adf01.pro* which allowing reading of (selected) data from the appropriate adf number at the IDL command line. Also note at (2.3) procedures which allow running of a complete ADAS code at the command line taking parameters from the command line rather than from widgets. We have added to both these categories in this release. Note at (2.4) a new type of named procedure, such as *xxdata_04.pro* which read a complete adf file from the IDL command line. They follow exactly their associated fortran subroutine (see below) in positional parameter arrangement. Note that *read_adf04.pro* remains the normally preferred access from IDL..

Fig. 2

We have introduced a parallel *fortran/read_adf/* directory in the fortran code region as shown in figure 3 [following on from (2) in figure 1]. The aim is to access ADAS complete adf datasets via *xxdata_<XX>.for* routines stored in *fortran/read_adf/* which will have their own scannable object module library *libread_adf.a* as shown at (4) in figure 1. Only a limited number have been prepared so far. A number of old style <letter><number>data.for routines have been replaced and although the old versions will remain they will not be updated. (3.1) in figure 3 shows the *xxdata_YY.for* routines which have been put in place in this release. The fortran scannable object module libraries which users can access from their own codes are summarised in figure 4 –in particular at (4.1) note the new *libread_adf.a* one. Note that for IDL procedures calling fortran subroutines, we operate via wrappers and a C interface. The interfacing codes are of less interest to the general user but their position is

noted at (3) in figure 1. This whole arrangement means that it will be easy and tidy to introduce calling from other languages such as MATLAB – a demonstration version of which is in preparation.

Fig. 3

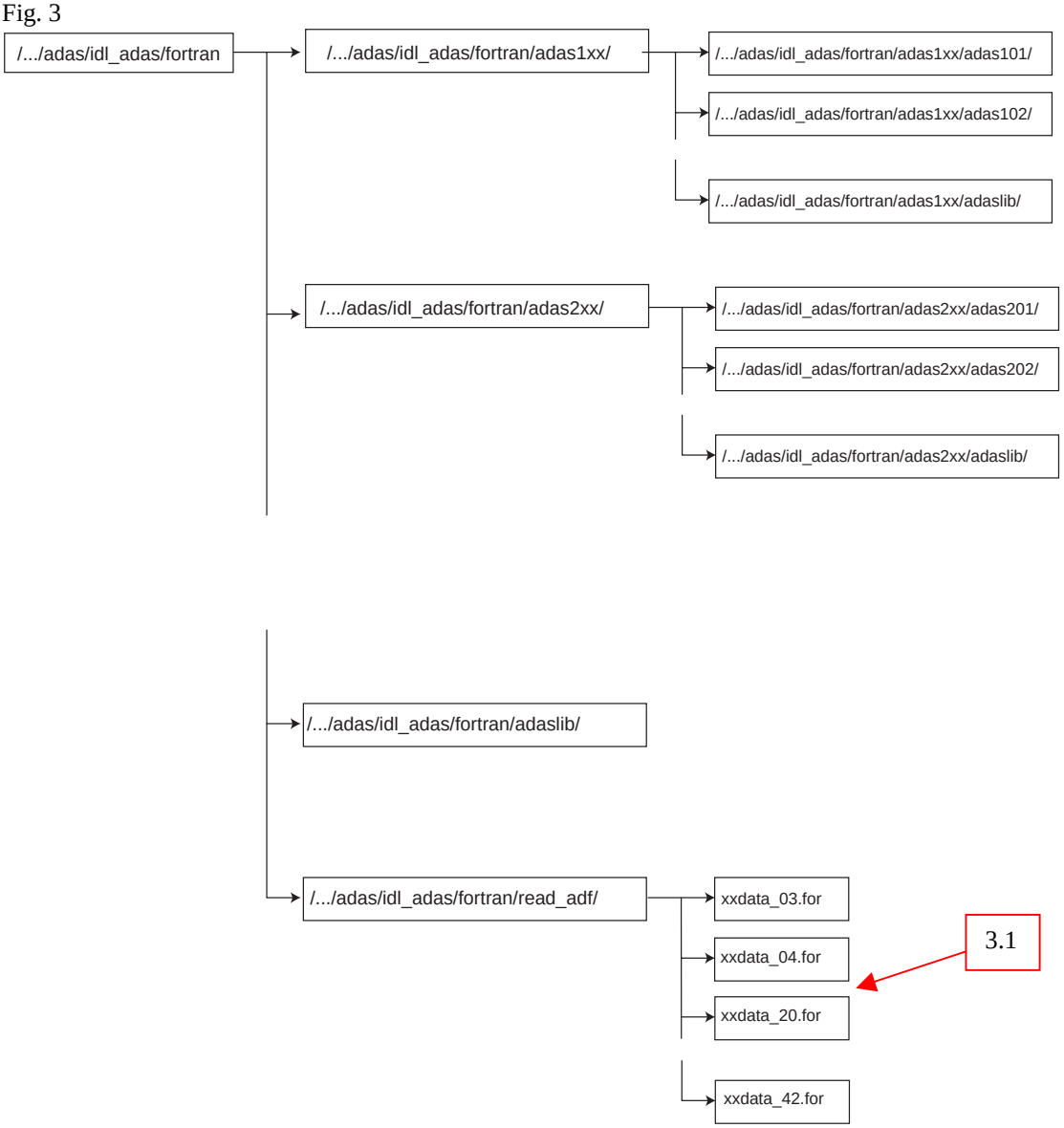
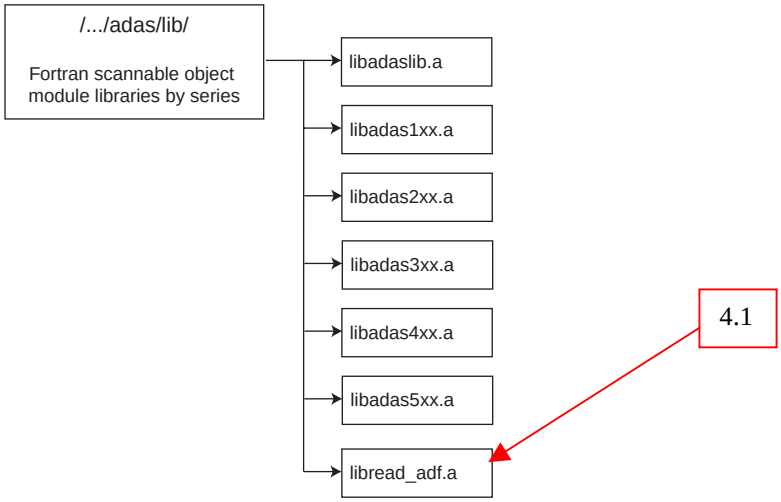


Fig. 4.



2. ADAS8#1 for the offline calculation of heavy species ionisation and emission data

A new principal directory *offline_adas/* has been set up as shown at (5) in figure 1 to hold the offline codes. At this time, we are releasing ADAS8#1. ADAS8#1 is essentially doing the work of ADAS801, ADAS407, ADAS408 and ADAS810 for all the ions of a heavy element in one go. It is set up to execute on large (probably) parallel machines under scripts. The scripts need to be adjusted for the disposition of code and data on the user's machine. The source code should reside in central ADAS, but the compilation script should be adjusted to put the executables at the user laboratory's chosen location.

Fig. 5.

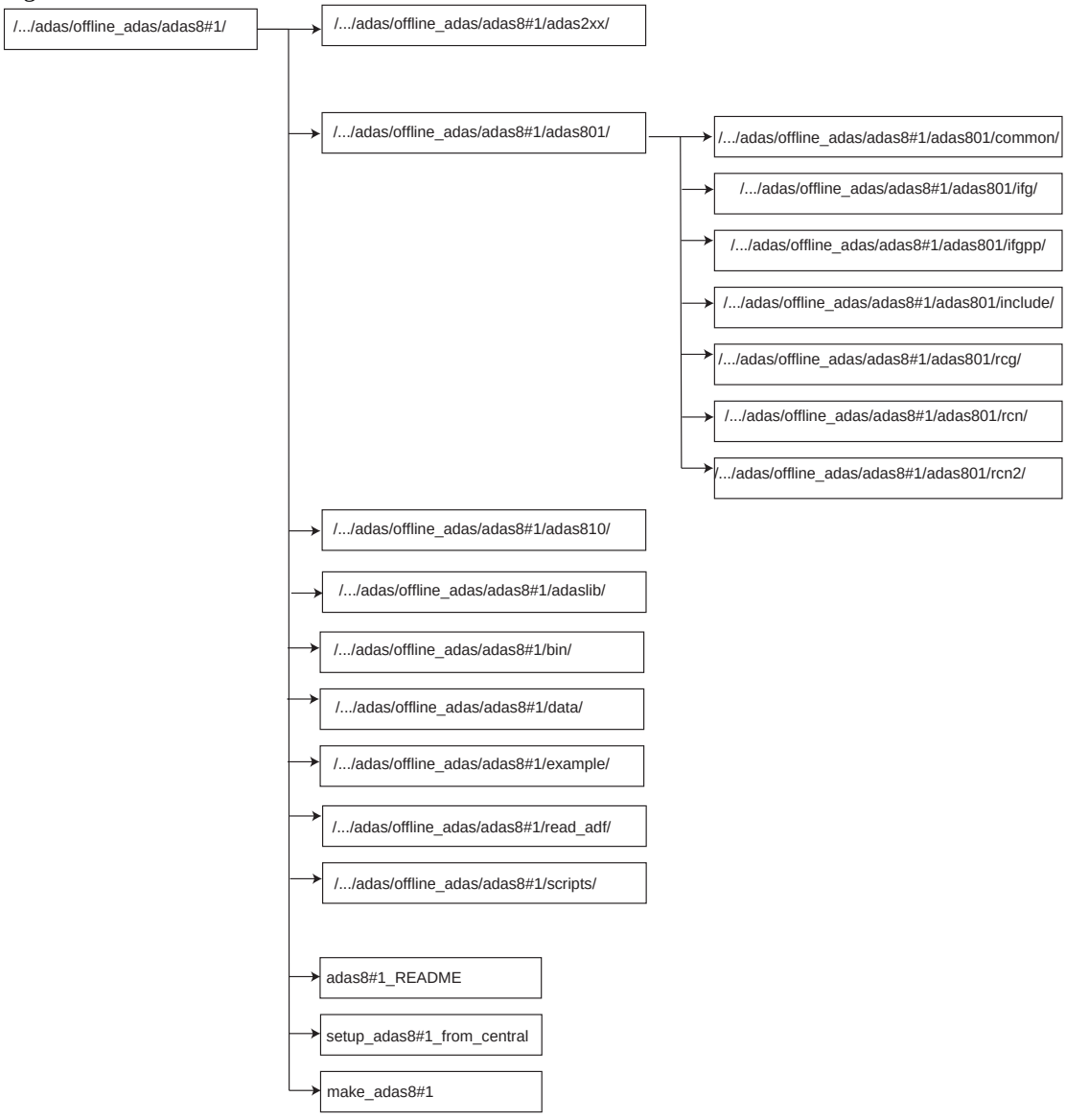


Figure 5 shows the sub-directory layout for ADAS8#1. Note the equivalence to the online organisation. Mostly the online routines are used but copied into the new structure. It will be an ongoing necessary task to keep the on-line and off-line systems properly aligned.

Concerning the directories as shown in the above figure within adas8#1:

adas2xx/: Contains the required fortran series 2 subroutines. These come from *adas2xx/adaslib adas208* and *adas210* but are all placed in the one directory.

adas801/common/: The same layout as in *fortran/adas8xx/adas801/*. Likewise *ifg*/, *ifgpp/, include/, rcg/, rcn/, rcn2/*

adas810/:	Same as *fortran/adas8xx/adas810* but *adas810.for* replaced with *adas810_offline.for*.
adaslib/:	Required files from *fortran/adaslib*.
bin/:	Location of the binaries *adas810_offline.x, ifgpp.x, ifg.x, rcg.x, rcn2.x* and *rcn.x*
data/:	Location for coefficients of fractional parentage. This has more f-shells than the central version.
example/ :	Xe39+ sample files.
read_adf/ :	*xxdata_04.for* and *xxdata_42*.for are copied to here.
scripts/:	*run_adas8#1* - a perl script which runs Cowan code and (optionally) the adas810_offline code.

For this first release the *adas8#1/adas801* and *fortran/adas8xx/adas801* are identical. We hope to keep them in synch but allowing the respective *include* files to differ.

Some top level files are also present:

README : Gives the user some idea of how to use it - this should be refined in time.

setup_from_central : Copies central ADAS files to *offline_adas/adas8#1*. There should be no need for an end-user to use this script.

make_adas8#1 : Compiles the fortran by calling the makefile in each directory.

We expect that we would not compile or set the paths for adas8#1 code. The user would copy it to his machine of choice and enable it himself. Any changes necessary should be communicated back to us for incorporation into the next release. Each release we shall keep the central ADAS files up to date. There are some omissions which we have not had time to finish. Allan's script to farm out the jobs to different processors on parallel system is not included.. An offline version of ADAS407 is also missing. We shall correct this soon and use offline version of ADAS407 and ADAS408 in the script. However the current online version copes with the larger adf04 files. This means that the ADAS407 and ADAS408 steps are done with interactive IDL-ADAS at this stage.

Then the steps to get the ADAS8#1 system running (to be considered along with the flow chart of figure 6 below) are as follow:

1. Compile by running the make_adas8#1 script. It may be necessary to change the location of the sh (bash or ksh family) shell location.

2. Alter *run_adas801* script to customise paths and file locations
   line 1 : perl path. Use "which perl" to find out where perl is on your system.
   Find "# Site specific location details" and alter the $fdir and $cfp variables to point to your file locations.
3. The example/ directory contains an Xe39+ example. Copy the *test_adf34.dat, test_inst.dat* and *test_pp.dat* files to your working directory and then *run_adas8#1* from the scripts directory. This generates the adf04 files only. In general you prepare the drivers for the ADAS801 part (*_adf34.dat* normally archived in *adf34/* and *inst.dat and pp.dat files*- normally archived in *adf41/)* and the drivers for the ADAS810 part *(_pec_drv.dat* normally archived in *adf42/*) for an element by executing ADAS808 in IDL_ADAS.

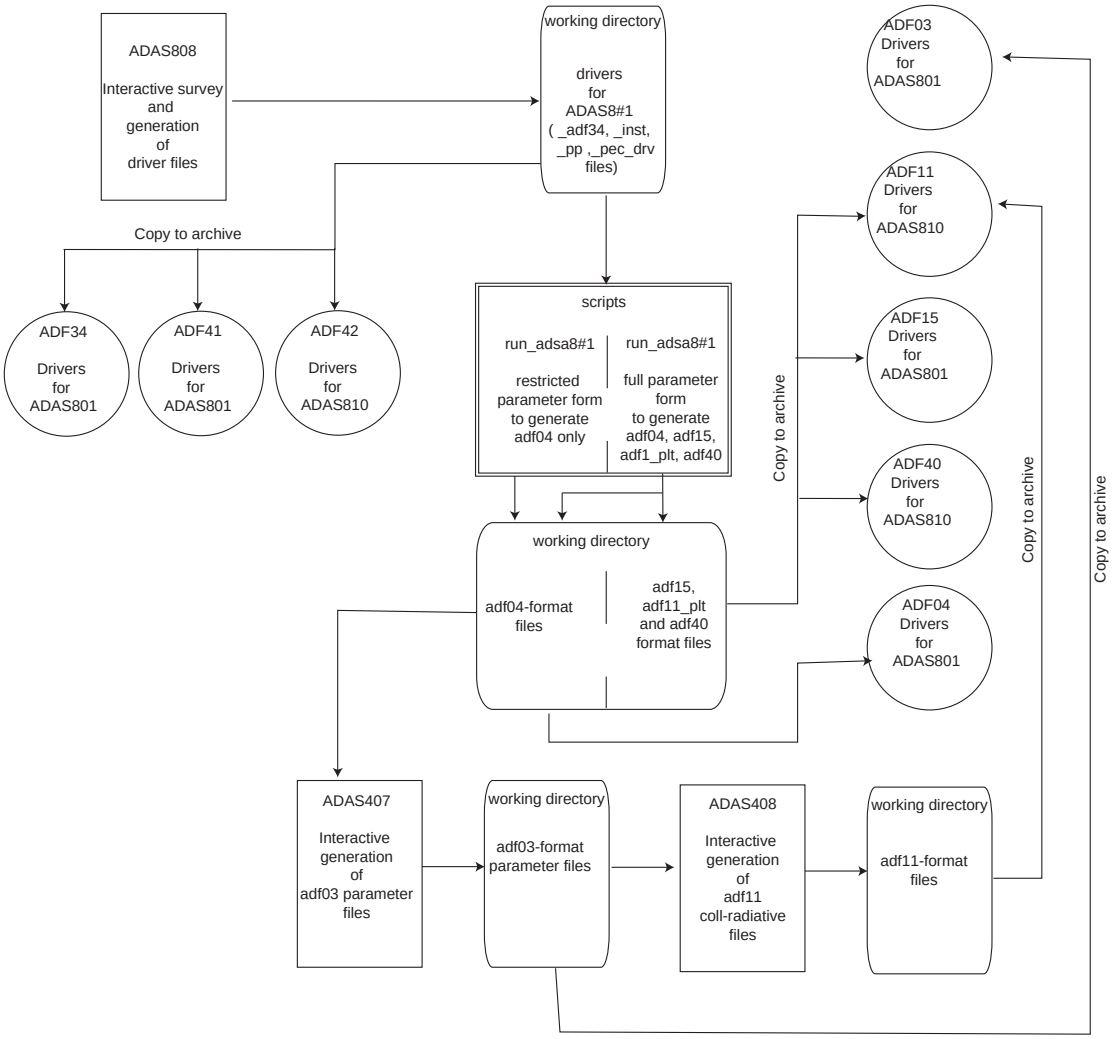
   The *loadleveler_submit* command is for those (lucky?) people with the LoadLeveler batch submission system.

4. Use ADAS407 and ADAS408 in IDL-ADAS for the moment to produce the adf03 files and then the main adf11 files

5. To produce adf15 pec, adf11 plt and adf42 fpec files (as well as adf04 files) copy the *test_adf42.dat* example and launch *run_adas8#1* with extra arguments as: ..
   ./offline_adas/adas8#1/scripts/run_adas8#1 test_adf34.dat test_inst.dat test_pp.dat test_adf42.dat.

Note that we can get the usual *paper.txt* file by adding its name after the *test_adf42.dat* in the command line above.

6. The output files are set in *test_adf42.dat*.



3. Corrections and additions to codes (ADAS v2.6 to ADAS v2.7)

C.1 Adding to the set of IDL run time versions of the interactive codes is *run_adas204.pro*.

C.2 The outer Te/density loop in *run_adas405*, which allows it to calculate arbitrary size inputs, redefined the loop variable inside the loop. This causes it to terminate early rather than failing spectacularly. (Fortran doesn't allow this!).

C.3 The *cw_adas8xx_infile* is out of step with *cw_adas_infile*. ADAS809 calls it but the only difference is that 809 requires a smaller number of lines in the input window - this functionalilty is provided by the *ysize* parameter in *cw_adas_infile.pro*.

Minor (non-fatal) syntax error in *adas809.for* call to *h9spf1*.

C.4 ADAS801 fortran has been synchronised with the offline ADAS8#1 version. Parameters may be different but the code should be the same. A new include directory, updated comp scripts and new binary names (*rcn.x, rcnx.x, rcg.x, ifg.x* and *ifgpp.x*) necessitate some changes in the make script and in the 801 IDL branch.

Note that *ifgpp* has been changed frequently since the last release.

Subsequent change to *rcg/coeff.for* to fix NAN problem uncovered by Thomas Kreucken. A variable was not initialised but this error does not invalidate any previous results.

Replace the infamous 'stop 57' with a 'stop Too many transitions' message in *rcg*. The logic in *spectr.for* is extremely convoluted and appears to use an energy grid overrun to test whether too many transitions are included.
The idea is that collision calculations require all transitions so they cannot be treated in batches. Some new variables were added to allow for a future separation between spectrum lines and maximum number of collisions.

For now set *klam* equal to *ktran* and trap the error in the interactive ADAS801. Output *ktran* at start of *rcg* and add a new function to ADAS801 to compare this with the number of spectral lines produced. If there are too many, pop-up a message and do not continue with the ADAS801 program.

In the *born.for* subroutine the interpolation of the generalised oscillator strength fails occasionally when using Aitken interpolation (*aknint.for*) and a negative value is returned. In these cases we try *xxspln* and if this fails the value is set to zero.

Batch operation of ADAS801 has now been activated.

C.7 New *idl/adas8xx/adaslib* directory with Allan's adf34 file verification code.

C.8 Add Hugh's *xxdata_04.f*or routine which reads adf04 files of arbitrary complexity. Two changes made to his original version: Replace the inline function, *indx*, with a call to *adaslib/i4idfl.for*. A new parameter, *itieactn* which determines how *xxdata_04* deals with untied levels. This is an integer switch and acts according to

0 : default - stops if untied levels are found.
1 : continues and fills *ltied* array but note that further processing will probably fail.
2 : effective untied levels.....NOT YET IMPLEMENTED

xxdata_04 requires *xxpars.for* and *xxprs1.for* in adaslib.

The introduction of this class of routines allows us to take the opportunity to gather all adf reading routines together. The IDL branch has begun this process so we continue with the same idea on the FORTRAN side. Introduce the *fortran/read_adf/* directory with associated library, *libread_adf.a*. Then to read any ADAS dataset from a user's program will require linking against -*ladaslib* and -*lreadadf*.

The ultimate aim must be to rationalise ADAS such that the *fortran/read_adf/xxdata_YY.*for routines are the only ones used for reading ADAS data. We can migrate to using the new system slowly by moving and renaming the various <*series*><*member*>*data.for* routines.

All codes which used *bxdata* have now been converted to use *xxdata_04* (C.9 below). Therefore a warning had been added to *bxdata* advising that it is deprecated. It will still run and there is no need to remove it from the system but it may gradually lose functionality.

C.9 The heavy species work has motivated us to begin replacing many different adf04 reading routines with *xxdata_04.for*. This affects quite a number of codes and routines. The opportunity for some tidying and rationalisation was taken at the same time.

Changed and new library routines.

adas_progressbar.pro - new general standalone widget which increments a progress bar by listening to output from fortran code. The number of steps is required input.

xxminv.for - dimensions increased (1200 max levels).

bxsetp.for - sends level identifer string to IDL. This was restricted to 99 levels irrespective of number in adf04 file. ADAS206 used this list for both metastables and transition table which overflowed. Change back to sending all levels and alter IDL to ask for metastable in an embedded panel (no longer pops up). Change 205 to keep appearances the same!

bxout0.for - format change to accommodate up to 9999 levels and 99999 transitions.

bxttyp.for - As this is usually called immediately after reading an adf04 file it was aligned with all the data returned from *xxdata_04*. The original version was replaced with *b8ttyp*. Affected codes are 201, 205, 206, 214, 407 and 412.

Codes affected by the above changes are:

All: Increase NDLEV to 1000 and NDTRN to 15000 and alter output file formats to accomodate these new maxima. This may cause problems if paper.txt is parsed.

ADAS201: Dimensions only.

ADAS205: As well as accepting the larger adf04 files

- Put up a progress bar during calculation. It scales as O(n**3) so we can be waiting for a long time.
- Any level can be designated as a metastable. Replace the pop-up selector box with one that can be set from the processing screen and scrolled to see all the levels. As it's energy ordered we rarely need to scroll!
- Hugh's change to put on the output graph only those levels which are plotted.

ADAS206: Same changes as ADAS205.

ADAS207: Can read new, larger, contour and adf04 files. Because of the increased number of transitions the selection of lines can become too large, even with the wavelength limits in place. IDL may refuse to render the widget or it may generate an X-server error.

- Transition selection is now a moveable window controlled by advance/retard 'tape recorder' buttons.
- *b7datc.for* has been rewritten to do a consistency check between the adf04 and contour files. Previously it read in the adf04 data also.

ADAS407: Lots of changes here in both functioning and at the user interaction level.

- All information is now read in from adf04 files and they are no longer rewound to read in extra information.
- The analysis routines, *d7exps*, *d7alfs* and *d7auts* have been tidied up, made implicit none, and redundant varaibles, calculations and format statements have been removed.
- It is no longer necessary to have a fake fully stripped adf04 file. The program knows when the first one in H-like. At the IDL level the option is de-sensitised.
- A bug whereby the metastable selection list was one pixel high when starting without a defaults file has been fixed.
- In the automatic branch a file prefix can be specified - adf04 files no longer need to conform to *ls#<element><iz>.dat*. Now they must look like *<prefix>#<element><iz>.dat*.
- In the automatic branch the type of parameterisation (A or B) can be chosen. Hitherto it forced type A.

- In cases where there is no spin system connection between adjacent ions we now force it to be ground. The proper solution is to use IC and not LS input files. A warning is sent to the screen. Note this only affects the adf03 file generation and not the mainbn (adf25) ADAS204 input files. The automatic branch will not produce these files in any case.
- The defaults file has changed.

ADAS412: Use *xxdata_04* to read in adf04 files but do not increase NDLEV and NDTRAN yet.

ADAS214: Use *xxdata_04* to read in adf04 files but do not increase NDLEV and NDTRAN yet. Essentially the same change as 412 but this has *bexcoef.for* rather than *xcoef.for* which are almost identical except for one additional output parameter. We should perhaps eventually make one universal routine!

ADAS810: Bring into line with new names and some minor alterations.
- Change *badata* and *b8ttyp* to *xxdata_04* and *bxttyp*.
- Move *haddat* to *read_adf/xxdata_42.for* to keep consistency in reading adf files.
- Increase dimensions (levels to 1200)
- Write title of run to paper.txt.
- *bxcoef.for,* the population calculation routine, had its dimensions increased to stay consistent with ADAS810. Unfortunately it's not possible to make it completely independent as there are internal dimensions which must be matched to calling sizes.
- Checking of metastables selected is improved at the IDL level.

C.10 Unfortunately intended changes to ADAS808 have not yet been made. It certainly remains sub-optimal.

C.11 Allan has extended *read_adf15.pro* and *read_adf13.pro* to return the atomic number and ion charge as optional outputs. The data is read from the first line of the file and although it is not part of the formal specification it is present in all central ADAS files. If it is not present values of -1 are returned.

Also increased the number of blocks *read_adf15* can accept in an adf15 file. There were some datasets in central ADAS which could not be read completely.

C.12 Dimension mismatch problem with *read_adf21*. Incorrect results were returned when more than one stopping impurity was requested.

Take opportunity to extend the number of energy/density/temperature points by blocking data in groups. Makes it similar to other *read_adfXX* routines.

Note that read_adf22 acquires these changes automatically.

C.13 Lingering IDL v5.5 array of 1 problems. This time in ADAS208 in the popup 208 version of 502 to get ionisation rates. This occured when checking the polynominal fit parameters. Why this option is there at all is perhaps now questionable.

C.14 Add extra error checking to *xxtext.pro* to check whether a file exists before trying to display it. Rare error condition if a directory is selected and browse comments is possible.

C.15 As *xxdata_04.for* is now the preferred method of accessing adf04 data, the *read_adf04.pro* routine has been changed to use it. This could have been done by adding a *read_adf04/*

directory with *readadf04.for*, *read_adf04.c* etc. as in some of the other routines. But the IDL *read_adf* routines are not direct implementation of the fortran access routines; typically they are more like the series 5 codes. Also the more complete set of data returned from *xxdata_04* is useful when other fortran routines acquire IDL wrappers.

Therefore there is now an IDL versions of *xxdata_04*. The argument list is the same as fortran except that the file name is passed rather than a unit number (and the adf04 file opened outside *xxdata_04*) and logicals are replaced by integers (0 and 1 for false and true) because IDL does not have a logical type. The parameters are not keywords so position and correct positioning are important. However it is not necessary to declare arrays before calling it.

Also required was an IDL version of *bxttyp*. Arguments are the same as the fortran and the same warnings about number and position apply.

These routines allow for a simple change to *read_adf04.pro*.

Note there is a new directory, *idl_adas/wrapper/read_ad*f to store the code for interfacing with the *fortran/read_adf* routines.

read_adf04.pro now traps for the case where there are no *S*-lines or no *zpla* data in the adf04 file and the */ecipcalc* option is chosen. Previously this would cause a crash.

C.16 A new *read_adf20.pro* for reading in adf20, G(T), data. We have taken the opportunity for making *fortran/read_adf/xxdata_20.for*.

Although G(T)s are mainly used in astrophysics where kelvin is the preferred temperature unit, the default assumption here is that Te is in eV to keep consistency between the various *read_adfXX* routines. There is always the */kelvin* keyword.

C.17 ADAS506 was changed to use *xxdata_20* rather than *e6data*. Just like *bxdata*, *e6data* now puts up a warning advising that it is deprecated. It still remains in central ADAS but will not be updated

C.18 As the batch system at JET keeps changing remove the handling of this to a new subroutine, *loadleveler.pro*. *batch.pro* still has some JET specific handling but it is now just a few lines.

C.19 ADAS408 has been overhauled and now gets its filter data from adf35 datasets rather than from a built-in routine which assummed a Be window and Si diode. This routine was actually quite poor in generating the transmission function. The interpolation routine, *d8tran*, has been re-written and it no longer calculates the transmission fraction from formulae with embedded coefficients. It was necessary to account for the various absorption edges and some extra library routines were required. The running time may be longer but this penalty will only occur if an adf35 filter file is chosen.

The ADAS408 screens have been completely changed.

The new adaslib routines are

i4indfvs.for:	Finds closest index in a non monotonic array. Note that vs stands for variable spacing!
xxpint.for:	Simple polynominal interpolation.
xxmerg.for:	Merges two grids and eliminates any duplicate enteries. Uses some netlib and BLAS routines which are included in the subroutine (following *xxeign.for*).

Note also that *xxdata_03.for* replaces *d8data.for* to keep with the new data access naming.

*** Remove all files from *fortran/adas4xx/adas408* and *idl/adas4xx/adas408* and replace with new set.

C.20 A *read_adf35.pro* was added also. This is similar to *read_adf04.pro* in that a *xxdata_35.pro* is also provided. This uses a wrapper version of *d8tran* to interpolate user requested values.

C.21 Files which should not be present! Remove from *idl_adas/idl/adas4xx/adas414* the following

plot_diff.pro
phenke.pro
plot_henke.pro
onoff.pro
filter.pro

C.22 There was a problem plotting the total energy excess plot in ADAS406. The *adas406_4_plot.pro* routine looks as if it was copied from another and edited without being tested. Two variables (first & last) were not set so I suspect that this option was never used. Odd considering that it was at version 1.2!

Add ADAS406 as another interactive code which has an IDL run time version (*run_adas406.pro*).

C.23 A new library *libreadadf.a* has been made in */home/adas/lib*

C.24 A late change has been made to ADAS205 and ADAS206. If ionisation from excited levels is chosen as an option and the adf04 file contains levels above the ionisation potential, then the resulting populations can appear as NANs. In this situation we switch off ionisation from ALL levels.

C.25 We have brought ADAS701 and ADAS702 into line with Nigel's latest versions (*autostructure* v1.18 and *adasdr* v1.10).

The file opening system in ADAS702 requires some changes. The input files expected o1, o2, o3 and o4 (or their unformatted equivalents o1u, o2u, o3u and o4u) are input as before. Previously these were opened simultaneously in ADAS702 with *g2open*. Now they are opened dynamically in adasdr.for. Therefore we no longer need *g2open*.

4. Corrections and updates to data (ADAS v2.6 to ADAS v2.7)

D.1 New adf04 files from Nigel and his collaborators:

adf04/belike/belike_dcg03#c2.dat	Don Griffin
adf04/belike/belike_nrb03#b1.dat	Nigel Badnell
adf04/helike/helike_mab03#ne8.dat	Manuel Bautista

D.2 Thomas Puetterich found an error in *adf00/re.dat* - Re28+ had too many electrons. There is a replacement

adf00/re.dat

For adf00/h.dat – adf00/ne.dat, the ionisation potentials are *replaced* with those from Kelly.

D.3 New argon charge exchange and CXS emissivity files are added. The cross section data is from ORNL. CTMC caluclation referenced from Whyte et. al.,1998, Physics of Plasmas,vol.5, no.10. Data from http://www-cfadc.phy.ornl.gov/eprints/argon.html.

adf01/qcx#h0/qcx#h0_ornl#ar16.dat
adf12/qef99#h/qef99#h_ornl#ar16.dat

The Li-like Ar16+ is new

For H-like Ar18+ there are new files which supercede Harvey's 1999 data.

adf01/qcx#h0/qcx#h0_ornl#ar18.dat
adf12/qef99#h/qef99#h_ornl#ar18.dat

We still retain Harvey's adf01 and adf12 files and note that they go to lower energies than the newer versions.

D.4 There are errors in some of the baseline specific line power files used principally in transport modelling. The problem affects some of the adf11/pls89 which were produced during 1990 on the IBM mainframe.

The Be, C, O, Ne, Cl, Si and Ni datasets have data in the wrong units. Unfortuntely none of the adf03 driver files are available as the data format was evolving at that time. A further problem is that these early adf11 files do not record what lines are selected for each stage (the current ones do but using adf15 pecs is a better solution).

It looks like an eV/s to W conversion but this does not hold for all stages and in the absence of a proper identification it is better to re-calculate them with ADAS407.

Use the adf03/atompars/atompars_vm#<el>.dat Abels VanManen data for Be, O, Ne, Cl and Ni as these date from about that time.

There is no Si data so an atompars_mm#si.dat has been added from an ADAS407 run on the adf04/copmm#14 data.

The carbon is the most problematic as the lines in the vm dataset do not make much sense. Therefore we use the lh (Lorne Horton) version instead.

James Spence discovered this problem.

adf03/atompars/atompars_mm#si.dat

adf11/pls89/pls89_be.dat
adf11/pls89/pls89_c.dat
adf11/pls89/pls89_cl.dat
adf11/pls89/pls89_ne.dat
adf11/pls89/pls89_ni.dat
adf11/pls89/pls89_o.dat
adf11/pls89/pls89_si.dat

D.5 Two example adf35 soft X-ray filter files have been added.

adf35/simple_2000.dat:	Simple cutoff below 200eV.
adf35/asdex_example.dat *t*	A 8micron Be/0.2micron Si/0.2micron Si with a 300micron Si detector example which shows the absorption edges clearly.

D.6 We have replaced some erroneous *prc89/* data. These are data sets giving the part of the radiated power driven by charge transfer. A power filter was incorrectly applied. The data sets replaced are:

adf11/prc89/prc89_he.dat
adf11/prc89/prc89_li.dat
adf11/prc89/prc89_f.dat
adf11/prc89/prc89_s.dat
adf11/prc89/prc89_ar.dat
adf11/prc89/prc89_cr.dat
adf11/prc89/prc89_fe.dat
adf11/prc89/prc89_kr.dat

D.7 Non--printing ascii characters were found in a few *plt89/* and *prb89/* datasets. This prevents their being read. The faulty datasets , which have been replaced are:

adf11/plt89/plt89_n.dat
adf11/plt89/plt89_s.dat

adf11/plt89/plt89_kr.dat

adf11/prb89/prb89_n.dat
adf11/prb89/prb89_s.dat
adf11/prb89/prb89_kr.dat

The plt data is the same to all digits but the prb is slight different, probably due to machine precision differences (IBM mainframe vs. linux workstations

D.8 There is a mistake in the 96 H prb data due to a type mis-identification in the IBM mainframe bundle n population code. This only affects H as it was processed separately from the other 96 GCR data. The file *prb96/prb96_h.dat* has been replaced.

D.9 Lots of data and more sequences from the DR Project.

DR.1 N-like data.

New Al, Ar, Cs, Cr, Fe, Mg Ni, S Si, Ti from Dario Mitnik.

adf09/dmm00#n/dmm00#n_<ion>ic22.dat
adf09/dmm00#n/dmm00#n_<ion>ic23.dat

and associated adf27 drivers

adf27/nlike/dmm00#n/<ion>ic22-2.dat
adf27/nlike/dmm00#n//<ion>ic22-n.dat
adf27/nlike/dmm00#n//<ion>ic22_str.dat
adf27/nlike/dmm00#n//<ion>ic23-3e.dat
adf27/nlike/dmm00#n//<ion>ic23-3o.dat
adf27/nlike/dmm00#n//<ion>ic23-ne.dat
adf27/nlike/dmm00#n//<ion>ic23-no.dat
adf27/nlike/dmm00#n//<ion>ic23_str.dat

and adf28 drivers

adf28/nlike/dmm00#n/<ion>ic22.dat
adf28/nlike/dmm00#n/<ion>ic23.dat

Note these were renamed from dmm03 to dmm00 in keeping with ADAS convention of year indicating method and were place with Dario's previous N-like calculations.

DR.2 Li-like data.

Li-like core 1-2 data from James Colgan. Again no adf27/adf28 driver files.

Nigel discovered that the IC (only) 2-2 data was incorrect so he has provided these as *nrb00#li/*. The corresponding files from jc00#li must be removed.

Removed LS files :	jc00#li_b2ls22.dat
	jc00#li_be1ls22.dat
Removed IC files :	jc00#li_al10ic22.dat
	jc00#li_ar15ic22.dat
	jc00#li_b2ic22.dat
	jc00#li_be1ic22.dat
	jc00#li_c3ic22.dat
	jc00#li_ca17ic22.dat
	jc00#li_cl14ic22.dat
	jc00#li_cr21ic22.dat
	jc00#li_f6ic22.dat
	jc00#li_fe23ic22.dat
	jc00#li_kr33icr22.dat

jc00#li_mg9ic22.dat
jc00#li_mo39icr22.dat
jc00#li_n4ic22.dat
jc00#li_na8ic22.dat
jc00#li_ne7ic22.dat
jc00#li_ni25ic22.dat
jc00#li_o5ic22.dat
jc00#li_p12ic22.dat
jc00#li_s13ic22.dat
jc00#li_si11ic22.dat
jc00#li_ti19ic22.dat
jc00#li_xe51icr22.dat
jc00#li_zn27ic22.dat
jc00#li_zn27icr22.dat

Note there are new Be1 2-2,2 LS and IC files in nrb00#li

nrb00#li_be1ic222.dat
nrb00#li_be1ls222.dat

DR.3 Be-like data.

Problems with James' core 2-2 Be-like data. Nigel suggests that these are unsafe and should be removed. They were only there for one release so should not present too much of a problem. A new directory *adf09/nrb00#be* contains the improved data.

Annoyingly the adf27 data uses *belike/jc00#be/* directory with the exception of B+ 2-2,2 case which is in *belike/nrb00#be/*.

Only the B+, C2+, N3+ and O4+ need new adf28 data and these are placed in *adf28/belike/nrb00#be/*. Use James' data for the other 2-2 production.

Removed adf09/jc00#be/*22.dat

DR.4 F-like data.

These data were calculated by Oleg Zatsarinny. His original naming of oiz03 was changed to use a 00 year designation.

The data did not come all at once - there was some in July which was revised in September. This was in addition to the core IC 2-2 which was revised in association with Nigel.

There are adf27 and adf28 files from the IC 2-2 revision by Oleg and Nigel. Generic input files are used for the rest. The examples given were for iron and these have been renamed as

adf27/flike/oiz00#f/fe17ic22-2.dat
fe17ic23-3.dat
fe17ic23-n.dat

Oleg does not use the radial wavefunctions of autostructure but provides his own from the Froese-Fischer code. These are archived in adf27 with a _rad identifier, eg for Ti13+:

adf27/flike/oiz00#f/ti13ic22_rad.dat
ti13ic23-3_rad.dat

The adf28 files for the IC 2-2 are in *adf28/flike/oiz00#f/*. Again the ic23 and ls23 are generic and are present as the Fe17+ example.

DR.5 Ne-like data

Data from Oleg. Again these are renamed to a 00 designation and his naming has been altered to fit the ADAS template.
(icn2 -> ic22 and icn3 -> ic23).

Sample adf27 input files are given for iron only.

Again he does not favour using autostructure's wavefunctions so these are included in adf27 with a _rad.dat identifer.

The adf28 files are the same for all elements but are included in the adf28 as Fe16+.

DR.6 O-like data.

Data from Oleg. Again these are renamed to a 00 designation and his naming has been altered to fit the ADAS template.
(icn2 -> ic22 and icn3 -> ic23).

The IC 2-2 core were correced by Nigel and Oleg before this release.

adf27 files for 2-2 and sample Fe18+ for the rest are present. As before the radial wavefunctions are given as these are not the autostructure ones.

The adf28 are present for the 2-2 core, where the energy corrections are explicitly given. The generic 2-3 IC and LS are represented by examples using Fe18+.

DR.7 C-like data.

The C-like data is also from Oleg but there were problems with the IC 2-2 core data. Nigel re-ran these and the results are in an adf09/nrb00#c directory.

Nigel's adf27 and adf28 file are achived in *adf27/clike/nrb00#c* and *adf28/clike/nrb00#c*

The remaining adf09 datasets are in *adf09/oiz00#c*. These have been renamed to fit ADAS name style.

Generic 23-3 and 23-n (Fe20+ example) and the radial wavefunctions are in *adf27/clike/oiz#c*. Note that the DR files in *adf09/nrb00#c* use the radial wavefunctions given here.

The *adf28/clike/oiz00#c* contains a sample Fe20+ for the IC and LS 2-3 core processing.

DR.8 B-like data.

Sequence calculated by Zikri Altun.

No adf27/28 files and unsure whether the adasdr bug was involved. Therefore hold this over until next release

H. P. Summers
18 Aug. 2003

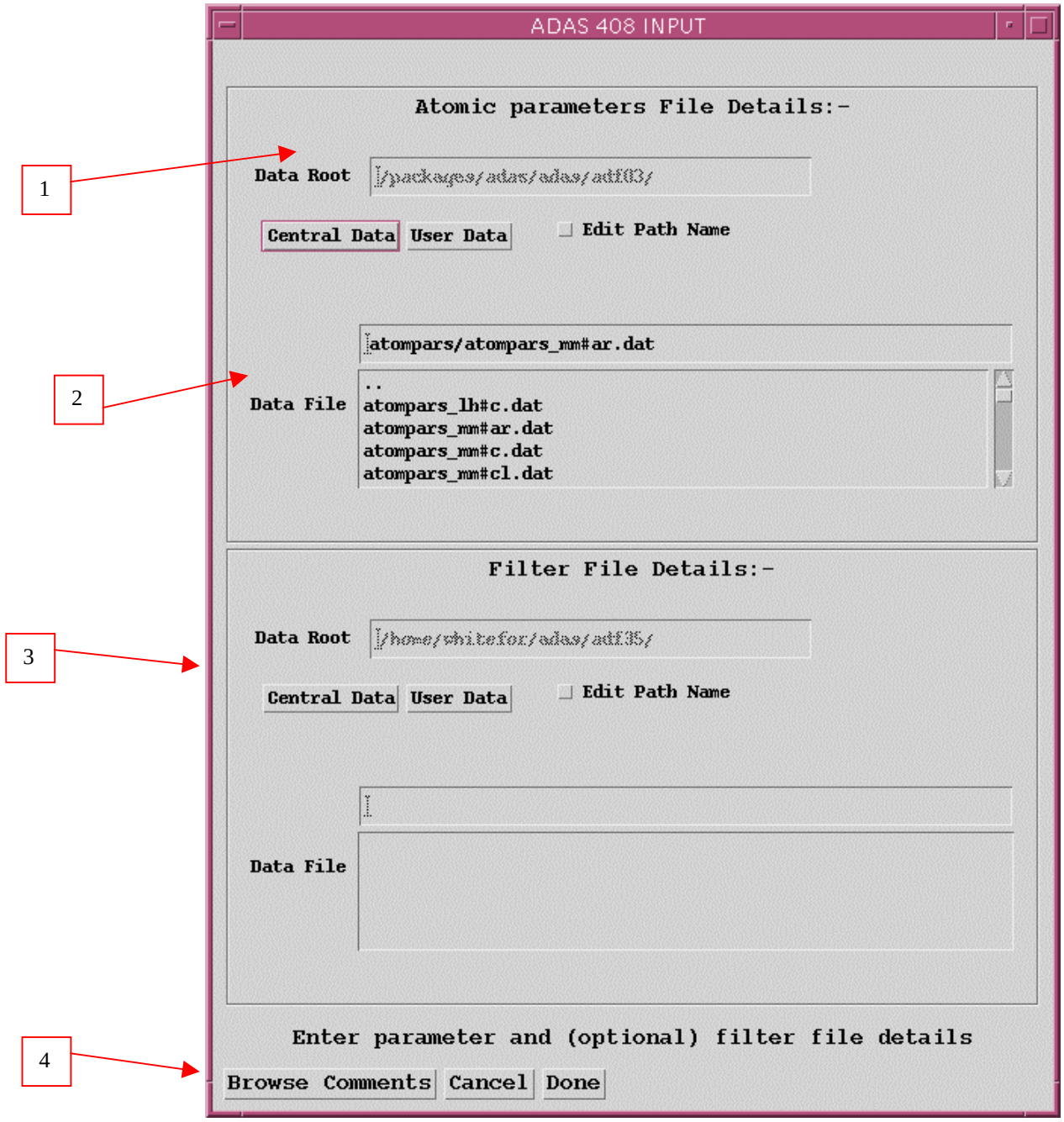
5.

ADAS408: Iso-nuclear master data - prepare from iso-nuclear parameter sets

Interactive parameter comments:

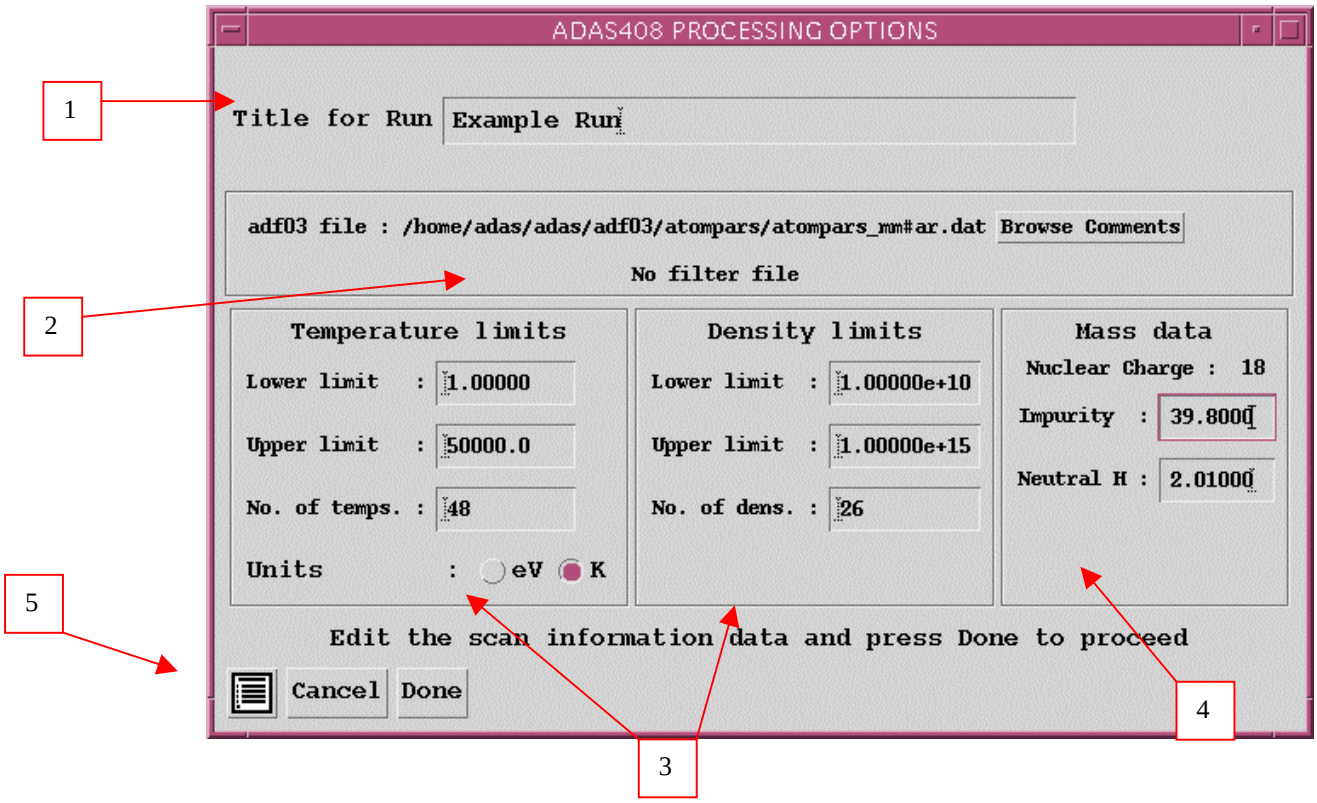
The program uses parametric forms for zero density recombination, ionisation and radiated power loss coefficients, type *adf03*, to prepare standard (unresolved, stage to stage) iso-nuclear master files for a particular element of type *adf11*. The iso-nuclear master files may be prepared over arbitrary ranges of electron temperature and electron density.

The **file selection window** is shown below:



1. Data root shows the full pathway to the appropriate data sub-directories. Click the *Central Data* button to insert the default central ADAS pathway to the correct data type. The appropriate ADAS data format for input to this program is *adf03* ('atompars files'). Click the *User Data* button to insert the pathway to your own data. Note that your data must be held in a similar file structure to central ADAS, but with your identifier replacing the first *adas*, to use this facility. The Data root can be edited directly. Click the *Edit Path Name* button first to permit editing.
2. Available sub-directories are shown in the large file display window. Click on a name to select it. The selected name appears in the smaller selection window above the file display window. Then its sub-directories in turn are displayed in the file display window. Ultimately the individual data-files are presented for selection. Data-files all have the termination *.dat*.
3. A second file may be selected which specifies a spectral filtration to be applied to the radiated power. Filter files are archived in format *adf35* and can be prepared and interrogated using the codes ADAS414 and ADAS415 respectively.
4. Once the data file is selected, the set of buttons at the bottom of the main window become active. Clicking on the *Browse Comments* button displays any information stored with the selected data-set. Clicking the *Done* button moves you forward to the next window. Clicking the *Cancel* button takes you back to the previous window

The **processing options window** has the appearance shown below



1. At the top of the window, an arbitrary title may be given for the case being processed.
2. The name of the data file under analysis and any filter file being used are shown. The button *Browse Comments* allows display of the information field section at the foot of the named atompars file, if it exists.

3. The lower sub-windows allow the plasma electron temperature and electron density for production of the output *adf11* standard master files to be specified. Select on the required temperature units. This choice determines to the units used in the adjacent temperature range selection window. Specify lower temperature limit, upper temperature limit and number of temperatures in the editable boxes. ADAS408 then creates the temperature grid equally spaced in the logarithm. Note that the output files in fact contain the temperatures in eV (see the ADAS User Manual, appxb-11). Similarly specify the electron density limits and number of grid points.
4. Enter the mass number for the actual isotope of the element required. For information the element chemical symbol is displayed. Also, the mass number of hydrogen isotope constituting the primary plasma species is required.
5. The *Exit to Menu* icon is present in ADAS408. Clicking the *Done* button causes the output options window to be displayed. Remember that *Cancel* takes you back to the previous window.

The **output options window** is of restricted form. It only offers the option of an output files. There is no output graph.

1. The *adf11* iso-nuclear master file output comprises several. The template shows the file naming structure
2. Collections of *adf11* files are held by year number and element. Enter a two digit year number for the output. Note that any two digits are acceptable and 'fictitious' years can be used for special collections if so desired. The element name is inserted automatically from the atompars input file.
3. The filter name field of the template is only sensitised if a filter file has been selected on input. The convention in the past was that in the simple cut-off case, the filter name had the prefix 'ev' followed by the numerical value of the cut-off energy in eV. In the true filter case (which was restricted to beryllium/silicon, the filter name had the prefix 'ft' followed by the first two significant figures of the beryllium and silicon thicknesses. The much greater flexibility of the full Henke filter implementation is not encompassed by the old convention. Filter names are at your own choice although central ADAS will continue to have *adf11* data following the old naming. Note also that the output files can be placed in a directory of your choice rather than entering the pass directory.
4. Click on the buttons for the output adf11 file classes you wish. The filtered power classes are only sensitized if a filter file has been selected.
5. The standard line printer text output file summarising the options selected for ADAS408 is available. The *Replace* and *Default File Name* buttons are present for the text output file as usual.
6. The *Exit to Menu* icon is present in ADAS408. Clicking the *Done* button causes the output options window to be displayed. Remember that *Cancel* takes you back to the previous window.

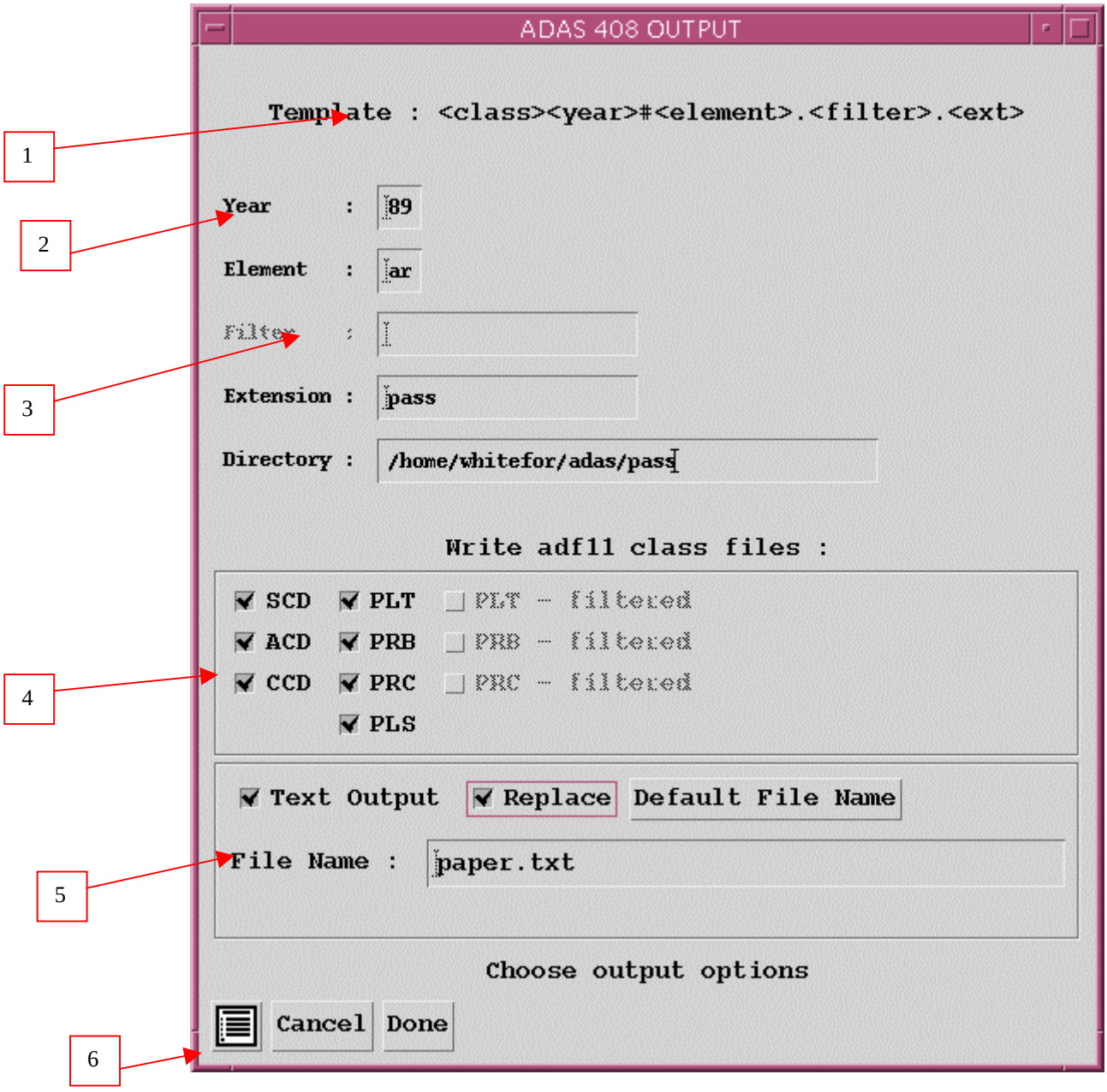


Illustration:

There is no graphical display from this code.

Table 5.8a

```
*****************    RUN SUMMARY FOR PROGRAM GENERATING STANDARD   ******************
 ****************   ISONUCLEAR MASTER FILES FROM PARAMETRIC FORMS  ******************
 ***************************** ADAS408 - DATE: 19.08.03 ******************************


INPUT PARAMETER FILE     : /home/adas/adas/adf03/atompars/atompars_mm#ar.dat


Output files:

 /home/mog/pass/acd89_ar.pass
 /home/mog/pass/scd89_ar.pass
 /home/mog/pass/ccd89_ar.pass
 /home/mog/pass/prb89_ar.fil_jet.pass
 /home/mog/pass/plt89_ar.fil_jet.pass
 /home/mog/pass/pls89_ar.pass
 /home/mog/pass/prc89_ar.fil_jet.pass
 /home/mog/pass/prb89_ar.pass
 /home/mog/pass/plt89_ar.pass
 /home/mog/pass/prc89_ar.pass

------------------                                              ------------------



IMPURITY INFORMATION:
 ---------------------
  ELEMENT SYMBOL         =      ar
  NUCLEAR CHARGE         =      18
  LOWEST ION CHARGE      =       0
  HIGHEST ION CHARGE     =      17
  ATOMIC MASS NUMBER     =   40.00


NEUTRAL DONOR INFORMATION:
 --------------------------
  ELEMENT SYMBOL         =       H
  NUCLEAR CHARGE         =       1
  ATOMIC MASS NUMBER     =    2.01


FILTER INFORMATION:
 -------------------

 Filter from : /home/adas/adas/adf35/jet_filter.dat


ELECTRON TEMPERATURE/DENSITY INFORMATION:
 ----------------------------------------

                           TEMPERATURE (EV)      DENSITY (CM-3)
                           ----------------      --------------
  NUMBER OF VALUES       =        48                  26
  MINIMUM VALUE          =    1.0000D+00            1.0000D+10
  MAXIMUM VALUE          =    5.0000D+04            1.0000D+15

  (NOTE: EQUAL INTERVALS IN THE LOGARITHM ARE SET)
```

Notes: