IRAF NEWSLETTER

December 1994 Number 13

Central Computer Services National Optical Astronomy Observatories* P. O. Box 26732 Tucson, AZ 85726

Table of Contents

System Update 1

IRAF Version 2.10.3BETA Release 2

ADASS Conference Update 3

IRAF in Japan 4

New Network Device Driver 5

X11 Support Package X11IRAF 6

Hints for Setting Up and Using XGTERM and XIMTOOL 7

IRAF Network Information Services 11

Frequently Asked Questions List Available 13

Reorganization of the Network Archive 14

Tips and Tools for Learning IRAF 15

The IRAF External Package GRASP 15

A New Package for Nebular Analysis 17

FORTRAN Program and IRAF Script Produce Fast Scientific Output 19

Using EVVEXPR to Process Vector Expressions 20

New Image Expression Evaluation Task 21

New Image Registration Task 22

Fitting a Surface with SURFIT 23

Editing and Modifying Headers with ASTHEDIT 23

* Operated by the Association of Universities for Research in Astronomy, Inc. (AURA) under cooperative agreement with the National Science Foundation

Revisions to IMCOMBINE for V2.10.2 25

An Enhancement to COSMICRAYS 26

CCDTIME—Predicting Exposure Times 26

New Radial Velocity Tasks, RVIDLINES and RVREIDLINES 27

New Conversion Tools for 1D Image Spectra 27

SFLIP—Flip Data or Dispersion Coordinates in Spectra 28

APPHOT Package Update 28

New Version of the DAOPHOT Crowded-Field Photometry Package Released 30

PHOTCAL Package Update 33

PTOOLS Package Update 34

Aperture Correction Tasks Installed in the PHOTCAL Package 34

Measuring Stellar Image Widths for Seeing and Focus Characterization 35

Image Conversions Using IMPORT/EXPORT 36

IRAF Documentation 37

Layered Software Available for IRAF Versions 2.8/2.9/2.10 38

Brief Summary of Application Software Modifications in IRAF V2.10.3BETA 40

IRAF NETWORK INFORMATION SERVICES

IRAF Site Support:	iraf@noao.edu noao::iraf or 5355::iraf
IRAF HOTline: FAX:	(602) 323-4160 (602) 325-9360 (Mid-March, area code → 520)
Network Archive: (anonymous ftp)	ftp iraf.noao.edu ftp 140.252.1.1
WWW URL:	http://iraf.noao.edu/
IRAFINFO Facility:	mail irafinfo@iraf.noao.edu index iraf
ADASS Newsgroups (information):	mail irafinfo@iraf.noao.edu get iraf newsgroups
ADASS Newsgroups (subscribe):	news@iraf.noao.edu
Archive Listserver:	mail listproc@iraf.noao.edu help

The IRAF NEWSLETTER is published by the Central Computer Services, National Optical Astronomy Observatories, P. O. Box 26732, Tucson, AZ 85726. Editors: Doug Tody, Jeannette Barnes

System Update

Why another Newsletter at this time? Our plan until recently has been to wait and come out with a Newsletter at the time of the IRAF V2.11 release. When we looked into this further however we found that if we did this it would have to be a very large newsletter to cover all the material that we wanted to include. Hence we decided to do two newsletters, one now and another in a few months when IRAF V2.11 is released.

The main focus of the current Newsletter is all the new science applications software added to IRAF in the past year or two. Most of the applications software that will be in V2.11 IRAF is available now, either in the V2.10.3BETA release for Sun platforms (including Solaris), or in the form of layered packages that can be installed into older versions of IRAF on various platforms. Getting this software out to users now gives us the opportunity to more extensively beta-test the software prior to the V2.11 release (this is why V2.10.3BETA is being called a beta release—in many respects 2.10.3 is really V2.11BETA).

The next Newsletter will come out about the time of the V2.11 release and will focus on this major new release of IRAF and all that entails. Also covered will be the new GUI facilities and the GUI applications being developed for IRAF, both at NOAO and elsewhere in the IRAF community.

Our system software activities this past year focused initially on continued development of the GUI system software. This was the major systems project in the first half of 1994. The project lead to the release of usable, fairly stable and complete versions of **ximtool** and **xgterm** in the summer of 1994. NOAO and the ICE observing environment on Kitt Peak were converted over to X11 at that time. Since that time many IRAF users have begun using these new facilities as well.

In the second half of 1994 we have had to spend less time on development of X11 support and the GUI facilities, and cut back on GUI applications development as well, in order to catch up on other things. IRAF was ported to Solaris and a new version of IRAF, V2.10.3BETA, was released.

The IRAF network information services, while a back-burner project for us, are continuing to evolve. This is getting to be quite a useful facility—check it out if you haven't already. The new FAQ, the ADASS newsgroups, and the Web support are all very useful new facilities. The FTP archives have been reorganized. In particular we would eventually like to see some of the 5000 or so email inquiries we receive each year move onto the ADASS newsgroups so that all may benefit from the discussion.

The main systems projects for the near future include IRAF ports and platform support, preparation of the V2.11 release, and further work on the GUI system facilities, the X11IRAF utilities, and the GUI applications. Porting activities will include enhancements to our Solaris support, full support for the DEC Alpha running OSF, and a major effort to support IRAF on personal computer platforms, mainly x86 boxes (including Pentium) running UNIX variants such as Linux, Solaris, and BSD. We are particularly excited about the potential of IRAF on PCs, especially the prospect of running interactive IRAF GUI applications on PC platforms.

Once this work is out of the way we plan to start work on the "Open IRAF" initiative. This includes many things but probably the most notable items will be programming-related enhancements such as enhanced support for IRAF programming in C and other languages, and a major upgrade to the IMFORT interface to support host level access to the IRAF runtime environment. We also plan to revise our applications development plan, and start looking into major revisions and upgrades to some of the older IRAF packages, as well as new software for (for example) reduction of data from the new IR instruments, improved support for Solar astronomy, enhancements to IRAF support for data acquisition and quick look, and reduction of data from the Gemini instruments.

Further information on these and many other topics will be found in the articles in this Newsletter.

Doug Tody

IRAF Version 2.10.3BETA Release

In late August, IRAF Version 2.10.3BETA was released for SunOS4 and for Sun Solaris 2.3. V2.10.3BETA is the first port of IRAF to the Solaris operating system. Solaris/IRAF was prepared for Solaris 2.3, using version 2.0.1 of the SunSoft Fortran and ANSI C compilers. The distribution is available in the *iraf/v210/SSOL-v2103-beta* directory on *iraf.noao.edu*.

For those sites that are running both SunOS and Solaris, it was decided to release V2.10.3BETA for SunOS as well as for Solaris, so that a compatible version of IRAF would be available for both platforms (see the *iraf/v210/SOS4-v2103-beta* directory on *iraf.noao.edu*). There are no plans currently to release V2.10.3BETA for any other platforms, but this could change depending on the timing of current porting activities and the V2.11 release. The next release of IRAF will be IRAF Version 2.11, and this version will be made available for all supported IRAF hosts beginning sometime in early 1995. We don't currently plan to support the DEC Alpha running OSF until V2.11, but this could change depending on when IRAF is ported to the Alpha (a preliminary port of an earlier version of V2.10 is available now from STScI).

The BETA annotation may be a bit confusing; the BETA does **not** mean that the software has not been well tested—it has! V2.10.3BETA is really just a snapshot of our V2.11 development system taken at the time of the Solaris port, including whatever happened to be in the system at the time (but not including everything planned for V2.11, obviously). Included are most of the science applications that will be in V2.11. IRAF Version 2.10.3BETA was well tested before its release, and the science application software has been in use for some time at NOAO.

A patch to V2.10.3BETA is planned for the very near future. Our goal has been to get this out in the fourth quarter of 1994 but we have been delayed waiting for Solaris 2.4 to be released to the public. Our initial Solaris/IRAF release was followed immediately by Sun shipping the V3 compilers to all our users, which broke our compiler support; we don't want the same thing to happen again, with Sun shipping Solaris 2.4 immediately after we upgrade our support! The patch will include a few minor bug fixes, plus support for Solaris 2.4 and the V3.x SunSoft compilers.

The X11 support package X11IRAF, including the X11 support utilities **xgterm**, **xtimool**, and **xtapemon**, is not part of the V2.10.3BETA distribution but can be obtained from the *pub/v2103-beta* directory in the IRAF archives; see the README file in this directory for further instructions. V2.10.3BETA users are encouraged to get the X11 support package. Its integration with V2.10.3BETA has been well tested, and users will find these tools straightforward to use, with **xgterm** and **ximtool** providing the needed replacements to the SunView tools, **gterm** and **imtool**. The X11IRAF utilities can also be used with older versions of IRAF, with minor limitations.

We do not plan to generate a complete revisions summary for the V2.10.3 release, but a short summary of the new science application software included in V2.10.3BETA is given at the end of this Newsletter, as well as in other individual articles throughout this issue.

Doug Tody

ADASS Conference Update

The *Fifth Annual Conference on Astronomical Data Analysis Software and Systems* (ADASS V) will be held in Tucson, 22-25 October 1995. The Conference will be hosted by the National Optical Astronomy Observatories. Additional sponsors include the Infrared Processing and Analysis Center, the National Aeronautics and Space Administration (tentative), the National Radio Astronomy Observatory, the National Research Council of Canada (tentative), the National Science Foundation (tentative), the Smithsonian Astrophysical Observatory, and the Space Telescope Science Institute. The ADASS Conferences provide a forum for scientists and programmers concerned with algorithms, software, and software systems employed in the reduction and analysis of astronomical data.

The Program Organizing Committee for ADASS V has the following members: Rudi Albrecht (ST-ECF/ESO), Roger Brissenden (SAO), Tim Cornwell (NRAO), Dennis Crabtree (DAO/CADC), Bob Hanisch—Chair (ST ScI), Rick Harnden (SAO), Gareth Hunt (NRAO), George Jacoby (NOAO), Barry Madore (IPAC), Dick Shaw (ST ScI), Karen Strom (U. Mass.), and Doug Tody (NOAO). The Local Organizing Committee is chaired by Jeannette Barnes (*softconf@noao.edu*).

As in previous years, the program will consist of invited and contributed talks on special topics, poster papers, software demos, and special sessions called "Birds of a Feather" (BOFs). A BOF session is usually scheduled for 1–2 hours, not concurrent with the general session but it can overlap with other BOFs. The structure of a BOF is usually at the leader's discretion. If anyone would like to sponsor a BOF, or has a suggestion for one, please send mail to *softconf@noao.edu*.

The previous Conference, ADASS IV, was held in Baltimore in late September 1994 and was hosted by ST ScI. There were over 300 registered participants for the Conference representing 17 countries. There were 38 oral presentations during the General Sessions including 11 invited talks. More than 90 poster papers and 15 software demos were presented as well. Five special sessions (BOFs) were also held. The Proceedings for ADASS IV, like those of the earlier conferences, will be published as part of the Astronomical Society of the Pacific (ASP) Conference Series. We expect this volume to be available mid-summer 1995. The ASP has also given ADASS permission to make the Proceedings available over the World Wide Wide, as was done with the Proceedings for ADASS III (held in Victoria in October 1993)—see the URL: *http://cadcwww.dao.nrc.ca/ADASS/adass_proc/adass_proc.html*.

Additional information will become available about ADASS V early next year. A preliminary announcement will be electronically mailed to the Conference mailing list at that time. Detailed registration information will be mailed by regular post in the spring of '95. If you would like have your name added to the Conference mailing list please send an email message to *softconf@noao.edu* (the current mailing list is comprised of attendees from the past four Conferences) and include your name and email/postal addresses. Interested parties can also browse the latest information via the World Wide Web using the URL:

`http://iraf.noao.edu/ADASS/adass.html`.

Materials for the Conference will also be made available by anonymous FTP in the directory *iraf/conf/adass-95* on the node *iraf.noao.edu*.

Jeannette Barnes
George Jacoby
Doug Tody

IRAF in Japan

IRAF is one of the most popular astronomical data reduction packages in Japan; others are AIPS and IDL. Currently IRAF is installed at more than 40 sites in Japan. IRAF is used by many visiting astronomers at the Astronomical Data Analysis Center (ADAC) at the National Astronomical Observatory of JAPAN (NAOJ), the Okayama Astrophysical Observatory (OAO), and the Kiso Observatory. The other installations average about 5 users per site. IRAF is used mainly for optical / IR and X-ray astronomy. The most common platform for IRAF is the Sun workstation and other Sun-compatible machines; about 80% of the IRAF hosts are Sun's. Many astronomers, including amateurs, are looking forward to the PC/IRAF. The percentage of Sun's used for IRAF may change when PC/IRAF is released.

NAOJ ADAC has the role of a distribution center for IRAF in Japan. This allows for a quick distribution of a new version of IRAF or addon package locally and saves work for the NOAO IRAF group as well. IRAF, addon packages, documents, and other related files are transferred by FTP from NOAO. Several directories are mirrored, but binary files (IRAF and some addon packages) for platforms other than Sun's are not transferred because of limited disk space. IRAF and other related files are distributed by FTP to about a half of the sites in Japan and by tapes to the others. Almost all of the sites are connected to the Internet, but the bandwidth is not always satisfactory for FTP transfers. We must send tapes to the sites with the poor Internet connections. Of course, we strongly recommend that all the site managers send the IRAF registration form to NOAO!

IRAF was introduced into Japan and installed on several workstations in 1990. At that time we had a serious problem: we did not have a good data reduction package available on our computers. IRAF was the best solution. But all Japanese astronomers are not fluent in English. Both site managers and users needed information about IRAF written in Japanese for efficient use of the software. Thus we organized the IRAF managers group and began to exchange information in 1990. The IRAF managers group was then changed into the Japan Association for Information Processing in Astronomy (JAIPA), and the range of its activities was widened. JAIPA has close relations with NAOJ ADAC; the office of JAIPA is located there. JAIPA has various responsibilities. For example, the PAONET project aims to establish the network connecting institutes with publicly-opened astronomical observatories and schools through public telephone lines to transfer astronomical images, and the ARCHIVE project group is developing an archival system for the data of OAO and the Kiso Observatory. JAIPA has grown to a group of astronomers (now the number of members is about 60) with broad interests, but the offer of information about IRAF is still its most important role.

JAIPA published the "IRAF Cookbook" 1st. and 2nd. editions, which were written in Japanese. The 2nd. edition has 500 pages, and the contents of it are the following: an install/management manual, an IRAF beginner's guide, manuals for the data reduction of several instruments in Japan, manuals for various tasks/packages in IRAF, manuals for our original software, guides to software development under the various IRAF environments, guides to presentation, trouble shooting in IRAF, and other manuals (Lick Mongo, FITS, etc.). We believe this cookbook is useful to almost all the observational astronomers in Japan. The 3rd. edition of the Japanese "IRAF Cookbook" will appear next spring (1995).

Shin-ichi Ichikawa
Astronomical Data Analysis Center
National Astronomical Observatory
of JAPAN

New Network Device Driver

A new network device driver has been installed in IRAF V2.10.3BETA which supports a variety of forms of interprocess communication. This driver is currently used primarily for image display but can be used for other client-server applications requiring a runtime network connection (see the comments in the files *fio$ndopen.x* and *os$zfiond.c* for more information).

In image display applications the new driver is used by an IRAF task to connect to the image display server through any one of several transport methods. The initial set of supported communications methods are Internet sockets, Unix domain sockets, and FIFO pipes. Unless overridden a connection is first attempted using Unix domain sockets; if this fails the familiar old */dev* FIFO pipes are used. The domain socket name is formed using the user's userid so each user gets his/her own private network socket and conflicts between users doing image display on the same host are automatically avoided. For accessing remote display servers TCP/IP sockets may be used to connect to any host on the Internet. (Don't confuse this with the X11 display connection, which occurs at a different level).

To override this connection scheme, e.g., to use private FIFO pipes or Internet sockets, users can define a host level environment variable 'IMTDEV' that specifies the domain and address of the desired connection. The form of the address depends upon the domain, as illustrated in the examples below:

inet:5187	Server connection to TCP/IP port 5187 on the local host. For a client, a connection to the given port on the local host.
inet:5187:foo.bar.edu	Client connection to TCP/IP port 5187 on Internet host foo.bar.edu. The numeric form of address may also be used.
unix:/tmp/.IMT%d	Unix domain socket with the given pathname, local host only. Any "%d" fields, if present, are replaced by the user's UID number.
fifo:/dev/imt1i:/dev/imt1o	FIFO or named pipes with the given pathnames. IPC method, local host only. Two pathnames are required, one for input and one for output, since FIFOs are not bidirectional. For a client the first FIFO listed will be the client's input FIFO; for a server the first FIFO will be the server's output FIFO. This allows the same address to be used for both the client and the server, as for the other domains.

Note that by defining IMTDEV to name a personal set of FIFO pipes it is no longer necessary for users to have personal graphcap files, however, personal graphcap files are another way to override the default connection scheme. In V2.10.3BETA the 'DD' strings of the *stdimage* devices were changed so the "/dev/imt1" pipe was removed leaving an entry that looks like the following:

```
imt1|imt512|imtool|Imtool display server:\
:cn#1:LC:BS@:z0#1:zr#200:DD=node!imtool,,512,512:tc=iism70:
```

Users who still have personal graphcap files will still be able to use them since specifying a pipe will use that as the connection, alternatively, using the domain address syntax will also define a connection type. For example the DD strings

```
:DD=node!imtool,inet\:5187,512,512:tc=iism70:
:DD=node!imtool,inet\:5187\:foo.bar.edu,512,512:tc=iism70:
:DD=node!imtool,unix\:/tmp/.IMT%d,512,512:tc=iism70:
```

define the same connection request as the IMTDEV definition above. Note that the ':' must be

escaped with a backslash to avoid terminating the graphcap entry prematurely. In this way users can specify a *stdimage* device that is a display server running on some other node, or one that uses a different type of input (as when there are two **ximtool** / **SAOimage** windows running simultaneously).

The accompanying article on *Hints...* has other examples of how the display connections can be established. Users with questions about how to set up a particular configuration should feel free to contact IRAF site support.

Doug Tody
Mike Fitzpatrick

X11 Support Package X11IRAF

The IRAF X11 support package, X11IRAF, contains the IRAF GUI development environment plus several client applications for use with IRAF running under the X window system. The client applications include **xgterm** (a graphics terminal emulator based on **xterm**), **ximtool** (an image display server), and **xtapemon** (a little program for monitoring magtape i/o). This package has been available for users ''as-is'' during its development period for the SunOS platform this past year.

xgterm provides an interactive graphics facility similar to the old SunView program **gterm**, with the addition of color graphics and the full window crosshair cursor that many users missed when they switched to X and **xterm**. But the full capabilities of **xgterm** are in its use as a ''widget server'' for GUI (graphical user interface) applications such as those being developed for IRAF. This new capability will be explored further in the next issue of the IRAF Newsletter.

ximtool has many of the features of the old **imtool** program (including the 4 frame buffers) but it also has many enhancements including pan and zoom windows and a control panel. The **xtapemon** program provides a mechanism for tape users to monitor what the tape device is doing during reading or taping. Documentation for these programs is available with the package distribution (see below), and users are encouraged to read these documents to explore the full capabilities of these new tools.

ximtool can be used with any version of IRAF. Simply replace the display server that you are currently using in X with **ximtool**. (See accompanying *Hints...* article for further details.) **xgterm** can be used with older versions of IRAF as well as with non-IRAF graphics programs (such as Mongo or SuperMongo) in xterm emulation mode—see the **xgterm** documentation for details. V2.10.3BETA users should remember to type `stty xgterm` at the CL prompt, or better yet, put this command into the login.cl or loginuser.cl file.

A new version of the X11IRAF package is being released as this Newsletter goes to press. This latest release is primarily a bug-fix release. Anyone running earlier versions of these programs will want to update to this latest version. Look in the *pub/v2103-beta* directory on *iraf.noao.edu*. Precompiled binaries taken from our SunOS and Solaris development platforms are provided. Support for platforms other than Sun (our development system) will follow when IRAF V2.11 is released, however, anyone can get the sources now and build these on their favorite platform. This software is still under development and additional updates are planned.

Doug Tody

Hints for Setting Up and Using XGTERM and XIMTOOL

Since work on the **xgterm** and **ximtool** utilities (as well as **xtapemon** and the GUI facilities, in general) is still actively in progress complete user documentation for these programs is not yet available. Information on basic usage is available however in the archive with the binaries (see the ".info" files in the *pub/v2103-beta* directory), some examples are given in the FAQ, and questions are always answered by contacting site support. What follows is an overview of each of the tasks and examples of how they are commonly used; feel free to contact us if you have any questions not answered here.

In what follows below there are many examples showing the use of X resources to override the default resources of the X11IRAF utilities. Users need to know about these things in order to customize the software and workaround configuration problems, but some caution is advised because 1) this software is still under development and things such as resource names and defaults are likely to change in future releases, and 2) it is possible to make any X program fail to function correctly if one is not careful when overriding resource values. If an X11IRAF program fails to function correctly after an upgrade it may be wise to comment out the relevant resources in your .Xdefaults file to make sure they are not the source of the problem.

xgterm

The **xgterm** utility is essentially the well-known **xterm** with an improved graphics capability; it can be used as a replacement for **xterm** whether IRAF is being run or not. It also serves as the system software for the new IRAF GUI applications and is required to run them. Currently binaries are only provided for Sun systems but the source code is available for those willing to port it to other platforms (something we will be doing as IRAF is upgraded to V2.11 on those systems in the next year). Some features of **xgterm** such as color graphics are only available when the program is used with V2.10.3BETA and later versions of IRAF, but it may also be used as the graphics terminal for earlier releases of IRAF where users will still benefit from the full screen cursor, interactive dialog capability (status line), and type-ahead capability.

Users on monochrome workstations may find that the graphics cursor is not visible because the default cursor color is indistinguishable from the background color on a monochrome display. This can be fixed by changing the "crosshairCursorColor" resource either by putting the line

```
XGterm*gterm.crosshairCursorColor:      white
```

in the .Xdefaults file and restarting the window system (or resource manager), or by setting it on the command line using

```
% xgterm -xrm "*crosshairCursorColor:white"
```

Another common question is how to set the graphics window geometry to be a specific size (e.g., customized for spectroscopic applications). As with **gterm** the graphics window geometry can be specified with the "-G" or "%geom" flag on the command line, for instance,

```
% xgterm -G 1100x350+10-10  # or alternatively
% xgterm %1100x350+10-10
```

will produce a wide rectangular graphics window in the lower portion of the screen suitable for examining spectra. The geometry may also be specified in the user's .Xdefaults file by specifying

```
XGterm*tekGeometry:         1100x350+10-10
```

Lastly, users often wish to change the background or text colors used for graphics plots (under V2.10.3BETA) from their defaults. The background and text colors are set using a combination of X resources to set the predefined colors, and an IRAF environment variable to say which colors are used for text, background, axis labels, etc. The color resources are named 'color0' through 'color9'; colors zero and one are the background and foreground colors, respectively,

and default to white on black. These resources and their builtin default values are as follows:

```
Xgterm*gterm*color0:    black
Xgterm*gterm*color1:    white
Xgterm*gterm*color2:    red
Xgterm*gterm*color3:    green
Xgterm*gterm*color4:    blue
Xgterm*gterm*color5:    cyan
Xgterm*gterm*color6:    yellow
Xgterm*gterm*color7:    magenta
Xgterm*gterm*color8:    purple
Xgterm*gterm*color9:    darkslategray
```

On some older workstation monitors darkslategray (color9) may appear too dark and a color such as slategray may be preferable.

The environment variable *glbcolor* defines the mapping of box-plot attributes onto the predefined graphics colors. It is predefined in V2.10.3BETA as

```
set glbcolor = "pt=3,fr=9,al=3,tl=6,ax=5,tk=5"
```

where the attribute codes are as defined in the table below.

pt	-	plot title
fr	-	viewport frame (the background around the plotting area)
al	-	axis labels
tl	-	tick label
ax	-	axis
tk	-	tick
gr	-	grid between tick marks

The numbers in *glbcolor* refer to the color numbers mentioned earlier, for example, "tk=5" indicates that the tick marks are drawn using color number 5 (cyan). To make use of **xgterm** color graphics you should have the following set in your login.cl or loginuser.cl file:

```
stty xgterm
```

Color graphics is enabled only if you are using V2.10.3BETA or a later version of IRAF, but using X resources you can still change the background and foreground colors when working with an earlier version of IRAF. Resources for the text window are the same as for **xterm**.

ximtool

With the new network device driver in V2.10.3BETA (see accompanying article) **ximtool** and IRAF support three different types of communication for image display. **ximtool** itself will listen on and accept a connection from each of these inputs (**ximtool** can also support simultaneous input from multiple clients). Users who were just getting comfortable with private FIFO pipes and graphcap files may be confused by the new protocol so we present some common scenarios and recipes for dealing with the various display problems users are likely to encounter.

1) Basic usage:

For users running **ximtool** and IRAF on a personal workstation with no other users, there is *no* setup required at all other than to start **ximtool**. Older versions of IRAF will display using the FIFO pipes in */dev*, while in V2.10.3BETA and later versions of IRAF, Unix domain sockets will be tried first, by default. Since **ximtool** listens for incoming connections on all input channels, no special flags or configuration is required to display an image in either case.

2) X-terminal users and multiple users on a single host:

With V2.10.3 systems, X-terminal users (and others) on multi-user servers no longer need to set up private FIFOs since the Unix domain sockets guarantee there will be no conflicts between users (provided there is only one **ximtool** per user). It is recommended however that the FIFO

pipes be disabled in **ximtool** if **SAOimage** is being run on the same host, to avoid any possible conflict with **SAOimage** users not using private FIFOs. The FIFO pipes are disabled by setting the "input_fifo" **ximtool** resource to "none"; TCP/IP sockets are disabled by setting the "port" resource to zero. This can be done on the command line using, e.g.,

```
% ximtool -xrm "*input_fifo:none" -xrm "*port:0"
```

This command can easily be aliased, but the X resources can also be permanently set by adding the following lines to the .Xdefaults file:

```
XImtool*port:         0
XImtool*input_fifo:   none
```

These values will take effect the next time the window system is started or can be loaded immediately using the command:

```
% xrdb -load ~/.Xdefaults
```

ximtool can then be started without any command line arguments and only the Unix domain socket will be used. These sockets are created automatically and will be unique for each user to avoid conflicts.

For V2.10.2 and earlier versions of IRAF, only FIFO pipes are available for communication so private FIFO pipes and graphcap files are still required in these cases. The pipes are created as before but **ximtool** must be started to use these pipes instead of the */dev* defaults. For example,

```
% ximtool -xrm "*input_fifo:/<path>/imt1i" \
         -xrm "*output_fifo:/<path>/imt1o"
```

or in the .Xdefaults file specify:

```
XImtool*input_fifo:   /<path>/imt1i
XImtool*output_fifo:  /<path>/imt1o
```

Note that the "input_fifo" takes the "imt1i" pipe and vice versa; this is opposite of the (confusing) pipe naming convention used by **SAOimage**.

3) Users running **ximtool** on a remote machine who are logged into IRAF on their home machines (or vice versa):

The traditional approach to this is to set the *node* environment variable to, e.g., "foobar", and rely on IRAF networking to display the images using the FIFO pipes. This still works fine but what happens if you are trying to display to a machine at a remote site that isn't in your local IRAF network? In this case you would need to use the TCP/IP socket to connect to the **ximtool** running on the remote machine. Before logging in to IRAF (on your home machine) you would first set the Unix environment variable IMTDEV to specify the connection type. For example,

```
% setenv IMTDEV       inet:5137:foo.bar.edu
```

This says to use an 'inet' connection on port '5137' to node 'foo.bar.edu.' The IMTDEV definition overrides the normal IRAF connection scheme so there is no need to reset or unset your *node* environment variable once logged in. On the remote machine (assuming there are local IRAF users) you would disable the FIFO pipes and the Unix domain sockets using something like:

```
% ximtool -xrm "*input_fifo:none" -xrm "*unixaddr:none"
```

Note that the Unix domain sockets should be unique so disabling them isn't strictly necessary.

The typical use for this feature is when you are running **ximtool** on the local workstation with IRAF executing on a remote machine somewhere on the Internet, and IRAF networking is not available between the two hosts. IMTDEV allows the remote IRAF session to display directly to your local **ximtool**. Another way to solve this problem would be to run **ximtool** remotely,

setting the Unix environment variable DISPLAY to cause XLIB to display to the X server on your local workstation. It is much better to use IMTDEV however, as since **ximtool** executes locally, once the image frame buffers have been loaded over the network everything else executes locally, and **ximtool** functions such as zoom and pan or frame blink will execute much faster than with a remote X display.

4) Users who want more than one **ximtool**:

Since **ximtool** has multiple frame buffers users should consider whether they really need two **ximtool**s in the first place. Usually it is preferable to load two or more frames and blink or tile the frames to view the different images.

If there is a need to run multiple **ximtool**s then the easiest way to do this is to control them from different IRAF sessions so that different socket numbers can be assigned to each session; each **ximtool** would be started with a different socket number as well. For example, in the first **xgterm** window you would define the socket and start **ximtool** using

```
% setenv IMTDEV       unix:/u2/smith/iraf/dev/IMT1%d
% ximtool -xrm "*input_fifo:none" -xrm "*port:0" \
        -xrm "*unixaddr:/u2/smith/iraf/dev/IMT1%d" &
```

In a second **xgterm** window you change the socket number slightly and start things using

```
% setenv IMTDEV       unix:/u2/smith/iraf/dev/IMT2%d
% ximtool -xrm "*input_fifo:none" -xrm "*port:0" \
        -xrm "*unixaddr:/u2/smith/iraf/dev/IMT2%d" &
```

Each IRAF session will write to a different socket, and each **ximtool** will be listening on a different socket. It should be noted that something similar can also be done using either FIFOs or TCP/IP sockets for the connection, similarly entries can be defined in a graphcap file naming each socket but it is cumbersome to switch the setting of *stdimage* between image display calls.

xtapemon

xtapemon is a magtape status monitor and display utility. You use it to display the status of an IRAF tape job while it is running. TCP/IP sockets are used for communication, so the IRAF tape job and **xtapemon** may be on the same host machine or on different hosts. All V2.10 versions of IRAF support tape status output (including even VMS/IRAF).

In the simplest use the **xtapemon** task is started in the background prior to using the tape. The *tapecap* IRAF environment variable is set to enable status output, for example,

```
cl> set tapecap = ":so"
cl> allocate mta
cl> rfits mta 1-999 images
```

At this point the **xtapemon** window will display the status of the tape as the images are being read. If **xtapemon** is being run on a client machine and the IRAF session is started on a server, the tapecap is set to send output through a TCP/IP connection to that node using the syntax

```
cl> set tapecap = ":so=<node>"
```

where <node> is the name of the local machine. On the other hand, if both IRAF and **xtapemon** are being run locally, but the tape is on a remote node, status may be displayed using

```
cl> rfits "ursa!mta[:so=pisces]" 1-999 images
```

where "ursa" is the remote machine with the drive and "pisces" is the local machine. The quotes are necessary in order to pass the full name to the task.

Mike Fitzpatrick
Doug Tody

IRAF Network Information Services

In addition to the FTP archive and site support services which most people using IRAF will already be familiar with, a number of new network facilities are available to help users find the IRAF information they need more easily. These services are constantly being expanded and updated as the network information system evolves and as new IRAF software is developed. Users are encouraged to try out the new services and to contact us with comments or suggestions.

Anonymous FTP

This is the primary distribution mechanism for IRAF software, documentation, and other file-oriented information. Users should FTP to the given address and login as `anonymous`, using your email address as the password. The FTP archive is located at:

```
iraf.noao.edu                (140.252.1.1)
```

Directories of special interest in the archive include:

```
/iraf/v210       - IRAF V2.10 distribution (all platforms)
/iraf/docs       - documentation
/iraf/extern     - external packages
/iraf/misc       - miscellaneous software
/contrib         - contributed software
```

Any WWW browser (see below) can also be used to browse the FTP archives and view or download files.

IRAF Site Support

To contact us directly with questions or comments you can use email, phone, or FAX, at any of the addresses listed below.† Alternatively the **gripes** command in the CL can be used to send email to IRAF.

```
Email   iraf@noao.edu  or  5355::iraf
Hotline (602) 323 - 4160
NOAO    (602) 327 - 5511
FAX     (602) 325 - 9360
```

General IRAF questions or comments can also be sent to one of the ADASS newsgroups (see below). This has the benefit (compared to private email) that others can benefit from the exchange and may pick up a few pointers in the process.

World Wide Web

Access to the FTP archive, documentation, FAQ, ADASS newsgroups, and more is available through the WWW Home Page for IRAF at the URL:

```
http://iraf.noao.edu/
```

Frequently-Asked-Questions List

Many questions we are asked are about things that affect most sites, which someone else has already run into. The most common of these have been collected into a Frequently-Asked-Questions list that may solve your problem before you need to contact us, or else offer tips and tricks to using IRAF more effectively. This list is available either as a plain-text document or on the Web as:

---

†The area code for the Tucson area will change to **520** in mid-March 1995.

```
ftp://iraf.noao.edu/iraf/docs/FAQ
http://iraf.noao.edu/faq/FAQ.html
```

The FAQ and various other documents can also be retrieved via email using the *irafinfo* service described in the next section.

The IRAFINFO Facility

General information about the IRAF software is available via email using the *irafinfo* facility. To get started send the following message to the *irafinfo* server:

```
% mail irafinfo@iraf.noao.edu
get iraf info
<EOF>
```

This will cause introductory information, including a list of the files available via the *irafinfo* service, to be returned via email. For more information on the *irafinfo* list server itself send the command "help" to the irafinfo list server. Multiple commands may be submitted on successive lines of your message.

ADASS Newsgroups

The ADASS newsgroups are an alternate news hierarchy for the discussion of astronomical software by the astronomical and space sciences communities. Sites wishing to carry these newsgroups locally may request a direct feed of these groups by writing to *news@iraf.noao.edu*. Individuals who do not have local access to the newsgroups may read or post to the ADASS newsgroups by using any NNTP-based news reader (which includes most news readers) to connect directly to the main IRAF network server. For example using **xrn** one could enter the following command:

```
% xrn -nntpServer iraf.noao.edu -newsrcFile .newsrc-adass
```

Reading and posting messages may also be done via email by subscribing to a newsgroup using the list server (to receive news articles in the mail) or by sending email (to post new articles). The email address for each ADASS newsgroup is formed by replacing the "." characters in the newsgroup name with "-" and appending "@iraf.noao.edu". For example, the email address of the adass.iraf.programming newsgroup is

```
adass-iraf-programming@iraf.noao.edu
```

Information on reading, posting, or subscribing to the ADASS newsgroups may be obtained by sending email to *irafinfo@iraf.noao.edu* with "get iraf newsgroups" as the body of the message, e.g.,

```
% mail irafinfo@iraf.noao.edu
get iraf newsgroups
<EOF>
```

Developers, site-managers, and users will find these newsgroups a valuable source of information. All users are encouraged to post their questions and comments to these newsgroups on issues ranging from applications usage to programming and software development, so that the entire IRAF community may benefit from the discussion.

A list of the current ADASS newsgroups follows.

adass.test	Test postings.
adass.admin	Administration of the adass news hierarchy.
adass.general	Items of general interest.
adass.misc	General discussion of astronomical software.
adass.conference	Software conference announcements.
adass.iraf.readme	General information about the IRAF news facility.
adass.iraf.announce	Announcements of IRAF software, documentation, etc.

adass.iraf.buglog	Project bug notices (this is not a discussion group)
adass.iraf.applications	Discussion of IRAF applications software.
adass.iraf.programming	Discussion of IRAF programming.
adass.iraf.sources	Archive newsgroup for small programs, scripts, etc.
adass.iraf.system	IRAF system issues and system administration.
adass.iraf.misc	Miscellaneous discussion of IRAF software.

The newsgroups adass.iraf.readme, adass.iraf.announce, and adass.iraf.buglog are not discussion newsgroups; please post queries and followup discussions to the discussion newsgroups (applications, programming, system, etc.) rather than to these newsgroups. In particular, don't post reports of possible bugs to the buglog newsgroup. This newsgroup is intended only to log bugs which have already been checked out and verified by the programmer responsible for the software in question.

Archive Listserver

The list server address is *listproc@iraf.noao.edu*. To get information about the list server send an email request as shown below:

```
% mail listproc@iraf.noao.edu
help
<EOF>
```

(You can omit the Subject: line). This will cause a list of all the commands recognized by the list server to be returned via email.

To subscribe to a newsgroup send the following request to the listserver:

```
subscribe <newsgroup> <your name and affiliation>
```

e.g.,

```
% mail listproc@iraf.noao.edu
subscribe adass-iraf-announce "Joe Smith, Boston University"
subscribe adass-iraf-buglog "Joe Smith, Boston University"
```

Note that the "dashed" form of each newsgroup (list name) is used for the listserver, rather than the "dotted" form, since this is a mail-based facility. Multiple requests can be included on successive lines, for example to subscribe to multiple newsgroups. To unsubscribe from a newsgroup send the "unsubscribe <newsgroup>" request.

Doug Tody
Mike Fitzpatrick

Frequently Asked Questions List Available

A list of the most Frequently-Asked-Questions about IRAF has been compiled and is available via FTP from the IRAF network archive, via email using the new *irafinfo* service (send mail to *irafinfo@iraf.noao.edu* with "send iraf FAQ" as the message body), or through our WWW page as

```
ftp://iraf.noao.edu/iraf/docs/FAQ
http://iraf.noao.edu/faq/FAQ.html
```

The FAQ contains over 150 questions covering general information, system and application problems, documentation, and windowing system issues. Aside from possibly answering a particular question you may have, the FAQ may also offer tips on using IRAF and serve as a starting point for learning the system and what is available.

Here is what some of our users have had to say about the new IRAF FAQ:

"The perfect gift for Christmas, readers young and old will LOVE it."
- K. Kringle

"Couldn't put it down, makes me want to write one of my own!!!"
- Bill Gates

"What an experience, changed my whole political viewpoint!!"
- Newt (Scooter) Gingrich

"I wish *my* book were that good, ... better than Genesis."
- Pope John Paul II

"Two BIG thumbs up, can't wait for the Schwarzenegger movie"
- Siskel and Ebert

This file is updated periodically as new questions are added or as information changes. Site managers will find it especially useful in helping their local users.

Mike Fitzpatrick
Suzanne Jacoby
Jeannette Barnes

Reorganization of the Network Archive

The IRAF network archive has undergone minor reorganization. The *iraf.old* directory, long used to distribute a miscellaneous collection of files including external packages and their installation instructions, special files created for a few users on request, etc., has finally been retired. The contents of the directory have been frozen and archived in *iraf.old/archive*. Except where they pertain to old versions of IRAF which are also frozen, any files found in this directory are liable to be old and out-of-date.

The function of *iraf.old* has been taken over by the new directories *iraf/extern*, used to distribute NOAO/IRAF external packages, and *iraf/misc*, used to distribute miscellaneous IRAF software not part of the general IRAF release. The *iraf/contrib* directory, which is the same as */contrib*, contains additional IRAF-related software contributed by projects or individuals outside the NOAO/IRAF group (this software is not supported by NOAO/IRAF). This directory is world writable and developers are invited to put their software, or at least a pointer to where it can be found, in this directory. When a new package is released or updated it may also be a good idea to post a message to the newsgroup adass.iraf.announce to let others know about the new software.

In summary, the subdirectories of most interest to users in the *iraf* directory are now:

conf	ADASS Software Conference information
contrib	Contributed software (same as */contrib*)
docs	IRAF user and technical documentation
extern	IRAF external or layered packages
misc	Miscellaneous IRAF software or other distributions
v210	The IRAF V2.10 distributions for various platforms

Doug Tody
Jeannette Barnes

Tips and Tools for Learning IRAF

One of the most frequently asked questions from our new users is how can I learn to use IRAF? The IRAF system has grown over the years, and it can be quite an intimidating experience for a novice user. We have a couple of suggestions for beginning users that may help them learn the basics of IRAF quickly, that will then lead to a more enjoyable experience as the full capabilities of the system are explored.

If a new user has just returned from an observing run at KPNO or CTIO, for example, and has a definite goal in mind, e.g., to reduce some echelle data, then the best thing to do is grab the appropriate cookbook(s) from the network archive (see the *iraf/docs* directory on the node *iraf.noao.edu*), sit down with a small set of data, and work slowly through the reduction steps, understanding each task and procedure as one goes along. Most of the cookbooks are straightforward to follow, and usually start with the basics, or recommend documents that do.

But perhaps a new user does not have a particular data set to work with, but just wants to learn about the system. There are two things this user can do. There is a draft document *A Beginner's Guide to Using IRAF (IRAF Version 2.10*) in */pub/beguide.ps.Z* on *iraf.noao.edu*. Reading this guide and working through the examples is another way to learn the basics of the core IRAF system.

Some IRAF exercises have also been developed that may provide the novice user with yet another way to learn IRAF. These are hands-on exercises using real data provided by our KPNO staff and cover general IRAF procedures, CCD reductions, photometry, spectral extractions and wavelength calibration, and data input / output. The exercises have been packaged into a tar file; the tar file contains the data, the exercises themselves, and charts for the stars to be identified for photometric calibration and the line identifications for the wavelength calibrations. The exercises were developed for IRAF Version 2.10.2 and assumed the user was using an **xterm** window and **SAOimage** (v1.07). The exercises have also been updated for IRAF Version 2.10.3BETA using the new **xgterm** and **ximtool** facilities. These exercises may be useful both for the scientist and for students in the classroom setting. Both versions of the exercises are available in the *iraf/misc* directory on *iraf.noao.edu*; see the readme.exer2102 and readme.exer2103 files for installation instructions.

Jeannette Barnes

The IRAF External Package GRASP

The Global Oscillation Network Group (GONG) is using IRAF as the software environment to support its reduction software. This software is found in the external package called GRASP (GONG Reduction and Analysis Software Package). The GONG network (with sites in the Canary Islands, Spain; Udaipur, India; Learmonth, Australia; Mauna Loa, Hawaii; Big Bear, California; and CTIO) is scheduled to begin observations in the early spring of 1995, and will provide to the Data Management and Analysis Center (DMAC) in Tucson at least one image of the sun every minute for a minimum of 3 years. Each site will be sending approximately 1.2 GB of data per week (one Exabyte tape) to the DMAC group. The GRASP package will be used to reduce this data with the major requirement of keeping cadence with the daily input.

The subpackages and tasks of GRASP are designed to support general (i.e., not only GONG) Helioseismic image data reduction and analysis. For the purposes of GONG, the tasks may be divided into three broad categories:

- Daily Upstream Tasks are those programs which input the raw GONG data (3-phase intensity full disk images, 256 x 256 x 3) and output calibrated velocity, average intensity, and modulation images (256 x 256 each).

The stages through which the data flow are analogous to bias, dark and flat-field corrections applied to night-time imaging CCD data but the velocity calibration is a complicated process.

In addition, the elliptical geometry of the disk image (center, major/minor radii, ellipse angle) and the modulation transfer function (MTF) are determined for each image.

- Daily Downstream Tasks are those programs which take the time series of calibrated velocity images, and produce a time series of spherical harmonic coefficients for ($0 \le L \le 250$).

  The stages of this processing are to remap the full disk velocity images into the Heliographic coordinate system, and perform the spherical harmonic transformation (SHT). The output is 251 3-dimensional time series images: one image for each L-value, with the columns being time, the rows being azimuthal order m ($0 \le m \le L$), and the bands being the real and imaginary parts of SHT.

  In addition power spectra of these time series will be generated for each site day of data. The spectra are examined to assure the data quality once the stamp of approval has been given; the daily spectra are discarded.

  The daily time series from the six sites are then merged using the image MTF's as a weighting function.

- Monthly Downstream Tasks are those programs which will take the daily time and concatenate them into monthly data sets, and produce power spectra.

  The monthly data sets will be 36 days long on 18 day centers for a monthly data set size of 13.6 GB. The power spectra will be generated using a non-power-of-2 FFT producing 251 3-dimensional images: one image for each L-value, with the columns being frequency, the rows being azimuthal order m ($-L \le m \le L$), and the bands being the power and phase. This data set will also be 13.6 GB.

  The final task performed on the monthly data sets is that of peak finding. This task is to automatically go through the 251 power spectra and fit the 10**7 line profiles to produce line frequencies, widths and amplitudes.

Over the course of the first three years of network operation, the DMAC will be archiving approximately 15 TB on Exabyte tape (like many astronomical research projects, data reduction is really data explosion) with the calibrated images, daily and monthly time series, monthly power spectra and mode-frequency tables (i.e., the peak fitting results) being the major archived data products. With the exception of the peak finding routine, the majority of the GRASP applications are I/O bound, and as such the applications have been written to be memory hogs (hold as many images in memory as possible without excessive paging/swapping) so that disk reads and writes will be at a minimum.

While much of the GRASP software was written by the GONG programming group, there has been significant software contributions by the GONG community. As a result, GRASP comprises some 43000 lines of SPP, 28000 lines of Fortran and 4000 lines of C code, demonstrating the ease of including non-IRAF applications into an IRAF applications package. GRASP is currently exported to about two dozen sites around the world, and will also be installed at the data collection sites for diagnostic use by the local operators.

The Global Oscillation Network Group (GONG) is a community-based project funded principally by the National Science Foundation and administered by the National Solar Observatory. Further information may be obtained by sending email to *grasp@noao.edu*.

Ed Anderson
GONG Programmer

A New Package for Nebular Analysis

A new package is available in the STSDAS external package, called **nebular**. Its various tasks can derive the physical conditions in a low-density (nebular) gas given appropriate diagnostic emission line ratios; and line emissivities and ionic abundances given appropriate emission line fluxes, and the electron temperature (T_e) and density (N_e). The tasks in this package are based on the 5-level atom program developed by De Robertis, Dufour & Hunt (Jou. Roy. Astron. Soc. Canada, 81, No. 6, 195, 1987). These tasks extend the functionality of the original FIVEL program to provide diagnostics from a greater set of emission lines, most particularly those in the vacuum ultraviolet that are now available from the IUE and HST archives. A brief summary of the tasks is given in the table below; details may be found in the following sections.

Table 1. NEBULAR Tasks

Task	Description
abund	Derive ionic abundances in a 3-zone nebula
diagcols	P-set of table column names for computed T_e & N_e for each zone
fivel	Documentation on the 5-level atom approximation
fluxcols	P-set of table column names for input emission line fluxes
ionic	Compute level populations, critical densities, line emissivities & abundance for a single ion
ntplot	Construct log N_e vs. T_e plot for observed diagnostic line ratios
temden	Compute T_e or N_e from diagnostic line ratios
zones	Derive T_e & N_e in 3-zone nebula from diagnostic emission line ratios

This package is not intended to offer a full nebular photo-ionization model, such as G. Ferland's CLOUDY program. Rather, it is most useful for calculating nebular densities and temperatures directly from the traditional diagnostic line ratios, either to provide some reasonable input parameters for a more complicated physical model, or to calculate ionic abundances (or other quantities) within some simplifying assumptions. This package is still under development, which is why it debuts under the **playpen** package, rather than under **analysis**. We hope to develop the **nebular** package more fully, to include tasks for deriving interstellar reddening, calculating ionic abundances from recombination lines, and calculating the nebular continuum flux.

Some of the tasks in this package make use of STSDAS binary tables for accessing potentially large lists of emission line fluxes for many nebulae. The various line fluxes are contained in different columns, and data for different nebulae (or different regions within nebulae) are contained in separate rows. The TABLES **ttools** package provides all the needed utilities for generating, editing, and printing the table contents. An example of a test table can be found in the directory *nebular$data*. Generally, the user has control of the table column names through named parameter sets.

*Nebular Diagnostics and Abundance*s

The **nebular** tasks make use of the fact that most of the common ions that dominate the nebular cooling rate have either p^2, p^3, or p^4 ground-state electron configurations, which have five low-lying levels. The major physical assumption within this algorithm is that only these five levels are physically relevant for calculating the observed emission line spectrum from a given ion.

The **temden** task will calculate N_e given T_e, or T_e given N_e, for a given ion and the associated diagnostic flux ratio. The result is displayed and stored in a task parameter. As an example,

suppose we wish to find the electron density from the [S II] diagnostic ratio: I(6716)/I(6731) = 0.9, assuming an electron temperature of 10,000 K.

```
cl> temden density 0.9 atom=sulfur spectrum=2 assume=10000.
[S ii] density ratio: I(6716)/I(6731) =   0.9
Density: 855.32 /cm^3
```

The **ionic** task will calculate the level populations, critical densities, and line emissivities for a specified ion, given N_e and T_e. It will also calculate the ionic abundance relative to H^+ if the wavelength and relative flux (on the scale I(Hβ) = 100) of one of the emission lines are also specified. For example, suppose we wanted to know the abundance of the O^+ ion, relative to ionized hydrogen. The observed flux in the [O II] 3726.1 + 3728.8 Å emission line doublet (relative to I(Hβ) = 100) is provided (along with a wavelength tolerance large enough to accommodate both lines in the pair), to relate volume emissivities to ionic abundance.

```
cl> ionic oxygen 2 temper=1.e4 dens=1000. wave=3728 wv_toler=2.0 \
>>> flxratio=0.7 verb+

Volume Emissivities for: O^1+
    T_e:  10000.0;  N_e: 1.000E3

Level Populations - Critical Densities

Level 1:   9.9E-1
Level 2: 6.207E-3     3.314E3
Level 3: 1.764E-3     4.491E3
Level 4: 1.210E-7     7.731E6
Level 5: 7.344E-8     4.362E6

        Wavelength:   3729.85
Upper->Lower Level:   (2-->1)
 Volume Emissivity: 1.263E-21

                      3727.14      512.76
                      (3-->1)     (3-->2)
                    1.551E-21   8.201E-28

                      2471.12     7322.36     7332.83
                      (4-->1)     (4-->2)     (4-->3)
                    5.478E-23   3.841E-23   2.013E-23

                      2471.05     7321.77     7332.24     9089.88
                      (5-->1)     (5-->2)     (5-->3)     (5-->4)
                    1.370E-23   1.225E-23   2.030E-23   3.338E-37

H-beta Volume Emissivity: 1.258E-25 N(H+) * N(e-) ergs/s

Log10(x) =   1.000E0

Ionic Abundance: N(O^1+) / N(H+) = 3.129E-7
```

The available ions from which abundances and/or diagnostics can be derived number about two dozen, are given in the online help.

3-Zone Nebular Model

The **zones** task calculates T_e and N_e within each of three zones of low-, medium-, and high-ionization. It uses iteration to make simultaneous use of temperature- and density-sensitive line ratios from different ions with similar ionization potential. The **abund** task computes the abundances for several ions using T_e and N_e as computed by **zones**. The ionization potential of the ion determines which T_e and N_e is used. The input line fluxes, which could exceed 50 in number, are taken from a specified STSDAS table. UV fluxes can be given on a separate flux scale, provided that the UV-to-optical scale factor is specified in the table. The input line fluxes can optionally be corrected for interstellar reddening.

The complementary **ntplot** task produces a plot from which a physical model can be inferred. Like **zones** and **abund**, this task takes its input from an STSDAS binary table (NOT an ASCII table) and plots curves in the N_e, T_e plane that are consistent with the observed diagnostic line ratios. The output curves (which are not shown here) can optionally be stored in an STSDAS table for use as input to a presentation graphics task, such as **igi** in the TABLES **tbplot** package.

For Further Information...

A complete description of the equations to be solved and their method of solution can be found in the paper by De Robertis, Dufour & Hunt. Additional enhancements, including atomic parameters for C III] and Si III], are described by Shaw & Dufour in "Astronomical Data Analysis Software and Systems III", ASP Conf. Ser., Vol. 61, p. 327, 1994. (This paper is also available on the WWW at URL:

http://cadcwww.dao.nrc.ca/ADASS/adass_proc/adass3/papers/shawr/shawr.html.)

Type `help fivel` in the **nebular** package for additional information about the 5-level atom approximation, and for references to the atomic parameters.

This package is available for personal installation for users who do not have STSDAS installed, but who do have IRAF (V2.10.2 or later) and TABLES (V1.3 or later) installed. Simply retrieve the files "neb.tar" (or "neb.tar.Z") and "neb.README" in the directory */software/stsdas/outgoing* from node *ftp.stsci.edu* and follow the installation instructions. Additional updates to this task are described in the "neb.README" file, and in postings to the *adass.iraf.applications* newsgroup. Funding to develop this package was provided by the NASA Astrophysics Data Program, through grant NAG5-1432 to the Space Telescope Science Institute.

This article, with additional tables and figures, appeared in the STSDAS Newsletter (Spring 1994 issue). All of the STSDAS Newsletters are available via WWW through URL: *http://ra.stsci.edu/Newsletter.html*. You may obtain hardcopies of the Newsletter by sending an email request to *hotseat@stsci.edu*.

Dick Shaw
shaw@stsci.edu

FORTRAN Program and IRAF Script Produce Fast Scientific Output

For more than 15 years we have been accumulating large numbers of single-order echelle spectra of stars in order to study the frequency and orbital characteristics of binary stars in a variety of stellar populations. We now have more than 120,000 spectra in our on-line data base, and in a good month we get more than 1,000 new spectra. To handle this volume of data we have developed procedures for mass-producing the data reductions and analysis.

Five years ago we converted from our custom software running on Data General Nova computers to workstations. We ported across the code for doing wavelength solutions and flat-fields, but we decided to do the velocity correlations inside of IRAF. We ended up developing our own cross-correlation application, **xcsao**.

Each month the new spectra are reduced using an observation of the dusk sky as the template. To adjust for the small shifts in the zero point of our velocity system, we take dozens of exposures each month, both of the dusk sky and of standard stars.

When we want the best possible velocities for a given star, to see if it is variable or to solve for its orbit, we have tools for easy extraction of all the spectra of that star from the online data base. Then we can recorrelate the spectra against an optimum template, or a grid of templates, chosen from our library of several thousand synthetic templates calculated by Jon Morse, based on the latest Kurucz stellar atmospheres.

The procedure for recorrelating the spectra of a star or group of stars is quite simple. We prepare a file which gives the name of the star and the characteristics of the template that we want to use (effective temperature, surface gravity, metallicity, and rotational velocity), one line per star/template combination. One option is to ask for a grid of several different rotational velocities. We have FORTRAN programs which take this input and create an IRAF script for correlating each spectrum. The code selects the nearest template from our grid, looks up the proper zero-point shift from another data base, figures out which telescope the exposure was made on, and creates an output file with the necessary IRAF commands to do the correlations, typically 3 or 4 lines per correlation. The FORTRAN program also sets several hundred IRAF parameters at the beginning of the script, so that we can be sure that all the relevant parameters have been set to our default choices.

This approach to mass-producing reductions can be quite powerful. For example, last Sunday I decided to rereduce nearly 4000 observations for 90 spectroscopic binaries using new information about the temperatures and metallicities to choose calculated templates. The biggest part of the job was the preparation of the input file specifying the star names and parameters. It took about 5 hours for my Sun to wade through the correlations, creating an output file which I could then feed into my orbital-solution programs. By the end of the day I had 90 new orbits!

Dave Latham
SAO

Using EVVEXPR to Process Vector Expressions

A new expression evaluator **evvexpr()** is available in the IRAF programming environment starting with V2.10.3BETA. This is similar to the old expression evaluator **evexpr()** except that it will operate on and can return vector operands, supports mixed scalar-vector expressions and a wider range of data types, and provides a number of new intrinstic functions. The conventional arithmetic, trigonometric, and type conversion intrinsic functions are available plus a number of projection functions (e.g., length of a vector, median value of a vector, etc.) and miscellaneous other functions. In addition there is a mechanism by which applications programs can define and use their own custom intrinsic functions.

This function is used in the new **imexpr** and **import/export** tasks (see accompanying articles) to evaluate expressions using image operands. It can also be used to evaluate scalar expressions and it is recommended that **evvexpr** be used for new tasks instead of the older **evexpr**, which is considered obsolete. For more information see the **imexpr** help page, the *sys$fmtio/evvexpr.gy* source file, or contact the IRAF group (*iraf.noao.edu*).

Doug Tody
Mike Fitzpatrick

New Image Expression Evaluation Task

A new task for evaluating general image expressions, **imexpr**, has been released with IRAF 2.10.3BETA. **imexpr** evaluates an arbitrary input image expression and writes the results to an output image. **imexpr** can be used for a wide variety of applications including, but not limited to, simple image arithmetic, image editing, image combining, function evaluation, and artificial image and pixel mask generation. The input images may be any dimension or size and any datatype. Expressions can combine images of different dimensions or datatypes. Image expressions may be read from the command line or from a file and may consist of one or more operands, operators, builtin functions, and user defined macros.

imexpr supports three types of operands: images and image sections, image header parameters, and numeric constants. Operands are represented symbolically in the expression by the letters a-z. Use of the symbolic operands permits the same expression to be used with different data sets, simplifies the expression syntax, and allows a single input image to be used several times in the same expression. The operands a-z are implemented as **imexpr** task parameters. The following examples show how **imexpr** can be used to compute the weighted sum of three images or to add two images together after normalizing by their respective exposure times.

```
cl> imexpr "x*a+y*b+z*c" ave a=im1 b=im2 c=im3 x=.25 y=.50 z=.25
```

or equivalently

```
cl> imexpr "0.25*a+0.50*b+0.25*c" ave a=im1 b=im2 c=im3

cl> imexpr "a/c+b/d" norm a=im1 b=im2 c=a.exptime d=b.exptime
```

or equivalently

```
cl> imexpr "a/a.exptime+b/b.exptime" norm a=im1 b=im2
```

where ‘‘exptime’’ is an image header parameter of images A and B.

In simple expressions like the above the same expression is evaluated at every point in the image, with only the input pixel values changing. It is also possible to implement space-variant functions by including the image pixel coordinates in the expression, using a special builtin set of operands. These operands have the names I, J, K, and so on, up to the dimensions of the image (upper case is required to avoid confusion with the input operand names i, j, k, etc.). This capability is especially useful for generating functions and test patterns.

The following example shows how this feature of **imexpr** can be used to create two artificial images, one a 2D test image where the pixel value gives the coordinates of each pixel, and the other a sampled 1D Gaussian.

```
cl> imexpr "I+J*512" coordsim dim=512,512 outtype=int

cl> imexpr "a*exp(-((I-b)/c)** 2/2.)" gauss \
>>> a=1000. b=255 c=3.5 dim=512 outtype=real
```

The **imexpr** expression syntax supports all the standard arithmetic and logical operators. Of special note is the conditional expression which permits pixel-dependent operations such as checking for and editing special pixel values. The following example replaces all the negative pixels in an image by the corresponding pixels from a second image "b". The two branches of a conditional expression may be arbitrary expressions.

```
cl> imexpr "(a < 0) ? b : a" image.noneg a=image
```

String and vector concatenation operators and a substring equality operator are also provided.

imexpr supports all the standard mathematics and type conversion scalar functions including conversion from degrees to radians and vice versa. In addition some special functions which accept both scalar and vector arguments are provided. Functions which operate on vectors are

applied to the lines of an image. When applied to an image of dimension 2 or greater these functions are evaluated separately for each line of the multidimensional image. The following two examples show 1) how **imexpr** can be used to compute the square root of an image and 2) how to subtract the median of a section of each image line from each image line using the scalar function *sqrt* and the vector projection function *median.*

```
cl> imexpr "sqrt(a)" image.sqrt a=image

cl> imexpr "a - median(b)" image.lmedsub a=image b=image[513:530,*]
```

The **imexpr** expression database provides a macro facility which can be used to create custom libraries of functions for specific applications. For example users may create their own ‘‘what to do if dividing by a small number function’’ by defining a simple macro like the following and entering it in a text file.

```
divz(a,b)   ((abs(b) < .000001) ? 0.0 : a/b)
```

Macros can be nested and are indistinguishable from intrinsic functions in expressions. Expression databases are particularly useful for expressions which are too large or too cumbersome to enter on the command line.

imexpr can also be used to create image masks. The following examples show how to create boolean mask images using 1) an input image and a conditional expression, and 2) a generating function defining a circular mask. The output mask is written to a compressed ".pl" pixel list image.

```
cl> imexpr "a >= 0.0 && a <=10000.0" mask.pl a=image

cl> imexpr "(((I-255.0)**2+(J-255.0)**2) < 150.0)" circle.pl \
>>> dim=512,512
```

imexpr is available in the **images** package.

Doug Tody
Lindsey Davis

New Image Registration Task

A new task for registering 1 and 2 dimensional images using cross-correlation techniques, **xregister**, has been released with IRAF 2.10.3BETA. **xregister** computes the x and y shifts required to register a list of input images to a reference image by finding the position of the peak of the input image reference image cross-correlation function, then performs the shift. **xregister** is designed to register 1 and 2 dimensional images which have the same size and pixel scale, are shifted relative to each other by simple translations in x and y, and contain at least one feature in common that will produce a peak in the computed cross-correlation function. These features may be point sources like stars, extended sources with well-defined centers like galaxies, or diffuse features like HII regions.

xregister permits the user to define starting x and y shifts either by inputting them directly or by inputting the coordinates in each image of a feature common to all the images, to fit and subtract the background before computing the cross-correlation function, to compute the cross-correlation function directly or using fourier techniques, and to compute the position of the peak of the cross-correlation function using a variety of algorithms. The user can also select the interpolant type and boundary extensions algorithm to be used to compute the shifted output image.

xregister can be run non-interactively or interactively. In non-interactive mode the parameters are set at task startup and the input images are processed sequentially. In interactive mode the user can mark the regions to be used to compute the cross-correlation function on the image display, define the initial shifts from the reference image to the input image on the image display, show or set the data and algorithm parameters, compute, recompute, and plot the cross-correlation function, experiment with the various peak fitting algorithms, and overlay row and column plots of the input and reference images with and without the initial or computed shifts factored in.

xregister is installed in the **images** package. Users who have not yet upgraded to IRAF 2.10.3BETA or who are running IRAF on platforms for which IRAF 2.10.3BETA is not yet available, can acquire the new task by retrieving and installing the external layered package **nmisc** from our FTP archive.

Lindsey Davis

Fitting a Surface with SURFIT

A new task is available in IRAF V2.10.3/V2.11 and also in the **nmisc** external package that fits a two dimensional surface to a set of possibly irregularly sampled points specified in a text file. This task is called **surfit** and is part of the general **utilities** package. The input data points are used in a weighted least-squares fit to a surface function; a function of two coordinates. The type and order of the surface is up to the user. The results of the fit can be output in three ways. The surface coefficients and the fitted values at the input points may be written out. Alternatively, another file of coordinates can be input, and the fitted values at those points will be output. Finally, the surface can be sampled in an even grid, as defined by the user, to produce an image of the surface. This task will have uses in smoothing irregular 2D data, and interpolating and creating smooth images of randomly sampled data. It is intended for data with relatively low spatial frequencies and not for creating and sampling images with high spatial frequencies.

Frank Valdes

Editing and Modifying Headers with ASTHEDIT

A new task **asthedit** has been added to **noao.astutil** in V2.10.3BETA and later versions of IRAF. The new task is specially tailored for editing the headers of astronomical images. One of the uses for this task is to fix up or translate image headers from various observatories, or the very minimal headers from commercial CCD systems, for use with the NOAO or other astronomical packages which require certain keywords. It takes commands from a text file so that one such command file may be created for a certain type of data.

The commands consist of a keyword to be modified, created, or deleted and an expression containing references to other keywords, columns from a text table with lines corresponding to each image, string and numeric constants, functions, and algebraic and boolean operators. Because it is oriented to astronomical images it has a number of special astronomical functions such as sidereal time computations and sexagesimal string formatting.

Below is an example of adding keywords to a minimal ESO / IHAP image. This example is shown since it provides great contrast between the keywords recorded and those used in NOAO tasks. It illustrates use of the astronomical functions (for example, *mst* for mean sidereal time),

arithmetic time computations recorded as sexagesimal strings (the *sexstr* function), access to the observatory database (the *obsdb* function), and reference to columns from a text file (the *$imagetyp* and *$filter*). In the latter the columns are identified with a task parameter.

The command file:

```
cl> type eso.dat
observat  "eso"
ut        sexstr ((@'tm-start'+0.1) / 3600.)
utend     sexstr ((@'tm-end'+0.1) / 3600.)
epoch     epoch (@'date-obs', ut)
st        mst (@'date-obs', ut, obsdb (observat, "longitude"))
exptime   (utend>ut)?(utend-ut)*3600.:(utend+24-ut)*3600.
ra        sexstr (@'postn-ra' / 15)
dec       sexstr (@'postn-dec')
airmass   airmass (ra, dec, st, obsdb (observat, "latitude"))
imagetyp  $imagetyp
filter    $filter
cl> type table.dat
object      V
```

An initial image header:

```
as> imhead eso
....
DATE-OBS= '12/12/92'            / Date this data created dd/mm/yy
TM-START=             84854.    / '23:34:14' measurement start time
TM-END  =             84974.    / '23:36:14' measurement end time (U
TIME-SID=                 1.    / '00:00:01' sidereal start time
POSTN-RA=           354.0709    / '23:36:17' tel. position right-asc
POSTN-DE=           6.556945    /'+06:33:25' tel. position declinati
....
```

Edit the header with **asthedit**:

```
as> asthedit eso eso.dat table=table.dat col="imagetyp filter"
```

The final header:

```
as> imhead eso
...
OBSERVAT= 'eso     '
UT      = '23:34:14'
UTEND   = '23:36:14'
EPOCH   =            1992.95
ST      = '0:18:56.76'
EXPTIME =             120.005
RA      = '23:36:17'
DEC     = '6:33:25 '
AIRMASS =           1.255875
IMAGETYP= 'object  '
FILTER  = 'V       '
...
```

Frank Valdes

Revisions to IMCOMBINE for V2.10.2

The task **imcombine** and its derivatives **onedspec.scombine**, **ccdred.combine**, and the **ccdred** ccd-type specific combining tasks were significantly revised for the V2.10.2 patch release of IRAF. This article summarizes the changes. The subsequent V2.10.3BETA / V2.11 releases contain bug fixes after the V2.10.2 release but no new functional features. The known bugs are documented in the buglog available by FTP from the IRAF archives, via the *irafinfo* mail server facility ("get iraf buglog"), or in the adass.iraf.buglog newsgroup; check these sources first if problems are encountered.

The revisions provide:

- weighting by the estimated variance rather than the sigma
- more flexibility in specifying the scaling and weighting factors
- a limit to the maximum number of pixels rejected by the clipping options
- a more general CCD noise model by including sensitivity noise
- additional parameters in the CCDRED type specific combining scripts.

The weights, when using a weighted average in all previous versions of the combine tasks (except the old **onedspec.combine** task), were based on estimates of the sigma (the square root of the variance). However, variance weighting is more commonly used so the weights are now based on variance estimates.

Specification of scaling and weighting factors was made more flexible by allowing the factors to be specified from text files (using the standard @file convention) and from image header keywords. The header keywords can now be different for the different factors and can be different than the exposure time. The documentation indicates that the scaling, both offset and gain, are to be numbers to add or multiply to the image. However, in V2.10.2 the code was incorrect and used the values to subtract and divide. In subsequent versions this was corrected so be aware that the interpretation of this type of input will change to that given in the documentation.

In V2.9 the clipping rejection algorithms were limited to rejecting a single pixel. In V2.10 this restriction was dropped and as many pixels as exceeded the clipping threshold could be rejected. However, this opened the possibility of rejecting all pixels. This might occur even with normal data by chance when the closest pair of pixels to the central median or average are above and below that value by a few sigma. This can become significant if the sigma factors are made small (2 sigma or less) or the sigmas are underestimated by the CCD noise parameters or the ‘‘avsigclip’’ estimate. This possibility is particularly a problem when combining flat-fields where it is better to preserve even a poor estimate of the flat-field level than using the blank value (which defaults to zero).

In this revision a new parameter, ‘‘nkeep’’, was added to allow specifying the minimum number of pixels to keep or, by using a negative value, the maximum number of pixels to reject. The default value is set at 1 which requires at least one pixel be retained by the clipping algorithms. Note that this parameter does not apply to bad pixel masking or threshold rejection.

In line with this change and the comments above the **ccdred** ccd-type specific combining scripts were modified to include some additional parameters. The ‘‘nkeep’’ parameter was added and the ‘‘blank’’ parameter was made a parameter of these tasks rather than being fixed at zero. For **flatcombine** the default ‘‘blank’’ value was set to one rather than zero. Also the new ‘‘noise’’ parameter, described below, was added.

The final change was to add a third CCD noise parameter called ‘‘snoise’’. This provides a sensitivity or flat-field noise component. This type of noise scales with the intensity rather than the square root as provided by the Poisson term. The default value is zero which preserves the previous noise model but does allow some additional flexibility which I believe is important in typical data. A 1% uncertainty in the flat-field correction can give uncertainties for high signals well in excess of the Poisson uncertainties. Without this the centers of bright stars could be

clipped by the ‘‘"ccdclip’’ or ‘‘crreject’’ algorithms. I have found the task **stsdas.hst_calib.wfpc.noisemodel** a nice one to characterize the CCD noise parameters including the sensitivity noise from groundbased CCD data as well as WFPC data. A point to note is that the read noise is specified in DN in that task instead of electrons as is used by **imcombine**.

The most common error message encountered by sites which updated to V2.10.2 when using the revised version of **imcombine** and related tasks is ‘‘parameter ‘snoise’ not found’’. This is due to an incomplete installation of the patch. The Frequently-Asked-Questions file describes this common error and its solution.

Frank Valdes

An Enhancement to COSMICRAYS

A new capability has been added to the **cosmicrays** task in V2.10.3BETA. When in interactive mode, the task now includes an optional ‘‘training’’ feature in which users can designate image objects as stars or cosmic rays. This information is then used to set the flux ratio threshold that best separates the two classes of objects. The user first loads the image to be cleaned into the image display device and then calls **cosmicrays** with the training feature turned on. The task will then read from the image display cursor, and allow the user to mark objects by typing ‘‘c’’ for cosmic rays or ‘‘s’’ for stars. As objects are assigned, their corresponding symbols are updated in the graphics display. The user may switch to the graphics display to view surface plots or find a feature’s image coordinates, and then back to the image display for more training. For non-interactive use, a list of training object positions and designations may also be supplied in a cursor file.

Additional information is given in the help page, and an example appears in a new IRAF document called *Cleaning Images of Bad Pixels and Cosmic Rays*, which is available by FTP to *iraf.noao.edu* in the */iraf/docs directory* as file ‘‘clean.ps.Z’’.

Dave Bell

CCDTIME—Predicting Exposure Times

There is an improved and generalized version of the **ccdtime** task in the **noao.astutil** package for V2.10.3BETA / V2.11. This task estimates exposure times to achieve a desired signal-to-noise ratio (SNR) or the converse, the SNR and magnitudes for a given exposure time. The important generalization is that the information used to derive the exposure times are obtained from a text database file. This allows new instruments to be added, the user to have a personal file, and any observatory or user of an observatory to use this task by creating and calibrating a database file. The database contains information about the telescopes, the filters, and the detectors. These divisions allows filters and detectors to move between telescopes; the main change in the calculation would be the telescope aperture, scale, and transmission. Standard files are distributed in the directory *ccdtime$* (defined when the NOAO package is loaded). To look at one such file try:

```
cl> page ccdtime$kpno.dat
```

It is very important to understand that this task provides only an estimate and that the SNR is the formal value for ideal aperture photometry once the center of the star is known. The SNR may sound good for the aperture photometry, while the per pixel SNR of even the central pixel may be barely above the noise. Generally, objects with SNR < 4 will not even be found, and often a SNR > 9 is required to assure completeness in surveys. Thus, please be conservative when using this task to estimate observations to define observing programs.

Frank Valdes

New Radial Velocity Tasks, RVIDLINES and RVREIDLINES

Two new tasks for measuring radial velocities from individual line identifications have been added to the **rv** package in V2.10.3BETA / V2.11. These are in addition to the template cross-correlation method already implemented. For very low signal-to-noise objects, such as distant redshift determinations based on a single or few lines, or for objects with a rich set of lines which can be accurately identified from a coordinate list, the new tasks may be more suitable than cross-correlation and can give very good results.

To provide a familiar interface, and since the functions are so similar, the task **rvidlines** works much like **identify**. A dispersion calibrated spectrum is read and the user marks lines and provides the rest or reference wavelength. One enhancement over **identify** is that the line centering algorithm includes Gaussian fitting and deblending in addition to the standard centering algorithm. With one or more lines a radial velocity is computed, and both rest and observed wavelength scales are shown. Line identifications can come from a coordinate list. The list is in rest wavelength so if there is a significant velocity shift one can mark one line to get a first estimate of the velocity, and then the rest of the lines in the coordinate list can be automatically found. When there are multiple lines the measurement produces a mean velocity and error estimate based on the scatter of the individual lines.

The state of the identifications can be saved in a database file and a log of the identified lines and measured velocities (including heliocentric corrections) and errors will be written.

The **rvreidlines** task is, naturally, modeled after **reidentify**. It takes the lines identified first with **rvidlines** and automatically finds, recenters, and recomputes velocities for other spectra. This is useful when there are many observations of the same object or objects of similar velocity and spectral features.

Frank Valdes

New Conversion Tools for 1D Image Spectra

One dimensional spectra are sometimes imported and exported in the form of a text file with columns of wavelength and flux. While there are tools scattered about that do some of the desired conversions—**sinterp**, **listpixels**, **r/wtextimage**, **mkmultispec** (in the STSDAS package)—two specific tasks that provide these conversions in a complete (both header and pixels) and flexible (linear and non-linear dispersion functions) way have been added to IRAF V2.10.3BETA / V2.11. These new tasks are in the **onedspec** package and are called **rspectext**, for reading a text file into a 1D IRAF spectral image, and **wspectext**, for writing a text file from an IRAF spectral image. Naturally, the output of **wspectext** can be read by **rspectext** without

loss of header or dispersion information. Often, **rspectext** is used for import of a purely wavelength/flux file without header information. The task will take care of defining the necessary header keywords.

One very important feature of **rspectext** is that it provides three alternative ways to store dispersion information in the output IRAF image. One is to fit and resample the input to a linear dispersion function. Alternatively, the spectrum can be represented using a non-linear function fit to the input wavelengths, or the wavelengths can be entirely stored as a lookup table (which is not recommended for very long spectra).

Frank Valdes

SFLIP—Flip Data or Dispersion Coordinates in Spectra

A new task called **sflip**, for flipping spectra has (re)appeared in IRAF V2.10.3BETA / V2.11. The task has four parameters, an input list of spectra, an output list, a parameter to select whether to flip the dispersion coordinates, and a parameter to select whether to flip the data (storage order in the image) along the dispersion axis. The task allows the output to be the same as the input to modify images rather than create new ones.

Besides providing an easily identifiable task for spectroscopy users this task's principal purpose is to separate flipping of the data and the coordinate system. Using image sections to flip spectra, say with **imcopy**, flips both the dispersion coordinates and data. To only flip the data or the coordinate system using the general system tasks requires several steps and some knowledge of the WCS coordinate system.

Actually this task is not entirely new since there was a task by that name in V2.9. The new task has much the same function but is implemented differently because of changes in how dispersion coordinate systems are handled. In principle this task should never be needed since the spectral software (V2.10 and beyond) works in dispersion coordinates, but there are rare cases, mainly to do with interfacing to user programs outside of IRAF, when it may be of use.

Frank Valdes

APPHOT Package Update

The following major changes have been made to the aperture photometry package **apphot** in IRAF V2.10.3BETA.

The **apphot** tasks **daofind**, **center**, **fitpsf**, **fitsky**, **qphot**, **polymark**, **radprof**, and **phot** were modified to support the IMDKERN graphics kernel, used to draw graphics overlays on the image display. These tasks will mark the positions of detected or selected objects and draw the sky, photometry, and fitting apertures as appropriate if the user 1) runs the task in interactive mode, 2) sets the task parameter **display** to "imd", and 3) sets the appropriate graphics parameters, e.g., ‘‘mkcenter’’, ‘‘mksky’’, and ‘‘mkapert’’ to "yes".

The star finding task **daofind** and its algorithm parameter sets have undergone major changes.

- The parameter **convolution** specifying the name of the output density enhancement image has been renamed **starmap**, and a new parameter specifying the name of the output background density image **skymap** has been added. The sum of these two images models the

input image except at the image edges and in regions containing bad data.

- **daofind** has been modified to read the **datapars** parameters ‘‘datamin’’ and ‘‘datamax’’. These parameters are used to reject bad data from the computed density enhancement and background density images.
- The object detection parameters ‘‘nsigma’’, ‘‘ratio’’, ‘‘theta’’, ‘‘sharplo’’, ‘‘sharphi’’, ‘‘roundlo’’, ‘‘roundhi’’, and ‘‘mkdetections’’ have been moved to the new **findpars** parameter set task.
- The **datapars** parameter ‘‘threshold’’ has been moved to the **findpars** parameter set and defined in units of the **datapars** parameter ‘‘sigma’’ rather than units of counts. Together ‘‘threshold’’ and ‘‘sigma’’ set the brightness of the faintest star that can be detected by the **daofind** task.

The **datapars** parameter ‘‘cthreshold’’ has been moved to the **centerpars** parameter set and redefined in units of the **datapars** parameter ‘‘sigma’’ rather than units of counts. Together ‘‘cthreshold’’ and ‘‘sigma’’ define the faintest pixels that can be used by the centering algorithms in the **center**, **qphot**, **phot**, **polyphot**, **radprof**, and **wphot** tasks.

Two new keystroke commands ‘a’ and ‘o’ were added to the photometry tasks **qphot**, **phot**, **polyphot**, **radprof**, and **wphot**. The ‘a’ keystroke command computes the sky value by averaging sky values computed at several cursor positions, and the ‘o’ keystroke command uses the computed value to do the photometry. The new commands add an offset sky capability to the **apphot** photometry tasks which can be useful for measuring extended asymmetric objects like comets.

The sky fitting algorithms and associated parameter set **fitskypars** used by the **fitsky**, **qphot**, **phot**, **polyphot**, **radprof**, and **wphot** tasks have undergone major changes.

- Two new parameters ‘‘sloclip’’ and ‘‘shiclip’’ have been added to the **fitskypars** parameter set. ‘‘sloclip’’ and ‘‘shiclip’’ define the percentage of pixels to be clipped from the low and high sides of the sorted sky pixel distribution after bad data is rejected but before the sky value is computed by the photometry tasks. Their default values are 0 and 0, respectively.
- The **fitskypars** parameter ‘‘skreject’’ specifying the k-sigma rejection limit has been replaced with two new parameters, ‘‘sloreject’’ and ‘‘shireject’’, specifying the low and high side k-sigma rejection criteria, respectively.
- The default value of the **fitskypars** parameter ‘‘salgorithm’’ was changed from "mode" to "centroid" to minimize sky determination problems for data taken in regions with very confused background (e.g., nebulosity) and with data taken where the background noise is poorly sampled.
- The definition of the median of the sky pixel distribution was changed from the average of the central 10 pixels to the average of the central 5% of the pixels. Both definitions were intended to help protect against quantization effects in the data but the newer definition gives a more consistent definition and better results.
- The bad sky pixel rejection code has been modified to provide better protection against deviant pixels in the sky distribution. In the first bad sky pixel rejection cycle, the data is limited to values between [sky−min (sky−mindata, maxdata−sky, sloreject∗skysigma)] and [sky + min (sky−mindata, maxdata−sky, shireject∗skysigma)] instead of values between [sky−sloreject∗skysigma] and [sky + shireject∗skysigma] as before. This change provides better protection against deviant pixels in cases where the user forgets to set ‘‘datamin’’ or ‘‘datamax’’, and against defects like cosmic rays which are inside the good data limits but can sometimes bias the computation of skysigma. The value of sky in the above expressions is the median of the original sky pixel distribution except in the case of the "mean" algorithm where it is the mean or the original distribution.

- The size and resolution of the sky pixel histogram required by the "histplot", "centroid", "gauss", "crosscor", and "ofilt" sky fitting algorithms is no longer determined by the value of the **datapars** parameter ‘‘sigma’’ if it is defined. Instead the computed median and standard deviation of the sky pixels, the values of the ‘‘khist’’ and ‘‘binsize’’ parameters, and limits defined by [median−min (median−mindata, maxdata−median, khist∗sigma)] and [median + max (median−mindata, maxdata−median, khist∗sigma)] are always used to determine the total width and binwidth of the sky histogram.

Two new keystroke commands ‘m’ and ‘n’ have been added to the **apphot** tasks. These commands permit the user to move to the next object in the coordinate list or measure the next object in the coordinate list, respectively, by typing a single keystroke rather than the two keystrokes required by the equivalent ‘:m’ and ‘:n’ commands.

Users who have not yet upgraded to IRAF 2.10.3BETA or who are running IRAF on platforms for which IRAF 2.10.3BETA or a later version is not yet available, can acquire the updated version of **apphot** by retrieving and installing the external package **digiphotx** from the IRAF network archive.

Lindsey Davis

New Version of the DAOPHOT Crowded-Field Photometry Package Released

A new version of the crowded-field photometry package **daophot** was released with IRAF V2.10.3BETA. The new version of **daophot** is the IRAF implementation of the DAOPHOT II package (Stetson, P. B. 1992, *Astronomical Data Analysis Software and Systems I*, PASP Conf. Series, Vol. 25, and references therein) and replaces the previous version of **daophot**. This article reviews the major differences between the two versions of **daophot**.

The major algorithm changes are listed below.

- The analytic component of the psf model is no longer restricted to a Gaussian function. The user may choose between 6 alternatives, including a Gaussian function, two Moffat functions, a Lorentzian function, and two Gaussian + Lorentzian functions, or allow the psf modeling code to choose the most appropriate analytic model based on a goodness of fit criteria.
- Constant psf models may be purely analytic (analytic function + 0 look-up tables) as well as empirical (analytic function + 1 look-up table).
- Variable psf models may vary quadratically with position in the image (analytic function + 6 look-up tables) as well as linearly with position in the image (analytic component + 3 look-up tables).
- The analytic component of the psf model is no longer computed by fitting the model to the first psf star. Instead the data from all the psf stars, appropriately weighted by signal-to-noise, is used to compute the analytic component of the model.
- The ability to down-weight bad pixels, always present in the psf fitting code, was added to the psf modeling code.
- The ability to combine data in the cores of relatively faint stars with data in the wings of saturated bright stars in order to compute high signal-to-noise look-up tables was added to the psf modeling code.
- The ability to refit the local sky brightness during psf fitting was added to the psf fitting tasks.

One major consequence of these algorithm changes is the psf models written by the new version of the **daophot psf** task are not compatible with psf models written by the previous version of

the **psf** task and vice versa, even if the analytic component of the psf model is Gaussian and the number of look-up tables is 1 or 3. All the photometry files are backwards compatible.

Six new tasks were added to **daophot**.

- **daoedit** is an interactive task which uses the IRAF **lpar**/**epar**/**unlearn** tasks, the image display, and where appropriate radial profile plots of selected stars to review and / or edit the algorithm parameter sets.
- **findpars** is a parameter set task which sets the values of the object detection algorithm parameters required by the **daofind** task.
- **pcalc** is a task which performs simple arithmetic operations on a numerical field in a **daophot** photometry file, e.g., shifting the coordinates of an object.
- **pfmerge** is a task which creates a new photometry file suitable for input to the **daophot** psf fitting tasks **peak**, **nstar**, and **allstar**, by combining existing photometry files that have been written by different tasks, e.g., **allstar** and **phot**.
- **pstselect** is a task which creates a candidate psf star photometry file suitable for input to the **psf** task by selecting bright relatively uncrowded stars from an existing photometry file.
- **setimpars** is a task which saves and restores the **daophot** algorithm parameters for a specific image.

Major changes have been made to the **daophot** algorithm parameter sets to support the package algorithm changes.

- The **datapars** parameter ‘‘threshold’’ has been moved to the **findpars** parameter set and defined in units of the **datapars** parameter ‘‘sigma’’ rather than in units of counts. Together ‘‘threshold’’ and ‘‘sigma’’ set the brightness of the faintest star that can be detected by the **daofind** task.
- The **daofind** object detection parameters ‘‘nsigma’’, ‘‘ratio’’, ‘‘theta’’, ‘‘sharplo’’, ‘‘sharphi’’, ‘‘roundlo’’, ‘‘roundhi’’, and ‘‘mkdetections’’ have been moved to the new **findpars** parameter set described above.
- The new parameter ‘‘function’’ has been added to, and the existing parameter ‘‘varpsf’’ has been replaced with ‘‘varorder’’ in, the **daopars** parameter set. ‘‘function’’ and ‘‘varorder’’ set the form of the analytic component and the number of look-up tables in the psf model computed by the **psf** task.
- The new parameters ‘‘nclean’’ and ‘‘saturated’’ have been added to the **daopars** parameter set. ‘‘nclean’’ sets the number of iterations the **psf** task must make when computing the psf model look-up tables. ‘‘saturated’’ specifies whether or not saturated stars will be provisionally accepted as psf stars by the **psf** task.
- The new parameters ‘‘fitsky’’, ‘‘groupsky’’, ‘‘sannulus’’, and ‘‘wsannulus’’ have been added to the **daopars** parameter set. These parameters are used by the **peak**, **nstar**, and **allstar** tasks to determine if and how the group sky value will be recomputed during the psf fitting process. If ‘‘fitsky’’ is "yes" the group sky value is recomputed; if ‘‘fitsky’’ is "no" (the default) the values computed by the **phot** task are averaged to compute the group sky value. If ‘‘groupsky’’ is "yes" (the default) the sky value for each pixel contributing to the fit is set to the average of the current sky values of all the stars in the group; if ‘‘fitsky’’ is "no" the sky value for each pixel is set to the average of the sky values of stars in the group within ‘‘fitrad’’ of that pixel. In the case of **peak** and **nstar** the pixels used to recompute the sky are within one ‘‘fitrad’’ of the star; in the case of the **allstar** task the sky pixels used to recompute the sky are defined by ‘‘sannulus’’ and ‘‘wsannulus’’ parameters.

- The parameters ‘‘flaterr’’ and ‘‘proferr’’ have been added to the **daopars** parameter set. ‘‘flaterr’’ and ‘‘proferr’’ are used by the psf fitting code in the **peak**, **nstar**, and **allstar** tasks to define the flat-fielding and interpolation components of the error model, respectively. In previous versions of **daophot** these values were fixed internally.

The existing **daophot** tasks have also undergone major revisions.

- The **daofind** task parameter ‘‘convolution’’ specifying the name of the output density enhancement image has been renamed ‘‘starmap’’, and a new parameter specifying the name of the output background density image ‘‘skymap’’ was added. The sum of these two images should model the input image except at the image edges and in regions containing bad data.
- The **daofind** task has been modified to read the **datapars** parameters, ‘‘datamin’’ and ‘‘datamax’’. These parameters are used by **daofind** to reject bad data from the computed density enhancement and background density images.
- The **psf** task has been modified to accept lists of input images and photometry files and write lists of output psf images and photometry files. Previous versions of **psf** accepted only a single input image and photometry file, and wrote only a single output psf image and photometry file.
- The **psf** task has been modified to accept an input list of candidate psf stars (new parameter ‘‘pstfile’’) and output a list of the psf stars (new parameter ‘‘opstfile’’) actually used to compute the psf model. The new parameter ‘‘matchbyid’’ specifies whether the psf stars are matched to stars in the input photometry file by id number (the default) or by positional coincidence.
- A new parameter ‘‘interactive’’ was added to the **psf** task. If ‘‘interactive’’ is "yes", the **psf** task accepts commands from the user via the image display cursor; if ‘‘interactive’’ is "no", **psf** can either accept and execute commands from a cursor command file or compute the psf model using stars in the input psf star list without intervention by the user.
- The residuals plots drawn by the new **psf** task are displayed after the psf model is fit instead of as the psf stars are selected, and have a different meaning than they did in the old **psf** task. The new residuals plots show the residuals from the best fit to the entire psf model, not from the best fit analytic component of the model. Ideal residuals plots are flat with per pixel rms noise values appropriate to the detector and the data, not the doughnuts users are used to seeing.
- The **psf** task plotting code is now capable of drawing radial profile plots in addition to surface and contour plots.
- The **peak** and **nstar** tasks have been modified to read the **daopars** parameter ‘‘recenter’’. The ‘‘recenter’’ parameter determines whether the psf fitting code will compute new x and y coordinates or use the input values. The **allstar** task has always been able to read the ‘‘recenter’’ parameter.
- The **peak** and **nstar** tasks have been modified to read the **daopars** parameters ‘‘clipexp’’ and ‘‘cliprange’’. In previous versions of **daophot** these values were fixed at 8.0 and 2.5 internally, and only the **allstar** task could read these parameters.
- A new parameter ‘‘rejfile’’ with a default value of "default", has been added to the **peak**, **nstar**, and **allstar** tasks. If ‘‘rejfile’’ is not equal to "" then stars which fail to be fit for one reason or another are written to a separate rejections file instead of to the output photometry file.
- Error code and message columns have been added to the output photometry files written by the **peak**, **nstar**, and **allstar** tasks so the user can determine precisely why the code failed to fit a particular star.

- The **substar** task has been modified to accept an input list of stars to be excluded from the subtraction process (new parameter ‘‘exfile’’). Stars in the "to be excluded list", e.g., psf stars, whose ids match those in the input photometry file are not subtracted from the input image.
- Minor changes have been made to the **pappend**, **pdump**, and **prenumber** tasks. These are described in the accompanying article on the **ptools** package.

A new manual called *A Reference Guide to the IRAF/DAOPHOT Package* has been written and is available in PostScript form from our FTP archive.

Users who have not yet upgraded to IRAF 2.10.3BETA, or who are running IRAF on platforms for which IRAF 2.10.3BETA is not yet available, can acquire the updated version of **daophot** by retrieving and installing the external package **digiphotx** from the IRAF network archive.

Lindsey Davis

PHOTCAL Package Update

The following major changes have been made to the photometric calibration package **photcal** in IRAF V2.10.3BETA.

Two new tasks for computing aperture corrections using the growth-curve method, **mkapfile** and **apfile**, have been added to the **photcal** package. See the accompanying article in this Newsletter for a more detailed description of these tasks.

The new Landolt UBVRI (Landolt, A. U. 1992, AJ, 104, 340) and Elias JHKL (Elias et al. 1982, AJ, 87, 1029) standard star catalogs have been added to the **photcal** standard star catalog library. Assuming that the catalog directory is set to its default value of *photcal$catalogs/*, the user can access these new catalogs by inputting "nlandolt" or "elias" opposite the ‘‘catalogs’’ parameter in the **mkconfig**, **fitparams**, **invertfit** and **evalfit** tasks.

Support for reading the time of the observations from the input apphot or daophot photometry files and writing it to the output observations files has been added to the **mkimsets**, **mknobsfile**/**mkobsfile**/**obsfile**, and **mkphotcors** tasks. The change was made to more easily enable users to check for zero point discontinuities / shifts in their fitted transformations during the course of the night, and to add a UT term to their transformation equations if appropriate. However, as a result of this change the output observations contain a new column not present in observations files created prior to V2.10.3BETA. As a result old configuration files will **not** work with new observations files, and vice versa, without modification. Old configuration files will still work with old observations files.

A new parameter ‘‘minmagerr’’ has been added to the **mknobsfile**/**mkobsfile**/**obsfile** tasks. Non-INDEF valued magnitude measurements with errors less than the current values of ‘‘minmagerr’’ are assigned the current value of ‘‘minmagerr’’. The new parameter is used to assign a reasonable error to a magnitude measurement when the error is too small to fit in the assigned space in the apphot/daophot database file, and to help prevent the dynamic range of the error measurements from seriously unbalancing the weighting scheme.

Users who have not yet upgraded to IRAF 2.10.3BETA, or who are running IRAF on platforms for which IRAF 2.10.3BETA is not yet available, can acquire the updated version of **photcal** by retrieving and installing the external layered package **digiphotx** from our FTP archive.

Lindsey Davis

PTOOLS Package Update

The following major changes have been made to the photometry tools package **ptools** in IRAF V2.10.3BETA.

A new task **pcalc**, and associated tasks **txcalc** and **tbcalc**, have been added to the **ptools** package. **pcalc** replaces the value of a numerical field in an apphot or daophot database with the value of a numerical expression composed of field names and constants supplied by the user. Typical applications of **pcalc** include adding an offset to stellar id fields, shifting the computed stellar coordinates, and correcting the zeropoint of the computed stellar magnitudes.

A new parameter ''expr'' has been added to the **pdump** and **tbdump** tasks in order to make the functionality of these tasks for the ST binary tables apphot/daophot database the same as the existing functionality of the **txdump** task for text apphot and daophot databases. The new parameter permits **pdump** to select and print records from a database by evaluating a user supplied boolean expression.

A new parameter ''idoffset'' has been added to the **prenumber** task, and to the associated tasks **txrenumber** and **tbrenumber**, in order to permit the renumbering to begin at an arbitrary offset rather then always beginning at one. The new parameter permits groups of stars, for example, artificial stars that have been added to an image, to be easily identified after apphot or daophot databases have been merged and subsequently sorted.

The tasks **pappend**, **txappend**, and **tbappend** have been renamed to **pconcat**, **txconcat**, and **tbconcat**, respectively, but are otherwise unchanged.

Users who have not yet upgraded to IRAF 2.10.3BETA, or who are running IRAF on platforms for which IRAF 2.10.3BETA is not yet available, can acquire the updated version of **ptools** by retrieving and installing the external layered package **digiphotx** from our FTP archive.

Lindsey Davis

Aperture Correction Tasks Installed in the PHOTCAL Package

Two new tasks for computing aperture corrections, **mkapfile** and **apfile**, have been released with IRAF 2.10.3BETA. **mkapfile** and **apfile** compute the magnitude correction required to place the instrumental magnitudes of program stars, which were measured through a small aperture, on the same system as those of the standard stars, which were measured through a large aperture, using the growth-curve method (Stetson, P. B. 1990, PASP, 102, 932). This algorithm computes the growth-curve for each image by combining a best fit analytic model growth-curve with the observed growth-curve, using weights which favor the observed curve at small radii where the observational errors are small, and the model curve at large radii where the observational errors are large. The adopted curve is integrated between the desired aperture radii to produce the aperture correction for each image.

mkapfile and **apfile** can be run in non-interactive or interactive mode. In non-interactive mode the parameters are set at task startup time, and the fits for each image are computed sequentially. In interactive mode, the user can interact with the growth-curve fitting process by examining plots of the model fit, residuals versus aperture radius, magnitude in the first aperture, or x and y coordinates, and aperture correction as a function of radius. The user can also change the number of model parameters to be fit and their default values, delete and undelete points with the cursor, and refit and reexamine the fits until satisfied with the results.

mkapfile and **apfile** are installed in the **photcal** package, and write aperture correction files that can be read directly by the existing tasks in **photcal**, **mknobsfile**/**mkobsfile**/**obsfile**. The two

tasks differ only in the type of input they expect. **mkapfile** reads the aperture photometry files produced by the IRAF **apphot**/**daophot** packages, whereas **apfile** reads aperture photometry files in a simple text file format created by the user.

Users who have not yet upgraded to IRAF 2.10.3 BETA, or who are running IRAF on machines for which IRAF 2.10.3BETA is not yet available, can acquire the new tasks by retrieving and installing the external package **digiphotx** from the IRAF network archive.

Lindsey Davis

Measuring Stellar Image Widths for Seeing and Focus Characterization

One commonly measured quantity in direct imaging is the width of stellar (unresolved) images. This is used to characterize the instrumental focus and point-spread function (PSF) and the atmospheric seeing. The **imexamine** task is often used for this purpose with its Gaussian fitting to the radial profile of the pixels. This can give a good approximation to the width for well sampled images, but the algorithm suffers from several limitations such as model dependence and inaccuracies for very small images when the number of profile points is small. Also, when it is used for determining the best focus from a sequence of focus images it is left to the user to mentally interpolate to the best focus. A task specifically designed to give accurate characterization of the image width and to automatically determine the best focus from a focus sequence was recently developed.

The width measurement algorithm is model independent and measures the radial enclosed flux profile. It provides a more continuous measure of the light profile, and special techniques are used to provide accurate measures for marginally sampled or even undersampled data. It also accumulates measurements with a possible focus value assigned. The results can then be displayed and interpolated as a function of focus and space, to characterize the widths for either instrumental analysis or simply as an accurate and quick way to make a seeing measurement. A reference to the algorithms and user interface is given below.

The width measurement algorithm, display input, and data accumulation and display functions are packaged in three tasks which are simply specialized versions of the same program. The general task for measuring psf widths of stars in images is called **psfmeasure**. The options for interpolating focus sequences are not used here. Instead the task **starfocus** is used for focus sequences which are either sequences of images with different focuses or multiple shifted images at different focuses recorded in a single image. Finally, there is a specialized version of this for Kitt Peak National Observatory that uses information about the focus sequence recorded in the header to greatly automate the focus determination by simply pointing at one stellar image and having the task find, measure, interpolate and display the best focus. This simple script task is called **kpnofocus**. Except for the last task, the software is quite general and should be valuable at other observatories and for use with any imaging data.

These tasks are available in the external package **nmisc** (NOAO miscellaneous package) which can be obtained from *ftp://iraf.noao.edu/iraf/extern*. Papers describing the algorithms and user interface were published in the ADASS III Proceedings (ASP Conf. Series, Vol. 61), and an electronic version is available from *ftp://iraf.noao.edu/iraf/docs/psfmeasure.ps.Z*.

Frank Valdes

Image Conversions Using IMPORT/EXPORT

New tasks for converting between various external data formats and IRAF images have been written. These tasks are scheduled to appear in the V2.11 release of IRAF, but they are available now for early testing as the external package **imcnv**. The new tasks are the following:

```
import – Convert some other format to an IRAF image
export – Convert IRAF images to some other format
```

With these tasks users can now convert more than a dozen predefined image formats such as Sun rasterfile, GIF, VICAR, X Window Dumps, byte-swapped (nonstandard) FITS files, or IRAF pixel files, to IRAF images using the **import** task. Any other raster format image can be imported as well by specifying the format. For data export, the reverse conversion, and more, is possible using the **export** task.

Features of these tasks include:

import:

- support for user-defined binary format; converts 15 specific formats; identifies additional 9
- automatic sensing of image formats, dimensions, pixel types
- automatic conversion of color images to grayscale
- ability to separate colors (RGB or colormap) to different bands in an IRAF image
- byte-swap or IEEE conversion options
- arbitrary arithmetic expression evaluation
- extensible by the user to support new formats

export:

- support for user-defined binary format; writes 10 specific formats
- ability to output (color) Encapsulated PostScript from images
- automatic intensity scaling of images using either the **display** task "zscale" algorithm or **ximtool** brightness/contrast values
- automatic colormap computation from 3 input images
- 11 builtin colormap options; user-defined colormaps also supported
- single output image option for mosaic / composite multiple input images

The help pages contain many examples but one that may be most useful to astronomers is converting IRAF images for use in publications. In the simplest case this can be done using a command such as:

```
im> export dev$pix example eps outbands="zscale(i1)"
```

The image will be rendered with the same intensity scaling as one would get from the **display** task using the default parameters. The EPS file produced may be be included directly in any document (e.g., LaTeX) that supports inclusion of Encapsulated Postscript figures.

A more complex example would be if you wanted to include in your paper two adjacent images (e.g., the infrared and optical image of the same field with stars marked on the frame). To do this, with a small gap between the images:

```
im> export im1,im2 tmp iraf outb="zscale(i1)//repl(255,10)//zscale(i2)"
im> tvmark tmp coords
....interactively mark the objects you wish
....save the result to an image named "new"
im> export new fig1 eps
```

The file "fig1.eps" then has the images side-by-side with the objects marked.

Images may also be combined to form color output, for instance, if you have three images taken through B, V, and R filters and would like to combine them into a color rasterfile:

```
im> export Rim,Vim,Bim colorim ras outbands="i1,i2,i3"
```

When converting foreign formats for use within IRAF the **import** task, by default, will sense the size and format of each image to be converted. Colormapped images will automatically be converted to grayscale but there are always ways to separate the colors into separate image bands; RGB formats will generate a 3-D image. Users can extend the database of known formats individually, and scripts can easily be written to propagate, e.g., the header information from a VICAR format image.

A User's Guide is planned to more fully explain how these tasks may be used. Interested users may contact Mike Fitzpatrick (*fitz@noao.edu*) for information on package availability, questions, or suggestions for new formats to support.

Mike Fitzpatrick

IRAF Documentation

Most of the IRAF documentation is available in the IRAF network archive (in the directory *iraf/docs*) as compressed PostScript or compressed ASCII text files. This includes past issues of the IRAF Newsletter. The *iraf/docs/README* file contains a complete listing of what is available in this directory.

A number of new or updated IRAF documents or manuals have become available since the last issue of this Newsletter. These documents are listed below; the corresponding names of the files in the *iraf/docs* directory are also listed.

- *Cleaning Images of Bad Pixels and Cosmic Rays Using IRAF*, by Lisa A. Wells and David J. Bell, September 1994, 30 pages. (clean.ps.Z)
- Frequently-Asked-Questions List for IRAF. (FAQ)
- *A User's Guide to Reducing Echelle Spectra With IRAF*, by Daryl Willmarth and Jeannette Barnes, May 1994, 24 pages. (ech.ps.Z)
- *Rectifying and Registering Images Using IRAF*, by Lisa Wells, April 1994, 31 pages. (reg.ps.Z)
- *Photometry Using IRAF*, by Lisa A. Wells, February 1994, 12 pages. (photom.ps.Z)
- *A Reference Guide to the IRAF/DAOPHOT Package*, by Lindsey E. Davis, January 1994, 85 pages. (daorefman.ps.Z)
- *FOCAS User's Guide*, by Francisco Valdes, September 1993, 11 pages. (focas/focasguide.ps.Z)
- *A Beginner's Guide to Using IRAF, IRAF Version 2.10 (DRAFT)*, by Jeannette Barnes, August 1993, 65 pages. (NOTE: /pub/beguide.ps.Z)
- *VMS/IRAF Installation and Site Manager's Guide*, by Doug Tody, Suzanne Jacoby, and Nigel Sharp, June 1993 (for V2.10), 31 pages. (vmsiraf.ps.Z)
- *HPUX/IRAF Installation Guide*, by Steve Rooke, Doug Tody, and Mike Fitzpatrick, revised June 1993, 12 pages. (hpuxiraf.ps.Z)
- *Macintosh A/UX-IRAF Installation Guide*, by Doug Tody, April 1993, 12 pages. (auxiraf.ps.Z)

- *IBM AIX/IRAF Installation Guide*, by Doug Tody, revised November 1992, 12 pages. (aixiraf.ps.Z)
- *Table of Contents for IRAF Newsletters*, by Jeannette Barnes. (TOC_news.txt)
- *An Introductory User's Guide to IRAF SPP Programming*, by Rob Seaman, October 1992, 76 pages. (sppguide.ps.Z)
- *An SPP/VOS Quick Reference Card*, by Rob Seaman, October 1992, 2 pages. (quickref.ps.Z)
- *VAX Ultrix/IRAF Installation Guide*, by Mike Fitzpatrick and Doug Tody, revised September 1992, 12 pages. (vxuxiraf.ps.Z)
- *Decstation Ultrix/IRAF Installation Guide*, by Doug Tody, revised September 1992, 12 pages. (dsuxiraf.ps.Z)
- *UNIX/IRAF Site Manager's Guide*, by Doug Tody, revised September 1992, 22 pages. (unixsmg.ps.Z)
- *IRIX/IRAF Installation Guide*, by Steve Rooke, Mike Fitzpatrick, and Doug Tody, revised July 1992, 12 pages. (irixiraf.ps.Z)
- *A User's Guide to CCD Reductions with IRAF*, by Philip Massey, June 1992, 55 pages. (ccduser2.ps.Z)

Please contact us if you would like to receive printed copies of any of the IRAF documents (*iraf@noao.edu*).

Jeannette Barnes

Layered Software Available for IRAF Versions 2.8/2.9/2.10

The following software packages are available from NOAO for optional installation in IRAF Versions 2.8, 2.9, and in some cases 2.10 (many additional layered packages not discussed here are available from outside NOAO). Note that some of these packages are included in V2.10 or later releases and so there is no need to install them if you already have a later version of IRAF. All packages are available via anonymous FTP from the IRAF network archive on *iraf.noao.edu* (Internet node 140.252.1.1) in the directory *iraf/extern* and have *readme* files containing instructions for transfer and installation. (See the accompanying article in this Newsletter about the reorganization of the directory structure for these packages.) Unless otherwise specified please contact the IRAF hotline for further information (*iraf@noao.edu*).

- ADC CD-ROM utility package - two IRAF tasks that extract data from Volume I of the ADC CD-ROM text version. See the article in IRAF Newsletter Number 12 (July 1992).
- CTIO - a package of miscellaneous IRAF tasks developed at CTIO. Contact Pedro Gigoux (*pgigoux@noao.edu*) for further information.
- COLOR - a prototype IRAF color image display package that provides conversion of three bandpass IRAF images to a Sun 24-bit RGB rasterfile format, a 24-bit to 8-bit compression algorithm and Floyd-Steinberg dithering, and an RGB 8-bit pixel dithering algorithm. These tasks allow rendering of three color images for display on common 8-bit color workstations.
- DIGIPHOTX - a new version of the **digiphot** package. This is included in IRAF Version 2.10.3BETA and later versions, but can be installed for use with other platforms prior to the release of V2.11. See accompanying articles in this Newsletter.
- FOCAS - a suite of programs for automatic detection, photometry, matching, classification, and cataloging of astronomical digital images. It also provides tools for accessing the catalogs and doing analysis. This is a collection of C programs that can be run in the IRAF environment or outside it as a standalone system (Unix platforms only).

Contact Frank Valdes (*fvaldes@noao.edu*) if you are interested in running FOCAS on a VMS platform.

- IMCNV - two new tasks for converting between various external data formats and IRAF images. These tasks, **import** and **export**, are scheduled to appear in the V2.11 release, but are made available now for early testing. See the accompanying article in this Newsletter.
- NMISC - miscellaneous new tasks from NOAO collected together in a layered package to make them available prior to their release in the main IRAF distribution. This package currently contains the following tasks:

kpnofocus	--	Determine the best focus from KPNO focus images
psfmeasure	--	Measure PSF sizes from stellar images
surfit	--	Fit a surface, z=f(x,y), to a set of x, y, z points
specfocus	--	Determine spectral focus and alignment variations
starfocus	--	Determine direct focus variations from stellar images
xregister	--	Register 1 and 2 D images using x-correlation techniques

- RVX - the IRAF radial velocity analysis package. This is included in IRAF V2.10.3BETA and later versions of IRAF but is also available as a layered package for users with older versions of IRAF, or who want the latest version incorporating any bug fixes.
- SAOimage - an X Window System based image display server for IRAF developed originally by SAO (see article in IRAF Newsletter Number 8, October 1989).
- SPPTOOLS - a collection of programming tools for IRAF developers working in SPP. Included are tasks to locate and print the calling sequences of procedures, reformat code in a standard format, create or query identifier databases, and create or rename external packages. Of special interest is the **spplint** task that can be used to perform a Lint-like compile time verification of SPP code. For further information contact Mike Fitzpatrick (*fitz@noao.edu*).
- Volume rendering software - software to visualize 3D data cubes. This software has been discussed in previous issues of the IRAF Newsletter, i.e., Number 5 (October 1988) and Number 6 (February 1989).
- Kernel server kits - this software may be installed on a host system to allow the host to be remotely accessed by IRAF sessions running on other hosts. This is used, for example, to remotely access tape drives or the workstation display. Note that if IRAF is already installed on another node in your local network which is architecturally compatible and accessible via NFS, it may be simpler to NFS mount IRAF than to install the kernel server kit.
- UISDISP display software for non-DECwindows VMS Workstations - this software was discussed in IRAF Newsletter Number 7 (June 1989). It is included with VMS/IRAF version 2.10. A recent update with some small bug fixes is available from the *contrib* directory in the IRAF archives. For further information please contact Nigel Sharp (*nsharp@noao.edu, 5355::nsharp*).

Additional layered packages from outside NOAO are also available in the *contrib* directory. The software in this directory is contributed by persons outside the IRAF group, and the author or authors are solely responsible for the software. Users with IRAF-related software they would like to share are encouraged to install their software to this directory (the directory is world writable). Even in cases where a software product is maintained and distributed elsewhere we try to include an entry in *contrib* pointing to the external software, to make it easier for people to learn about and locate the software.

The IRAF Group

Brief Summary of Application Software Modifications in IRAF V2.10.3BETA

We'll close this Newsletter with a very brief summary of the software modifications in IRAF V2.10.3BETA. Numerous tasks have been updated with bug fixes and new features, and several other tasks are completely new. There's not enough room to go through everything here, so we'll just try to hit a few highlights. There will be more extensive revisions notes for V2.11.

There is a new spectral format, *equispec*, which is useful for the common case in which all spectra in a multispec image have the same linear dispersion relation (be aware that spectra of this type are not compatible with earlier IRAF versions). The **digiphot** packages have been redone, and **daophot** now uses Stetson's DAOPHOT II algorithms. The line fitting algorithms in **splot** and **fitprofs** now provide error estimates. The **cosmicrays** task can now interact with the display device to allow users to select cosmic rays visually. A powerful new task (**imexpr**) allows far more sophisticated image expressions than are possible with **imarith**. Other new tasks include one to register images with cross-correlation (**xregister**), tasks to convert text files to / from spectra (**rspectext**, **wspectext**), and tasks to measure radial velocities from spectral lines (**rvidlines**, **rvreidlines**). See accompanying articles in this Newsletter for further details about many of these new tasks.

The following tasks are new (or resurrected or renamed) in V2.10.3BETA:

imexpr	(images)	-	Evaluate a general image expression
xregister	(images)	-	Register 1-D or 2-D images using x-correlation techniques
surfit	(utilities)	-	Fit a surface, z=f(x,y), to a set of x, y, z points
asthedit	(astutil)	-	Astronomical header editor
keywpars	(astutil)	-	Translate the image header keywords used in **astutil** package (pset)
daoedit	(daophot)	-	Review/edit algorithm parameters interactively
findpars	(daophot)	-	Edit the star detection parameters (pset)
setimpars	(daophot)	-	Save/restore parameter sets for a particular image
pcalc	(daophot)	-	Do arithmetic operations on a list of daophot databases
pconcat	(daophot)	-	Concatenate a list of daophot databases (was pappend)
pfmerge	(daophot)	-	Merge a list of photometry databases
pstselect	(daophot)	-	Select candidate psf stars based on proximity
mkapfile	(photcal)	-	Prepare aperture corrections file from apphot/daophot output
mkphotcors	(photcal)	-	Prepare the photometric corrections files
apfile	(photcal)	-	Prepare an aperture corrections file from a text file
pconcat	(ptools)	-	Concatenate a list of daophot databases (was pappend)
pcalc	(ptools)	-	Do arithmetic operations on a list of daophot databases
tbconcat	(ptools)	-	Concatenate a list of apphot/daophot tables databases (was tbappend)
tbcalc	(ptools)	-	Do arithmetic on a list of apphot/daophot tables databases
txconcat	(ptools)	-	Concatenate a list of apphot/daophot text databases (was txappend)
txcalc	(ptools)	-	Do arithmetic on a list of apphot/daophot text databases
disptrans	(onedspec)	-	Transform dispersion units and apply air correction
rspectext	(onedspec)	-	Convert ascii text spectra to image spectra

sbands	(onedspec)	-	Bandpass spectrophotometry of spectra
sflip	(onedspec)	-	Flip data and/or dispersion coordinates in spectra
specshift	(onedspec)	-	Shift spectral dispersion coordinate systems
wspectext	(onedspec)	-	Convert 1D image spectra to ascii text spectra
rvidlines	(rv)	-	Measure radial velocities from spectral lines
rvreidlines	(rv)	-	Reidentify spectral lines and measure radial velocities
apnoise	(apextract)	-	Compute and examine noise characteristics of spectra

Dave Bell