User's Manual for DAOPHOT II

This manual is intended as a guide to the use of the digital stellar photometry reduction program DAOPHOT II: The Next Generation. Brief descriptions of the major routines are provided to aid the user in the use of the program, and to serve as an introduction to the algorithms for programmers called upon to make modifications. A more complete understanding of the methods used can best be obtained by direct examination of the source code and the comment statements embedded therein.

DAOPHOT Classic was originally constructed within the framework of a computer program given to Linda Stryker and the Dominion Astrophysical Observatory by Jeremy Mould (Caltech). The aperture-photometry portion of the current program is still somewhat influenced by algorithms and subroutines developed at Kitt Peak National Observatory, by way of the POORMAN code developed by Jeremy Mould and Keith Shortridge at Caltech. POORMAN was first made to run on the DAO VAX by Linda Stryker, and was modified and generalized by Ed Olszewski. Over the years, I have replaced all of the subroutines with others of my own writing. In DAOPHOT II: The Next Generation all of the major algorithms and actual FORTRAN coding are my own, with the exception of those Figaro, IRAF, or Midas routines which are used for the image input and output. I am solely responsible for bugs and algorithm-related problems. If any such problems are found, please contact me; for major problems, hard-copy documentation of the circumstances and nature of the difficulty would be highly desirable.

***** NOTE *****

This manual has been modified to reflect the new VMS/Unix IRAF/Midas compatible version of DAOPHOT II: The Next Generation. If you are still running DAOPHOT Classic you will find many places where this manual is inappropriate.

2000 June 5

Peter B. Stetson

(604)363–0029 stetson@dao.nrc.ca

Dominion Astrophysical Observatory

Herzberg Institute of Astrophysics

5071 West Saanich Road

Victoria, British Columbia V8X 4M6

Canada

Contents

A. Introduction

DAOPHOT II is a computer program for obtaining precise photometric indices and astrometric positions for stellar objects in two-dimensional digital images. It is intended to run as non-interactively as possible and, furthermore, the possibility that DAOPHOT II would be used at other places than the DAO was kept in mind as it was approaching its present form. Therefore DAOPHOT II performs no operations related to the display or manipulation of the digital image on an image-display system, even though at some stages in the data reduction it is useful to be able to examine the picture visually. Picture-display operations and some other steps in the reduction procedure, such as editing intermediate data files or combining results from different frames to obtain instrumental colors, may be done outside of DAOPHOT II using IRAF, Midas, or whatever software you have handy or feel like writing.

It is assumed that (1) before running DAOPHOT II, the user will have performed all necessary preparation of the images, such as flat-fielding, bias-level subtraction, and trimming worthless rows and columns from around the perimeter of the picture, and (2) the brightness data in the image are linearly related to true intensities. The user is also assumed to have *a priori* knowledge of the following pieces of information: (1) the approximate size (full-width at half-maximum) of unresolved stellar objects in the frame; (2) the number of photons corresponding to one analog-to-digital conversion unit; (3) the readout noise per pixel; and (4) the maximum brightness level (in analog-to-digital units) at which the detector still operates linearly. These conditions being satisfied, DAOPHOT II will perform the following primary tasks: (1) find star-like objects above a certain detection threshold, rejecting with a certain degree of reliability bad pixels, rows, and columns, and avoiding multiple hits on individual bright objects (although it continues to have some trouble with grossly saturated objects. I'm still thinking about it.); (2) derive concentric aperture photometry for these objects, estimating a local sky brightness for each star from a surrounding annulus of pixels; (3) obtain a point-spread function for the frame from one star or from the average of several stars, in an iterative procedure intended to fit and subtract faint neighbor stars which contaminate the profile; (4) compute precise positions and magnitudes for the program stars by fitting the point-spread function to each star, either individually or in simultaneous multiple-profile fits for up to 60 stars at a time; and (5) erase stars from the picture by subtracting appropriately scaled point-spread functions corresponding to the positions and magnitudes derived for the stars during the photometric reductions. ALLSTAR II is a separate stand-alone program which performs a much more sophisticated multiple-profile fit to all the stars in a frame simultaneously. Hereinafter I will include ALLSTAR II under the generic heading of DAOPHOT II even though it is, as I said, a separate, stand-alone program. In addition to the aforementioned tasks, DAOPHOT II contains routines to perform some bookkeeping operations more easily than may be the case with standard facilities: e.g., estimating an average sky brightness for a frame, sorting the stars' output data according to their positions in the frame or their apparent magnitudes, and dividing the stars in the frame into natural groupings (for optimal multiple-star reductions with NSTAR). There is also a routine for adding artificial stars to the picture, at random, so that the effectiveness of the star-finding and profile-fitting routines can be studied quantitatively in the particular circumstances of your own picture. A few other global considerations of which you should be aware:

(1) Although DAOPHOT II is designed to be non-interactive, in fact many of the operations run

quickly enough that they are conveniently executed directly from the terminal or workstation. Only the multiple-star profile fits take long enough that they are more conveniently performed in batch mode: they may require anywhere from a few CPU minutes to a few CPU hours per frame, depending upon the number of stars to be reduced, the degree of crowding, and — of course — the speed of your machine. If computer time is expensive for you, you may want to decide just how many stars must be reduced in order to fulfill your basic mission. For instance, if your goal is to locate the principal sequences of some remote cluster, it may not be necessary to reduce every last crowded star on the frame; instead, only a subset consisting of the less-crowded stars might be reduced.

(2) The derivation of the point-spread function can also be performed non-interactively with a reasonable degree of success, but it may be to your advantage to check the quality of the profile fits visually on an image display before accepting the final product.

(3) The shape of the point-spread function is assumed to be spatially constant or to vary smoothly with position within the frame; it is assumed *not* to depend at all on apparent magnitude. If these conditions are not met, systematic errors may result.

(4) Although the star-finding algorithm is by itself not sophisticated enough to separate badly blended images (two stars whose centers are separated by significantly less than one FWHM), by iteratively substracting the known stars and searching for fainter companions, it is still possible to identify the separate stars in such a case with a good degree of reliability: First, one runs the star-finding algorithm, derives aperture magnitudes and local sky values for the objects just found, and obtains a point-spread function in the manner described in an Appendix to this manual. Second, one performs a profile-fitting reduction run for these objects, and they are subtracted from the data frame. This new picture, with the known stars subtracted out, is then subjected to the star-finding procedure; stars which were previously concealed in the profiles of brighter stars stand out in this frame, and are picked up quite effectively by the star-finding algorithm. Sky values and aperture magnitudes for these new stars are obtained from the original data frame, and the output from this reduction is appended to the most recent photometry file for the original star list. This augmented set of stars is then run through the profile-fitting code, and the entire list of fitted stars can be subtracted from the original frame. The process through this point can *easily* be set up in a command procedure (or script, or whatever your favorite rubrik is), and carried out in batch mode while you are home sleeping or drinking beer or whatever. Finally, if absolute completeness is wanted, the star-subtracted picture can be examined on an image display. Any stars that were still undiscovered by the program can be picked out by eye and added to the star list manually. Then one final reduction run may be performed. Visual examination is also a reasonable way to identify galaxies among the program objects — they are easily recognizable, with over-subtracted centers surrounded by luminous fuzz.

My experience is that the number of stars found in the second pass (the automatic star-finding on the first subtracted frame) amounts to of order one-third the number of stars found in the first pass. The number of stars missed in the first two passes and later picked out by eye is of order one to three percent of the total found in the first two passes. This procedure assumes that computer time is cheap for you, and your own time is valuable. If the converse is the case, you may prefer to skip the second or even both automatic star-finding passes, and go directly to interactive star identification.

(5) A principal source of photometric error for the faint stars is the difficulty of defining what is meant by the term "sky brightness" in crowded fields. This is not simply the practical difficulty of identifying "contaminated" pixels in the sky annulus so that they can be omitted from the average, although certainly this is a significant part of the problem. There is also an underlying philosophical ambiguity. For aperture photometry the term "sky brightness" encompasses not only emission from the terrestrial night sky, from diffuse interplanetary and interstellar material, and from faint, unresolved stars and galaxies. It also includes the possibility of a contribution of light from some bright star or galaxy. That is to say, for aperture photometry the relevant sky brightness is defined by the answer to the question, "If the *particular star* we are interested in were not in the center of this aperture, what intensity would we measure there from all other sources of brightness on or near this line of sight?" If there is available a set of pixels whose brightness values are uncontaminated by the star we are trying to measure, but which are subject to all other sources of emission characteristic of that portion of the frame (including the possibility of a contribution from individual bright objects nearby), then we may answer the question: "Most probably the intensity would be such-and-such". The specific value "such-and-such" is well predicted by the modal value of the brightnesses in the sample of sky pixels. This is why DAOPHOT II uses the mode of the intensities within the sky annulus to define the value that should be subtracted from the intensities inside the star aperture; not because it is a "robust estimator of the local diffuse sky brightness", but because it is a sort of maximum-likelihood estimator — it yields the "most probable value of the brightness of a randomly-chosen pixel in this region of the picture". In the case of photometry from multiple simultaneous profile fits, on the other hand, the sky brightness is defined by the answer to a different question altogether: "If *none* of the stars included in my star list were in this region of the picture, what would the typical brightness be?" This is a much harder question to answer, partly because the answer cannot be obtained by an empirical method as simple as finding the mode of an observed distribution, and partly because the criteria for including a star in the star list are hard to define absolutely. For instance, a faint star is far easier to see in a clear-sky zone, where its contamination can be identified and then ignored in the sky-brightness estimate, than it would be if it happened to lie near a much brighter star whose intrinsic brightness we are trying to measure. Clearly then, when we detect a faint star in the sample sky region, the decision whether to include or reject that star's photons in the sky estimate becomes some function of the magnitude of the star we are interested in measuring. Further, a serious attempt to estimate the sky brightness using probabilistic methods would require an accurate model for the full noise spectrum of the instrument, including the arrival rate and energy spectrum of cosmic rays, the surface density of stars and galaxies on the sky, their apparent luminosity functions, and the variations of these quantities across the frame. Thus, a definitive answer to the question "How bright is the sky here?" is exceedingly hard to obtain with full statistical rigor. For the present we must be content with a merely adequate answer.

I have discussed the problem of sky determination in such philosophical detail as a warning to the user not to regard DAOPHOT II (or any other program of which I am aware) as *the* final solution to the problem of accurate stellar photometry in crowded fields. As it stands now, the aperture photometry routine is the only place in DAOPHOT II proper where sky brightness values are estimated; these estimates are based on the modal values observed in annuli around the stars, and they are carried along for use by PEAK and NSTAR. ALLSTAR II has the capability (as an *option*) of iteratively *re*-determining the sky brightness value for each star, defined as the median value found in pixels surrounding the star *after* all known stars have been subtracted from the frame using the current, provisional estimates of their position and brightness. These sky brightnesses are *assumed* to be adequate for the purposes of defining and fitting point-spread functions, but of course this is only approximately true. The extent to which this false assumption affects the observational results and the astrophysical conclusions which you derive from your frames can best be estimated, at present, by the addition of artificial stars of known properties to your pictures, with subsequent identification and reduction by procedures identical to those used for the program stars.

B. A Typical Run of DAOPHOT II

Before you run DAOPHOT II: The Next Generation you must arrange for your pictures to exist on the computer's disk in a format acceptable to the program. On the DAO VAXen the standard format for images intended for processing with DAOPHOT is the Caltech data-structure (.DST) file, and routines exist for copying data from a number of different formats, including FITS, to this type of file. On our Unix machines we use IRAF for image manipulation and IRAF image files (*.imh and *.pix) for image storage. At ESO there exists a version of DAOPHOT II that operates on Midas-format image files on a variety of hardwares. If you don't happen to be at the DAO, then you will have to check with your local curator of DAOPHOT II to learn how to put your data into the proper format. In all that follows, I shall assume that you are either at DAO, or are using an unadulterated DAO/VMS/Unix version of DAOPHOT II. See your local curator for changes that are specific to your facility.

I will now talk you quickly through the separate steps in reducing a typical data frame, from beginning to end. I suggest that you read quickly through this section and the following chapters on the major and minor routines in DAOPHOT II, and then come back and reread this section more carefully. Words written in boldface **CAPITALS** will be DAOPHOT II commands, which you may issue in response to a "Command:" prompt.

I. From a system prompt run DAOPHOT II. Either read or don't read the latest news (if it's even offered to you). The values of certain user-definable parameters will be typed out. *Check their values! You might want to change some of them.*

II. Use **OPTIONS** to change the values of any of the optional reduction parameters. (This step is, itself, optional.)

III. Use **ATTACH** to tell the program which picture you want to reduce. (In the VMS version, you do not need to include the filename-extension, .DST, when you specify the filename, but you may if you like. In the Unix IRAF version, your life will be simpler if you *do not* include the .imh extension.)

IV. You might want to use **SKY** to obtain an estimate of the average sky brightness in the picture. *Write this number down in your notes.* This step is not really necessary, because **FIND** below will do it anyway.

V. Use **FIND** to identify and compute approximate centroids for small, luminous objects in the picture. One of the "user-definable optional parameters" which you are permitted to define is the significance level, in standard deviations, of a luminosity enhancement in your image which is to be regarded as real. Two other parameters which you *must* define are the readout noise and gain in photons (or electrons) per data number which are appropriate to a *single* exposure with your detector. When you run **FIND**, it will ask you whether this particular image is the average or the sum of several individual exposures. From the available information, **FIND** will then compute the actual brightness enhancement, in data numbers above the *local* sky brightness, which corresponds to the significance level you have specified. See the section on **FIND** and the Appendix on "The **FIND** Threshold" for further details. According to a parameter set by the user, **FIND** will also compute a "Lowest good data-value": any pixel whose brightness value is less than some number of standard deviations below the mean sky value will be regarded as bad, and will be ignored by **FIND** and by all subsequent reduction stages.

VI. Use **PHOTOMETRY** to obtain sky values and concentric aperture photometry for all objects found by the star-finding routine.

VII. Use **PICK** to select a set of reasonable candidates for PSF stars. **PICK** first sorts the stars by magnitude, and then rejects any stars that are too close to the edge of the frame or to a brighter star. It will then write a user-specified number of good candidates to a disk file for use by **PSF**.

VIII. Use **PSF** to define a point-spread function for the frame. In crowded fields this is a subtle, iterative procedure requiring an image processing system; it is outlined in detail in the Appendix on "Obtaining a Point-Spread Function". Consider, then, that this step is a self-contained loop which you will go through several times.

IX. **GROUP**, **NSTAR**, and **SUBSTAR**; *or* ALLSTAR. **GROUP** divides the stars in the aperture-photometry file created in step VI above into finely divided "natural" groups for reduction with the multiple-star PSF-fitting algorithm, **NSTAR**. **NSTAR** will then produce improved positions and instrumental magnitudes by means of multiple-profile fits, and **SUBSTAR** may then be used to subtract the fitted profiles from the image, producing a new image containing the fitting residuals. Alternatively, you could feed the aperture-photometry file directly to ALLSTAR, which will reduce all the stars in the image simultaneously and produce the star-subtracted picture without further ado.

X. Use **ATTACH** to specify the star-subtracted picture created in step IX as the one to work on.

XI. Use **FIND** to locate new stars which have become visible now that all the previously-known stars have been subtracted out.

XII. Use **ATTACH** again, this time specifying the *original* picture as the one to work with, and use **PHOTOMETRY** to obtain sky values and crude aperture photometry for the newly-found stars, using the coordinates obtained in step XI. (You are performing this operation on the original picture so that the sky estimates will be consistent with the sky estimates obtained for the original star list.)

XIII. Use **GROUP** on the new aperture-photometry file you just created. Use **GROUP** again on the profile-fitting photometry file created in step IX (this step is unfortunately necessary to put both the old and new photometry into files with the same format, so that you can ...). Use **APPEND** to combine the two group files just created into one.

XIV. **GROUP** + **SELECT** + **SELECT** + **GROUP** + **SELECT** + **SELECT** + ... + **NSTAR** + **SUBSTAR**, *or* ALLSTAR: If for some reason you prefer **NSTAR** to ALLSTAR (I sure don't), the file just created in step XIII needs to be run through **GROUP** once again to sort the combined starlist into the best groupings for the next pass of **NSTAR**. Watch the table of group sizes that gets created on your terminal very carefully. The multiple-PSF fitting routine is at present capable of fitting no more than 60 stars at a time. If any of the groups created by **GROUP** is larger than 60 stars, the SELECT command can be used to pick out only those groups within a certain range of sizes. You would run

(1) **SELECT** once to pick out those groups containing from 1 to 60 stars, putting them in their own file. You could discard all groups larger than 60 if you only wanted a representative, as distinguished from a complete, sample. Alternatively, you could run

(2) **SELECT** again to pick out those groups containing 61 or more stars, putting them in their own file. Then you would run

(3) **GROUP** with a *larger* critical overlap on the file created in (2), to produce a new group file with smaller groups. *The photometry for these stars will be poorer than the photometry for the less-crowded stars picked out in XIV-1.*

Return to (1) and repeat until (a) all stars are in groups containing less than or equal to 60 stars, or (b) (preferred, and cheaper) enough stars are in groups smaller than 60 that you feel you can perform your basic astronomical mission. *Then,*

(4) **NSTAR** as many times as necessary to reduce the group files just created, and

(5) **SUBSTAR** as many times as necessary to subtract the stars just reduced from the data frame.

Or, you could get around the whole thing just by running the **APPEND**ed group file through ALLSTAR.

XV. **EXIT** from DAOPHOT II. Display the star-subtracted picture created in step XIV on your image-display system. Look for stars that have been missed, and for galaxies and bad pixels that have been found and reduced as if they were stars. If desired, create a file containing the coordinates of stars you wish to add to the solution and run **PHOTOMETRY** on these coordinates. To make one more pass through the data, you should run this aperture photometry file through **GROUP**, run the previous **NSTAR** or ALLSTAR results through **GROUP** (again, necessary in order to get both the old and new stars into files with the same format), **APPEND** these files together, and return to step XIV. *Repeat as many times as you like, and have the time for.*

XVI. **EXIT** from DAOPHOT and examine your picture on the image-display system. Choose several minimally-crowded, bright, unsaturated stars. Make a copy of the output file from your very last run of **NSTAR** or ALLSTAR and, with a text editor, delete from this file the data lines for those bright stars which you have just chosen. Run DAOPHOT, **ATTACH** your original picture and invoke **SUBSTAR** to subtract from your picture all the stars remaining in the edited data file. With equal ease, you can also create a file containing *only* the stars you want to retain — you could even use the file containing the list of PSF stars — and you can tell **SUBSTAR** to subtract all the stars from the image *except* the ones listed here. In either case, the stars which you have chosen will now be completely alone and uncrowded in this new picture — measure them with the aperture **PHOTOMETRY** routine, using apertures with a range of sizes up to very large. These data will serve to establish the absolute photometric zero-point of your image.

***** NOTE *****

This has been the reduction procedure for a program field, assumed to contain some hundreds or thousands of stars, most of which you are potentially interested in. The reduction procedure for a standard-star frame, which may contain only some tens of objects, only a few of which you are interested in, may be different. It may be that you will want to run **FIND** on these frames, and later on match up the stars found with the ones you want. Or perhaps you would rather examine these frames on the image-display system, use the cursor to measure the coordinates of the stars you are interested in, and create your own coordinate file for input into **PHOTOMETRY** (step VI). In any case, for your standard fields it is possible that you won't bother with profile fits, but will just use the aperture photometry (employing a growth-curve analysis) to define the stars' instrumental magnitudes.

C. Descriptions of Main Routines
(In approximate order of use)

I. DAOPHOT II itself

When you run DAOPHOT II, the first thing that may occur is the typing out on your terminal of a brief message, notifying you that some information to your advantage is now available. If this message has changed since the last time you ran the program, answer the question with a capital or lower-case "Y<CR>". The program will then type out the text of the entire message, a section at a time. It will pause at the end of each section of the message to allow you to read what it's written before it rolls off the screen, or to escape to the main program without reading the rest of the message. When you reach the main part of the program, the current values of the optional reduction parameters will appear on your screen (see the Appendix on Optional Parameters, and the **OPTIONS** command below). When you see the "Command:" prompt, the program is ready to accept the commands described below.

II. **ATTACH**

If you want to work on a digital picture, the first thing you should do is specify the disk filename of that picture with the ATTACH command:

```
================================================================
| COMPUTER TYPES:                                YOU ENTER:    |
|                                                              |
| Command:                                       AT filename   |
|                                                              |
|      Your picture's header comment (if any)                  |
|                                                              |
|                  Picture size:   nnn  nnn                    |
+--------------------------------------------------------------+
| or,                                                          |
|                                                              |
| COMPUTER TYPES:                                YOU ENTER:    |
|                                                              |
| Command:                                       AT            |
|                                                              |
|               Enter file name:                  filename      |
|                                                              |
|      Your picture's header comment (if any)                  |
|                                                              |
|                  Picture size:   nnn  nnn                    |
================================================================
```

Some commands, the ones which operate only on data files, (*e.g.* **SORT**, **OFFSET**, **APPEND**), and others which set the optional parameters (**OPTIONS**, **MONITOR**, **NOMONITOR**) may be issued to DAOPHOT II without a prior **ATTACH** command having been given. The program will refuse to let you tell it to perform tasks requiring a picture file (e.g., **FIND**, **PHOTOMETRY**, **PSF**, **GROUP**, **NSTAR**) unless a picture has been **ATTACH**ed.

In all implementations of DAOPHOT II of which I am aware, if the extension part of your picture's filename is the standard one for that image format ('.DST' for Caltech data structures, '.imh' for IRAF, '.bdf' for Midas) it may be omitted. If it is not the standard extension (or, on VMS machines, if you wish to specify other than the most recent version of the image file), the filename-extension must be included in the **ATTACH** command.

The **ATTACH** command is the only one in DAOPHOT II which allows the user to include additional information (*viz.* a filename) on the command line. All other commands are issued simply and without modification — the routines will then prompt the user for any necessary input.

This is also a good time to point out that DAOPHOT II commands are *case insensitive*: commands and parameter options (see below) may be entered using either upper or lower case letters, even on Unix machines. On VMS machines filenames are also case insensitive: FILE.EXT and file.ext refer to the same file. On Unix machines FILE.EXT, file.ext, FILE.ext, FiLe.ExT, etc. are all different. Finally, for Unix afficionados, I have taught DAOPHOT II to recognize a string of characters terminated with a colon (“:”) at the beginning of a filename as a *directory* name, à la VMS. Thus, if in your .cshrc file or some similar location, you have a statement like

```
setenv ccd /scr/nebuchadnezzar/mountstromloandsidingspring/1989/ccd-data
```

then while running DAOPHOT II in some other directory, you can refer to an image or other file, obs137, in *this* directory as ccd:obs137. In fact, I recommend that you do so, because all file names used by DAOPHOT are limited to 30 characters. Finally, on the VMS side you can create multiple versions of the same filename ad libitum, but Unix doesn’t allow it. Therefore, on the Unix side, if you try to create a file with the same name as one that already exists, DAOPHOT II will type out a warning and give you an opportunity to enter a new filename. If you respond to the prompt with a simple carriage return, the pre-existing file will be deleted before the new one is created. If you write command procedures to handle your DAOPHOT II reductions (I know I do), I recommend that you include in your procedures Unix “rm” commands to remove any files you know you’re going to be creating. Otherwise you run the risk of getting unexpected prompts asking whether you want to overwrite pre-existing files, and your list of commands and filenames can get out of synch with what the program is asking for. Chaos could result.

III. **OPTIONS**

Numerous parameters which optimize the reduction code for the specific properties of your picture may be altered. The sigificance of each one is described below in reference to the particular routines affected, and a reference table is presented as an Appendix. Here the options will be simply enumerated, and the procedures available for changing the parameters will be described.

At present the user is permitted to specify values for nineteen parameters:

(1) "READ NOISE": The readout noise, in data numbers, of a *single* exposure made with your detector. Later on, the software will allow you to specify whether a given data frame is in fact the sum or the average of several individual exposures. If you have a separate subdirectory for data from a given night or observing run, the readout noise needs to be specified only once (in the DAOPHOT.OPT file, see below).

(2) "GAIN": The gain factor of your detector, in photons or electrons per data number. As with the readout noise, you want to specify the gain corresponding to a *single* exposure; allowance can be made later for frames that are in fact the averages or sums of several exposures.

For both READ NOISE and GAIN the default values are **deliberately invalid**. *You must put correct values for these parameters in a DAOPHOT.OPT file, or the program will hassle you again, and again, and again, and ...*

(3) "LOW GOOD DATUM": The level, in standard deviations below the frame's mean sky value, below which you want the program to consider a pixel defective. If the background is flat across the frame, then you can set a tight limit: maybe 5σ or so. If there is a strong background gradient, you will need to set a more generous limit — maybe 10σ or more — to keep legitimate sky pixels from being rejected in those parts of the frame where the background is faint. Intelligent use of the **DUMP** command and/or an image display will help you decide what to do.

(4) "HIGH GOOD DATUM": The level, in data numbers, above which a pixel value is to be considered defective. Note that this differs from the "LOW GOOD DATUM" just discussed. The "LOW GOOD DATUM" is defined as a certain number of *standard deviations it below a frame's mean sky value*. Thus, assuming that all your frames have comparably flat backgrounds, it needs to be specified only once; the actual numerical value used will ride up and down as frames with different mean background levels are considered. The "HIGH GOOD DATUM" is specified as a single, fixed number which represents the absolute level in data numbers at which the detector becomes non-linear or saturates. (Note that since your data have been bias-level subtracted and flat-fielded, this number will *not* be 32767, but will be somewhat lower.)

(5) "FWHM": The approximate FWHM, in pixels, of the objects for which the **FIND** algorithm is to be optimized. This parameter determines the width of the Gaussian function and the size of the array with which your picture is numerically convolved by **FIND** (see detailed discussion under **FIND** command below). If conditions during your run were *reasonably* constant, a single value should be adequate for all your frames.

(6) "THRESHOLD": The significance level, in standard deviations, that you want the program to use in deciding whether a given positive brightness enhancement is real. Normally, somewhere

around 4σ is good, but you may want to set it a little higher for the first pass. Then again, maybe not.

(7), (8) "LOW" and "HIGH SHARPNESS CUTOFF": Minimum and maximum values for the sharpness of a brightness enhancement which **FIND** is to regard as a real star (intended to eliminate bad pixels; may also help to eliminate low-surface-brightness galaxies). In most cases the default values given in the program are adequate, but if you want to fine-tune them, here they are.

(9), (10) "LOW" and "HIGH ROUNDNESS CUTOFF": Minimum and maximum values for the roundness of a brightness enhancement which **FIND** is to regard as a real star (intended to eliminate bad rows and columns, may also reject some edge-on galaxies). Again, I think you'll find that the program-assigned default values are adequate for all but special cases.

(11) "WATCH PROGRESS": Whether to display results on the computer terminal in real time as they are computed. Displaying the results may keep you entertained as the reductions proceed, but it may slow down the execution time, and in batch mode, it will fill up your logfile excessively.

(12) "FITTING RADIUS": Most obviously, this parameter defines the circular area within which pixels will actually be used in performing the profile fits in **PEAK** and **NSTAR**: As the point-spread function is shifted and scaled to determine the position and brightness of each of your program stars, only those pixels within one fitting radius of the centroid will actually be used in the fit. More subtly, the same region is also used in fitting the analytic first approximation to the point-spread function for the PSF stars. Moreover, the parameter will also contribute in a minor way to the determination of when stars overlap "significantly." Under normal circumstances, this radius should be of order twice the half-width at half-maximum of a stellar image which is, obviously, the same as the FWHM. When the crowding is extremely severe, however, it may be advantageous to use a value somewhat smaller than this. On the other hand, if the point-spread function is known to vary across the frame, then *increasing* the fitting radius beyond the FWHM may improve the photometric accuracy provided, of course, that the field is *not* horribly crowded.

(13) "PSF RADIUS": The radius, in pixels, of the circle within which the point-spread function is to be defined. This should be somewhat larger than the actual radius of the brightest star you are interested in, as you would measure it on your image display. If, toward the end of your reductions (see §B above), you notice that the subtracted images of your bright stars are surrounded by luminous halos with sharp inner edges, then your PSF radius is too small. On the other hand, the CPU time required for the profile-fitting reductions is a strong function of the PSF radius, so it is counterproductive to make this parameter too large.

(14) "VARIABLE PSF": The degree of complexity with which the point-spread function is to be modeled. In its infancy, DAOPHOT Classic allowed only one form for the model PSF: a Gaussian analytic first approximation, plus a look-up table of empirical corrections from the approximate analytic model to the "true" PSF. This now corresponds to VARIABLE PSF = 0. Later on, I added the possibility of a point-spread function which varies linearly with position in the frame; this is VARIABLE PSF = 1. DAOPHOT II now allows two more possibilities: a point-spread function which varies quadratically with position in the frame (VARIABLE PSF = 2), and a purely analytic model PSF, with no empirical lookup table of corrections, as in ROMAFOT

(VARIABLE PSF = –1). Probably best to leave it at 0.0 (= "Constant") until you are *sure* you know what you're doing.

(15) "FRACTIONAL PIXEL EXPANSION": Not implemented. Leave it alone.

(16) "ANALYTIC MODEL PSF": DAOPHOT Classic always used a Gaussian function as an analytic first approximation to the point-spread function. DAOPHOT II allows a number of alternatives, which will be discussed below under the **PSF** command.

(17) "EXTRA PSF CLEANING PASSES": DAOPHOT II is now empowered to recognize and reduce the weight of obviously discrepant pixels while generating the average model point-spread function for the frame — cosmic rays, poorly subtracted neighbors and the like. This parameter specifies the number of times you want the program to go through the data and reevaluate the degree to which any given pixel is discordant and the amount by which its weight is to be reduced. Set this parameter to 0 if you want every pixel accepted at face value with full weight. The amount of time taken by the routine increases with the number of extra passes, and in my experience the point-spread function has usually converged to a stable final value within five passes, so I guess that's a reasonable guess at the largest value you'd want to use.

(18) "PERCENT ERROR": In computing the standard error expected for the brightness value in any given pixel, the program obviously uses the readout noise and the Poisson statistics of the expected number of photons. This parameter allows you to specify a particular value for the uncertainty of the *fine-scale* structure of the flat field. The readout noise is a constant; the Poisson error increases as the square root of the intensity; the "PERCENT ERROR" increases linearly with the intensity of (star + sky). You may think of it as the graininess of the inappropriateness of the flat-field frame which you used to calibrate your program images — not just the photon statistics but also any fine structure (on a scale smaller than a seeing disk) in the mismatch between the illumination of your flat-field frame and your program images.

(19) "PROFILE ERROR": In fitting the point-spread function to actual stellar images, there will also be some error due to the fact that the point-spread function is not known to infinite precision: not only will there be interpolation errors due to the finite sampling, but the point-spread function may vary in a greater than quadratic fashion with position in the frame, or with apparent magnitude, or with color, or with something else. This parameter defines the amplitude of this further contribution to the noise model; the "PROFILE ERROR" increases linearly with the intensity of the star alone (no sky), and inversely as the fourth power of the full-width at half-maximum. Therefore, this error grows in importance relative to the PERCENT ERROR as the seeing improves (since interpolation becomes harder as the data become more undersampled). Leave parameters (18) and (19) alone until much, much later.

There are four ways in which the user can supply values for these parameters to the program:

(1a) Whenever you run DAOPHOT II, the first thing the program will do is look in your current default directory for a file named DAOPHOT.OPT (daophot.opt on Unix machines). If it finds such a file, then it will read in the parameter specifications according to the format described below. When you run DAOPHOT II the file DAOPHOT.OPT acts pretty much the way LOGIN.COM does when you log onto a VMS VAX. Note that DAOPHOT II will only look for DAOPHOT.OPT in the directory from which you are currently running the program, so you should have a copy of DAOPHOT.OPT in each subdirectory where you are likely to work. If you have one subdirectory for every set of matched frames you are working on, it is easy to set up optimum parameter values for each set, and have them invoked automatically whenever you run DAOPHOT from that set's directory. *At the very least, you should have a* DAOPHOT.OPT *file specifying the READOUT NOISE, GAIN, FWHM, FITTING RADIUS, and HIGH GOOD DATA VALUE in* **every** *subdirectory where you keep images for* DAOPHOT II.

(1b) If no file named DAOPHOT.OPT is found, then default values for the parameters, as defined in the table in the Appendix, will be supplied by the program.

(2a) Whenever you have the 'Command:' prompt, you may use the **OPTIONS** command. The program will type on the terminal the current values for all user-definable parameters, and then ask you for an input filename. You may then enter the name of a file containing values (same format as used in a DAOPHOT.OPT file) you want to specify for the parameters.

(2b) When the **OPTIONS** command asks you for a filename, you may simply type a carriage return, and the program will then permit you to enter parameter values from the keyboard.

The syntax for specifying a parameter value, either from within a file or from the keyboard, is as follows. The parameter you wish to define is indicated by two alphanumeric characters; it doesn't matter whether they are upper or lower case, and any spaces or additional characters (except an equals sign and a number) after the first two are optional. The parameter identifier is followed by an equals sign, and this is followed by a number. The following commands would all set the FWHM of the objects for which the search is to be optimized to the value 3.0:

```
FW=3.0
fwhm = 3
Fwied wice is nice = 3.
```

When inputing parameter values from a file, one parameter is specified per line. Note that only those parameters whose values you want to change from the program-supplied defaults need be supplied by the user, either in manner (1a) above, or in (2a) or (2b).

You exit from OPTIONS by responding to the OPT> prompt with a carriage return or a CTRL-Z (CTRL-D on some Unix machines? Whatever means END-OF-FILE on your system).

EXAMPLE: CHANGING VALUES FROM THE KEYBOARD

To change the estimated full-width at half-maximum from 2.5 to 3.5, and the high sharpness cutoff from 1.0 to 0.9:

```
==============================================================================
| COMPUTER TYPES:                                        YOU ENTER:          |
|                                                                            |
| Command:                                               OP                  |
+----------------------------------------------------------------------------+
| COMPUTER TYPES:                                                            |
|                                                                            |
|  READ NOISE (ADU; 1 frame) =  5.00      GAIN (e-/ADU; 1 frame) =    10.00 |
|    LOW GOOD DATUM (sigmas) =  7.00    HIGH GOOD DATUM (in ADU) = 32766.50 |
|             FWHM OF OBJECT =  2.50       THRESHOLD (in sigmas) =     4.00 |
|  LS (LOW SHARPNESS CUTOFF) =  0.20  HS (HIGH SHARPNESS CUTOFF) =     1.00 |
|  LR (LOW ROUNDNESS CUTOFF) = -1.00  HR (HIGH ROUNDNESS CUTOFF) =     1.00 |
|             WATCH PROGRESS =  1.00              FITTING RADIUS =     2.00 |
|                 PSF RADIUS = 11.00                VARIABLE PSF =     0.00 |
| FRACTIONAL-PIXEL EXPANSION =  0.00          ANALYTIC MODEL PSF =     1.00 |
|  EXTRA PSF CLEANING PASSES =  5.00        PERCENT ERROR (in %) =     0.75 |
|       PROFILE ERROR (in %) =  5.00                                        |
+----------------------------------------------------------------------------+
| COMPUTER TYPES:                                        YOU ENTER:          |
|                                                                            |
|     Parameter file (default KEYBOARD INPUT):           <CR>                |
|                                                                            |
| OPT>                                                   FW=3.5              |
|                                                                            |
| OPT>                                                   HS=0.9              |
|                                                                            |
| OPT>                                                   <CR>                |
+----------------------------------------------------------------------------+
| COMPUTER TYPES:                                                            |
|                                                                            |
|  READ NOISE (ADU; 1 frame) =  5.00      GAIN (e-/ADU; 1 frame) =    10.00 |
|    LOW GOOD DATUM (sigmas) =  7.00    HIGH GOOD DATUM (in ADU) = 32766.50 |
|             FWHM OF OBJECT =  3.50       THRESHOLD (in sigmas) =     4.00 |
|  LS (LOW SHARPNESS CUTOFF) =  0.20  HS (HIGH SHARPNESS CUTOFF) =     0.90 |
|  LR (LOW ROUNDNESS CUTOFF) = -1.00  HR (HIGH ROUNDNESS CUTOFF) =     1.00 |
|             WATCH PROGRESS =  1.00              FITTING RADIUS =     2.00 |
|                 PSF RADIUS = 11.00                VARIABLE PSF =     0.00 |
| FRACTIONAL-PIXEL EXPANSION =  0.00          ANALYTIC MODEL PSF =     1.00 |
|  EXTRA PSF CLEANING PASSES =  5.00        PERCENT ERROR (in %) =     0.75 |
|       PROFILE ERROR (in %) =  5.00                                        |
==============================================================================
```

IV. **SKY**

The first time you start to work on a new frame with DAOPHOT II, you might want to issue the **SKY** command, which will return an estimate of the typical sky brightness for the frame.

```
========================================================================
| COMPUTER TYPES:                                     YOU ENTER:       |
|                                                                      |
| Command:                                            SK               |
|                                                                      |
|     Approximate sky value for this frame =   156.8                   |
|     Standard deviation of sky brightness =     4.16                  |
|                                                                      |
|                  Clipped mean and median =   157.9     157.5         |
|       Number of pixels used (after clip) =  8673                     |
========================================================================
```

The sky value returned is an estimate of the mode of the intensity values in somewhat fewer than 10,000 pixels scattered uniformly throughout the frame. That is, it picks 10,000 pixels, clips the low and high tails after the fashion of Kitt Peak's Mountain Photometry code, and computes the mean and median from what is left. The mode is taken as three times the median minus twice the mean. The standard deviation is the one-sigma width of the peak of the sky-brightness histogram about the *mean* sky brightness — (*not* the mode or the median — after clipping; for all but horrendously crowded frames this distinction is negligible for our present purposes). If you don't want to run the **SKY** command, **FIND** will do it for you anyhow.

V. **FIND**

You are now ready to find stars.

```
=======================================================================
| COMPUTER TYPES:                                     YOU ENTER:      |
|                                                                     |
| Command:                                            FI              |
|                                                                     |
|                                                                     |
|   Approximate sky value for this frame =   156.8                    |
|   Standard deviation of sky brightness =     4.16                   |
|                                                                     |
|                      Relative error = 1.14                          |
|                                                                     |
|        Number of frames averaged, summed:           5,1             |
|                                                                     |
|   File for the positions (default ?.COO):           <CR>            |
|                                                  or filename.ext    |
|                                                                     |
=======================================================================
```

The "Number of frames averaged, summed" question is in case the frame you are about to reduce represents the average or the sum of several independent readouts (readsout?) of the chip. The program uses this information to adjust the readout noise and the gain appropriately. In the example here, I have assumed that five individual exposures were *averaged* together to make the frame being reduced here; that is the five frames were added together and the sum was divided by the scalar value 5. If the frames had been added together and *not* divided by five, I would have entered "1,5" instead of "5,1". If I had taken six frames, summed them by pairs, and then averaged the three different sums, I would have entered "3,2": meaning "the average of three sums of two". If I had taken six frames, averaged the first three, averaged the second three, and then summed the two averages, I would also have entered "3,2": "averages of three, sum two of them". One final nuance: it happens that from a statistical noise point of view, the median of three frames is about as good as the average of two. Therefore, if the frame you are reducing represents the *median* of several independent exposures (to remove cosmic rays or whatever) enter it as if it were the average of two-thirds as many frames: the median of three images would be entered as "2,1", and the median of five would be entered as "3.3,1".

With this information, the routine will calculate a star-detection threshold corresponding to the number of standard deviations which you entered as the "THRESHOLD" option. The derived value of this threshold represents the minimum central height of a star image, in ADU, above its *local* sky background, which is required for a detection to be regarded as statistically significant. The theory behind star-detection is discussed briefly below, and the method used for computing the optimum threshold is described in the Appendix on "The **FIND** Threshold." The routine also computes a lowest good data-value corresponding to the number of standard deviations you specified in the "LOW GOOD DATUM" option, relative to the *average* sky brightness. That is, having determined that the modal sky level in this frame is 156.8, it determines that the brightness 7σ below this is 136.8.

For this calculation, it uses the specified readout noise, gain, and the fact that this is the average of five frames, *not* the *observed* σ(sky) = 4.16, because this latter value may have been puffed up by stars and defects. Then, any pixel *anywhere in the frame* which has an observed brightness value less than 136.8 or greater than the "HIGH GOOD DATUM" which you specified directly, will now and ever after be discarded as "defective." If you want to see what numerical values the routine gave to the star-detection threshold and the lowest good data value, you can find them in the second line of the output file created by this routine. If you want to *change* either of these values for subsequent reduction steps, you *must* do it in the file header lines; all other routines read these numbers from the input files, *not* from the optional-parameter table. The same goes for the HIGH GOOD DATUM value. Therefore, it is entirely in your own best interests to provide the program with the most reasonable possible estimates of the readout noise and gain, and of the minimum and maximum valid data values; doing it correctly at the beginning will save you hassles later. For instance, if the sky background in your frame is dead flat (a random Galactic star field, not in any cluster), a strict value for the lowest good data-value might be, say, five sigma below the average sky. A five-sigma or greater deviation has a normal probability of about 3×10^{-7}, so in a 300×500 image, there would be only about one chance in twenty that even one legitimate datum would be unfairly rejected. Of course, if the sky background does vary significantly across the frame (in a globular cluster, H II region, or external galaxy), you would want to set the minimum good data value maybe ten or more sigma below the average sky.

Finally, **FIND** asks you to provide a name for the disk file where it is to store the coordinates of the stars it finds. Here, as you will find nearly everywhere in DAOPHOT II, when asking you for a filename the program will offer you a default. If you are satisfied with the default filename, you need only type a carriage return and this name will be used; if you want to change the filename but keep the default filename extension, type in the filename you want, without any extension or period, and the default filename's extension will be tacked onto the filename you enter. Similarly, if you like the filename part but want to change the extension, simply type a period and the new extension: the filename which was offered to you will be retained, but with your extension replacing the one offered. I strongly suggest that you use the default filenames unless you have some very good reason for not doing so — it reduces the amount of typing that you have to do (and thereby reduces the probability of an unnoticed typographical error) and it helps you to keep your bookkeeping straight.

If you have elected to watch the output of the program on your terminal (either by using the option WATCH PROGRESS = 1.0 or by using the MONITOR command, see below), you will now see the computer count through the rows of your picture. As it is doing this, it is convolving your picture with a lowered truncated Gaussian function whose FWHM is equal to the value set by the FWHM option (see the **OPTIONS** command above). The Gaussian convolution includes compensation for the local background level, and since the Gaussian is symmetric, smooth gradients in the sky brightness also cancel out. These properties enable the sky-finding algorithm to ignore smooth, large-scale variations in the background level of your frame, such as those caused by a sea of unresolved fainter stars — the threshold that you have specified represents the minimum central brightness-enhancement *over the local background* which an object must have in order to be detected.

After having performed the convolution the program will then go through the convolved data looking for local maxima in the brightness enhancement. As the program finds candidates, it computes a couple of image-shape statistics (named SHARP and ROUND) which are designed to weed out delta functions (bad pixels), and brightness enhancements that are elongated in x or y (bad rows and columns).

SHARP is defined as the ratio of the height of the bivariate delta-function which best fits the brightness peak in the original image to the height of the bivariate Gaussian function (with the user-supplied value of the FWHM) which best fits the peak. If the brightness enhancement which was found in the convolved data is due to a single bright ("hot") pixel, then the best-fitting delta-function will have an amplitude equal to the height of this pixel above the mean local background, while the amplitude of the best-fitting Gaussian will be pulled down by the surrounding low-valued pixels, hence SHARP > 1. On the other hand, where there is a cold pixel, that is to say, where there is an isolated pixel whose brightness value is below the local average (but still above the "lowest good data-value") in the convolved data there will tend to be brightness-enhancements found approximately 0.5 FWHM away from this pixel in all directions; in such a case, the height of the delta function which best fits one of these spurious maxima tends to be close to zero, while the height of the best-fitting Gaussian is some small positive number: SHARP ~ 0. To reject both types of bad pixel, the default acceptance region for the SHARP parameter is:

$$0.20 \leq \text{SHARP} = \frac{\text{height of best-fitting delta function}}{\text{height of best-fitting Gaussian function}} \leq 1.00$$

ROUND is computed from the data in the original picture by fitting one-dimensional Gaussian functions to marginal sums of the data in x and y. Specifically, for each brightness enhancement which passes the SHARP test, if the height of either of these one-dimensional Gaussian distributions happens to be negative (a local minimum in the brightness distribution in that coordinate — sometimes happens) or zero, the object is rejected. If both turn out to be positive, then the ROUND parameter is computed:

$$\text{ROUND} = \frac{\text{difference between the heights of the two one-dimensional Gaussians}}{\text{average of the heights of the two one-dimensional Gaussians}}$$

Thus, if the two heights are, say, 90 and 150 ADU, then the average is 120 ADU and the difference is ± 60 ADU, so that ROUND would be ± 0.5. The sense of the difference is such that an object which is elongated in the x-direction has ROUND < 0 and one which is elongated in the y-direction has ROUND > 0. If the brightness enhancement which has been detected is really a charge-overflow column, for instance, then the brightness distribution as a function of x would be sharply peaked, while the distribution as a function of y would be nearly flat; the height of the x-Gaussian function would have some significant value, while that of the y-Gaussian would be near zero. In this case, ROUND would have a value near $+2.0$. The default acceptance limits for ROUND are

$$-1.0 \leq \text{ROUND} \leq 1.0,$$

i.e., if the heights of the x- and y-Gaussian distributions for a brightness enhancement were 60 and 180 ADU, difference/average $= \pm 120/120$ and the object would be right at the limit of acceptance. Note that ROUND discriminates only against objects which are elongated along either rows or columns — objects which are elongated at an oblique angle will not be preferentially rejected.

The numerical values for the limits of the acceptance interval for SHARP and ROUND may be changed by the user (see the **OPTIONS** command above), if normal stars in your frame should happen to have unusual shapes due to optical aberrations or guiding problems. However, I recommend that you take great care in deciding on new values for these cutoffs. It might be useful to perform a preliminary run of **FIND** with very generous limits on the acceptance regions, and then to plot up both SHARP and ROUND as functions of magnitude for the objects detected (see Stetson 1987, PASP, 99, 191, Fig. 2). Observe the mean values of SHARP and ROUND for well-exposed stars, observe the magnitude fainter than which these indices blow up, and determine the range of values of SHARP and ROUND spanned by stars above that magnitude limit. Having decided on the region of each of the two diagrams occupied by worthwhile stars, you could then rerun **FIND** with a new threshold and new values for the limits on SHARP and ROUND. Alternatively, you could write yourself a quickie program which goes through the output file produced by **FIND** and rejects those objects which fall outside your new acceptance limits (which could even be functions of magnitude if you so desired).

Back in **FIND**: Once a brightness enhancement passes muster as a probable star, its centroid is computed (approximately). When the FIND routine has gone through your entire picture, it will ask

```
=========================================================================
| COMPUTER TYPES:                                     YOU ENTER:        |
|                                                                       |
| Are you happy with this?                              Y               |
|                                                    or N               |
=========================================================================
```

If you answer "Y", the program will exit from **FIND** and return you to DAOPHOT "Command:" mode. If you answer "N", it will ask for a new threshold and output filename, and will then search through the convolved picture again.

A couple more comments on star-finding:

(1) Stars found in the outermost few rows and columns of the image, that is, where part of their profile hangs over the edge of the frame, do not have their positions improved. Instead, their positions are returned only to the nearest pixel. Furthermore, the ROUNDNESS index is not computed for such stars, although the SHARPNESS test is still performed.

(2) Please don't try to set the threshold low enough to pick up every last one-sigma detection — this will cause you nothing but grief later on. The CPU time for **NSTAR** goes as something like the fourth power of the surface density of detected objects in your frame (ALLSTAR isn't quite so bad), so including a large number of spurious detections greatly increases the reduction time. Even worse, as the iterative profile fits progress, if these fictitious stars have no real brightness enhancements to anchor themselves to, they can migrate around the frame causing real stars to be fit twice (in **PEAK** and **NSTAR**, probably not in ALLSTAR) or fitting themselves to noise peaks in the profiles of brighter stars (all three routines). This will add scatter to your photometry. Try to set a reasonable threshold. If you want, you can experiment by repeatedly

replying to "Are you happy with this?" with "N", and on a piece of graph paper build yourself up a curve showing number of detected objects as a function of threshold. This curve will probably have an elbow in it, with the number of detected objects taking off as the threshold continues to be lowered. The best threshold value will be somewhere near this elbow. (This method of experimenting with different thresholds is much faster than just running **FIND** many times, because by answering "N" and giving a new threshold, you avoid the need to recompute the convolution of the picture.)

(3) The crude magnitudes which **FIND** computes and includes in your terminal and disk-file output are defined relative to the threshold which you gave it — a star with a **FIND** magnitude of 0.000 is right at the detection limit. Since most stars are brighter than the threshold **FIND** will obviously give them negative magnitudes. For the faintest stars the magnitudes may be quantized, since if the original image data are stored as integers, the convolved image data will be, too. Thus, if your **FIND** threshold came out to 20.0 ADU, the next brighter magnitude a star may have, after 0.000, is $-2.5 \log (21/20) = -0.053$. When the image data are stored as floating-point numbers, this quantization will not occur.

VI. **PHOTOMETRY**

Before you can do concentric aperture photometry with DAOPHOT, you need to have an aperture photometry parameter file. At the DAO, a prototype parameter file named DAO:PHOTO.OPT is available. The file looks something like this:

```
=========
 A1 = 3
 A2 = 4
 A3 = 5
 A4 = 6
 A5 = 7
 A6 = 8
 A7 = 10
 IS = 20
 OS = 35
=========
```

When you give DAOPHOT the **PHOTOMETRY** command to invoke the aperture photometry routine,

```
======================================================================
| COMPUTER TYPES:                                  YOU ENTER:        |
|                                                                    |
| Command:                                         PH                |
|                                                                    |
|      Enter table name (default PHOTO.OPT):       <CR> or table name |
======================================================================
```

and a table like this one will appear on your terminal screen:

```
=============================================================================
 A1  RADIUS OF APERTURE  1 =    3.00    A2  RADIUS OF APERTURE  2 =     4.00
 A3  RADIUS OF APERTURE  3 =    5.00    A4  RADIUS OF APERTURE  4 =     6.00
 A5  RADIUS OF APERTURE  5 =    7.00    A6  RADIUS OF APERTURE  6 =     8.00
 A7  RADIUS OF APERTURE  7 =   10.00    A8  RADIUS OF APERTURE  8 =     0.00
 A9  RADIUS OF APERTURE  9 =    0.00    AA  RADIUS OF APERTURE 10 =     0.00
 AB  RADIUS OF APERTURE 11 =    0.00    AC  RADIUS OF APERTURE 12 =     0.00
 IS       INNER SKY RADIUS =   20.00    OS       OUTER SKY RADIUS =    35.00

 PHO>
=============================================================================
```

When you have the "PHO>" prompt, you can alter any of the values displayed in the table. To do this, you first enter the two-character identifier of the item that you want to change, followed by an "=" sign, and the new numerical value for that parameter. It works just like the **OPTIONS** command: any characters between the first two and the equals sign will be ignored, and anything but a legitimate decimal number after the equals sign will produce an error message. For instance, to change the radius of the first aperture to 2.5 pixels, and change the inner and outer radii of the sky annulus to 10 and 20 pixels, respectively:

```
========================================================================
| COMPUTER TYPES:                                  YOU ENTER:          |
|                                                                      |
| PHO>                                             A1=2.5              |
|                                                                      |
| PHO>                                             IS=10               |
|                                                                      |
| PHO>                                             OS=20               |
|                                                                      |
| PHO>                                             <CR>                |
|                                                                      |
========================================================================
```

and the modified table will appear on the screen:

```
===============================================================================
 A1  RADIUS OF APERTURE  1 =     2.50    A2  RADIUS OF APERTURE  2 =     4.00
 A3  RADIUS OF APERTURE  3 =     5.00    A4  RADIUS OF APERTURE  4 =     6.00
 A5  RADIUS OF APERTURE  5 =     7.00    A6  RADIUS OF APERTURE  6 =     8.00
 A7  RADIUS OF APERTURE  7 =    10.00    A8  RADIUS OF APERTURE  8 =     0.00
 A9  RADIUS OF APERTURE  9 =     0.00    AA  RADIUS OF APERTURE 10 =     0.00
 AB  RADIUS OF APERTURE 11 =     0.00    AC  RADIUS OF APERTURE 12 =     0.00
 IS       INNER SKY RADIUS =    10.00    OS       OUTER SKY RADIUS =    20.00

         File with the positions (default ?.COO):
===============================================================================
```

Note that you are allowed to specify radii for up to twelve concentric apertures. These do not need to increase or decrease in any particular order*, except that only the magnitude in the first aperture will appear on the screen of your terminal as the reductions proceed, and the magnitude in the first aperture will be used to define the zero point and to provide starting guesses for the profile-fitting photometry. Photometric data for all apertures will appear in the disk data file created by this routine. The first zero or negative number appearing in the list of aperture radii terminates the list. Thus, the tables above both specify that photometry through seven apertures is to be obtained, and that the first aperture is to be 3 pixels in radius in the first instance, 2.5 pixels in radius in the second. Items IS and OS are the inner and outer radii, respectively, of a sky annulus centered on the position of each star.

* If you plan to use DAOGROW, the aperture radii must increase monotonically.

Now you can proceed to do the photometry:

```
=========================================================================
| COMPUTER TYPES:                                   YOU ENTER:          |
|                                                                       |
|    File with the positions (default ?.COO):      <CR> or filename    |
|                                                                       |
|     File for the magnitudes (default ?.AP):      <CR> or filename    |
|                                                                       |
=========================================================================
```

If you are monitoring the progress of the reductions on your terminal, the computer will start spitting out the star ID's and coordinates from the star-finding results, along with the instrumental magnitudes from the first aperture and the sky brightness values for all the stars. When all the stars have been reduced, a line like the following will appear:

Estimated magnitude limit (Aperture 1): nn.nn +– n.n per star.

This is simply a very crude estimate of the apparent instrumental magnitude of a star whose brightness enhancement was exactly equal to the threshold you specified for the star-finding algorithm. It is intended for your general information only — it is not used elsewhere in DAOPHOT II, and it has meaning only if all the stars were found by the **FIND** routine. If you have entered some of the coordinates by hand, or if you are redoing aperture photometry from data files that have already undergone later stages of reduction, the magnitude limit is not meaningfully defined.

Other things you should know about **PHOTOMETRY**:

(1) The maximum outer radius for the sky annulus depends on how much virtual memory your system manager will let the program use, and on how big you set the inner radius. If you set too large a value the program will complain and tell you the largest radius you can get away with.

(2) If any of the following conditions is true, **PHOTOMETRY** will assign an apparent magnitude of 99.999 ± 9.999 to your star:

   (a) if the star aperture extends outside the limits of the picture;

   (b) if there is a bad pixel (either above the "High good datum" or below the "Low good datum") inside the star aperture;

   (c) if the star + sky is fainter than the sky; or

   (d) if for some reason the program couldn't determine a reasonable modal sky value (very rare; this latter case will be flagged by $\sigma_{sky} = -1$).

(3) *Very important*: If, after the rejection of the tails of the sky histogram, the sky-finding algorithm is left with fewer than 20 pixels, **PHOTOMETRY** will refuse to proceed, and will return to to DAOPHOT II command mode. *Check your inner and outer sky annulus radii, and your good data-value limits.* In particular, if you are reducing a short-exposure frame, where the mean sky brightness is small compared to the readout noise, the lowest good data-value (computed in

FIND, see above) should have been set to some moderate-sized negative number. If this is not the case there'll be "bad" pixels all over the place and you'll never get any photometry done. As suggested above, I generally set the bad pixel threshold somewhere around five to six sigma below the typical sky brightness (obtained from the **SKY** command, see above) unless the background is significantly non-uniform, in which case I set it even lower. But, if you are absolutely sure that the sky background does not change significantly across your frame and you want to be really clever, you can set the threshold to something like 4.35 sigma below the mean sky brightness: the one-sided tail of the cumulative normal distribution, Prob$(X < -4.35\sigma) = 7 \times 10^{-6}$, or about one legitimate random pixel being spuriously identified as "bad" in a 300×500 picture.

VII. **PICK** and **PSF**

In my opinion, obtaining a point-spread function in a crowded field is still done best with at least a small modicum of common (as distinguished from artificial) intelligence. One possible procedure for performing this task is outlined in an Appendix. At this point I will say only that in the beginning you may find it to your advantage to perform this task interactively, while you are able to examine your original picture and its offspring (with various of the stars subtracted out by procedures described in the Appendix) on an image display system. After you find that you are performing exactly the same operations in exactly the same sequence for all your frames, you may choose to write command procedures to carry out the bulk of this chore, with a visual check near the end.

DAOPHOT Classic assumed that the point-spread function of a CCD image could be adequately modeled by the sum of (a) an analytic, bivariate Gaussian function with some half-width in the x-direction and some half-width in the y-direction, and (b) an empirical look-up table representing corrections from the best-fitting Gaussian to the actual observed brightness values within the average profile of several stars in the image. This hybrid point-spread function seemed to offer both adequate flexibility in modelling the complex point-spread functions that occur in real telescopes, with some hope of reasonable interpolations for critically sampled or slightly undersampled data. This approximation has since turned out to be not terribly good for significantly undersampled groundbased data, and it has turned out to be particularly poor for images from the Hubble Space Telescope. To help meet these new requirements, DAOPHOT II offers a wider range of choices in modelling the point-spread function. In particular, different numerical models besides the bivariate Gaussian function may be selected for the analytic first approximation. These are selected by the "ANALYTIC MODEL PSF" option, as I will explain later.

But to back up a bit, to aid you in your choice of PSF stars, I have provided a very simple routine called **PICK**, which does some fairly obvious things. First, it asks for an input file containing a list of positions and magnitudes: by no coincidence at all, the aperture photometry file you just created with **PHOTOMETRY** is ideal:

```
========================================================================
| COMPUTER TYPES:                                  YOU ENTER:          |
|                                                                      |
| Command:                                         PICK                |
|                                                                      |
|            Input file name (default ?.AP):       <CR> or filename    |
|                                                                      |
|Desired number of stars, faintest magnitude:      <some numbers>      |
|                                                                      |
|          Output file name (default ?.LST):       <CR> or filename    |
|                                                                      |
|                                                                      |
|  <Some number of> suitable candidates were found.                    |
|                                                                      |
|                                                                      |
| Command:                                                             |
========================================================================
```

You tell the thing how many PSF stars you'd like to have, and the limiting instrumental magnitude for the faintest star you'd like to consider using for the PSF. It then sorts the input star list by apparent magnitude (note that if you have set the HIGH GOOD DATUM parameter appropriately, any saturated stars should have magnitudes of 99.999 and will appear at the *end* of the list), and then it uses the FITTING RADIUS and the PSF RADIUS that you have specified among the optional parameters to eliminate: (a) stars that are too close to the edge of the frame (within one FITTING RADIUS), and (b) stars that are too close to *brighter* stars or possibly saturated stars (within one PSF RADIUS *plus* one FITTING RADIUS). Any stars that remain after this culling process will be written out in order of increasing apparent magnitude, up to the number that you have specified as a target. Stars fainter than the magnitude limit will be included *only* if otherwise there would be too few PSF stars to map out the degree of spatial variation that you specified with your "VARIABLE PSF" option. If you don't know what magnitude to specify, just enter 99 or some such number, and learn empirically what looks bright enough to be good.

With the ".LST" file containing the candidate PSF stars, you are ready to run the **PSF** routine:

```
========================================================================
| COMPUTER TYPES:                                  YOU ENTER:          |
|                                                                      |
| Command:                                         PSF                 |
|                                                                      |
|  File with aperture results (default ?.AP):      <CR> or filename    |
|                                                                      |
|        File with PSF stars (default ?.LST):      <CR> or filename    |
|                                                                      |
|           File for the PSF (default ?.PSF):      <CR> or filename    |
|                                                                      |
========================================================================
```

What happens next depends upon the "WATCH PROGRESS" option. If "WATCH PROGRESS" has been set to 1 or 2, the program will produce little pseudo-images of your PSF stars on your terminal, one by one. These images are made up of standard alphanumeric characters, so it should work whether you are on a graphics terminal or not. After showing you each picture, the routine will ask you whether you want this star included in the average PSF. If you answer "y" or "Y", the star will be included, if you answer "n" or "N", it won't.

If the PSF routine discovers that a PSF star has a bad pixel (defined as being below the low good data value or above the HIGH GOOD DATUM value) *inside* one FITTING RADIUS of the star's centroid, it will refuse to use the star. If it discovers that the star has a bad pixel *outside* one FITTING RADIUS but inside a radius equal to (one PSF RADIUS plus two pixels), what it does again depends on the WATCH PROGRESS option. If WATCH PROGRESS = 0, 1, or 2, **PSF** will inform you of the existence of the bad pixels and ask whether you want to use that star anyway. If you answer "y" or "Y" you are counting on the bad-pixel rejection scheme later on in **PSF** to eliminate the flaw, while extracting whatever information it can from the remaining, uncontaminated part of the star's profile. In my experience, this has always worked acceptably well. If you have set WATCH

PROGRESS to –2 or –1, **PSF** will type out a message about the bad pixel(s), but will go ahead and use the star anyway without prompting for a response.

After **PSF** has read your input list of PSF stars…

```
=======================================================================
| COMPUTER TYPES:                                                     |
|                                                                     |
|     Chi      Parameters...                                          |
|   0.0282    0.71847   0.83218   0.30568                             |
|                                                                     |
|  Profile errors:                                                    |
|                                                                     |
|   180 0.023    555 0.026    122 0.026    531 0.023    725 0.026     |
|   821 0.026    759 0.029     82 0.063 ?   19 0.028    303 0.027     |
|   784 0.027    189 0.029    660 0.028    536 0.026    427 0.025     |
|    92 0.028    766 0.027    873 0.025    512 0.029    456 0.096 *   |
|   865 0.023    752 0.027    715 0.026    125 0.026    440 0.026     |
|   375 0.026    467 0.030     99 0.028    652 0.029                  |
|                                                                     |
| File with PSF stars' neighbors = ?.NEI                              |
|                                                                     |
=======================================================================
```

If WATCH PROGRESS ≥ -1 the numbers under the "Chi Parameters..." heading will dance about for a while. (If you are doing these calculations in a batch job which creates a logfile, you should set WATCH PROGRESS = –2. This will suppress the dancing, which — especially under Unix — could cause a clog in the log.) When the dancing stops, the computer has fit the analytic function of your choice to the (in this case) 29 PSF stars. The value labeled "Chi" represents the root-mean-square residuals of the actual brightness values contained within circles of radius one FITTING RADIUS about the centroids of the PSF stars. That is to say: the routine has done the best job it could of fitting the analytic function to the pixels within one FITTING RADIUS of the PSF stars — one function fitting all the stars. The "Chi" value is the root-mean-square of the residuals that are left, expressed as a fraction of the peak height of the analytic function. In this case, the analytic first approximation matched the observed stellar profiles to within about 2.8%, root-mean-square, on average. Part of this number presumably represents the noise in the stellar profiles, but most of it is due to the fact that the analytic function does not accurately reflect the true stellar profile of the frame. It is this systematic difference between the true profile and the analytic first approximation that is to go into the look-up table of profile corrections.

The actual derived parameters of the best-fitting analytic function are typed out next. In this case, the analytic function chosen had three free parameters. For *all* the different analytic first approximations, the first two parameters are *always* the half-width at half-maximum in x and y. Any other parameters the model may have differ from function to function, but the first two are, as I say, always the half-width at half-maximum in x and y.

Next, the routine types out the *individual* values for the root-mean-square residual of the actual stellar profile from the best-fitting analytic model, star by star. Again, these values are computed only from those pixels lying within a FITTING RADIUS of the stars' centroids — cosmic rays or companion stars in the PSF stars' outer wings do not contribute to these numbers. Any star with an individual profile scatter greater than three times the average is flagged with a "*"; a star showing scatter greater than twice the average is flagged with "?". The user may want to consider deleting these stars from the .LST file and making another PSF without them. Apart from the two objects flagged, the scatter values given in the sample above demonstrate that systematic differences between the "true" stellar profile and the analytic first approximation dominate over raw noise in the profiles: the typical root-mean-square residual does not increase very much from the beginning of the list to the end — I happen to know that these stars were sorted by increasing apparent magnitude in the .LST file.

Finally, **PSF** reminds you that it has created a file named ?.NEI which contains the PSF stars and their recognized neighbors in the frame. This file may be run through **GROUP** and **NSTAR** or through ALLSTAR for a quickie profile fit, and then the neighbors may be selectively subtracted from the original image to help isolate the PSF stars for an improved second-generation PSF.

Unlike DAOPHOT Classic, DAOPHOT II: The Next Generation *is* able to use PSF stars that are within a PSF RADIUS of the edge of the frame, provided that they are at least a FITTING RADIUS from the edge.

Finally, the analytic first approximations. At this particular point in time (you like that verbose stuff?) there are six allowed options:

(1) A Gaussian function, having two free parameters: half-width at half-maximum in x and y. The Gaussian function may be elliptical, but the axes are aligned with the x and y directions in the image. This restriction allows for fast computation, since the two-dimensional integral of the bivariate Gaussian over the area of any given pixel may be evaluated as the product of two one-dimensional integrals.

(2) A Moffat function, having three free parameters: half-width at half-maximum in x and y, and (effectively) a position angle for the major axis of the ellipse. Since it's necessary to compute the two-dimensional integral anyway, we may as well let the ellipse be inclined with respect to the cardinal directions. In case you don't know it, a Moffat function is

$$\propto \frac{1}{(1+z^2)^{\beta}}$$

where z^2 is something like $x^2/\alpha_x^2 + y^2/\alpha_y^2 + \alpha_{xy}xy$ (Note: *not* $\ldots + xy/\alpha_{xy}$ so α_{xy} can be zero). In this case, $\beta = 1.5$.

(3) A Moffat function, having the same three parameters free, but with $\beta = 2.5$.

(4) A Lorentz function, having three free parameters: ditto.

(5) A "Penny" function: the sum of a Gaussian and a Lorentz function, having four free parameters. (As always) half-width at half-maximum in x and y; the fractional amplitude of the Gaussian function at the peak of the stellar profile; and the position angle of the tilted elliptical Gaussian. The Lorentz function may be elongated, too, but its long axis is parallel to the x or y direction.

(6) A "Penny" function with five free parameters. This time the Lorentz function may also be tilted, in a different direction from the Gaussian.

It is possible that these assignments will be changed or augmented in the future. If you are worried about the details, I suggest you consult the source code: the routine PROFIL in the file MATHSUBS.FOR.

VIII. **PEAK**

PEAK is a single-star profile-fitting algorithm. Because it is not to be trusted for the reduction of overlapping images, **PEAK** should never be used for the final photometric reduction in fields where stars of interest may be blended. On the other hand, for sparsely-populated frames aperture photometry is often fine, and **NSTAR** or ALLSTAR photometry is virtually as fast. Thus, **PEAK** is largely a historical artifact, reminding us that once life was simpler. For you archaeology buffs, here is how **PEAK** used to be used. I now quote from the original DAOPHOT manual.

"Obviously, before PEAK can be run, you must have created a point-spread function. Assuming this to be the case, then

```
======================================================================
| COMPUTER TYPES:                                  YOU ENTER:        |
|                                                                    |
| Command:                                         PE                |
|                                                                    |
|     File with aperture results (default ?.AP):   <CR> or filename  |
|                                                                    |
|             File with the PSF (default ?.PSF):   <CR> or filename  |
|                                                                    |
|          File for PEAK results (default ?.PK):   <CR> or filename  |
======================================================================
```

If you have issued the **MONITOR** command (see below) or if WATCH PROGRESS = 1.0 (see the **OPTIONS** command above), you will then see the results of the peak-fitting photometry appear on your screen. If WATCH PROGRESS = 2.0, then you will also see an eleven-level gray-scale image of the region around each star as it is being reduced; since it is very time-consuming to produce these pictures on your terminal screen, WATCH PROGRESS should be set to 2.0 only when it is strictly necessary.

"The **PEAK** algorithm also makes use of the optional parameter FITTING RADIUS: only pixels within a FITTING RADIUS of the centroid of a star will be used in fitting the point-spread function. Furthermore, for reasons related to accelerating the convergence of the iterative non-linear least-squares algorithm, the pixels within this FITTING RADIUS are assigned weights which fall off from a value of unity at the position of the centroid of the star to identically zero at the fitting radius. Since the least-squares fit determines three unknowns (x and y position of the star's centroid, and the star's brightness) it is absolutely essential that the fitting radius not be so small as to include only a few pixels with non-zero weight. FITTING RADII less than 1.6 pixels (that would include as few as four pixels in the fit) are explicitly forbidden by the program. I suggest that a fitting radius comparable to the FWHM be used as a general rule, but suit yourself.

"In addition to an improved estimate of the x, y-position of the centroid of each star, and of its magnitude and the standard error of the magnitude, **PEAK** also produces two image-peculiarity indices which can be used to identify a disturbed image. The first of these, CHI, is essentially the ratio of the observed pixel-to-pixel scatter in the fitting residuals to the expected scatter, based on the values of readout noise and the photons per ADU which you specified in your aperture photometry table. If your values for the readout noise and the photons per ADU are correct, then in a plot of CHI against derived magnitude (*e.g.* Stetson and Harris 1988, *A.J.* **96**, 909, Fig. 28), most stars should scatter around unity, with little or no trend in CHI with magnitude (except at the very bright end, where saturation effects may begin to set in).

"The second image-peculiarity statistic, SHARP, is vaguely related to the *intrinsic* (*i.e.* outside the atmosphere) angular size of the astronomical object: effectively, SHARP is a zero-th order estimate of the square of the quantity (actual one-sigma half-characteristic-width of the astronomical object as it would be measured outside the atmosphere, in pixels).

$$\mathrm{SHARP}^2 \sim \sigma^2(\text{observed}) - \sigma^2(\text{point-spread function})$$

(This equation is reasonably valid provided SHARP is not significantly larger than the square of the one-sigma Gaussian half-width of the core of the PSF; see the .PSF file in Appendix IV.) For an isolated star, SHARP should have a value close to zero, whereas for semi-resolved galaxies and unrecognized blended doubles SHARP will be significantly greater than zero, and for cosmic rays and some image defects which have survived this far into the analysis SHARP will be significantly less than zero. SHARP is most easily interpreted when plotted as a function of apparent magnitude for all objects reduced (*e.g.*, Stetson and Harris 1988, *A.J.* **96**, 909, Fig. 27). Upper and lower envelopes bounding the region of single stars may be drawn by eye or by some automatic scheme of your own devising.

"Finally, **PEAK** tells you the number of times that the profile fit had to be iterated. The program gives up if the solution has been iterated 50 times without achieving convergence, so stars for which the number of iterations is 50 are inherently more suspect than the rest. Frequently, however, the solution was oscillating by just a little bit more than the convergence criterion (which is fairly strict: from one iteration to the next the computed magnitude must change by less than 0.0001 mag. or 0.05 sigma, whichever is larger, and the x- and y-coordinates of the centroid must change by less than 0.001 pixel for the program to feel that convergence has been achieved). Therefore, in many cases stars which have gone 50 iterations are still moderately well measured.

"One other thing of which you should be aware: **PEAK** uses a fairly conservative formula for automatically reducing the weight of a "bad" pixel (not a "bad" pixel as defined in **FIND** — **PEAK** ignores those — but rather any pixel which refuses to approach the model profile as the iterative fit proceeds). This formula depends in part on the random errors that the program expects, based on the values for the readout noise and the photons per ADU which you, the user, specified in the aperture photometry table. It is therefore distinctly to your advantage to see to it that the values supplied are reasonably correct. "

IX. **GROUP**

Before performing multiple, simultaneous profile fits using the routine **NSTAR**, it is necessary to divide the stars in your frame up into natural groups, each group to be reduced as a unit. The principle is this: if two stars are close enough together that the light of one will influence the profile-fit of another, they belong in the same group. That way, as the solution iterates to convergence, the influence on each star of all of its relevant neighbors can be explicitly accounted for.

```
========================================================================
| COMPUTER TYPES:                                 YOU ENTER:          |
|                                                                     |
| Command:                                        GR                  |
|                                                                     |
|    File with the photometry (default ?.AP):     <CR> or filename    |
|                                                                     |
|          File with the PSF (default ?.PSF):     <CR> or filename    |
|                                                                     |
|                           Critical overlap:     n.n                 |
|                                                                     |
|    File for stellar groups (default ?.GRP):     <CR> or filename    |
========================================================================
```

The "critical overlap" has the following significance. When **GROUP** is examining two stars to see whether they could influence each others' fits, it first identifies the fainter of the two stars. Then, it calculates the brightness of the brighter star (using the scaled PSF) at a distance of one fitting radius plus one pixel from the centroid of the fainter. If this brightness is greater than "critical overlap" times the random error per pixel (calculated from the known readout noise, sky brightness in ADU and number of photons per ADU), then the brighter star is known to be capable of affecting the photometry of the fainter, and the two are grouped together. You must determine what value of "critical overlap" is suitable for your data frame by trial and error: if a critical overlap = 0.1 divides all of the stars up into groups smaller than 60, then you may be sure that unavoidable random errors will dominate over crowding. If critical overlap = 1.0 works, then crowding will be no worse than the random errors. If critical overlap $\gg 1.0$ is needed, then in many cases crowding will be a dominant source of error.

After **GROUP** has divided the stars up and created an output disk file containing these natural stellar groups, a little table will be produced on your terminal showing the number of groups as a function of their size. If any group is larger than the maximum acceptable to **NSTAR** (currently 60 stars), then the critical overlap must be increased, or the **SELECT** command (see below) should be used to cut the overly large groups out of the file. When crowding conditions vary across the frame, judicious use of **GROUP** and **SELECT** will pick out regions of the data frame where different critical overlaps will allow you to get the best possible photometry for stars in all of the crowding regimes.

X. **NSTAR**

NSTAR is DAOPHOT's multiple-simultaneous-profile-fitting photometry routine. It is used in very much the same way as PEAK:

```
========================================================================
| COMPUTER TYPES:                                  YOU ENTER:          |
|                                                                      |
| Command:                                         NS                  |
|                                                                      |
|             File with the PSF (default ?.PSF):   <CR> or filename    |
|                                                                      |
|      File with stellar groups (default ?.GRP):   <CR> or filename    |
|                                                                      |
|        File for NSTAR results (default ?.NST):   <CR> or filename    |
|                                                                      |
========================================================================
```

The principal difference is that **NSTAR** is much more accurate than **PEAK** in crowded regions, although at the cost of requiring somewhat more time per star, depending on the degree of crowding. **NSTAR** automatically reduces the weight of bad pixels just as **PEAK** does, so it is highly advisable that your values for the readout noise and the number of photons per ADU be just as correct as you can make them. **NSTAR** also produces the same image-peculiarity statistics CHI and SHARP, defined as they were in **PEAK**. Plots of these against apparent magnitude are powerful tools for seeing whether you have specified the correct values for the readout noise and the photons per ADU (Do most stars have CHI near unity? Is there a strong trend of CHI with magnitude?), for identifying stars for which the profile fits just haven't worked (they will have values of CHI much larger than normal for stars of the same derived magnitude), for identifying probable and possible galaxies (they will have larger values of SHARP than most), for identifying bad pixels and cosmic rays that made it through **FIND** (large negative values of SHARP), and for seeing whether your brightest stars are saturated (the typical values of SHARP and CHI will tend to increase for the very brightest stars).

The maximum number of iterations for **NSTAR** is 50, but *every* star in a group must individually satisfy the convergence criteria (see **PEAK**) before the program considers the group adequately reduced.

NSTAR also has a slightly sophisticated star-rejection algorithm, which is essential to its proper operation. A star can be rejected for several reasons:

(1) If two stars in the same group have their centroids separated by less than a critical distance (currently set more or less arbitrarily to $0.37\times$ the FWHM of the stellar core), they are presumed to be the same star, their photocentric position and combined magnitude is provisionally assigned to the brighter of the two and the fainter is eliminated from the starlist before going into the next iteration.

(2) Any star which converges to more than 12.5 magnitudes fainter than the point-spread function (one part in ten to the fifth; e.g., central brightness < 0.2 ADU/pixel if the first PSF star had a

central brightness of 20,000 ADU/pixel) is considered to be non-existent and is eliminated from the starlist.

(3a) After iterations 5, 6, 7, 8, and 9, if the faintest star in the group has a brightness less than one sigma above zero, it is eliminated;

(3b) after iterations 10 – 14, if the faintest star in the group has a brightness less than 1.5 sigma above zero, it is eliminated;

(3c) after iterations 15 – 50, or when the solution thinks it has converged, whichever comes first, if the faintest star in the group has a brightness less than 2.0 sigma above zero, it is eliminated.

(4a,b,c) Before iterations 5 – 9, before iterations 10 – 14, and before iterations 15 – 50, if two stars are separated by more than 0.37× the FWHM and less than 1.0× the FWHM, and if the fainter of the two is more uncertain than 1.0, 1.5, or 2.0 sigma (respectively), the fainter one is eliminated.

Whenever a star is eliminated, the iteration counter is backed up by one, and the reduction proceeds from that point with the smaller set of stars. Backing up the iteration counter gives the second least certain star in the group as much as two full iterations to settle into the new model before it comes up for a tenure decision. Since the star-rejection formula depends in part upon the user-specified values for the readout noise and the number of photons per ADU, it is once again important that the values you give for these parameters be reasonable.

XI. **SUBSTAR**

The **SUBSTAR** command takes the point-spread function for a frame and a data file containing a set of x, y coordinates and apparent magnitudes for a set of stars, shifts and scales the point-spread function according to each position and magnitude, and then subtracts it from your original frame. In the process a new data frame is produced (your original picture file is left inviolate). In principal, this can be done using photometry from **PHOTOMETRY** as well as from **PEAK**, **NSTAR**, or ALLSTAR, but I can't imagine why on earth you'd want to. In general, this star subtraction is done after the photometric reductions have been performed, as a check on the quality of the profile fits, and to highlight non-stellar objects and tight binaries. Additional uses for **SUBSTAR**:

(1) decrowding bright stars for the generation of an improved point-spread function (see Appendix on constructing a PSF); and

(2) decrowding bright stars for establishing the magnitude zero-point of the frame by means of aperture photometry through a series of apertures.

Here's how it goes.

```
========================================================================
| COMPUTER TYPES:                                  YOU ENTER:          |
|                                                                      |
| Command:                                         SU                  |
|                                                                      |
|             File with the PSF (default ?.PSF):   <CR> or filename    |
|                                                                      |
|          File with photometry (default ?.NST):   <CR> or filename    |
|                                                                      |
|                Do you have stars to leave in?    N                   |
|                                                                      |
|    Name for subtracted image (default ?s.DST):   <CR> or filename    |
|                                                                      |
========================================================================
```

This will subtract from the image that is currently **ATTACH**ed all the stars in the "File with photometry." The picture produced will have a format identical to your original picture — it will be acceptable as input to DAOPHOT, if you so desire.

```
=======================================================================
| COMPUTER TYPES:                                   YOU ENTER:        |
|                                                                     |
| Command:                                          SU                |
|                                                                     |
|            File with the PSF (default ?.PSF):     <CR> or filename  |
|                                                                     |
|         File with photometry (default ?.NST):     <CR> or filename  |
|                                                                     |
|               Do you have stars to leave in?      Y                 |
|                                                                     |
|          File with star list (default ?.LST)      <CR> or filename  |
|                                                                     |
|   Name for subtracted image (default ?s.DST):     <CR> or filename  |
|                                                                     |
=======================================================================
```

This time **SUBSTAR** will subtract from the image all the stars in the "File with photometry" *except* those that appear in the "File with star list." This makes it easy to clean out stars around your PSF stars or around the stars for which you wish to perform concentric-aperture photometry. Note, however, that stars are cross-identified solely on the basis of their ID numbers in the various files. If you use the renumber option of the **SORT** command or if you **APPEND** together files where ID numbers are duplicated, you may find yourself leaving the wrong stars in the image.

XIII. **MONITOR/NOMONITOR**

If you want simply to turn off the sending of the results to your terminal screen, this can be accomplished without going through all the foofarah of the **OPTIONS** command, by using the **NOMONITOR** command:

```
========================================================================
| COMPUTER TYPES:                                   YOU ENTER:         |
|                                                                      |
| Command:                                          NO                 |
========================================================================
```

Similarly, after the **NOMONITOR** command or the WATCH PROGRESS = 0 option, output to your terminal screen can be restored with the **MONITOR** command:

```
========================================================================
| COMPUTER TYPES:                                   YOU ENTER:         |
|                                                                      |
| Command:                                          MO                 |
========================================================================
```

MONITOR and **NOMONITOR** set the WATCH PROGRESS parameter to 1 and 0, respectively. If you want the special effects produced by setting WATCH PROGRESS to –2, –1, or 2, you must set it explicitly in your DAOPHOT.OPT file or with the **OPTIONS** command.

XIV. **SORT**

This routine will take in any stellar data file produced by DAOPHOT (*viz.* files produced by **FIND**, **PHOTOMETRY**, **PEAK**, **GROUP**, **NSTAR**, ALLSTAR, or any of the other auxiliary routines discussed below) and re-order the stars according to position within the frame, apparent magnitude, identification number, or other available datum (e.g., magnitude error, number of iterations, CHI, or SHARP). Of course, if a file produced by **GROUP** or **NSTAR** is sorted, then the association of stars into groups will be destroyed.

```
=======================================================================
| COMPUTER TYPES:                                     YOU ENTER:      |
|                                                                     |
| Command:                                            SO              |
|                                                                     |
|                                                                     |
|      The following sorts are currently possible:                    |
|                                                                     |
|   +/- 1  By increasing/decreasing star ID number                    |
|                                                                     |
|   +/- 2  By increasing/decreasing  X  coordinate                    |
|                                                                     |
|   +/- 3  By increasing/decreasing  Y  coordinate                    |
|                                                                     |
|   +/- 4  By increasing/decreasing magnitude                         |
|                                                                     |
|   +/- n  By increasing/decreasing OTHER (n <= 30)                   |
|                                                                     |
|                                                                     |
|                          Which do you want?          n               |
|                                                                     |
|                            Input file name:          filename        |
|                                                                     |
|               Output file name (default ?):          <CR>            |
|                                                  or filename         |
|                                                                     |
|           Do you want the stars renumbered?          Y or N          |
|                                                                     |
=======================================================================
```

If you answer the question, “Which do you want?” with “4”, the stars will be reordered by increasing apparent magnitude; if by “–4”, they will be reordered by decreasing apparent magnitude, and so forth. If you say that you want the stars renumbered, the first star in the new output file will be given the identification number “1”, the second star “2”, and so on. The output file will contain exactly the same data in exactly the same format as the input file — the stars will just be rearranged within the file. The “+/– n” option permits you to sort according to any of the auxiliary information in any output file — stars can thus be reordered by their sharpness or roundness indices, by their magnitudes in any aperture, by their sky brightness, by the number of iterations they required to converge in **PEAK** or **NSTAR**, or whatever. The value of n which should be entered to specify one of these items is that which is given at the bottom of each sample output file in Appendix IV below. (Note: data on the second line for a star in an .AP file are always designated by 16, 17, 18, . . ., regardless of how many apertures were used. Thus, if two apertures were used, then the magnitude in aperture 1 is number 4, the magnitude in aperture 2 is number 5, the sky brightness is number 16, the standard deviation of the sky brightness is number 17, the skewness of the sky brightness is number 18, the error in the first magnitude is 19, and the error in the second magnitude is 20. Numbers 6–15 and 21–30 are useless, in this example.)

XV. **SELECT**

When you run **GROUP** on a photometry file to create a group file suitable for input for **NSTAR**, you may find that some groups are larger than 60 stars, which is the current maximum group size allowed. The **SELECT** command allows you to cut out of the group file only those groups within a certain range of sizes, and put them in their own file.

```
================================================================
| COMPUTER TYPES:                                  YOU ENTER:        |
|                                                                    |
| Command:                                         SE                |
|                                                                    |
|                    Input group file:             filename          |
|                                                                    |
|         Minimum, maximum group size:             nn nn             |
|                                                                    |
|   Output group file (default ?.GRP):             <CR> or filename  |
|                                                                    |
|                                                                    |
|         nnn stars in nnn groups.                                   |
|                                                                    |
================================================================
```

If you want to try to reduce every star in the frame, regardless of the errors, then you will need to run **SELECT** at least twice: once with minimum, maximum group size = 1,60, and again with the same input group file, a different output group file, and minimum, maximum group size = 61,9999 (say). This latter file would then be run through GROUP again, with a larger critical overlap. (Note: You'd be better off using ALLSTAR.)

XVI. **OFFSET**

If you have a set of coordinates for objects found in one frame, and want to use these as centroids for aperture photometry in another frame, and if there is some arbitrary translational shift between the two frames, then **OFFSET** can be used to add constants to all the x- and y-coordinates in a stellar data file:

```
========================================================================
| COMPUTER TYPES:                                YOU ENTER:            |
|                                                                      |
| Command:                                       OF                    |
|                                                                      |
|                   Input file name:             filename              |
|                                                                      |
|  Additive offsets ID, DX, DY, DMAG:            n nn.nn nn.nn n.nnn   |
|                                             or n nn.nn nn.nn/         |
|                                                                      |
|   Output file name (default ?.OFF):            <CR> or filename      |
|                                                                      |
========================================================================
```

As you can tell from the example, you can also use this routine to add some (integer) constant to the ID numbers, or some (real) constant to the magnitudes. Why would you ever want to do these things? Well, if you've just run **FIND** on a star-subtracted image, you'll probably want to add 50000 or something to the ID numbers before appending this new list to the original star list for the frame, so there will be no doubt which star number 1 is referred to in the .LST file. If you have a list of artificial stars that you've just added to one frame of a field, you may want to offset the positions and instrumental magnitudes so that you can add exactly the same stars into another frame of the same field. Just for instance.

An unformatted READ is used to input these data, so the numbers may be separated by spaces or commas, and the list can be terminated with a slash ("/") if you want all remaining numbers to be zero.

Use of this routine for transferring the starlist from one frame to another is not recommended in crowded fields, particularly when the two frames were taken in different photometric bandpasses. You really need to know about all of a star's neighbors in order to reduce it properly, and different neighbors may be prominent in frames taken in greatly different colors. It would be better simply to run **FIND** on each frame, and to match up the stars you are interested in after the photometry is done.

XVII. **APPEND**

APPEND provides a simple way for the user to concatenate any two of the stellar data files which DAOPHOT has written to the disk. A special DAOPHOT command has been written to perform this function, because if the user were to leave DAOPHOT and use the operating system's COPY, MERGE, or APPEND command (or equivalent), it would then be necessary to use an editor to remove the extra file header from the middle of the newly created file. **APPEND** does this for you automatically.

```
========================================================================
| COMPUTER TYPES:                                  YOU ENTER:          |
|                                                                      |
| Command:                                         AP                  |
|                                                                      |
|                  First input file:               filename            |
|                                                                      |
|                 Second input file:               filename            |
|                                                                      |
|                                                                      |
|       Output file (default ?.CMB):               <CR> or filename    |
|                                                                      |
========================================================================
```

Note that **APPEND** does no checking to ensure that the input files are of the same type — the user is perfectly able to **APPEND** output from **FIND** onto output from **PHOTOMETRY**. The resulting hybrid file would be illegible to most other DAOPHOT routines, and might cause a crash if DAOPHOT tried to read it in. Note further that the **GROUP** command (above) makes a point of leaving a blank line at the end of every group file, so that the groups will remain distinct when **APPEND**ed. If in editing group files you delete that last blank line, then when **APPEND**ing those files the last group of the first input file and the first group of the second input file will be joined together in the output file.

XVIII. **DUMP**

DUMP permits you to display on your terminal screen the raw values in a specified square subarray of your picture. This is useful for figuring out why DAOPHOT isn't working with your data (maybe all the pixels have intensities of –32768?), or in deciding whether a particular star has a central intensity above the maximum value where your detector behaves linearly.

Example:

```
========================================================================
| COMPUTER TYPES:                                   YOU ENTER:         |
|                                                                      |
| Command:                                          DU                 |
|                                                                      |
|                                    Box size:      9                  |
|                                                                      |
|              Coordinates of central pixel:        201,378            |
|                                                                      |
+----------------------------------------------------------------------+
| COMPUTER TYPES:                                                      |
|                                                                      |
|              197   198   199   200   201   202   203   204   205    |
|          +----------------------------------------------------      |
|       383 |  543   556   600   633   643   630   589   538   514    |
|       382 |  581   644   760   884   930   865   732   623   570    |
|       381 |  651   864  1248  1823  2062  1657  1116   800   626    |
|       380 |  775  1303  2791  5995  7442  4802  2166  1096   732    |
|       379 |  916  1955  5933 16430 22029 11974  4104  1526   846    |
|       378 |  977  2259  6364 16623 23622 13658  4751  1762   933    |
|       377 |  936  1836  3878  7751 10436  7269  3380  1611   949    |
|       376 |  798  1217  1963  3179  3815  3109  2006  1273   861    |
|       375 |  656   847  1138  1563  1778  1621  1300  1003   790    |
|       374 |  602   682   798   962  1090  1073   959   824   698    |
|       373 |  550   587   653   729   801   807   782   705   638    |
|                                                                      |
|                  Minimum, median, maximum:    570   1248  23622       |
|                                                                      |
+----------------------------------------------------------------------+
|                                                                      |
| To get out ...                                                       |
|                                                                      |
| COMPUTER TYPES:                                       YOU ENTER:     |
|                                                                      |
|              Coordinates of central pixel:            0,0            |
|                                                    or CTRL-Z         |
|                                                                      |
========================================================================
```

(I just happened to know that there is a bright star centered near 201,378.) The user specifies a box size that will conveniently fill his screen, without any wraparound; with the column and row ID's across the top and down the left side, a box 12 pixels on a side is the largest that can be accomodated on an 80-column terminal screen, while a box 21 pixels on a side is the largest that can be fit into a 24×132 screen.

Responding to the “Coordinates of central pixel:” prompt with a position that is outside the picture or with a CTRL-Z will return you to DAOPHOT command mode.

XIX. **FUDGE**

Although it is morally wrong, someday there may come a time when you just *have* to fudge some of your image data. Suppose, for instance, that you are trying to derive a point-spread function from the sole acceptable star in the frame, and way, way out in the corner of the box wherein the point-spread function is to be defined there is a cosmic-ray hit. If you do nothing, then the cosmic ray will produce a spike in the point-spread function which will generate a hole whenever this PSF is used to subtract stars. *In such a desperate case it may be* **slightly** *the lesser of two evils to fudge the bad datum*, which DAOPHOT's **FUDGE** routine will do. Let us assume that from your image display you have ascertained that the cosmic ray occupies the two pixels (267,381) and (268,381); let us further suppose that you see from the aperture photometry for this star that the sky background brightness in its neighborhood is 893 ADU. Then:

```
========================================================================
| COMPUTER TYPES:                                   YOU ENTER:         |
|                                                                      |
| Command:                                          FU                 |
|                                                                      |
|  Name for output picture (default ?f.DST):        <CR> or filename   |
|                                                                      |
|                          Border (pixels):         0                  |
|                                                                      |
|                First, last column number:         267,268            |
|                                                                      |
|                   First, last row number:         381,381            |
|                                               or  381/               |
|                                                                      |
|                         Brightness value:         893                |
|                                                                      |
|                                                                      |
|                First, last column number:         CTRL-Z             |
|                                                                      |
| Command:                                                             |
|                                                                      |
========================================================================
```

As may be inferred from the example, **FUDGE** allows you to insert any constant brightness value into any rectangular subsection of an otherwise exact copy of the image. Alternatively, . . .

```
=======================================================================
| COMPUTER TYPES:                                   YOU ENTER:        |
|                                                                     |
| Command:                                          FU                |
|                                                                     |
|     Name for output picture (default ?f.DST):     <CR> or filename  |
|                                                                     |
|                                Border (pixels):   n                 |
|                                                                     |
| Polynomial order (0=constant, 1=plane, etc.):     n                 |
|                                                                     |
|                      First, last column number:   267,268           |
|                                                                     |
|                         First, last row number:   381,381           |
|                                                or 381/              |
|                                                                     |
|                                                                     |
|                      First, last column number:   CTRL-Z            |
|                                                                     |
| Command:                                                            |
|                                                                     |
=======================================================================
```

In this case you don't want to insert a single constant brightness value and/or you don't know what value you would like to insert. Instead you are asking the routine to consider a border n pixels wide around the rectangular area you have specified, and use a least-squares polynomial surface to interpolate values into the fudged region. Now for the tricky bit. The routine does not fit a single polynomial surface to the border and then use that polynomial to predict values for all the pixels in the fudged region. Oh, no. Instead, for each pixel in the rectangle to be fudged it fits a *different* polynomial surface to the border pixels, employing $1/r^4$ weights. Thus, pixels in the corners and near the edges of the rectangular fudge region will closely reflect the data values and gradients in the pixels next to them and will be minimally affected by the gross gradient across the gap; pixels in a long rectangular region will be affected by the pixels next to them and less by the pixels at the far ends. Thus, even a "constant" or a "plane" polynomial order will produce some complex surface that flows smoothly between the borders, and doesn't have discontinuous values or gradients at the edges. This is quite a bit slower than a simple surface fit, so keep the border narrow — only a couple-three pixels — but I think you'll be pleased with the results.

Use your fudged image to generate a point-spread function free of the cosmic-ray spike, *then delete it before anyone finds out what you have done!*

XX. **ADDSTAR**

This routine is used to add synthetic stars, either placed at random by the computer, or in accordance with positions and magnitudes specified by you, to your picture. They can then be found by **FIND**, reduced by **PHOTOMETRY** and the rest, and the star-finding efficiency and the photometric accuracy can be estimated by comparing the output data for these stars to what was put in.

Example 1: Random star placement

```
=================================================================
| COMPUTER TYPES:                                 YOU ENTER:       |
|                                                                  |
| Command:                                        AD               |
|                                                                  |
|        File with the PSF (default ?.PSF):       <CR> or filename |
|                                                                  |
|                       Seed (any integer):       n                |
|                                                                  |
|                          Photons per ADU:       nn.n             |
|                                                                  |
|   Input data file (default RANDOM STARS):       <CR>             |
|                                                                  |
|                                                                  |
|              Magnitude of PSF star is nn.nnn                     |
|                                                                  |
|                                                                  |
|      Minimum, maximum magnitudes desired:       15.0 18.0        |
|                                                                  |
|     Number of stars to add to each frame:       100              |
|                                                                  |
|                     Number of new frames:       5                |
|                                                                  |
|                           File-name stem:       FAKE             |
|                                                                  |
=================================================================
```

This will produce five different, new data frames, each containing 100 artificial stars with instrumental magnitudes between 15.00 and 18.00 mag. added to what was already there. The five frames will be named FAKE01.DST, . . ., FAKE05.DST. There will also be created five files named FAKE01.ADD, . . ., FAKE05.ADD containing the x,y-positions and the magnitudes of the new stars. I have you specify a seed for the random number generator so that (a) if you lose the images, by specifying the same seed you can make them again, and (b) exactly the same artificial image will be created on any computer, provided the same seed is used.

Example 2: Deliberate star placement

```
================================================================
| COMPUTER TYPES:                                  YOU ENTER:         |
|                                                                     |
| Command:                                         AD                 |
|                                                                     |
|        File with the PSF (default ?.PSF):        <CR> or filename   |
|                                                                     |
|                       Seed (any integer):         n                  |
|                                                                     |
|                          Photons per ADU:         14.1               |
|                                                                     |
|   Input data file (default RANDOM STARS):         ARTSTAR.MAG        |
|                                                                     |
|   Output picture name (default ARTSTARa):         <CR>               |
|                                                                     |
================================================================
```

This presumes that by some means you have already created on the disk a file named "ARTSTAR.MAG" (or whatever) which contains centroid positions and instrumental magnitudes for the stars you want added to the picture. This permits you to become just as sophisticated with your artificial-star tests as you want — you can simulate any color-magnitude diagram, luminosity function, and spatial distribution in the frame that you want, just by writing yourself a program which creates the necessary input files.

Note that in both examples the code asks for a number of photons per ADU. It uses this to add the appropriate Poisson noise to the star images (the correct amount of readout noise already exists in the frame). For realistic tests you should specify the correct value for this number. If for some reason you would like to have the scaled PSF added into the image *without* extra Poisson noise, just specify some enormous number of photons per ADU, such as 99999.

To avoid small number statistics in your artificial-star analysis, you should create a number of different frames, each containing only a few extra stars. If you try to add too many stars at once your synthetic frames will be significantly more crowded than your original frame, making it difficult to apply the artificial-star conclusions to your program-star results.

XXI. **LIST**

If at your installation the standard disk format for digital images to be reduced with DAOPHOT is the Caltech data-structure file (as it is on the VMS machines at the DAO), then the **LIST** command will enable you to examine the contents of the image header. For instance, let us suppose that your images originated as FITS files from Kitt Peak or Cerro Tololo, and that you want to learn the right ascension, declination, sidereal time, and integration time of your image. It happens that all this information is contained in the OBS substructure of the Caltech data structure, so. . .

```
========================================================================
| COMPUTER TYPES:                                   YOU ENTER:         |
|                                                                      |
| Command:                                          LI                 |
|                                                                      |
|           File = ?.DST                                               |
|                                                                      |
|      Components: OBS                                                 |
|                  Z                                                   |
|                                                                      |
| LIST>                                             OBS.RA             |
|                                                                      |
|           ' 9:23:41'  / right ascension                              |
|                                                                      |
| LIST>                                             OBS.DEC            |
|                                                                      |
|           '-77: 4:48' / declination                                  |
|                                                                      |
| LIST>                                             OBS.ST             |
|                                                                      |
|           '12:53:46'  / sideral time                                 |
|                                                                      |
| LIST>                                             OBS.ITIME          |
|                                                                      |
|           150.0                                                      |
|                                                                      |
| LIST>                                             <CR> or CTRL-Z     |
|                                                                      |
| Command:                                                             |
|                                                                      |
========================================================================
```

If you don't happen to remember the FITS keyword for the particular information you want, respond to the "LIST>" prompt with just "OBS" and all the keywords will be typed to your terminal; just taken note of the one you want as it flies past.

I have not yet gotten around to implementing this on the Unix side. There, the **LIST** command merely reminds you of the name of the image you are working on and returns you to DAOPHOT Command: mode.

XXII. **HELP**

This routine simply produces an alphabetical listing of the currently defined commands on your computer terminal; it does not allow you to obtain detailed information on any of the routines. It is included primarily as a memory-jogger in case you momentarily forget what some routine is called.

```
========================================================================
| COMPUTER TYPES:                                    YOU ENTER:        |
|                                                                      |
| Command:                                           HE                |
|                                                                      |
+----------------------------------------------------------------------+
| COMPUTER TYPES:                                                      |
|                                                                      |
| The commands currently recognized are:                               |
|                                                                      |
|     ADDSTAR      APPEND       ATTACH       DUMP         EXIT         |
|     FIND         FUDGE        GROUP        HELP         LIST         |
|     MONITOR      NOMONITOR    NSTAR        OFFSET       OPTION       |
|     PEAK         PHOTOMETRY   PICK         PSF          SELECT       |
|     SKY          SORT         SUBSTAR                                |
|                                                                      |
| Any command may be abbreviated down to its first two characters.     |
|                                                                      |
========================================================================
```

XXIII. **EXIT**

This command allows you to exit cleanly from DAOPHOT, with all files properly closed and everything nice and neat.

```
========================================================================
| COMPUTER TYPES:                                      YOU ENTER:      |
|                                                                      |
| Command:                                             EX              |
|                                                                      |
| Good bye.                                                            |
|                                                                      |
========================================================================
```

E. ALLSTAR

Unlike everything else in this manual, ALLSTAR is not a routine within DAOPHOT II, which can be executed in response to a "Command:" prompt. Rather, ALLSTAR is a separate stand-alone program which one executes directly from the operating system. I have done it this way because that makes it easier to conserve virtual memory, so that DAOPHOT and ALLSTAR can both operate on the largest possible images.

In general, ALLSTAR works pretty much the same as the **NSTAR** routine in DAOPHOT, fitting multiple, overlapping point-spread functions to star images in your CCD frames. Input and output images and data files are fully compatible with those produced by DAOPHOT. Some of the noteworthy differences between ALLSTAR and **NSTAR**:

1. ALLSTAR reduces the entire starlist for a frame at once (current maximum: 15,000 stars). With every iteration, ALLSTAR subtracts *all* the stars from a working copy of your image according to the current best guesses of their positions and magnitudes, computes increments to the positions and magnitudes from examination of the subtraction residuals around each position, and then checks each star to see whether it has converged or has become insignificant. When a star has converged, its results are written out and the star is subtracted permanently from the working copy of the image; when a star has disappeared it is discarded. In either case, the program has a smaller problem to operate on for the next iteration. Throughout this process, ALLSTAR maintains a noise map of the image, including knowledge of the Poisson statistics of stars that have previously converged and been permanently subtracted from the working copy. Since the entire star list is a "group" (in the sense of **NSTAR**), ALLSTAR does not require that the starlist be **GROUP**ed ahead of time, and is perfectly happy to run from your .AP files.
2. In order to make the problem tractable (after all, we're potentially dealing with a 45,000×45,000 matrix with every iteration), ALLSTAR does associate stars with one another in order to make the big matrix block-diagonal, so it can be inverted in a finite amount of time. These associations are temporary, they are reconsidered every iteration, and they do not compromise the full, rigorous, simultaneous, least-squares solution of the entire star-list. ALLSTAR will do the best it can to reduce your data frame regardless of the degree of crowding. What happens is the following: during the process of each iteration, ALLSTAR automatically associates all the stars into the frame into manageable bite-sizes, on the basis of a critical separation which is calculated from the fitting radius. If you have the "WATCH PROGRESS" option equal to 1, then in a densely-packed frame, you might notice ALLSTAR typing out messages like

   "Group too large: 107 (2.50)."

   That means that a group of 107 stars resulted from use of the nominal critical separation (2.50 pixels, in this case). ALLSTAR will attempt to break up this group by regrouping it with smaller and smaller critical separations until it falls apart into subgroups each containing fewer than some user-specified number of stars. (Other associations, which are already smaller than the maximum size, will *not* be regrouped with the smaller critical separation. Only the ones that are too big will.) If the group is so dense that the critical separation drops to less than 1.2 pixels, the program will arbitrarily delete the faintest star in the group, and proceed from there. By increasing the value of the optional parameter MAXIMUM GROUP SIZE, the user may reduce the number of faint stars that get rejected by this mechanism; on the other hand, these stars will be so poorly measured that they may not be *worth* retaining. Of course, increasing the MAXIMUM GROUP SIZE will also greatly increase the amount of time required for reducing the frame, since the reduction time goes roughly as the cube of the size of the largest associations.

3. ALLSTAR will produce the star-subtracted image for you directly, without need to run the DAOPHOT routine **SUBSTAR**. (If you don't want the output picture produced, when the program asks you for the star-subtracted image filename respond with a CTRL-Z or type the words END OF FILE, in capitals.)

4. If you think that the stars' positions are very well known ahead of time — for instance because you have averaged their observed positions on a number of frames, or because you can apply positions derived from an excellent-seeing frame to poorer frames — then it is possible to tell ALLSTAR not to attempt to adjust the stellar positions you give it, but just to solve for the magnitudes. If you are very, very careful, this can produce more accurate photometry for your stars than otherwise. But *please be careful!* Remember that if you have frames taken at different airmasses or in different photometric bandpasses, then stars' positions in the various frames will not be related by simple zero-point offsets. There may also be slight rotations, scale changes, compression along the direction toward the zenith, and shifts dependent upon the stars' intrinsic colors (due to atmospheric dispersion). This, then, is an option to use only if you are sure your predicted positions correctly include these effects.

5. Unlike **NSTAR**, ALLSTAR II is now empowered to redetermine sky values for the stars in the frame. This is done as follows: the user sets the optional parameters OS (= "Outer sky radius") > 0 and IS (= "Inner sky radius") $<$ OS. *These need not — and probably shouldn't — be the same as the sky radii that you used in the PHOT routine in DAOPHOT.* Here OS should be large compared to the FITTING RADIUS — if possible, comparable to or somewhat larger than the PSF RADIUS — but definitely smaller than the spatial scales of any significant variations in the sky background. The inner sky radius, IS, can be quite small: comparable to or less than the FITTING RADIUS, or even zero. It is probably best to set IS to at least one or two pixels, to keep the enhanced photon noise and the uncertain PSF near the center of a star image from overly influencing the derived sky values. As long as OS $>$ IS, before every third iteration (starting with iteration 3), after *all* the stars have been subtracted from the working copy of the image, the program will redetermine the median brightness value in the annulus bounded by these two radii. This is used as the sky estimate for the star for the next three iterations. In my opinion, this is just about the best way currently available to get at the ever-sought, ever-elusive *SKY UNDER THE STAR!* I do it this way rather than solving for the sky brightness (or some analytic function representing the spatial distribution of the sky brightness) as part of the least-squares profile fits, because it is far, far easier, faster, and more precise to determine the median of several hundred pixels than to fit a robust least-squares surface to a few dozen. This is discussed at some length in Stetson, 1987 PASP, 99, 101, §III.D.2.a. Perhaps in ALLSTAR II.v (decimal Roman numerals — far out) I will include optional fitting of a sky model in the least-squares problem. If OS $\leq$

IS, ALLSTAR will continue to use the sky-brightness values that were in the input data file.

Like DAOPHOT, ALLSTAR begins by showing you a table of optional parameter settings; unlike DAOPHOT, (but like **PHOTOMETRY**) it immediately gives you a chance to change any you don't like.

```
=========================================================================
           FITTING RADIUS =  2.50       CE (CLIPPING EXPONENT) =  6.00
    REDETERMINE CENTROIDS =  1.00          CR (CLIPPING RANGE) =  2.50
           WATCH PROGRESS =  1.00           MAXIMUM GROUP SIZE = 50.00
     PERCENT ERROR (in %) =  0.75         PROFILE ERROR (in %) =  5.00
    IS (INNER SKY RADIUS) =  0.00        OS (OUTER SKY RADIUS) =  0.00
 OPT>
=========================================================================
```

The FITTING RADIUS and the WATCH PROGRESS options you are familiar with from DAOPHOT: the fitting radius is the size of the region around each star which will actually be used in performing the profile fits. The WATCH PROGRESS option determines just how much garbage will be typed onto your terminal screen or written into your batch job's logfile. The MAXIMUM GROUP SIZE option has been explained above. The REDETERMINE CENTROIDS, CLIPPING EXPONENT, and CLIPPING RANGE options are new.

The REDETERMINE CENTROIDS option is how you tell the program whether to improve the stars' positions in the profile fits. RE = 0 means "no", 1 means "yes." If you enter "no," the program will assume that the positions of the stars are known with absolute accuracy, and will redetermine only the stars' magnitudes (this is one way of imposing a star list from a good frame onto a poor frame of the same region); if "yes," you will get the full-blown astrometric and photometric reduction you are used to from **NSTAR**.

The CLIPPING EXPONENT and CLIPPING RANGE options are explained in Stetson, 1987 PASP, 99, 191, §III.D.2.d, "Resisting bad data". The CLIPPING RANGE is variable a in the formula given there, and the CLIPPING EXPONENT is b. The clipping exponent you specify will be rounded to the *nearest integer*. I have given the default values RANGE = 2.5 (i.e., a 2.5–σ residual gets half weight), and EXPONENT = 6.0, because in my own experiments they seem to work reasonably well. I do not have a profound understanding of some fundamental way to obtain "best" values for these parameters; this subject still needs *much* more experimentation (by me and, if you want, by you). Anyway, on the basis of whatever religious tenets you hold, adopt values for these parameters. Experiment with them only if you are prepared to burn up a *lot* of CPU time. If you are thoroughly conservative, setting CLIPPING EXPONENT to 0.0 turns the clipping off altogether.

Your own default parameter values may be set by creating a file named ALLSTAR.OPT (allstar.opt under Unix) in your directory. It works exactly the same as the DAOPHOT.OPT file does in DAOPHOT, except of course that it should include only those ten parameters recognizable to ALLSTAR.

The rest of it's pretty darn trivial.

```
=======================================================================
| COMPUTER TYPES:                                  YOU ENTER:          |
|                                                                      |
|                              Input image name:   filename            |
|                                                                      |
|       Object name from file header                                   |
|                                                                      |
|                    Picture size: nnn nnn                             |
|                                                                      |
|             File with the PSF (default ?.PSF):   <CR> or filename    |
|                                                                      |
|                     Input file (default ?.AP):   <CR> or filename    |
|                                                                      |
|              File for results (default ?.ALS):   <CR> or filename    |
|                                                                      |
|    Name for subtracted image (default ?s.DST):   <CR> or filename    |
|                                                                      |
=======================================================================
```

And away we go! With each iteration, the program will keep you updated on how many stars remain to be reduced, how many have disappeared due to insignificance, and how many have converged and been written out.

Finally, I would like to reiterate that in the final analysis I wrote ALLSTAR, like DAOPHOT, for *me*. I make no promise to leave it alone or to notify you of minor changes. If I discover some major bug that destroys the science, I may remember that I gave you a copy and then again I may not. As I get time to play, I will certainly be attempting to make the program more powerful and reliable for *my* applications. Therefore, in some sense you use this copy of the program (and of DAOPHOT II) at your own risk. Use it as long as you are happy with the results. If you don't like what you are getting, stop using the program and complain to me. Maybe I will have already fixed your problem, or maybe your problem will be interesting or important enough that I will want to fix it for you. However, I get bent all out of shape when somebody has a problem with my software and publishes complaints in the literature, without ever coming to *me* to give me a chance to fix it for them, or to explain some point they may have misunderstood.

APPENDIX I

Optional parameters

ID	Description (Note)	Routines Affected	Permitted values	Default value
RE	Readout noise, 1 exposure (ADU) (1)	FIND	positive	0
GA	Gain, 1 exposure (photons per ADU) (1)	FIND	positive	0
LO	Low good datum (standard deviations) (1)	FIND	non-negative	7
HI	High good datum (ADU) (1)	FIND	non-negative	32766.5
FW	FWHM of objects for which FIND is to be optimized (in pixels)	FIND	0.2 - 15.0	2.5
TH	Significance threshold for detection (standard deviations)	FIND	non-negative	4.0
LS	Low sharpness cutoff	FIND	0.0 - 1.0	0.2
HS	High sharpness cutoff	FIND	0.6 - 2.0	1.0
LR	Low roundness cutoff	FIND	-2.0 - 0.0	-1.0
HR	High roundness cutoff for the profile fits.	FIND	0.0 - 2.0	1.0
WA	Watch progress of reductions on terminal?	FIND,PHOT,PEAK, PSF,NSTAR, SUBSTAR,SORT	-2 - 2	1
FI	The fitting radius (in pixels)	PSF,PEAK, GROUP,NSTAR	1.0 - 10.0	2.0
PS	PSF radius: radius (in pixels) within which the point-spread function is to be defined. (2)	PSF	1.0 - 35.0	11.0
VA	Degree of variation in the PSF (2)	PSF	-1 - 2	0
FR	***** NOT IMPLEMENTED *****			
AN	Which analytic formula for PSF (2)	PSF	1 - 6	1
EX	How many passes to clean discordant pixels from the PSF table(s)	PSF	0 - 9	0
PE	Percent error (e.g. flat-field)	PEAK,NSTAR	0 - 100	0.75
PR	Profile error (inadequate PSF)	PEAK,NSTAR	0 - 100	5.0

Notes:

(1) **FIND** is the only routine where these values are read from the options table. However, the file headers carry these numbers along to other routines, which will also be affected.

(2) **PSF** is the only routine where these values are read from the options table. However, the .PSF file will carry these numbers along to other routines, which will then act accordingly.

APPENDIX II

The **FIND** Threshold: Theory and Practice

Assume that a given frame has an average sky brightness of s (in units of ADU), a gain factor of p photons per ADU, and a readout noise per pixel of r (in electrons or, equivalently, photons. Note that since "electrons" and "photons" are simply number counts, they really have no physical dimensions — this is inherent in the use of Poisson statistics, where $\sigma(N) = \sqrt{N}$ would be meaningless if any physical dimensions were involved.) The expected random noise per pixel may now be computed as follows:

IN UNITS OF PHOTONS:

If sky brightness in photons $= p \times s$(ADU), then for the…
Poisson statistics of the sky:

$$\begin{aligned}\text{Variance (sky)} &= \sigma^2(\text{sky}) \\ &= \text{number of photons} && \text{[Poisson statistics]} \\ &= p \times s && \text{(variance is in dimensionless units.)}\end{aligned}$$

Readout noise:

$$\begin{aligned}\text{Variance (readout)} &= \sigma^2(\text{readout}) \\ &= r^2 && \text{(dimensionless units) [by definition]}\end{aligned}$$

Total noise:

$$\begin{aligned}\text{Variance(total)} &= \text{variance(sky)} \\ &\quad + \text{variance(readout)} && \text{[propagation of error]} \\ &= p \times s + r^2 && \text{(dimensionless units)} \\ \text{standard deviation} &= \sqrt{p \times s + r^2} && \text{(dimensionless units)}\end{aligned}$$

(Note that in this equation p has units of photons/ADU, s has units of ADU, and r has units of electrons or photons. This is very clumsy, but that's the way we usually think of these numbers.)

IN UNITS OF ADU:

$$\sigma(\text{ADU}) = \frac{\sigma(\text{photons})}{(\text{photons per ADU})} \qquad \text{[propagation of error]}$$

Let R = readout noise in ADU $\equiv r/p$. Then

$$\sigma(\text{total}) = \sqrt{s/p + R^2} \qquad \text{(units are ADU).}$$

Please note that if the frame you are working on is the average or the sum of several raw data frames, the values of the gain factor (in photons per ADU) and the readout noise will have to be adjusted accordingly:

	If N frames were averaged	If N frames were summed
photons/ADU	p(N) = N*p(1)	p(N) = p(1)
r-o noise	R(N) = R(1)/SQRT(N)	R(N) = R(1)*SQRT(N)
total	std. dev. (N) = SQRT(s/[N*p(1)] + [R**2]/N)	std. dev. (N) = SQRT(s/p(1) + N*[R(1)]**2)

The value of *s* which **FIND** uses in these equations is the number you get when you issue the DAOPHOT command "**SKY**".

Note that the expectation value of *s* scales as follows:

If N frames were averaged	If N frames were summed
s(N) = s(1)	s(N) = N*s(1)
std. dev.(N) = SQRT (s(1)/[N*p(1)] + [R(1)]**2/N)	std. dev.(N) = SQRT (s(1)*N/p(1) + N*[R(1)/p(1)]**2)
= std. dev.(1)/SQRT(N)	= std. dev.(1)*SQRT(N)

Therefore, to ensure that the statistics are being done properly, for your frames **FIND** takes the readout noise per frame IN UNITS OF ADU, and the ratio of photons per ADU, and corrects them for the number of frames that have been averaged or summed according to the first table above.

TO ESTABLISH A REASONABLE THRESHOLD FOR YOUR STAR-FINDING:

FIND computes the random error per pixel *in units of ADU* from

$$\text{random error in 1 pixel} = \sqrt{\frac{s}{p_N} + R_N^2}$$

where s is the number you get from **SKY** and appropriate values of p_N and R_N have been computed according to the table above. When you then run **FIND**, you will discover that it also gives you a number called "relative error". This is because in trying to establish whether there is a star centered in a certain pixel, **FIND** operates on weighted sums and differences of several adjacent pixels. The "relative error" is merely a scaling parameter — it is the number that the standard error of one pixel must be multiplied by to obtain the standard error of the smoothed/differenced data in the convolved picture. Therefore, to arrive at a reasonable threshold, **FIND**, computes the standard error per pixel as described above, multiplies it by the "relative error" factor, and sets the threshold at the multiple of this number which you have specified with the "THRESHOLD" option, say,

$$\text{Threshold} = 3.5 \times (\text{relative error}) \times (\text{standard error in one pixel})$$

for 3.5-σ detections. (A +3.5-σ excursion occurs about 233 times per one million independent random events. In an otherwise empty 300 $\times$ 500 frame this would produce about 35 false detections. In a frame some non-trivial fraction of whose area was occupied by real stars, the false detections would be fewer.)

TO SUMMARIZE:

(1) Ascertain the values of the readout noise (in electrons or photons) and the number of photons per ADU for a *single frame* for the detector you used. Specify these as *options*. **FIND** will

(2) compute the readout noise in ADU:

$$R(1\text{ frame; ADU}) = \frac{r(1\text{ frame; photons})}{p(1\text{ frame; photons/ADU})};$$

(3) correct the ratio of photons per ADU and the readout noise (in ADU) for the number of frames that were averaged or summed:

	If N frames were averaged	If N frames were summed
photons/ADU	p(N) = N*p(1)	p(N) = p(1)
r-o noise	R(N) = R(1)/SQRT(N)	R(N) = R(1)*SQRT(N)

(4) determine the typical sky brightness, s, in your data frame, by using the DAOPHOT command **SKY**;

(5) compute the random error per pixel:

$$\text{random error in 1 pixel} = \sqrt{s/p_N + R_N^2};$$

(6) multiply by the relative error defined above to arrive at the random noise in the sky background of the *convolved* data frame:

$$\text{background noise} = (\text{relative error}) \times (\text{random error per pixel})$$

Some multiple (of order 3 – 5, say) of this background noise, as specified by the user) is used as your desired star-detection threshold.

APPENDIX III

DERIVING A POINT-SPREAD FUNCTION IN A CROWDED FIELD

Obtaining a good point-spread function in a crowded field is a delicate business, so please do not expect to do it quickly — plan on spending a couple of hours in the endeavor the first few times you try it. After that it gets easier, and once you know what you're trying to do, it's fairly easy to write command procedures to handle major parts of the FIT–SUBTRACT–MAKE A NEW PSF–FIT AGAIN–. . . loop. I recommend that you proceed *approximately* as follows:

Invoke DAOPHOT and:

(1) Run **FIND** on your frame.

(2) Run **PHOTOMETRY** on your frame.

(3) **SORT** the output from **PHOTOMETRY** in order of increasing apparent magnitude (decreasing stellar brightness), with the renumbering feature. This step is optional, but it can be more convenient than not.

(4) **PICK** to generate a set of likely PSF stars. How many stars you want to use is a function of the degree of variation you expect, and the frequency with which stars are contaminated by cosmic rays or neighbor stars. I'd say you'd want a rock-bottom minimum of three stars per degree of freedom, where the degrees of freedom are 1 (constant PSF), 3 (linearly varying PSF), and 6 (quadratically varying PSF). I'm referring here to the number of degrees of freedom you expect you'll need *ultimately*. PSF stars are weighted according to their magnitudes, so it doesn't hurt to include many faint (but good) stars along with a few bright (but good) ones. *The more crowded the field, the more important it is to have many PSF stars, so that the increased noise caused by the neighbor-subtraction procedure can be beaten down.* Furthermore, if you intend to use the variable-PSF option, it is vital that you have PSF stars spread over as much of the frame as possible. I don't feel particularly bad if I end up using as many as 25 or 50 PSF stars in such a case, but maybe I'm too cautious.

(5) Run **PSF**, tell it the name of your complete (sorted, renumbered) aperture photometry file, the name of the file with the list of PSF stars, and the name of the disk file you want the point-spread function stored in (the default should be fine).

   (a) If you have the "WATCH PROGRESS" option equal to 1 or 2, **PSF** will produce on your terminal a gray-scale plot of each star and its environs, it will tell you the number of ADU in the brightest pixel within the area displayed, and then it will ask whether you wish to include this star in the point-spread function, to which you should answer "Y" or "N", as appropriate.

   (b) If "WATCH PROGRESS" is 0, it will go ahead and use the star, unless it finds a bad pixel more than one fitting radius and less than (one PSF radius plus 2 pixels) from the star's center; in such a case it will ask whether you want the star.

   (c) If "WATCH PROGRESS" = −1 or −2, it will use the star, regardless, counting on the bad data rejection to clean out any bad pixels. It will *report* the presence of bad pixels, but it will use the star anyway.

If the frame is crowded, it is probably worth your while to generate the first PSF with the "VARIABLE PSF" option set to –1 — pure analytic PSF. That way, the companions will not generate ghosts in the model PSF that will come back to haunt you later. You should also have specified a reasonably generous fitting radius — these stars have been preselected to be as isolated as possible, and you want the best fits you can get. But remember to avoid letting neighbor stars intrude within one fitting radius of the center of any PSF star.

(6) Run **GROUP** and **NSTAR** or ALLSTAR on your .NEI file. If your PSF stars have many neighbors this may take some minutes of real time. Please be patient (or submit it as a batch job and perform steps 1 – 5 on your next frame while you wait).

(7) After **NSTAR** is finished, run **SUBSTAR** to subtract the stars in the output file from your original picture. This step is unnecessary if you used ALLSTAR. (And why didn't you?)

(8) **EXIT** from DAOPHOT and send this new picture to the image display. Examine each of the PSF stars and its environs. Have all of the PSF stars subtracted out more or less cleanly, or should some of them be rejected from further use as PSF stars? (If so, use a text editor to delete these stars from the .LST file.) Have the neighbors mostly disappeared, or have they left behind big zits? Have you uncovered any faint companions that **FIND** missed? If the latter, then

  (a) use the cursor on your image display to measure the positions of the new companions;

  (b) use your system's text editor to create a .COO file containing star ID numbers which you invent and the (x, y) coordinates of the new stars [FORMAT (1X, I5, 2F9.?)];

  (c) re-enter DAOPHOT, **ATTACH** the original image, run **PHOTOMETRY** to get sky values and crude magnitudes for the faint stars;

  (d) run **PEAK**, **GROUP** + **NSTAR**, or ALLSTAR to get slightly improved positions and magnitudes for the stars; and

  (e) **APPEND** this .PK, .NST, or .ALS file to the one generated in step (6).

  (f) Using the original picture again, run **GROUP** + **NSTAR** or ALLSTAR on the file created in step 8(e).

(9) Back in DAOPHOT II, **ATTACH** the original picture and run **SUBSTAR**, specifying the file created in step 6 or in step 8(f) as the stars to subtract, and the stars in the .LST file as the stars to keep. You have now created a new picture which has the PSF stars still in it, but from which the known neighbors of these PSF stars have been mostly removed. If the PSF was made with "VARIABLE PSF" = –1, the neighbors are maybe only 90 or 95% removed, but still that's a big gain.

(10) **ATTACH** the new star-subtracted frame and repeat step (5) to derive a new point-spread function. This time you should have "VARIABLE PSF" = 0 (unless the pure analytic PSF did a great job of erasing the stars). Specify the file which you created in step 6 or 8(f) as the input file with the stellar photometry, since this file has the most recent and best estimates for the positions and magnitudes of the PSF stars and their neighbors. If the zits left behind when some of the neighbor stars were subtracted trigger the "bad pixel" detector, answer "Y" to the question

"Try this one anyway?" and count on the bad-pixel rejection to fudge them away (if "EXTRA PSF CLEANING PASSES" is greater than zero; otherwise you're counting on the large number of PSF stars to beat the zits down in the average profile).

(11+...) Run **GROUP** + **NSTAR** or ALLSTAR on the file you created in step 6 or 8(f); loop back to step (5) and iterate as many times as necessary to get everything to subtract out as cleanly as possible. Remember that each time through the loop you should be obtaining the new point-spread function from a frame in which the neighbors (but *not* the PSF stars) have been subtracted, while **NSTAR** or ALLSTAR should be run on the original picture, with all the stars still in it (except when you are trying to get crude data for new stars you have just identified by hand and eye on the image display). Increase the "VARIABLE PSF" option by one per iteration, if the neighbors are subtracting out relatively cleanly. Once you have produced a frame in which the PSF stars and their neighbors all subtract out cleanly, one more time through PSF should produce a point-spread function you can be proud of.

APPENDIX IV

DATA FILES

DAOPHOT II writes its output data to disk in ordinary ASCII sequential-access files which may be TYPEd, PRINTed, and EDITed with your operating system's utilities, and may be shipped freely from one machine to another. Files produced by the different routines have some things in common. The first of these is a three-line file header which, in its most elaborate form, looks like this:

```
=====================================================================
 NL    NX    NY  LOWBAD HIGHBAD  THRESH     AP1  PH/ADU  RNOISE    FRAD
  1   284   492   400.0 24000.0    20.0    3.00   20.00    6.50     2.0

=====================================================================
```

The purpose of this header is to provide a partial record of the analysis which produced the file, and to supply some auxiliary information to subsequent routines (so you don't have to). As the data reduction proceeds through **FIND**, **PHOTOMETRY**, and profile fitting, each routine adds more information to the header. NL is a historical artifact — it started out meaning "Number of lines", where NL=1 indicated that the file contained one line of data per star (output from **FIND** or **PEAK**, for example), and NL=2 flagged output from **PHOTOMETRY**, where two lines of data are generated per star. NL has since ceased to have precisely this significance; now it serves more as a flag to the program to tell it where in the data record the sky brightness for each star may be found. For files produced by **FIND**, **PEAK**, **NSTAR**, and ALLSTAR, NL=1; for files produced by **PHOTOMETRY**, NL=2; for files produced by **GROUP** (and **SELECT**), NL=3. **SORT**, **OFFSET**, and **APPEND** produce output files retaining the form of whatever file was used for input.

Items NX and NY in the file header are the size of the picture in pixels. LOWBAD and HIGHBAD are the good data limits, the former calculated in **FIND** and the latter defined as an optional parameter. THRESH is the threshold that was calculated in **FIND**. AP1 is the radius in pixels of the first aperture specified for **PHOTOMETRY**, PH/ADU and RNOISE are the numerical values of photons/ADU and readout noise which were specified as options when **FIND** was run. FRAD is the value of the fitting radius (user-alterable optimizing parameter) which was in effect when the file was created.

One other thing which the output data files have in common is the format of the first four numbers in the data for each star. Always they are:

Star ID, X centroid, Y centroid, magnitude

followed by other stuff, with format

(1X, I5, 3F9.3, nF9.?) .

In the output from PHOTOMETRY, alone, this is followed by another data line per star.

In the pages that follow, a sample of each of the output formats is shown. An example of the disk file that contains the numerical point-spread function is also provided.

Sample output from **FIND** (a .COO file)

```
============================================================
 NL  NX  NY  LOWBAD HIGHBAD  THRESH
  2 284 492   400.0 24000.0    20.0

  1    6.953    2.612   -0.053    0.333   -0.546   -0.532
  2  200.061    2.807   -3.833    0.597    0.000    0.054
  3  156.254    5.171   -3.306    0.653   -0.144   -0.046
  4  168.911    5.318   -1.137    0.613    0.288    0.182
  5  110.885    9.742   -7.080    0.590    0.040    0.045
  6  147.949   10.208   -0.440    0.489    0.034    0.134
  7   64.002   13.161   -2.427    0.643   -0.124   -0.120
  8  270.856   12.738   -2.304    0.602    0.074    0.034
  9   14.925   13.842   -2.976    0.627   -0.045   -0.065
 10   38.813   15.268   -1.491    0.643    0.065    0.045
 11   93.695   15.164   -3.687    0.625   -0.203   -0.103
 12  139.798   15.715   -1.156    0.599    0.034    0.054
 13  207.929   16.209   -3.608    0.649   -0.062   -0.162
                                .
                                .
                                .
============================================================
 (1)    (2)      (3)      (4)      (5)      (6)      (7)
```

(1) Star ID number.

(2) X coordinate of stellar centroid.

(3) Y coordinate of stellar centroid.

(4) Star's magnitude, measured in magnitudes relative to star-finding threshold (hence never positive, since stars fainter than the threshold are rejected, obviously).

(5) Sharpness index (see **FIND** above).

(6) Roundness index (see **FIND** above).

(7) Roundness index based on the marginal distributions.

Sample output from **PHOTOMETRY** (a .AP file)

```
=================================================================
 NL   NX   NY  LOWBAD HIGHBAD  THRESH     AP1  PH/ADU  RNOISE
  2  284  492   400.0 24000.0    20.0    3.00   20.00    6.50

     1    6.953     2.612   99.999   99.999   99.999   99.999 . . .
      464.618  9.71  0.52   9.999    9.999    9.999    9.999 . . .

     2  200.061     2.807   99.999   99.999   99.999   99.999
      465.180  7.79  0.16   9.999    9.999    9.999    9.999

     3  156.254     5.171   14.610   14.537   14.483   14.438
      462.206  7.26  0.37   0.013    0.014    0.015    0.016

     4  168.911     5.318   16.261   16.056   15.855   15.658
      463.292  7.16  0.36   0.055    0.053    0.050    0.048

     5  110.885     9.742   10.792   10.728   10.688   10.660
      463.926  6.81  0.24   0.001    0.001    0.001    0.001

     6  147.949    10.208   17.167   17.084   17.019   16.878
      462.241  7.16  0.37   0.124    0.135    0.145    0.144

     7   64.002    13.161   15.620   15.569   15.549   15.538
      462.009  5.93  0.25   0.025    0.028    0.032    0.035

     8  270.856    12.738   15.566   15.527   15.493   15.460
      460.965  6.28  0.14   0.026    0.029    0.032    0.035

     9   14.925    13.842   14.951   14.863   14.800   14.744
      463.874  6.65  0.25   0.016    0.017    0.018    0.019

    10   38.813    15.268   16.200   16.113   16.052   16.019
      462.127  8.63  0.50   0.062    0.066    0.072    0.079
                                      .
                                      .
                                      .
==================================================================
     (1)     (2)       (3)       (4)       (5)       (6)       (7)  ...
        (16)    (17)   (18)     (19)      (20)      (21)      (22) ...
```

(1) Star ID number.

(2) X coordinate of stellar centroid, same as in .COO file.

(3) Y coordinate of stellar centroid, same as in .COO file.

(4) Star's magnitude in aperture 1, measured in magnitudes relative to a zero-point of 1 star ADU = 25.0 mag.

(5) Star's magnitude in aperture 2, ditto ditto.

(6) Star's magnitude in aperture 3, ditto ditto.

(7) Star's magnitude in aperture 4, ditto ditto.

(8)-(15) Ditto ditto.

(16) Estimated modal sky value for the star.

(17) Standard deviation of the sky values about the mean.

(18) Skewness of the sky values about the mean.

(19) Estimated standard error of the star's magnitude in aperture 1.

(20) Ditto ditto aperture 2.

(21) Ditto ditto aperture 3.

(22) Ditto ditto aperture 4.

(23)-(30) Ditto ditto.

Magnitudes for a number of stars are 99.999 +– 9.999, because the aperture extends beyond the boundary of the frame.

Sample output from **PEAK**, **NSTAR**, or ALLSTAR
(a .PK, .NST, or .ALS file)

```
================================================================================
 NL    NX    NY  LOWBAD HIGHBAD   THRESH     AP1  PH/ADU  RNOISE    FRAD
  1   284   492   400.0 24000.0     20.0    3.00   20.00    6.50     2.0

    1     7.413     2.652    18.771     0.421  464.618      10.      0.71   -0.399
    2   200.131     2.807    14.094     0.012  465.180       5.      0.92   -0.013
    3   156.454     5.421    14.629     0.018  462.206       4.      1.09   -0.012
    4   168.921     5.738    16.528     0.053  463.292       6.      0.63    0.056
    5   110.865     9.732    10.793     0.007  463.926       3.      0.76   -0.015
    6   147.759    10.478    17.330     0.128  462.241       9.      0.86    0.044
    7    64.032    13.401    15.595     0.022  462.009       4.      0.54   -0.021
    8   270.776    12.748    15.522     0.029  460.965       3.      0.93   -0.006
    9    14.925    13.882    14.982     0.015  463.874       3.      0.58   -0.016
   10    38.833    15.508    16.316     0.055  462.127       5.      0.94    0.075
   11    93.625    15.354    14.188     0.015  462.560       4.      1.21    0.026
   12   139.818    15.875    17.009     0.081  465.570       3.      0.63   -0.069
   13   207.909    16.459    14.385     0.012  463.223       4.      0.74   -0.009
                                           .
                                           .
                                           .
================================================================================
  (1)     (2)       (3)       (4)       (5)      (6)         (7)      (8)      (9)
```

(1) Star ID number.

(2) X coordinate of stellar centroid; a more accurate value than before.

(3) Y coordinate of stellar centroid; a more accurate value than before.

(4) Star's magnitude, measured in magnitudes relative to the magnitude of the PSF star (see discussion of .PSF file below).

(5) Estimated standard error of the star's magnitude.

(6) Estimated modal sky value for the star (from **PHOTOMETRY**, see above).

(7) Number of iterations required for the non-linear least-squares to converge.

(8) CHI - a robust estimate of the ratio: the observed pixel-to-pixel scatter from the model image profile DIVIDED BY the expected pixel-to-pixel scatter from the image profile.

(9) SHARP: another image quality diagnostic (see **PEAK** command above). SHARP is most easily interpreted by plotting it as a function of apparent magnitude. Objects with SHARP significantly greater than zero are probably galaxies or unrecognized doubles; objects with SHARP significantly less than zero are probably bad pixels or cosmic rays that made it past **FIND**.

Sample of a point-spread function file (a .PSF file)

```
  (1)       (2)   (3)  (4)  (5)   (6)            (7)          (8)       (9)
(10)         (11)          (12)          (13)
(14)...
============================================================================
 PENNY1     51    4    3    0   12.033      33419.105    158.5    255.0
   1.02691E+00  1.06132E+00  7.23460E-01  1.44567E-01
  1.348391E-02 1.348391E-02 1.348391E-02 1.348391E-02 1.348391E-02 1.348391E-02
  1.348391E-02 1.348391E-02 1.348391E-02 1.348391E-02 1.348391E-02 1.348391E-02
  1.348391E-02 1.348391E-02 1.348391E-02 1.348391E-02 1.348391E-02 1.348391E-02
 -6.484396E+01-6.400895E+01-6.459602E+01-6.713283E+01-6.921574E+01-6.974660E+01
 -6.971574E+01-6.964603E+01-7.014906E+01-7.076376E+01-7.028298E+01-6.946377E+01
 -6.923672E+01-6.868890E+01-6.699422E+01 1.348391E-02 1.348391E-02 1.348391E-02
  1.348391E-02 1.348391E-02 1.348391E-02 1.348391E-02 1.348391E-02 1.348391E-02
  1.348391E-02 1.348391E-02 1.348391E-02 1.348391E-02 1.348391E-02 1.348391E-02
  1.348391E-02 1.348391E-02 1.348391E-02 1.348391E-02 1.348391E-02 1.348391E-02
  1.348391E-02 1.348391E-02 1.348391E-02 1.348391E-02 1.348391E-02 1.348391E-02
  1.348391E-02 1.348391E-02 1.348391E-02 1.348391E-02 1.348391E-02 1.348391E-02
 -6.271802E+01-6.469427E+01-6.637348E+01-6.902106E+01-6.919862E+01-6.852924E+01
 -6.951303E+01-7.280193E+01-7.551125E+01-7.588057E+01-7.524973E+01-7.459446E+01
 -7.402506E+01-7.357459E+01-7.343834E+01-7.333043E+01-7.215495E+01-7.010623E+01
 -6.792765E+01-6.512524E+01-6.251689E+01 1.348391E-02 1.348391E-02 1.348391E-02
  1.348391E-02 1.348391E-02 1.348391E-02 1.348391E-02 1.348391E-02 1.348391E-02
  1.348391E-02 1.348391E-02 1.348391E-02 1.348391E-02 1.348391E-02 1.348391E-02
  1.348391E-02 1.348391E-02 1.348391E-02 1.348391E-02 1.348391E-02 1.348391E-02
                                          .
                                          .
                                          .
==============================================================================
```

(1) An alphanumeric string uniquely defining the module used to define the analytic first approximation to the PSF.

(2) Size of the array containing the PSF look-up table. Values are tabulated at half-pixel intervals in a square array which extends somewhat beyond the user-specified PSF radius: $N = 2(2 \cdot \text{radius} + 1) + 1$. Here the PSF radius was 12.

(3) The number of shape parameters in the analytic function.

(4) The number of look-up tables in the PSF. Here a linearly varying PSF was used.

(5) Fractional-pixel expansion. ***** NOT IMPLEMENTED *****

(6) The instrumental magnitude corresponding to the point-spread function of unit normalization.

(7) Central height, in ADU, of the analytic function which is used as the first-order approximation to the point-spread function in the stellar core.

(8),(9) X and Y coordinates of the center of the frame (used for expanding the variable PSF).

(10)-(13) The shape parameters of the analytic function. In this case there are four of them. The first two are always the half-width at half-maximum in x and y.

(14). . . The look-up table of corrections from the analytic first approximation to the "true" point-spread function.