Cooking with **ALLFRAME**: Photometry and the H_0 Key Project

Anne M. Turner
aturner@as.arizona.edu

Version 3.0, last update September 19,1997

Abstract

This cookbook assists members of the HST H_0 Key Project team in the use of **ALLFRAME** to extract photometry for the project. Included are instructions for the preparation of data for use in **ALLFRAME**, preparing and running **ALLFRAME**, detecting variables, and calibrating the photometry.

Contents

1 Introduction

This document presents a recipe for reducing images with Peter Stetson's **DAOPHOT** family of software, which includes **ALLSTAR** and **ALLFRAME**. This procedure is tailored specifically to WFPC2 images of crowded fields in nearby galaxies taken for the HST H_0 Key Project. This document grew out of notes I took during my stays at the DAO during August 1993 and July/August 1994. It has two primary sources of information, stuff Peter Stetson told me and stuff I or fellow **ALLFRAME** users screwed up on as we were learning and so know to be careful about. I have tried to write this document from a beginner's standpoint, so that confusing issues are explained and many potential areas of trouble are pointed out. I have also included checklists which I hope will prove helpful as the user begins actual data reduction.

In order to use this document successfully, it is strongly recommended that you have on hand the DAOPHOT II manual and the "guinea pig" documentation (somewhat out of date but still quite useful) described in the *Useful References* section of this paper. These are both available in the **DAOPHOT** family package you should get from P. Stetson (hereafter PBS). You should also have access to the Key Project Archive (maintained by Nancy Silbermann, nancys@ipac.caltech.edu) where important documents and other necessities for reduction can be found.

Suggestions for improvements and additions to this manual are welcomed and should be addressed to the author (aturner@as.arizona.edu) and the guru (Peter.Stetson@hia.nrc.ca). If you encounter problems with the software or documentation described herein, contact PBS. In his words, "if anyone encounters a problem, a source of confusion, or a suspected bug, I want to be the first person on the planet to hear about it. You should not waste your time chasing down some problem that I may be able to fix in a couple of minutes. And I don't want people bad-mouthing my software behind my back. Do it to my face so I can enjoy it, too. . . . people should contact me if they think they see a discrepancy between the documentation and reality."

Finally, before you get started, check the Archive to make sure you have the most recent copy of this document available. This cookbook and its Frequently Asked Questions (FAQ) list, found in Appendix B, are updated frequently to reflect the needs and growing collective wisdom of **ALLFRAME** users. As you work through the reductions process and you have questions or encounter problems, send them to us so that not only you get help, but everyone can benefit from your blunders, the way you are benefiting from ours.

2 Data Processing

The pipeline processed data will come from STScI labeled with the file extensions '.c0h' for the header file and '.c0d' for the data (pixel) file. Also needed are the data quality files (DQF), which have extensions '.c1h' and '.c1d'. Note that unless you have requested that a special tape be written your files will NOT read off the tape this way, and you will have to check your copy of the list of files on your tape to figure out which files are which. The data files are of type real, while the DQF's can be of type integer. In using IRAF `rfits`, if the type specification is left blank, `rfits` will automatically choose the proper precision. Some **ALLFRAME** users have reported success using `r49fits` or `strfits` in the `stsdas.fitsio` package usually installed with IRAF. These tasks should preserve the 3D group nature of the HST images, and may do a better job of preserving file extensions.

If available, public-domain images may be obtained from the Canadian Astronomy Data Centre at

`http://cadcwww.dao.nrc.ca/hst/science.html`

The advantage to this approach is that the data are reprocessed on the fly, using the most appropriate biases and flats for each exposure. STScI distributes pipeline-reduced data, which use the best flats and biases that were available AT THE TIME OF THE EXPOSURE, i.e. it does not include any biases or flats that were taken after the exposure, no matter how immediately after they were taken. CADC also often has faster turnaround than STScI: the data are permanently on-line in two 500-disk juke boxes, and the recovery, recalibration, and distribution are fully automatic, so it works at night and on weekends and holidays. Distribution is normally by FTP, rather than tape, so you can usually have your data within four hours of so of submitting the request. CADC distributes FITS by default, but other options are possible.

You will also need a vignetting mask made by PBS and a pixel area map which are available from him directly (see below) or from the Archive to process your data. A mask made from the DQFs and vignetting mask will be used to mark bad pixels in the images. The pixel area map is used to restore the integrity of the flux measurements (see below). Standard team procedure for processing these files was formulated by PBS and Abi Saha. Items quoted below come directly from PBS's summaries.

- "All images will be retained in their 800x800 format, rather than being trimmed. Vignetted regions and other bad pixels will be masked. Contrary to what is said on p. 88 on the "HST Data Handbook," vignetted pixels have *NOT* been identified as such in the data quality files. Instead, I have generated my own vignetting mask: I acquired copies of the two pipeline flats (F555W and F814W). Pixels that differ from the median for the chip by LESS than a factor of root(2) [chosen arbitrarily] have been reset to 0; pixels that differ from the median for that chip by MORE than a factor of root(2) have been reset to 64. Then the V and I masks were summed, so that all pixels which are bad in EITHER the V or I flats will be masked off in ALL the data." To obtain, ftp anonymously to ftp.dao.nrc.ca, cd pub/staff/pbs, prompt, binary, mget vignetting.*,mget pixarea.*

- "Images will be multiplied by 4 and stored as short integers. This will halve the disk usage (and halve it again when images are compressed while not being needed for a while), while still adequately sampling the readout noise: effective gain will be 1.75 e^{-}/ADU and effective read noise will be 4 ADU (or 3.5 e^{-}/ADU and 2 ADU for gain 14)."

- ". . . our science exposures need to be multiplied by a pixel-area map. The problem is that, due to geometrical distortion in the WFPC2 optics, individual pixels on the CCD do not map to exactly equal areas on the sky. When the flat-field exposures are taken, the pixels that are smaller *in projection* receive fewer photons, and as a result appear less sensitive. When the science exposures are corrected by these flats, the "smaller" pixels near the edges and corners of the chips have their recorded fluxes artificially boosted. Thus, while frames that have been rectified by the pipeline flats correctly represent the surface brightnesses of astronomical objects, integrated fluxes have been distorted. Multiplying the normalized science images by a map of the projected pixel areas restores the integrity of the flux measurements. The pixel area maps kindly provided by Holtzman have been normalized to the area of pixel (400,400) on each chip. The nature of the geometric distortion is such that in each case pixel (400,400) turns out to be among the largest on the chip. We therefore renormalized the maps to the *median* pixel area on each chip, which was approximately 0.985 times the area of pixel (400,400). This slight adjustment means that the effective readout noise and gain in the science exposures will be unchanged, when averaged over the area of each chip, but it does also mean that a small ($\approx$0.015 mag) difference will appear between the calibration zero-points appropriate to Holtzman's reductions and those appropriate to ours."

To process your images relatively painlessly, you need to create three files containing lists of image names. The "data" list should contain the list of the names of your galaxy images (the .c0h files), the "mask" list should contain the names of your DQFs (the .c1h files), and the "names" list should contain the list of names you want your images to be called after processing. Note that for use in **DAOPHOT** the image names in the "names" list should have the IRAF image extension ".imh". These three files can be created with your favorite UNIX commands and/or editor. If you do have images which have .c0h extensions (i.e. if you ftp'ed them or had a special tape written), you can use PBS's procedure below to create these files.

- Create a file containing the names of the data images (e.g. *.c0h) one filename per line, e.g.

```
% ls *.c0h >!  data
```

- Create a copy with c0h changed to imh

```
% cat data | sed s/c0h/imh/ >!  name
```

- Create a file containing the names of the data-quality images, e.g.

```
% cat data | sed s/c0h/c1h/ >!  mask
```

Once you have the three files, use PBS's procedure for masking bad pixels. In IRAF, epar imarith, set pixtype = real, then

```
imarith @mask + vignetting @mask          ! add vignetting mask
imarith @mask / @mask @mask divzero=1.25   ! good pixels --> 1.25, bad --> 1
imarith @mask - 1. @mask                  ! good pixels --> 0.25, bad --> 0
imarith @data * pixarea @name
imarith @name / @mask @name div=32767.1 pix=short
imdel @data,@mask
```

Or if you are short on disk space, use

```
imarith @data * pixarea @data
imarith @data / @mask @name div=32767.1 pix=short
imdel @data,@mask
```

in place of the last 3 lines above. This saves creating a set of real*4 images by overwriting the image you read off tape.

The imarith manipulations in the previous step can all be performed on the 800x800x4 images, thus saving some typing. In that case, apply `imslice` as the last step. You can also use the above steps to process 800x800 data if the vignetting and pixarea files have been appropriately sliced and you have created separate mask, data, and name lists for each chip.

3 Running ALLFRAME

3.1 Overview

The **ALLFRAME** process will require a lot of disk space and cpu time. I've divided the process somewhat arbitrarily into three steps: making the medianed image, making the star list, and preparing and running **ALLFRAME**. Each step is a prerequisite for the next. Noticeably missing from this list is the step of making the psf. Good psfs are next to impossible to make for the individual galaxy images in the Key Project data. PBS has had success making psfs from WFPC2 globular cluster frames. He has been supplying these psfs to the H_0 Key Project team. I will be assuming you will be using these psfs. Make sure you have a copy of the latest (Jan 95) psfs. Earlier versions are not compatible with the data processing routines described above. If you are making your own psfs (presumably for another project), you will require lots of time and patience. The process for making a psf is outlined in the DAOPHOT II manual and I will not elaborate on it here.

If you are using an older version of **DAOPHOT** or **ALLFRAME**, it is worthwhile to check with PBS to see if a new improved version of any of the routines is available, before you spend considerable time reducing data with an out-of-date version.

Words that are in **BOLDFACE CAPS** in this section are the names of programs to be run, e.g. **DAOPHOT** or **ALLFRAME**. Words that are in STANDARD CAPS in this section (with the exception of WFPC2 and IRAF) are either a parameter required for a program, e.g. PSF RADIUS, or a command to be executed within **DAOPHOT**, e.g. ATTACH. For the sake of demonstrating various commands, I will assume you have a data set of `image1.imh`, `image2.imh`, `image3.imh`, etc, and are working on a single chip, which I refer to as `chip1`.

Finally, all the procedures that are described in this section can be performed on all of the images in a data set, regardless of the filter or exposure time. All images can be used in making the medianed image, and all images can use the same star list in the **ALLFRAME** reductions. There is no need to separate images by filter or exposure time when performing any of the procedures described in this section, with only one exception in the final step of making the medianed image. **ALLFRAME** was in fact designed to overcome some of the difficulties of reducing the data separately (see Stetson 1994 for more info).

3.2 Making the Medianed Image

The first step is to gather all the processed images you have for your field and chip together in the same directory or tell **DAOPHOT** how to find them in the following manner. For each directory containing data make a line in your .cshrc file like the ones in the following example:

```
setenv kwa /scr/kwakiutl/stetson
setenv m101 $kwa/m101
setenv m101w $m101/wfc
alias kwa cd $kwa
alias m101 cd $m101
alias m101w cd $m101w
setenv m101p $m101/pc
alias m101p cd $m101p
```

The alias commands are optional; they are convenient, but not required for the following to work. File names can then be addressed as `directory:filename`, e.g. in **DAOPHOT**,

```
Command:  attach m101w:image1
```

This naming convention will work when any program written by PBS requires a filename in a separate directory and permits you to fit paths of almost arbitrary length into the 30-character limit that most **DAOPHOT** routines impose on filenames.

Now run **DAOPHOT** (look in the DAOPHOT II manual for a more thorough description of each task). A list of options appropriate to Key Project data is shown below.

```
 READ NOISE (ADU; 1 frame) =   4.00          GAIN (e-/ADU; 1 frame) =     1.75
LOW GOOD DATUM (in sigmas) = 100.00        HIGH GOOD DATUM (in ADU) = 32000.00
            FWHM OF OBJECT =   1.50           THRESHOLD (in sigmas) =     5.00
 LS (LOW SHARPNESS CUTOFF) =   0.20      HS (HIGH SHARPNESS CUTOFF) =     1.00
 LR (LOW ROUNDNESS CUTOFF) =  -0.70      HR (HIGH ROUNDNESS CUTOFF) =     0.70
            WATCH PROGRESS =  -1.00                  FITTING RADIUS =     2.00
                PSF RADIUS =  20.00                    VARIABLE PSF =     0.00
FRACTIONAL-PIXEL EXPANSION =   0.00              ANALYTIC MODEL PSF =    -3.00
 EXTRA PSF CLEANING PASSES =   9.00          USE SATURATED PSF STARS =     1.00
      PERCENT ERROR (in %) =   0.75             PROFILE ERROR (in %) =     5.00
```

Notice the LOW GOOD DATA parameter is set to the very large value of 100 sigma. **This is important.** The lowest good datum value is calculated as the specified number of sigmas below the *typical* value of the sky. Since the images for the Key Project often have a significant surface brightness gradient across them due to the underlying galaxy, the sky value in any one location can be considerably fainter than the *typical* sky. If the LOW GOOD DATA value is not large enough, so that the LOWBAD value is greater than the lowest sky values, serious problems can result later on, including problems with the calibration of your photometry. Bad columns and pixels are masked by the DQFs so the LOW GOOD DATA parameter is not needed to exclude them. The PSF radius of 14 is appropriate for the WF chips (2-4). The PSF radius for the PC chip (1) is 20 pixels. Notice also that the FITTING RADIUS is set to 2 pixels instead of the default 2.5. Using a FITTING RADIUS of 2 or even 1.75 for WFPC2 data will result in better fits and smaller errors than using the larger values. The GAIN, READ NOISE, and HIGH GOOD DATUM values listed are appropriate for data that has been processed as described in section 2 of this paper. WATCH PROGRESS can be set to whatever value you like; see the DAOPHOT II manual for details. Make sure your options are appropriate. Use OPT to change them if necessary or exit and make/edit your `daophot.opt` file.

ATTACH your first picture and run FIND. If you have set the LOW GOOD DATA value to 100 as directed above, everything should be fine. To be sure you might want to check the FIND output file. The LOWBAD data value given in ADU, the fourth number in the second row, should be smaller than the lowest good values of the sky. The next step is to run PHOT. Use the following `photo.opt` file, where the first column of numbers is for the PC (chip 1); it should be deleted when the file is used with WFC (chips 2–4).

```
A1 = 3.261 1.500
A2 = 3.567 1.641
A3 = 3.952 1.818
A4 = 4.413 2.030
A5 = 4.952 2.278
A6 = 5.567 2.561
A7 = 6.259 2.879
A8 = 7.026 3.232
A9 = 7.872 3.621
AA = 8.793 4.045
AB = 9.793 4.505
AC = 10.87 5.000
IS = 20.  14.
OS = 40.  35.
```

ATTACH your second picture, then run FIND and PHOT on it, keeping in mind the precautions outlined above. Continue with this until you have aperture photometry for each image. EXIT from **DAOPHOT**.

The next step is to find rough coordinate transformations between the images. Try running **DAOMATCH**. Give it your `image1.ap` file when it asks for the first input file. Name your output file something like `chip1.mch`. Then give it your `image2.ap` file for your second input file. If the cosmic rays are not confusing **DAOMATCH** too badly, it will make a reasonable guess at a coordinate transformation between `image1` and `image2`. You can have it recalculate the transformations using more stars (it will go to up to 35) by answering 'y' to the 'Another level?' prompt. When you are happy with the transformations, or when it reaches 35 stars it will write the transformations to the `chip1.mch` file. Then give it `image3.ap` as the next input file, and it will calculate transformations for this image. Continue feeding your `imageN.ap` files into **DAOMATCH**, one at a time. The first 6 columns of numbers in a **DAOMATCH** output file, called here `chip1.mch`, are the coefficients A through F in alphabetical order, where

$$x_2 = A + Cx_1 + Ey_1$$

$$y_2 = B + Dx_1 + Fy_1.$$

Unfortunately, **DAOMATCH** is easily confused by cosmic rays, so it may not (and probably won't) be able to make a good guess at the transformations. On the plus side, if most of the transformations between images are simple translations and the rotations are small, it is relatively easy to figure out a rough transformation between images by measuring the position of a bright star or two in each frame, figuring out the shift to a pixel or two. Shifts should be computed in the sense that *shift* = (*object position on image*1) − (*object position on imageN*). If you have a `chip1.mch` file after running **DAOMATCH**, it should be edited so that A and B are the measured shifts, C and F (scale terms) are approximately 1, D and E (rotation terms) are approximately zero in the case of no or small rotation. If you choose to skip the **DAOMATCH** step altogether, it is possible to create your own `chip1.mch` file with a format as follows. For example, your file should look something like,

```
'image1.ap' 0 0 1 0 0 1
'image2.ap' 0 0 1 0 0 1
'image3.ap' 27 -19 1 0 0 1
etc.
```

The file is in FORTRAN free format; all items are separated by spaces but column alignment is unimportant. The file name containing the photometry is in single quotes. The next six columns should contain your best guesses at coefficients A through F in alphabetical order. **DAOMASTER** will refine these initial guesses.

Now run **DAOMASTER**. **DAOMASTER** will refine the transformations generated by **DAOMATCH** (or you) and it is not as easily confused by cosmic rays since it works on the entire star list. Give it `chip1.mch` as the input file containing the transformations. **DAOMASTER** asks for several parameters, see the 'guinea pig' documentation for a more complete description. For the minimum number, minimum fraction, and enough frames, I use something like 2-3, 0.5, 0.5N where N is the total number of frames. Presumably cosmic rays will not be at the same location in more than half the frames and so will be rejected. An alternative would be to ask yourself, "What is the minimum number of observations required to define a Cepheid light curve?" If you answer "10," then use 2-3, 0.5, 10. I then use a maximum sigma of about 0.5 to reject problem stars, very faint stars and stars that did not have an aperture magnitude determined for some reason. You should probably play with these numbers somewhat and see what effect it has on the transformations **DAOMASTER** derives. You are next asked for the number of geometric coefficients to solve for. A secret feature of **DAOMASTER** is that it can handle quadratic and cubic distortions in addition to translations, scale changes, and linear distortions. If you need to solve for these higher order distortions, when you are prompted for "Desired degrees of freedom," you can enter 12 for quadratic and 20 for cubic, in addition to the 2, 4, or 6 listed. There is higher order distortion in each of the WFPC2 chips, and if your images are shifted a few tens of pixels or more, you will need to solve for cubic transformations.

To begin matching, **DAOMASTER** asks for a critical radius. The critical radius is the maximum permissible distance between a star's position in `imageN` as transformed to the system of the master list, and its average position in the master list. The first critical radius you give **DAOMASTER** should be at least twice as large as the last critical radius you intend to use, since for the first iteration **DAOMASTER** defines a star's position in `image1` as its master list position. Beginning matching with a relatively large critical radius, something like 5 or 7, works well for the Key Project data sets and is easily more than twice the smallest critical radius you will use, which is almost always 1. Decrease the critical radius with each iteration of the transformation equations (or repeat the radius of 1) until the number of stars it finds and the transformation remain constant. RMS errors in coordinates, the first two columns output to the screen are typically on the order of 0.1 pixel. As **DAOMASTER** exits, the only output file you are interested in at this point is a file with the new transformations. After you have answered 'y' to this output option only, and overwrite the previous `chip1.mch` file, type your EOF control character (usually CTRL-D, but try 'stty -a' at a UNIX prompt and look under 'eof' to check) to exit the program.

For Key Project data, you will have many more V frames than I frames, so taking the median of all images will essentially give you a V image. This may cause problems if there are significantly more red than blue stars, as might happen if your galaxy is close enough to see the tip of the red giant branch. To avoid this problem, you might want to make separate median images for the V and I frames and then sum the two for use in making the star list. To do this copy `chip1.mch` to `v1.mch` and `i1.mch`. Use your favorite text editor to remove all I band frames from `v1.mch` and all V band from `i1.mch`.

Run **MONTAGE2**. Give it the `v1.mch` file as input. Hit return at the image suffix prompt; giving it an 's' or a 'j' here will allow it to montage subtracted images produced by **ALLSTAR** or **ALLFRAME**. **MONTAGE2** will then ask you for the minimum number of frames a pixel should appear in before it is included and the percentile from the ensemble that should make up the new image. In a data set of twenty-something frames, something like 3 is a reasonable value for the minimum number of frames a pixel should appear in, but feel free to play with this value. Answer '0.5' for the percentile to get a median image. You might want to make note of the values you give it for later use. At the X and Y limits prompts answer CTRL-D (or your EOF control character) for the default values, or 1, 800 for an image the same size and on the same coordinate system as the first image. An expansion factor of 1 makes an image with the same scale as the input images. Calculated limits for each image will then appear on your screen along with a line that looks like:

```
Offsets:  (Star list) - (montaged image) = 0, 1
```

Be sure to write these offsets down–you will need them later. Tell the program the name you want for your image, such as `v1`, and in a few minutes it will be done. Repeat this procedure giving **MONTAGE2** the `i1.mch` file.

MONTAGE2 will subtract off an approximate sky value for each input image. These values are reported on the screen as **MONTAGE2** runs. You will probably want to add a typical sky value back into the finished image **MONTAGE2** produces, so **DAOPHOT** will do the statistics correctly when you begin to FIND stars on the image. You can use the IRAF task `imarith` to do this addition. For example if the typical sky value **MONTAGE2** subtracts is 100, then type,

```
imarith v1 + 100 v1 pix=short
```

Finally, add `v1.imh` and `i1.imh` with

```
imarith v1 + i1 chip1 pix=short
```

to produce the `chip1.imh` image used for making the star list.

3.3 Making the Star List

Run **DAOPHOT** and ATTACH on `chip1.imh`. If you need to change any of your OPT values, do it now. Double check that your LOW GOOD DATUM value (in sigma) is set to 100, since the noise in the sky in the medianed image will be small and background gradients in the galaxy relatively large. To be safe, you should again check the LOWBAD value in the header of your `chip1.coo` file after running FIND for the first time. Run FIND on your image. Since you are working on a medianed image you will want to tell FIND that you have averaged 5 images. From PBS:

> I'm starting to dislike the notion of saying that you have averaged together 0.5-0.7 times the number of images that you medianed. The reason is HST-specific and does not apply to other data: generally all of the images have been debiased with the same bias frames and flattened with the same flats. If a significant fraction of the images have the SAME shift to within less than a pixel, any defects in the biases or flats will NOT average out, and the positive noise spikes in the biases and the negative noise spikes in the flats will come out as highly significant "detections" in the .coo file. I would tend to say that you should tell FIND that of order 5 images have been averaged, regardless of how many have been medianed. Even this low an "averaged" number of frames should detect any star that will later be worth measuring in individual images.

To get a FIND threshold appropriate for your medianed image, you may need to experiment so you don't get a large number of noise spikes masquerading as stars while picking up most of the real stars. One useful method is to plot up the number of stars found versus the threshold value. Then pick a threshold value near the elbow of the curve where the number of "stars" starts increasing rapidly as the threshold is lowered. You can easily obtain the numbers for this plot by answering 'n' to the 'Are you happy with this?' prompt, giving it a new threshold, and recording the threshold along with the number of stars found each time. Note that at times you may need to use a very large threshold, on the order of 20 or so, to find the elbow (see Appendix B for a possible explanation); other times a threshold of 5 may do the job. Occasionally, the curve may flatten out at low thresholds as it begins to run out of noise spikes to hit, creating a second 'elbow.' If this happens don't confuse this flattening with the elbow you are looking for, where the number of stars begins to *increase* more rapidly with decreasing threshold.

After you get a list of coordinates you are relatively happy with, run PHOT on it. EXIT from **DAOPHOT**, and give this image and aperture photometry file to **ALLSTAR**. You can use a psf for an individual frame for this **ALLSTAR** reduction; it will work well enough for the purposes of making the star list and certainly works better than any psf I was ever able to make for a medianed image. The parameters used for **ALLSTAR** are given below, and can be included in an `allstar.opt` file, made in a similar manner as the `daophot.opt` file. Note the FITTING RADIUS is again 2 instead of the default of 2.5. It is also a good idea to check to make sure the OUTER SKY RADIUS is not larger than the typical length scales of significant sky variation in your images. Recommended values are 0.7 for the IS radius, 10 for the WF chips OS radius, and 16 for the PC OS radius.

```
      FITTING RADIUS =  2.00     CE (CLIPPING EXPONENT) =   6.00
REDETERMINE CENTROIDS =  1.00        CR (CLIPPING RANGE) =   2.50
      WATCH PROGRESS =  0.00       MAXIMUM GROUP SIZE =  50.00
 PERCENT ERROR (in %) =  0.75      PROFILE ERROR (in %) =   5.00
IS (INNER SKY RADIUS) =  0.70     OS (OUTER SKY RADIUS) =  10.00
```

When **ALLSTAR** is finished look at the subtracted image. You will probably want to run **DAOPHOT**, ATTACH the subtracted image and run FIND and PHOT on it to pick up the stars that were missed the first time. Remember to use the THRESHOLD value you chose for the original image. Run OFFSET on this new aperture photometry file and add some large number less than 1 000 000 (e.g. 50000) to the ID numbers to insure that each star has a unique ID. This is important because many of the **DAOPHOT** routines identify stars by their ID number and will get confused if two stars have the same number, possibly causing bad results. Run GROUP on the **ALLSTAR** output and on the new PHOT/OFFSET output (the GROUP step is a bookkeeping step necessary to put the files into the same format so just pick a critical radius that minimizes your run time), and APPEND these two files. Then exit from **DAOPHOT** and run **ALLSTAR** on the original montaged image with this combined star list.

If a large number of stars are still left on the subtracted image after the second **ALLSTAR** run, you may want to run FIND on it yet again. This time try tweaking the parameters used by FIND if necessary, particularly the HIGH SHARPNESS CUTOFF or the FWHM, since the stars are undersampled and appear very pointy, especially on the WF chips. If you do tweak these or other parameters, it is a good idea to check that you are actually finding stars and not just noise spikes. The IRAF task `tvmark` can be used with an edited form of the coordinate file for this purpose, or you can ask PBS to send you a copy of his IRAF script which will do the same thing without any need to edit the coordinate file. When you are happy with that, repeat the PHOT, OFFSET, GROUP, APPEND, **ALLSTAR** process above. Continue repeating if necessary until most of the stars are found.

Finally, you will probably have to make a file containing the last few (one hundred or so?) stars by hand, by editing the output of your favorite IRAF task (an `imexam` logfile for example), or with PBS's IRAF script, `load.cl`. When making a coordinate file like this it should again be in FORTRAN free format, with space between each number. Star ID numbers must be less than or equal to 999999. The file should also have a three line **DAOPHOT** FIND header. Check a coordinate file output by FIND for reference. Run PHOT on the file you created, OFFSET coordinates if necessary, GROUP and APPEND this file and the **ALLSTAR** output. Then run **ALLSTAR** one last time.

When the last **ALLSTAR** run is finished, run **DAOPHOT** again. Remember the offsets you wrote down when **MONTAGE2** was running? Run OFFSET on the final **ALLSTAR** output and give it those offset values for X and Y. Rename the output file `chip1.mag` instead of the default `chip1.off`. `chip1.mag` is the star list you will give to **ALLFRAME**.

3.4 ALLFRAME at Last

ALLFRAME is very simple to run. In addition to the individual images it needs four types of files: the file containing the coordinate transformations (`chip1.mch`), the star list (`chip1.mag`), a psf named for each image (`image1.psf`, `image2.psf`, etc.), and an **ALLSTAR** output file (`image1.als`, `image2.als`, etc.). If your image files are in separate directories and you are using the `directory:filename.extension` naming convention as discussed in Section 3.2, it is legitimate to have that `directory:filename.extension` inside your .mch file. You need only be certain that ALL files with the same `filename` are located in the same `directory`. Output files generated with the same `filename` will likewise be deposited in the same `directory`, so make sure there is room! The `chip1.mag` star list was completed in the last step and is ready to use. The `chip1.mch` file was created during the process of making the median image. It needs one change. You must edit this file so that the list of files it contains are `image1.als` file names and not `image1.ap` file names. This can be done with your favorite text editor.

You will need to make symbolic links from the WFPC2 psf files to psf files named appropriately for each image. For example, if `image1`, `image5`, and `image6` all need to use the `v1.psf`, `v1.psf` should be linked to `image1.psf`, `image5.psf` and `image6.psf`. You can use the UNIX command `ln`, for example,

```
ln -s v1.psf image1.psf
```

to create symbolic links to the psfs.

Finally **ALLFRAME** needs `image.als` files. Fortunately **ALLFRAME** needs only the three line header to get the read noise, gain, high and low good data values for the frame, and fitting radius appropriate for the image. Run **ALLSTAR** for each image using the parameters given above, give it an appropriate .psf and .ap file (these were made in previous steps), type CTRL-D (or your EOF control character) at the subtracted image prompt to suppress its creation, after **ALLSTAR** begins to think, telling you the number of stars in the list, the iteration number, etc., type CTRL-C to stop execution. The `imageN.als` file will be created.

Now run **ALLFRAME**. You may want to '`nice`' it if you are running it on a computer you are sharing with other people (type `man nice` at a UNIX prompt if you are unfamiliar with this command). Typical parameters are shown below and can again be edited in an `allframe.opt` file. You may need to change the GEOMETRIC COEFFICIENTS parameter to 12 or 20 if you need quadratic or cubic distortions taken into account in the coordinate transformations. The MAXIMUM ITERATIONS parameter can be decreased to something like 50 on a slower computer if necessary. Again check the OUTER SKY RADIUS parameter to make sure it isn't larger than typical length scales of sky variation in your image. Recommended values are 10 for the WF chips and 16 for the PC.

```
CE (CLIPPING EXPONENT) =   2.00         CR (CLIPPING RANGE) =    2.50
GEOMETRIC COEFFICIENTS =   6.00              WATCH PROGRESS =    1.00
    MINIMUM ITERATIONS =   5.00          MAXIMUM ITERATIONS =  200.00
  PERCENT ERROR (in %) =   0.00        PROFILE ERROR (in %) =    0.00
 IS (INNER SKY RADIUS) =   0.70       OS (OUTER SKY RADIUS) =   10.00
```

Give **ALLFRAME** the `chip1.mch` file as the file containing the list of images. It will then ask for a star list. Give it the `chip1.mag` file. It will take **ALLFRAME** a long time to run, sometimes many days on a slower computer. As it works **ALLFRAME** creates two new REAL copies of each image, twice as large as the original short integer images, so make sure you have the necessary free space.

To prevent tragedy in the event of a crash, **ALLFRAME** also creates a backup file called e.g. `chip1.bck`. If the computer crashes while **ALLFRAME** is working, this backup file will cause **ALLFRAME** when restarted to pick up almost where it left off, *unless* the crash takes place while files are being updated; then all is lost. During long **ALLFRAME** runs of several days or more PBS recommends backing up the entire directory to tape every couple of days and if necessary, restoring it after a crash during an update. To restart **ALLFRAME**, just start it as you normally would with the `chip1.mch` file. If **ALLFRAME** finds the backup file, it will automatically restart from the beginning of the last iteration. If the backup file is incomplete, it will want to go back to the very beginning and will request permission to overwrite what it had done up until the crash. **ALLFRAME** outputs a `chip1.tfr` file which is similar to the one in the **DAOMASTER** description in the "guinea pig" documentation and allows you to match up the stars in each image, a `chip1.nmg` file containing the new star list, and `imageN.alf` files for each image containing the actual photometry.

After **ALLFRAME** is finished it will leave behind a subtracted image with a 'j' suffix, e.g. `image1j.imh`, `image2j.imh`, etc. and associated pixel files, for each individual image. Run **MONTAGE2** with the same numbers you used in making the median image, but this time answer 'j' to the Image Suffix prompt. This will produce a median image for the star subtracted images. Compare this to the unsubtracted median. You also might want to run **MONTAGE2** on the images from each filter individually, i.e. make a medianed V image and a medianed I image. This will allow you be make sure the I images are as clean as the V images. If there are stars left behind on any of these subtracted images you may want to make a new star list from the montage of the *subtracted* images (see the Making the Star List step, but use this subtracted image instead of the original) and APPENDing the new list to the .nmg file, remembering to OFFSET your star ID numbers. This will create a complete star list you can feed to **ALLFRAME** and run again. This step will probably be necessary even if you have made a good star list the first time around.

3.5 Post-ALLFRAME

DAOMASTER can be used to match up stars and photometry from the `imageN.alf` files. The `chip1.mch` file needs to be edited so that **DAOMASTER** looks for the `.alf` files instead of the `.als` or `.ap` photometry files. When running **DAOMASTER** on the **ALLFRAME** output, I try to keep everything that shows up in something like half the total number of V frames, (sig $<$ 99) on the presumption that if it was bad, it would have disappeared in the reduction. When matching, the RMS errors in the coordinates based on the transformation should theoretically be zero, since the coordinates in the photometry files are set by the transformation that **ALLFRAME** determined.

The `chip1.mch` file can be further edited, depending on what output you want. For example, if you want to examine the V photometry only, you can copy the `chip1.mch` file to `v1.mch` and then edit `v1.mch` to delete all the non-V frames from the list. Then use **DAOMASTER** to match up photometry and output the desired information, maybe a file with corrected magnitudes to search for variables or a file with mean magnitudes and scatter to be used for stellar population analysis.

4 Calibration

The recommended calibration procedure is described in this section and makes extensive use of the programs in the `ccd` directory in the **DAOPHOT** distribution. These are described in much more detail in the 'Guinea pig' documentation also available in that distribution as file `ccd/ccdpck.man`. Refer to this document for additional information if needed, but bear in mind much of the software it describes has undergone revision more recently than that document, so it should be used only as a supplement to this cookbook.

Many of the programs in the `ccd` directory underwent substantial revision recently. If you have copies that are older than March 1997, you need to contact PBS to get new ones.

I have divided the calibration process again somewhat arbitrarily into several steps, selecting a list of program stars, finding growth curves with **DAOGROW**, finding aperture corrections, calibrating local standards, calibrating all stars, and quality assurance.

4.1 Selecting Program Stars

To calibrate your data, you first need to create a list of 20-30 program stars to serve as local standards. These stars should be relatively bright and isolated, and located on "clean" parts of the images. To do this, sort your `chip1.nmg` files by sigma, by starting up **DAOPHOT** and running SORT with option 5 and `chip1.nmg` as the file to be sorted. Edit the resulting `chip1.srt` file, retaining 30 to 50 stars with the smallest sigma and deleting any stars with large chi values in column 8. Examine each potential program star in your median image, using PBS's IRAF script `check` or your favorite image display technique. Delete crowded stars and stars very near background features such as H II regions and strong dust lanes from the `chip1.srt` file.

An alternative — possibly better — approach would be to plot sigma vs. magnitude, and pick that magnitude at which the TYPICAL sigma exceeds some pleasant value. Trim the star list at that magnitude, and then delete any stars with very large CHI or very large external/internal errors (large values in columns 6 or 9). This avoids the danger of including faintish stars whose errors have been randomly underestimated, or excluding brighter ones whose errors have been randomly overestimated.

Next, you want to pull the photometry for each of your program stars out of your `imageN.alf` files. To do this, edit your `chip1.mch` file, duplicating the first line in that file and changing the filename in the first line to `chip1.srt`. Make sure all the photometry files are listed as `imageN.alf` files and not .als files. Run **DAOMASTER** using `Minimum number, minimum fraction, enough frames = n,1,n` where n is the total number of files listed in the `chip1.mch` file, including the `chip1.srt` file on the line you just added. After the matching is done, answer `n` to every output option except `Simply transfer stars IDs?` Rename all the resulting `imageN.mtr` output files to `imageN.lst` files. The `imageN.lst` files contain the profile-fitting photometry of your program stars only.

The next step is very simple but somewhat tedious. You have to subtract off every star in every image in every filter in every data set except the program stars you selected above. Start up **DAOPHOT** and ATTACH `image1`. Run SUBSTAR and give it the `image1.psf` file and the `image1.alf` photometry file. At the `Do you have stars to leave in?` prompt, answer `y` and give it the `image1.lst` file you made up. Then ATTACH the subtracted image, `image1s`, and get aperture photometry of your program stars with the PHOTOMETRY subroutine. Use the appropriate `photo.opt` file described in Section 3.2 and the `image1.lst` file as the file with the positions. If necessary overwrite the `image1.ap` file created during the preparation for running **ALLFRAME**. Repeat the ATTACH-SUBSTAR-ATTACH-PHOT sequence for every image. When you have finished, create a file called `chip1v.lis` containing the list of the `imageN.ap` files for the *V* images. You can use the UNIX command,

```
ls *.ap >!  chip1v.lis
```

to make this file and then edit out the *I* band images. Make a similar `chip1i.lis` file for the *I*-band images.

Finally, you need to prepare a `chip1.inf` file. The simplest way to do this is with the **AIRMASS** program available from PBS. The program takes most of the information needed for the `chip1.inf` file from the headers of you images and runs in a fairly self-explanatory fashion. **AIRMASS** reads in the right ascension, declination, time of observation, and exposure time, and computes the heliocentric Julian date (HJD) of mid-exposure and flux-weighted mean airmass for the exposure (the latter is identically zero if you specify "Geocentric" for the observatory). These quantities are used in later steps to correct the calibrated magnitudes for exposure time, and to provide the HJD of each epoch for finding variables and their periods. If the output file, `chip1.inf` does not already exist, it will be created. If it does already exist, it will be appended to. Documentation for running **AIRMASS** is included in Appendix D should you need it. You should be sure to label (or have **AIRMASS** label) your *V* images as filter 1 and *I* images as filter 3 for later use.

4.2 Getting Growth Curves with DAOGROW

DAOGROW is described fully in Stetson (1990). Read this paper. **For key project data, make sure you have a version of DAOGROW which does not extrapolate the magnitude to twice the largest aperture.** If you are in doubt about which version you have, type the Unix command

```
grep xyz daogrow.f
```

You will get back a line like this

```
        cum(l) = 0.     !xyz
```

or like this

```
c       cum(l) = 0.     !xyz
```

If you see the latter, edit the file and REMOVE the c from the first column. Then remake **DAOGROW**.

Run **DAOGROW**. Give it your `photo.opt` file, your `chip1.inf` file, and your `chip1v.lis` file. Have **DAOGROW** solve for 4 parameters, and give it a value of 0 for the remaining 5th parameter and a maximum error of 0.05. **DAOGROW** will create several output files. These are `chip1.gro`, which is a summary of the data and growth curves for all images, `chip1.see`, which contains model growth curves suitable for plotting, `imageN.tot`, which contains the total instrumental magnitude of all the stars at the largest aperture (which should be 0.5″ if you used the recommended `photo.opt` file), `imageN.cur` which contains growth curves, and `imageN.poi` containing the actual observed magnitude differences in a form suitable for plotting.

At this point you should examine the growth curves to make sure everything is all right. Plot up the data in `image1.poi`, and overplot the curves from `image1.cur`. You may want to remove any outlyer stars from your program star list, although the robust software SHOULD ignore them adequately, saving you some work. IF you do decide to try removing outlyer stars, to help figure out which stars are which, look through your `chip1.gro` file. The magnitudes listed for each star are magnitudes at each aperture corrected to a 0.5″ aperture with the adopted growth curve. Any star that shows a large variation in these corrected magnitudes with respect to the other stars, might better be removed from the list. Remove the problem stars from your program star list and `imageN.ap` files, and rerun **DAOGROW**. Iterate examining growth curves, removing problem stars, and running **DAOGROW** until you are happy with the growth curves.

Another new procedure you should try is to include the standard globular cluster growth curves generated by PBS along with your local standards. To get these FTP anonymously to `ftp.dao.nrc.ca`, `cd pub/staff/pbs`, `binary`, `get templates.tar`, `bye`. The tarfile contains eight files representing the template growth curves: `vN.ap` and `iN.ap`, where N=1-4. To use these, simply add the appropriate ??.ap file to each of your normal *.lis files, and run **DAOGROW** normally. These template growth curves take the form of concentric aperture photometry for one image of nearly infinite weight. When you run **DAOGROW**, each of the galaxy growth curves will be allowed to differ by one degree of freedom at small radii, but will be forced to approach the template growth curve asymptotically at large radii.

When you finish with the V images, rerun **DAOGROW** with the `chip1i.lis` file to make growth curves for your I-band data.

As of this writing the "best" way to generate growth curves for WFPC2 data is still being explored. Some experimenting with your galaxy is probably in order.

4.3 Calibrating the Local Standards

To calibrate your local standards you need to make yet another list of your program stars and put this in a file called `chip1.fet`. The simple way to do this is copy your `chip1.srt` file to `chip1.fet`. Insert a line containing a zero and the name of the star before each line containing photometry (PBS will give his software to insert these lines to anyone who requests it). An example of the first few lines of the chip2.fet file I created for NGC 4414 is given below.

```
0n4414_2-5189
  5189   309.781   371.431   14.101   0.0112   0.   38.   0.639   0.026
0n4414_2-5940
  5940   283.651   559.38   15.128   0.0121   0.   38.   0.679   -0.008
0n4414_2-5174
  5174   668.277   369.589   16.493   0.0148   0.   38.   0.74   -0.026
0n4414_2-4653
  4653   537.008   321.654   16.333   0.0157   0.   38.   0.764   -0.002
0n4414_2-6006
  6006   205.691   608.083   16.404   0.0164   0.   38.   0.815   0.043
```

The next step is to run **COLLECT**. Name your output file `chip1.obs`. Give **COLLECT** your label for each filter number, probably 'v' and 'i'. Mid-times for first and last exposure are irrelevant, so something like 00 00 works fine. The file with exposure information is your `chip1.inf` file. The critical radius is the number of pixels within which a star must fall to be identified as the same star **COLLECT**. Since your program stars should be uncrowded by design, something like '1' should do nicely. Your file with positional transformations is `chip1.mch` and you just created the fetch file above. Next give **COLLECT** each of your `imageN.tot` files and `imageN.alf` files in turn. **COLLECT** will also create a `chip1.apc` file which contains the aperture corrections for your galaxy.

When **COLLECT** is finished you need to create a `chip1.clb` file. This file contains the equations for the transformation to the standard system, and should look as follows.

```
I1=M1
I3=M1-M3
O1 = M1 + A1 + A2*I2 + A3*I2*I2
O2 = M3 + B1 + B2*I3 + B3*I3*I3
A3 = -0.027
B3 = -0.025
A2 = 0.052
B2 = 0.063
A1 = 0.973
B1 = 1.870
```

In more readable terms, this represents

$$v = V + A_1 + A_2(V - I) + A_3(V - I)^2$$

and

$$i = I + B_1 + B_2(V - I) + B_3(V - I)^2.$$

v and i are the $0.5''$ instrumental magnitudes, A_1 and B_1 are zero points appropriate for data processed as described in Section 1, and the color term coeffients A_2, A_3, B_2, and B_3 are are taken directly from Holtzman *et al.* (1995) and multiplied by -1, since instrumental quantities are on the left here and standard quantities are on the right, and are identical for each chip. A table of recommended values for A_1 and B_1, from Hill *et al.* (1997), follows. Note that these are the long exposure zero points which are about 0.05 mag fainter (larger) than the short exposure zero points. Be sure to make sure that both **ALLFRAME** and DoPHOT photometry have been calibrated with the long exposure zero point before making any comparisons.

	PC1	WF2	WF3	WF4
V	0.969	0.957	0.949	0.973
I	1.863	1.822	1.841	1.870

There is one more file to create before running **CCDAVE**, which will apply the calibration to your program stars. Make a file called `chip1.lib` with the star names, one per line, starting in column two, with a single header line consisting of

```
 2 INDICES:                    V                 I
```

You can easily make the `chip1.lib` file by

```
grep ^0 chip1.fet | sed s/^0/"  "/ > chip1.lib
```

and adding the header above.

Run **CCDAVE**. Name the output file `chip1.net`. The file with the observations is `chip1.obs`. There should be no non-library stars in the file at this point, so it doesn't matter if you include these non-existent stars or not. Copy the output file `chip1.net` to `chip1.lib`, overwriting the previous library file. You might also want to edit out of the library file any star showing poor frame-to-frame agreement, as evidenced by a large number in column 2. Now your program stars are local standards which will be used to calibrate all the other stars, including Cepheids, in your data set.

4.4 Applying the Calibration

As I write this, some of the software described in the next sections is being extensively updated. Check with PBS on the status of the software before spending a lot of time on this.

A program currently called **TRIAL** will do the rest, calculate and apply aperture corrections, convert instrumental magnitudes to the standard system, find the weighted mean magnitude for each star over all epochs, and if desired, look for variable stars and output files containing magnitudes at each individual epoch for each variable candidate. **TRIAL** will also return fitted light curves, periods, amplitudes, and flux-weighted (intensity averaged) mean magnitudes. The last file that needs to be created for this grand finale is called `chip1.prs`. This file gives **TRIAL** information about which images are part of a CR split pair, and contains three columns, the first two containing the paired image names and the third the weight to be given that pair. If each image is taken with the same filter, the weight should be 1. If a V and I image are paired for some reason, the weight should be more like 0.5, which is the ratio of the amplitude of the V-band light curve to the I-band light curve for Cepheids.

Now run **TRIAL**. It needs `chip1.obs`, `chip1.tfr`, `chip1.prs`, `chip1.lib`, and `chip1.fet` files, one each at the first several prompts. For the first run through **TRIAL** give it values of 1 1 for the sigma multipliers and something like 1 10 8 28 for the limits on variability, weight, period, and magnitude. You will need to adjust these later if you want to use **TRIAL** to look for variables. **TRIAL** uses the local standards in the `chip1.lib` file to calculate zero point offsets between the standards in the library and the standards in the photometry file, and then applies this offset to each star. This offset is the aperture correction for the frame and will be printed to the screen and the output `chip1.zer` file. **TRIAL** also uses the exposure times in the `chip1.inf` file to calculate the correct magnitudes for your stars. **TRIAL** will create a `chip1.fnl` file, containing the V magnitude and $V - I$ color for each star averaged over all epochs, a `chip1.zer` file, which contains the zero point offset for each local standard in each frame and the mean offset, i.e. the aperture correction, for each frame, and a `chip1.vry` file which contains information on any variables found. In addition **TRIAL** outputs a `chip1v.X` file containing calibrated photometry at each epoch for each suspected variable, where X is equal to star ID. **TRIAL** will print sigma multipliers to the screen when it is finished. Multiply these new sigma mulitpliers by the ones you used previously, and rerun **TRIAL** using the product as input for your next sigma multipliers. You may wish to continue this iteration until the output sigma multipliers are approximately 1.

4.5 Quality Assurance

Aperture corrections are difficult to calculate accurately, and the Team has recently adopted a procedure with built in quality assurance. This procedure is described by Jeremy Mould and PBS and is included in Appendix E. Note that most of the files you put together for running **DAOGROW** should also be sent to PBS so you can check each other. If your aperture corrections are within a few hundreths of PBS's, use his aperture corrections, i.e. get his `imageN.tot` files for your galaxy for use in your calibration.

As an independent cross check on the photometry, you should compare your local standards and Cepheid candidates (see next section) with those reduced in parallel with DoPHOT. If there are problems with the comparison, check with the P.I.s and then PBS and Abi Saha to try to locate the source of the problem. Currently, the comparison often shows differences of up to 0.15 magnitudes at I, and the reasons for this are being explored.

5 Finding Variables

This section concentrates on using **TRIAL** to locate variables in data reduced with **ALLFRAME** and calibrated as above. There are other variable finding programs written and maintained by team members, including Shaun Hughes and Abi Saha. **TRIAL** can generate calibrated epoch-to-epoch photometry for stars found by any method by creating a file called `force.dat` and including the stars' ID numbers, one per line.

After running **TRIAL** once as described in the previous section, you should plot up the variability index vs. magnitude. The variability index is given in the `chip1.fnl` file in column 12 and magnitude in column 4. Decide how faint you want to go and how small a variability index you are interested in. Then make a histogram of number of stars as a function of total weight to decide how small a total weight you are interested in. Run **TRIAL** again using these values. **TRIAL** will select all stars meeting these criteria and create several output files containing calibrated epoch-to-epoch photometry, named `chip1v.X`, where X is the ID number of the star in question. The first 100 lines of this file contains information on the best fitting light curve for the variable with the photometry following. **TRIAL** will also create a file called `chip1.vry` which contains a summary of each variable based on Cepheid light curve fitting, including possible periods, mean magnitudes, amplitudes, and errors, along with a score showing how likely each solution is. The details of the light curve fitting procedure can be found in a recent paper, Stetson (1996).

Now that you have a list of variable candidates you'll want to perform several more steps to confirm these candidates. First, examine the median image to make sure that each Cepheid is in a relatively clean part of the image and not outrageously crowded, near a significant background feature, or too near the pyramid edge of the chip. You might also want to check the individual images to see if the Cepheid blinks and make note of any frames in which the variable is contaminated by a cosmic ray hit. Next you'll want to plot up and examine the light curves for each likely period. Also check variables found in the DoPHOT photometry and have the DoPHOTer for your galaxy check yours. Remove any variable candidates that don't appear to be good variables in both sets of photometry. Send off a copy of your final list to John Graham for confirmation of Cepheid status and periods.

Even though **TRIAL** produces magnitudes for each Cepheid, after you've settled on your final variable list, you need to calculate the intensity-averaged and phase-weighted mean magnitudes, as per team policy. Since the **ALLFRAME/TRIAL** procedure produces magnitudes for each image, you'll first need to average the magnitudes in each CR split pair, excluding magnitudes that are CR contaminated. Then calculate the intensity-averaged magnitude with

$$m = -2.5 \log \sum_{i=1}^{n} \frac{1}{n} 10^{-0.4 \times m_i}$$

and the phase-weighted magnitude with

$$m = -2.5 \log \sum_{i=1}^{n} 0.5 \left(\phi_{i+1} - \phi_{i-1}\right) 10^{-0.4 \times m_i},$$

where n is the total number of observations, and m_i and ϕ_i are the magnitude and phase of the ith observation in order of increasing phase. For variables with uniformly sampled light curves, these two magnitudes coincide. Phase-weighted magnitudes, however, provide a more accurate estimate of the mean magnitude when the light curve is not uniformly sampled.

Since there are typically few epochs with I data, the resulting poor phase coverage makes both the intensity averaged and phase weighted magnitudes poor representations of the I mean magnitude. Cepheids have very good correspondence between V and I light curves, and the ratio of the V to I amplitude is 1:0.51. The I mean magnitudes can be corrected by calculating the mean V magnitude at only those epochs were I observations were also made. The difference in the all-epoch V magnitudes calculated above and the V-and-I-epoch V magnitudes should be multiplied by 0.51 and then added to the average I magnitude. For a typical galaxy with 12 V epochs and 4 I,

$$I_{12} = I_4 + 0.51(V_{12} - V_4).$$

After performing these calculations, send off the magnitudes and periods for each variable to Barry Madore, who will run them through the official Key Project PL fitting software. A thorough description is included in Ferrarese *et al.* (1996). Use these numbers in your paper. To find the distance yourself, for unofficial purposes, do a least squares fit to

$$V - \mu_V = -2.76(logP - 1) - 4.16$$

and

$$I - \mu_I = -3.06(logP - 1) - 4.87$$

to find μ_V and μ_I. To correct for reddening and get the true distance modulus μ_0 for your galaxy, you also need to know

$$\mu_0 = \mu_V - A_V = \mu_I - A_I,$$

which requires that

$$\mu_V - \mu_I = A_V - A_I \equiv E(V - I).$$

The Cardelli *et al.* (1989) extinction law, applied to the Johnson V and Cousins I passbands, gives

$$A(I)/A(V) = 0.773 - 0.587/R_V.$$

The project has assumed a value for $R_V = R_V^{LMC} = 3.30$. This gives you enough information to find the distance to your galaxy.

6 Archive Contributions

The Key Project Team maintains an online archive with URL

```
http://www.ipac.caltech.edu/H0kp/
```

This archive contains, among other things, ASCII data tables from Key Project papers. After your paper is accepted for publication, pass along an electronic copy, preferably ASCII, of your data tables to Nancy Silbermann for inclusion in the archive.

7 References

- by P. Stetson
  - 1987 PASP 99, 191. **DAOPHOT** publication.
  - 1990 PASP 102, 932. **DAOGROW**.
  - 1994 PASP 106, 250. Includes **ALLFRAME** description.
  - 1996 PASP 108, 851. Variable finding and light curve fitting.
  - User's Manual for DAOPHOT II (very useful)
  - "Guinea pig" notes. included in **DAOPHOT** software as ccdpck.man. These are old, incomplete, and not necessarily up to date, but still quite useful.
- the DoPHOT reductions cookbook (available from the Key Project archive)
- Other potentially useful sources of information are available from the Key Project Archive maintained by Nancy Silbermann (nancys@ipac.caltech.edu). Check the latest README file for information.
- Calibration
  - Hill *et al.* 1997, ApJ, in press. Everything you wanted to know about the calibration and more, specifically for Key Project.
  - Holtzman *et al.* 1995 PASP, 107, 156. Calibration of WFPC2.
  - Ferrarese *et al.* 1996, ApJ, 464, 568. First paper with standardized procedures described, M100.
  - Cardelli *et al.* 1989 ApJ 345, 245. Reddening law.

I am grateful to Peter Stetson, without whom this document would not have been possible. Information gleaned from his answers to many questions and help with many problems in learning **ALLFRAME** make up the bulk of this document. Thanks also go to John Graham and Nancy Silbermann, who have been willing guinea pigs using early versions of this cookbook to reduce their data and have provided many useful suggestions for improvements and additions.

A Checklist

- Data Processing
  - Flag bad pixels using mask made from vignetting and DQFs.
  - Restore integrated flux integrity with the pixel area file.
  - Multiply data by 4 and convert to short integer format.
- Prepare and Run **ALLFRAME**.
  - Make medianed image
    - Check to make sure your **DAOPHOT** options are appropriate.
    - Run FIND and PHOT on each image.
    - Use **DAOMATCH** or by hand make initial guess at coordinate transformation between images.
    - Use **DAOMASTER** to refine transformation.
    - Use **MONTAGE2** to create the median $V$and Iimages. Make note of coordinate offsets. Add sky value. Sum $V$and Iimages.
  - Making the star list
    - Run FIND and PHOT on the medianed image.
    - Run **ALLSTAR** on the medianed image.
    - ATTACH subtracted image and pick up additional stars with FIND and PHOT.
    - Use OFFSET, GROUP, APPEND to combine star lists and run result through **ALLSTAR**.
    - Repeat last two steps as necessary, adding stars by hand on the last iteration.
    - Finish the star list by applying the OFFSETS from the **MONTAGE2** output from above.
  - Running **ALLFRAME**.
    - Create .als files for each image. Link psf files for each image.
    - Run **ALLFRAME**.
    - Wait.
    - Twiddle parameters, add stars, and run **ALLFRAME** again.
- Use **DAOMASTER** to match up **ALLFRAME** output as needed.
- Calibrate data.
  - Select list of program stars
  - Run **DAOGROW** to get growth curves
  - Calibrate local standards
    - Create `chip1.fet` file
    - Run **COLLECT**

        * Create `chip1.clb` file
        * Run **CCDAVE**, record sigma multipliers, and rerun.
    - Use **TRIAL** to apply calibration to all stars.
    - Cross check growth curves with PBS and calibrated photometry with DoPHOT.

- Find variables and periods
    - Find optimal parameters for variable finding in **TRIAL**.
    - Confirm candidates on image and light curve quality, and presence in both **ALLFRAME** and DoPHOT data sets.
    - Calculate mean magnitudes
    - Send magnitudes and periods to Barry Madore for PL fitting, and fit it yourself if desired.

- Write paper, submit to Team, publish, send e-copy of data tables to Nancy Silbermann for inclusion in the Team archive.

B Frequently Asked Questions

Q: I have run into a problem that is not covered in this document and am not sure what to do about it. Who can help me?

AT: Send email to PBS at Peter.Stetson@hia.nrc.ca and cc me at aturner@as.arizona.edu. He'll help you with your problem, and I'll add it to this FAQ to help the next person who encounters it.

Q: I wonder if one really wants to go for 200 iterations on ALLFRAME. . . . hardly anything of significance seems to be added after the first 50 yet it takes time for ALLFRAME to relentlessly move to its elegant conclusion.

PBS: It is indeed reasonable to cut back the maximum number of iterations. I put 200 as the default because it is the horrendously largest number I could ever think of anybody using. I would say that numbers in the range 25 - 50 are entirely reasonable. Unless your starting guesses were horrible, most of your good stars will converge in the first dozen or so iterations. After that, it is mostly the problematic stars dithering around at the 0.01 mag level. One really should run a few experiments to confirm this impression.

Q: While making the star list from a medianed image, I have at times been using what seems like a very high threshold (e.g. 17) in spite of the fact that FIND has the correct values for the read noise and gain and is told I've averaged half the number of frames I medianed. I chose this threshold by picking a value at/near the 'elbow' of the threshold vs. # of stars curve, and it seems to do a good job of picking out all the stars on the image. On other sets of images (same data set different chip–PC in this case) a threshold of 5 does the same job. I am confused. Can you think of a reason this is happening, besides getting the gain and/or read noise wrong (which I've triple-checked)?

PBS: I, too, have found that if I use a "statistically correct" (or "SC") threshold on medianed images I get an unrealistically large number of false detections. I do not know why this would be, at all, at all. Unless conceivably we are starting to see pattern noise in the images. Most of our exposures are matched up, pixel to pixel, so any errors in the bias, the superdark, or the flat field will NOT be averaged out in the montaging. It would be of interest to see whether the problem persists when completely dis-registered images are montaged.

Q: I'm having trouble with the 12 and 20 constant geometric corrections. Here's what I've been trying:

Start with a subset of my full star list that contains only the brighter stars (but make sure it is well sampled over the frame). Run daomaster with geometric coeff=6. Run the match-up radius down to 1 pixel. Write out the .mch file. The rerun on same star list with coeff=12. Do one iteration with match-up radius = 1 pixel. Write out the .mch file. Now run with coeff=20. Start at 1 pixel and try working down slowly. As soon as I get much below 1 pixel, I start rapidly losing stars. Near 0.5 it gets so bad that if I repeat at the same radius, I lose 1/2 the number of stars each time until there are no more matches. Extra iterations of coeff=12 don't help, they do the same thing.

I'm trying this on two photometry files. Each one contains coordinates and magnitudes obtained from previously averaging results of 7 images of the same field, so the positions ought to be pretty good. I'm requiring the stars to be found in both files. I tried starting with a subset which contained only stars with no neighbors within 5 pixels, but that didn't do any better. Am I overlooking something?

PBS: Try starting your subsequent runs with an initial radius rather larger than 1 pixel. The meaning of "critical radius" is this star in the input file agrees with that star in the master list within the radius. In daomaster iterations 2 through n, the "master list" position is the average of all available input positions. On the first iteration ONLY, the master list position is the position in frame 1. So suppose you've got a star in frame 1 and a star in frame n, each of which is 1 pixel from the average position. For iteration 1 they are TWO pixels apart. So your first critical radius must be at least twice the final critical radius you plan to use. Then, you must make sure to do several different radii. When there is only a small overlap between frames, that is there are few stars in common, and those near an edge or corner, it is very easy for there to be a lot of mismatches with the first, largest critical radius. Therefore, on the first daomaster iteration, the program is allowed to tweak only the zero-points, and even that only if there is a certain minimum number of stars matched. On subsequent iterations, it is allowed to tweak, respectively, 6, 12, and 20 coefficients if the minimum number of stars is matched, and that minimum number drops by one for each iteration. Therefore, if you have fewer than 20 matches in some given frame, it takes quite a few iterations before all 20 coefficients are being adjusted. Specifically, it will tweak NCOEFF coefficients if NMATCH - NCOEFF is greater than MIN, where NMATCH is the number of stars matched in that frame, and MIN starts out at 18 and drops by one per iteration.

Bottom line, even if you've gotten down to a critical separation of 1.0 when fitting 6 coefficients, when you do 12 use radii 2.0 1.8 1.6 1.4 1.2 1.0 0.9 0.8 0.7 0.6 0.5 or something like that.

Alternatively, PBS ruefully admits that he has found some subtle bugs in the treatment of quadratic and cubic transformations in early generations of DAOMASTER and ALLFRAME. Check with him to make sure you are using the most current version of each.

Q: I had a lot of trouble with **DAOMASTER** when trying to gear up for **ALLFRAME**. Just could not get good transformations for 40 frames.

PBS: In a case like the present one, it is often a good idea to sneak up on the solution with **DAOMASTER**. If you know for a fact that there is no rotation, run **DAOMASTER** with Your choice = 2, and output a new .mch file (if there may be significant rotation, start with Your choice = 4). Also, be very, very restrictive with your acceptance criteria: e.g. require that the star be found in at least 8 or 10 images, and that sigma be less than 0.3 or something. With the new .mch file, run daomaster again with Your choice = 6 and create a new .mch file. Then put this .mch file into **DAOMASTER** and run with Your choice = 20 and more generous acceptance criteria, to produce the .mag and .mch files for **ALLFRAME**. The problem with HST data is that there are so many cosmic rays, that even if the preliminary transformation is dead wrong, there is still a very good chance of finding some sort of "detection" near the star's predicted position. The small number of free parameters and the stringent acceptance criteria in the first pass or two force the program to come up with a truly valid solution, not just one that lines up a whole bunch of cosmic rays.

C Commonly used DAOMATCH, DAOMASTER, and ALLFRAME Output files

C.1 A .mch file

Both **DAOMATCH** and **DAOMASTER** output a .mch file which contain coordinate transformation of the form:

$$x_2 = A + Cx_1 + Ey_1$$

$$y_2 = B + Dx_1 + Fy_1.$$

The first few lines of a sample output .mch file follows.

```
=============================================================================
 'u101a.als'   0.000   0.000  1.00000  0.00000  0.00000 1.00000  0.000  0.0972
 'u102a.als'  -0.080  -0.133  1.00043  0.00000 -0.00007 1.00040  0.105  0.9402
 'u103a.als'  -0.161   0.235  0.99986 -0.00007  0.00010 0.99973 -1.367  0.4068
 'u201a.als'  -0.142   0.094  0.99996  0.00000  0.00007 1.00003 -0.011  0.0749
 'u301a.als'   0.248   0.238  1.00012  0.00003 -0.00004 1.00018  0.045  0.0885
 'u302a.als'   0.083   0.113  1.00054  0.00001  0.00001 1.00055  0.394  0.9072
=============================================================================
    (1)          (2)     (3)     (4)      (5)      (6)      (7)     (8)      (9)
```

Explanation of Columns:

1. File name objects' coordinates are/were taken from.
2. Coefficient A (as described above)
3. Coefficient B
4. Coefficient C
5. Coefficient D
6. Coefficient E
7. Coefficient F
8. Crude estimate of the magnitude offset from the master frame.
9. Crude estimate of the star-to-star scatter in δmag.

DAOMASTER output may have additional columns following column 9 which contain coefficients for the quadratic and cubic distortion terms if these were requested.

C.2 A .mag file

DAOMASTER creates a .mag file if you request a file containing mean magnitudes and scatter as output. The first few lines of a sample .mag file follow.

```
==============================================================================
 NL   NX   NY  LOWBAD HIGHBAD  THRESH     AP1  PH/ADU  RNOISE    FRAD
  1  800  800    27.0 32000.0   58.68    3.00    1.75    4.00    2.00

 30089  760.692  62.386 20.127  0.2507  0.1037   6.  0.915 -0.012  0.23  0.667
   172  772.768  62.509 19.050  0.1224  0.1107   8.  0.885 -0.033  0.56  0.625
   188  763.974  63.302 18.513  0.0858  0.1619  11.  1.059  0.037  0.74  0.455
   200  768.406  63.790 19.624  0.1194  0.3220  10.  1.288  0.027  0.87  0.400
   199  756.208  63.799 18.946  0.0871  0.1199  11.  1.212  0.018  0.55  0.636
   198  605.568  63.823 18.933  0.1584  0.2681  11.  1.304  0.017  0.92  0.727
 30100  585.812  64.254 19.606  0.1364  0.2495   6.  0.787  0.075  0.83  0.833
==============================================================================
   (1)     (2)     (3)    (4)    (5)     (6)     (7)   (8)    (9)   (10)   (11)
```

Explanation of Columns:

1. Star ID
2. Average X coordinate, in system of master frame
3. Average Y coordinate, in system of master frame
4. Robust intensity-weighted mean instrumental magnitude
5. Standard error of the mean magnitude, based on individual frame sigma's
6. External standard error of one measurement, based on frame-to-frame repeatability
7. Number of frames in which the star appeared
8. Weighted average CHI value
9. Weighted average SHARP value
10. Variability index: ratio of external error to internal error
11. Blunder index: fraction of residuals that are positive

C.3 A .cor or .raw file

DAOMASTER creates a .cor file if a file with corrected magnitudes is requested as output. All instrumental magnitudes in this file are corrected to the system of the first image by an additive constant. **DAOMASTER** creates a .raw file if a file with raw magnitudes and errors is requested. This file contains the raw instrumental magnitudes from each image. Both of these types of files have the same format. The first few lines of a .cor file are shown below. For the purposes of explaining the file and avoiding confusion, most of the magnitudes and errors have been edited out and replaced with ellipses.

```
======================================================================
 30089  760.692   62.386  20.1690   0.2990  19.1093   1.7340  . . .
                         . . .   20.0072   2.5060  16.5978   1.2110
                           0.9154  -0.0119
   172  772.768   62.509  18.9870   0.1340  19.1543   0.3130  . . .
                          . . .   19.4592   1.0250  18.5088   4.1230
                           0.8848  -0.0329
   188  763.974   63.302  18.6480   0.1760  18.3863   0.1980  . . .
                          . . .   19.3542   2.1350  19.0938   1.0200
                           1.0592   0.0375
======================================================================
   (1)    (2)       (3)     (4)       (5)      (6)      (7)
                                    (8)      (9)      (10)     (11)
                            (12)     (13)
```

Explanation of columns:

1. Star ID
2. Average X coordinate, in system of master frame
3. Average Y coordinate, in system of master frame
4. Instrumental magnitude in Image 1
5. Estimated standard error in magnitude 1
6. Instrumental magnitude in Image 2
7. Estimated standard error in magnitude 2
8. Instrumental magnitude in Image N-1
9. Estimated standard error in magnitude N-1
10. Instrumental magnitude in Image N
11. Estimated standard error in magnitude N
12. Weighted average CHI value
13. Weighted average SHARP value

If there are more than 6 input images, then a maximum of 6 (magnitude/error) pairs are put on a line. The actual FORMAT is

FORMAT (I6, 2F9.3, 12F9.4:/ (24X, 12F9.4))

Any software people want to write to use these files must be able to read this format.

C.4 A .alf file

ALLFRAME outputs a .alf file for each input image. The format of a .alf file is identical to that of a .als file. See Appendix IV in the DAOPHOT II manual for more information.

D Running AIRMASS

From PBS:

Suppose you want to add the following images to n925_3.inf:

```
ue01t_3.pix
ue02t_3.pix
ue03t_3.pix
ue04t_3.pix
```

Here's what you do. Items entered by you are marked with an '*'; the rest are generated by the computer.

```
% airmass                                                         *
                                  Output file name: n925_3        *

                                       Geocentric = 0
               Dominion Astrophysical Observatory = 1
                                        Mauna Kea = 2
                                     Cerro Tololo = 3
                                        Kitt Peak = 4
                                         La Palma = 5
                                            Other = 6

                                 Enter observatory: 0             *
                                Readout time (sec): 0             *
                        Shutter-timing error (sec): 0             *
                                        RA keyword: RA_TARG       *
                                       Dec keyword: DEC_TARG      *
                                   Equinox keyword: EQUINOX       *
                                    Filter keyword: FILTNAM1      *
                                      Date keyword: DATE-OBS      *
                                        UT keyword: TIME-OBS      *
                                  Exposure keyword: EXPTIME       *
                                   Filename prefix: u             *
                 Intended name of calibration file: n925_3        *
                                   Photometry file: e01t_3        *

     ue01t_3...



                                      Picture size:   800  800
```

```
                                               Date:  1994  8  8
                                        Object name: NGC 925       *

                             Coordinates = 02 27 06.8 +33 35 07  (2000)


 F555W

                                   Enter filter number: 1              *
                                         UT (start):  20 51 16.0
                                           Exposure: 1300.000000000

21 02 06      1300   X = 0.000  at HJD = 2449573.3765

                                      Photometry file: e02t_3       *

    ue02t_3...



                                         Picture size:   800  800
                                               Date:  1994  8  8
                             Object name (default NGC 925): <CR>       *

                             Coordinates = 02 27 06.8 +33 35 07  (2000)

                                             Filter: 1
                                         UT (start):  22  8 16.0
                                           Exposure: 900.0000000000

22 15 46       900   X = 0.000  at HJD = 2449573.4277

                                      Photometry file: e03t_3       *

    ue03t_3...



                                         Picture size:   800  800
                                               Date:  1994  8  8
                             Object name (default NGC 925): <CR>       *

                             Coordinates = 02 27 06.8 +33 35 07  (2000)


 F814W
```

```
                                     Enter filter number: 2            *
                                            UT (start):  22 27 16.0
                                              Exposure: 1300.000000000

22 38 06      1300   X = 0.000  at HJD = 2449573.4432

                                        Photometry file: e04t_3        *

     ue04t_3...



                                           Picture size:   800  800
                                                   Date:  1994  8  8
                      Object name (default NGC 925): <CR>          *

                     Coordinates = 02 27 06.8 +33 35 07  (2000)

                                                 Filter: 2
                                            UT (start):  23 45 16.0
                                              Exposure: 900.0000000000

23 52 46       900   X = 0.000  at HJD = 2449573.4951

                                        Photometry file: EXIT          *

 Good bye.

%
```

Easy, ain't it? You just generated this output file (with spacing altered to fit on this page–see examples in `ccd` directory for correct spacing).

```
 ue01t_3.imh    1  21 02  0.000  1300.0  2449573.3765 n925_3     NGC 925
 ue02t_3.imh    1  22 16  0.000  900.00  2449573.4277 n925_3     NGC 925
 ue03t_3.imh    2  22 38  0.000  1300.0  2449573.4432 n925_3     NGC 925
 ue04t_3.imh    2  23 53  0.000  900.00  2449573.4951 n925_3     NGC 925
```

Notes:

- The output file above was edited to fit on this page. The format is strictly fixed at

  ```
  FORMAT(1X, A30, I3, F4.0, F3.0, F8.1, 2X, A30, 2X, A30)
  ```

  See the example file included in the **DAOPHOT** package for an example.

- The program computes the heliocentric Julian date of mid-exposure, and flux-weighted mean airmass for the exposure (the latter is identically zero if you specify "Geocentric" for the observatory).

- If the output file does not already exist, it will be created. If it does already exist, it will be appended to.
- You can use this for your ground based data, too. If your observatory is not among the list of choices, enter 6 ("Other") and be prepared to enter the observatory's longitude and latitude (west longitude > 0, east longitude < 0). Or you can contact me and ask me to add your favorite observatory to the list of options.
- As usual with many of my programs, if you respond to a prompt with an EOF character (CTRL-D or CTRL-Z, usually), it will back up to the previous question.
- If one or more of the required keywords do not exist in your FITS header, type <CR> upon receiving the request; you will be prompted to enter the necessary data manually at the appropriate time. The correct keywords for HST data are indicated.
- "Filename prefix" is any string of characters that is a common beginning to all your filenames; entering it here just saves typing it over and over again later. I could also add "Filename suffix" if there is enough uproar.
- "Intended name of calibration file" is the filename part of the .clb file you intend to use, if you are using ccdstd and/or ccdave. Otherwise it is irrelevant and you can enter anything you want.
- The first time the program encounters a new filter ID string (e.g. F555W or F814W in FILTNAM1) the program will ask you which number you wish to assign to that filter; after that it will recognize the filter name and will remember the number all by itself.
- If you are reducing ground-based data, and have an image that is the sum or average of several exposures taken consecutively, as we used to do, enter a <CR> for the exposure time keyword, and when it asks you to enter the exposure time for the image, enter "7x350" if you have seven 350-sec exposures in the average, for instance. It will use the exposure time and the readout time that you have specified to calculate when the middle of the middle exposure should have been, and calculate the flux-weighted mean airmass for the whole thing.
- **AIRMASS** has recently been modified so that when it sees that the RA keyword is RA_TARG, it will assume it is dealing with HST data, and won't ask for the rest of the keywords.
- PBS has modified **AIRMASS** to compute azimuth as well as airmass, with azimuth between airmass and exposure time in the output `chip1.inf` file. All programs which utilize the .inf file have been instructed to accept BOTH the old AND the new formats, so there will be no additional hassle for people with the old airmass executables and old .inf files. It is essential to note that anyone using the new **AIRMASS** ON GROUND BASED DATA should also be using the new **CCDSTD**, **CCDAVE**, and **TRIAL**. For HST it doesn't matter.
- The last character string on each .inf line (e.g. NGC4559, NGC2419, NGC4147) should be the exact name of the .mch and .fet file appropriate to that observation.

```
<directory-name>:
```

should be included if these files are not in the same directory as the .inf, .obs, etc. files.

E Aperture Correction Determination and Quality Assurance

REPORT ON COMPARING APERTURE CORRECTIONS FOR NGC3621
Jeremy Mould & Peter Stetson
7 May 97

For reliable calculation in crowded fields aperture corrections are best defined by equations (1) & (2) (Stetson PASP 102,932), rather than (1) alone:

$$\text{Ap cor} = \text{Mag (0.5")} - \text{PSF mag} \quad (1)$$
$$\text{Mag (0.5")} = \text{Mag (best aperture)} + \text{Growth curve correction to } 0.5'' \quad (2)$$

It was not hard to verify that our calculations of small aperture mags and PSF mags were quite close ($\lesssim$ 0.01 mag), isolating any differences resulting from different procedures in deriving our respective growth curves.

As a result of our investigations, we recommend the following procedure for each galaxy in future.

RECOMMENDED PROCEDURE WITH BUILT IN QUALITY ASSURANCE*

1. Select 20-30 bright stars with clean images (or the most/best you can). Inspection on cr-cleaned V and I frames is essential. If in doubt, consult* JRM or PBS.

   I (PBS) have found it more useful to examine the residual image (subtract all stars from all frames, and then median-average the star-subtracted frames) than to examine the image with the stars still in — if the star subtracts cleanly then that means it truly has a stellar profile and there are no neighbors .OR. the neighbors are accurately accounted for. Then the aperture photometry is done on images from which all EXCEPT the aperture-correction stars have been subtracted.

2. Carry out aperture photometry in at least 0.2, 0.3, 0.4 & 0.5" apertures on each frame. Ensure that the sky annulus is at least 1" from centroid. Reject stars with skew values outside the range (-0.1,0.4)in the sky. We saw correlations between aperture corrections and skew. [Actually, this doesn't make a lot of difference if you have equal numbers of high sky and low skew values.]

3. Construct mean-over-all-epoch growth curves for V and I. You may wish to use DAOGROW. Send a copy of these curves to JRM*.

4. Send the starset, .alf files and images to PBS for his separate aperture correction determination.

5. For V and I form <Mag (best aperture)> for each star. Apply growth curve correction (2) to get <Mag (0.5")>.
6. Form <PSF mag> for each star.
7. Apply (1) to each star and take the average to obtain <Ap cor>.
8. Compare* with <PBS>. If the difference < 0.02 mag, adopt the PBS corrections and consider the range to indicate the systematic uncertainties. If the difference is larger, reexamine the starset and compare your growth curves with those which PBS can supply on request. The PC remains ultra-hard for accurate growth curve determination.