Stellar Photometry Using IRAF

W.E.Harris
September 2008

Part I: Aperture Photometry, Calibration, and Curve-of-Growth

Suppose you have an image called "pic.fits" that has lots of stars you would like to measure, or perhaps just one bright standard star -- the techniques of photometry are the same for both.

To CALIBRATE the photometry means knowing how to convert your "instrumental" measurements of brightness direct from the image, into its equivalent magnitude on a standard photometric system such as UBVRI or ugriz. The key to doing this is "aperture photometry", and that's the subject of the first part of this text. In the second part, we discuss a different way of doing photometry, namely "point-spread-function (PSF) fitting". Measuring brightnesses for things on the image that are NOT stars (galaxies, nebulae) falls into a third and more complex category of individual profile-fitting and surface-brightness measurement. Here we deal only with stars.

The basic principle of aperture photometry is that you enclose the star in a circular aperture of some radius r, and add up all the light in that circle. Then, you subtract off the amount of light contributed by the sky background in that same area, and you are left with the brightness of the star. The units of this measurement are in ADU and can be converted directly to the number of electrons captured by the CCD once you multiply it by the CCD gain factor.

The big advantage of aperture photometry is that it gives you an "absolute" measurement of the star's brightness, in true numbers of photoelectrons N(e-). This is what allows it to be converted to a true magnitude on the standard system. As we will see later, PSF fitting gives only "relative" brightnesses between stars (but PSF fitting has several powerful advantages of its own, which is why we end up using both).

NOTE: In all of the following, I am assuming you already know the basics of IRAF: e.g. how to move around between packages and menus, how to use "epar" to edit parameter files, how to set commands running either through epar or command-line, and so on. I am also assuming you have a standard image display running, such as ds9 .

Here are the steps to go through for aperture photometry in IRAF:

(0) First, inspect the image you are about to measure and find out certain things about it. You need to know particularly:

(a) the profile width of a typical star, conventionally measured as its full-width-at-half-maximum (FWHM): run "imexamine", and put the cursor onto several obvious stars. At each one, hit "r" or "a" to get the radial plot or printout of the star parameters. The best estimate of the FWHM is the second-last number on the screen that "r" or "a" produces (actually, that number is the FWHM of a Moffat profile that has been fitted to the star).

(b) the mean sky intensity and the standard deviation of the sky. Again, run "imexamine" on the image: move the cursor around to lots of different spots where there is blank sky, and hit "m" at each spot to print

out the mean, median, and standard deviation of the sky at that spot.
Average up the std.devs. to get a reasonable mean value.
If, however, there is a pretty big gradient of light across the image
(such as if there is a big galaxy in the field, or nebula with patchy
light) then you won't be able to settle on one sigma-value that is
truly valid for the whole frame. Just pick some compromise mean
and go with that.

(1) Within IRAF, go into the   noao/digiphot/apphot menu.
The first thing you should notice are several commands marked by @.
These are input parameter files used by the other commands in
the menu list, so you should set those up first.
First is "datapars" and it gives
some relevant numbers for the CCD and the raw image.
It should look something like this:

```
ap> lpar datapars
       (scale = 1.)             Image scale in units per pixel
     (fwhmpsf = 2.)             FWHM of the PSF in scale units
    (emission = yes)            Features are positive ?
       (sigma = 10.)            Standard deviation of background in counts
     (datamin = 0.)             Minimum good data value
     (datamax = INDEF)          Maximum good data value
       (noise = "poisson")      Noise model
     (ccdread = "")             CCD readout noise image header keyword
        (gain = "")             CCD gain image header keyword
   (readnoise = 5.)             CCD readout noise in electrons
       (epadu = 2.)             Gain in electrons per count
    (exposure = "")             Exposure time image header keyword
     (airmass = "")             Airmass image header keyword
      (filter = "")             Filter image header keyword
     (obstime = "")             Time of observation image header keyword
       (itime = 1.)             Exposure time
    (xairmass = INDEF)          Airmass
     (ifilter = "INDEF")        Filter
       (otime = "INDEF")        Time of observation
        (mode = "ql")
```

Then "centerpars", which is used by PHOT and determines whether it will
try to redetermine each object's (x,y) coordinates based on some weighting
scheme given by "calgorithm":

```
ap> lpar centerpars
  (calgorithm = "centroid")     Centering algorithm
         (cbox = 5.)            Centering box width in scale units
  (cthreshold = 0.)             Centering threshold in sigma above background
  (minsnratio = 1.)             Minimum signal-to-noise ratio for centering alg
    (cmaxiter = 10)             Maximum number of iterations for centering algo
    (maxshift = 1.)             Maximum center shift in scale units
       (clean = no)             Symmetry clean before centering ?
      (rclean = 1.)             Cleaning radius in scale units
       (rclip = 2.)             Clipping radius in scale units
      (kclean = 3.)             Rejection limit in sigma
    (mkcenter = no)             Mark the computed center on display ?
        (mode = "ql")
```

Then "photpars", which gives the aperture radius surrounding the star,
and a photometric zeropoint (which is an arbitrary number! but keep it at
25, which is the default iraf value). Notice in the example below that
8 aperture sizes are listed. This means that PHOT will measure the total
light in each of 8 concentric circles of increasing size. We will use
that later for "curve of growth":

```
ap> lpar photpars
```

```
   (weighting = "constant")     Photometric weighting scheme for wphot
   (apertures = "2,3,4,5,7,9,12,15") List of aperture radii in scale units
        (zmag = 25.)            Zero point of magnitude scale
     (mkapert = no)             Draw apertures on the display
        (mode = "ql")
```

And finally "fitskypars" which defines the annular region around each star that will be used to measure the local sky brightness:

```
ap> lpar fitskypars
  (salgorithm = "centroid")     Sky fitting algorithm
     (annulus = 25.)            Inner radius of sky annulus in scale units
    (dannulus = 10.)            Width of sky annulus in scale units
      (smooth = no)             Boxcar smooth the histogram
       (rgrow = 0.)             Region growing radius in scale units
       (mksky = no)             Mark sky annuli on the display
        (mode = "ql")
```

As always, you can read more about the guts of each routine with the internal iraf "help" command. e.g. "help fitskypars" will tell you all you need to know about what it really does.

(2) Now start doing photometry. And here you have two choices .... interactive mode, or non-interactive? Let's look at "interactive" mode first because it is simpler and more intuitive to develop a feel for what is going on. This is also a good mode to use for the standard-star images. We'll deal with non-interactive mode later under the PSF-fitting discussion.

Go directly to the command "phot" and set up its parameter file:

```
ap> lpar phot
        image = "pic.fits"      The input image(s)
      skyfile = ""              The input sky file(s)
      (coords = "")             The input coordinate files(s) (default: image.c
      (output = "default")      The output photometry file(s) (default: image.m
    (plotfile = "")             The output plots metacode file
    (datapars = "")             Data dependent parameters
  (centerpars = "")             Centering parameters
  (fitskypars = "")             Sky fitting parameters
    (photpars = "")             Photometry parameters
 (interactive = yes)            Interactive mode ?
    (radplots = no)             Plot the radial profiles in interactive mode ?
   (icommands = "")             Image cursor: [x y wcs] key [cmd]
   (gcommands = "")             Graphics cursor: [x y wcs] key [cmd]
```

In interactive mode, PHOT is pretty simple. Most of its internal parameters are switched off, and you can just go right to the image and start measuring stars one by one. Make sure you have your image loaded up in your ds9 display window and you are set to go.

In the iraf window, type phot
and if the cursor does not jump over into the display window, move it there. Move the cursor onto the star you want to measure. Now hit the spacebar. Some numbers should pop up into the iraf window, which give the measured magnitudes inside each aperture radius you specified. If you have more stars you want to measure, just move the cursor onto them one by one and hit the spacebar each time. Try a few just to confirm how it works.

When you have finished the stars you want, hit q (= quit) with the cursor still in the display window. A message will come up in the iraf window. Click on the iraf window, then hit w (= write; if you hit q again, it will terminate without saving anything). This finishes the routine.

(3) Clean up the output from PHOT. By default, the output goes into a file called pic.fits.mag.1 (if you run phot again on the same image, it will put the output into pic.fits.mag.2; that is, it does not over-write the previous file).

Look at the .mag file. You will see that it is a big complicated file with way too much stuff in it. Decide what numbers you actually want to save! In most cases, this will be -

- The (x,y) centered coordinates of each object
- The radius of each concentric aperture
- The measured magnitude (or magnitudes, if several apertures were used)
- The internal error on the magnitude.

You can generate a simpler output file (suitable for later plotting and conversion) by using the utility routine "pdump":

```
ap> lpar pdump
      infiles = "pic.fits.mag.1" Input apphot/daophot databases(s)
       fields = "xc,yc,rap,mag,merr" Fields to be extracted
         expr = "yes"           Boolean expression
     (headers = no)             Print field headers?
  (parameters = yes)            Print parameters?
      (inlist = "")
        (mode = "ql")
```

This set of parameters says that you want to save the coordinates, aperture radii, magnitudes, and errors.
NOTE: if you just run pdump as is, it will write its output to the screen, which is useless. To pipe the output to another file, you need to say

pdump > pic.apdat

or some other filename you prefer.

(4) Develop a "curve of growth".
Now, plot up the measured magnitude from PHOT versus the aperture radius r. Notice that for small r, the magnitude gets rapidly brighter with increasing radius, then it levels off so that for larger r, it asymptotically approaches the "total" brightness of the star at infinite radius (which by definition must enclose all of its light). This is the curve of growth.

The c-o-g can be accurately defined only for bright stars; for faint ones, the outer wings of the image rapidly fall below the sky brightness and are subject to random noise fluctuations. Also, notice that the c-o-g is subject to CROWDING: any other neighboring star that happened to fall within your measurement aperture will make your central star measurement brighter than it should be (it shows up in your c-o-g as a sudden jump in brightness at the radius where the contaminating star sits).

Use the c-o-g to adopt a FIDUCIAL RADIUS which is
(a) large enough to enclose about 95% of the total light, but
(b) not so large that it is subject to big fluctuations due to sky noise, or that you get "INDEF" answers for the magnitude.
Some experience and consultation will help here.

NB: one very good reason to use a rather large aperture for the standard stars will come up later, but it relates to the fact that they are short exposures of very bright stars, they may not have been taken with the telescope focus sharply tuned up, they may not have been guided exposures like the program fields were, and so on. All this means that the "cores" of the standard stars don't have the same structure (=PSF) as the stars in long, focussed, guided exposures. But if you get far enough out into the wings where all the light is comfortably included, all of that can be safely ignored. You can correctly compare big-aperture photometry of the standards with

big-aperture photometry in the program fields, but not small-aperture photometry.

To go deeper into this method, you can read more about the principles of curve-of-growth in Stetson 1990, PASP 102, 932. Some of Stetson's precepts for determining curve-of-growth are incorporated in the menu noao/digiphot/photcal.

(5) Do final aperture photometry.
Having used several bright standard stars to settle on all your parameters, measure all the standard stars in the entire night (or run of several nights) with these parameters.
The next step after this is to convert the aperture magnitudes into "true" standard-system magnitudes, which is described elsewhere.

Part II: Running Daophot within IRAF

Now we will discuss the other technique for stellar photometry -- PSF fitting. The basic principle of PSF fitting is that (unlike galaxies or nebulae) every star on the image has the same intrinsic profile shape: they differ only in their brightness or amplitude. All of them are just images of point sources of light, convolved with the telescope optics and (for ground-based telescopes) blurring by the Earth's atmosphere. This means you can establish a "standard profile" for the star images on an image and simply scale that standard profile up or down in amplitude to fit any individual star.

A huge advantage of PSF fitting is that it is much more immune to crowding than aperture photometry. The best-fit scale factor for any star can be determined just by using the pixels in the core of the star's profile, which are the brightest and contain most of the signal. You do not necessarily have to use the wings, because all stars have the same profile and so any portion of it will scale in the same way as any other portion. Profiles of nearby pairs of stars can overlap partially and you can still obtain accurate PSF scaling factors for both of them.

Suppose, as above, you have an image called "pic.fits". The package noao/digiphot/daophot within IRAF will do a superb job of finding and measuring all the starlike objects in the image, including both classic aperture photometry and photometry by profile fitting to a PSF (point spread function). Here is the basic sequence of steps that you will need to run.
Once you have become comfortable with the sequence, it will be easier to read the "help" files within IRAF/daophot and figure out other tricks.

(0) [Repeated from the previous section]
First, inspect the image you are about to measure and find out certain things about it. You need to know particularly:
(a) the FWHM of a typical star: run "imexamine", and put the cursor onto several obvious stars. At each one, hit "r" or "a" to get the radial plot or printout of the star parameters. The best estimate of the FWHM is the second-last number on the screen that "r" or "a" produces (actually, that number is the FWHM of a Moffat profile that has been fitted to the star).
(b) the mean sky intensity and the standard deviation of the sky. Again, run "imexamine" on the image: move the cursor around to lots of

different spots where there is blank sky, and hit "m" at each spot to print out the mean, median, and standard deviation of the sky at that spot. Average up the std.devs. to get a reasonable mean value.
If, however, there is a pretty big gradient of light across the image (such as if there is a big galaxy in the field, or nebula with patchy light) then you won't be able to settle on one sigma-value that is truly valid for the whole frame. Just pick some compromise mean and go with that.

(1) Prepare to run daophot by getting the initialization parameters together: with "epar", edit the files

centerpars
daopars
datapars
findpars
fitskypars
photpars

Particular things to pay attention to:

- In daopars: set function=auto (this will select the best PSF analytical model automatically from the choice of 6 models; see "psf" below in step 5). Also set "nclean=5" or something bigger than zero so that it will clean the PSF stars of their neighbors iteratively.

psfrad: I like to make the "psfrad" equal to something like 10 times the FWHM so that you can see the shape of the wings of the stars to a good distance outward. The wings won't determine the fitting, but it's useful to see them.

fitrad: Here also set "fitrad" to be something roughly equal to the FWHM. Various calculations in the literature show that this is an optimum -- small enough to avoid including much sky noise, but large enough to enclose most of the signal from the star. If the image is relatively uncrowded, you can make the fitting radius reasonably big, but if it is a very crowded field, you can make fitrad < FWHM and it will still be OK.

- In datapars: find the "gain keyword" and "readnoise keyword" from the image header and put those in here. Also, put in your best estimate of the standard deviation "sigma" of the sky noise. Finally, enter your best guess for the FWHM "fwhmpsf" of the stars. (See step 0 above.)

- In photpars: start with a rather small aperture radius (similar to the FWHM of the star profiles) and make the sky annulus reasonably generous, e.g. set the inner radius to 5 or 6 FWHM and make the annulus width equal to 2 or 3 FWHM. You might want to go back later and do larger-aperture photometry for calibration purposes, but at this first stage we are just setting up to do the PSF-fitting photometry.

(2) Find the candidate stars.

In most cases, the purpose of PSF fitting is to do it wholesale -- i.e. apply it to the hundreds, or thousands, of star all across the image. So you want to start by generating a catalog of the stars on the image, defined in an objective and uniform way.

The key to doing this is the parameter "thresho" in findpars. This parameter is in units of sigma(sky), i.e. the pixel-to-pixel standard deviation of the sky noise. You want to avoid "finding" random clumps of brighter-than-average sky noise, but you also want to make sure you are actually finding the real stars that are sticking up above the sky. The larger you make thresho the more faint stars you miss.

But the smaller you make thresho the more sky noise and false detections you let in. So it's a compromise. thresho=4 is a good starting point. Less than 0.1% of the sky pixels will be brighter than 4 sigma just by random Gaussian statistics.

Run "daofind". This generates a file with the default name "pic.fits.coo.1". (If you run it again, it will generate a second version pic.fits.coo.2 without erasing the previous one. The same is true of any of the other functions below -- they do not erase the previous versions of their files.)

(3) Do aperture photometry.

Run "phot". Do this in NON-interactive mode ("interac = no"). Also, do this with just ONE aperture size and not several; i.e. in photpars, specify apertur = (x) where (x) is about equal to the FWHM. The reason is that you are not trying to generate a curve-of-growth now; you are just trying to set up the definition of the PSF and you need a good initial guess at the magnitudes of the stars.

This will generate a file with the default name "pic.fits.mag.1".

(4) Select candidate psf stars.

Daophot creates a model psf by averaging together several bright, isolated stars in the image that are under your control to select. It then fits this combined, empirical bright star profile with a combination of an analytical function such as a Gaussian or Moffat function (which usually fits the core and bright parts of the star quite well) and an empirical lookup table (which fits the residual wings of the star that are usually in excess of the analytical profile).

You can pick out candidate psf stars by hand (this can be ugly, frustrating, and time-consuming), but there is a faster way to have daophot do it for you. This is the function "pstselect". Run pstselect in NON-interactive mode, and it will look for all the best bright, isolated stars that it can find. You set the number of candidates it looks for by the parameter "maxnpsf" in pstselect. The parameter for pstsel looks like this:

```
da> lpar pstsel
        image = "pic.fits"        Image for which to build psf star list
     photfile = "default"       Photometry file (default: image.mag.?)
      pstfile = "default"       Output psf star list file (default: image.pst.?)
      maxnpsf = 50              Maximum number of psf stars
     (mkstars = no)             Mark deleted and accepted psf stars
    (plotfile = "")             Output plot metacode file
    (datapars = "")             Data dependent parameters
     (daopars = "")             Psf fitting parameters
 (interactive = no)             Select psf stars interactively ?
    (plottype = "mesh")         Default plot type (mesh|contour|radial)
   (icommands = "")             Image cursor: [x y wcs] key [cmd]
   (gcommands = "")             Graphics cursor: [x y wcs] key [cmd]
```

First, however, there is one important decision you have to make here, which is to decide whether or not you want the psf to be a function of location on the image. In practice, the detailed shape of the PSF may actually depend a bit on location for any number of reasons -- e.g. the focal plane of the camera was a bit tilted, or the optics are subject to some minor aberrations, or (in the case of HST/ACS) the entire camera sits off the optical axis.

However, somewhat unhelpfully, you control this choice by a parameter that
is set within "daopars" and NOT "pstselect" itself. This parameter is called "varorder". If you set varorder=0, you are adopting a constant psf. The options are
varorder = 0 --> psf is the same everywhere in the image
varorder = 1 --> psf is linear function of x and y
varorder = 2 --> psf is quadratic function of x and y
varorder = -1 --> psf is constant, AND only the analytic model for the psf is used, with no extra lookup table to match the wings of the stars (see the help file for daopars to read more).

Start with varorder=0 and see how well it works. Try a modest number of candidates such as "maxnpsf=25" and see what you get. The routine will find 25 of what it thinks are good bright, isolated stars. The output file generated has the default name "pic.fits.pst.1" which lists the ones chosen.

What if the psf does vary across the image? In that case, you need a larger number of candidate stars to get a good solution for the shape changes. If a linear variation is good enough (varorder=1), select 50 - 70 candidate stars. If you need a quadratic variation (varorder=2), select 100 or even 200 candidates. The more the better, as long as they are not so faint that they are just adding noise.

(5) Build the PSF.

Now you need to take the candidate psf stars and use them to construct the complete model psf. THIS IS THE MOST IMPORTANT STEP IN THE WHOLE CHAIN. The quality of the final photometry depends very much on how good the psf model is.

The function psf is the one which actually builds the PSF, starting from the list of candidates. This is the part that is heavily interactive (as it should be). Its input file should look like this:

```
da> lpar psf
        image = "pic.fits"        Input image(s) for which to build PSF
     photfile = "default"       Input photometry file(s) (default: image.mag.?)
      pstfile = "default"       Input psf star list(s) (default: image.pst?)
     psfimage = "default"       Output PSF image(s) (default: image.psf.?)
     opstfile = "default"       Output PSF star list(s) (default:image.pst.?)
    groupfile = "default"       Output PSF star group file(s) (default:
image.psg.?)
    (plotfile = "")             Output plot metacode file
    (datapars = "")             Data dependent parameters
     (daopars = "")             Psf fitting parameters
   (matchbyid = yes)            Match psf star list to photometry file(s) by id
number ?
 (interactive = yes)            Compute the psf interactively ?
     (mkstars = no)             Mark deleted and accepted psf stars ?
   (showplots = yes)            Show plots of PSF stars ?
    (plottype = "mesh")         Default plot type (mesh|contour|radial)
   (icommands = "")             Image cursor: [x y wcs] key [cmd]
   (gcommands = "")             Graphics cursor: [x y wcs] key [cmd]
```

When you start it up, a new graphics window will appear which gives a pseudo-3D plot of the first candidate psf star in the list. Inspect it. With the cursor in that graphics window, you can type
a (= "accept" this star)
d (= "delete" or reject it)
n,s,e,w (to change the orientation angle of your view of the star)

You can read more about keystroke commands in the "help psf" file, but these are the basic ones you need.

Go through the entire list of candidates. When it gets to the end, move the cursor to the ds9 image display window (THIS IS VERY IMPORTANT!) and type "q". Then move the cursor back to the iraf text window and type "w". Then wait, while the routine grinds through a fit of all 6 analytical psf models and decides which one is the best. When it's finished, again move the cursor back to the image display window (THIS IS IMPORTANT!) and type "q" again. Finally, move the cursor back to the iraf text window and hit "q" and you are done.

Keeping the cursor in the correct place at each step is crucial. If you accidentally hit "q" or "w" or who-knows-what in the wrong window at the wrong time, the whole routine will hang up, you will have to exit iraf, kill various windows, etc., then restart a lot of stuff (in extreme cases, reboot the computer) accompanied by much swearing and emotional pain.

You should be quite careful to select only the best candidates. With a bit of experience you will know what a genuine star is as opposed to some faint galaxy, or a cosmic-ray spike, or something that is too crowded by neighbors. You can go back and forth to the image display window and see each candidate where it is on the image, to help you decide. You may want to iterate 2 or 3 times through "psf", discarding a few more of the candidates each time.

NB: If you already know pretty well WHICH analytical model to pick for the PSF shape, you can save a bit of time by going to the "daopars" parameter file and editing the first line "functio". The 7 choices you can enter are:

```
gauss  <-- generally works well for ground-based oversampled images
moffat15   <-- ditto
moffat25  <-- generally works OK for HST-type images
lorentz   <-- also works OK for HST-type images
penny1    <-- usually not much different from the moffat functions
penny2    <-- ditto
auto  <-- this one goes through all 6 of the functions and picks the best.
```

The output from psf is a new, tiny image called pic.fits.psf.1.fits. Display it. You will see a rather ugly-looking thing like a distorted doughnut. This is because it is just the "extra" non-analytical part of the PSF, that is, the wings that were not part of the Gaussian function, or Moffat function, or whatever the analytical part of the psf was.
To see the full psf (analytical+residuals), use the auxiliary function seepsf whose parameter file looks like this:

```
da> lpar seepsf
     psfimage = "pic.fits.psf.1.fits" PSF image name
        image = "picpsf.fits"    Output image name
   (dimension = INDEF)          Dimension of the output PSF image
        (xpsf = INDEF)          X distance from the PSF star
        (ypsf = INDEF)          Y distance from the PSF star
   (magnitude = INDEF)          Magnitude of the PSF star
        (mode = "ql")
```

Run this function and then display the output image picpsf.fits. Now you see the whole PSF in all its glory, and the result of your patient work.

(6) Measure all the stars on the image.

The hard part is over. The last step, and the big payoff, is to run ALLSTAR, which has a simple input file:

```
da> lpar allstar
        image = "pic.fits"      Image corresponding to photometry
     photfile = "default"       Input photometry file (default: image.mag.?)
     psfimage = "default"       PSF image (default: image.psf.?)
  allstarfile = "default"       Output photometry file (default: image.als.?)
      rejfile = "default"       Output rejections file (default: image.arj.?)
     subimage = "default"       Subtracted image (default: image.sub.?)
    (datapars = "")             Data dependent parameters
     (daopars = "")             Psf fitting parameters
       (wcsin = )_.wcsin)       The input coordinate system
(logical,tv,physical,world)
      (wcsout = )_.wcsout)      The output coordinate system
(logical,tv,physical)
      (wcspsf = )_.wcspsf)      The psf coordinate system (logical,tv,physical)
       (cache = yes)            Cache the data in memory?
      (verify = )_.verify)      Verify critical allstar parameters?
      (update = )_.update)      Update critical allstar parameters?
     (verbose = )_.verbose)     Print allstar messages?
     (version = 2)              Version
        (mode = "ql")
```

Start allstar running and wait (if you have a starlist of tens of thousands of stars, go have coffee, or do something else in parallel). It will get busy fitting the PSF simultaneously to all the stars in the *.mag photometry file (actually, it divides them up into connected subgroups on the image and then fits the subgroups, but the point is that this is a highly nonlinear and CPU-intensive step).
The output file pic.fits.als.1 will be your final list of "instrumental magnitudes" (magnitudes of all the stars on a relative scale).

There is a second important output file pic.fits.sub.1.fits which is the so-called "subtracted picture". Display it, along with the original image, and blink back and forth between them. The subtracted image shows the picture with all the PSF-fitted stars removed. Look at this image very carefully. Are the faint stars cleanly removed with no residual traces? (The very brightest stars may show noisy junk around their locations, which is normal.) What about opposite corners of the image --- are the stars equally well removed in all parts of the frame? If not, you may need to use a position-dependent PSF. If something looks consistently abnormal, like little holes or doughnuts everywhere there was a star, then something is wrong with the PSF. Try again .....

To make some initial tests that will prevent you from wasting a lot of time, you can clip out some small section of the original image and run allstar just on that section. You will get quick results and a subtracted image to test things out with.

(7) Calibrating your PSF magnitudes

The *.als file gives RELATIVE magnitudes; it has no zeropoint that means anything much. Now you have to convert them to instrumental magnitudes like those in the aperture photometry file.

To do this, you essentially need to know the mean difference ("offset") between m(allstar) and m(phot). That is, correlate the output from ALLSTAR and from PHOT star-by-star (write your own routine for cross-identifying them) and plot up [m(PH)-m(ALS)] versus m(PH). Find the mean difference, by concentrating most on the bright stars. Reject outliers and ignore the big scatter at the faint end. Add this mean difference to all the .als magnitudes,

and you have now converted them to the aperture-photometry scale.

But now notice that the aperture photometry was done with a SMALL aperture, not the large "fiducial" one that contains all of the light and that you decided on from the standard stars (see Part I step 4 above). So the last thing you need to know is the magnitude difference [m(fiducial ap) - m(small ap)]. Use the curve-of-growth technique to determine this. That is, for a few bright isolated stars, do multiple-aperture photometry and find the magnitude difference directly on that image. Then, add that mean difference to all the aperture photometry. At last, you now have a full list of instrumental magnitudes of all the stars on the image, on the right large-aperture magnitude scale.

Translating the instrumental magnitudes into their magnitudes on the standard system (UBVRI or ugriz) is a separate topic and not discussed here.

(8) Additional routines: SUBSTAR

You can manually subtract out any individual stars you want from the image (or any subgroup of stars selected from the whole .als list). The function substar is what does this. It can be very helpful for certain purposes. For example, suppose you want to subtract out all the faint stars from the image and leave behind only the few brightest ones. These bright ones will then be isolated and cleaned up from all their contaminating neighbors, and you can use them better for constructing curve-of-growth (see step 7 above). To do that, just sort the .als file by magnitude using the PSORT function; delete all the bright stars from the sorted list; and then use the remaining list of faint stars as the input to SUBSTAR.

(9) Additional routines: ADDSTAR

You can not only "subtract" stars from the image but also "add" some according to whatever prescription you want. The function addstar puts scaled PSFs into the image at whatever location and brightness you specify. These are "fake" stars -- they were not really there on the original image -- but they look just like real stars because the PSF was built from real stars. ADDSTAR gives you the ability to do all kinds of very powerful tests of the photometry, including crucially important things such as

(a) what fraction of the artificial stars you recovered, as a function of input magnitude

(b) the random errors of the photometry (that is, the rms difference between the input magnitudes of the fake stars and their measured magnitudes)

(c) any systematic bias in the photometry (that is, any net trend in the mean difference between input magnitude and measured magnitude).

Notice that "allstar" also estimates the random error of the measured magnitude of each star. "addstar" gives you another, and reasonably independent, way to do that. Usually you will find that the addstar route gives a slightly larger uncertainty at any given magnitude than does allstar. This is because the measurement errors from allstar include mostly just the Poisson statistics in the local sky intensity and star brightness. By contrast, the artificial-star experiments also implicitly include more effects -- the global differences in background and crowding, for example.

The detection completeness f is normally defined as

f = (number of stars detected) / (number added in)

This is a strong function of the magnitude m(in). At very bright levels, you see everything; at very faint levels, you see nothing; and there is a transition region where you see some fraction of what is really there. You will need to determine f by binning your artificial stars into fairly narrow magnitude bins and finding out what fraction of them were successfully found and measured in that bin. Alternately, you can add in a bunch of fake stars that all have the SAME input magnitude, and get a well defined f-value at that magnitude.

How many fake stars should you add in? This depends on the degree of object crowding in the original frame. If it's really uncrowded in an absolute sense (i.e. almost all the stars are isolated) then you can add lots more and not affect the actual degree of crowding significantly. I've found that for really uncrowded fields you can easily add in as many fake stars as original ones (doubling the population). At the opposite extreme, if the frame is really crowded then you can't add in large numbers of fake stars without changing the degree of crowding significantly. A "rule of thumb" in the literature is that then you should add in only about 5 percent of the numbers that were already there.

Developing a full understanding of the photometric completeness curve requires doing a lot of artificial-star runs -- that is, generating several different fake images and measuring them all. Work out some scripts to do this and save your own tedium ... at least, one thing you do NOT have to do is to regenerate the psf itself for each artificial-star frame. You can just use the "true" psf that you generated from the original frame. So to measure each fake frame, you just have to go through the sequence daofind / phot / allstar.

Suppose you've done all that. Now if you plot up f versus m(in), you will find that f drops rather rapidly from nearly 100% to nearly 0% over a span of about 1.0 - 1.5 magnitudes. The point at which f = 0.5 (50% detection rate) is defined as the "completeness limit" or "limiting magnitude" of the image.

This ends the basic run-through of daophot and stellar photometry. From here onward, it's a matter of gaining experience and picking up other extra features of the system as you go.