Specifications for the Aperture Photometry Package

Lindsey Davis

National Optical Astronomy Observatories*
Revised October 1987

ABSTRACT

The APPHOT package is a set of tasks for performing aperture photometry on uncrowded or moderately crowded fields, in either interactive or batch mode. The photometric technique employed is fractional pixel aperture integration. Point spread function fitting techniques are not used and no knowledge of the point spread function is required for the computation of magnitudes. Separate tasks are provided for creating and modifying object lists, computing accurate centers for a list of objects, computing sky values for a list of objects and performing aperture photometry inside circular or polygonal apertures.

July 8, 1990

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

Specifications for the Aperture Photometry Package

Lindsey Davis

National Optical Astronomy Observatories*
Revised October 1987

1. Introduction

The APPHOT package is a set of tasks for performing aperture photometry on uncrowded or moderately crowded fields. The photometric technique employed is fractional pixel integration. Point spread function techniques are not used and no knowledge of the point spread function is required for the computation of magnitudes.

The APPHOT package performs multi-aperture photometry on digitized starfields maintained as IRAF image files. Input to the package consists of an image file, a list of object coordinates, numerous parameters controlling the analysis algorithms and, optionally, the graphics terminal and/or display. APPHOT output consists of successive lines of text where each line summarizes the results of the analysis for a particular object.

Given starting coordinates and an IRAF image, the principal APPHOT task computes accurate centers, sky values and magnitudes for a list of objects. Separate IRAF callable tasks in APPHOT exist to create and modify object lists, to determine image characteristics such as the full width half maximum of the point spread function or standard deviation of the sky pixels, to compute accurate centers for a list of objects, to compute accurate local sky values for a list of objects, and to compute magnitudes inside a polygonal aperture.

2. APPHOT Package Requirements

2.1. APPHOT Package Input

2.1.1. The IRAF Image

APPHOT assumes that images exist on disk in IRAF format. Facilities for reading and writing images exist elsewhere in IRAF such as in package DATAIO or MTLOCAL.

APPHOT assumes that the image data is linear. The input image is assumed to have been corrected for those instrumental defects which affect the intensity of a pixel. These include pixel to pixel variations in the bias values, pixel to pixel gain variations, cosmic rays, cosmetic defects, geometric distortion, and detector non-linearities.

APPHOT assumes that the local sky background is approximately flat in the vicinity of the object being measured. This assumption is equivalent to assuming that the local sky region has a unique mode. Therefore any variations in the sky background which occur at the same scale as the sky region should be removed prior to entering APPHOT.

The point spread function of the image is assumed to be constant for all regions of the image. This is particularly critical in the case of faint objects for which small apertures which minimize the effects of crowding and sky noise in the aperture are used. The wings of the object will almost certainly extend beyond the aperture and good results will only be obtained if

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

objects are consistently well centered and the shape and diameter of an object is constant throughout the image and invariant to magnitude.

2.1.2. The Coordinate Lists

All APPHOT tasks operate on lists of object coordinates or interactive cursor input. Lists are maintained as text files, one object per line with the x and y coordinates in columns one and two respectively. List files may be created interactively with either the graphics or image cursor, by a previously executed APPHOT task, by a previously executed IRAF task or by a user task.

The x and y coordinates contained in the list file can be either the starting centers for the objects of interest or the true centers. In either case the aperture centered around the object position must not extend beyond the boundary of the image. To obtain reliable results from the centering algorithm the starting centers should not be more than 1 fwhmpsf pixels from the true centers.

2.1.3. Algorithm Parameters

Many tasks in the APPHOT package have an associated set of algorithm parameters which control the operation of the analysis routines. For example the centering and sky fitting algorithms each have an associated group of parameters. The APPHOT tasks supply reasonable defaults for these parameters. However the user may alter them at any time in interactive mode or by editing the appropriate parameter file before running the task in batch.

2.1.4. Terminal Graphics and the Image Display

Most APPHOT programs may be run in either interactive or batch mode. In interactive mode, the user can mark the positions of objects by interactively, positioning the image cursor on the display device. Simple cursor commands can interactively alter the algorithm parameters. In batch mode the algorithm parameters are fixed and object coordinates are taken from the user supplied coordinate list. The display device is not required in batch mode.

At present there is no high level IRAF display interface. Therefore the present version of APPHOT replaces the display device with terminal graphics. For example it is possible to load a contour plot of an image or image section, and run the APPHOT tasks, interactively marking positions on the plot. This is a temporary measure to tide thing over until the display interface is available. Furthermore this option is only really suitable for those terminal which have both text and graphics windows displayed at the same time.

Certain APPHOT tasks such as FITSKY occasionally require an interactive graphics capability. For example it is sometimes desirable to plot the histogram of the sky pixels and mark the peak of the histogram interactively rather than using the automatic sky fitting routines. Graphics capablity has been added to APPHOT tasks as deemed useful.

2.2. APPHOT Package Functions

2.2.1. Creating Coordinate Lists

APPHOT task(s) shall exist to create coordinate lists interactively within APPHOT by loading the IRAF image into the image display and successively marking the objects to be measured with the image cursor. It shall be possible to mark the positions of objects on the display, and draw circles of arbitrary size around the objects of interest. It shall be possible to verify existing coordinate lists by marking the object coordinates on the image display.

At present the full image display cababilities do not exist in IRAF. Therefore as a temporary measure most tasks will run with a contour plot and the graphics cursor as a substitute

for the image display and image cursor by setting the image cursor to stdgraph and the display parameter for each task to stdgraph. The output coordinates will be written to a text file and may be used as starting coordinates for the APPHOT routines.

Those sites which have SUN workstations can use the IMTOOL facilites to create cursor lists.

APPHOT tasks shall exist to automatically detect objects by specifying the IRAF image to be searched, plus a few image characteristics such as a detection threshold and the full width half maximum of the point spread function. The output coordinates plus a few object statistics will be written to a text file.

2.2.2. Coordinate List Operations

General list processing tasks are or will be available in the IRAF lists package. Examples of useful currently existing tasks are list sorting by for example magnitude and performing a linear transformations on coordinate lists.

It may eventually be necessary to add some specialized list processing tasks to APPHOT. One example is a list matching facility in which objects in common to two lists are combined and output to a third list.

2.2.3. Determining the Image Characteristics

APPHOT tasks shall exist to estimate certain image characteristics such as the full width half maximum of the image point spread function and the standard deviation of the background. In order to obtain meaningful error estimates it is also necessary to specify the noise charactersitics of the detector and a noise model. In this version of APPHOT two noise functions will be supported a constant sky background noise, and constant background noise plus Poisson statistics in the object.

The reason for this capability is that the majority of the APPHOT algorithm parameters scale with these two image characteristics. For example all the pixel scale dependent parameters such as the centering aperture, object apertures, the two sky annuli and the maximum shift parameter all scale with the full width half maximum of the point spread function. Similarly most of the pixel intensity dependent parameters scale with the standard deviation of the sky pixels.

2.2.4. Centering

An APPHOT task shall exist to determine accurate centers for a list of objects, using the approximate object coordinates as a starting point. The centering aperture may extend beyond the boundaries of the image but a warning message will be generated.

A choice of centering algorithms with a range of efficiency and accuracy will be available. The center determination must be resistant to the affects of nearby contaminating objects.

The user may opt to use the starting center as the actual center in all cases, to use the starting center as the actual center if the object is very faint, or tweak up the center if the signal to noise is above a certain threshold.

The output of the centering algorithm will the set of computed coordinates and their errors.

2.2.5. Fitting the Sky

An APPHOT task shall exist to compute a constant background value by analysis of an annular region surrounding the object. The background is assumed to be flat in the region of the object, but may contain contaminating objects or defects which shall be detected and eliminated by the fitting algorithm. It shall be permissible for the background region to extend beyond the boundaries of the image; the out of bounds region of the background shall be excluded from the fit.

The user may supply a constant background, with zero-valued noise and skew value or read appropriate values from a file instead of computing the background.

The output of the sky fitting algorithm shall be the mode, standard deviation and skew of the sky as well as the number of pixels left in the background region after pixel rejection and the number rejected.

2.2.6. Multi-aperture Photometry

APPHOT routines shall exist to compute the integral of object minus background within one or more circular apertures centered upon the object. The integration shall be performed using partial pixel techniques, to minimize the effects of sampling. If the aperture contains any indefinite pixels, or if the aperture extends beyond the boundary of the image, an indefinite result shall be returned. Both normal and weighted photometry routines shall be available.

It shall be possible to normalize the output magnitudes to a user specified integration time using values stored in the IRAF image header.

The output shall consist of magnitudes and errors for each of the specified apertures.

2.2.7. Polygonal Aperture Photometry

Determine the integral of the object within a specified polygon. Aperture integration shall be by fractional pixel summation of successive image lines.

3. Specifications

3.1. Apphot CL Callable Tasks

The CL callable part of the APPHOT package consists of the following routines:

```
    mark    -- create/verify coordinate lists interactively
polymark    -- create/verify polygon lists interactively
 daofind    -- automatic image finder from DAOPHOT
 lintran    -- linearly transform a list of coordinates (LISTS package)

 radprof    -- compute the radial profile of an object.
  fitpsf    -- model the PSF

  center    -- compute accurate centers for a list of objects
  fitsky    -- compute accurate sky values for a list of objects
    phot    -- perform multi-aperture photometry on a set of objects
polyphot    -- compute the flux inside a set of irregular polygons
   wphot -- perform weighted multi-aperture photometry
```

3.2. Standard Analysis Procedures

A standard reduction procedure would be something as follows.

1. Estimate the image and data characteristics in the following manner.
    1. Use the interactive setup option (i key) in the PHOT task to specify the fwhm of the psf, the centering apperture, the sky annuli, the photometry apertures as well as estimate the sky sigma. Examine the results (:show command) and optionally store the new parameters in the parameter files (w key).
    2. Use the EPAR task to edit the parameter files directly. For example the detector gain and readout noise must be known before running PHOT in order to get reliable error estimates.
2. Prepare a coodinate list in one of the following ways.
    1. Running MARK or POLYMARK.
    2. Run DAOFIND or some other automatic image finder.
    3. Run another APPHOT task such as CENTER .
    4. Run any other IRAF or user program which generates the appropriate format.
    5. Transform an existing list using the LISTS package facilities
3. Run PHOT or WHOT in non-interactive mode to perform the multi-aperture photometry. The user should be familiar with the algorithms used to measure the object center, to fit the background, and compute the aperture integral, magnitude, and errors. The values of all visible and hidden PHOT parameters should be inspected before doing any serious processing. Note that CENTER and FITSKY can be run independently of PHOT and their output used as input to PHOT.

3.3. The APPHOT Algorithms

The principal APPHOT algorithms are described below.

3.3.1. The RADPROF Algorithm

The function of the RADPROF task is to compute the radial profile of selected, isolated, high signal to noise objects in an IRAF image.

Given the IRAF image, an initial center, an aperture and a step size, RADPROF computes the radial profile of an object in the following manner.

1. The APPHOT centering routines are used to compute an accurate center for the object. The new center is used to compute the radial coordinate for each pixel in the subraster.
2. The APPHOT sky fitting routines are used to compute an accurate sky value for the object. The new sky value is used to compute the object intensity for each pixel in the subraster.
3. The CURFIT package routines are used to produce a smooth intensity versus radius curve by fitting a cubic least squares spline function to the data using linear least squares techniques.
4. The fitted curve is evaluated at equally spaced intervals to produce the final radial profile. The profile is normalised using the r = 0 value.
5. The smoothed curve is integrated using the 1D integration routines IMINTERP to evalute the fraction of the total integral at each interval.
6. The three quantities I_r the raw intensity, $Inorm_r$ the normalized intensity, and $Ifraction_r$ the fraction of the total intensity inside radius at r, are plotted on the terminal and output as a function of r to a text file.

RADPROF is used principally to compute the stellar image and setup the parameters for doing photometry. Its default output is a radial profile plot with the phot parameters marked on it.

3.3.2. The FITPSF Algorithm

The function of the FITPSF task is to fit an analytic model to an object in order to derive information about the point spread function. Given an IRAF image, an initial center and an aperture, FITPSF computes the model parameters in the following manner.

1. The fitting function is chosen. The present version of FITPSF offers two function choices a 2D radial Gaussian function and an elliptical Gaussian function. A third option, moment analysis, has been added to the FITPSF package. The image characteristics are evaluted by using the 0th, 1st and 2nd order moments of the image.
2. Initial guesses for the parameters are made from the subraster data.
3. The parameters and their errors are derived using standard non-linear least squares techniques and the NLFIT package.

3.3.3. The DAOFIND Algorithm

The function of the DAOFIND task is to automatically detect objects in an IRAF image above a certain intensity threshold. The user specifies an intensiyt threshold, the estimated full width half maximum of the point spread function, and limits on the sharpness and roundness of the objects to be detected. DAOFIND produces a list of x and y coordinates and a sharpness and roundness statistic for each detected object.

The DAOFIND algorithm works in the following way.

1. The user specifies an intensity threshold above local sky and estimates the full width half maximum of the point spread function.
2. DAOFIND computes a convolution kernel which is a linear function of the brightness values in an approximately circular array of pixels. The original data is convolved with the computed kernel. The equivalent mathematical operation is a convolution of the original data with truncated, lowered circular Gaussian function with the specified FWHM, computed in such a way as to be the mathematical equivalent of fitting a Gaussian stellar profile to the object data by least squares. The coefficients of the linear function sum to zero so that the overall bias level (local sky) cancels out exactly. Since the function is symmetric about a single pixel, a smooth gradient in the sky brightness also cancels exactly. Therefore the user does not have to specify an absolute brightness threshold but only a threshold above local sky.
3. The convolved data is searched for local maxima. Local maxima which are less than the specified threshold are rejected. A pixel is defined to be a local maximum when it is the brightest pixel within $n * \sigma_{Gauss}$.
4. Using the original data, DAOFIND computes an object sharpness statistic which can be used to reject hot or cold pixels. The sharp parameter is defined below. Detected objects with a sharpness parameter outside the range specifed by sharplo and sharphi are rejected. Typical values for sharplo and sharphi are 0.2 and 1.0 respectively.

   I_δ = height of best fitting δ function = pixel value

   I_{Gauss} = height of best fitting Gaussian function = convolved value

   $$\text{sharplo} <= \text{sharp} = \frac{I_\delta}{I_{Gauss}} <= \text{sharphi}$$

If the brightness enhancement detected in the convolved data is due to a single bright pixel, then the best fitting delta function will have an amplitude equal to the height of this pixel above the local background, while the amplitude of the best fitting Gaussian will be pulled down by the surrounding low valued pixels. Sharp will be a number significantly greater than one. A single cold pixel will produce brightness enhancements approximately 0.5 * FWHM away from the pixel in question in all directions. In this case the height of the delta function which best fits these spurious maxima is close to zero, while the height of the best fitting Gaussian is some small positive number. In this case sharp will be close to zero.

5. The roundess statistic is computed from the original picture data by fitting 1D Gaussians to the marginal sums in x and y. If the height of either 1D Gaussian is negative the object is rejected. If both heights are positive the roundness parameter is computed.

$$\Delta I = I_{x\ Gauss} - I_{y\ Gauss}$$

$$<I> = I_{x\ Gauss} + I_{y\ Gauss}$$

$$roundlo \leq round = \frac{\Delta I}{<I>} \leq roundhi$$

Objects which are elongated in the x direction have roundness values less than zero and those elongated in the y direction have roundness greater than zero. Round discriminates only against objects which are elongated along either rows or columns. Objects elongated at a 45 degree angle will not be rejected.

6. Finally the approximate x and y centroids of the detected objects, rough magnitudes relative to threshold are computed, and object is added to the output file.

3.3.4. The CENTER Task

3.3.4.1. Centering Package Routines

The following are the principal programmer callable routines routines in the centering package.

```
                apcinit (ap, cfunction, cbox, fwhmpsf, noise)
          ier = apfitcenter (ap, im, xinit, yinit)
          ier = aprefitcenter (ap)
        value = apstat[ir] (ap, param)
                apstats (ap, param, str, maxch)
                apset[sir] (ap, param, value)
                apcfree (ap)
```

The following quantities can be examined or set by apstat/apset calls

```
  data          1. fwhmpsf       full width half maximum of psf in pixels
parameters      2. threshold     minimum intensity for centering
                3. noise         noise model
                4. sigma         standard deviation of background
                5. readnoise     readout noise of CCD in electrons
```

```
                6. epadu         electrons per adu

centering       1. calgorithm    centering algorithm
parameters      2. positive      emission / absorption features
                3. cbox          centering radius in fwhmpsf
                4. cmaxiter      maximum number of fitting iterations
                5. maxshift      maximum permitted shift in fwhmpsf
                6. minsnratio    minimum permitted signal to noise ratio
                7. clean         clean subraster before centering
                8. rclean        cleaning radius in fwhmpsf
                9. rclip         clipping radius in fwhmpsf
                10.kclean        k-sigma rejection for cleaning in sigma
```

The following computed quantities can be examined by apstat calls.

```
1. xcenter          computed x coordinate
2. ycenter          computed y coordinate
3. xerr             computed x coordinate error
4. yerr             computed y coordinate error
```

See the manual pages for CENTER and CENTERPARS for a detailed description of the centering parameters.

3.3.4.2. The Centering Algorithm

The function of the CENTER task is to compute accurate centers and errors for objects in an IRAF image given a list of initial positions and a centering aperture.

1. If the centering algorithm is disabled, the computed center is set to the initial center and the uncertainty estimate is set to zero.
2. If the cleaning algorithm is enabled, clean the subraster before centering.
3. Estimate is made of the signal to noise of the object. If this quantity is less than a certain minimum value the computed center is kept but a warning message is issued.
4. The center and errors are computed using one of several algorithms.
5. If the computed center is greater than a user specified distance from the initial center then the computed center is returned with an error message.

3.3.4.3. Symmetry Clean Algorithm

The symmetry-clean algorithm attempts to remove defects in the centering subraster by assuming that the object has radial symmetry and comparing pixels on the opposite sides of the center of symmetry. The algorithm works in the following way.

1. The center of symmetry is computed. Normally the center of symmetry is assumed to coincide with the position of the brightest pixel in the subarray. However if the maximum pixel is more than a user specified distance away from the intial center, the initial center is used as the center of symmetry.
2. Pixels inside a specified cleaning radius are unaltered.
3. Pairs of pixels diametrically opposed about the center in the cleaning region between the cleaning and clipping radii are tested for equality. If the difference between the pixels is greater than a specified k-sigma rejection limit, the larger value is replaced by the smaller. In this region sigma is computed from Poisson statistics.

4. Pairs of pixels in the clippping region are compared in the same manner as those in the cleaning region except that sigma is the standard deviation of the sky pixels.

The effect of the symmetry-clean algorithm is to edit the raster, removing any contaminating objects in the vicinity of the primary object. This simplifies the fitting algorithm and increases its reliability, since it does not have to deal with multipeak marginal distributions.

3.3.4.4. Signal to Noise Estimate

The signal to noise of the object is estimated from the data values in the subraster in the following way.

$$SNR = \frac{N_{object}}{\sqrt{n_{pix} * \sigma_{sky}^2}}$$

or

$$SNR = \frac{N_{object}}{\sqrt{N_{object} + n_{pix} * \sigma_{sky}^2}}$$

where N_* is the number of counts in the object above threshold, σ_{sky} is the standard deviation of the pixels in the sky region and n_{pix} is the number of pixels in the object aperture. The first approximation corresponds to constant sky noise only, the second includes Poisson noise in the object.

3.3.4.5. Centroid

The centroid centering algorithm is similar to that used in MPC and can be briefly described as follows. For more detailed description see (Stellar Magnitudes From Digital Pictures, 1980, Adams et al).

1. The marginal distributions in x and y are accumulated.
2. The intensity weighted centroid positions for the marginals is computed using only data pixels which are above the threshold intensity. If the threshold parameter is 0 the mean intensity of the marginal is used in place of the threshold

$$I_i = I_i - threshold \qquad I_i > 0.0$$

$$x_c = \frac{\sum I_i \, x_i}{\sum I_i}$$

3. The errors are estimated in the following way

$$\sigma^2 = \frac{\sum I_i \, x_i^2}{\sum I_i} - x_c^2$$

$$err_{xc} = \sqrt{\sigma^2 \; / \; \sum I_i}$$

3.3.4.6. Gaussian Fit to the Marginals

The fit is performed in the following way.

1. The marginal distributions in x and y are accumulated.
2. Initial guesses for the parameters of the 1D Gaussians I_{Gauss}, x_c and I_{sky} are derived from the marginal distributions themselves. The width σ is held constant.
3. The best fit parameters and their errors are derived using non-linear least-squares techniques and the NLFIT package.

3.3.4.7. Optimal Filtering of Marginals

The fit is performed is the following way.

1. The marginal distributions in x and y are accumulated.
2. The centroid of the observed distribution is computed by solving the following equation.

$$\Psi(x_c) = \sum \omega_i(x_c)\, \Phi_i = 0$$

$$\omega_i = \frac{\partial \phi_i \,/\, \partial x_c}{\phi_i + b}$$

The assumptions are that the observed distribution Φ_i is correctly modeled by the profile function ϕ_i. The usual choise of profile model for centering is a Gaussian. Howver in the interests of speed a triangle function has been substituted. This causes at most a 7% increase in the random centering error.

3. The startup procedure is based on the following fact.

$$\Psi(x_n) > 0 \quad \text{for} \quad x_n < x_c$$

$$\Psi(x_n) < 0 \quad \text{for} \quad x_n > x_c$$

The iteration is initialized by assuming that $x_1 = x_c$ and computing $\Psi(x_1)$. The initial x value is incremented by $\pm\sigma_{Gauss}$ depending on the sign of Ψ. The search is repeated until $\Psi(x_{n-1})$ and $\Psi(x_n)$ have opposite signs. At this point the true center is bracketed by the estimated positions (x_n, x_{n-1}) and we have a table of at least 2 values of $\Psi(x_n)$.

4. The computation proceeds by interpolating in the table of values for the estimated position x_{n+1} where $\Psi = 0$. If the table contains only two values as may be the case for the inititial interpolation, linear interpolation is used. In all other cases a quadratic fit to the three most recent Ψ values is used. The computation is complete when two successive estimates differ by less than some tolerance typically 0.001 pixel.
5. The errors are estimated as follows.

$$\sigma^2 = \left(\int \frac{(\partial \phi \,/\, \partial x_c)^2}{\phi + b}\right)^{-1}$$

$$err_{xc} = \sqrt{\sigma^2}$$

3.3.4.8. Other Centering Methods

The code is constructed in such a way that new algorithms may be easily added at a future date, such as the more sophisticated techniques required for accurate astrometry.

3.3.5. The FITSKY Task

3.3.5.1. Sky Fitting Routines

The following are the main entry points in the sky fitting package.

```
        apsinit (ap, function, annulus, dannulus, fwhmpsf, noise)
   ier = apfitsky (ap, im, xpos, ypos, sd, gd)
   ier = aprefitsky (ap, sd, gd)
 value = apstat[ir] (ap, param)
        apstats (ap, param, str, maxch)
        apset[irs] (ap, param, value)
        apsfree (ap)
```

The following quantities can be examined or set with apset/apstat calls.

data parameters	1. fwhmpsf	full width half maximum of the psf
	2. sigma	standard deviation of sky pixels
sky fitting parameters	1. annulus	inner radius of sky annulus in fwhmpsf
	2. dannulus	outer radius of sky annulus in fwhmpsf
	3. sfunction	sky fitting algorithm
	4. smaxiter	maximum number of fitting iterations
	5. kreject	k-sigma rejection limit sigma
	6. nreject	maximum number of rejection cycles
	7. rgrow	region growing radius in fwhmpsf
	8. khist	half-wdth of histogram in sigma
	9. binsize	binsize of histogram in sigma
	10. smooth	Lucy smooth the histogram
	11. skyfile	name of text file containing sky values
	12. skyvaluser	supplied constant value for sky

The following computed quantities can only be examined with apstat calls.

1. skymode	computed sky value
2. skysig	computed sky sigma
3. skyskew	computed sky skew
4. nsky	number of sky pixels
5. nsky_reject	number of rejected sky pixels

3.3.5.2. The Sky Fitting Algorithms

A good background fit is essential to aperture photometry. Fitting the background is trivial in a sparse field, but difficult in a crowded field. In general the background region will contain contaminating objects which must be detected and excluded if a good fit is to be obtained.

The main algorithm used in APPHOT is the following.

1. If the skyfitting switch is disabled either read the sky values from a text file or accept a user supplied constant for the sky.
2. Perform the initial sky fit using one of the specified algorithms. The sky fitting algorithms fall into three general categories, those that use the actual sky pixel array itself, those that operate on a histogram of sky values and those that rely on user interaction.
3. If the pixel rejection flags are set perform pixels rejection with optional region growing.

3.3.5.3. Sky Pixel Array Techniques

3.3.5.3.1. Median

1. Sort the array of sky pixels. This is necessary to avoid quantization effects.
2. Compute the median, and the standard deviation and skew with respect to the mean.
3. If the k-sigma rejection limit is greater than zero and the maximum number of rejection cycles is greater than one, perform pixel rejection. Pixels greater than k-sigma from the median are rejected. Region growing is optional.
4. Stop the rejection cycle on any given iteration if the maximum number of rejection cycles ix exceeded, no more sky pixels are left or no more pixels are rejected.

3.3.5.3.2. Mode

1. Sort the array of sky pixels. This is necessary to avoid quantization effects.
2. Compute the mode, and the standard deviation and skew with respect to the mean.

$$I_{mode} = 3.0 * I_{median} - 2.0 * I_{mean}$$

3. If the k-sigma rejection limit is greater than zero and the maximum number of rejection cycles is greater than one, perform pixel rejection. Pixels greater than k-sigma from the mode are rejected. Region growing is optional.
4. Stop the rejection cycle on any given iteration if the maximum number of rejection cycles ix exceeded, no more sky pixels are left or no more pixels are rejected.

3.3.5.4. Histogram Techniques

The following three techniques all operate on the histogram of the sky pixels. The routines all construct the histogram in the following identical manner.

1. The mean of the sky distribution is computed.
2. If the user specified standard deviation of the sky pixels is INDEF the algorithm computes the standard deviation of the sky pixels with respect to the mean.

3. All pixels within plus or minus sigma standard deviations are accumulated into a histogram. The user specifies the bin size.
4. The histogram may optionally be Lucy smoothed before any operation is performed on it.

3.3.5.4.1. Centroid

The mode, sigma and skew of the sky pixels are computed in the following manner.

1. The histogram is compiled as above.
2. The mode, standard deviation and skew of the sky pixels are computed in the following manner.

$$I_0 = \sum I_i$$
$$I_1 = \sum I_i \, x_i$$
$$I_2 = \sum I_i \, x_i^2$$
$$I_3 = \sum I_i \, x_i^3$$

$$I_{mode} = I_1 \, / \, I_0$$
$$\sigma = (I_2 \, / \, I_0 - I_{mode}^2)^{1/2}$$
$$skew = (I_3 \, / \, I_0 - I_{mode} * \sigma^2 - I_{mode}^3)^{1/3}$$

3 If pixel rejection is enabled sky pixels within a user supplied limit of the mode are rejected with optional region growing.

3.3.5.4.2. Gaussian Fit

The mode, standard deviation and skew of the sky pixels are derived from a model fit in the following way.

1. The histogram of the sky pixels is compiled as above.
2. Initial guesses to the model parameters, $N_{\max}$, I_{mode}, σ, and *skew* are made from the histogram itself.
3. Final parameters and their errors are derived using non-linear least squares techniques and the NLFIT package.
4. If pixel rejection is enabled sky pixels within a user supplied limit of the computed mode are rejected with optional region growing.

3.3.5.4.3. Optimal Filtering

The method is as follows.

1. The histogram is compiled as above.
2. Using the mean of the sky pixels as the intital value of the sky mode, a new mode is computed using the optimal filtering technique described for centering.

4 If pixel rejection is enabled sky pixels within a user supplied limit of the computed mode are rejected with optional region growing.

3.3.5.4.4. Cross Correlation

The method is as follows.

1. The histogram is compiled as above.
2. The noise function is estimated using the standard deviation of the sky pixels and the cross-correlation function is computed.
3. The mode is computed using quadratic interpolation around the peak of the distribution.
4 If pixel rejection is enabled sky pixels within a user supplied limit of the mode are rejected with optional region growing.

3.3.5.5. Interactive Techniques

3.3.5.5.1. Histogram Plot

The histogram is compiled as described above and the user marks the peak on the histogram plot with the graphics cursor. The sigma and skew of the sky distribution with respect to the mean is also computed.

3.3.5.5.2. Radial Distribution

A radial profile plot of the sky region is plotted and the user marks the sky value on the plot with the graphics cursor. The sigma and skew of the sky distribution with respect to the mean is computed.

3.3.5.6. Pixel Rejection and Region Growing

All the sky fitting algorithms permit pixel rejection and optional region growing. Pixel rejection and region growing are performed by locating all pixels more than k * sigma from the mode, and blindly rejecting all pixels within a certain radius of each deviant pixel. This simple algorithm works well because the sample is large, and therefore there is little penalty for discarding pixels that might not be deviant. Region growing also tends to accelerate convergence significantly.

Very faint contaminating objects are difficult to detect and reject. If there are enough such objects, they should not be rejected, because there are probably a few in the object aperture as well. A higher sky sigma will be calculated and the computed uncertainty in the magnitude will increase. The best solution to this problem may be to increase the size of the annulus to minimize the bias.

3.3.5.7. The Principal PHOT Routines

The main entries in the photometry routine are the following.

```
            apinit (ap, cfunction, cbox, sfunction, annulus, dannulus,
             aperts, napert, fwhmpsf, noise)
      ier = apfitcenter (ap, im, xinit, yinit)
      ier = aprefitcenter (ap)
      ier = apfitsky (ap, im, xcenter, ycenter, sd, gd)
      ier = aprefitsky (ap, sd, gd)
      ier = apmag (ap, im, xcenter, ycenter, skyval, skysig, nsky)
      ier = apwmag (ap, im, xcenter, ycenter, positive, skyval, skysig,
             nsky)
      ier = apremag (ap, positive, skyval, skysig, nsky)
      ier = apwremag (ap, positive, skyval, skysig, nsky)
    value = apstat[ir] (ap, param)
```

```
apstats (ap, param, str, maxch)
apset[sir] (ap, param, value)
apfree (ap)
```

The following parameters can be examined or altered by apset/apstat calls.

```
1. weighting        weighting scheme for wphot
2. aperts           list of apertues
3. naperts          number of apertures
4. zmag             zero point of magnitude scale
5. itime            effective integration time
```

The following quantities can be examined with apstat calls.

```
1. sums            array of aperture sums
2. areas            array of areas
3. mags             array of magnitudes
4. magerrs          array of magnitude errors
```

3.3.5.8. The PHOT Aperture Integration Algorithm

The integral of the flux within a circular aperture is computed by fractional pixel techniques. Pixels are assumed to be square apertures arranged in a rectangular grid. The fraction of a pixel which lies within the circular APPHOT aperture is computed by an approximation, and all such contributions are summed to produce the total integral.

The inclusion of a partial pixel inside the aperture is done as follows.

1. If the distance of the current pixel from the center of the star, r, is exactly equal to the radius of the aperture R then one-half the counts in the pixel are included.
2. If r < R - 0.5 the entire pixel is included while if r > R + 0.5 the pixel is wholly excluded.
3. In between the fraction of the counts varies linearly. A circular aperture is approximated by an irregular polygon.

The simplicity of aperture photometry limits the amount of information available for error analysis. The following three sources of error are considered.

1. The error due to sky noise in the aperture.

$$error_1 = \sigma_{sky} * A_{apert}^{1/2}$$

2. The error in the aperture sum.

$$error_2 = (A_{sum} / phpadu)^{1/2}$$

3. The mean error of the sky.

$$error_3 = \sigma_{sky} * A_{apert} / nsky^{1/2}$$

where σ_{sky} is either computed by the background fitting algorithm or set by the user, and

A_{apert} is the fractional pixel area of the aperture.

3.3.5.9. The WPHOT Algorithm

The WPHOT algorithm computes a weighted aperture sum in an attempt to minimize noise in the sky. The algorithm is the following where w is the weight for each pixel, p is the noise free profile value and σ is the noise per pixel from all sources. (See the paper by Stover and Allen 1987 for details)

$$A_{\sum} = \sum w_i * (I_i - sky)$$

$$w_i = C * p_i / \sigma^{2_i}$$

$$C = \sum p_j / \sum p_j * w_j$$

3.3.5.10. The POLYPHOT ROUTINES

The principal polyphot routines are the following.

```
        apyinit (ap, sfunction, annulus, dannulus, noise)
  ier = apfitcenter (ap, im, wx, wy)
  ier = aprefitcenter (ap)
  ier = apfitsky (ap, im, xcenter, ycenter, sd, gd)
  ier = aprefitsky (ap, sd, gd)
  ier = polyfit (ap, im, xver, yver, nver)
value = apstat[ir] (ap, param)
        apstats (ap, param, str, maxch)
        apset[sir] (ap, param, value)
        apfree (ap)
```

3.3.5.11. The POLYPHOT Algorithm

The function of the POLYPHOT task is to compute the flux inside an irregular polygon given a list of the coordinates of the vertices of a polygon. The polygon must be entirely inside the image and the vertices of the polygon must be specified in clockwise or counterclockwise order. The actual algorithm used is as follows.

1. The range of image lines which intersect the polygon are computed.
2. For each image line in the specified range the intersection points with the polygon are computed.
3. The flux between pairs of limits is summed using a fractional pixel approximation for the endpoints.
4. The sky is fitted using any of the methods previously discussed and a user specified annular region.
5. The errors are computed as specified in the PHOT specifications.

4. Example

A brief example may help illustrate the use of the package. Suppose we want to process a few hundred stars on image "blue".

The first step is to prepare a list of objects to be measured. The simplest way to do this is to interactively mark the objects with the image cursor using the display (graphics) device and the RIMCURSOR (RGCURSOR) task.

```
... load image on the display ...

ap> rimcursor > starlist

    ... move cursor and mark stars ...


... load contour plot on graphics terminal ...

ap> rgcursor > starlist

    ... move cursor and mark stars ...
```

Alternatively one can run DAOFIND to compute a list of candidate objects in the frame. The name of the coordinate file is stored in the PHOT parameter set.

```
ap> phot.coords=starlist
```

The next step is to set up the PHOT parameters interactively. First we load the image (contour plot) blue on the display (graphics terminal). Next we call up PHOT in interactive mode.

```
ap> phot blue
... cursor appears ...
```

PHOT takes input by reading the image (graphics) display (terminal) cursor. In order to display the available commands we tap the ? key and the following text appears on the screen.

```
    Interactive Phot Commands

?   Print options
:   Colon command see below
i   Setup PHOT parameters interactively
w   Write PHOT parameters to the parameter files
l   Process the remainder of the coordinate list
r   Rewind the coordinate list
c   Fit center around the current cursor position
t   Fit sky around the current cursor position
```

```
s   Fit sky around the current center position
p   Compute magnitudes around the cursor position
f   Fit center, sky and compute magnitudes
sp  Fit center, sky, compute magnitudes, and save
q   Exit program

Phot parameters are listed or set with the following commands.

:m [n]  Move cursor to the [nth] object in the coordinate list
:n [n]  Measure the [nth] object in the coordinate list

:show [center/sky/phot/all]    List the aphot parameters
:fwhmpsf [value]               Full width half maximum of the PSF
:noise [string]                Noise model
:threshold [value]             Threshold value for centering
:sigma [value]                 Standard deviation of the background
:ccdread                       CCD readout noise keyword
:readnoise                     Readout noise in electrons
:gain                          Gain keyword
:epadu                         Electrons per adu

:calgorithm [string]           Centering function
:positive [y/n]                Emission or absorption feature
:cbox [value]                  Width of the centering box in fwhmpsf
:cmaxiter [value]              Maximum number of centering iterations
:maxshift [value]              Maximum shift in fwhmpf
:minsnratio [value]            Minimum signal to noise ratio of pixels
:clean [y/n]                   Clean subraster before centering
:rclip [value]                 Clipping radius in fwhmpsf
:rclean [value]                Cleaning radius in fwhmpsf
:kclean [value]                Sigma for clean algorithm
:mkcenter [y/n]                Mark the centers on the display

:salgorithm [string]           Sky fitting algorithm
:annulus [value]               Inner radius of sky annulus in fwhmpsf
:dannulus [value]              Width of sky annulus in fwhmpsf
:skyvalue [value]              User supplied sky
:smaxiter [value]              Maximum number of rejection cycles
:skreject [value]              +/- Pixel rejection limits in sky sigma
:snreject [value]              Maximum number of rejection interations
:khist [value]                 +/- Sky histogram size in sky sigma
:binsize [value]               Resolution of sky histogram in sky sigma
:smooth [y/n]                  Lucy smooth the sky histogram
:rgrow [value]                 Region growing radius in fwhmpsf
:marksky [y/n]                 Mark the sky annuli on the display

:weighting                     Weighting for wphot
:aperts [string]               Aperture radii in fwhmpsf
:zmag [value]                  Zero point of magnitude scale
:exposure [string]             Exposure time keyword
:itime [value]                 Integration time
```

We select the interactive setup option, move the image cursor to a high signal-to-noise, isolated star and tap the i key. PHOT responds by plotting the radial profile of the star on the screen and requesting the user to mark the fwhm of the psf, the centering aperture, the inner and outer sky annuli, the sky background and sigma and the set of circular apertures. The parameters so set can be examined and/or reset with the : commands as shown above. Sample measurements can be made of several stars by moving the cursor and typing the f command. Finally when we are happy with the parameter set we type w to store the parameters and q to exit the program.

Now we are ready to do photometry. We enter the PHOT program in batch mode.

```
ap> phot blue inter- &
```

The batch job is now running, appending output lines to the file "blue.mag.#". We can proceed to set up the job for the red image, in much the same way that we set up the job for the blue image. When both jobs finish, we can use the list processing tools to filter out the good objects and calculate colors.

5. The APPHOT Tasks

Manual pages for the APPHOT tasks are available in the IRAF on line help database.