APPENDIX: HOW THESE FOCUSING MASK CALCULATIONS ARE DONE by HR Suiter

These calculations were performed in the formalism of Fourier optics, and while there isn't sufficient room to present the totality of Fourier optics, I can present some sources for those interested in it. Please see my article on the Strehl ratio in the "Articles" section on this same website, which explains many of the same concepts and the ATM Journal article. (I did not write the Matlab script used in the spider article, but it and my own below must have been very similar.)

Joseph W. Goodman, *Introduction to Fourier Optics*, pp. 120+ in particular, McGraw-Hill 1968.

George B. Parrent, Jr. and Brian J. Thompson, *Physical Optics Notebook*, SPIE 1969.

The Best of ATM Journal, Volume 1, reprint (*circa* 1997) ed. by William J. Cook, "Dialogue on Spider Diffraction," by H.R. Suiter and William Zmek, pg. 385-406, Willmann-Bell, Inc.

The model Fourier-transforms a description of what is called the pupil function over the unit circle, i.e.

$$P(x,y) = T(x,y)e^{(i2\pi\theta(x,y))} \quad . \tag{A-1}$$

Here x and y are the dimensions across the aperture pupil, T is the transmission or maximum wave amplitude at the position x and y, phase θ is in units of wavelength, and the complex exponential is equivalent to cos(2πθ)+ i sin(2πθ) by the Euler formula. We assume that the point source is on axis and the mask (which is far in front of the axial position of the pupil) is projected to the pupil. The unit circle is defined as a discrete square sampling, 256 samples in diameter. Perhaps 512 would have been better, but I decided 256 was a good compromise between results and fast computation. To have an Airy disk computed to finer divisions, I must blank-pad the pupil by additional zeros till it is much larger than the lit portion. Default is a radix-2 value of 2048, or about (2048/256) x 1.22 ~ 10 values from the center to the edge of the focused Airy disk (1.22 comes from the Bessel-function expression for the Airy disk radius, i.e., the common r = 1.22 λ*F*, where *F* is the aperture ratio *f*/D).

To describe the residual aberration, I use the lowest Zernike polynomials (M. Born and E. Wolf, 7th ed pg. 523+, 1st ed 1959, Cambridge 1990) for the phase error, with the exception of piston and the two distortions, which only add uniform constants or move the image around. Defocus would be enough for this article, but to make this program of greater use to me for other calculations,[*] I also added several other aberrations. The set of Zernikes used are given by coefficients scaled to the root mean square (RMS) deviation of the surface. The model doesn't include them all, but uses a sequential coefficient "Coeff" to denote the ones it does use.

Standard Coeff	Polynomial	Name	Coeff entry
Z 4	3^(1/2) (2p^2 - 1)	defocus	Coeff(1)
Z 5	6^(1/2) (p^2) * COS(2A)	astigmatism	Coeff(2)
Z 6	6^(1/2) (p^2) * SIN (2A)	astig 45 deg	Coeff(3)
Z 7	8^(1/2) (3p^3 - 2p) * COS (A)	x coma	Coeff(4)
Z 8	8^(1/2) (3p^3 - 2p) * SIN (A)	y coma	Coeff(5)
Z 11	5^(1/2) (6p^4 - 6p^2 + 1)	low order sph aberr	Coeff(6)
Z 9	8^(1/2) (p^3) * COS(3A)	trefoil	Coeff(7)
Z 10	8^(1/2) (p^3) * SIN (3A)	trefoil 30 deg	Coeff(8)
Z 22	7^(1/2) (20p^6 - 30p^4 + 12p^2 - 1)	zonal spherical aberr	Coeff(9)

[*] With the passage of time and bit-depth, my *STAT* programs ASYMM and APERTURE no longer work. This program, and the commercial program ZEMAX, are work-arounds.

Here "p" is the radius on the unit circle defining the lit pupil. Trefoil is a 3-sided deformation that serves to model poorly mounted optics. The advantage of Zernike polynomials is that they are mutually orthogonal. Thus, they may be used independently. The yellow-background coefficient is the only one used in this article.

There is also a Coeff(10) tacked on to add a turned-down edge. It is not a Zernike polynomial and it doesn't work well over a coarse 256-sample diameter pupil. I have not yet removed it.

I wrote the script PupilGenerMask.m in the language Matlab. Afterward I tried it with the open source Matlab clone GNU Octave and had to modify only one line.

OPERATION OF PUPILGENERMASK

The program script is at the end of this file.

Before starting the program adjust the coefficients. The first coefficient (defocus) is set in the distribution version at the value giving approximately a Strehl ratio of 0.8. The direction of focus is positive, meaning that it is outside of focus.

```
% Zernikes with turned-down edge glued-on at end
% have neglected Z's between trefoil and 5th-order spherical
% All in RMS units except turned edge.
Coeff = [0.075          ... defocus
         0              ... x-y astig
         0              ... 45-degree astig
         0              ... x-axis coma
         0              ... y-axis coma
         0              ... 3rd-order spherical aberration
         0              ... trefoil
         0              ... trefoil + 30 degrees
         0              ... 5th order spherical aberration
         0              ... turned-down edge in wavelength units
         ];
```

The program, when it begins, asks four questions:

```
>> PupilGenerMask
Array size around pupil (radix 2 and more than 512)? [2048]
Fractional radius of obstruction? [0 or built-in]
Renormalize (0 is no; any is yes)? [1]
Power law of PSF display (root power > 0)[3.0]?
```

The first is the size of the blank padded-array. It must be radix-2 (i.e., an integer power of two) and larger than 512. In effect, this limits it to 512, 1024, 2048, or 4096. You can go larger, but I can't think why. The square brackets (2048) are the default if you hit <enter>.

The second is the fractional radius of obstruction. This defaults to 0, but if you have a built-in obstruction in the mask it will still work. Spider masks, especially curved ones, tend to have built-in obstructions.

The third is whether or not the point-spread function (PSF) is renormalized to 1.0 after computation. This is turned on by default. The user should turn it off only when interested in the computation of numerical values. Otherwise the PSF will appear too dark to display properly.

Last is the contrast-decreasing inverse power to apply to the displayed PSF image. At one stage the image is a number less than 1. Then it is taken to the (1/power). Two means a square root; the default is a cube root. Ordinary full-aperture diffraction behavior works fine with the cube root. Should delicate low amplitude behavior be desired to show, however, you may have to input a number of 4 to 6. It is multiplied by 255 to resemble other images and displayed.

Then a dialog box opens and you must select a pupil function. I have a set of examples, but this is by no means complete. I made them TIF files to distinguish them from output JPG files. If you have no way of making or storing a TIF file, then you can change the line (near the front) to add JPG too: you must specify the secondary choices in the File Type box. Pupil files should be in black and white and exactly 256 by 256.

The transmittance pattern Blank.tif results in a circular pupil, not a square one. If you want to see the effect of a square pupil, you have to draw it inside the aperture, such as in SquarePup.tif.

THE OUTPUT

"Figure 1" or the Point Spread Function (PSF).
Example Matlab Figure 1 displays are shown in this appendix Figure A-1. You can use this program to reproduce many of the monochromatic figures appearing in my book *Star Testing Astronomical Telescopes (STAT)*, as long as you know how to scale these RMS coefficients to the peak-to-valley coefficients appearing there. This is not as straightforward as it seems; no simple multiplier is used to do it. You are better off fishing for the correct RMS because of the rough edges of the sampled pupil. Still, you are able to very closely duplicate them. You can also generate the wildly distorted and beautiful patterns of extreme aberrations as in Figure A-1(L).

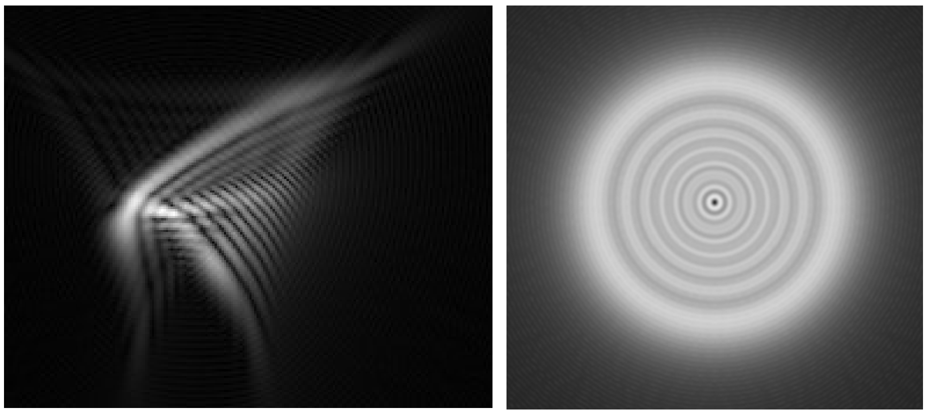

Figure A-1. (L) Example image with C(3) = 1.0, C(5) = 0.5 , and C(7) = 0.5 on circular pupil with defaults on queries. (R) Approximately 8 wavelengths P-V out of focus C(1) = 2.31 RMS. Blank.tif pupil. Compare to Figure B.12 of *STAT*.

The chief thing concerning us in this article is the appearance of the image with zero or small amounts of defocus in the presence of an aperture mask.

"Figure 2" or the Modulation Transfer Function (MTF) depicted as an image
I seldom use this for this application and so will not comment on it here.

"Figure 3" or the Cross Sections of the MTF
The x and y cross sections of the MTF image are displayed as red and blue curves in Figure A-2. The MTF curves are centered at 1024 and have a radius of 256, same as the diameter of the lit pupil. This is the cross section of the three-dimensional figure plotted in "Figure 2."

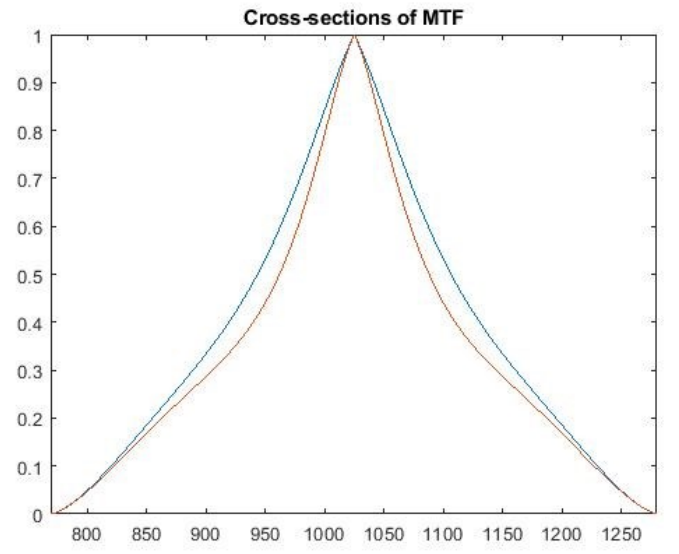

Figure A-2. "Figure 3" of Matlab output. MTF cross sections of a comatic image in each direction.

"Figure 4" or the Cross-Sections of the Phase Function
The phase function θ of equation A-1 is next depicted in "Figure 4" which is shown here for the same coma in Figure A-3. The coma is exhibited only in one dimension.

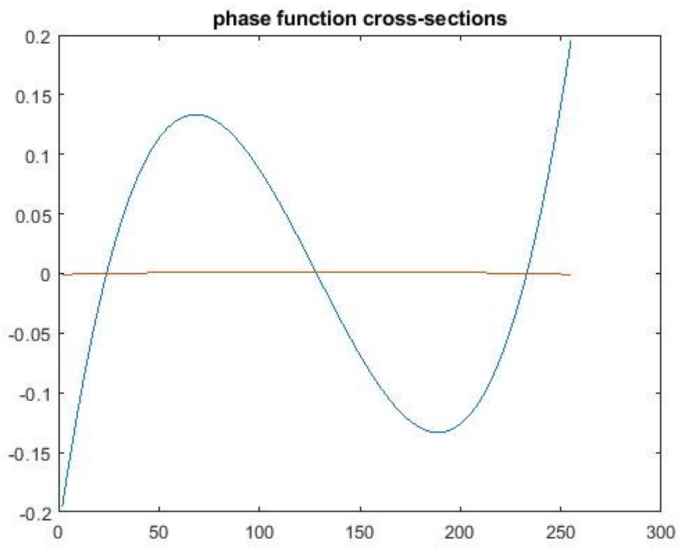

Figure A-3 A small amount of coma is depicted

"Figure 5" or the Cross Sections of the Point Spread Function

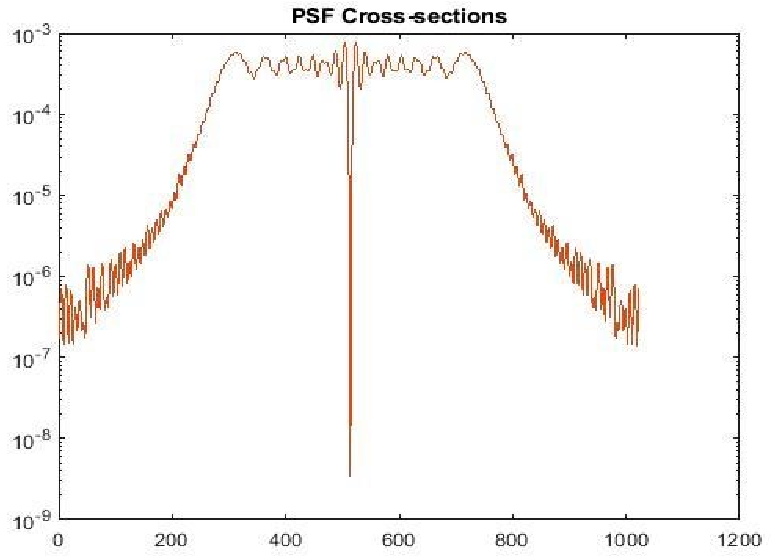

Figure A-4. Two identical cross-sections are plotted on top of one another, this one for defocus value ~ 8 wavelengths

We can perform a validation of the performance of the Matlab script by plotting a piece of the above curve against the old values calculated by `APERTURE` and the semi-analytic Lommel function expansion.[1] This comparison appears in Figure A-5 and can be compared with Figure B.8 on page 351 of *Star Testing Astronomical Telescopes (2nd ed)*. I used the Matlab script to set the value of peak-to-valley aberration to 8 wavelengths and did the calculation without renormalizing it. Remaining differences are

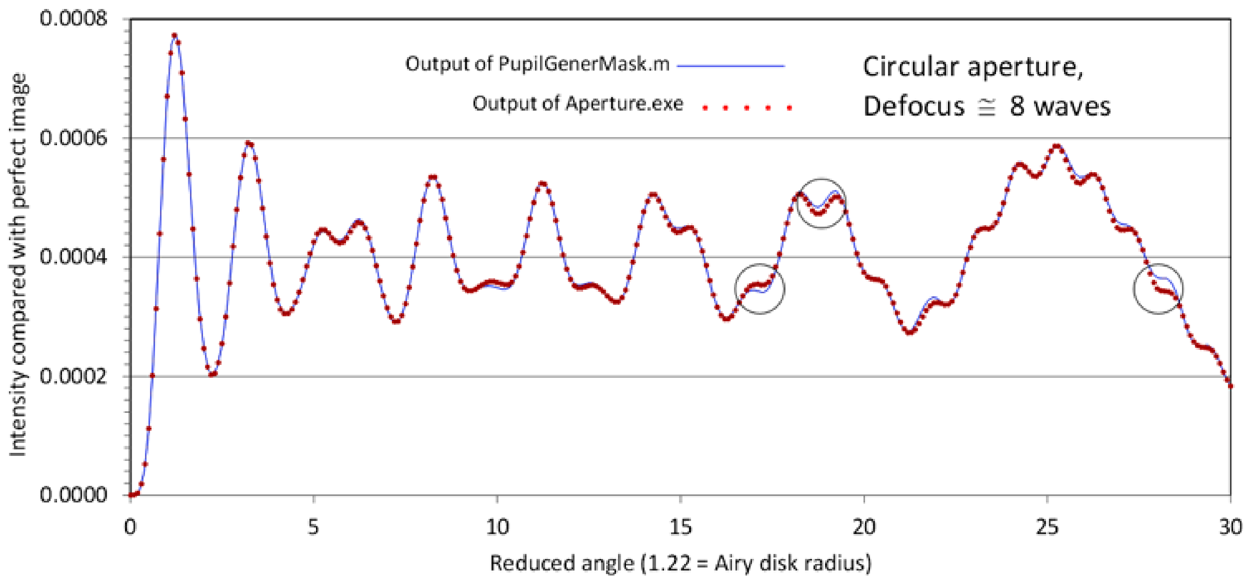

Figure A-5. Validation of performance of MATLAB script. Circles indicate regions of greatest deviation.

[1] M. Born and E. Wolf, *Principles of Optics, 7th ed*., p. 484+, Cambridge 1999 (1959 Pergamon).

probably caused by the ragged edges of the sampled pupil function. The one-dimensional APERTURE, because it compares almost perfectly with the Lommel function expansion, is probably the more accurate of the two routines in this symmetric case.

"Figure 6" or a Depiction of the Transmission Function

The last picture is a confirmation that the pupil transmission is what the user believes to be the case. When I drew the transmission function, I made it any size. As the last step, I put it in an image manipulation program and resampled it to size 256x256. Sometimes I forgot to do this, leading to bizarre results. The example in Figure A-6 shows the fuzzy diffraction-spike spider that is presented in Jean Texereau's *How to Make a Telescope,* as well as its spread pattern.

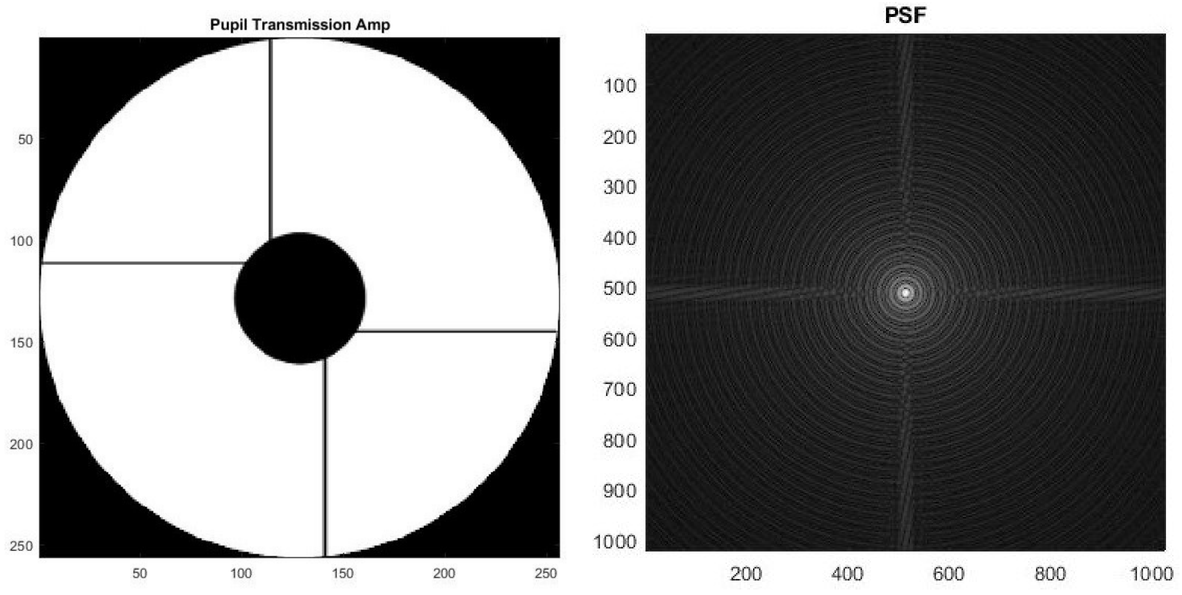

Figure A-6. (L) Confirmation of the transmission function. (R) PSF with contrast factor of 7.

THE SCRIPT

```
%%      PupilGenerMask,
%       by Harold R. Suiter. A program to calculate the PSF and MTF of a
%       low-order Zernike polynomial with non-symmetric mask supplied
%       externally in a 256x256 tiff file, T = 1 is 255 in file. Central
%       obstruction and aperture edge need not be in mask.

clear                                 % clear memory
close all;                            % close existing figures
Endmessage = ' ';                     % can fill with any info

% Zernikes polynomials with added turned-down edge appended at end
% have neglected Z's between trefoil and 5th-order spherical
% All in RMS units except turned edge.
% ============================================================
% Color images giving Strehl ~ 0.8 RGB: 0.068|0.075|0.083 rms
% adap. from Bruce H. Walker, Optical Design for Visual Systems, SPIE 2000
% image resampled externally RGB 1.22|1.11|1.0, cropped and reassembled
```

```
Coeff = [0.075          ... defocus
         0              ... x-y astig
         0              ... 45-degree astig
         0              ... x-axis coma
         0              ... y-axis coma
         0              ... 3rd-order spherical aberration
         0              ... trefoil
         0              ... trefoil + 30 degrees
         0              ... 5th order spherical aberration
         0              ... turned-down edge in wavelength units
         ];

Big = input('Array size around pupil (radix 2 and more than 512)? [2048] ');
if isempty(Big)
    Big = 2048;
end
if Big < 512
    Endmessage = 'Array size < 512 results in erroneous MTF';
end

Obst = input('Fractional radius of obstruction? [0 or built-in] ');
if isempty(Obst)
    Obst = 0;
end

renormFlag = input('Renormalize (0 is no; any is yes)? [1] ');
if  renormFlag == 0
    renormFlag = 0;
else
    renormFlag = 1;
end

powerPSF = input('Power law of PSF display (root power > 0)[3.0]? ');
if isempty(powerPSF)
    powerPSF = 0.3333;
else
    powerPSF = 1/powerPSF;
end

[fn, pn] = uigetfile('*.tif','Pick mask to process ');
%[fn, pn] = uigetfile({'*.tif';'*.jpg'},'Pick mask to process ');
% Blank.tif yields simple circular aperture.

pathfile = [pn, fn];
T = double(imread(pathfile));
T = T/255;              % normalize transmission coeff to 1

if Obst > 0.99          % 1 or greater probably means percentage
    Obst = 0.01*Obst;
end

R0 = 128;               % will initially have pupil
                        % inside padded 256x256 array,
                        % but may change

for i = 1:256               % center is at 128.5
%   i                       % index number as marker shown for progress
                            % when comment mark removed
    pxpup = (i-128.5)/R0;   % pixel x value as measured from center

    for j = 1:256
        pypup = (j-128.5)/R0;                  % pixel y value as
```

```
                                                % measured from center
        theta = atan2(pypup,pxpup);
        r = sqrt((i-128.5)^2+(j-128.5)^2);      % Array starts at 1!
        rho = r/R0;                             % normalized pupil

        TestRho = (rho <= 1) & (rho >= Obst);   % separate logical operator
                                                % for Octave
        if TestRho                              % query whether in lit pupil
            ph = zernike(Coeff,rho,theta);      % standard Zernikes
            ph = ph - Coeff(10)*rho^60;         % -- add a non-Zernike edge
                                                % with P-V units
            phas(i,j)= ph;
            Pupdisp(i,j) = T(i,j);
            Pup(i,j)= T(i,j)*exp(sqrt(-1)*2*pi*ph); % pupil function
        end
    end
end

% pupil-to-image transformation
% padded to BigxBig and reshuffled corners of 2D array to center
PSF = fftshift(fft2(Pup,Big,Big));

    % get rid of unused memory
      clear Pup

PSF = abs(PSF);                 % convert field to observed wave
PSF = PSF.*PSF;                 % intensity

renormPSF = 2.648955e+09;       % empirical strength of lit circ pupil
if renormFlag == 1
    renormPSF = max(max(PSF));
end
MTF = abs(fftshift(fft2(PSF))); % MTF before prettying PSF

YCS = PSF(Big/2+1,1:Big)/renormPSF; % accurate versions before I scale
XCS = PSF(1:Big,Big/2+1)/renormPSF; % for visual

renormMTF = max(max(MTF));      % maxes one axis at a time

MTF = MTF/renormMTF;            % rescale field
PSF = PSF/renormPSF;

% can change graphed image contrast here to emphasize dim radiances
PSF = PSF.^ (powerPSF);
PSF = PSF*256;              % Depending on application, might want to
                            % multiply this by a constant before displaying.

BegXY = Big/2-512;

PSF(:,Big-BegXY:Big-1)= [];
PSF(:,1:BegXY)=[]     ;
PSF(Big-BegXY:Big-1,:)=[] ;
PSF(1:BegXY,:)=[]     ;

% The point of the script is the Figure 1 PSF on top. You may want
% to comment the others out, but I prefer just to ignore them.

% PupDisp plat
figure(6);
colormap(gray(256));
Pupdisp = uint8(255*Pupdisp);
image(Pupdisp);
axis equal tight;              % square image w/ no borders
```

```
title('Pupil Transmission Amp');

% Fig 5
figure(5);
XCScenter = XCS(Big/4:(3*Big/4)-1);
YCScenter = YCS(Big/4:(3*Big/4)-1);
semilogy(XCScenter);
hold on;
semilogy(YCScenter);
title('PSF Cross-sections');
hold off

% Fig 4
figure(4);
plot([2:255],phas(2:255,128),[2:255],phas(128,2:255));
title('phase function cross-sections');
PtoV = max(max(phas))-min(min(phas));

% Fig 3
figure(3);
YCSmtf = MTF(Big/2+1,1:Big);             % For off-line review;
XCSmtf = MTF(1:Big,Big/2+1);             % center at Big/2
plot([1:Big],YCSmtf,[1:Big],XCSmtf);
axis([Big/2-255 Big/2+256 0 1])      % MTF of a 256 diam pupil is 512 wide
title('Cross-sections of MTF');
MTF = 256*MTF;

% Fig 2
figure(2);
colormap(gray(256));
image(MTF);
axis equal tight;                % square image w/ no borders
title('2D MTF function');

% Fig 1                          % 2D PSF moved to outside so won't
figure(1);                       % need to click through others
colormap(gray(256));
image(PSF);
axis equal tight;                % square image w/ no borders
title('PSF');

disp(Endmessage)
disp('Array size'); disp(Big);
disp('Obstruction'); disp(Obst);

% exit program

function ph=zernike(Coeff,rho,theta)
% Zernike circle polynomials in same order as used in ZEMAX
% limited to low order trefoil except no distortion, no piston,
% and coefficients between trefoil and zonal spherical aberration
% are neglected.
  ph =      Coeff(1)*sqrt(3)*(2*rho^2-1);                      % defocus
  ph = ph + Coeff(2)*sqrt(6)*rho^2*cos(2*theta);               % astig
  ph = ph + Coeff(3)*sqrt(6)*rho^2*sin(2*theta);               % 45 astig
  ph = ph + Coeff(4)*sqrt(8)*(3*rho^3-2*rho)*cos(theta);       % x coma
  ph = ph + Coeff(5)*sqrt(8)*(3*rho^3-2*rho)*sin(theta);       % y coma
  ph = ph + Coeff(6)*sqrt(5)*(6*rho^4-6*rho^2+1);              % sph aberr
  ph = ph + Coeff(7)*sqrt(8)*rho^3*cos(3*theta);               % trefoil
  ph = ph + Coeff(8)*sqrt(8)*rho^3*sin(3*theta);               % 30 tref
  ph = ph + Coeff(9)*sqrt(7)*(20*rho^6-30*rho^4+12*rho^2-1);  % 5th sph
end
```