W. W. Hansen Experimental Physics Laboratory

STANFORD UNIVERSITY

Stanford, California 94305-4085

Gravity Probe B Relativity Mission

S0630 Rev. –

March 22, 2002

Linearity of Flight Telescope

Prepared by

J. P. Turneaure

ITAR Assessment Performed Date

Tom Langenstein ______ ITAR Control Required? Yes/No ______ 3/24/02

APPROVALS

3/22/02

John Turneaure — Co-PI, Preparer — Date

3/22/02

Mac Keiser — Chief Scientist — Date

CHANGE RECORD

LIST OF ACRONYMS AND SYMBOLS

θ	Angle of knife-edge with respect to coordinate system of modified Zernike polynomials
coeff	Magnitude of aberration term in waves at 668 nm
i	Label identifying the four photodetector pairs, see Table 1-1
m	Angular order (integer) of modified Zernike polynomial
n	Radial order (integer) of modified Zernike polynomial
$dS_n/d\theta$	Derivative of the normalized telescope pointing signal with respect to pointing angle
$(dS_n/d\theta)_{Av,i}$	Weighted average of $dS_n/d\theta$ across the 400 nm to 1000 nm wavelength band for photodetector pair i
$T(\theta)$	Pointing angle derived from normalized telescope signal as a function of the true pointing angle

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

March 22, 2002

0 SUMMARY

This document contains analyses of the linearity of the flight telescope/window system. The worst-case nonlinearity found in these analyses is given in Table 0-1, which corresponds very closely to the nonlinearity of a flight telescope/window system with no aberration and in focus. As expected, the nonlinearity is found to increase approximately as the third power of pointing angle range.

Table 0-1. Worst-case nonlinearity for the flight telescope/window system.

Range (marcsec)	Maximum Absolute Deviation from Linearity (marcsec)
-100 to +100	0.41
-200 to +200	3.4

In verification document S0619, we were able to approximate the observed derivative of the normalized telescope signal with respect to pointing angle assuming an aberration using a modified Zernike polynomial with $n = 4$, $m = 4$, and with an amplitude of –0.4 waves at 668 nm. The analysis in verification document S0570 found that the telescope is about 1.3 waves out of focus. Using these conditions, the most likely nonlinearity should be close the values in Table 0-2.

Table 0-2. Most likely nonlinearity for the flight telescope/window system.

Range (marcsec)	Maximum Absolute Deviation from Linearity (marcsec)
-100 to +100	0.13
-200 to +200	1.1

This document is maintained in the GP-B database in two forms: first as an Adobe Acrobat file accessible directly in the database, and second as the set of files given in Table 0-3 in their native form available in the GP-B database directory.

Table 0-3. Set of S0630 files in their native source form.

Description	Filename	File type
Main text	S0630 Rev.-.doc	Microsoft *Word*®
Appendix 1	S0630 Apndx 1 Rev.-.nb	*Mathematica*®
Appendix 2	S0630 Apndx.for	Text File

March 22, 2002

1 INTRODUCTION

This report documents the analyses of the linearity of the flight/telescope system. To first order, one would expect on general principles that the greatest nonlinearity would occur for a perfect telescope in focus and with no aberration since this is the condition that yields the sharpest image and thus greatest change in the derivative of the normalized telescope pointing signal with respect to pointing angle averaged across the optical band, $(dS_n/d\theta)_{Av,i}$. To test this hypothesis, we examine the nonlinearity due to various degrees of aberration and of defocus.

2 Methodology

We first need to calculate the optical intensity distribution at wavelength λ in the plane of the knife-edges. This needs to be done numerically for all cases except for the perfect telescope when in focus. These calculations are made with the Fortran program listed in appendix 2, which allows one to include any aberration up to 8^{th} order in modified Zernike polynomials for the clear aperture of the telescope, which is a circular annulus. We verified that the numerical calculations for a perfect telescope are in agreement with the analytical results for this same case.

The optical intensity distribution is calculated along lines spaced at pointing angle intervals of 10 marcsec from –200 marcsec to +200 marcsec at wavelengths intervals of 100 nm from 400 nm to 1000 nm. The *Mathematica*® notebook listed in appendix 1 uses the optical intensity distribution to calculate $dS_n/d\theta$ at the pointing angle intervals and wavelength intervals. Also with the same *Mathematica*® notebook, weighted averages across the optical band of $dS_n/d\theta$ are then found $(dS_n/d\theta)_{Av,i}$ at the 10 marcsec intervals for each of the photodetector pairs using Eq 6-4 in verification document S0619. This is done for each of the photodetector pairs listed in Table 1-1 in S0619. The weighted averages use the same data for the spectrum of HR8703, transmission coefficients, and quantum efficiency of the photodetectors found in S0619.

The values of $(dS_n/d\theta)_{Av,i}$ are then integrated with respect to pointing angle using the triangle rule to give a signal $T(\theta)$ at 10 marcsec intervals that represents the measured angle. A linear fit of the signal $T(\theta)$ is made to the pointing angle θ for ranges of both ±100 marcsec and ±200 marcsec. The nonlinearity is the maximum absolute deviation from the linear fit.

3 Telescope with no Aberration

The nonlinearity of the flight telescope/window system with no aberration was calculated for three values of defocus: 0.0 waves, 0.65 waves, and 1.3 waves. The defocus refers to the spherical wavefront error in the optical beam incident on a perfect telescope at a reference wavelength of 668 nm. This is equivalent to displacing the position of the knife-edges relative to the focal position. Figure 3-1 is a plot of the deviation from linearity for the case of no defocus (0.0 waves) over a ±200 marcsec range. For this case as well as all of the other cases explored, the deviation from linearity over the

±200 marcsec range is proportional to the third power of the pointing angle with an accuracy of better than 2%. This proportionality is expected for a narrow pointing range since $dS_n/d\theta$ is to first order parabolic about null pointing of the telescope.

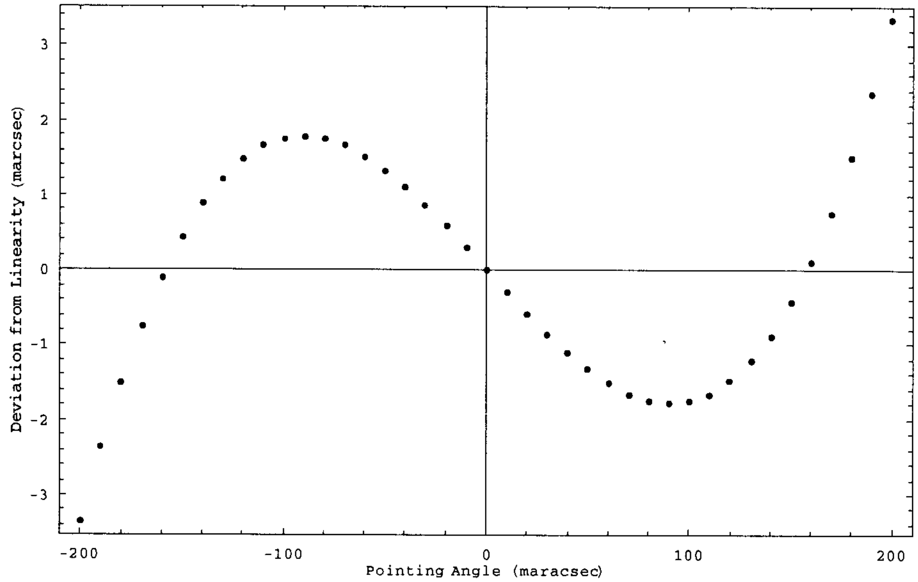


Figure 3-1. Deviation from linearity for a perfect flight telescope with no aberration or defocus for photodetector pair 1.

Table 3-1 lists the results for all three degrees of defocus for each of the photodetector pairs. As can be seen from the table, the nonlinearity decreases with increasing defocus. This is of course expected since the intensity distribution at the knife-edges is not as sharply focused. The measured defocus of the flight telescope/window system was found to be about 1.3 waves (see Fig. 9-1 of S0570). One might assume that the nonlinearity for this degree of defocus might be the most likely value for the flight telescope/window system; however, as seen in the next section it is probably an underestimate.

Table 3-1. Maximum deviation from linearity for telescope with no aberration and with three degrees of defocus for each of the four photodetector pairs.

Defocus (waves)	Max Deviation for ±100 marcsec range for Detector Pair 1…4 (marcsec)				Max Deviation for ±200 marcsec range for Detector Pair 1…4 (marcsec)			
	1	2	3	4	1	2	3	4
0.00	0.400	0.380	0.390	0.391	3.36	3.20	3.28	3.28
0.65	0.240	0.239	0.243	0.236	2.02	2.01	2.05	1.98
1.30	0.062	0.055	0.057	0.060	0.50	0.45	0.46	0.49

4 Telescope with Aberration and Defocus

In this section we investigate the affects of aberration terms on the nonlinearity. The aberration terms are expressed as coefficients of modified Zernike polynomials for the clear aperture of the telescope, which is a circular annulus. These polynomials, described more fully in S0619, are characterized by the integers n and m, which correspond to the radial and angular orders. The coefficient is expressed in waves at the reference wavelength of 668 nm. Further, the knife-edges can be placed at any angle θ relative to the coordinate system of the Zernike polynomials. We only investigate aberrations with even values of m (it follows that n must also be even) since the experimental values of $dS_n/d\theta$ as a function of decollimation (equivalent to defocus) for the flight telescope/window system are essentially the same for the x- and y-axes of the telescope. This selection is necessary because of the computational demands for these calculations, which for the results given in the tables took about 2 weeks of computation time. The Zernike coefficients used in the calculations are chosen to approximate the measured maximum value of $dS_n/d\theta$ at 668 nm, which is 0.8 /arcsec for both axes (see S0570).

The results are given in Tables 4-1, 4-2, and 4-3. Each of the tables contains the order of the modified Zernike polynomial $\{n, m\}$, the coefficient in waves (Coeff), the defocus in waves, and the angle of the knife-edge θ. Table 4-1 lists the calculated values of $(dS_n/d\theta)_{Av,i}$ for each of the four photodetector pairs demonstrating that $(dS_n/d\theta)_{Av,i}$ is reduced from the value of 1.25 for a perfect telescope in focus. Tables 4-2 and 4-3 list the maximum absolute deviations from linearity for a ±100 marcsec range and a ±100 marcsec range, respectively.

Inspection of the nonlinearity data in the tables corroborates the general conclusion that the greatest nonlinearity is for a perfect telescope in focus. There are only 2 examples out 28 in the table that have a nonlinearity essentially equal to that of a perfect telescope. These examples are the one with $n = 4$, $m = 4$, coeff = -0.4 waves, defocus = -0.8 waves, and $\theta = 0$ degrees, and its symmetrically related partner with defocus = 0.8 waves, and $\theta = 45$ degrees. For these two examples, the nonlinearity is slightly higher but within 5% for the case of a perfect telescope in focus. Although, these two examples are far from the measured conditions for the flight telescope/window system, we take them as the worst-case estimate of nonlinearity. For these two examples the maximum deviations from linearity are 0.41 marcsec for the ±100 marcsec range and 3.4 marcsec for the ±200 marcsec range.

As discussed in S0619, the measured properties of the flight telescope/window system are best matched with an aberration term with $n = 4$, $m = 4$, coeff = -0.4 waves, defocus = 1.57 waves (1.6 waves in the tables of this document). Note that for this aberration a defocus of 1.57 waves is 1.3 waves from the maximum value of $dS_n/d\theta$, which is at 0.27 waves. The maximum deviations from linearity from the tables for this aberration term are 0.13 marcsec for the ±100 marcsec range and 1.1 marcsec for the ±200 marcsec range. We take these values as the most likely values for the actual flight telescope/window system. Note that, the nonlinearity for this case is greater than the nonlinearity for a perfect telescope with a comparable amount of defocus.

Table 4-1. $(dS_n/d\theta)_{Av,i}$.for various aberration terms.

n	m	Coeff (waves)	Defocus (waves)	θ (degrees)	$(dS_n/d\theta)_{Av,I}$ (/arcsec)			
					1	2	3	4
6	0	0.4	0.00	-	0.568	0.577	0.571	0.573
8	0	0.4	0.00	-	0.662	0.669	0.666	0.665
2	2	0.5	0.00	0.0	0.867	0.868	0.872	0.864
2	2	0.5	0.00	45.0	0.228	0.225	0.223	0.229
4	2	0.5	0.00	0.0	0.553	0.558	0.555	0.555
4	2	0.5	0.00	45.0	0.275	0.271	0.272	0.275
6	2	0.5	0.00	0.0	0.764	0.766	0.767	0.763
6	2	0.5	0.00	45.0	0.571	0.578	0.573	0.575
4	4	-0.4	-1.60	0.0	0.110	0.110	0.110	0.110
4	4	-0.4	-0.80	0.0	0.357	0.356	0.355	0.358
4	4	-0.4	0.00	0.0	0.828	0.828	0.831	0.826
4	4	-0.4	0.80	0.0	0.759	0.754	0.760	0.755
4	4	-0.4	1.60	0.0	0.429	0.426	0.429	0.426
4	4	-0.4	-1.60	45.0	0.426	0.423	0.426	0.423
4	4	-0.4	-0.80	45.0	0.758	0.753	0.758	0.753
4	4	-0.4	0.00	22.5	0.673	0.675	0.675	0.674
4	4	-0.4	0.80	45.0	0.358	0.394	0.357	0.359
4	4	-0.4	1.60	45.0	0.110	0.110	0.110	0.111
4	4	-0.4	-0.30	22.5	0.630	0.631	0.630	0.631
4	4	-0.4	0.00	22.5	0.673	0.675	0.675	0.674
4	4	-0.4	0.30	22.5	0.630	0.631	0.631	0.631
4	4	-0.4	-0.30	67.5	0.630	0.631	0.630	0.631
4	4	-0.4	0.00	67.5	0.673	0.675	0.675	0.674
4	4	-0.4	0.30	67.5	0.630	0.631	0.631	0.631
6	4	0.4	0.00	0.0	0.850	0.850	0.854	0.847
6	4	0.4	0.00	45.0	0.850	0.850	0.854	0.847
8	4	0.4	0.00	0.0	0.894	0.893	0.898	0.890
8	4	0.4	0.00	22.5	0.815	0.818	0.820	0.814

Table 4-2. Maximum deviation from linearity over a ±100 marcsec range for various aberration terms.

n	m	Coeff (waves)	Defocus (waves)	θ (degrees)	Max. Dev. from Linearity (marcsec) 1	2	3	4
6	0	0.4	0.00	-	0.274	0.271	0.277	0.268
8	0	0.4	0.00	-	0.308	0.301	0.310	0.300
2	2	0.5	0.00	0.0	0.300	0.292	0.300	0.293
2	2	0.5	0.00	45.0	0.367	0.377	0.394	0.354
4	2	0.5	0.00	0.0	0.277	0.266	0.275	0.269
4	2	0.5	0.00	45.0	0.261	0.251	0.256	0.258
6	2	0.5	0.00	0.0	0.300	0.290	0.298	0.294
6	2	0.5	0.00	45.0	0.326	0.317	0.326	0.318
4	4	-0.4	-1.60	0.0	0.119	0.115	0.115	0.119
4	4	-0.4	-0.80	0.0	0.408	0.394	0.407	0.397
4	4	-0.4	0.00	0.0	0.378	0.362	0.374	0.368
4	4	-0.4	0.80	0.0	0.221	0.214	0.218	0.217
4	4	-0.4	1.60	0.0	0.127	0.122	0.125	0.124
4	4	-0.4	-1.60	45.0	0.126	0.122	0.125	0.124
4	4	-0.4	-0.80	45.0	0.222	0.214	0.219	0.218
4	4	-0.4	0.00	22.5	0.117	0.121	0.120	0.119
4	4	-0.4	0.80	45.0	0.408	0.394	0.406	0.396
4	4	-0.4	1.60	45.0	0.119	0.115	0.115	0.119
4	4	-0.4	-0.30	22.5	0.127	0.128	0.126	0.128
4	4	-0.4	0.00	22.5	0.117	0.121	0.120	0.119
4	4	-0.4	0.30	22.5	0.126	0.128	0.126	0.128
4	4	-0.4	-0.30	67.5	0.127	0.128	0.126	0.128
4	4	-0.4	0.00	67.5	0.117	0.121	0.120	0.119
4	4	-0.4	0.30	67.5	0.126	0.128	0.126	0.128
6	4	0.4	0.00	0.0	0.351	0.337	0.347	0.342
6	4	0.4	0.00	45.0	0.351	0.337	0.347	0.342
8	4	0.4	0.00	0.0	0.340	0.328	0.337	0.331
8	4	0.4	0.00	22.5	0.339	0.327	0.337	0.330

Table 4-3. Maximum deviation from linearity over a ±200 marcsec range for various aberration terms.

n	m	Coeff (waves)	Defocus (waves)	θ (degrees)	Max. Dev. from Linearity (marcsec)			
					1	2	3	4
6	0	0.4	0.00	-	2.31	2.29	2.35	2.27
8	0	0.4	0.00	-	2.60	2.54	2.62	2.54
2	2	0.5	0.00	0.0	2.52	2.46	2.53	2.46
2	2	0.5	0.00	45.0	3.05	3.13	3.26	2.95
4	2	0.5	0.00	0.0	2.34	2.25	2.33	2.27
4	2	0.5	0.00	45.0	2.11	2.03	2.07	2.08
6	2	0.5	0.00	0.0	2.54	2.46	2.52	2.48
6	2	0.5	0.00	45.0	2.72	2.65	2.73	2.65
4	4	-0.4	-1.60	0.0	0.96	0.93	0.93	0.96
4	4	-0.4	-0.80	0.0	3.36	3.26	3.36	3.27
4	4	-0.4	0.00	0.0	3.17	3.05	3.14	3.08
4	4	-0.4	0.80	0.0	1.88	1.82	1.86	1.85
4	4	-0.4	1.60	0.0	1.08	1.04	1.07	1.05
4	4	-0.4	-1.60	45.0	1.08	1.04	1.07	1.05
4	4	-0.4	-0.80	45.0	1.89	1.83	1.86	1.85
4	4	-0.4	0.00	22.5	1.02	1.06	1.04	1.04
4	4	-0.4	0.80	45.0	3.36	3.25	3.36	3.27
4	4	-0.4	1.60	45.0	0.96	0.93	0.93	0.96
4	4	-0.4	-0.30	22.5	1.10	1.11	1.09	1.11
4	4	-0.4	0.00	22.5	1.02	1.06	1.05	1.04
4	4	-0.4	0.30	22.5	1.10	1.10	1.10	1.11
4	4	-0.4	-0.30	67.5	1.10	1.10	1.11	1.09
4	4	-0.4	0.00	67.5	1.02	1.06	1.04	1.04
4	4	-0.4	0.30	67.5	1.10	1.10	1.09	1.11
6	4	0.4	0.00	0.0	2.96	2.85	2.93	2.88
6	4	0.4	0.00	45.0	2.96	2.85	2.93	2.88
8	4	0.4	0.00	0.0	2.86	2.77	2.85	2.79
8	4	0.4	0.00	22.5	2.84	2.76	2.84	2.77

Appendix 1

Listing of *Mathematica*® Notebook used for Calculating the Linearity Averaged across the Optical Band

Calculate Linearity Averaged across the Optical Band for a Given Intensity Distribution at the Knife Edges

John P. Turneaure
March 12, 2002

■ Prepare Input Data

■ Define Filename

Define input filename.

```
In[1]:= filename = "z0001.txt"

Out[1]= z0001.txt
```

Set working directory.

```
In[2]:= SetDirectory["C:/Telescope/Linearity/Data Z"];
```

Turn off spelling warnings.

```
In[3]:= Off[General::spell1]; Off[General::spell];
```

Load graphics package.

<<Graphics`Graphics`;

■ Define Parameters and Scale Factor Function

The parameters describing the telescope are

```
In[4]:= values = {f → 150 .0254, a → 5.66/2 .0254,
        b → 2.795/2 .0254, ε → b / a, λ → 6.68 10^-7, area → π (a^2 - b^2)};
        Print["f = ", f /. values, "   a = ", a /. values, "   b = ",
         b /. values /. values, "   λ = ", N[λ /. values]];

        f = 3.81   a = 0.071882   b = 0.0354965   λ = 6.68×10^-7
```

Evaluate pow0, which is the total power incident on the telescope mirror.

```
In[5]:= pow0 = π (a^2 - b^2) /. values /. values;
```

Define function sf that calculates the normalized signal differential with respect to angle with units of 1/arcsec. It is a function of the same variables as power.

```
In[6]:= sf[n_, Δr_, dataA_] :=
          Module[{xx}, xx = Table[If[Or[i = 1, i == n], 1/3, If[i/2 ∈ Integers, 4/3, 2/3]], {i, n}];
           π f/(180 3600) (2 Δr)/pow0 Sum[xx[[i]] dataA[[i]]^2, {i, n}] /. values];

In[7]:= dIn = Get[filename];
```

■ Read and Print Input Data

Read fortran generated list containing the input data, and absolute amplitudes.

```
In[8]:= dIn = Get[filename];
        {theta, iN, dR, jL, jU, dW, kL, kU, dLam} = dIn[[1]];
        aC = dIn[[2]];
        aS = dIn[[3]];
        waves = dIn[[4]];
```

Print input data for reference.

```
In[13]:= Print["Theta = ", theta];
         Print[2 iN + 1, " radial points, spaced at ", dR, " m steps"];
         Print["Waves of de-collimation = j*dW, with j running from ",
           jL, " to ", jU, " and dW = ", dW];
         Print["Wavelength = k*dLam, with k running from ",
           kL, " to ", kU, " and dLam = ", dLam, " m"];
         Print["Coefficients in waves for symmetric Zernike polynomials", TableForm[aC]];
         Print["Coefficients in waves for antisymmetric Zernike polynomials", TableForm[aS]];
```

Theta = 0.

501 radial points, spaced at $2.\times10^{-6}$ m steps

Waves of de-collimation = j*dW, with j running from 1 to 1 and dW = 0.

Wavelength = k*dLam, with k running from 4 to 10 and dLam = $1.\times10^{-7}$ m

Coefficients in waves for symmetric Zernike polynomials

0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.

Coefficients in waves for antisymmetric Zernike polynomials

0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.

■ Establish Wavelength Dependencies from S0619

■ Define Constants

Define physical constants.

```
In[19]:= eCharge = 1.60218 × 10^-19;
         h = 2 π 1.05457 × 10^-34;
         c = 299792458;
```

Define clear area of telescope. The outside diameter of the telescope aperture is specified in Drawing SUGPB 25060F, and the inside diamter in Drawing SUGPB 25079 D.

```
In[22]:= a = (5.660 .0254)/2 (* aperture outer radius *);
         b = (2.795 .0254)/2 (* aperture inner radius *);
         ε = b/a (* ratio of inner to outer radius of aperture *);
         area = π (a^2 - b^2) (* area of telescope aperture *);
```

▪ Optical Wavelength Dependencies

This section provides the wavelength dependencies for the spectrum of HR8703, the transmission coefficient of the flight Windows, the transmission coefficients of the telescope and transport to the four photodiode pairs, and the quantum efficiency of the photodiodes.

▪ HR8703 Spectrum Data

Import spectrum data for HR8703, which is maintained in a separate file. The units of the data in the file are in Å for the wavelength and erg/cm^2/s/Å for the irradiance.

```
In[26]:= data = Import["HR8703_b.txt", "Table"];
```

Change units to μm and W/m^2/μm.

```
In[27]:= dataSI = Table[{data[[i, 1]]/10^4, 10^1 data[[i, 2]]}, {i, Length[data]}];
```

Change the wavelength domain so it has a dimensionless span from -1 to +1, which is more appropriate for the interpretation of the fit to a Legendre polynomial series. Print fit coefficients.

```
In[28]:= dataF = Table[{(dataSI[[i, 1]] - 0.7)/0.3, dataSI[[i, 2]]}, {i, Length[data]}];
         funcHR8703 =
           Fit[dataF, {1, x, LegendreP[2, x], LegendreP[3, x], LegendreP[4, x], LegendreP[5, x],
             LegendreP[6, x], LegendreP[7, x], LegendreP[8, x], LegendreP[9, x]}, x];
         Print[Table[If[i == 1, funcHR8703[[1]], funcHR8703[[i, 1]]], {i, 10}]];
```

$\{1.36789\times10^{-10}, 3.24244\times10^{-12}, -5.55415\times10^{-11}, 1.98633\times10^{-11}, -9.33645\times10^{-12}, 4.54866\times10^{-12}, 8.26535\times10^{-12}, 1.14348\times10^{-11}, 1.65046\times10^{-12}, -9.39463\times10^{-12}\}$

▪ Window Transmission

This section gives the window transmission as a function of wavelength. The data are for the flight windows.

```
In[31]:= dataWindows = {{0.400, .54}, {0.425, .59}, {0.450, .64}, {0.475, .67}, {0.500, .72},
            {0.525, .73}, {0.550, .73}, {0.575, .75}, {0.600, .77}, {0.625, .78},
            {0.650, .79}, {0.675, .79}, {0.700, .80}, {0.725, .80}, {0.750, .80},
            {0.775, .81}, {0.800, .80}, {0.825, .80}, {0.850, .79}, {0.875, .79},
            {0.900, .78}, {0.925, .76}, {0.950, .75}, {0.975, .72}, {1.000, .70}};
```

Convert wavelength domain to -1.0 to +1.0.

```
In[32]:= dataWindowsF =
    Table[{(dataWindows[[i, 1]] - 0.7)/0.3, dataWindows[[i, 2]]}, {i, Length[dataWindows]}];
```

Fit data to Legendre polynomial series and print coefficients.

```
In[33]:= funcWindows =
    Fit[dataWindowsF, {1, x, LegendreP[2, x], LegendreP[3, x], LegendreP[4, x]}, x];
  Print[Table[If[i == 1, funcWindows[[1]], funcWindows[[i, 1]]], {i, 5}]];

  {0.749539, 0.0629847, -0.110047, 0.0141777, -0.0209327}
```

■ Telescope and Transport Optics Transmission

This section gives the transmission through the telescope and through the transport optics from the beam splitters to the 4 photodetector pairs as a function of wavelength. The data were provided by Lynn Huff. The format of the data is wavelength in nm, followed by four transmission coefficients in the following order: TRE-A X, TRE-A Y, TRE-B X, TRE-B Y.

```
In[35]:= dataTelChanIn =
    {{0.425, .078/2, .098/2, .064/2, .119/2}, {0.475, .073/2, .088/2, .062/2, .104/2},
     {0.525, .076/2, .094/2, .070/2, .103/2}, {0.575, .072/2, .095/2, .073/2, .094/2},
     {0.625, .062/2, .088/2, .068/2, .080/2}, {0.675, .050/2, .082/2, .061/2, .068/2},
     {0.725, .042/2, .080/2, .051/2, .066/2}, {0.775, .039/2, .079/2, .046/2, .067/2},
     {0.825, .039/2, .081/2, .042/2, .075/2}, {0.875, .043/2, .083/2, .043/2, .082/2},
     {0.925, .057/2, .079/2, .051/2, .088/2}, {0.975, .068/2, .068/2, .056/2, .083/2}};
  dataTelChan = Table[Table[{dataTelChanIn[[i, 1]], dataTelChanIn[[i, j + 1]]},
      {i, Length[dataTelChanIn]}], {j, 4}];
```

Change wavelength domain to -1.0 to +1.0, fit data to a Legendre polynomial series, and print the four sets of coefficients.

```
In[37]:= dataTelChanF = Table[Table[{(dataTelChan[[j, i, 1]] - 0.7)/0.3, dataTelChan[[j, i, 2]]},
      {i, Length[dataTelChan[[1]]]}], {j, 4}];
  funcTelChan = Table[Fit[dataTelChanF[[i]], {1, x, LegendreP[2, x], LegendreP[3, x],
       LegendreP[4, x], LegendreP[5, x], LegendreP[6, x]}, x], {i, 4}];
  Do[Print[Table[If[i == 1, funcTelChan[[j, 1]], funcTelChan[[j, i, 1]]], {i, 7}]], {j, 4}];

  {0.0292101, -0.00762134, 0.0103442, 0.00899648, -0.0006824, -0.00397466, 0.00246417}

  {0.0423798, -0.00647787, 0.000392022, -0.00196364, -0.00183847, -0.00506374, 0.00466432}

  {0.0286861, -0.00645668, 0.00140041, 0.00804512, 0.00155598, -0.00529179, 0.00166112}

  {0.0429687, -0.00889728, 0.0130011, -0.000583278, -0.00547864, -0.00464486, 0.00481833}
```

■ Photodetector Quantum Efficiency

This section gives the quantum efficiency of the photodetectors. The source data is typical data for the blue enhanced photodiodes from Centronic Inc.

```
In[40]:= dataQE = {{0.400, 0.356}, {0.450, 0.546}, {0.500, 0.637}, {0.550, 0.681},
            {0.600, 0.709}, {0.650, 0.723}, {0.700, 0.735}, {0.750, 0.742}, {0.800, 0.749},
            {0.850, 0.751}, {0.900, 0.754}, {0.950, 0.756}, {1.000, 0.697}};
```

Change wavelength domain to -1.0 to +1.0, fit data to Legendre polynomial series, and print coefficients.

```
In[41]:= dataQEF = Table[{(dataQE[[i, 1]] - 0.7)/0.3, dataQE[[i, 2]]}, {i, Length[dataQE]}];
         funcQE = Fit[dataQEF, {1, x, LegendreP[2, x],
             LegendreP[3, x], LegendreP[4, x], LegendreP[5, x], LegendreP[6, x]}, x];
         Print[Table[If[i == 1, funcQE[[1]], funcQE[[i, 1]]], {i, 7}]];

        {0.695007, 0.1139, -0.106033, 0.0510264, -0.0458113, 0.00610627, -0.0164247}
```

■ Calculate Linearity for ± 100 marcsec and ± 200 marcsec

■ Calculate Scale Factors and Place in a Table

Define Table of Scale Factors. The first index is the frequency, and the second

```
In[44]:= tableSF = Table[{0.1 (3 + i), Table[0.0, {j, -20, 20}]}, {i, 7}];
```

Establish number of wavelengths kN, number of de-focus conditions jN, number of points

```
In[45]:= kN = kU - kL + 1 (* number of wavelengths *);
         If[kN < 7 || kN > 7, Print["Number of Wavelengths not equal to 7"]; Exit[]];
         jN = jU - jL + 1 (* number of de-focus conditions *);
         If[jN < 1 || jN > 1, Print["Number of De-focus conditions not equal to 1"]; Exit[]];
         iNF = 2 iN + 1 (* number of points *);
```

Fill tableSF with scale factors.

```
In[50]:= Do[ampT = dIn[[4 + k, 1]];
           Do[ampA = Table[ampT[[j, i]], {i, iNF}];
            tableSF[[k, 2, j]] = sf[iNF, (3 + k) dLam dR / (0.668 10^-6), ampA], {j, 41}], {k, kN}];
```

■ Find Average Scale Factor Across Band as a Function of Angle

In[51]:=
```
tableSFAv = Table[{i, 0.0}, {i, 4}, {j, 41}];
tableSFTmp = Table[{(0.1 (3 + k) - 0.7)/0.3, 0.0}, {k, kN}];
```

In[53]:=
```
Do[
 Do[tableSFTmp[[k, 2]] = tableSF[[k, 2, j]], {k, kN}];
 cc = Fit[tableSFTmp, {1, LegendreP[1, x], LegendreP[2, x], LegendreP[3, x]}, x];
 Do[tableSFAv[[i, j]] = (∫_{-1}^{1} cc (x + 7/3) funcHR8703 funcWindows funcTelChan[[i]] funcQE dx) /
   (∫_{-1}^{1} (x + 7/3) funcHR8703 funcWindows funcTelChan[[i]] funcQE dx), {i, 4}], {j, 41}]
```

In[54]:=
```
angleS = Table[Table[{10.0 (m - 21), 0.0}, {m, 41}], {i, 4}];
```

In[55]:=
```
Do[
  angleS[[i, 21, 2]] = 0.0;
  Do[angleS[[i, 21 + m, 2]] =
    angleS[[i, 20 + m, 2]] + 5 (tableSFAv[[i, 20 + m]] + tableSFAv[[i, 21 + m]]);
   angleS[[i, 21 - m, 2]] = angleS[[i, 22 - m, 2]] -
     5 (tableSFAv[[i, 22 - m]] + tableSFAv[[i, 21 - m]]), {m, 20}], {i, 4}];
```

■ Calculate Linearity for ±100 marcs

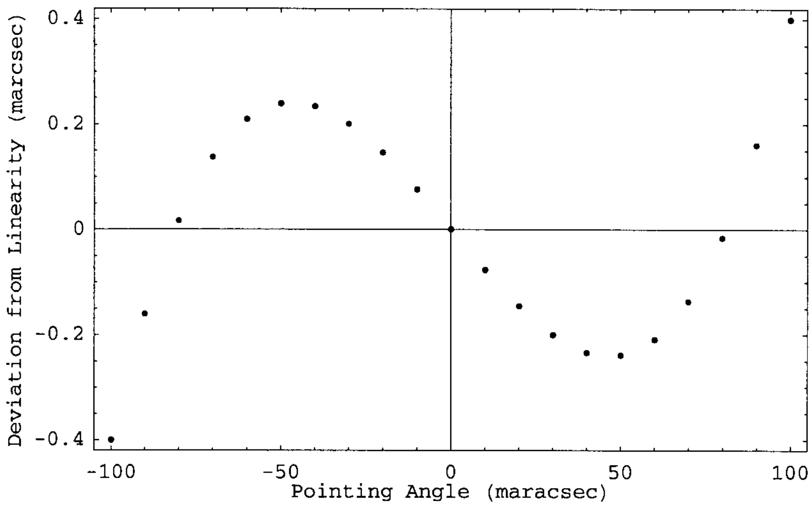
In[56]:=
```
Do[
 angleS100 = Table[{angleS[[i, j, 1]], angleS[[i, j, 2]]}, {i, 4}, {j, 11, 31}];
 coef = Fit[angleS100[[i]], {x}, {x}];
 fitAngle = Table[{angleS100[[i, m, 1]], angleS100[[i, m, 2]]/coef[[1]]}, {m, 21}];
 difFitAngle =
  Table[{angleS100[[i, m, 1]], angleS100[[i, m, 1]] - fitAngle[[m, 2]]}, {m, 21}];
 Print["Detector ID = ", i] ;
 Print["Scale Factor = ", tableSFAv[[i, 21]]];
 Print["Deviation at -100 marcs = ", difFitAngle[[1, 2]]];
Print["Deviation at +100 marcs = ", difFitAngle[[21, 2]]];
ListPlot[difFitAngle, Frame → True, FrameLabel →
   {"Pointing Angle (maracsec)", "Deviation from Linearity (marcsec)"}], {i, 4}]
```

```
Detector ID = 1

Scale Factor = 1.26647

Deviation at -100 marcs = -0.399809

Deviation at +100 marcs = 0.399809
```

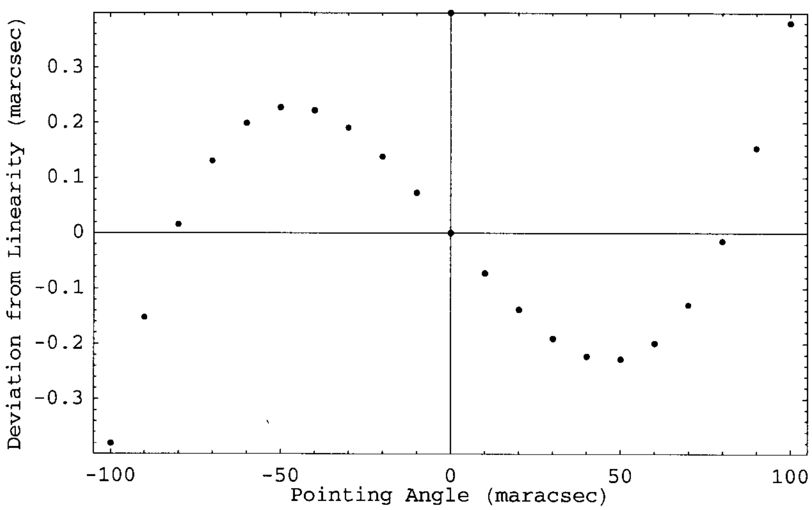


Detector ID = 2

Scale Factor = 1.24718

Deviation at -100 marcs = -0.380247

Deviation at +100 marcs = 0.380247

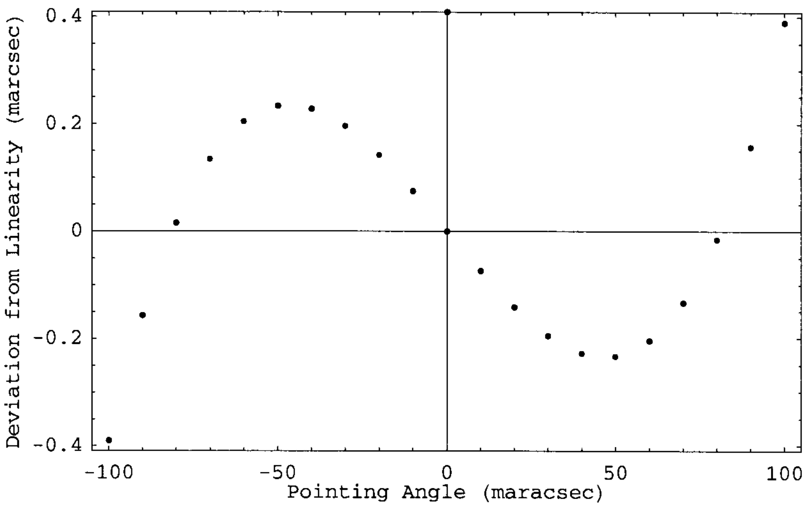


Detector ID = 3

Scale Factor = 1.26547

Deviation at -100 marcs = -0.389964

Deviation at +100 marcs = 0.389964

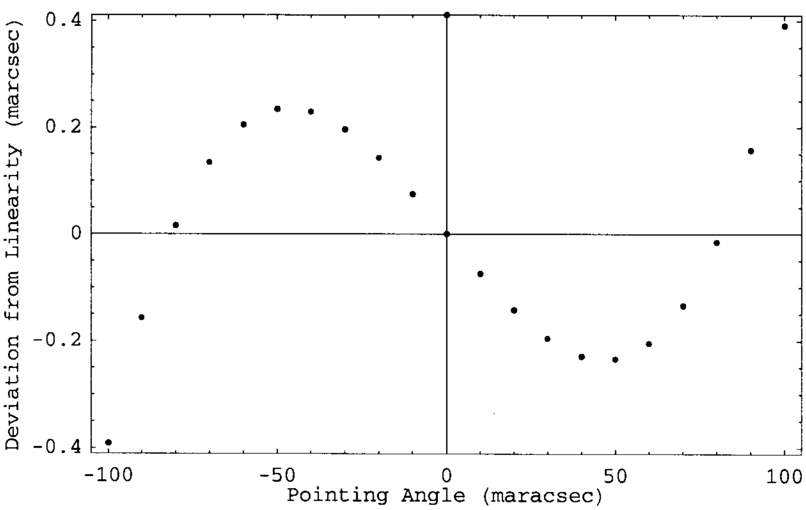


Detector ID = 4

Scale Factor = 1.25047

Deviation at -100 marcs = -0.391062

Deviation at +100 marcs = 0.391062



▪ Calculate Linearity for ±200 marcs

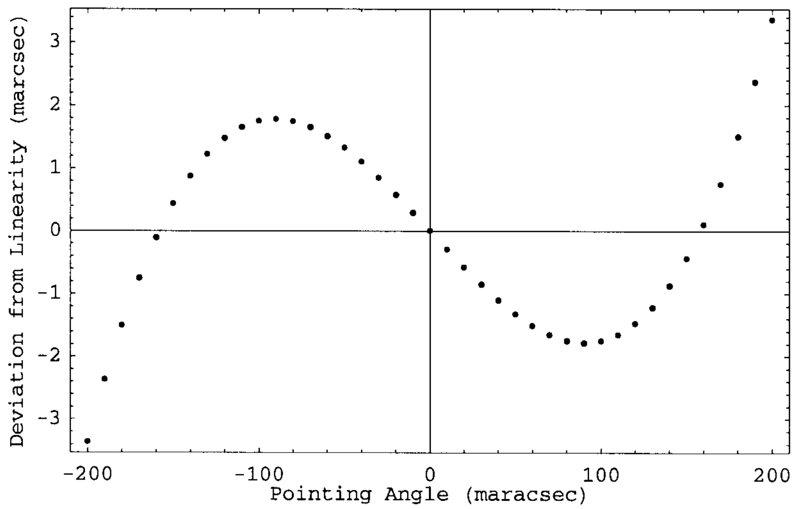
```
In[57]:= Do[
   coef = Fit[angleS[[i]], {x}, {x}];
   fitAngle = Table[{angleS[[i, m, 1]], angleS[[i, m, 2]] / coef[[1]]}, {m, 41}];
   difFitAngle = Table[{angleS[[i, m, 1]], angleS[[i, m, 1]] - fitAngle[[m, 2]]}, {m, 41}];
   Print["Detector ID = ", i] ;
   Print["Scale Factor = ", tableSFAv[[i, 21]]];
   Print["Deviation at -200 marcs = ", difFitAngle[[1, 2]]];
  Print["Deviation at +200 marcs = ", difFitAngle[[41, 2]]];
  ListPlot[difFitAngle, Frame → True, FrameLabel →
     {"Pointing Angle (maracsec)", "Deviation from Linearity (marcsec)"}], {i, 4}]
```

```
Detector ID = 1

Scale Factor = 1.26647

Deviation at -200 marcs = -3.35936

Deviation at +200 marcs = 3.35936
```

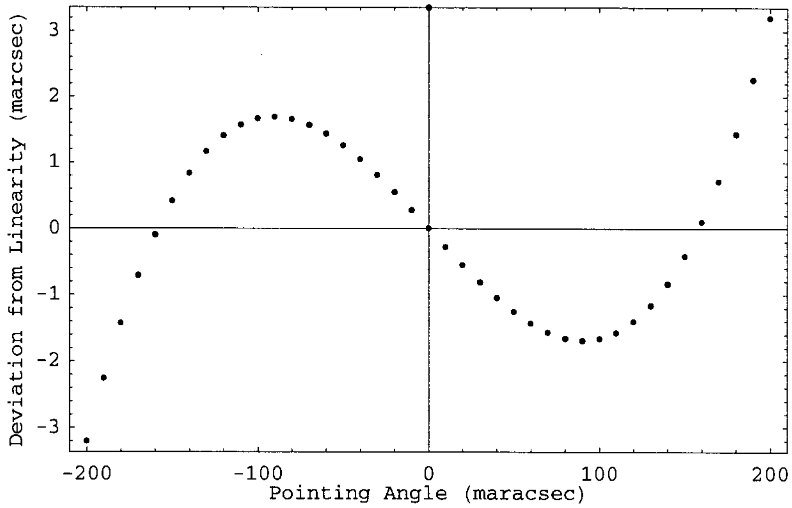



```
Detector ID = 2

Scale Factor = 1.24718

Deviation at -200 marcs = -3.20164

Deviation at +200 marcs = 3.20164
```

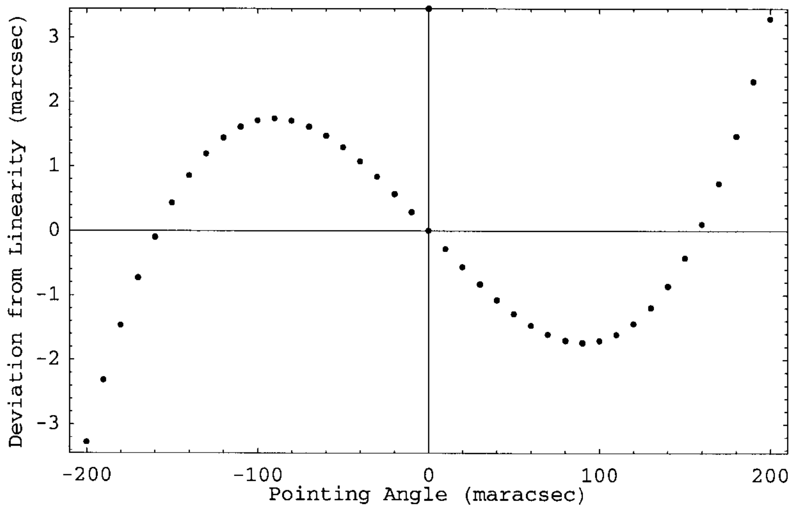


Detector ID = 3

Scale Factor = 1.26547

Deviation at -200 marcs = -3.28465

Deviation at +200 marcs = 3.28465

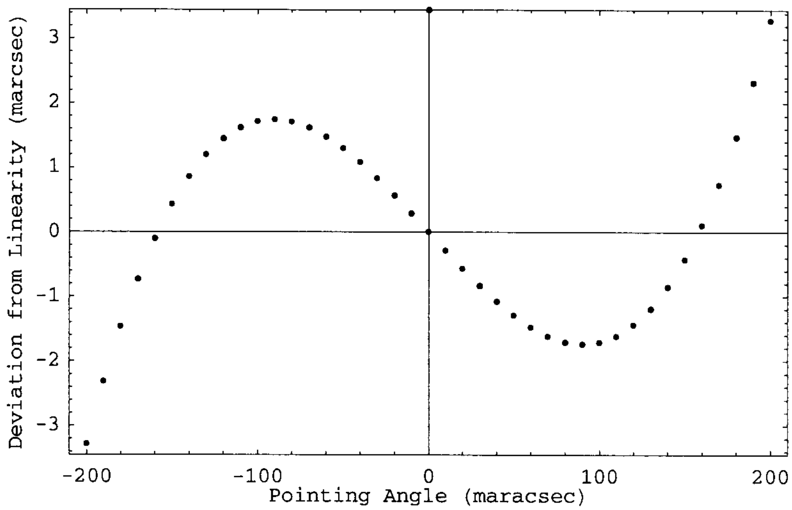


Detector ID = 4

Scale Factor = 1.25047

Deviation at -200 marcs = -3.28457

Deviation at +200 marcs = 3.28457


Deviation from Linearity (marcsec)
3
2
1
0
-1
-2
-3
-200
-100
0
100
200
Pointing Angle (maracsec)

Appendix 2

Listing of Fortran Program used for Calculation of Amplitudes at Knife-Edge Plane for Aberrations Characterized by Modified Zernike Polynomials up to Order 8

```
C Integration Routine for Telescope Model that incorporates
C any abberation term up to n=8
C
C This program is designed to give output at increments of
C 10 marcsec between -200 marcsec and 200 marcsec to
C investigate the linearity
C
C MODIFIED 3 MARCH 2002.
C
C DECLARATIONS
C
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 AMP(-500:500,-20:20),LAM,R1,THETA1,WAVES(101),X1,Y1
      INTEGER I,II,J,K,L,M,M1,MN
      LOGICAL LX,LY,LZ
      REAL*8 AC(0:8,0:8),AS(0:8,0:8)
      COMMON /ZCOEF/AC,AS
      REAL*8 V1,V2,V3,V4,Z1
      COMMON /VALUES/V1,V2,V3,V4,Z1
      REAL*8 ZC(0:8,0:8),ZS(0:8,0:8)
      COMMON /PCOEF/ZC,ZS
      CHARACTER FLNM*11,SUBDIR*80,FULLNM*80
      COMMON /NAMES/FLNM,SUBDIR,FULLNM
      INTEGER IN,KL,KU,JL,JU,JN
      REAL*8 STEPR,STEPJ,STEPK,PI,LAM0
      COMMON /INVAL/IN,KL,KU,JL,JU,JN,STEPR,STEPJ,STEPK,PI,LAM0
      REAL*8 F,B,A,E,THETAD,THETA0,THETAP
      COMMON /TELE/F,B,A,E,THETAD,THETA0,THETAP
      EXTERNAL FUNC1,FUNC2
C
C CODE SECTION
C
C SET VALUE OF MN, NORMALLY 20
      MN=20
C OPEN FILE FOR INPUT
C
      OPEN(4,FILE='C:\Telescope\Linearity\Input.txt')
C
C INITIALIZE AND READ INPUT
C
10    CALL INIT(LX)
      IF(.NOT.LX) GOTO 6000
C
C WRITE INPUT DATA TO FILE
C
C Write initial left brace { into output file
      WRITE(1,'(1H{)')
C Write namelist data into output file
      WRITE(1,'(1H{,F7.2,2H, ,I4,1H,,F11.7,1H,,I4,1H,,I4,1H,,F7.3,
     2 1H,,I4,1H,,I4,1H,,F14.9,2H},)') THETAD,IN,STEPR,JL,JU,
     3 STEPJ,KL,KU,STEPK
      CALL OUTB(AC)
      CALL OUTB(AS)
C Write WAVES vector into output file
      DO 20 J=1,JN
      WAVES(J)=STEPJ*(JL+J-1)
20    CONTINUE
      WRITE(1,'(1H{,\)')
      DO 30 J=1,JN-1
30    WRITE(1,'(F7.3,1H,,\)') WAVES(J)
      WRITE(1,'(F7.3,2H},)') WAVES(JN)
C
C SET LY TO .TRUE. IF AC & AS ARE NON-ZERO FOR ONLY EVEN N
C
      LY=.TRUE.
      DO 40 I=1,7,2
      DO 40 J=1,7,2
      IF(AC(I,J).NE.0.0D0.OR.AS(I,J).NE.0) LY=.FALSE.
40    CONTINUE
C
C SET LZ TO .TRUE. IF AC & AS ARE NON-ZERO FOR ONLY N=0
C
      LZ=.TRUE.
      DO 50 I=1,8
      DO 50 J=1,8
      IF(AC(I,J).NE.0.0D0.OR.AS(I,J).NE.0) LZ=.FALSE.
```

```
50     CONTINUE
C
C START CALCULATION LOOPS
C
C Start loop for wavelengths
       DO 5000 K=KL,KU
        LAM=K*STEPK
C
C Start loop for decollimation at STEPJ wave steps
        DO 4000 J=1,JN
         Z1=STEPJ*(JL+J-1)*2*F**2*LAM0/B**2
C
C Start loop for steps of 10 marcsec from -200 to + 200 marcsec
         DO 3000 M=0,MN
          WRITE(*,'(1H ,3I5)') K,J,M
          X1=10*PI*F*M/(60*60*180*1000)
          DO 2000 M1=1,2
           IF(M1.EQ.2.AND.M.EQ.0) GOTO 2000
           IF(M1.EQ.2.AND.LZ) GOTO 2000
           IF(M1.EQ.2) X1=-X1
C
C Start loop for IN evaluations at STEPR radial steps
           DO 1000 I=0,IN
            Y1=(LAM/LAM0)*STEPR*I
            R1=SQRT(Y1**2+X1**2)
            IF(Y1.NE.0.0D0.AND.X1.NE.0.0D0) THETA1=ATAN2(Y1,X1)
            IF(Y1.EQ.0.0D0.AND.X1.EQ.0.0D0) THETA1=0.0
C Define quantities that will be used to optimize calculation speed
            V1=1/(F*LAM)
            V2=2*PI/LAM
            V3=(F-Z1)/(2*F**3)
            V4=2*F*R1
C
C Calculate coefficients for powers of R and Cos/Sin(n*theta)
            CALL SETRS
C
C Logic for calculating amplitude at +R and -R
            DO 500 L=1,2
             IF(L.EQ.2.AND.I.EQ.0) GOTO 500
C SKIP INTEGRATION IF THERE ARE ONLY ABERRATIONS WITH N EVEN
             IF(L.EQ.2.AND.LY) GOTO 500
             THETAP=THETA0+THETA1
             IF(L.EQ.2) THETAP=PI+THETA0+THETA1
C Evaluate integrals integrals and assign to amplitude vectors
             CALL QROMB1(FUNC1,-PI,PI,SC)
             CALL QROMB1(FUNC2,-PI,PI,SS)
C
C Calculate absolute value of amplitude
             AMPC=SC-PI*(B**2-A**2)/(F*LAM)
             AMPS=SS-PI*(B**2-A**2)/(F*LAM)
             AMPM=SQRT(AMPC**2+AMPS**2)
C Assign for case of M1=1 (Positive X1)
             IF(L.EQ.1.AND.M1.EQ.1) AMP(I,M)=AMPM
             IF(L.EQ.2.AND.M1.EQ.1) AMP(-I,-M)=AMPM
C Assign for case of M1=2 (Negative X1)
             IF(L.EQ.1.AND.M1.EQ.2) AMP(I,-M)=AMPM
             IF(L.EQ.2.AND.M1.EQ.2) AMP(-I,M)=AMPM
500         CONTINUE
C
1000       CONTINUE
C
C ASSIGN AMPLITUDES FOR INVERSION SYMMETRY LY=.TRUE.
           IF(.NOT.LY) GOTO 2000
           DO 1100 II=1,IN
            AMP(-II,-M)=AMP(II,M)
            AMP(-II,M)=AMP(II,-M)
1100       CONTINUE
2000      CONTINUE
C ASSIGN AMPLITUDES FOR AXIAL SYMMETRY LZ=TRUE.
          IF(.NOT.LZ) GOTO 3000
          AMP(0,-M)=AMP(0,M)
          DO 2100 II=1,IN
           AMP(II,-M)=AMP(II,M)
           AMP(-II,M)=AMP(II,M)
2100      CONTINUE
3000     CONTINUE
```

```
C
C Write absolute amplitude to file
        IF(J.EQ.1) WRITE(1,'(1H{)')
        CALL OUTA(AMP,IN,MN)
        IF(J.LT.JN) WRITE(1,'(1H,)')
        IF(J.EQ.JN.AND.K.LT.KU) WRITE(1,'(2H},)')
        IF(J.EQ.JN.AND.K.EQ.KU) WRITE(1,'(1H})')
C
4000   CONTINUE
5000  CONTINUE
C
C End of loops
C
C Write final right brace } into output file
      WRITE(1,'(1H})')
      CLOSE(1)
      GO TO 10
C
6000  CONTINUE
      STOP
      END
C
C
      SUBROUTINE SETRS
C
C DECLARATIONS
C
      REAL*8 B2,B3,B4,B5,B6,B7,B8
      REAL*8 E2,E4,E6,E8,E10,E12,E2M1,E2M2,E2M3,E2M4
      REAL*8 ZC(0:8,0:8),ZS(0:8,0:8)
      COMMON /PCOEF/ZC,ZS
      REAL*8 F,B,A,E,THETAD,THETA0,THETAP
      COMMON /TELE/F,B,A,E,THETAD,THETA0,THETAP
      INTEGER IN,KL,KU,JL,JU,JN
      REAL*8 STEPR,STEPJ,STEPK,PI,LAM0
      COMMON /INVAL/IN,KL,KU,JL,JU,JN,STEPR,STEPJ,STEPK,PI,LAM0
      REAL*8 AC(0:8,0:8),AS(0:8,0:8)
      COMMON /ZCOEF/AC,AS
C
C CODE
C
C Evaluate powers of B
        B2=B*B
        B3=B2*B
        B4=B3*B
        B5=B4*B
        B6=B5*B
        B7=B6*B
        B8=B7*B
C
C Evaluate powers of even powers of E
        E2=E*E
        E4=E2*E2
        E6=E4*E2
        E8=E6*E2
        E10=E8*E2
        E12=E10*E2
        E14=E12*E2
        E16=E14*E2
        E18=E16*E2
        E20=E18*E2
C Evaluate powers of (1-E*E)
        E2M1=(1-E2)
        E2M2=E2M1*E2M1
        E2M3=E2M2*E2M1
        E2M4=E2M3*E2M1
C
C Evaluate coefficients for powers of R and powers of Cos/Sin
C
C m = 0
        ZC(0,0)=LAM0*(
     2     AC(4,0)*(1+4*E2+E4)/E2M2
     3    +AC(6,0)*(1+9*E2+9*E4+E6)/E2M3
     4    +AC(8,0)*(1+16*E2+36*E4+16*E6+E8)/E2M4)
        ZC(2,0)=LAM0*(
     2     -6*AC(4,0)*(1+E2)/E2M2
```

```
     3    +12*AC(6,0)*(1+3*E2+E4)/E2M3
     4    -20*AC(8,0)*(1+6*E2+6*E4+E6)/E2M4)/B2
      ZC(4,0)=LAM0*(
     2      6*AC(4,0)/E2M2
     3    -30*AC(6,0)*(1+E2)/E2M3
     4    +30*AC(8,0)*(3+8*E2+3*E4)/E2M4)/B4
      ZC(6,0)=LAM0*(
     2      20*AC(6,0)/E2M3
     3    -140*AC(8,0)*(1+E2)/E2M4)/B6
      ZC(8,0)=LAM0*70*AC(8,0)/(E2M4*B8)
C
C m = 1
      ZC(1,1)=LAM0*(
     2       AC(1,1)
     3    -2*AC(3,1)*(1+E2+E4)/(1+E2-2*E4)
     4    +3*AC(5,1)*(1+4*E2+10*E4+4*E6+E8)/(1+4*E2-8*E4+3*E8)
     5    -4*AC(7,1)*(1+9*E2+45*E4+65*E6+45*E8+9*E10+E12)/
     6      (E2M3*(1+12*E2+18*E4+4*E6)))/B
      ZS(1,1)=LAM0*(
     2       AS(1,1)
     3    -2*AS(3,1)*(1+E2+E4)/(1+E2-2*E4)
     4    +3*AS(5,1)*(1+4*E2+10*E4+4*E6+E8)/(1+4*E2-8*E4+3*E8)
     5    -4*AS(7,1)*(1+9*E2+45*E4+65*E6+45*E8+9*E10+E12)/
     6      (E2M3*(1+12*E2+18*E4+4*E6)))/B
      ZC(3,1)=LAM0*(
     2      3*AC(3,1)*(1+E2)/(1+E2-2*E4)
     3    -12*AC(5,1)*(1+4*E2+4*E4+E6)/(1+4*E2-8*E4+3*E8)
     4    +30*AC(7,1)*(1+E2)*(1+3*E2+E4)*(1+5*E2+E4)/
     5      (E2M3*(1+12*E2+18*E4+4*E6)))/B3
      ZS(3,1)=LAM0*(
     2      3*AS(3,1)*(1+E2)/(1+E2-2*E4)
     3    -12*AS(5,1)*(1+4*E2+4*E4+E6)/(1+4*E2-8*E4+3*E8)
     4    +30*AS(7,1)*(1+E2)*(1+3*E2+E4)*(1+5*E2+E4)/
     5      (E2M3*(1+12*E2+18*E4+4*E6)))/B3
      ZC(5,1)=LAM0*(
     2     10*AC(5,1)*(1+4*E2+E4)/(1+4*E2-8*E4+3*E8)
     3    -60*AC(7,1)*(1+9*E2+15*E4+9*E6+E8)/
     4      ((E2M3)*(1+12*E2+18*E4+4*E6)))/B5
      ZS(5,1)=LAM0*(
     2     10*AS(5,1)*(1+4*E2+E4)/(1+4*E2-8*E4+3*E8)
     3    -60*AS(7,1)*(1+9*E2+15*E4+9*E6+E8)/
     4      ((E2M3)*(1+12*E2+18*E4+4*E6)))/B5
      ZC(7,1)=LAM0*(35*AC(7,1)*(1+9*E2+9*E4+E6)
     2    /(E2M3*(1+12*E2+18*E4+4*E6)))/B7
      ZS(7,1)=LAM0*(35*AS(7,1)*(1+9*E2+9*E4+E6)
     2    /(E2M3*(1+12*E2+18*E4+4*E6)))/B7
C
C m = 2
    ZC(2,2)=LAM0*(
     2      AC(2,2)
     3   -3*AC(4,2)*(1+E2)*(1+E4)/(E2M2*(1+2*E2+3*E4))
     4   +6*AC(6,2)*(1+3*E2+E4)*(1+E2+6*E4+E6+E8)/
     5    (E2M2*(1+6*E2+21*E4+16*E6+6*E8))
     6  -10*AC(8,2)*(1+E2)*(1+5*E2+E4)*
     7    (1+3*E2+21*E4+20*E6+21*E8+3*E10+E12)/
     8    (E2M3*(1+12*E2+78*E4+164*E6+165*E8+60*E10+10*E12)))/B2
    ZS(2,2)=LAM0*(
     2      AS(2,2)
     3   -3*AS(4,2)*(1+E2)*(1+E4)/(E2M2*(1+2*E2+3*E4))
     4   +6*AS(6,2)*(1+3*E2+E4)*(1+E2+6*E4+E6+E8)/
     5    (E2M2*(1+6*E2+21*E4+16*E6+6*E8))
     6  -10*AS(8,2)*(1+E2)*(1+5*E2+E4)*
     7    (1+3*E2+21*E4+20*E6+21*E8+3*E10+E12)/
     8    (E2M3*(1+12*E2+78*E4+164*E6+165*E8+60*E10+10*E12)))/B2
    ZC(4,2)=LAM0*(
     2    4*AC(4,2)*(1+E2+E4)/(E2M2*(1+2*E2+3*E4))
     3  -20*AC(6,2)*(1+E2)*(1+3*E2+7*E4+3*E6+E8)/
     4     (E2M2*(1+6*E2+21*E4+16*E6+6*E8))
     5  +60*AC(8,2)*(1+E2*(9+E2*(45+E2*(115+E2*(15+4*E2+E4)*
     6     (10+5*E2+E4)))))/(E2M3*
     7     (1+12*E2+78*E4+164*E6+165*E8+60*E10+10*E12)))/B4
    ZS(4,2)=LAM0*(
     2    4*AS(4,2)*(1+E2+E4)/(E2M2*(1+2*E2+3*E4))
     3  -20*AS(6,2)*(1+E2)*(1+3*E2+7*E4+3*E6+E8)/
     4     (E2M2*(1+6*E2+21*E4+16*E6+6*E8))
     5  +60*AS(8,2)*(1+E2*(9+E2*(45+E2*(115+E2*(15+4*E2+E4)*
```

```
     6     (10+5*E2+E4)))))/(E2M3*
     7     (1+12*E2+78*E4+164*E6+165*E8+60*E10+10*E12)))/B4
      ZC(6,2)=LAM0*(
     2    15*AC(6,2)*(1+4*E2+10*E4+4*E6+E8)/(E2M2*
     3       (1+6*E2+21*E4+16*E6+6*E8))
     4  -105*AC(8,2)*(1+E2)*(1+8*E2+37*E4+48*E6+37*E8+8*E10+E12)/
     5       (E2M3*(1+12*E2+78*E4+164*E6+165*E8+60*E10+10*E12)))/B6
      ZS(6,2)=LAM0*(
     2    15*AS(6,2)*(1+4*E2+10*E4+4*E6+E8)/(E2M2*
     3       (1+6*E2+21*E4+16*E6+6*E8))
     4  -105*AS(8,2)*(1+E2)*(1+8*E2+37*E4+48*E6+37*E8+8*E10+E12)/
     5       (E2M3*(1+12*E2+78*E4+164*E6+165*E8+60*E10+10*E12)))/B6
      ZC(8,2)=LAM0*(
     2  56*AC(8,2)*(1+9*E2+45*E4+65*E6+45*E8+9*E10+E12)/
     3     (E2M3*(1+12*E2+78*E4+164*E6+165*E8+60*E10+10*E12)))/B8
      ZS(8,2)=LAM0*(
     2  56*AS(8,2)*(1+9*E2+45*E4+65*E6+45*E8+9*E10+E12)/
     3     (E2M3*(1+12*E2+78*E4+164*E6+165*E8+60*E10+10*E12)))/B8
C
C m = 3
      ZC(3,3)=LAM0*(
     1       AC(3,3)
     2   -4*AC(5,3)*(1-E10)/(1-5*E8+4*E10)
     3  +10*AC(7,3)*(1+4*E2+10*E4+20*E6+35*E8+20*E10+10*E12+4*E14+E16)/
     4      (E2M2*(1+6*E2+21*E4+56*E6+51*E8+30*E10+10*E12)))/B3
      ZS(3,3)=LAM0*(
     1       AS(3,3)
     2   -4*AS(5,3)*(1-E10)/(1-5*E8+4*E10)
     3  +10*AS(7,3)*(1+4*E2+10*E4+20*E6+35*E8+20*E10+10*E12+4*E14+E16)/
     4      (E2M2*(1+6*E2+21*E4+56*E6+51*E8+30*E10+10*E12)))/B3
      ZC(5,3)=LAM0*(
     2    5*AC(5,3)*(1-E8)/(1-5*E8+4*E10)
     3  -30*AC(7,3)*(1+4*E2+10*E4+20*E6+20*E8+10*E10+4*E12+E14)/
     4      (E2M2*(1+6*E2+21*E4+56*E6+51*E8+30*E10+10*E12)))/B5
      ZS(5,3)=LAM0*(
     2    5*AS(5,3)*(1-E8)/(1-5*E8+4*E10)
     3  -30*AS(7,3)*(1+4*E2+10*E4+20*E6+20*E8+10*E10+4*E12+E14)/
     4      (E2M2*(1+6*E2+21*E4+56*E6+51*E8+30*E10+10*E12)))/B5
      ZC(7,3)=LAM0*(
     2  21*AC(7,3)*(1+4*E2+10*E4+20*E6+10*E8+4*E10+E12)/
     3     (E2M2*(1+6*E2+21*E4+56*E6+51*E8+30*E10+10*E12)))/B7
      ZS(7,3)=LAM0*(
     2  21*AS(7,3)*(1+4*E2+10*E4+20*E6+10*E8+4*E10+E12)/
     3     (E2M2*(1+6*E2+21*E4+56*E6+51*E8+30*E10+10*E12)))/B7
C
C m = 4
      ZC(4,4)=LAM0*(
     2       AC(4,4)
     3   -5*AC(6,4)*(1-E12)/(1-6*E10+5*E12)
     4  +15*AC(8,4)*
     5  (1+4*E2+10*E4+20*E6+35*E8+56*E10+35*E12+20*E14+10*E16+4*E18+E20)
     6  /(E2M2*
     7  (1+6*E2+21*E4+56*E6+126*E8+126*E10+91*E12+48*E14+15*E16)))/B4
      ZS(4,4)=LAM0*(
     2       AS(4,4)
     3   -5*AS(6,4)*(1-E12)/(1-6*E10+5*E12)
     4  +15*AS(8,4)*
     5  (1+4*E2+10*E4+20*E6+35*E8+56*E10+35*E12+20*E14+10*E16+4*E18+E20)
     6  /(E2M2*
     7  (1+6*E2+21*E4+56*E6+126*E8+126*E10+91*E12+48*E14+15*E16)))/B4
      ZC(6,4)=LAM0*(
     1    6*AC(6,4)*(1-E10)/(1-6*E10+5*E12)
     2  -42*AC(8,4)*
     3  (1+4*E2+10*E4+20*E6+35*E8+35*E10+20*E12+10*E14+4*E16+E18)/
     4  (E2M2*
     5  (1+6*E2+21*E4+56*E6+126*E8+126*E10+91*E12+48*E14+15*E16)))/B6
      ZS(6,4)=LAM0*(
     1    6*AS(6,4)*(1-E10)/(1-6*E10+5*E12)
     2  -42*AS(8,4)*
     3  (1+4*E2+10*E4+20*E6+35*E8+35*E10+20*E12+10*E14+4*E16+E18)/
     4  (E2M2*
     5  (1+6*E2+21*E4+56*E6+126*E8+126*E10+91*E12+48*E14+15*E16)))/B6
      ZC(8,4)=LAM0*(
     2  28*AC(8,4)*(1+4*E2+10*E4+20*E6+35*E8+20*E10+10*E12+4*E14+E16)/
     3  (E2M2*
     4  (1+6*E2+21*E4+56*E6+126*E8+126*E10+91*E12+48*E14+15*E16)))/B8
```

```
      ZS(8,4)=LAM0*(
     2  28*AS(8,4)*(1+4*E2+10*E4+20*E6+35*E8+20*E10+10*E12+4*E14+E16)/
     3  (E2M2*
     4  (1+6*E2+21*E4+56*E6+126*E8+126*E10+91*E12+48*E14+15*E16)))/B8
C
C m = 5
      ZC(5,5)=LAM0*(
     2     AC(5,5)
     3  -6*AC(7,5)*(1-E14)/(1-7*E12+6*E14))/B5
      ZS(5,5)=LAM0*(
     2     AS(5,5)
     3  -6*AS(7,5)*(1-E14)/(1-7*E12+6*E14))/B5
      ZC(7,5)=LAM0*(
     2  7*AC(7,5)*(1-E12)/(1-7*E12+6*E14))/B7
      ZS(7,5)=LAM0*(
     2  7*AS(7,5)*(1-E12)/(1-7*E12+6*E14))/B7
C
C m = 6
      ZC(6,6)=LAM0*(
     2     AC(6,6)
     3  -7*AC(8,6)*(1-E16)/(1-8*E14+7*E16))/B6
      ZS(6,6)=LAM0*(
     2     AS(6,6)
     3  -7*AS(8,6)*(1-E16)/(1-8*E14+7*E16))/B6
      ZC(8,6)=LAM0*(8*AC(8,6)*(1-E16)/(1-8*E14+7*E16))/B8
      ZS(8,6)=LAM0*(8*AS(8,6)*(1-E16)/(1-8*E14+7*E16))/B8
C
C m = 7
      ZC(7,7)=LAM0*AC(7,7)/B7
      ZS(7,7)=LAM0*AS(7,7)/B7
C
C m = 8
      ZC(8,8)=LAM0*AC(8,8)/B8
      ZS(8,8)=LAM0*AS(8,8)/B8
      RETURN
      END
C
C
      SUBROUTINE INIT(LX)
C DECLARATIONS
C
      LOGICAL LX
      INTEGER LS,LF
      REAL*8 AC(0:8,0:8),AS(0:8,0:8)
      COMMON /ZCOEF/AC,AS
      CHARACTER FLNM*11,SUBDIR*80,FULLNM*80
      COMMON /NAMES/FLNM,SUBDIR,FULLNM
      INTEGER IN,KL,KU,JL,JU,JN
      REAL*8 STEPR,STEPJ,STEPK,PI,LAM0
      COMMON /INVAL/IN,KL,KU,JL,JU,JN,STEPR,STEPJ,STEPK,PI,LAM0
      REAL*8 ZC(0:8,0:8),ZS(0:8,0:8)
      COMMON /PCOEF/ZC,ZS
      REAL*8 F,B,A,E,THETAD,THETA0,THETAP
      COMMON /TELE/F,B,A,E,THETAD,THETA0,THETAP
      NAMELIST /INPUT/FLNM,THETAD,IN,STEPR,JL,JU,STEPJ,KL,KU,STEPK,AC,AS
C
C CODE
C
      FULLNM=''
C Telescope properties
C
C F is the focal length of the telescope in m
      F=150*.0254D0
C B is the maximum telescope radius in m
      B=5.660D0*.0254D0/2
C A is the minimum telescope radius in m
      A=2.795D0*.0254D0/2
C E is the ratio of the minimum to maximum telescope radius
      E=A/B
C
C THETAD is the angle in degrees at which the calculations are made
C THETA0 is this angle converted to radians
C
      THETAD=0.0D0
C
C Initialize to standard values before NameList input
```

```
C
      SUBDIR='C:\Telescope\Linearity\Data\'
      FLNM='Test.txt'
      DO 10 J=0,8
      DO 10 I=0,8
      AC(I,J)=0.0D0
      AS(I,J)=0.0D0
      ZC(I,J)=0.0D0
      ZS(I,J)=0.0D0
10    CONTINUE
C KL and KU are the lower and upper bounds (integer) for wavelength
C with stepsize STEPK in units of m (LAM=K*STEPJ)
      KL=4
      KU=10
      STEPK=1.0D-7
C JL and JU are the lower and upper bounds (integer) for axial position
C stepsize STEPJ as measured in waves at LAM0=670 nm (WAVES=J*STEPJ)
      JL=-20
      JU=20
      JN=JU-JL+1
      STEPJ=0.1D0
C IN is the number of radial positions for stepsize STEPR in m (R=(IN-1)*STEPR)
      IN=201
      STEPR=0.000002D0
C Set value of PI
      PI=3.141592653589793D0
C LAM0 is the reference wavelength in m
      LAM0=6.68D-7
C
C READ NAMELIST INPUT
C
      READ(4,INPUT,END=200)
      THETA0=PI*THETAD/180
C
C Set file name and open output file
      LS=LEN_TRIM(SUBDIR)
      FULLNM(1:LS)=SUBDIR(1:LS)
      LF=LEN_TRIM(FLNM)
      FULLNM(LS+1:LS+LF+1)=FLNM(1:LF)
      OPEN(1,FILE=FULLNM)
      JN=JU-JL+1
      LX=.TRUE.
      RETURN
C
C End of input file
      LX=.FALSE.
      RETURN
      END
C
C
      SUBROUTINE OUTB(AMP)
      REAL*8 AMP(0:8,0:8)
      WRITE(1,'(1H{)')
      DO 20 J=0,8
        WRITE(1,'(1H{,\)')
        DO 10 I=0,7
10        WRITE(1,'(F8.3,1H,\)') AMP(I,J)
        IF (J.EQ.8) THEN
          GOTO 30
          ENDIF
20      WRITE(1,'(F8.3,2H},)') AMP(8,J)
30    WRITE(1,'(F8.3,3H}},)') AMP(8,8)
      RETURN
      END
C
      SUBROUTINE OUTA(AMP,IN,MN)
      INTEGER IN,MN
      REAL*8 AMP(-500:500,-20:20)
      WRITE(1,'(1H{,\)')
      DO 20 M=-MN,MN
       WRITE(1,'(1H{,\)')
       DO 10 I=-IN,IN-1
        WRITE(1,'(F10.3,1H,\)') AMP(I,M)
10     CONTINUE
       IF(M.LT.MN) WRITE(1,'(F10.3,2H},)') AMP(IN,M)
20    CONTINUE
```

```
30    WRITE(1,'(F10.3,2H}})') AMP(IN,MN)
      RETURN
      END
C
C
      FUNCTION FUNCOS(R)
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 R2
      REAL*8 V1,V2,V3,V4,Z1
      COMMON /VALUES/V1,V2,V3,V4,Z1
      REAL*8 ZC(0:8,0:8),ZS(0:8,0:8)
      COMMON /PCOEF/ZC,ZS
      REAL*8 COST(8),SINT(8),COST1
      COMMON /SINES/COST,SINT,COST1
      R2=R*R
      FUNCOS=R*V1*(COS(V2*(-V3*(Z1*R+V4*COST1)*R
     2  +(((ZC(8,0)*R**2+ZC(6,0))*R**2+ZC(4,0))*R**2
     3  +ZC(2,0))*R**2+ZC(0,0)
     5  +((((ZC(7,1)*COST(1)+ZS(7,1)*SINT(1))*R2
     6     +(ZC(5,1)*COST(1)+ZS(5,1)*SINT(1)))*R2
     7     +(ZC(3,1)*COST(1)+ZS(3,1)*SINT(1)))*R2
     8     +(ZC(1,1)*COST(1)+ZS(1,1)*SINT(1)))*R
     9  +((((ZC(8,2)*COST(2)+ZS(8,2)*SINT(2))*R2
     1     +(ZC(6,2)*COST(2)+ZS(6,2)*SINT(2)))*R2
     1     +(ZC(4,2)*COST(2)+ZS(4,2)*SINT(2)))*R2
     2     +(ZC(2,2)*COST(2)+ZS(2,2)*SINT(2)))*R2
     3   +(((ZC(7,3)*COST(3)+ZS(7,3)*SINT(3))*R2
     4     +(ZC(5,3)*COST(3)+ZS(5,3)*SINT(3)))*R2
     5     +(ZC(3,3)*COST(3)+ZS(3,3)*SINT(3)))*R**3
     6   +(((ZC(8,4)*COST(4)+ZS(8,4)*SINT(4))*R2
     7     +(ZC(6,4)*COST(4)+ZS(6,4)*SINT(4)))*R2
     8     +(ZC(4,4)*COST(4)+ZS(4,4)*SINT(4)))*R**4
     9    +((ZC(7,5)*COST(5)+ZS(7,5)*SINT(5))*R2
     1     +(ZC(5,5)*COST(5)+ZS(5,5)*SINT(5)))*R**5
     1    +((ZC(8,6)*COST(6)+ZS(8,6)*SINT(6))*R2
     2     +(ZC(6,6)*COST(6)+ZS(6,6)*SINT(6)))*R**6
     3     +(ZC(7,7)*COST(7)+ZS(7,7)*SINT(7))*R**7
     4     +(ZC(8,8)*COST(8)+ZS(8,8)*SINT(8))*R**8))
     5  +1.0)
      RETURN
      END
C
C
      FUNCTION FUNSIN(R)
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 R2
      REAL*8 V1,V2,V3,V4,Z1
      COMMON /VALUES/V1,V2,V3,V4,Z1
      REAL*8 ZC(0:8,0:8),ZS(0:8,0:8)
      COMMON /PCOEF/ZC,ZS
      REAL*8 COST(8),SINT(8),COST1
      COMMON /SINES/COST,SINT,COST1
      R2=R*R
      FUNSIN=R*V1*(SIN(V2*(-V3*(Z1*R+V4*COST1)*R
     2  +(((ZC(8,0)*R**2+ZC(6,0))*R**2+ZC(4,0))*R**2
     3  +ZC(2,0))*R**2+ZC(0,0)
     5  +((((ZC(7,1)*COST(1)+ZS(7,1)*SINT(1))*R2
     6     +(ZC(5,1)*COST(1)+ZS(5,1)*SINT(1)))*R2
     7     +(ZC(3,1)*COST(1)+ZS(3,1)*SINT(1)))*R2
     8     +(ZC(1,1)*COST(1)+ZS(1,1)*SINT(1)))*R
     9  +((((ZC(8,2)*COST(2)+ZS(8,2)*SINT(2))*R2
     1     +(ZC(6,2)*COST(2)+ZS(6,2)*SINT(2)))*R2
     1     +(ZC(4,2)*COST(2)+ZS(4,2)*SINT(2)))*R2
     2     +(ZC(2,2)*COST(2)+ZS(2,2)*SINT(2)))*R2
     3   +(((ZC(7,3)*COST(3)+ZS(7,3)*SINT(3))*R2
     4     +(ZC(5,3)*COST(3)+ZS(5,3)*SINT(3)))*R2
     5     +(ZC(3,3)*COST(3)+ZS(3,3)*SINT(3)))*R**3
     6   +(((ZC(8,4)*COST(4)+ZS(8,4)*SINT(4))*R2
     7     +(ZC(6,4)*COST(4)+ZS(6,4)*SINT(4)))*R2
     8     +(ZC(4,4)*COST(4)+ZS(4,4)*SINT(4)))*R**4
     9    +((ZC(7,5)*COST(5)+ZS(7,5)*SINT(5))*R2
     1     +(ZC(5,5)*COST(5)+ZS(5,5)*SINT(5)))*R**5
     1    +((ZC(8,6)*COST(6)+ZS(8,6)*SINT(6))*R2
     2     +(ZC(6,6)*COST(6)+ZS(6,6)*SINT(6)))*R**6
     3     +(ZC(7,7)*COST(7)+ZS(7,7)*SINT(7))*R**7
     4     +(ZC(8,8)*COST(8)+ZS(8,8)*SINT(8))*R**8))
```

```
     5  +1.0)
      RETURN
      END
C
C
      SUBROUTINE qromb1(func,a,b,ss)
      IMPLICIT REAL*8(A-H,O-Z)
      INTEGER JMAX,JMAXP,K,KM
      REAL*8 a,b,func,ss,EPS
      EXTERNAL func
      PARAMETER (EPS=1.D-5, JMAX=20, JMAXP=JMAX+1, K=5, KM=K-1)
CU    USES polint,trapzd
      INTEGER j
      REAL*8 dss,h(JMAXP),s(JMAXP)
      h(1)=1.
      do 11 j=1,JMAX
        call trapzd1(func,a,b,s(j),j)
        if (j.ge.K) then
          call polint(h(j-KM),s(j-KM),K,0.D0,ss,dss)
          if (abs(dss).le.EPS*abs(ss).AND.J.GT.7) return
        endif
        s(j+1)=s(j)
        h(j+1)=0.25D0*h(j)
11    continue
      pause 'too many steps in qromb'
      END
C
C
      SUBROUTINE trapzd1(func,a,b,s,n)
      IMPLICIT REAL*8(A-H,O-Z)
      INTEGER n
      REAL*8 a,b,s,func
      EXTERNAL func
      INTEGER it,j
      REAL*8 del,sum,tnm,x
      if (n.eq.1) then
        s=0.5D0*(b-a)*(func(a)+func(b))
      else
        it=2**(n-2)
        tnm=it
        del=(b-a)/tnm
        x=a+0.5D0*del
        sum=0.
        do 11 j=1,it
          sum=sum+func(x)
          x=x+del
11      continue
        s=0.5D0*(s+(b-a)*sum/tnm)
      endif
      return
      END
C
C
      REAL*8 FUNCTION FUNC1(THETA)
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 F,B,A,E,THETAD,THETA0,THETAP
      COMMON /TELE/F,B,A,E,THETAD,THETA0,THETAP
      REAL*8 COST(8),SINT(8),COST1
      COMMON /SINES/COST,SINT,COST1
      EXTERNAL FUNCOS
      REAL*8 TMP
      COST1=COS(THETA-THETAP)
      DO 10 I=1,8
        COST(I)=COS(I*THETA)
        SINT(I)=SIN(I*THETA)
10    CONTINUE
      CALL qromb2(funcOS,a,b,TMP,J1)
      FUNC1=TMP
      RETURN
      END
C
C
      REAL*8 FUNCTION FUNC2(THETA)
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 F,B,A,E,THETAD,THETA0,THETAP
      COMMON /TELE/F,B,A,E,THETAD,THETA0,THETAP
```

```
      REAL*8 COST(8),SINT(8),COST1
      COMMON /SINES/COST,SINT,COST1
      EXTERNAL FUNSIN
      REAL*8 TMP
      COST1=COS(THETA-THETAP)
      DO 10 I=1,8
        COST(I)=COS(I*THETA)
        SINT(I)=SIN(I*THETA)
10    CONTINUE
      CALL qromb2(funsIN,a,b,TMP,J1)
      FUNC2=TMP
      RETURN
      END
C
C
      SUBROUTINE trapzd2(func,a,b,s,n)
      IMPLICIT REAL*8(A-H,O-Z)
      INTEGER n
      REAL*8 a,b,s,func
      EXTERNAL func
      INTEGER it,j
      REAL*8 del,sum,tnm,x
      if (n.eq.1) then
        s=0.5D0*(b-a)*(func(a)+func(b))
      else
        it=2**(n-2)
        tnm=it
        del=(b-a)/tnm
        x=a+0.5D0*del
        sum=0.
        do 11 j=1,it
          sum=sum+func(x)
          x=x+del
11      continue
        s=0.5D0*(s+(b-a)*sum/tnm)
      endif
      return
      END
C
C
      SUBROUTINE qromb2(func,a,b,ss,J1)
      IMPLICIT REAL*8(A-H,O-Z)
      INTEGER JMAX,JMAXP,K,KM,J1
      REAL*8 a,b,func,ss,EPS
      EXTERNAL func
      PARAMETER (EPS=1.D-6, JMAX=20, JMAXP=JMAX+1, K=5, KM=K-1)
CU    USES polint,trapzd
      INTEGER j
      REAL*8 dss,h(JMAXP),s(JMAXP)
      h(1)=1.
      do 11 j=1,JMAX
        call trapzd2(func,a,b,s(j),j)
        J1=J
        if (j.ge.K) then
          call polint(h(j-KM),s(j-KM),K,0.D0,ss,dss)
          if (abs(dss).le.EPS*abs(ss).AND.J.GT.7) return
        endif
        s(j+1)=s(j)
        h(j+1)=0.25D0*h(j)
11    continue
      pause 'too many steps in qromb'
      END
C
C
      SUBROUTINE polint(xa,ya,n,x,y,dy)
      IMPLICIT REAL*8(A-H,O-Z)
      INTEGER n,NMAX
      REAL*8 dy,x,y,xa(n),ya(n)
      PARAMETER (NMAX=10)
      INTEGER i,m,ns
      REAL*8 den,dif,dift,ho,hp,w,c(NMAX),d(NMAX)
      ns=1
      dif=abs(x-xa(1))
      do 11 i=1,n
        dift=abs(x-xa(i))
        if (dift.lt.dif) then
```

```
        ns=i
        dif=dift
      endif
      c(i)=ya(i)
      d(i)=ya(i)
11    continue
    y=ya(ns)
    ns=ns-1
    do 13 m=1,n-1
      do 12 i=1,n-m
        ho=xa(i)-x
        hp=xa(i+m)-x
        w=c(i+1)-d(i)

        den=ho-hp
        if(den.eq.0.)pause 'failure in polint'
        den=w/den
        d(i)=hp*den
        c(i)=ho*den
12      continue
      if (2*ns.lt.n-m)then
        dy=c(ns+1)
      else
        dy=d(ns)
        ns=ns-1
      endif
      y=y+dy
13    continue
    return
    END
```