LOST

Layer Oriented Simulation Tool

User Manual 2.1

July 21, 2004

Carmelo Arcidiacono, carmelo@arcetri.astro.it. *University of Florence*
Emiliano Diolaiti , diolaiti@pd.astro.it. *INAF- Astrophysical Observatory of Bologna*
Roberto Ragazzoni, ragazzoni@arcetri.astro.it, *INAF- Astrophysical Observatory of Arcetri and Max Planck Institut fur Astronomie (MPIA)*
Massimiliano Tordi, tordi@pd.astro.it, *University of Padova*
Elise Vernet, elise@arcetri.astro.it, *INAF- Astrophysical Observatory of Arcetri*
Jacopo Farinato, farinato@arcetri.astro.it, *INAF- Astrophysical Observatory of Arcetri*

*http://www.arcetri.astro.it/lost

TABLE OF CONTENTS

1 Acronyms

AD	Applicable Document
AO	Adaptive Optics
CCD	Charge Coupled Device
DM	Deformable Mirror
ELT	Extremely Large Telescope
ESO	European Southern Observatory
FFT	Fast Fourier Transform
FoV	Field of View
FWHM	Full Width at Half Maximum
GS	Guide Stars
LO	Layer Oriented
LBT	Large Binocular Telescope
LOST	Layer Oriented Simulation Tool
LOWFS	Layer Oriented Wavefront Sensor
MAD	Multi-Conjugate Adaptive Optics Demonstrator
MCAO	Multi-Conjugate Adaptive Optics
NGS	Natural Guide Star
OWL	OverWhelmingly Large Telescope
PS	Power Spectrum
PSF	Point Spread Function
RD	Referenced Document
RMS	root mean square
RON	Read Out Noise
SH	Shack-Hartmann
SHWFS	Shack-Hartmann Wavefront Sensor
VLT	Very Large Telescope
WFE	Wavefront Sensor Error-budget
WFS	Wavefront Sensor

2 Scope

The scope of this document is to describe the software code LOST (Layer Oriented Simulation Tool) used to simulate the **Layer-Oriented WaveFront Sensor** (LOWFS) channel of MAD on VLT and of LINC-NIRVANA on LBT.

3 Introduction

This simulation code has been developed to investigate the properties of the LOWFS system before its implementation on a real instrument. The actual software based on the first simulations done by Ragazzoni, Farinato & Marchetti (2000) and Tordi, Ragazzoni & Diolaiti (2001) is able to reproduce the main characteristics of either classical or MCAO using Shack-Hartmann (SH) or Pyramids WFSs. Reasonable simplifying assumptions have been done to limit the computation time allowing a wider study of the different configurations and an optimisation of peculiar parameters. After the description of the pyramid WFS and Layer-Oriented (LO) system, the different sub-systems of the simulation are listed in a second part before to present the structure of the code itself. Because of it is a very useful tool to study the MCAO system based on the LO approach we call it LOST: Layer Oriented Simulation Tool.

3.1 The Pyramid Wave Front Sensor

The concept of a Pyramid WFS is first described in Ragazzoni (1996). It offers the possibility to gain in sensitivity and to change the pupil sampling of the system. The core of this WFS is a pyramid pin placed at the focal plane of the telescope. The stellar light is split into four beams which are re-imaged in the pupil plane. This Foucault-like two dimensional knife-edge test allows to measure directly the wavefront aberrations in the pupil plane.

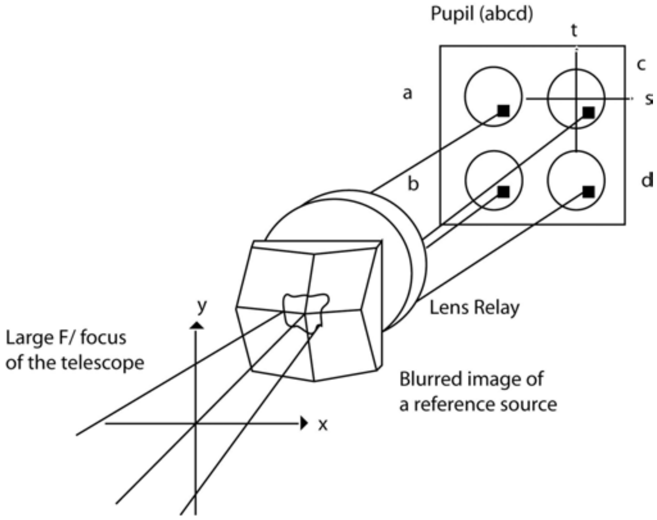


Figure 1: Layout of the Pyramid wavefront sensor.

The wavefront derivative is deduced from the signal intensity in the four pupils (*a, b, c* and *d* as shown in Figure 1). The WFS relies on the partial illumination of the four pupil images to be able to determine the

wavefront derivative. To enlighten the four pyramid faces different techniques have been proposed; a possible solution is to oscillate the pyramid (Ragazzoni,1996) but Esposito & Riccardi (2001) preferred to use a tip-tilt mirror in their experiment, method also used for the AdaOpt@TNG system of the Telescopio Nazionale Galileo at the Canary Island. Another way proposed recently (Ragazzoni, Diolaiti & Vernet, 2002) is to use a diffusing plate to enlarge the spot of the star without any modulation. It has also being shown recently both on the sky and by simulation that when measuring in the visible wavelength range and correcting in the near-infrared wavelength range the residual aberration is high enough to enlarge the spot of the star without any modulation or diffusing plate.
The pyramid WFS sensitivity has been widely studied in various regimes (Ragazzoni & Farinato (1999), Esposito & Riccardi (2001)) showing a gain in sensitivity respectively to the SHWFS in classical AO systems.

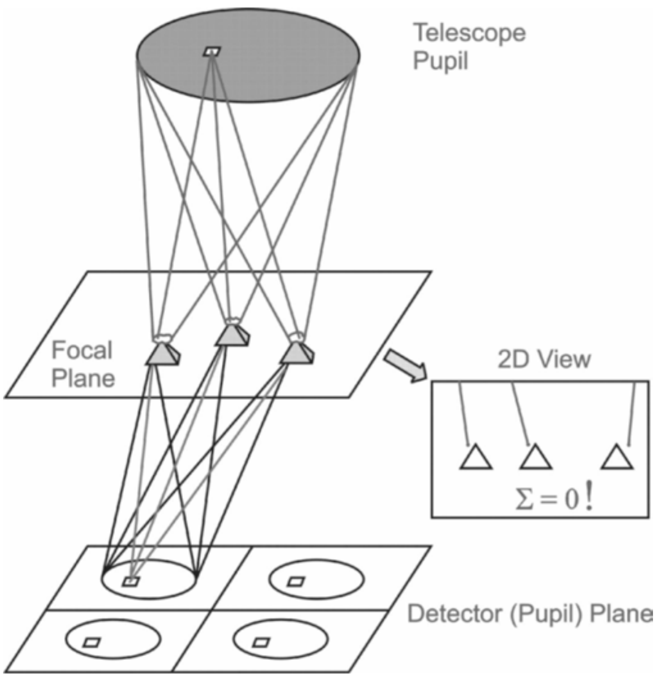
The realisation of pyramid pins is currently done by a few industries. The first pyramid used for the AdaOpt@TNG system has been manufactured at the Astronomical Observatory of Merate by removing material from a convex lens to reach a vertex angle of a few degree. The pyramid delivered have the specified roof size and turned edges but the technique does not permit a perfect repeatability in terms of vertex angles. To avoid such a problem several techniques are investigated to produce exact replica of a master (Ghigo et al., 2001)

3.2 The Layer-Oriented Wavefront Sensor: an MCAO WFS

MCAO is a technique described for the first time by Beckers (1989) to overcome the limitations of single-reference AO systems, namely limited sky coverage and residual anisoplanatism. The basic principle is to use more than one Deformable Mirror (DM) conjugated to different planes in the atmosphere and more than one Guide Star (GS) to measure the atmospheric turbulence in different directions, in order to reconstruct its 3-dimensional properties. In principle the DMs allow perfect correction of the turbulence in the conjugated planes, whereas the non-conjugated layers can be corrected only approximately, up to a given spatial frequency.

In 1999, Ragazzoni introduced a new concept for the WFS sensing in an MCAO system. The so-called “Layer-Oriented” system considers the overall information associated to a given altitude instead of using the stars information. It accomplishes optically an anamorphic copy of the atmosphere over the telescope aperture, where one can couple in a linear fashion, by optical means, the information on the wavefront perturbation coming from different reference sources. In this way the detector can be directly conjugated to a specific layer and gives the optimum information to drive such a deformable mirror, at least in a linear sense (although non-linear behavior are possible by specific 'tricks', see for instance Farinato et al., 2001). This can be obtained with any pupil plane WFS and, under some special circumstances, also by using non pupil plane WFSs, whereas the information is arranged in a pupil fashion, like it is in SH sensors. Finally, the LO approach can be accomplished also by numerical integration, loosing in this way some of the potentiality of the full optical concept (but, on the other hand, gaining some flexibility at the expense of a much larger real-time computational requirements).
The pyramid WFS is ideal to have a compact LO WFS system like the one for MAD. In this particular case, using 8 pyramid pins (as much as the number of reference stars) the light coming from each reference is collected by specific optical systems called star-enlargers which increase individually the stars dimensions without changing the distances. Each enlarged star image is then split by a pyramid and the four spots coming from the 8 pyramids are re-imaged by an objective on a CCD detector in case that the detector is conjugated to the pupil plane (i.e. focused on the ground) the 4 pupils coming from each pyramid are superimposed, thus increasing uniformly the SNR of the system, while if the detector plane is

conjugated to a different altitude, there is a partial overlapping of the pupil images coming from the different references, generating what is called "metapupil".



The shape of this metapupil depends upon the geometry of the considered asterism, i.e. the position of the references in the field the superimposition area also depends on the asterism geometry and additionally on the conjugation altitude (the higher the altitude the lower the superimposition).

4 Wavefront sensing simulation

We have seen previously that the layer oriented wavefront sensor is an instrument which benefits from the pyramid wavefront sensor properties to reproduce an anamorphic copy of the turbulent atmosphere in the optical system. The wavefront aberrations are directly obtained on the detector planes, each of them being optically conjugated to a given altitude. In the following sections we analyse the main structures composing the code: a description of the procedures to generate the phase screens is given in section 4.1, the details relative to the loop procedures are presented in section 4.2, and a description of the procedures generating the noise is described in sections 4.3, 4.4 before to give the loop procedure relative to the reconstruction and correction processes in section 4.5.

4.1 Layers Generation

As in most of the simulation codes we assume that the atmosphere introduced only phase variations. A realistic layer should present the same turbulence distribution theoretically described by Kolmogorov statistics (in terms of phase variance and isoplanatic patch size) and by Noll (1976) (in terms of distribution of the power in the Zernike modes).
Our procedures generate the phase screens by using a Kolmogorov power spectrum, *PS*, defined by the Wiener relation

Equation 1
$$PS = 0.023 r_0^{-5/3} k^{-11/3}$$

where r_0 is the Fried parameter and k is the spatial frequency. We obtain the phase screen with a Fast Fourier Transform (FFT) procedure using this *PS*, where to every frequency is associated a random value of the phase, with values between 0 and 2π.

This procedure presents, in the case of small layer size, a poor sampling for low frequency and it is necessary to add in the spectrum the frequencies that are initially excluded to reproduce the whole Kolmogorov power spectrum. These sub-harmonics are computed with the right amplitude, coming from Equation 1, and then are added to the layer.

The phase variance of the screen has to be normalized to the values predicted by Kolmogorov and Noll, which is expressed by the relation:

Equation 2

$$\sigma_\varphi^2 = 1.0299\left(\frac{D}{r_0}\right)^{5/3}$$

But in this case r_0 is defined in a more complex way using hyper-geometrical functions, see the reference (Winker, 1991)

In order to normalize the phase variance we compute the average value of the variance relative to pupils projected upon different positions on screen and dividing the layer by the square root of this value. After that we multiply for the value of the correct standard deviation computed using Equation 2, which is of course the average value. The distribution of the variance being not symmetric as shown in Figure 2, the average value of the variance inside different pupils taken on the screen is equal to one predicted by Noll but the median is smaller. The median and the average values of a distribution are more and more different as the deviation from symmetry increases.

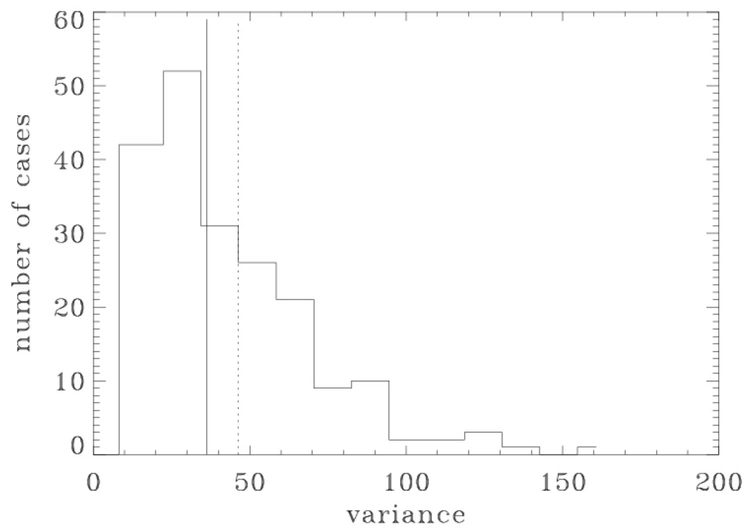


Figure 2. Distribution of the variances between 200 layers. The distribution is not symmetric, the solid line represents the median of the value while the dotted one is the average that, according to Noll, is equal to the value expected from equation 2.

In the simulation of the LO WFS for MAD we used a set of seven phase screens provided by ESO. They are produced by using FFT too, but with a Von Karman PS of 20 meters outer-scale.

4.2 Loop

The phase screens allow to compute the wavefront of each star for a given instant but if we integrate the wavefront on time scale larger than the atmosphere evolution, it must be taken into account by simulating the evolution of the phase screens for every temporal interval Δt. This user-defined parameter sets the lower temporal step of the simulation. Every other temporal parameter, as the integration time of the wavefront sensors and the delay applied to the DM, must be an integer multiple of this number. For every temporal step the code computes the measured wavefronts of the NGS and then uses it in LO mode to reconstruct the atmospheric layers at the conjugation the altitudes of the DMs (and WFSs). The following equation express the LO way to combine the measured wavefront:

Equation 3

$$M = \frac{\sum_{i=1}^{nstar} WF_i I_i}{\sum_{j=1}^{nstar} I_j}$$

where M is the weighted sum of the NGS wavefronts as seen focused to the conjugation altitude, WF_i is the i[th] NGS wavefront and I_j is the j[th] value of star intensity (in linear unit). All the quantities above are arrays.
The code is based on geometrical approximation of the projection of the GSs footprints on the phase screens and on the DM. We simulate the evolution of the phase screens for every temporal interval Δt moving the layers of the quantities

$$\Delta\vec{s}_i = \vec{v}_{wind,i} \cdot \Delta t$$

where $\vec{v}_{wind,i}$ is the speed vector of the i[th] layer. The integer part of the movement $\Delta\vec{s}_i$ corresponds to a simple shift of the screen, while the residual non-integer difference is applied by a linear interpolation.
The control algorithm is a pure integrator. For every temporal step the code computes the measured wavefronts of the NGSs by co-adding the portions of the phase screens where the footprints are projected. This procedure is repeated for every GS in order to obtain the wavefronts of all NGSs. Now we are able to focus to the desired layer altitude according to the LO scheme. The obtained NGS wavefronts are superimposed considering the positions of the footprints at the conjugation altitude. The resulting arrays (one for every DM) are now weighted for the different illuminations of the footprints because of the different magnitudes of the NGSs. The final result is an array that contains the phase measurements of the layers at the conjugation altitude.
The evolution time step, Δt, of the atmosphere has to be smaller than the integration time on the WFS. This time is a user-defined parameter (as the Δt). The user defines over how many temporal steps the WFSs have to integrate the received signal.

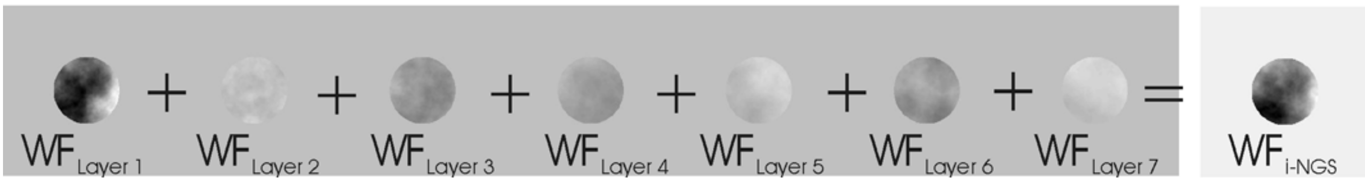


Figure 3. A single NGS wavefront is the sum over the layers of the projected footprint of the GS. Every single footprint has the dimension of the pupil. The resulting array is the WF_i of the Equation 3 that always is a pupil containing the phase delay experienced by the starlight and collected by the telescope aperture in the direction of NGS. Different grey levels indicate different phase values.

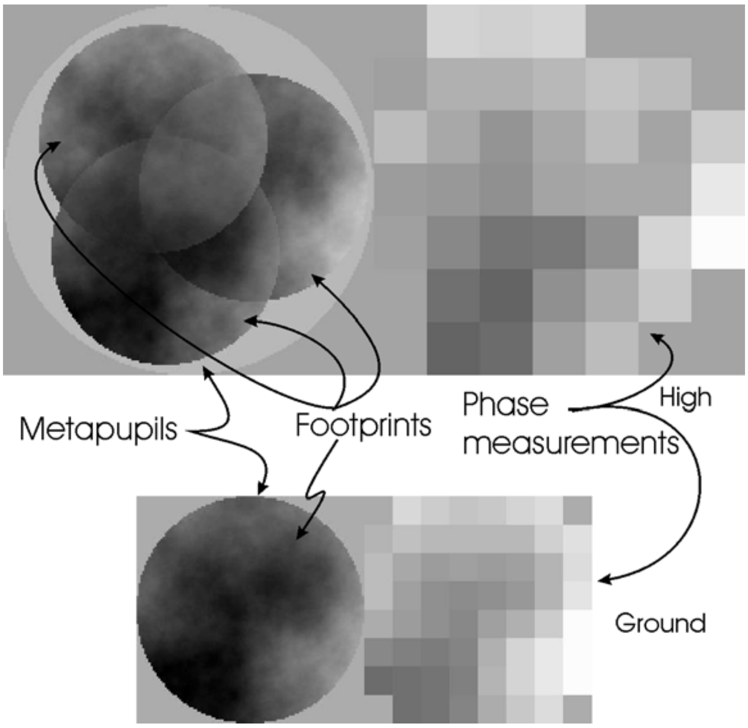


Figure 4. The two pictures show the results of the LO procedure described in the text. The figure in the top shows what the high altitude WFS measurements: the footprints of the 3 stars are not perfectly overlapping because of the different direction of the NGS. Here is also showed the metapupil circle, but only for graphics convenience. The three pupils can't cover all the metapupil. What is measured is the right part of the image where is visible the effect of the spatial sampling. The figure on the bottom shows the measurements on the ground WFS. In this case the pupil footprints and the metapupil are overlapping. In the two images we used the same scale: the dimension of the highest metapupil is bigger than the ground one as it was computed. The pictures show the footprints relative to the 3 stars over 120" FoV case. Different grey levels indicate different phase values.

These reconstructed layers are then used to compute the shapes of DMs and, after a number of steps equivalent to the time delay, these ones are subtracted to the wavefronts of the GSs.
The incoming phases combined in LO mode focused to the conjugation altitude are continuous quantities. They have to be discrete ones to become real measurements. They are the average of the wavefronts in the area corresponding to the pixel of the WFS 's CCD (excepting the noise contribution). In the real system these pixels can be binned 2×2 or 4×4 in order to minimize the noise of WFS. In the simulator we also consider this possibility by averaging the phase values of the 2×2 or 4×4 pixels composing the binned pixel and associating to this value the relative noise (see sec. 4.3 and sec. 4.4).
For every temporal step the instantaneous PSF of all the stars ("dummy" and guide) are computed by FFT in order to measure their instantaneous SR and to integrate the incoming light to produce the long exposure PSF. This method is able to compute a Strehl Ratio map over the FoV by linear-interpolating the SR values of the "dummy" stars. The starlight integration is made by co-adding the PSFs step by step for every "dummy" stars. In this way the user is able to compute the SR and the PSF for the position over the FoV set before. We want to stress that the loops relative to different DMs are fully independents.

4.3 Noise

We consider two different possibilities for the generation of the noise on the measurements of the phase made by WFS. One is the method described in section 4.3.1 that takes into account only the photon noise without considering any other sources of noise, the other one is the simulation of the phase error noise generated by using a SHWFS, described in section 4.3.2.

4.3.1 Calibration

We first developed some routines that simply add on the measurements coming from WFS a random noise. We want to simulate the photon noise that is dependent on the square root of the number of photoelectrons received by every WFS. A numerical factor is set to scale the variance of the phase noise to add. The variance of photon noise is inversely proportional to the number of photoelectrons collected n_{ph} and, if we call K the numerical factor, we can write the phase variance of the noise array as:

$$\sigma^2_{\varphi,K} = K n_{ph}^{-1}$$

4.3.2 Shack Hartmann

In order to simulate the response of a Shack-Hartmann wavefront sensor and to take into account all the noise sources, due to the Poissonian distribution of photons, Read-Out-Noise (RON), dark current and sky background we apply the formulas found by Rousset (1999). These relate the noise relative to the intensities measurements to the noise relative to measured phase. The noise due to every considered source is given in terms of variance per sub-aperture, σ^2_φ:

Equation 4 Photon

$$\sigma^2_\varphi = \frac{\pi^2}{2}\frac{1}{n_{ph}}\left(\frac{N_T}{N_D}\right)^2$$

Dark and RON

$$\sigma^2_\varphi = \frac{\pi^2}{3}\frac{\sigma_e^{\,2}}{n_{ph}^{\,2}}\left(\frac{N_S^2}{N_D}\right)^2$$

Sky background

$$\sigma^2_\varphi = \frac{\pi^2}{3}\frac{n_{bg}}{n_{ph}^{\,2}}\left(\frac{N_S}{N_D}\right)^2$$

all are expressed in rad^2, where n_{ph} is the number of photons detected in the sub-aperture, N_S^2 is the total number of pixels per sub-aperture, N_T is the image FWHM and N_D the FWHM of the diffraction pattern of a sub-aperture (N_T and N_D are measured in pixels) and n_{bg} are photons received from the sky background. The σ_e is the Root Mean Squared (RMS) of photoelectrons due to RON and dark counts. It is equal to the RON value in the RON case and to the square root of the dark counts in the dark case.

Equation 5 Dark and RON RMS $\sigma_e = \sqrt{RON^2 + N_{e,dark}}$

The number $N_{e,dark}$ is the number of photoelectrons measured by frame and due to the dark current only:

$$N_{e,dark} = dark \cdot N_{px} \Delta t$$

where *dark* is the value of the dark current expressed in electrons by pixel by second, N_{px} is the number of pixels composing a single binned pixel (for MAD N_{px} is 1, 4 or 16 according to the binning) and Δt is the integration time relative to a single frame. If we assume the quantity $N_{e,dark}$ follows a Poissonian distribution, we can consider that the RMS due to dark current only is the square root of this number:

$$\sigma_{e,dark} = \sqrt{N_{e,dark}} = \left(dark \cdot N_{px}\Delta t\right)^{1/2}$$

that is the value considered in Equation 7.

Applying this equation we obtain an array of variance value of the same dimension of the array that reproduces the phase measurements. This variance allows the computation of random phase noise maps. Those are arithmetically added to the phase maps given by LO procedure.

On the WFS conjugated to the ground layer the footprints of the NGSs are perfectly superimposed. The illumination of the metapupil is complete and uniform, so all the quantities used in the relations before are the same for every sub-aperture completely illuminated. The phase noise variance is constant over the metapupil excluding the edge pixels that may be partially illuminated by the NGS footprint.
On the WFS conjugated to the high altitude layers the footprints of the stars are not completely overlapping and the illumination of the sub-apertures depends on the direction and the magnitude of the GSs. This affects the map of the phase noise variance that is not uniform with valley where the footprints are overlapping and peak where only the light of the faintest NGS is projected.
We said above, sec 4.5 and Figure 7, that only some sub-apertures are useful, while those not well illuminated are discarded (we call them *dark pixel* in the following). This is true also in the noise computation: the code gives numerical errors when it tries to compute the phase noise variance relative to the dark pixel, being the value of SNR exactly 0. So, also in this case, not all the binned pixels in the metapupil are considered in the computation.

When the integration over the WFS CCD ends the noise map is then computed. Every time an array of random values is generated, where every value has unitary variance. This array is multiplied point-by-point with the array containing the variance values of each binned pixel, retrieving the noise array.

The noise arrays computed (one for each WFS) are added to the measurements arrays obtaining the arrays of real measurements used to reconstruct the wavefront and to calculate the shape of the DMs.

4.4 Pyramids

The modeling of the pyramid WFS in the simulator is based on geometrical approximation and on the WFS noise theory developed for SH. The main idea is to consider the SH sub-aperture equivalent, in the pyramid case, to the dimension of the binned pixel used to sample the image of the four re-imaged pupils. Under this assumption the pyramid is quite equivalent to the quad-cell Shack-Hartmann. The four pixels of the SH quad-cell are corresponding to the same pixel on the four pupils of the pyramid case. These four pixels, every ones in a different pupil, allow measurements of the phase in a single sub-aperture of the main pupil.
In order to reproduce the effect of a pyramid-based wavefront sensor we consider the SH procedure described above assuming a modulation of the pyramids that give always a linear response of this device for every degree of correction. We model the coefficient N_S, N_D, and N_T to be consistent with the characteristics of the pyramid WFS and we substitute this values in the Equation 4. Some studies predict a gain in limiting magnitude of the pyramid with respect to SH, anyway we use a conservative approach where no gain is considered.
In the SH case the limiting factor is the dimension of the sub-aperture while in the pyramid case this one becomes the dimension of the binned pixel (the equivalent to the sub-aperture) used to measure the intensities over the four pupils. From the comparison to the quad-cell we obtain

$$N_S^2 = 4$$

because this is the number of pixel used to sense.

N_D in SH is the dimension of the PSF pattern in the diffraction limited spot of the sub-aperture *d*, measured in pixel. Instead of it we use in the pyramid case the linear dimension of the quad-cell itself (its side) in pixel, so

$$N_D = 2$$

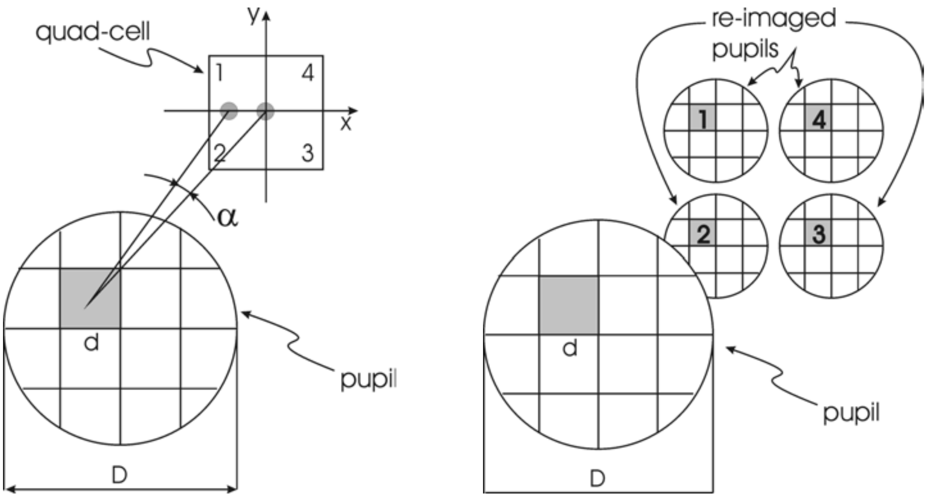


Figure 5. These two pictures show the analogy between the quad-cell SH (left) and the pyramid WFS (right). In particular, the pictures refer to the case of a WFS conjugated to the ground. In both pictures *D* is the dimension of the metapupil and *d* of the binned pixels (or sub-aperture). In both cases the light collected by a single sub-aperture is measured by four pixels: the quad-cell in the SH case and the four correspondent pixels on the four re-imaged pupils in the pyramid case. In the SH we measure the shift α of the spot while in the pyramid we compute the intensities on the four pixels (1, 2, 3 and 4) to retrieve the wavefront derivatives.

This assumption is justified considering that the pixels composing the quad-cell in the pyramid case are completely illuminated (excepting the pixel on the edge of the metapupil): if in the SH N_D is the portion of pixels illuminated by the diffraction limited spot, in pyramid N_D becomes the portion of pixel illuminated by the starlight.

If we refer to the variance of a single sub-aperture the Equation 4 for dark and RON becomes:

Equation 6

$$\sigma^2_{\varphi,RON} = \frac{16\pi^2}{3}\frac{\sigma^2_{e,RON}}{N_D^2}\frac{1}{n_{ph}^2} = \frac{4\pi^2}{3}\left(\frac{RON}{n_{ph}}\right)^2$$

Equation 7

$$\sigma^2_{\varphi,DARK} = \frac{16\pi^2}{3}\frac{\sigma^2_{e,DARK}}{N_D^2}\frac{1}{n_{ph}^2} = \frac{4\pi^2}{3}\left(\frac{(dark\cdot N_{px}\Delta t)^{1/2}}{n_{ph}}\right)^2 = \frac{4\pi^2}{3}\left(\frac{dark}{n_{ph}^2}\right)N_{px}\Delta t$$

where $\sigma^2_{\varphi,RON}$ is the variance due to the RON and $\sigma^2_{\varphi,DARK}$ due to the dark. Here RON is expressed in electrons by (binned) pixel by frame and the dark current, *dark*, in electrons by pixel by second. N_{px} is the number of pixels composing a single binned pixel and Δt is the integration time relative to a single frame.

The N_T parameter is present only in the Equation 4 relative to the photon noise and its value should be computed at every step of the loop by using FFT. In the simulations we presented in this document we set the value of the ratio N_T/N_D of this equation constant and equal to 1, instead of the result coming from the

computation of this quantity. In this case also the N_D value should be computed by FFT as the N_T, because they are the two members of the same ratio and must be expressed in the same unit. The N_D is the FWHM of the diffraction pattern relative to a sub-aperture and, of course, it is constant over all the simulation. The N_T is the value of the image FWHM through the sub-aperture. After few movements of DMs N_T reaches the dimension of N_D because the FWHM relative to the residual phase variance due to the partially corrected turbulence becomes smaller than the diffraction effect of the sub-aperture.
This is true when the dimension of the sub-aperture is small, as in the SH and pyramid cases, and, at the same time, the observation are referred to a good seeing case.

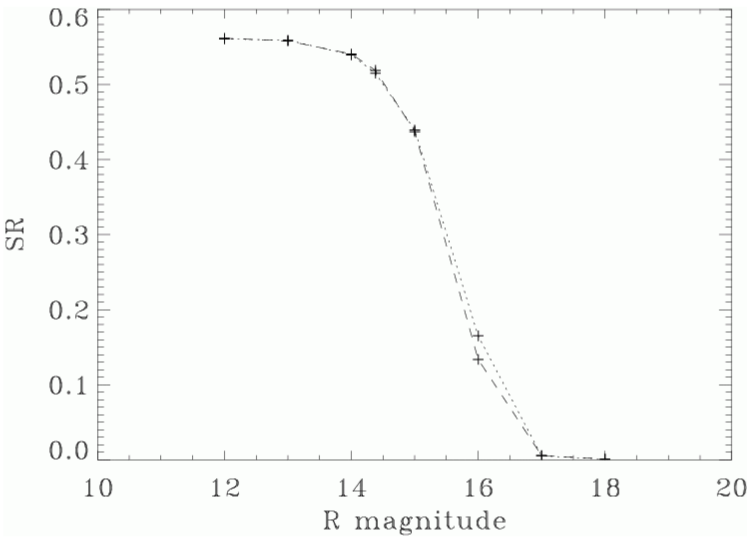


Figure 6. AO case with DM conjugated to the ground, $r_{0,V}$=0.25, $r_{0,J}$=0.90, gain of 0.603, RON of 5 e^-/pixel/frame, 10×10 sampling of WFS, zonal reconstruction, integration time for WFS 11 msec. Bandwidth Δλ=0.22 μm, quantum efficiency of 50%, observation at H band (1.6 μm), WFS wavelength 0.7 μm, diameter 7.9 m, central obstruction 1.2 m and integration time for long exposure of 0.1 sec.

The equation relative to the sky background noise is given by substituting the parameters founded for the pyramid WFS:

$$\sigma^2_{\varphi,SKY} = \frac{\pi^2}{3}\frac{n_{bg}}{n^2_{ph}}$$

The number of photons detected, n_{ph}, takes into account the splitting of the light. The optimisation of the light between the two WFSs was not analysed and we always consider the 50% and 50% case (with a loss of light due to the beam splitter of 4% taken in account in the total quantum efficiency). We want to point out that these procedures are based on the Rousset formulas as they are.

4.5 Zonal and Modal reconstruction and correction

The user can choose two ways to compute the deformable mirror shape either *zonal* or *modal*.
The measurements coming from the simulate phase sensor are the input data of the reconstruction-correction procedure. These are expressed in arrays of phase values, where the number of array is equal to the number of WFS. There is a one to one correspondence between the elements of the arrays and the measurements coming from the binned pixels. But not all the binned pixel measurements are used in the reconstruction procedure. Only the binned pixel illuminated by the stars footprints at least of a user-define amount (the standard is 10%) are considered in the reconstruction process.

The maximum number of DM modes to be used is limited first by the number of binned pixels composing the metapupil image and in second instance by the number of pixel illuminated by the footprint. It is possible also set the value of the gain to apply to the correction.

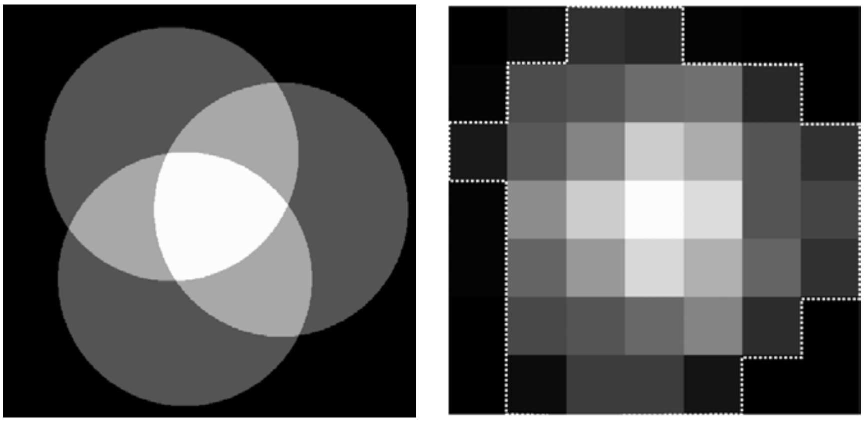


Figure 7. The pictures show the footprints relative to the 3 stars over 120" FoV. On the left: the footprints as they are projected to the high altitude DM. The intensity is higher in the overlapping region because the light of the three NGSs is summed. On the right: the same footprints as seen by the WFS, the effect of the spatial-sampling is visible. Only the binned pixel illuminated over a threshold value are considered in the calculations. Here these pixels are delimited by the white dotted line.

4.5.1 Zonal

In zonal reconstruction a DM is assumed to have the shape given by linear spline interpolation of the phase measurements made by the WFS. We use the zonal reconstruction approach to check the simulator while the performances results are obtained with a modal reconstruction such as in MAD.

4.5.2 Modal

In modal reconstruction the measurements of LO WFSs are interpolated with a user-defined number of Zernike polynomials or with a user-defined DM modes surfaces. This interpolation is made by inverse matrix operations: we solve the linear system that relates the LO phase measurements to the DM modes and obtaining as results the coefficients relative to every DM mode considered.
We assume that the DMs are able to reproduce the mode used to interpolate the measurements and we set them equal to the linear combination of these modes with the coefficients coming from the interpolation above.

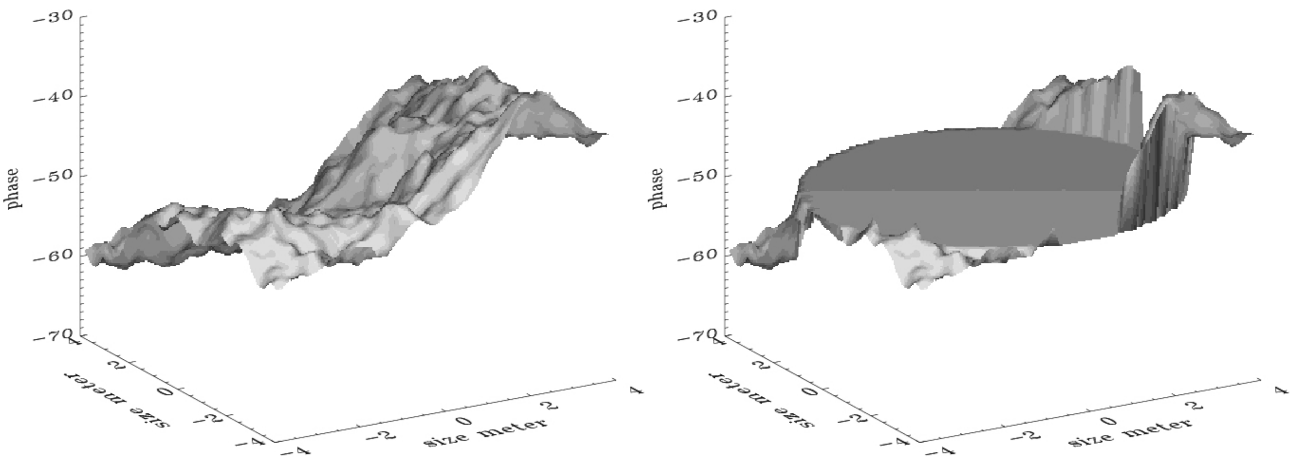


Figure 8. Left: the phase delay due to atmospheric turbulence on the guide star (in radians). Right the effect of the correction on this wavefront - the non-piston terms on the phase screen inside the pupil were completely removed.

We said that not all the measurements coming from binned pixel inside the metapupil are used for the reconstruction of the wavefront. Thus only some binned pixels are used to compute the interaction matrix. We considered only the binned pixel illuminated or partially illuminated by the NGS footprints, even if the computation of DM surfaces is made for the entire metapupils.
The maximum number of modes usable to reconstruct is determined by the number of binned pixel considered. Nevertheless some of these modes may be discarded: the selection is defined by keeping only the polynomials that has an eigenvalue of the interaction matrix inverse bigger than a threshold value (usually define as the 10 % of the maximum eigenvalue).

5 Code description

The code is written in IDL (Interactive Data Language, version 5.2 and newer) and run under Unix/Linux and Windows Platforms. It's made by a series of routines organized from high to lower level. The two higher routines are mcaosim.pro, for the initialisation of the input parameters and loopmcao.pro to realize the adaptive loop in LO fashion. All these parameters are summarized in the section 0.

5.1 Script Description - Input Parameters

The characterization of the system is defined by filling a script file. The user has to define:

- the main parameters of the telescope like the diameter or the value of the central obstruction,
- the main parameters of the AO like the number of deformable mirrors or the spatial-temporal sampling of the wavefront sensor,
- all the others relevant technical details like wavefront sensing or correcting ones.

The parameters linked to the DMs and WFSs can be different for different conjugation altitude. In Table 1 we give a complete list of the simulation parameters.

Telescope parameters
Diameter, central obstruction, FoV, azimuth angle, layer dimension, pupil dimension, scientific wavelength, temporal step
System parameters
Number of DM, conjugation altitude, gains apply to the correction, spatial sampling of WFS, integration time WFS
Loop parameters
Total number of iteration, iteration number when code starts integration, number of DM's modes, WFS noise model, dimension of the array used to compute PSF, threshold value for correct detection, threshold eigenvalue to choose good DM's modes
Atmospheric parameter
Number of layer, D/r_0, wind speed, layer heights, fine tuning parameter for layer normalization, outer-scale
SR map parameters
Model of sky-directions for SR measurement, number of sky-directions where SR is measured, sky-directions
NGS parameters
Model for NGS asterism, NGS directions, NGS magnitudes
Noise and detectors parameters
Integration bandwidth, quantum efficiency, RON, dark current, dimension of the projection of the pyramids in the sky, sky magnitude, dimension of the CCD, WFS wavelength, delay time between detection and correction, number of iteration composing long exposure data

Table 1. List of the user-defined parameters to be set in the script.

5.2 Script description:

Before to run the simulation, the user has to fill the script file with the appropriate values. All these parameters are described in the following indicating also their type, the fundamental limits and when a reasonable value. All these parameters are indicated as they appear in the script. The optional parameters can be skipped if they are unused. The needed parameters have to be set even if they are not used in a specific simulation. We divided the parameter description in several sub-sections as it is presented in the script.

SPECIFICATION: optional
It defines the prefix of all the files generated by the simulation. If not set, the code will automatically use the first free number as prefix. This parameter is a string variable.

5.2.1 Telescope parameters:

RestoreFileTelescope: needed
This parameter is used if we want to use a existent telescope file or if we define new parameters. If the input is '' the code will use the following parameters while if RestoreFileTelescope is a file name, the parameters saved in this file will be restored. This parameter is a string variable.

D: needed
It is the telescope diameter in meter in real or double precision (scalar variable).

Eps: Needed
It defines the central obstruction ratio (percentage of the telescope diameter therefore lower than 1). Real number.

FoV: needed
It defines the FoV. If it is a scalar variable, the system will use the single FoV mode. If the parameter is a vector variable then it must have ndm+1 elements (with 2 DMs, three numbers!) and the system will use the multiple field of view mode. In this case the first number is the inner diameter in arcsec of the ground annulus, the second number is the outer one, the following numbers define the FoV of the next DMs. The FoVs are given in arcsec. The second number must be larger than the first one. If the MfoV mode is set the keyword **EXTRA_FIELD** must be also set to any value in the script or in the command line..

Theta: needed
Azimuthal angle in degree (between 0 and 60 degrees). The zenith is defined by zero. This is a scalar variable.

Nsize: needed
Dimension of the phase screen in pixel. It is a scalar long variable. It must be always larger than npupil.

Npupil: needed
Define the telescope diameter in pixel and therefore the pixel scale of the phase screen. It must be always smaller then nsize. High values of npupil will increase the computation time. It is a long scalar variable.

WaveLength: needed
It defines the scientific wavelength and also the reference wavelength for the phase screen. This scalar number is in micron.

DeltaT: needed
Basic temporal unit for the overall simulation, it is also the evolution step value for the atmosphere. It is defined in seconds (scalar variable).

5.2.2 System parameters:

RestoreFileSystem: needed
If the input is ‘’ the code will use the parameters defined in this section while if RestoreFileSystem is a file name, the parameters saved in this file will be restored. This parameter is a string variable.

NDM: needed
Define the number of DMs. It is a scalar integer variable and must be larger than zero. The following parameters must have the same number of elements as NDM.

RangeDM: needed
Conjugation altitude of each DM in meter. It is a real vector variable.

GainDM: needed
This vector defines the gain used when applying the correction. A gain is defined for each DM, an usual value is between 0.3 and 0.9. It is a real vector variable.

SpaceDM: needed

Define the spatial sampling of each CCD (in sub-aperture unit). If it is zero, the simulation uses an infinite sampling (the sub-aperture size becomes the pixel dimension). It is a long vector.

RateDM: needed
Define the temporal sampling of each CCD (in DeltaT unit). It is a long vector and must have its elements larger than zero.

5.2.3 Loop parameters:

RestoreFileLoop: needed
If the input is '' the code will use the parameters defined in this section while if RestoreFileLoop is a file name, the parameters saved in this file will be restored. This parameter is a string variable.

NLoop: needed
Number of iterations in the simulation. Nloop × DeltaT gives the total simulation time.

StartClosing: needed
Define the iteration when the WFS begins to integrate. It is a long scalar variable and must be smaller than (Nloop-1).

NZernike: optional
When this parameter is set, the modal correction will be used. Its is a vector containing the numbers of mirrors modes to be used by each mirror. If the keyword RestoreDM is not set in the script or in the command line then the Zernike modes will be used otherwise if RestoreDM gives the file where the mirror modes are stored, these mirror modes will be restored. If Nzernike is undefined the simulation will use a zonal correction.

switchnoise: needed
This parameter allows to introduce the noise computation. It is a string variable. If it is 'no' then the simulation is noise free. If it is 'calibration' then using the variable knoise we can set directly the noise variance for each CCD. If the parameter is 'sh' the code uses the Shack-Hartmann noise defined by Rousset. In this last case, the variable sh_mode defined in section Noise Data has to be carefully set.

Makeanimation: needed
The animation works only with IDL5.2 and windows. If it is '' then nothing is done otherwise it defines the file name where the animation is stored. This option is not compliant with most of the IDL versions and therefore should not be used.

PathMovie: needed
Define the path of the animation file. Standard is ''. This parameter is a string variable.

PSFsize: needed
This parameter is the dimension of the array used to compute the PSF. The ratio PSFsize/npupil defines the sampling of the FWHM of the star used to compute the Strehl ratio. It is a long scalar variable.

Limit: needed
This parameter allows to define the lowest limit of illumination of the sub-aperture to be considered in the metapupil. If it is equal to 0, no sub-aperture will be rejected while if it is equal to 1 only the full

illuminated ones will be used in the correction computation. It is a vector variable defined for each corrector.

coeff_cut: optional
If the correction is modal, this parameter is needed to define the condition number. It is a vector variable defined for each corrector.

5.2.4 Atmosphere:

RestoreFileAtmosphere: needed
This parameter is used if we want to use a existent atmosphere file or if we define new parameters. If the input is ‘’ the code will use the following parameters while if RestoreFileAtmosphere is a file name, the parameters saved in this file will be restored. This parameter is a string variable.

NLayer: needed
Number of layers. All the following parameters must be vector with length equal to Nlayer. It is a long variable.

DoverR0: needed
It is the ratio D/r_0. It is a real vector.

V: needed
It is the absolute value of the wind for each layer. In the simulation, the wind is vector with a random direction. This parameter is a real vector in meter per seconds.

L0: optional
This parameter is the outer scale defined in meter. It is needed to use the von Karman power spectrum. If not set, the Kolmorogov spectrum will be used. The parameter is a real vector. A typical value of outerscale is 20 meters.

H: needed
Define the altitude of each layer. It is a real vector in meter.

Spacing: needed
It defines the spacing of the grid point centre of the pupils used to normalise each layer. It is a long scalar.

5.2.5 Dummy:

RestoreFileDummy: needed
This parameter is used if we want to use a existent dummy file or if we define new parameters. If the input is ‘’ the code will use the following parameters while if RestoreFileDummy is a file name, the parameters saved in this file will be restored. This parameter is a string variable.

selectstrehl: needed
If it is set to 0 a grid with nsidedummy per nsidedummy stars and a step of gridstep will be created.
If it is set to 1 the code will used the reference stars. If it is set to 2 the user will define the coordinates of each test star xdummy and ydummy. It is a integer parameter.

Nsidedummy: optional

It is needed if selectstrehl is set to 0. It defines the elements number of the square grid side used to compute the Strehl map. It is a long number.

Gridstep: optional
It is needed if selectstrehl is set to 0. It defines the step of the square grid side used to compute the Strehl map. It is a long scalar in arcsec.

Xdummy: optional
It is needed if selectstrehl is set to 2. It defines the x-coordinate of the test stars used to compute the Strehl map. It is a real vector. The coordinates are in arcsec.

Ydummy: optional
It is needed if selectstrehl is set to 2. It defines the y-coordinate of the test stars used to compute the Strehl map. It is a real vector. The coordinates are in arcsec.

Vdummy: optional
It is needed if selectstrehl is set to 0 or 2. It defines the magnitude of the test stars used to compute the Strehl map. It is a real number.

5.2.6 Stellar Field

RestoreFileField: needed
If the input is ‘’ the code will use the following parameters while if RestoreFileField is a file name, the parameters saved in this file will be restored. This parameter is a string variable.

switchasterism: needed
If it is set to 0 then the code will use the star density model of Bahcall and Soneira (1981) with the typical density of the galactic_latitude and galactic_longitude that must be set. If it is set to 1 the user will define the parameters: xaster, yaster and vaster.

galactic_latitude: optional
Define the galactic latitude in degree (from 0 to 90). This parameter is needed if switchasterism is equal to 0. It is a real number.

galactic_longitude: optional
Define the galactic longitude in degree (from 0 to 360). This parameter is needed if switchasterism is equal to 0. It is a real number.

Allstar: optional
Define the number of stars to be selected. If it is equal to 1 all the stars are selected otherwise the stars are selected using the parameter LimitMagnitudeDiff. This parameter is needed if switchasterism is equal to 1. It is a long number.

Xaster: optional
It is needed if switchasterism is set to 1. It defines the x-coordinate of the reference stars. It is a real vector. The coordinates are in arcsec.

Yaster: optional

It is needed if switchasterism is set to 1. It defines the y-coordinate of the reference stars. It is a real vector. The coordinates are in arcsec.

Vaster: optional
It is needed if switchasterism is set to 1. It defines the magnitude of the reference stars. It is a real vector.

LimitMagnitudeDiff: optional
It is needed if switchasterism is set to 1. Define the difference between the faintest magnitude and the brightest allowable one. It is a real vector.

5.2.7 Noise data

RestoreFileElectronic: needed
If the input is '' the code will use the following parameters while if RestoreFileElectronic is a file name, the parameters saved in this file will be restored. This parameter is a string variable.

BandWidth: needed
Define the bandwidth integration (micron). Even if the noise is not computed this parameter has to be set.

Eff: needed
It defines the quantum efficiency (normalized unit). Even if the noise is not computed this parameter has to be set.

RON: needed
It defines the Read Out Noise RMS in unit of electron per pixel per frame. Is a real vector with length equal to the number NDM.

dark: needed
It defines the dark RMS in unit of electron per pixel per second. Is a real vector with length equal to the number NDM.

pyr_size: needed
It defines the dimension of the WFS element projected on the sky. It is a long scalar in arcsec.

Sky: needed
It defines the sky luminosity background magnitude. It is a real scalar.

Ccdside: needed
It defines the number of pixel on one side of the CCD. It is a long vector (one number for each CCD).
The CCD is supposed to be square.

Fwhmside: needed
This parameter is the dimension of the array used to compute the PSF for noise purposes. The ratio Fwhmside/(sub-aperture dimension) defines the sampling of the FWHM of the star used to compute the Strehl ratio. It is a long scalar variable.

lambda_wfs: needed
It is the WFS wavelength in micron. This parameter is a real scalar.

Delay: needed

Define the temporal delay in DeltaT unit. It is a real vector, an element for each loop. It must be lower than two times the RateDM value.

Percent: needed
If sh_mode is set to 'percent', this parameter defines the iso-illumination curve in percent that represents the value Nt. It is a real vector.

sh_mode: needed
This parameter can be defined as:

- 'percent': then the ratio Nt/Nd is computed by using the iso-illumination curve diameter.
- 'fixed', the Nt/Nd ration is fixed to 1.
- 'fwhm', the Nt/Nd ratio is computed by using the measured FWHM.

5.2.8 Miscellaneous

Path: needed
Define the path where the output files are saved. It is a string.

SAVE_ZERNIKE: optional
If set the Zernike coefficients for each loop are saved in a file .dat.

Graphics: optional
If it is set to 'yes', the code computes the drawings during the simulation.

Border: optional
If set to 1 the code considers the border effects due to the numerical rebin in the WFS measurement.

iterlong: needed
This parameter defines the loop iteration from which the long exposure PSF is saved for each dummy star. For long exposure integration.

W_iter: optional
The logfile is updated every W_iter loop iterations.

sr_limit: optional
Define the maximum SR level considered in the graphics if the graphics keyword is set to 1.

fieldsr: optional
If set to 'yes', the code computes the long exposure SR also on the reference stars.

Restoredm: optional
If set defines the file where are stored the mirror modes to be used in the simulation run.

Mismatch_St_enl:
Consider the xy shift of the star-footprints on the WFS, due to the Star enlargers, measured in pixel of the simulation. It is in the form:
([xy,dm,xystar])

Mismatch_Pupil_dm:

Consider the xy shift of the star's pupils on the DM, in pixel of the simulation. It is in the form:
([[xy,star,xy.on_dm]])

Mismatch_Mask_WFS:
Consider the xy shift of the mask on the WFS in pixel of the simulation. It is in the form:
[[x-shift(wfs1),x-shift(wfs2),…],[y-shift(wfs1),y-shift(wfs2)],…]

Mismatch_Mask_DM:
Consider the xy shift of the mask on the DM in pixel of the simulation. It is in the form:
[[x-shift(dm1),x-shift(dm2),…],[y-shift(dm1),y-shift(dm2)],…]

Silent:
If set print actions are suppressed

Extra_field:
if set the code considers also the NGS outside the FoV. This keyword is needed in the MfoV approach and MUST be set in the script or in the command line.

SubZern:
If this keyword is set to 'yes' and if the correction is modal, then in the computation of the Interaction Matrix will be used all the modes specified in the Nzernike variable. If this keyword is not set then the upper limit of the number of modes taken into account is defined by the number of sub-apertures in each metapupil.

Nocut:
If this keyword is set then the un-useful part of the phase screens is not discarded in order to speed up the atmosphere evolution preocedure. This keyword must be set if the phase screens are circular and are fully useful.

5.3 Output files

For each simulation run the following files are always generated:

- Sim+prefix.log : log file with general information about the script and the simulation run
- prefix.tel : telescope data
- prefix.sys : system data
- prefix.dum: SR map star coordinates data
- prefix.lop : loop data
- prefix.elc: noise data
- prefix.fld: NGS data
- psflong+prefix.dat: here are stored the lomg exposure PSF for each dummystar
- strehl+prefix.dat: here are saved the SR values for each test star an for each iteration computed on the centre of the PSF
- inststrehl.dat: here are saved the SR values for each test star an for each iteration computed on the maximum of the PSF

Using specific keywords others files can be generated:

- Keyword **WFE :** the variance measurements on the ccd are saved for each iteration in WFE+prefix.dat
- If keyword **save_Zernike** is set then a savezernike+prefix.dat is generated.

- If keyword **nowin** is not set then a plot of SR map on srmap+prefix.ps, of SR evolution on realstrehl+prefix.ps, of instantaneous SR evolution on srevolution+prefix.ps, dummy stars map on dummy+prefix.ps and reference star map on field+prefix.ps are generated. This keyword must no be set on system without 'X' or 'WIN' display.

5.4 Script Examples

In order to have a clearer view of the script, we give in this section three examples of script:

- Example 1 parameters (section 5.4.1) have been chosen to simulate a MCAO system with 2 DMs, without any noise and using the zonal approach. The atmosphere is assumed Kolmogorov.
- Example 2 (section 5.4.2) simulates an MCAO system with two DMs.The noise is taken into account, the correction is done using the modal approach and atmospheric screens follow the von Karman spectrum.
- With example 3 (section 5.4.3) we want to confront the simulation of a MCAO system which use a single FoV as in Example 2 with a similar simulation which uses the Multiple FoV approach. The different parameters files produced with the example 2 are restored except the stellarfield and the telescope files which contains the FoV parameters.

5.4.1 Zonal correction, noise free simulation:

```
; Script: EXAMPLE1
;
;       TARGET:     Simple simulation without noise with zonal correction approach. A 7 layers
;                   model of the atmosphere typical of Paranal is taken into account (with
;                   Kolmogorov Power Spectrum). We consider an 8m telescope with 2 DMs
;                   conjugated to 0 km and 8.5 km, We want to retrieve the SR map on the
;                   whole corrected FoV of 120 arcsec.
;-------------------------
;
SPECIFICATION = 'example1'        ; prefix attached to all the output files

; Telescope

RestoreFileTelescope = ''         ; No file is restored
D = 8.                            ; Telescope diameter = 8 meters
eps = 0.1                         ; Central obstruction 1/10 of the diameter
FoV = 120.                        ; Corrected FoV of 120 arcsec
Theta = 0.                        ; Azimuthal angle equal 0 (Zenith)
nsize = 1024L                     ; Layer dimension in pixel = 1024 px
npupil = 128L                     ; Pupil dimension in pixel = 128 px
WaveLength = 2.2                  ; Imaging wavelength  = 2.2 micron (K band)
DeltaT = .001                     ; Time evolution step of 1 millisecond.

; System

RestoreFileSystem = ''            ; No file is restored
NDM = 2                           ; 2 Deformable mirrors are considered
RangeDM = [0.d0,8500.d0]          ; The conjugation altitude of the 2 DMs are 0 meter and 8500 meters
GainDM = [0.5,0.45]               ; Gains used when applying the correction to the two DMs.
SpaceDM = [0,0]                   ; The WFSs sampling is infinite so is set to 0.
```

RateDM = [4,3]	; The integration time of the two loops is 4 and 3 times the time ; evolution step (4 ms,3ms)

; Loop

RestoreFileLoop = ''	; No file is restored
NLoop = 100	; 100 iterations are considered.
StartClosing = 0	; The integration starts from the first (0^{th}) loop iteration
; NZernike =[59L,49L]	; As the zonal correction is used this parameter indicating the ; number of modes is commented.
switchnoise = 'no'	; No noise is considered
; knoise = [10.,10.]	; Because no noise is considered this line is useless
MakeAnimation = ''	; The animation file is not generated
PathMovie = ''	; Useless
PSFsize = 256L	; The dimension of the array used to compute PSF of test stars is ; 256 px, twice the npupil value
limit = [.1d,.1d]	; Pixels illuminated for more than 10 % of their area are considered.
;coeff_cut=[8.5,8.5]	; Condition number is commented as we have a zonal approach

; AtmoSphere

RestoreFileAtmosphere = ''	; No file is restored
NLayer = 7	; 7 layers will be generated
DoverR0 = [7.38,2.11,2.67,1.28,1.05,2.11,0.77]	; D/r0 of each layer
V = [6.6,12.4,8.0,33.7,23.2,22.2,8.]	; Absolute speed value of each layer
H = [0.,1800.,3200.,5800.,7400.0004,13100.,15800.]	; Altitude of each layer in meters
; L0 = [12,13,14,15,16,17,18,19]	; The outer scale is commented as we use the ; Kolmogorov Power Spectrum.
Spacing = nsize/8	; The layers are normalised considering ; a grid of pupil every nsize/8=1024/8 pixels

; Dummy

RestoreFileDummy = ''	; No file is restored
selectstrehl = 0	; A square grid of test stars will be created
nsidedummy = 7	; The side of the square will have 7 elements
gridstep = 20.	; The distance between to close star is 20 arcsec
vdummy=16.	; The magnitude of the stars is 16. This parameter has to be set even ; if it is not used by the simulation.

; Stellar Field

RestoreFileField = ''	; No file is restored
switchasterism = 1	; Asterism of reference will be read in the script
xaster = [15.,-28.,42.,-10.,45.]	; X coordinates of 5 natural guide stars in arcsec
yaster = [3.,29.,-8,-13.,30.]	; Y coordinates of 5 natural guide stars in arcsec
vaster = [8.56,8.81,9.02,9.15,9.36]	; Magnitude of each references

; Noise data The following lines define the characteristics of WFS. They have to be fill even if
; the input values are not used by the simulation in this proper case (switchnoise = 'no', noise
; free simulation.

```
RestoreFileElectronic = ''          ; No file is restored
BandWidth = 0.4d0                   ; bandwidth integration (micron)
Eff = 0.1972d0                      ; quantum efficiency of all the system
RON = [4.5,3.5]                     ; Read Out Noise variance in electron per pixel per frame for each
CCD
dark= sqrt([500,100])               ; Dark current RMS in electron per pixel per seconds
pyr_size = 1.0                      ; Dimension of the pyramid in the sky is 1 square arcsec
sky = 20.0                          ; Sky brightness
ccdside = [16.,28.]                 ; Number of pixels on the side of each WFS CCD.
fwhmside = 256                      ; Array dimension used to compute PSF of test stars. It is twice the
                                    ; npupil value. This number is useful when the mode set in sh_mode
                                    ; is not 'fixed' and when switchnoise is 'sh' (Shack-Hartmann).
lambda_wfs = 0.55                   ; Wavefront sensing wavelength in micron is set to 0.55 (V band)
delay=[2,2]                         ; Temporal delay for each loop in deltaT unit.
sh_mode = 'fixed'                   ; The Rousset coefficient Nt/Nd is equal to 1.

; Miscellaneous

Path = ''                           ; Path where be placed the output-file
; SAVE_ZERNIKE=1                    ; Keyword to set if the code has to save the modal coefficients.
;graphics = 'yes'                   ; Set or not the display of the loop by window during the simulation
border = 1                          ; Keyword is set. If set takes into account the numerical effect of
                                    ; the WFS sampling.
iterlong = 5                        ; Indicate the 1st iteration from which the long exposure PSF is
                                    ; integrated.
W_iter = 50                         ; It defines the rate to be used to write the .log file
sr_limit = 0.3                      ; Set in the graphics displayed the maximum value of SR to be plot
fieldsr = 'no'                      ; Not set. If set the long exposure PSF of the GS is also computed.
; save = 1                          ; No atmospheric file is save because the keyword is undefined
; EXTRA_FIELD = 1                   ; Not used now
; SILENT = 1                        ; Not used now
```

5.4.2 Single FoV simulation, Modal correction and noise considered.

; Script: EXAMPLE2

```
;
;       TARGET:   Simulation with noise produced by two Shack-Harmann quadcell WFSs.
;                 The CCDs sampling is used and we consider the modal approach with the
;                 Zernike modes basis. The typical 7 layers model of the Paranal atmosphere
;                 (Von Karman Power Spectrum) is computed. Using an 8-m telescope and
;                 2 DMs conjugated to 0 km and 8.5 km, we want to retrieve the SR map on
;                 the whole corrected FoV of 120 arcsec and on each guide star.
;------------------------
;
```

```
SPECIFICATION = 'example2'          ; prefix attached to all the output files

; Telescope

RestoreFileTelescope = ''           ; No file is restored
D = 8.                              ; Telescope diameter = 8 meters
eps = 0.1                           ; Central obstruction 1/10 of the diameter
FoV = 120.                          ; Corrected FoV of 120 arcsec
Theta = 0.                          ; Azimuthal angle equal 0 (Zenith)
nsize = 1024L                       ; layer dimension in pixel = 1024 px
npupil = 128L                       ; pupil dimension in pixel = 128 px
WaveLength = 2.2                    ; Imaging wavelength  = 2.2 micron (K band)
DeltaT = .001                       ; Time evolution step of 1 msec

; System

RestoreFileSystem = ''              ; No file is restored
NDM = 2                             ; 2 DMs are considered
RangeDM = [0.d0,8500.d0]            ; The conjugation altitude of the 2 DM are 0 meter and 8500 meters
GainDM = [0.5,0.45]                 ; The gain used for the correction of the two DMs is 0.6 and 0.45
SpaceDM = [8,7]                     ; The WFSs sampling is 8x8 for the ground and 7x7 for the high.
RateDM = [4,3]                      ; Integration time of the two loops is 4 and 3 times the time
                                    ; evolution step (4 ms,3ms)

; Loop

RestoreFileLoop = ''                ; No file is restored
NLoop = 100                         ; 100 iterations will be done
StartClosing = 0                    ; The integration starts from the first (0th) loop iteration
NZernike =[59L,49L]                 ; As the modal correction is considered we define the number
                                    ; of modes. The two DMs will use respectively 59 and 49 modes.
switchnoise = 'sh'                  ; The noise is considered. We select the SH WFS.
; knoise = [10.,10.]                ; Because no 'calibration' noise is considered this line is useless.
MakeAnimation = ''                  ; No animation file is generated.
PathMovie = ''                      ; Useless without animation.
PSFsize = 256L                      ; Array dimension used to compute PSF of test stars equal to twice
                                    ; the npupil value
limit = [.1d,.1d]                   ; All the pixels illuminated for more than 10 % of their area are
                                    ; considered.
coeff_cut=[8.5,8.5]                 ; This coefficient is used when computing the interaction matrix
                                    ; to eliminate the singular values to small. This line is needed
                                    ; for the modal correction.

; AtmoSphere

RestoreFileAtmosphere = ''                          ; No file is restored
NLayer = 7                                          ; 7 layers will be generated
DoverR0 = [7.38,2.11,2.67,1.28,1.05,2.11,0.77]      ; D/r0 of each layer
V = [6.6,12.4,8.0,33.7,23.2,22.2,8.]                ; Absolute speed value of each layer
```

```
H = [0.,1800.,3200.,5800.,7400.0004,13100.,15800.]     ; Altitude of each layer in meters
L0 = [12,13,14,15,16,17,18,19]                         ; The outer scale is used to compute screens
                                                       ; with Von Karman Power Spectrum. (meters)
Spacing = nsize/8                                      ; The layer are normalised taking into account
                                                       ; a grid of pupil every nsize/8=1024/8 px

; Dummy

RestoreFileDummy = ''            ; No file is restored
selectstrehl = 0                 ; A square grid of test stars will be created
nsidedummy = 7                   ; The side of the square will have 7 elements
gridstep = 20.                   ; The distance between to close star is 20 arcsec
vdummy=16.                       ; The magnitude of the stars is 16, it has to be defined even if
                                 ; the simulation does not use it


; Stellar Field

RestoreFileField = ''            ; No file is restored
switchasterism = 1               ; Asterism of reference will be read in the script
xaster = [15.,-28.,42.,-10.,45.] ; X coordinates of 5 natural guide stars in arcsec
yaster = [3.,29.,-8,-13.,30.]    ; Y coordinates of 5 natural guide stars in arcsec
vaster = [8.56,8.81,9.02,9.15,9.36] ; Magnitude of each references


; Noise data     The following lines define the characteristics of WFS. They are needed

RestoreFileElectronic = ''       ; No file is restored
BandWidth = 0.4d0                ; bandwidth integration (micron)
Eff = 0.1972d0                   ; quantum efficiency of all the system
RON = [4.5,3.5]                  ; Read Out Noise variance in electron per pixel per frame
                                 ; for each CCD
dark= sqrt([500,100])            ; Dark current RMS in electron per pixel per seconds
pyr_size = 1.0                   ; Dimension of the pyramid in the sky is 1 square arcsec
sky = 20.0                       ; Sky brightness
ccdside = [16.,28.]              ; Number of pixel of the side of CCD for each sensor.
fwhmside = 256                   ; The array dimension used to compute PSF of test stars is 256 px,
                                 ; twice the npupil value. This number is useful when the mode set in
                                 ; sh_mode is not 'fixed' and when switchnoise is 'sh'
                                 ; (Shack-Hartmann)
lambda_wfs = 0.55                ; Wavefront sensing wavelength in micron is set to 0.55 (V band)
delay=[2,2]                      ; temporal delay for each loop in deltaT unit
sh_mode = 'fixed'                ; Set to 1 the ratio of the Nt/Nd Rousset coefficient ratio value.

; Miscellaneous

Path = ''                        ; Path where is placed the output-file
SAVE_ZERNIKE=1                   ; The keyword is set to save the modes coefficients for each loop
                                 ; of the WFSs
```

```
;graphics = 'yes'              ; Set or not the display of the loop by window during the simulation
border = 1                     ; Keyword is set. If set considers the numerical effect of the WFS
                               ; sampling .
iterlong = 5                   ; Set the first iteration from which the long exposure PSF is
                               ; integrated (here the PSF is integrated from the 5th iteration).
W_iter = 25                    ; It defines the rate to be used to write the .log file
sr_limit = 0.3                 ; Set in the graphics displayed the maximum value of SR to be plot
fieldsr = 'yes'                ; Not set. If set the long exposure PSF of the GSs is also computed.
; RESTOREDM = 'dm.dat'         ; The line is commented because no assumption on the mirror mode
                               ; is done. Here Zernike modes are used instead of the mirror modes.
save = 1                       ; The atmo file is save because the keyword is set.
; EXTRA_FIELD = 1              ; Not used now
; SILENT = 1                   ; Not used now
```

5.4.3 A multiple FoV example: comparison with example 2

```
; Script: EXAMPLE3
;
;       TARGET:     We want to confront the simulation single FoV of EXAMPLE2 with a similar
;                   simulation with the Multiple FoV approach. This simulation has some noise
;                   produced by 2 SH quadcell WFs. The CCDs sampling is taken into account
;                   and the modal approach with Zernike modes is considered. A 7 layers model
;                   of the typical Paranal atmosphere is used as well as Von Karman theory. We
;                   simulate a 8m telescope and 2 DMs conjugated to 0 km and 8.5 km. We want
;                   to retrieve the SR map on the whole corrected FoV of 120'' and on each GS.
;-------------------------
;
SPECIFICATION = 'example3'     ; prefix attached to all the output files

; Telescope

RestoreFileTelescope = ''      ; No file is restored
D = 8.                         ; Telescope diameter = 8 meters
eps = 0.1                      ; Central obstruction 1/10 of the diameter
FoV = [120.,360.,120]          ; Corrected FoV of 120 arcsec and annular Fov of 360 arcsec.
                               ; This line also set the use of the Multiple FoV approach
Theta = 0.                     ; Azimuthal angle equal 0 (Zenith)
nsize = 1024L                  ; layer dimension in pixel = 1024 px
npupil = 128L                  ; pupil dimension in pixel = 128 px
WaveLength = 2.2               ; Imaging wavelength  = 2.2 micron (K band)
DeltaT = .001                  ; Time evolution step of 1 millisecond

; System

RestoreFileSystem = 'example2.sys'  ; The example2 system file is restored
;
;                       The system information are now not needed
```

```
;
;NDM = 2                          ; 2 Deformable mirrors (DMs) are considered
;RangeDM = [0.d0,8500.d0]         ; The conjugation altitude of the 2 DMs are 0 meter and 8500 meters
;GainDM = [0.5,0.45]              ; Gain used when applying the correction to the 2 DMs.
;SpaceDM = [8,7]                  ; The WFSs sampling is 8x8 for the ground and 7x7 for the high.
;RateDM = [4,3]                   ; Integration time of the two loops.

; Loop

RestoreFileLoop = 'example2.lop' ; The example2 loop file is restored
;
;                    The loop information are now not needed
;
;NLoop = 100                      ; Here are considered 100 iterations
;StartClosing = 0                 ; Integration starts from the first (0th) loop iteration
;NZernike =[59L,49L]              ; Modal correction is considered so the number of modes is set.
                                  ; Here are considered 59 and 49 mode for the two DMs
;switchnoise = 'sh'               ; Noise is considered. We select the SH WFS
;; knoise = [10.,10.]             ; Because no 'calibration' noise is considered this line is useless
;MakeAnimation = ''               ; No animation file is generated
;PathMovie = ''                   ; Useless without animation
;PSFsize = 256L                   ; The dimension of the array used to compute PSF of test stars is
                                  ; 256 px, twice the npupil value
;limit = [.1d,.1d]                ; The pixel illuminated for more than 10 % of their area are
                                  ; considered
;coeff_cut=[8.5,8.5]              ; This coefficient is used when computing the interaction matrix
                                  ; to eliminate the singular values to small. This line is needed
                                  ; for the modal correction


; AtmoSphere

RestoreFileAtmosphere = 'example2.atm' ; The example2 atmosphere file is restored and the same layers
will be restored
;
;                    The atmosphere information are now not needed and so commented or ignored
;
;NLayer = 7                                           ; 7 layers will be generated
;DoverR0 = [7.38,2.11,2.67,1.28,1.05,2.11,0.77]       ; D/r0 of each layer
;V = [6.6,12.4,8.0,33.7,23.2,22.2,8.]                 ; Absolute speed value of each layer
;H = [0.,1800.,3200.,5800.,7400.0004,13100.,15800.]   ; Altitude of each layer in meters
;L0 = [12,13,14,15,16,17,18,19]                       ; The outer scale defines to generate the screen
                                                      ; with Von Karman Power Spectrum. (meters)
;Spacing = nsize/8                                    ; The layer are normalised taking into account
                                                      ; a grid of pupil every nsize/8=1024/8 px

; Dummy

RestoreFileDummy = 'example2.dum' ; The example2 dummy star position file is restored
```

```
;
;                    The grid information are now not needed and so commented or ignored
;

;selectstrehl = 0                 ; A square grid of test stars will be created
;nsidedummy = 7                   ; The side of the square will have 7 elements
;gridstep = 20.                   ; The distance between to close star is 20 arcsec
;vdummy=16.                       ; The magnitude of the stars is 16, needed but useless.


; Stellar Field

RestoreFileField = ''                            ; No file is restored
switchasterism = 1                               ; Asterism of reference will be read in the script
xaster = [15.,-28.,42.,-10.,45.,0.,90,0,-90.]    ; X coordinates of 5+4 natural guide stars in arcsec
yaster = [3.,29.,-8,-13.,30.,90.,0.,-90.,0.]     ; Y coordinates of 5+4 natural guide stars in arcsec
vaster = [8.56,8.81,9.02,9.15,9.36,7.2,8.1,8.3,8.0]  ; Magnitude of each references


; Noise data    The following line define the characteristics of WFS. They are needed
;                    but in this case the values are restored, and so can be commented

RestoreFileElectronic = 'example2.elc' ; The example2 WFS information file is restored
;
;                    The grid information are now not needed and so commented or ignored
;

;BandWidth = 0.4d0                ; bandwidth integration (micron)
;Eff = 0.1972d0                   ; quantum efficiency of all the system
;RON = [4.5,3.5]                  ; Read Out Noise variance in electron per pixel per frame for each
                                  ; CCD
;dark= sqrt([500,100])            ; Dark current RMS in electron per pixel per seconds
;pyr_size = 1.0                   ; Dimension of the pyramid in the sky is 1 square arcsec
;sky = 20.0                       ; Sky brightness
;ccdside = [16.,28.]              ; Number of pixel of the side of CCD for each sensor
;fwhmside = 256                   ; The dimension of the array used to compute PSF of test stars is
                                  ; 256 px, two times the npupil value
                                  ; this number is useful when the mode set in sh_mode is not 'fixed'
                                  ; and when switchnoise is 'sh' (Shack-Hartmann)
;lambda_wfs = 0.55                ; Wavefront sensing wavelength in micron is set to 0.55 (V band)
;delay=[2,2]                      ; temporal delay for each loop in deltaT unit
;sh_mode = 'fixed'                ; The Rousset coefficient Nt/Nd is equal to 1.

; Miscellaneous

Path = ''                         ; Path where be placed the output-file
SAVE_ZERNIKE=1                    ; The keyword is set to save the modes coefficients of each WFSs
                                  ; loop
;graphics = 'yes'                 ; Set or not the display of the loop by window during the simulation
```

border = 1	; If set the code considers the numerical effect of the WFS sampling.
iterlong = 5	; Set the 1st iteration from which the long exposure PSF is integrated.
W_iter = 25	; It defines the rate to be used to write the .log file
sr_limit = 0.3	; Set in the graphics displayed the maximum value of SR to be plot
fieldsr = 'yes'	; Not set. If set the long exposure PSF of the GS is also computed.
; RESTOREDM = 'dm.dat'	; The line is commented because no assumption on the mirror mode ; is done. Here Zernike modes are used instead of the mirror modes.
; save = 1	; The atmo file is not saved because the keyword is undefined.
EXTRA_FIELD = 1	; The keyword MUST be set because in the MFoV approach the ; reference can be outside the corrected FoV.
; SILENT = 1	; Not used now

5.5 Installation

The code is easy to install because the user just need to put all the procedures in the work directory. Another possibility is to put all files in a directory and then add its path in the preferences of the IDL main program.

5.6 Calling sequence

The calling sequence by the IDL command line is:
Mcaosim,'script-filename','directory-path'

or if there are keeywords:

mcaosim,'script-filename','directory-path',/KEYWORDS

Under UNIX machine without X-display (only command line) the KEYWORD /NOWIN has to be set.

5.6.1 Example

In order to run the simulation described in the section 5.4.1 the user just have to follow this procedure:

1. Open IDL or IDLDE ;
2. Write in the command line the following : `mcaosim,'example1.txt','',/NOWIN;`
3. Wait for the end of simulation;
4. Look at the output files in the work directory.

Under UNIX machine is possible to use the non-interactive mode by writing the commands:

```
mcaosim,'example1.txt','',/NOWIN
exit
```

in a file (for example call it 'batch.txt') and then in the terminal line write the following:

- `idl batch.txt &`

5.7 General Aspect – Main routines

Once these parameters entered the mcaosim.pro routine (section 6.54) reads and executes the script for the initialisation of the simulation. All the parameters are stored in the files with the .tel, .sys, .dum, .lop, .elc, .fld extensions before to be restored at the beginning of the loopmcao.pro routine (section 6.46) for the simulation of the LO WFS. The simulation computes the AO system correction in closed loop, each cycle comprising the image formation, the computation of the DM commands, the correction and the atmosphere evolution which is not necessarily done with the same time step. The overall schematic of the simulation is given in Figure 9.

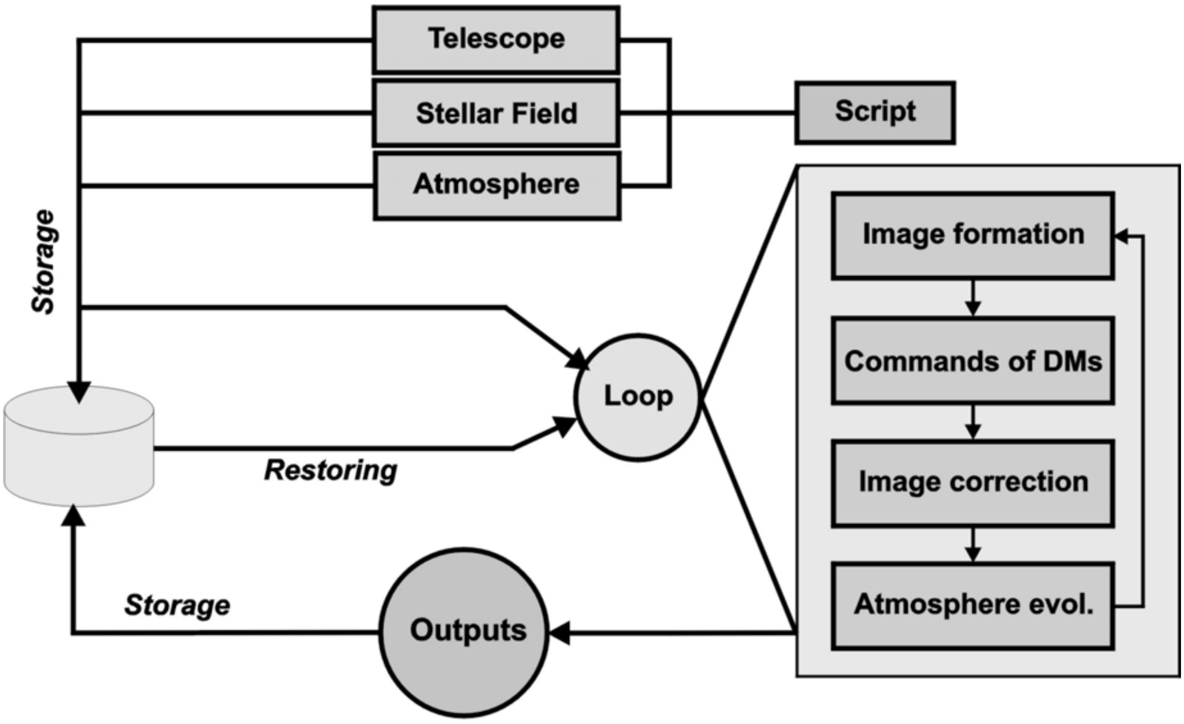


Figure 9: Overall flow chart of the LO WFS simulation in close loop.

The code uses seven main routines to simulated the LO WFS. We briefly describe them before to give the complete syntax and description of all the functions and procedures in Section 6.

5.7.1 Atmosphere selection – Evolution

The atmosphere screens are generated in the initialisation phase. Section 4.1 described the statistic of these phase screens.
During the main loop the atmosphere evolution is simulated by the layers displacement following the wind speed. This evolution is made by the evolveatmosphere.pro procedure (section 6.24) at each time step of the wind evolution. To model a more realistic system the atmosphere evolution time step has been chosen as the time unit and the integration time step is defined for each WFS as an integer factor of the atmosphere evolution time step. The evolveatmosphere.pro routine (section 6.24) is used at the end of each loop.

To optimise computation time the interesting sections of each screens are extracted at each loop by the retail_layer.pro function (section 6.74) before to compute the wavefront of each star using the getwf.pro procedure (section 6.31). The star WF is obtained by adding all the screens sections that introduce aberrations on a given star. This computation is explained in Section 4.2.

5.7.2 The WFS signal - Noise

Once the WF of each star obtained the procedure layerWFS.pro (section 6.45) co-adds the WFS in a LO manner: each WFS signal is computed by superimposing each star WF pondered by the star intensity with a displacement that depends on the altitude of the conjugated plane and on the position of the stars. The integration time being a multiple number of the atmosphere evolution time, several simulations loops can be done before to add the noise to the WFS signal. The procedure add_noise.pro computes only the photon noise using the parameter noise_type=calibration or both photon and detector noises using the parameter noise_type=modal to simulated the noise with a pyramid WFS. Section 4.3 explains the hypothesis we used for the noise computation and section 4.4 describes more precisely the pyramid case.

5.7.3 DM correction – Modal or Zonal

After the WFS signal computation an additional delay is used to simulate the computation delay like in real systems. The delay step is also an integer number of the atmosphere evolution timescale. Finally the correction can be applied to the DM. Two kinds of corrections are possible in the simulation: the procedure modalcorrection.pro (section 6.61) applies the correction using mirror of Zernike modes while the procedure zonalcorrection.pro (section 6.96) computes the DM correction directly from the wavefront signal by changing the sampling of the phase map. The two methods are described in Section 4.5.

5.7.4 Strehl Ratio Estimation

This computation is made independently of the each DM loop. Using a matrix of dummy stars, the Strehl Ratio is computed at each step of the atmosphere evolution. The procedure srcalculation.pro calls the function computePSF.pro to determine the Point Spread Function shape of each dummy star and compute the Strehl Ratio over the Field of View by dividing the maximum of the PSF of each dummy star by the maximum of the diffraction limited PSF.

6 Reference Guide

In this section are reported all the routines used in the simulation by alphabetical order. We indicate for each of them its detailed function, the input and output variables and the keywords if they are. The optional statements are enclosed in square brackets.

6.1 add_noise

The procedure adds to the wavefronts measurement a random noise due to the photon and CCD noises.
The array of measurement and noise must be in the same units (phase).

Syntax:
add_noise, averagewf, footprint, rebinfoot, sampling, ratedm, knoise, ndm, spot, metapupil, indpupil, indsignal, narray, spacedm, pyramid, ron, dark, sky, indnorebinsignal, noise, NOISE_TYPE, znkn, ccdside, nd, Nts

Variables:

- Inputs:

averagewf : wavefronts measurements

footprint : footprint of guide star on the layers
rebinfoot : footprint resized to the binning size defined by spacedm
sampling : yes=1 no = 0
knoise : constant that define the amplitude of noise if NOISE Type equal to 'calibration'
ratedm : temporal integration of each detector (time/deltaT)
ndm : number of DM
spot : spot size of the guide stars
metapupil : the metapupil(s) array(s)
indpupil : indexes where metapupil is defined
indsignal : indexes where signal is defined
narray : define the size of metapupil & footprint array
spacedm : size of the rebinned pupil
pyramid : structure that contains some specific information
ron : Read Out Noise in electron per pixel
dark : root mean square due to dark electrons
sky : intensity of the sky background in photoelectrons per pixel
indnorebinsignal: indexes where the footprint are non-zero
NOISE_TYPE : it defines the way to add the noise : 'calibration' or 'sh' or 'modal'
znkn : cube of normalized Zernike polynomials
ccdside : the number of real pixel composing the CCD
nd : the coefficient Nd of Rousset formula ('sh' case)
Nts : the coefficient Nd of Rousset formula ('sh' case) for every NGS

- Outputs:

noise : noise array

6.2 add_overlap

This procedure finds the overlapping region of two 2D arrays, after ideally superposing the relative positions of a reference point, and adds the overlapping region of the second array to the overlapping region of the first one.

Syntax:
add_overlap, Array1, Array2, R1, R2

Variables:

- Inputs:

Array1, Array2: Input arrays
R1 : 2-components vector of coordinates of the reference point in Array1
R2 : 2-components vector of coordinates of the reference point in Array2

- Outputs:

Array1 : Input Array1 plus the overlapping region of Array2

6.3 arcsec_to_rad

Function allowing to convert a parameter given in arc second into radian unit.

Syntax:

G = arcsec_to_rad (arcsec [, DOUBLEP = doublep])

Variables:
- Inputs:

arcsec : parameter in arc second unit.

- Outputs:

G : parameter in radian unit.

Keywords:
DOUBLEP: for double precision computation.

6.4 array_overlap

This procedure finds the bounds of the overlap region of two 2D arrays by ideally matching the coordinates of a reference point.

Syntax:
array_overlap, size1, size2, r1, r2, lx1, ux1, ly1, uy1, lx2, ux2, ly2, uy2

Variables:
- Inputs:

Size1 : 2-components vector, size of array 1, as returned by size52(array1, /DIM)
Size2 : 2-components vector, size of array 2
R1 : 2-components vector, coordinates of reference pixel in array 1
R2 : 2-components vector, coordinates of reference pixel in array 2

- Outputs:

Lx1, Ux1, Ly1, Uy1: Lower and Upper X- and Y- bounds of intersection in array 1
Lx2, Ux2, Ly2, Uy2: Lower and Upper X- and Y- bounds of intersection in array 2

6.5 Ba_so

This function generates a field of stars using the Bahcall and Soneira model.

Syntax:
Result = ba_so (l1, b1, mag [, A=a])

Variables:
- Inputs:

l1 : galactic latitude
b1 : galactic longitude
mag: magnitude

- Output:

Result: density per square degree

- Keyword:

A : Absorption by the interstellar medium.

6.6 border_effect

This procedure computes each WFS signal. This signal is a superimposition of phases accumulated by each reference star along its line of sight and then superimposed with a displacement related to the altitude of the conjugated plane and to the position of the stars. This procedure updates the ptr_averagewf pointer.

Syntax:
border_effect, averagewf, rebinfoot, indrebinsignal

Variables:

- Inputs:

averagewf : wavefront measurements
rebinfoot : footprint resized to the binning size defined by spacedm
indrebinsignal: pointer to the array of indexes for the footprint

- Outputs:

ptr_averagewf : pointer to WFS measurements (radian)

6.7 calculate_indeces

This procedure is used in the loop of the LO-MCAO code to determine the positions where the sensor gives an output. In this case each matrix is a pupil or a metapupil which corresponds to a given DM conjugated altitude and to a distribution of stars. The sampling is related to the read-out applied to a certain LO WFS. The output is an array where each row is an array of indexes used for the modal correction.

Syntax:
calculate_indeces, InputMask, NMask, spacedm, indeces, sampling, limit, count

Variables:

- Inputs:

inputmask : matrix cube
nmask : variable, number of plane in the cube
spacedm : dimension of the rebinned array
limit : lower limit of illumination over which the meta pixel is considered in the indexes.
sampling : 0 or 1. If sampling equal to 0, the rebinning is not computed (meaning that its value is infinite).
count : counter

- Outputs:

indeces : indexes of the illuminated meta pixels.

6.8 call_makezernike

The procedure initialises all the variables needed for the initialisation phase of the DMs.

Syntax:

Call_MakeZernike, n, ndm, dimenmetapupil, spacedm, narray, indmeta, ptr_rebinmeta, ptr_zernike, ptr_sampledzer, SwitchSampling [, SHAPEDM=shapedm]

Variables:

- Inputs:

n : number of Zernike 's polynomials
ndm : number of conjugated planes (i.e. number of DMs)
dimenmetapupil : diameter of the metapupils in pixels
spacedm : dimension of the rebinned array
narray : dimension of the metapupils array in pixels
indmeta : pointers with the index where the metapupils are defined
ptr_rebinmeta : pointer to cube of rebinned metapupils
SwitchSampling : if 0 no spatial-sampling if 1 spatial sampling is applied

- Outputs:

PTR_zernike : pointer of Zernike 's polynomials
ptr_sampledzer : pointer of Zernike 's polynomials rebinned (if it is specified)

- Keywords:

SHAPEDM : this keyword contains a user-defined set of DM modes.

6.9 call_zernikematrixinversion

The procedure calls the routine which computes the matrix inversion and returns the interaction matrix, the transpose and inverse of the modes and the number of modes considered.

Syntax:
Call_ZernikeMatrixInversion, ptr_sampledzer, indsignal, nzernike, narray, coeff_cut, ptr_alldmIntM, ptr_alldmZtr, ptr_alldmLinv, P, eigenvalues, oreigen [, SILENT=silent] [, LOAD_IM=load_im]

Variables:

- Inputs:

ptr_sampledzer : cube of Zernike polynomials X*Y*Nzernike
indsignal : indexes where the footprint-arrays are not zero (narray-side or spacedm side)
nzernike : number of Zernike polynomials
narray : array dimension of the metapupil
coeff_cut : threshold for eigenvalues cutting

- Outputs:

ptr_alldmIntM : SVD result Interaction_Matrix
ptr_alldmZtr : transpose of the DM modes
ptr_alldmLinv : inverse of the DM modes
P : number of modes considered
eigenvalues : eigenvalues of the interaction matrix
oreigen : eigenvalues of the interaction matrix ordered

- Keywords:

SILENT : if the keyword is null, the name of the file loaded appears on the display window.
LOAD_IM : if the keyword is set the procedure loads the mirror modes used for the MAD simulation.

6.10 centroid

The function computes the centroid of a 2D array, defined as the "centre of mass" of a 2D intensity distribution. The routine can be used only with 2D arrays.

Syntax:
Result = CENTROID(Array)

Variables:

- Inputs:

Array: 2D array

- Outputs:

Result: 2-components floating point vector containing the coordinates of the centroid.

6.11 checkfilename

This function generates a file with a specific name without overwriting existing files: if the file exists already, the routine generates a new name defined by Name + 'new' + counter + extension. The function getfilename.pro must be used with IDL5.4

Syntax:
Result = CHECKFileName (Name, ext [, SPECIFICATION=SPECIFICATION])

Variables:

- Inputs:

Name: a string containing the path and name of the file (say 'c:\simulation\idl\filename')
ext : a string containing the extension (say, '.dat'). Notice the presence of the dot '.' inside the extension

- Outputs:

Result: the filename with a progressive number and its path (say, 'c:\pippo\pluto\topolino\filename23.dat', if files of the form 'c:\simulation\idl\filenameXX.dat' already exist with XX from 0 to 22).

- Keywords:

SPECIFICATION: integer number. If the keyword is done, the function does not search for the first free number to use into the filename. If the keyword is not set, the function begins with a null counter and searches the first nonexistent file to create.

6.12 compute_fwhm

This function computes the FWHM of the star whose PSF is saved in the array PSF. The FWHM is computed in radians by using the plate scale value.

Syntax:
fwhm = compute_fwhm (psf, plate_scale)

Variables:

- Inputs:

psf : PSF array
plate_scale: plate scale on the array containing the guide star PSF

- Outputs:

fwhm : FWHM of the star

6.13 compute_percent

This function determines the radial distance from the centre of the PSF to a given position at which the encircled energy has reached a percentage of the total PSF energy.

Syntax:
result = compute_percent (psf, plate_scale, percent)

Variables:

- Inputs:

psf : PSF array
plate_scale: plate scale on the array containing the guide star PSF
percent : threshold value in percent

- Outputs:

result : dimension of the circle in which the chosen percentage of the total PSF energy is contained.

6.14 compute_variance

The procedure computes the variance and the noise due to R.O.N. and Dark of the CCD and due to the Sky background luminosity in the case of the pyramid.

Syntax:
compute_variance, averagewf, footprint, rebinfoot, sampling, ndm, ccdside, indpupil, indsignal, narray, spacedm, ron, dark, sky, ndpx, indnorebinsignal, sigma_electronic_ron, sigma_electronic_dark, sigma_sky, noise

Variables:

- Inputs:

averagewf : wavefronts measurements
footprint : footprint of guide star on the layers
rebinfoot : footprint resized to the binning size defined by spacedm
sampling : yes = 1 no = 0. If sampling equal to 0, the rebinning is not computed (meaning that its value is infinite).
ndm : number of DMs
ccdside : number of pixel of the side of CCD (square CCD!)
indpupil : indexes where metapupil is defined
indsignal : indexes where signal is defined
narray : size of metapupil & footprint arrays
spacedm : size of the rebinned pupil
ron : Read Out Noise in electron per pixel
dark : RMS due to dark electrons
sky : intensity of the sky background in photoelectrons per pixel

ndpx : dimension in pixel of the diffraction pattern of a sub-aperture
indnorebinsignal : indexes where the footprint are non-zero

- Outputs:

sigma_electronic_ron : RMS due to RON
sigma_electronic_dark: RMS due to DARK
sigma_sky : RMS due to SKY background luminosity
noise : noise array

6.15 compute_wind

This procedure generates the wind directions and speeds.

Syntax:
compute_wind, v, vx, vy, seed

Variables:

- Inputs:

v : absolute value of wind speed vector

- Outputs:

vx : X-wind
vy : Y-wind
seed : seed for the random process.

6.16 computepsf

The function computes the PSF of a star in a field with phase-delay.

Syntax:
PSF = computePSF (psfsize, largeplane, phase)

Variables:

- Inputs:

psfsize : array size used to compute PSF
largeplane: plane array of the phase (wavefront)
phase : phase screen (layer)

- Outputs:

psf : PSF of the star on the array

6.17 Consistencycheck

The function verifies the consistency between data generated using the information included into the script and those obtained by restoration.

Syntax:
Result = consistencycheck (atmodata, telescopedata, systemdata, loopdata, fielddata, elecdata [, ITERLONG = iterlong] [, NOCUT = nocut] [, EXTRA_FIELD = extra_field] [, SILENT = silent])

Variables:

- Inputs:

atmodata : structure of data related to the atmosphere
telescopeData: structure of data related to the telescope
systemdata : structure of data related to the MCAO system
loopdata : structure of data related to the loop
fielddata : structure of data related to the field of reference stars
elecdata : structure of data related to the WF-CCD specifics and photon flux

- Outputs:

Result : a structure containing the value true or false and a message string

- Keywords:

ITERLONG : iteration loop number when it begins to generate long exposure
NOCUT : if set code cuts the not useful layers regions
EXTRA_FIELD: if set code consider also the NGS outside the FoV
SILENT : if set no messages are written.

6.18 continuity_count

This function allows to shift an array of a non integer quantity using a linear interpolation.

Syntax:
new_array = continuity_count (dx, dy, array)

Variables:

- Inputs:

dx : non integer quantity to shift the array in x direction.
dy : non integer quantity to shift the array in y direction.
array : array to be interpolated.

- Outputs:

new_array: shifted array using the linear interpolation.

6.19 coo2d

This procedure defines an 2D matrix of indexes.

Syntax:
coo2d,n,xx,yy

Variables:

- Inputs:

n : dimension of the vector

- Outputs:

xx: x coordinates of the indexes
yy: y coordinates of the indexes

6.20 coordinate

Giving as input a length and assuming that the indexes are equally spread around zero, the function computes a vector of coordinates.

Syntax:
result = coordinate (length)

Variables:

- Inputs:

length : dimension of the vector

- Outputs:

Result: coordinates of the indexes

6.21 count_energy

This function computes the integrated luminosity of a 2D PSF and then determines the best fit of this integrated luminosity with a Gaussian profile centred at the maximum of intensity.

Syntax:
result = count_energy (data, plate_scale, percent)

Variables:

- Inputs:

data : the input array to fit
plate_scale : the plate scale on the array
percent : threshold value in percent

- Outputs:

result : fit of the integrated luminosity profile.

6.22 display_field

This procedure calls the procedure TvField to compute the PSF image of both reference and dummy stars and displays the two fields in different windows.

Syntax:
splay_field, ndm, npupil, epsilon, FoV, Xstar, Ystar, Vstar, WFS, DummyX, DummyY, DummyV, WFSDummy [, TVFACTOR = tvfactor]

Variables:

- Inputs:

ndm : number of DMs
npupil : pupil size of the telescope
epsilon : central obstruction ratio
FoV : field of view
Xstar : x-coordinates of the reference stars
Ystar : y-coordinates of the reference stars
Vstar : magnitudes of the reference stars

WFS : array with the integrated phases of each star
DummyX : x-coordinates of the dummy stars
DummyY : y-coordinates of the dummy stars
DummyV : magnitudes of the dummy stars
WFSdummy : array with the integrated phases of every dummy stars

- Keywords:

TVFACTOR: this variable defines the screen-view dimension

6.23 display_label

Display on the screen images of the loop:

- 2-dimesional image of the conjugated layers, DMs surface, DMs correction
- noise phase map. Instantaneous image of the star's field and
- ;3D surface of the instantaneous SR map.

Syntax:

display_label, npupil, tvfactor, ndm, dimenmeta, narray, wf, ptr_dm, srmap, ratedm, selectstrehl, dummystar, dummyx, dummyy, smallayer, SWITCHNOISE, sampling, noise, step_dm, fielddata [, SR_LIMIT = sr_limit] , ptr_metapupil, ptr_rebinfoot, ptr_footprint

Variables:

- Inputs:

npupil : inner radius in normalized units
tvfactor : magnification factor
ndm : number of DMs
dimenmeta : diameter of the metapupils in pixels
narray : define the size of metapupil & footprint array
wf : array with the integrated phases of each star
ptr_dm : pointer to deformable mirrors
srmap : array of SR in the user-defined directions
ratedm : temporal integration of each detector (time/deltaT)
selectstrehl : way to arrange dummy stars
dummystar : structure containing the dummy stars data (x and y coordinates, magnitude and type of Strehl map)
dummyx : x-coordinates of the dummy stars
dummyy : y-coordinates of the dummy stars
smallayer : layer retailed by the function retail_layer
SWITCHNOISE: parameter to turn off the noise. Can be equal to ‘no’ or ‘modal’
Sampling : yes=1, no=0. If sampling equal to 0, the rebinning is not computed (meaning that its value is infinite).
Noise : noise array
step_dm : matrix containing all the DM corrections for the loop
fielddata : structure containing the star data (number, coordinates and magnitudes)
ptr_metapupil : pointer to array of matrices, each matrix is a metapupil
ptr_rebinfoot : pointer to array of matrices, each matrix is a rebinned footprint
ptr_footprint : pointer to array of matrices, each matrix is a footprint

- Keywords:

sr_limit : it defines the maximum SR of the axis of the 3D plot where is drawn the SR map if not set it is the maximum SR of the map

6.24 evolveatmosphere

The routine simulates the atmosphere evolution of the layers contained in Layer using a linear interpolation of the layers.

Syntax:
evolveatmosphere, nsizex, nsizey, deltat, pixlayer, layer, step, vx, vy, shiftedlayer

Variables:
- Inputs:

nsizex : array size in the x direction
nsizey : array size in the y direction
deltat : integration time
pixlayer : layer sampling
layer : step of the simulation
step : atmospheric layers (first generation)
vx : wind velocity of the layers (x direction)
vy : wind velocity of the layers (y direction)Outputs:

- Outputs:

shiftedlayer : layers shifted

6.25 findnm

Find the radial and azimuthal orders of one Zernike 's polynomial of degree j.

Syntax:
result= FindNM(j)

Variables:
- Inputs:

j: polynomial degree

- Outputs:

result: array containing the radial and azimuthal orders

6.26 frequencies

The function generates an array of frequency.

Syntax:
u = frequencies(nsize, pixlayer)

Variables:
- Inputs:

nsize : size of the layer in pixels
pixLayer : layer scale m/pixel

- Outputs:

u : frequency array

6.27 gauss2to1

The procedure computes an unidimensional Gaussian fit of a bidimensional array. The maximum of PSF is taken the centre of the Gaussian pattern.

Syntax:
gauss2to1, data, plate_scale, fit, a, lineardata, x

Variables:

- Inputs:

data : the input array to fit
plate_scale : the plate scale on the array

- Outputs:

fit : the Gaussian fit
a : vector with the parameters of the Gaussian
lineardata : array with the mean of the data array computed over rings centred at the maximum value
x : the x coordinate used to fit

6.28 generatefield

To initialise the field of reference stars

Syntax:
generatefield, SwitchAsterism, FoV, XAster, YAster, VAster, NLASER = nlaser, laserflag, Galactic_Latitude, Galactic_Longitude, StarsIntegralDensityDataFile, AllStar, Nstar, LimitMagnitudeDiff, FieldData

Variables:

- Inputs:

MFoV : field of view
Xaster : an array with the user-defined x-coordinate of the stars of the asterism
Yaster : an array with the user-defined y-coordinate of the stars of the asterism
Vaster : an array with the user-defined magnitudes of the stars of the asterism
Galactic_Latitude : galactic latitude for the random field of natural guide stars
Galactic_Longitude : galactic longitude for the random field of natural guide stars
Allstar : a switch to select only a limited number of stars between those generated
Nstar : number of stars to select between those generated
LimitMagnitudeDiff: the difference between the faintest magnitude and the brightest allowable one

- Outputs:

FIELDDATALOOP: the initialised structure of data
NLASER : swith on/off the laser.

6.29 getfilename

This function generates a series of files with a progressive number (to use with IDL 5.4, otherwise use the routine checkfilename.pro).

Syntax:
GetFileName, Name, Extension, counter [, SPECIFICATION=SPECIFICATION]

Variables:

- Inputs:

Name : a string containing the path and name of the file (say 'c:\simulation\idl\filename')
Extension : a string containing the extension (say, '.dat'). Notice the presence of the dot '.' inside the extension

- Outputs:

Result : filename with a progressive number and its path (say, 'c:\simulation\idl\filename23.dat', if the files of the form 'c:\ simulation\idl \filenameXX.dat' already exist with XX from 0 to 22).

- Keywords:

SPECIFICATION: integer number. If the keyword is done, the function does not search for the first free number to use into the filename. If the keyword is not set, the function begins with a null counter and searches the first nonexistent file to create.

6.30 Getreferencestars

Select a subset of stars from the generated ones (only for NGS randomly generated)

Syntax:
GetReferenceStars, FieldData, XS, YS, VS, Nstar

Variables:

- Inputs:

Fielddata : structure of data related to the field of reference stars

- Outputs:

XSTAR : array of x-coordinates of the reference stars
YSTAR : array of y-coordinates of the reference stars
VSTAR : array of magnitudes of the reference stars
NSTAR : variable with the number of stars actually used

6.31 getwf

This procedure computes the integrated phase of each reference star.

Syntax:
getwf, npupil, narray, ndm, nlayer, epsilon, Layer, coord, ptr_DM, WFS [, MISMATCH_PUPIL_DM = mismatch_pupil_dm] [, MISMATCH_MASK_DM = mismatch_mask_dm]

Variables:

- Inputs:

npupil : telescope pupil size
narray : define the size of metapupil & footprint array
ndm : number of DMs
nlayer : number of phase screens
epsilon : central obstruction
Layer : phase screens array
coord : xy-coordinates of the reference stars
Ptr_DM : pointer to deformable mirrors

- Outputs:

wfs : integrated phases for each star

- Keywords:

MISMATCH_PUPIL_DM: it sets the displacement of the star's pupils on DMs
MISMATCH_MASK_DM: indicates the displacement of the DM respectively to the layers (in x and y for each DM, in pixel unit).

6.32 graphmad

This procedure allows to plot specific results of a given simulation saved in a logfile. Various keywords can be used to define display settings.

Syntax:
graphmad, lognumber [, NITER = niter] , SMAP[, RATE_DM = rate_dm] [, MAXSR = maxsr] [, OLD = old] [, BW = bw] [, KOL = kol] [, SR_NGS = sr_ngs] [, SET_FOV = set_fov] [, SET_AS = set_as] [, SET_BORDER = set_border] [, SET_LIVELLI = set_livelli]

Variables:

- Inputs:

lognumber: number of the file to restore

- Outputs:

SMAP : SR map

- Keywords:

NITER : number of iterations considered
RATE_DM : 0 or 1. If 1, the Strehl variation during the simulation is plotted
MAXSR : if set the maximum level on the contour is fixed to the maxsr value.
OLD : used to read old file done before summer 2002
BW : for black and white setting
KOL : for colour setting
SR_NGS : if set the simulation searches the file containing the SR evolution of the reference stars and used it to display the SR map.
SET_FOV : define the FoV
SET_AS : define the central coordinate in arcsec
SET_BORDER: define the dimension of the box in arcsec

SET_LIVELLI: define the contour lines.

6.33 highfootprint

The procedure generates the footprint of the reference stars at a given altitude. Every star's footprint value is proportional to the stellar intensity.

Syntax:
highfootprint, narray, npupil, smallpupil, coo, intensity, footprint

Variables:

- Inputs:

narray : layer size in pixels
npupil : pupil size in pixels
smallpupil : shape of the footprint
coo : position of the of the star's footprint centre on the layer
intensity : intensities of the stars

- Outputs:

footprint : resulting footprint

6.34 init_display

Initialise the window to display the loop image and initialise the .mpeg file

Syntax:
init_display, npupil, ndm, pathmovie, unit1, moviefile, tvfactor, mpegId [, NOCALIBRATION = nocalibration] , MAKEANIMATION

Variables:

- Inputs:

Npupil : telescope pupil size
ndm : number of DMs
pathmovie : movie path
unit1 : main output log-file logical number
moviefile : file of the movie with the path and the extension.
tvfactor : magnification factor
mpegId : movie file logical number
makeanimation : f 'mpeg' or 'mpg' then initialise mpeg file. If ‘gif’, initialise a gif file.

- Keywords:

nocalibration : if not keyword set, the routine determines the window dimension using the pupil size.

6.35 init_intensity

This function computes the photon number received from a set of star of magnitude magstar

Syntax:
result = init_intensity (bandwidth, eff, D, epsilon, npupil, lambda, deltat, magstar)

Variables:

- Inputs:

bandwidth: bandwidth in micron
eff : overall quantum efficiency
D : telescope diameter
epsilon : central obstruction ratio
npupil : pupil dimension in pixel
lambda : wavelength in micron
deltat : integration time
magstar : stars magnitude (scalar or vector)

- Outputs:

result : number of photons

6.36 init_mfov

Procedure defining the mfov_struc structure when simulating MCAO systems with the multiple field of view technique.

Syntax:
init_mfov, telescopedata, systemdata, loopdata, fielddata, switch_mfov, mfov_struc

Variables:

- Inputs:

telescopedata: structure containing the aperture, the central obstruction, the field of view, the multiple field of view mode, the azimuthal angle, the layer size, the pupil size, the scientific wavelength and the time step.
systemdata: structure containing the number of DM, the altitude of DM, the gain applied to each DM and the integration time of each DM in DeltaT units.
loopdata: contains the number of steps, the number of steps to do before to close the loop, the noise, the animation file format, the size of the arrays used to compute the PSFs, the limit for valid detection on rebin array, the number f corrected modes for each DM and the coefficient cut limit for eigenvalues of SVD.
fielddata: structure containing the number of reference stars used, the coordinates X and Y in arcsec, their magnitudes and the filename.

- Outputs:

switch_mfov: 0 or 1. If equal to 1, the simulation is in the MFOV way.
mfov_struc : structure which contains the inner and outer FoV for each DM, the parameter mode which is equal to zero when the light of the central FoV is split between the two DMS and equal to 1 if the light of the central FoV is used only by the high DM. The structure contains also the stars placed in the central FoV and the stars placed in the outer ring.

6.37 init_pyr

Initialise the pyramid parameters.

Syntax:

init_pyr, lambda_wfs, lambda, d_r0, D, pyr_size, npupil, epsilon, ndm, spacedm, fwhmside, fwhm_plate_scale, percent, sh_mode, pyramid

Variables:

- Inputs:

lambda_wfs : the sensing wavelength in micron
lambda : the imaging wavelength in micron
d_r0 : vector with the D over r0 value of the layers
D : telescope diameter
pyr_size : dimension of the pyramid un the sky
npupil : dimension of the pupil io pixel
epsilon : central obstruction ratio
ndm : number of DM
spacedm : sampling of the wavefront sensor CCD
fwhmside : side of the array used to compute the FWHM of the guide stars
fwhm_plate_scale: plate scale on the array containing the guide star PSF
percent : energy threshold to compute Nd (if sh_mode is set to "percent")
sh_mode : select the mode to compute SH noise:

- fixed if Nt/Nd = 1
- FWHM if Nt/Nd equal to the ratio of fwhm
- percent if Nt/Nd equal to the ratio of iso-energy count

- Outputs:

pyramid : output structure containing the diameter d, the parameter λ/r_0, λ, the pyramid dimension on the sky, θ_D, N_D,D/r_0, and the number of DMs.

6.38 init_range

It transforms the layers altitudes from meters to pixel.

Syntax:
init_range, hdm, h, pixlayer [, THETA = theta] , rangedm, hlayer

Variables:

- Inputs:

hdm : DMs altitude
h : altitude of the layers
pixlayer: size of the layers in pixel

- Outputs:

rangedm: DMs altitude in pixel
hlayer : layers altitude in pixel

- Keywords:

theta : angle from the zenith.

6.39 Initializedm

Set the targets of pointer that defines the DMs

Syntax:
initializeDM, narray, ratedm, delay, ptr_dm, ptr_corr, ptr_savecorr

Variables:
- Inputs:

narray: layer size in pixels
ratedm: temporal integration of each detector (time/deltaT)
delay : Time delay (in temporal step)

- Outputs:

ptr_corr : pointer to the matrix containing the corrections applied to the DMs.
ptr_savecorr: identical to ptr_corr but when ratedm is smaller than delay.

6.40 Initializeintensity

This function computes the number of photons received from a set of stars of magnitudes magstar.

Syntax:
result = init_intensity (bandwidth, eff, D, eps, npupil, lambda,deltat, magstar)

Variables:
- Inputs:

bandwidth: bandwidth in micron
eff : overall quantum efficiency
D : telescope diameter
eps : central obstruction ratio
npupil : pupil dimension in pixel
lambda : wavelength in micron
deltat : integration time
magstar : stars magnitude (scalar or vector)

- Outputs:

Result : number of photons

6.41 Initializeis

This function sets the targets of pointer that define the DMs

Syntax:
ptr_integratesignal = initializeis(narray)

Variables:
- Inputs:

narray: dimension of the metapupils array in pixels

- Outputs:

ptr_integratesignal: pointer to the signal array

6.42 Initializesr

This function computes the peak-intensity values of the diffraction limited PSF.

Syntax:
result = InitializeSR (npupil, epsilon, psfsize)

Variables:

- Inputs:

npupil : dimension of the pupil in pixels
epsilon: central obstruction diameter
psfsize : dimension of the array

- Outputs:

result : peak-intensity values

6.43 Kolmogorov

Compute the Kolmogorov Power Spectrum.

Syntax:
ps = kolmogorov(u)

Variables:

- Inputs:

u : array of frequency values.

- Outputs:

ps: Kolmogorov power spectrum

6.44 layer_alpha

The following code works when the /Alpha keyword is added in mcaosym when calling this procedure. It overlap some words on the bottom left angle of the layer. If the code works right the word are superposed.

Syntax:
layer_alpha, layer [, THISLAYER = thislayer]

Variables:

- Inputs:

layer : layers

- Outputs:

layer : Same layer but the word superposed

- Keywords:

THISLAYER: select the layer to process

6.45 Layerwfs

Calculation of the signal of each wave front sensor. The signal is here intended as the phases accumulated by each reference star along its line of sight and then superimposed with a displacement that depends on the altitude of the conjugated plane and the position of the stars.

Syntax:

layerwfs, narray, npupil, epsilon, coord, spacedm, intensity, ptr_footprint, ind_norebin_signal, smallpupil, ndm, nstar, wfs, SAMPLING, ptr_averageWF [, BORDER=border] , ptr_rebinfoot, indsignal [, MISMATCH_ST_enl = mismatch_st_enl] [, MISMATCH_MASK_WFS = mismatch_mask_wfs]

Variables:

- Inputs:

narray : layer size in pixels
npupil : telescope pupil size
epsilon : central obscuration ratio
coord : x & y stars positions on layers and on DMs (px)
spacedm : spatial sampling (ex.:8 for the 8x8 case)
intensity : intensities of the guide stars
ptr_footprint : pointer to array of matrices, each matrix is a footprint
ind_norebin_signal: pointer to index ogf the no rebinned footprint
smallpupil : array of the dimension of the pupil with the pupil inside
ndm : number of DMs
nstar : number of NGS
wfs : wavefront sensor measurements
SAMPLING : 0 or 1. If sampling equal to 0, the rebinning is not computed (meaning that its value is infinite).
ptr_rebinfoot : pointer to array of matrices, each matrix is a rebinned footprint
indsignal : pointer to the array of indexes for the footprint

- Outputs:

ptr_averageWF : pointer to WFS measurements (radian)

- Keywords:

BORDER : if set take in to account the border effect (in sampling=1 case)
MISMATCH_ST_enl : it sets the mismatch of the footprint due to the star enlargers
MISMATCH_MASK_WFS: consider the xy shift of the mask on the WFS ([[xy,xy.on_wfs]])

6.46 loopmcao

The procedure extracts the information from the various structures defined in the set-up part. The logfile is defined as well as the variables used in the simulation. The display is initialised before to begin the main loop of the simulation.

Syntax:

loopmcao, TelescopeData, AtmoData, FieldData, DummyStar, SystemData, loopdata, elecdata, unit1, lognumber, srmap, srsave, zernike_save [, GRAPHICS = graphics] [, BORDER = border] [, ITERLONG = iterlong] [, W_ITER = w_iter] [, SR_LIMIT = sr_limit] [, FIELDSR = fieldsr] [, SAVE_ZERNIKE = save_Zernike] [, NOCUT = nocut] [, SHAPEDM = shapedm] [, WFE = wfe] [, SUBZERN = sub_zern] [, SILENT = silent] [, MISMATCH_ST_enl = mismatch_st_enl] [, MISMATCH_PUPIL_dm =

mismatch_pupil_dm] [, MISMATCH_MASK_WFS = mismatch_mask_wfs] [, MISMATCH_MASK_DM = mismatch_mask_dm] [, LOAD_IM = load_IM]

Variables:

- Inputs:

TelescopeData : structure of data related to the telescope
Atmodata : structure of data related to the atmosphere
FieldData : structure of data related to the field of reference stars
DummyStar : structure of data related to the field of dummy stars
Systemdata : structure of data related to the MCAO system
Loopdata : structure of data related to the loop
elecdata : structure of data related to the WF-CCD specifics and photon flux
unit1 : logical number of the unit of the log file
lognumber : suffix to output file

- Outputs:

srmap : last Strehl ratio results
srsave : history of the SR
zernike_save : history of the Zernike coefficients

- Keywords:

BORDER : compute the border effect on WF measurement when the sampling is done.
GRAPHICS : if set make graphics during the loop
ITERLONG : is the iteration loop number when it begins to generate long exposure
W_ITER : is a parameter that defines when write the loopdata in the logfile
SR_LIMIT : when graphics are done set the upper limit to the SR reference value
FIELDSR : compute and write the SR on guide stars
SAVE_ZERNIKE : if set save the modal coefficients of every iteration
NOCUT : if set code cuts the not useful layers regions
SHAPEDM : defines the DM surface to be used (must have the metapupil dimension)
WFE : if set code saves the value of wavefront error of every iteration
SUBZERN : if set the number of mode can be bigger than the number of sub-aperture
SILENT : print action are suppressed
MISMATCH_ST_enl : consider the xy shift of the Star enlargers ([xy,dm,xystar])
MISMATCH_PUPIL_dm : consider the xy shift of the star's pupils DM ([[xy,star,xy.on_dm]])
MISMATCH_MASK_WFS : consider the xy shift of the mask on the WFS ([[xy,xy.on_wfs]])
MISMATCH_MASK_DM : consider the xy shift of the mask on the DM ([[xy,xy.on_dm]])

6.47 Makeasterism

Generation of the field of reference stars (ngs).

Syntax:
make_asterism,L, B, FOV, XStar, YStar, VStar

Variables:

- Inputs:

L : galactic longitude

b : galactic latitude
FoV : field of view

- Outputs:

xstar : x-ccordinate of the ngs
ystar : y-coordinate of the ngs
vstar : magnitudes of the ngs

6.48 makelayer

Generation of the phase screens.

Syntax:
makelayer, nsize, npupil, nlayer, PixLayer, Dr0, Spacing, layer, seed, ps [, CHECK = check] [, SUBS = subs] [, OUTERSCALE = outerscale]

Variables:

- Inputs:

nsize : variable, size of the layer in pixels
npupil : variable, size of the pupil in pixels
nlayer : number of layer
PixLayer: variable, layer scale m/pixel
Dr0 : D/r0 for the variance normalization
Spacing : Parameter to normalise layer with Noll

- Outputs:

Layer : Layers
ps : power spectrum

- Keywords:

Check : if set makes a control on the normalization of layer created
SUBS : add subharmonics to the layers
Outerscale: Use Von Karman Power Spectrum

6.49 makemetadiameter

This function allows to compute the metapupil diameter on a layer located at a specified altitude.

Syntax:
Result = makemetadiameter (npupil, h, angle)

Variables:

- Inputs:

npupil : size of pupil in pixel
h : layer altitude
angle : angular dimension of FoV

- Outputs:

Result : metapupil diameter in pixel.

6.50 makemetapupil

Generate a metapupil.

Syntax:
makemetapupil, npupil, FoV, h, sampling, spaceDM, metapupil, metadiameter, narray

Variables:

- Inputs:

npupil : pupil size
pixelayer : pixel size in arcsec
FoV : Field of view in arcsec
h : conjugated plane altitude

- Outputs:

metapupil : metapupil array
metadiameter : size of the metapupil in pixels

6.51 Makepupil

This function computes the pupil array.

Syntax:
pupil= makepupil (n, d, EPSILON = epsilon)

Variables:

- Inputs:

n: size of array
d: diameter of the pupil

- Outputs:

pupil: array containing the pupil shape.

- Keywords:

EPSILON: central obscuration ratio

6.52 makezernike

Generation of Zernike's polynomials and initialisation of modal reconstruction variables. Application of the spatial-sampling.

Syntax:
MakeZernike, n, dimenmetapupil, spaceDM, narray, indmeta, rebinmeta, zern, sampledzer, SwitchSampling [, DM = dm]

Variables:

- Inputs:

n : number of Zernike 's polynomials
dimenmetapupil : diameter of the metapupil in pixels
spaceDM : dimension of the rebinned array

narray : dimension of the metapupil array in pixels
indmeta : index where the metapupil is defined
rebinmeta : rebinned metapupil
SwitchSampling : if 0 no spatial-sampling if 1 spatial sampling is applied

- Outputs:

zern : Zernike 's polynomials (or user DM modes if DM keyword is set)
sampledzer : pointer of Zernike 's polynomials rebinned (if sampling is specified)

- Keywords:

DM : this keyword contains a user-defined set of DM modes

6.53 mask

Generate the metapupils and footprints of each deformable mirror and wave-front sensor in the simulation. Make the pointer for the vector of indexes where metapupil and footprint are greater then the LIMIT value. Compute the maximum number of Zernike polynomials usable.

Syntax:
mask, npupil, epsilon, FoV, spaceDM, Sampling, fieldcoo, intensity, ndm, rangedm, limit, dimenmetapupil, PTR_metapupil, PTR_footprint, narray, indmeta, indsignal, indpupil, indnorebinsignal, PTR_rebinfoot, PTR_rebinmeta, smallpupil, ptr_rebfootu1, maxzern

Variables:

- Inputs:

npupil : pupil size
epsilon : inner radius in normalized units
FoV : FoV angle in arcsec
spacedm : sampling value
SAMPLING : switch on off the sampling of the arrays. If sampling equal to 0, the rebinning is not computed (meaning that its value is infinite).
fieldcoo : cube with the coordinate of the guide stars on the layers and DMs
intensity : intensities of the guide stars
ndm : the number of DM
rangedm : vector with conjugation altitudes of the deformable mirrors
limit : the minimum value to take in account for consider valid the point on the metapupil

- Outputs:

PTR_footprint : pointer to array of matrices, each matrix is a footprint
PTR_rebinfoot : pointer to array of matrices, each matrix is a rebinned footprint
PTR_metapupil : pointer to array of matrices, each matrix is a metapupil
PTR_rebfoot1 : pointer to array of matrices, each matrix is a footprint normalized (stars have the same intensities)
PTR_rebinmeta : pointer to array of matrices, each matrix is a rebinned metapupil
dimenmetapupil : array with the size of each metapupils
narray : dimension of the metapupils array in pixels
indsignal : pointer to the array of indexes for the footprint
indpupil : pointer to the array of indexes for the pupil

indnorebinsignal: pointer to the array of indexes for the no rebinned signal
smallpupil : array of the dimension of the pupil with the pupil inside
maxzern : array with the maximum number of Zernike for each DM

6.54 mcaosim

This procedure reads and executes a script for the initialisation of the simulation.

Syntax:
mcaosim, scriptfile, pathlog, sr_result [, ALPHA = alpha] [, SPECIFICATION = specification] [, SAVE = save] [, NOCUT = nocut] [, RESTOREDM = restoredm] [, NOWIN = nowin] [, EXTRA_FIELD = extra_field] [, WFE = wfe] [, SILENT = silent] [, LOAD_IM = load_im]

Variables:

- Inputs:

scriptfile : a string containing the path and name of the script
pathlog : a string containing the path of the logfile

- Outputs:

sr_result : a structure with the SR results from the simulation

- Keywords:

ALPHA : Makes a general check of the procedures that work with layer by printing on the layers some characters.
SPECIFICATION : this keyword obliges the code to write every output file with the same suffix defined with the keyword itself.
NOWIN : if this keyword is set the video-graphics are not done.
FIELDSR : computes and writes the SR on guide stars
SAVE : if this keyword is set the atmosphere data is saved in a file
RESTOREDM : defines the file where the DM surface to be used are defined (must have the metapupil dimension)
WFE : if this keyword is set the code saves the value of wavefront error of every iteration
NOCUT : if this keyword is set the code cuts the not useful layers regions
EXTRA_FIELD : if this keyword is set the code considers also the NGS outside the FoV

6.55 meanstdev

Evaluation of the mean standard deviation on the array L as mean of many mini-array with same shape and dimension of the pupil . The centre of two consecutive pupils are separated by as fixed value, Spacing. If Spacing is so great to allow only one pupil then this is putted on the centre of the array, four and more pupil will be putted symmetrically to the centre.

Syntax:
Result = MeanStDev (L, nsize, npupil, spacing)

Variables:

- Inputs:

L : phase screen
nsize : layer size in pixels

npupil : pupil size in pixels
spacing: distance between the centre of the pupils used to evaluate the standard deviation in pixels.

- Outputs:

Result : standard deviation

6.56 mfov_getwf

Compute the integrated phase of each star in the case of Multiple field of view.

Syntax:
mfov_getwf, mfov_struc, npupil, narray, ndm, nlayer, epsilon, Layer, coord, ptr_DM, fielddata, wfs_inner, wfs_outer

Variables:

- Inputs:

mfov_struc: structure containing the initial MFOV parameters
npupil : telescope pupil size
narray : dimension of the metapupils array in pixels
epsilon : inner radius (norm. units)
pixlayer : layer sampling
ndm : number of DM
nlayer : number of phase screens
layer : phase screens
coord : xy-coordinates of the reference stars
ptr_DM : pointer to deformable mirrors
fielddata : structure of data related to the field of reference stars

- Outputs:

wfs_inner : integrated phases for each star in the internal FOV
wfs_outer : integrated phases for each star in the external FOV

6.57 mfov_layerwfs

Calculation of the signal of each wave front sensor the signal is here intended as the phases accumulated by each reference star along its line of sight and then superimposed with a displacement that depends on the altitude of the conjugated plane and the position of the stars. In the case of Multiple field of view.

Syntax:
mfov_layerwfs, mfov_struc, narray, npupil, epsilon, coord, spacedm, intensity, ptr_footprint, ind_norebin_signal, smallpupil, ndm, nstar, wfs_inner, wfs_outer, SAMPLING, ptr_averageWF [, BORDER = border] , ptr_rebinfoot, indsignal

Variables:

- Inputs:

mfov_struc : structure containing the initial MFOV parameters
narray : dimension of the metapupils array in pixels
npupil : telescope pupil size
epsilon : central obstruction ratio
pixlayer : pixels spacing (m/pixel)

wfs_inner : integrated phases (radian) for the stars in the central FoV
wfs_outer : integrated phases (radian) for the stars in the external FoV
coord : array, xy-coordinate of the stars (arcsec)
ndm : number of DM
sampling : Set or not binning. If sampling equal to 0, the rebinning is not computed (meaning that its value is infinite).
spacedm : Rebin of the metapupils
intensity : Guide star's intensities
nstar : number of guide stars
ptr_footprint : pointer to the no rebinned footprint
ind_norebin_signal : pointer to index of the no rebinned footprint
smallpupil : array containing the pupil
ptr_rebinfoot : pointer to array of matrices, each matrix is a rebinned footprint
indsignal : pointer to the array of indexes for the footprint

- Outputs:

ptr_averagewf : pointer to the WFS measurements (radian)

- Keywords:

BORDER : Correct for the wrong effect of the rebin procedures

6.58 mfov_mask

Generate the metapupils and footprints of each deformable mirror and wave-front sensor in the simulation.

Syntax:
mfov_mask, npupil, epsilon, FoV, spaceDM, Sampling, fieldcoo, intensity, ndm, rangedm, limit, mfov_struc, dimenmetapupil, PTR_metapupil, PTR_footprint, narray, indmeta, indsignal, indpupil, indnorebinsignal, PTR_rebinfoot, PTR_rebinmeta, smallpupil, ptr_rebfootu1, maxzern

Variables:

- Inputs:

npupil : telescope pupil size
epsilon : inner radius in normalized units
FoV : FoV angle in arcsec
spacedm : sampling value
SAMPLING : switch on off the sampling of the arrays. If sampling equal to 0, the rebinning is not computed (meaning that its value is infinite).
fieldcoo : cube with the coordinate of the guide stars on the layers and DMs
intensity : intensities of the guide stars
ndm : the number of DM
rangedm : vector with conjugation altitudes of the deformable mirrors
limit : the minimum value to take in to account for consider valid the point on the metapupil

- Outputs:

PTR_footprint : pointer to array of matrices, each matrix is a footprint
PTR_rebinfoot : pointer to array of matrices, each matrix is a rebinned footprint
PTR_rebfoot1 : pointer to array of matrices, each matrix is a footprint normalized (stars has the same intensities)

PTR_rebinmeta : pointer to array of matrices, each matrix is a rebinned metapupil
dimenmetapupil : array with the size of each metapupils
narray : dimension of the metapupils array in pixels
indsignal : pointer to the array of indexes for the footprint
indpupil : pointer to the array of indexes for the pupil
indnorebinsignal: pointer to the array of indexes for the no rebinned signal
smallpupil : array of the dimension of the pupil with the pupil inside
maxzern : array with the maximum number of Zernike for each DM

6.59 mism_mask

This procedure shifts the array in input.

Syntax:
mism_mask, layerwf, MISMATCH_MASK, narray [, NOT_INTERPOLATION = not_interpolation]

Variables:

- Inputs:

narray : dimension of the metapupils array in pixels
layerwf : array to shift
MISMATCH_MASK : it sets the mismatch

- Outputs:

layerwf : array shifted

- Keywords:

NOT_INTERPOLATION : If set does not allow the interpolation for the not integer part.

6.60 mism_mask_wf

This procedure shifts the array in input.

Syntax:
MISM_MASK, layerwf, MISMATCH_MASK, narray

Variables:

- Inputs:

MISMATCH_MASK: it sets the mismatch
layerwf : array to shift
narray : dimension of the metapupils array in pixels

- Outputs:

layerwf : array shifted

6.61 modalcorrection

Calculation of the correction applied to the deformable mirrors.

Syntax:

modalcorrection, narray, nzernike, signal, index, ArrayZNK, Interaction_Matrix, Ztr,Coeff, corr, invZtZ_Zt

Variables:

- Inputs:

narray : dimension of the metapupils array in pixels
nzernike : number of polynomials
signal : signal from wavefront sensor
index : index where metapupil is defined
arrayZNK : cube of arrays where are defined the zernike poynomials
interaction_matrix: interaction matrix
Ztr : the transpose of the Zernike matrix with the zernike's polynomials used for the fit of the data
invZtZ_Zt : inverse matrix of the matricial product of the (transpose Zernike matrix ## Zernike) ## transpose Zernike matrix

- Outputs:

corr : matrix with the correction applied to the DM

6.62 modalnoise

Modal noise calculation.The procedure computes the variance due to every Zernike mode and multiplies it (as it is a coefficient) for the correspondent Zernike polynomials that has its variance normalized to 1. The value of the coefficients is extrapolated from Ragazzoni & Farinato 1999.

Syntax:
modalnoise, znkn, dr0, sigmaph, gain, noise

Variables:

- Inputs:

znkn : cube of Zernike polynomials with variance over the metapupil equal to 1
dr0 : Diameter over r0
sigmaph : the variance due to photons equal to one over sqrt(footprint array)
gain : gain of the pyramid: d/r0 where d is given by d = lambda/spot in pixel

- Outputs:

noise : noise array

6.63 move_dm

This function computes the new shape of a specific DM using the formula :
DM+GainDM*corr*metapupil.

Syntax:
Result = move_DM (ptr_DM, GainDM, ptr_corr, ptr_metapupil [, MISMATCH_MASK_DM = mismatch_mask_dm])

Variables:

- Inputs:

ptr_dm : pointer to deformable mirrors
GainDM : the gain of each DM
ptr_corr : pointer to the corrections applied to the DMs
ptr_metapupil : pointer to array of matrices, each matrix is a metapupil

- Outputs:

Result : DM shape updated

- Keywords:

MISMATCH_MASK_DM:

6.64 ntoperation

Select the way to compute Nt coefficients.

Syntax:
nt_operation, sh_mode, psfgs, fwhm_platescale, percent, NTs

Variables:

- Inputs:

sh_mode : select the mode to compute SH noise:

  - Fixed if Nt/Nd = 1
  - FWHM if Nt/Nd is equal to the ratio of FWHM
  - percent if Nt/Nd is equal to the ratio of iso-energy count

psfgs : cube matrix containing the PSF of NGS measured on the WF sensors
fwhm_platescale : plate scale
percent : threshold value in the percent mode

- Outputs:

NTs : Nt values for every NGS

6.65 phase_of_complex

This function allows to compute the phase of a complex number.

Syntax:
phi= phase_of_complex(z)

Variables:

- Inputs:

z : complex number

- Outputs:

phi: phase

6.66 phase_of_real

This function computes the phase of a real number.

Syntax:

phi= phase_of_real(n)

Variables:

- Inputs:

n : real number

- Outputs:

phi: phase

6.67 pix_layer

Determine the pixel size for a layer.

Syntax:

Result = pix_layer (D, npupil)

Variables:

- Inputs:

D : diameter of the telescope
npupil : dimension of the telescope pupil in pixels

- Outputs:

Result : pixel size

6.68 platescale

Compute the plate scale

Syntax:

Result= platescale (nside, npupil, wavelength, D)

Variables:

- Inputs:

Nside : dimension for the layer to compute PSFs
npupil : dimension of the telescope pupil in pixels
Wavelength: WF sensing wavelength
D : telescope diameter

- Outputs:

Result: plate scale

6.69 projection_dist

It computes the projection of the star on the plane at the Altitude h. The distances are from the centre of the array.

Syntax:

Result= projection_dist (x, h [, NOCONVERSION=noconversion])

Variables:

- Inputs:

x: position on the layer in arcsec o rad according to NO_CONVERSION keyword
h: plane altitude

- Outputs:

Result: distance from the centre of a projection

- Keywords:

NOCONVERSION: if the keyword is set the functions returns the distance instead of arc seconds.

6.70 psf_ngs

Compute the PSF of the reference stars

Syntax:
Result = psf_ngs (nstar,ndm,fwhmside,spacedm,npupil,wfs,lambda_sc,lambda_wfs)
Variables:

- Inputs:

Nstar : number of stars
ndm : number of DMs
fwhmside : Number of pixel of the array used to compute fwhm of the guide stars
spacedm : spatial integration of each detector (pix)
npupil : telescope pupil size in pixel
wfs : WFS measurements
lambda_sc : scientific wavelength
lambda_wfs: wavefront sensing wavelength

- Outputs:

Result : PSF of the guide stars at the scientific wavelength

6.71 Psfourier

Compute the Fourier transform of an array defined by its Power Spectrum and its Phase.

Syntax:
func = PSFourier(nsize,PowSp,Phase)

Variables:

- Inputs:

nsize : dimension of the array
PowSp : Power Spectrum
Phase : Phase

- Outputs:

func : Fourier transformed function

6.72 pyr_noise

Generation of the noise for pyramidal wavefront sensor. This function computes an error noise that is random and that has the property to be small for the high frequency and via biggest for the low frequency. The value of the variance of the noise over the pupil, regarding to the tip tilt mode, is normalized according to Ragazzoni & Farinato 1999.

Syntax:
pyr_noise, nsize, metapupil, D, dr0, lambda, seed, ptr_rebinfoot, intensity, ndm, sampling, spacedm, noise [, PS = ps]

Variables:

- Inputs:

nsize : dimension of the array used to compute FFT
metapupil : dimension in pixel of the metapupil
D : Diameter of the telescope
dr0 : Diameter over r0
lambda : wavelength
seed : seed for the random process
ptr_rebinfoot : pointer to array of matrices, each matrix is a rebinned footprint
intensity : number of photoelectrons from the star (footprint)
ndm : number of DMs
sampling : 0 or 1. If sampling equal to 0, the rebinning is not computed (meaning that its value is infinite).
spacedm : spatial integration of each detector (pix)

- Outputs:

noise : the noise array

- Keywords:

PS : if you want to preset the power spectrum

6.73 random_phase

Generate an array of random phase values.

Syntax:
Phi = random_phase (n, seed)

Variables:

- Inputs:

n : size of the phase array
Seed: seed for the random process

- Outputs:

Phi : array of phase values.

6.74 retail_layer

Resize the layer leaving only the useful part (reduce computation time).

Syntax:
Result = retail_layer (nsizex, nsizey, h, r, npupil, layer)

Variables:

- Inputs:

nsizex: x dimension of the layer
nsizey: y dimension of the layer
h : altitude of the layer
r : projected distance of the farther star from the reference point
npupil: telescope pupil diameter
layer : phase screens matrix

- Outputs:

Result: resized phase screens matrix

6.75 savedm

This procedure saves the mirror shape at each loop by adding the new shape to the previous ones.

Syntax:
savedm, saved, corr

Variables:

- Inputs:

saved: matrix containing all the previous mirror shapes.
corr : mirror shape

- Outputs:

saved: matrix containing all the mirror shapes of a specific mirror.

6.76 select_sh_mode

Select and initialise the SH noise generation fashion.

Syntax:
select_sh_mode, percent, switchnoise, sh_mode, FIELDSR, knoise, loopdata, systemdata

Variables:

- Inputs:

percent : energy threshold to compute Nd (if sh_mode is set to "percent")
switchnoise : variable used to set or to unset the noise computation (values 'no', 'sh', 'calibration')
sh_mode : select the mode to compute SH noise:

  - Fixed for Nt/Nd = 1
  - fwhm for Nt/Nd equal to the ratio of fwhm
  - percent if Nt/Nd equal to the ratio of iso-energy count

FIELDSR : if 'yes' evaluate the NGS SR
knoise : calibration factor used if switchnoise = 'calibration'
loopdata : Structure containing the loop data

systemdata : Structure containing the system data

6.77 sh_noiser

Compute the variance and the noise due to Photon noise, R.O.N. and Dark of the CCD and for the Sky background luminosity. In the case of the SH WFS. Here are used the coefficients defined by Rousset.

Syntax:

h_noiser, averagewf, footprint, rebinfoot, sampling, ndm, ccdside, nt, indpupil, indsignal, narray, spacedm, ron, dark, sky, nd, ndpx, indnorebinsignal, sigma_photon, sigma_electronic_ron, sigma_electronic_dark, sigma_sky, noise

Variables:

- Inputs:

footprint : footprint of guide star on the layers
rebinfoot : footprint resized to the binning size defined by spacedm (rebinned footprint array)
sampling : yes=1 no = 0. If sampling equal to 0, the rebinning is not computed (meaning that its value is infinite).
ccdside : number of true pixel of the side of CCD (square CCD!)
ndm : number of DM (or WFS! if MFOV)
coeff : coefficient N_t / N_d
indpupil : indexes where metapupil is defined (rebinned or not according to sampling)
indsignal : indexes where signal is defined (rebinned or not according to sampling)
narray : define the size of metapupil & footprint array
spacedm : size of the rebinned pupil (number of binned-pixel in the side of the binned pupil array)
ndpx : dimension in pixel of the diffraction pattern of a sub-aperture (micro-lens)
ron : Read Out Noise in electron per pixel
dark : RMS due to dark electrons
sky : intensity of the sky background in photoelectrons per pixel
pyramid : structure containing the noise data
indnorebinsignal : indexes where the footprint are non-zero

- Outputs:

sigma_photon : RMS due to Photon noise (Poissonian)
sigma_electronic_ron : RMS due to RON
sigma_electronic_dark: RMS due to DARK
sigma_sky : RMS due to SKY background luminosity
noise : noise array

6.78 size52

Alias of IDL 5.2 intrinsic SIZE function for previous versions.

Syntax:

Result = size52 (x [, N_DIMENSION = ndim] [, DIMENSION = dim] [, TYPE = type])

Variables:

- Inputs:

X: IDL variable

- Outputs:

Result: Same as output of SIZE if no keyword is set. Otherwise returns the result specified by the KEYWORD.

- Keywords:

N_DIMENSION: Set this keyword to a nonzero value to retrieve the number of dimensions of X
DIMENSION : Set this keyword to a nonzero value to retrieve a long-integer vector containing the size of each dimension of X
TYPE : Set this keyword to a nonzero value to retrieve the IDL type code of X

6.79 SpatialSampling

Samples an image. The image obtained is the interpolated to the dimension of the input image and the result is returned.

Syntax:
Result = SpatialSampling (Step, Image, N)

Variables:

- Inputs:

Step : rebinning applied to the image (ex. 5x5, 6x6, 8x8 etc.)
Image: the image to be sampled
N :

- Outputs:

Result: sampled image

6.80 srcalculation

Calculation of the SR for MCAO-LO simulation.

Syntax:
SRCalculation, npupil, epsilon, psfsize, wfsdummy, DiffLimOnAxis, indpupil, selectstrehl, spacedm, lambda_sc, lambda_wfs [, VARSR = varSR] [, GRAPHICS = graphics] , srmap, psf, iterlong [, SAVEPSF = savepsf] , fwhmside, DiffLimWFS, sh_mode [, FIELDSR = fieldsr] , fielddata, wfs, sr_field, switchnoise, psfgs, sr_inst, srngs_inst, sr_field_WFS

Variables:

- Inputs:

NPupil : pupil size
Epsilon : inner radius in norm. units
psfsize : dimension to compute PSF by FFT
wfsdummy : wavefront of the dummy stars
DiffLimOnAxis : diffraction limited value of the images of the stars on axis
indpupil : index of the point where pupil is defined
selectstrehl : way to arrange dummy stars
spacedm : sampling of the WF sensors
lambda_sc : imaging wavelength
lambda_wfs : WF sensing wavelength

iterlong : threshold iteration to start to compute log exposure PSFs
fwhmside : dimension to compute PSF by FFT for WF sensing purposes
DiffLimWFS : diffraction limited value on axis of the NGS seen by SH-WFS lenslet
sh_mode : way to simulate SH WF sensor
fielddata : structure with the x & y position of the NGS
wfs : wavefront of the NGS
switchnoise : select the noise type

- Outputs:

SRmap : array of SR in the user-defined directions
psf : PSF of the dummy stars
srfield : SR of the NGS if FIELDSR = 'yes'
sr_inst : instantaneous SR of dummy stars
srngs_inst : instantaneous SR of NGS
sr_field_WFS : SR of NGS seen by the SH-WFS lens-let

- Keywords:

VARSR : compute SR using exp(-sigma^2)
GRAPHICS : if set prepare array for graphics purposes
SAVEPSF : save dummy stars PSF to array
FIELDSR : if 'yes' then compute NGS SR

6.81 strehlfield

Calculation of the Strehl ratio in a field of dummy stars.

Syntax:
strehlfield, nsize, npupil, epsilon, wfs, vstar, OnAxis_Obs_Star

Variables:

- Inputs:

nsize : array size
npupil : pupil size
epsilon: inner radius in normalized units
wfs : array of matrices: each matrix is the integrated phase corresponding to the direction of one star.
vstar : array with the magnitude of the stars

- Outputs:

onaxis_obs_star : array with the on axis value of the image of each star.

6.82 sub_armonics

Compute the subharmonics of the frequencies used in a layer generation via FFT.

Syntax:
subs = sub_armonics(nsize,f_sampling,ntot,seed,K_0=k_0)

Variables:

- Inputs:

nsize : size of the layer in pixels

f_sampling: dimension of the layer in meters
ntot : number of subharmonics to consider
seed : seed for random generation of the phase

- Outputs:

subs : the subharmonics

- Keywords:

K_0 : if set defines the outer scale frequency of Von Karman Power Spectrum

6.83 subs_to_coord

Convert array subscripts to pixel coordinates.

Syntax:
SUBS_TO_COORD, Subs, N_columns, X, Y

Variables:

- Inputs:

Subs : Long integer vector of array subscripts
N_columns: Number of columns in the 2D array

- Outputs:

X, Y : Column and row coordinates of pixels subscripted by Subs

6.84 tt_shift

This function compute shift of the FWHM of the star whose PSF is save in the array PSF. The FWHM is computed in radians by using the plate scale value.

Syntax:
rho = tt_shift (psf, plate_scale)

Variables:

- Inputs:

psf : PSF array
plate_scale: plate scale on the array containing the guide star PSF in arcsec / pixel

- Outputs:

rho : shift of the star

6.85 tvfield

Calculation of the image of each reference star

Syntax:
TvField, NPupil, Epsilon, tvfactor, FoV, XStar, YStar, VStar, WFS, Field, PSF = psf

Variables:

- Inputs:

npupil : inner radius in normalized units
epsilon : central obstruction ratio
tvfactor : magnification factor
FoV : field of view
Xstar : x-coordinates of the reference stars
Ystar : y-coordinates of the reference stars
Vstar : magnitudes of the reference stars
WFS : array with the integrated phases of each star

- Outputs:

field : field of stars

- Keywords:

PSF : If set routine shows the PSF it contains

6.86 von_karman_var

Compute the phase variance using the von Karman power spectrum and the outer scale. It substitutes the old vonkarman function.

Syntax:
Result = von_karman_var (D,r0,L0,TIPTILT=tiptilt)
Variables:

- Inputs:

D : Telescope diameter
r0 : r_0 in meter at the wavelength lambda (scientific)
L0 : Outerscale length in the same unit of D

- Outputs:

result : the phase variance

- Keywords:

TIPTILT : Not used for realistic phase screen.

6.87 vonkarman

Compute the Von Karman Power Spectrum

Syntax:
ps = vonkarman(u,L0)

Variables:

- Inputs:

u : array of frequency value
K0 : Outer scale frequency

- Outputs:

ps : von Karman power spectrum

6.88 write_log

It writes in the main output log
file the main computed simulation
parameters

Syntax:
write_log, unit, dimenmetapupil, intensity, maxzern, p, switchmodal, plate_scale [, FWHM_PLATESCALE = fwhm_platescale] , pyramid

Variables:

- Inputs:

unit : logic unit number of output file
dimenmetapupil : diameter of the metapupil in pixels
intensity : NGS intensity by pixel
maxzern : number of the sub-aperture used to compute the WF
p : number of modes used to compute the interaction matrix
switchmodal : 0 in the case of zonal reconstruction 1 for modal one
plate_scale : plate scale for imaging PSF
pyramid : structure containing the data relevant for the pyramids

- Keywords:

FWHM_PLATESCALE : this keyword contains the value of the WF-sensing PSF (if used)

6.89 write_loop

Write information about the loop to a file

Syntax:
write_loop, unit1, i, srmap, ndm, ptr_averagewf, indsignal [, W_ITER = w_iter] [, FIELDSR = fieldsr] , OnAxis_ref_Star, selectstrehl, srngs_inst, sr_field_WFS, NTs, pyramid, wfe_save [, WFE = wfe]

Variables:

- Inputs:

unit1 : logical number of the unit of the log file
i : iteration number
srmap : SR map values
ndm : number of DMs
ptr_averagewf : pointer to the WFS measurements
indsignal : pointer to the index where WFS measurements are defined
OnAxis_ref_Star: Intensity on axis
selectstrehl : SR map mode
srngs_inst : instantaneous SR
sr_field_WFS : SR of NGS at sensing wavelength
NTs : Nt value
pyramid : Structure that defines pyramid
wfe_save : value of wavefront error of every iteration

- Keywords:

W_ITER : is a parameter that defines when write the loopdata in the logfile

FIELDSR : compute and write the SR on guide stars
WFE : if set code saves the value of wavefront error of every iteration

6.90 write_modal_log

Syntax:
write_modal_log, unit1, switchModal, nzernike, coeff, i [, W_ITER = w_iter]

Variables:

- Inputs:

unit1 : logical number of the unit of the log file
switchModal : 0 in the case of zonal reconstruction 1 for modal one
nzernike : number of polynomials
coeff : array of coefficients
i : iteration number

- Keywords:

W_ITER : is a parameter that defines when write the loopdata in the logfile

6.91 xy_on_dm

Compute the xy position of the star footprint centre on each plane (layer & DM)

Syntax:
result = xy_on_dm (hlayer, hdm, xstar, ystar)

Variables:

- Inputs:

hlayer : layers altitude in pixel
hdm : DMs altitude in pixel
xstar : x star coordinate (arcsec)
ystar : y star coordinate (arcsec)

- Outputs:

Result : projected distance between the reference point and the star considered.

6.92 Zernike

Calculate the Zernike 's polynomial of degree j in the points (x,y)

Syntax:
out = zernike (var1, arr1, arr2)

Variables:

- Inputs:

var1 : polynomial degree
arr1 : array with the x (Cartesian) coordinates of the points
where the polynomial should be evaluated

arr2 : array with the y (Cartesian) coordinates of the points
where the polynomial should be evaluated

- Outputs:

out : array with the radial and azimuthal orders

6.93 zernikecoeff

Calculation of the coefficients in modal correction

Syntax:
ZernikeCoeff, Nzernike, indeces, DimenPupil, Interaction_Matrix, Ztr, Detection, invZtZ_Zt, C

Variables:

- Inputs:

Nzernike : number of polynomials
indeces : array of indexes for the sampled signal used to compute interaction matrix
dimenpupil : size of the unsampled metapupil
interaction_matrix : interaction matrix
Ztr : array with the transposed Zernike 's polynomials used for the figure of the dm
invZtZ_Zt : inverse matrix of the matricial product of the (transpose Zernike matrix ## Zernike) ## transpose Zernike matrix
detection : sampled signal

- Outputs:

C : coefficients of the SVD inversion

6.94 zernikecorrection

Calculation of the correction applied to the deformable mirrors using the modal expansion

Syntax:
Result = zernikecorrection (nzernike, arrayznk, coeff)

Variables:

- Inputs:

Nzernike: number of polynomials
arrayznk : array with the Zernike 's polynomials used for the figure of the dm
coeff : array of coefficients

- Outputs:

Result : matrix with the correction applied to the DM

6.95 zernikematrixinversion

Procedure for the inversion of the matrix of polynomials
;
Syntax:

ZernikeMatrixInversion, ptr_sampledzer, indsignal, nzernike, narray, coeff_cut, ptr_alldmIntM, ptr_alldmZtr, ptr_alldmLinv, P, eigenvalues, oreigen [, SILENT = silent]

Variables:

- Inputs:

ptr_sampledzer : cube of Zernike polynomials X*Y*Nzernike
indsignal : indexes where the footprint-arrays are not zero (narray-side or spacedm side)
nzernike : number of Zernike polynomials
narray : array dimension of the metapupil
coeff_cut : threshold for eigenvalues cutting

- Outputs:

ptr_alldmIntM : SVD result Interaction_Matrix
ptr_alldmZtr : Transpose of the DM modes
ptr_alldmLinv : inverse
P : number of modes really considered
eigenvalues : eigenvalues of the interaction matrix
oreigen : eigenvalues of the interaction matrix ordered

- Keywords:

Silent: if the keyword is null, the name of the file loaded appears on the display window.

6.96 Zonalcorrection

Calculation of the correction applied to the deformable mirrors

Syntax:
zonalcorrection, narray, signal, sampling, Corr

Variables:

- Inputs:

narray : define the size of metapupil & footprint array
Signal : WFS signals
Sampling: 0 or 1. If sampling equal to 0, the rebinning is not computed (meaning that its value is infinite).

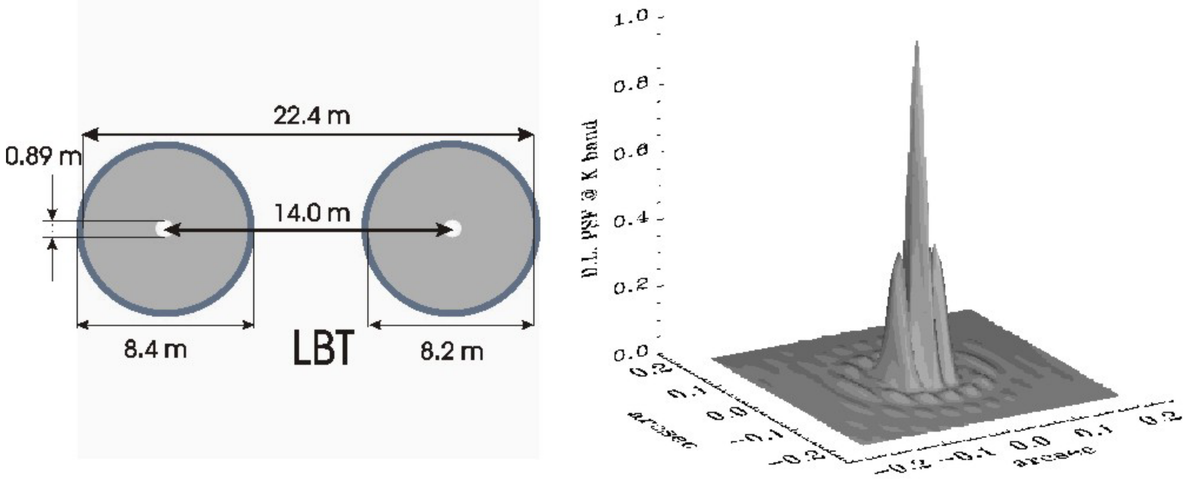
- Outputs:

Corr : matrix with the correction applied to the DM

7 Interferometry with LOST

In the LINC NIRVANA framework we implemented the procedures to simulate a binocular system, such as the Large Binocular Telescope (LBT), with an interferometric focal station.
These proceduere simulate single reference AO, LO MCAO, and Multiple Field of View (MFoV) system mounted on the two arm of the binocular telescope.
The main routine is called nirvana as the instruments it was planned to simulate. The interferometric features have been implemented to simulate the case of the LBT telescope with two independent 8-meters class telescopes on the same mount, fig.\ref{arcidiacono9-10}. The two systems run in parallel using the same parameters but looking to their own portion of the same layers. The two wave-fronts of each test star

are collected in a single array (accordingly to the specific dimension of the LBT, Figure 10) and used to compute the interferometric PSF of the stars by performing FT. As in the single channel case, after a number of iterations defined by the user the long exposure PSFs start to be integrated collecting the PSFs computed for each step of the simulation. These instantaneous PSFs are used also to compute the instantaneous interferometric SR (defined as the ratio between the central values of the PSF computed and of the PSF in the diffraction limited case). However a complete interferometric feature will be reached by a future specific development of the code, in order to ensure a more accurate and realistic results, including, for instance, the detailed behavior of the fringe tracker (Straubmaier et al., 2003).



(a) (b)

Figure 10 In the (a) picture are shown the specific dimension of LBT. The size of the primary seen by the secondary is 8.2 m instead of the overall 8.4 diameter. In (b) the inteferometric diffraction limited PSF at K-band is shown.

To date, LOST assumes that piston term between the two arms is corrected by the system excepting small, random errors.

8 List of the Interferometric routines

8.1 Combinewf

It puts the wave-front of the probe and field star in unique array according to the LBT geometry. It assumes the piston perfectly corrected for piston-guide-star between the two arms and it eventually adds the piston error to the star wave-fronts. The input array containing the two WF is updated only. So to add the two WFs this procedure must be called twice.

Syntax:

combine_wf, npupil,wf, piston_refdata, channel, nstar,maschera, index, combined_WF, PISTON_ERROR = piston_error,pis_value,piston

Variables:

- Inputs:

NPupil	: pupil size
wf	: wf to be add in the array
piston_refdata	: piston correction relative to a piston reference star
channel	: 0:left side 1: right side
nstar	: number of stars
maschera	: array with the binocular mask (0:black 1:position of the two pupils)
index	: index where the "maschera" array is defined (equal to 1)

combined_WF : the array containing the two sides of WF
pis_value : value of the piston error that is applied if the KEYWORD PISTON_ERROR is set

- Outputs:

combined_wf : the array containing the two sides of WF
piston : array, with the piston values for each star

- Keywords:

PISTON_ERROR : if defined the pis_value value is applied to the left side only (0 is left channel)

8.2 Initialize_nirvana_dm

This function sets the targets of pointer that defines the DMs
Syntax:
initialize_nirvana_DM,narray,ratedm,delay,ptr_dm,ptr_corr,ptr_savecorr
Variables:

- Inputs:

narray : integer, dimension of the array containing the DM
ratedm : array, integration time [in iteration units]
delay : delay between the end ofthe measurement and the correction application
ptr_dm : pointer to the DM
ptr_corr : pointer to the correction computed
ptr_savecorr : pointer to the correction saved

- Outputs:

The pointers are updated.

8.3 Initialize_nirvana_sr

Initialization of the variables used for the calculation of the strehl ratio.
The function return the max value of the interferometric diffraction limited PSF.
Syntax:
result = Initialize_nirvana_sr(npupil,epsilon,psfsize)
Variables:

- Inputs:

npupil : dimension of the pupil in pixels
epsilon : obstruction
psfsize : dimension of the array used for FFT generation of the PSF

8.4 Initializeis_nirvana

This function set the targets of pointer that defines the integrated signal of the WFS.
Syntax:
result = initializeIS_nirvana(narray)
Variables:

- Inputs:

narray : dimension of array containing the signal integrated by the WFS

- Outputs:

none

- Keywords:

none

8.5 Make_binocular_wf

It arranges the single arm WF in the unique one.

Syntax:
result = make_binocular_wf(npupil,wf,channel)

Variables:

- Inputs:

npupil : size of the telescope aperture in pixels
wf : it is the WF
Channel : the arm (0: left, 1: right)

- Outputs:

none

- Keywords:

none

8.6 make_maschera

Function. It makes the mask for the LBT.

Syntax:
result = make_maschera(npupil,EPSILON=epsilon)

Variables:

- Inputs:

npupil : size of the telescope aperture in pixels

- Outputs:

none

- Keywords:

Epsilon : It defines the central obstruction ratio

8.7 Nirvana

Read and execute a script for the initialization of the Nirvana simulation
Syntax:
nirvana,scriptfile, sr_result, ALPHA=alpha, SPECIFICATION=specification, SAVE=save, NOCUT=nocut, RESTOREDM=restoredm, NOWIN=nowin, EXTRA_FIELD=extra_field, WFE=wfe, SILENT=silent, LOAD_IM=load_im
Variables:

- Inputs:

scriptfile : a string containing the path and name of the script

- Outputs:

sr_result : a structure with the SR results from the simulation

- Keywords:

ALPHA : Make a general check of the procedures that work with layer by printing on the layers some characters
SAVE : If set the initially phase screens are saved
NOCUT : If set un-used portions of the phase screens are not deleted to save memory
RESTOREDM : string, if set then mirror modes are read from the file indicated by the keyword
NOWIN : Must be set in UNIX machine! It define that no output screen will be presented during the run

EXTRA_FIELD : Allow stars outside Internal FoV (in the ring-FoV)
WFE : compute Wave front error (useless for NIRVANA)
SILENT : is set then not run verbosely
LOAD_IM : string, if set then interaction matrix are read from the file indicated by the keyword
SPECIFICATION : this keyword obliges the code to write every output file with the same suffix defined with the keyword itself

8.8 Nirvana_dim

It converts the LBT meter dimension in pixel units.

Syntax:

nirvana_dim,npupil,largesize,center_shift

Variables:

- Inputs:

npupil : size of the telescope aperture in pixels

- Outputs:

largesize,center_shift

- Keywords:

8.9 Nirvana_loopmcao

Simulation of a layer-oriented MCAO system for LINC-NIRVANA

Syntax:

Nirvana_loopmcao, TelescopeData, AtmoData, FieldData, DummyStar, SystemData, loopdata, elecdata, unit1, lognumber, srmap, srsave,zernike_save ,GRAPHICS=graphics, BORDER=border, ITERLONG=iterlong, W_ITER=w_iter, SR_LIMIT=sr_limit, FIELDSR=fieldsr, SAVE_ZERNIKE=save_zernike, NOCUT=nocut, SHAPEDM=shapedm, WAFFLE_CONTROL=waffle_control, WFE=wfe,SUBZERN ==sub_zern, SILENT=silent, MISMATCH_ST_enl=mismatch_st_enl, MISMATCH_PUPIL_dm=mismatch_pupil_dm, MISMATCH_MASK_WFS=mismatch_ mask_wfs, MISMATCH_MASK_DM=mismatch_mask_dm, LOAD_IM=load_IM, PISTON_ERROR=piston_error

Variables:

- Inputs:

Atmodata : structure of data related to the atmosphere
FieldData : structure of data related to the field of reference stars
DummyStar : structure of data related to the field of dummy stars
Systemdata : structure of data related to the MCAO system
Loopdata : structure of data related to the loop
elecdata : structure of data related to the WF-CCD specifics and photon flux
unit1 : logical number of the unit of the log file
lognumber : suffix to output file

- Outputs:

srmap : last strehl ratio results
srsave : history of the SR
zernike_save : history of the zernike coefficents

- Keywords:

GRAPHICS=graphics, BORDER=border, ITERLONG=iterlong, W_ITER=w_iter, SR_LIMIT=sr_limit, FIELDSR=fieldsr, SAVE_ZERNIKE=save_zernike, NOCUT=nocut, SHAPEDM=shapedm, WAFFLE_CONTROL=waffle_control, WFE=wfe,SUBZERN ==sub_zern, SILENT=silent, MISMATCH_ST_enl=mismatch_st_enl, MISMATCH_PUPIL_dm=mismatch_pupil_dm, MISMATCH_MASK_WFS=mismatch_ mask_wfs, MISMATCH_MASK_DM=mismatch_mask_dm, LOAD_IM=load_IM, PISTON_ERROR=piston_error

8.10 Nirvanasr

Calculation of the interferometric SR for NIRVANA MCAO-LO simulation.

Syntax:

nirvanasr, npupil,ndummy,psfsize,combined_wf,DiffLimOnAxis,srmap,psf,sr_inst

Variables:

- Inputs:

NPupil : pupil size
Ndummy : number of dummy stars
combinedWF : the LBT-WF
DiffLimOnAxis : diffraction limited value of the images of the stars on axis

- Outputs:

srmap : array of SR in the user-defined directions
psf : the psf
sr_inst : sr tip-tilt removed

- Keywords:

none

8.11 Piston_correction

Compute the piston values over the WF (single arm)

Syntax:

result = piston_correction(wf,ref,index)

Variables:

- Inputs:

WF : cube with the WF of each stars (It is not the WF of both side together, it refer to one side only)
ref : the star you want the piston value
index : the index where the WF is defined (it is the pupil of each side of the telescope)

- Outputs:

none

- Keywords:

none

8.12 Prepare_binocular_wf

It computes the arrays that will contains the wf

Syntax:

result = prepare_binocular_wf(maschera,largesize,psfsize,npupil)

Variables:

- Inputs:

maschera,largesize,psfsize,npupil

- Outputs:

none

- Keywords:

none

8.13 Prepare_psf_array

Prepare the array for the interferometric PSF computation

Syntax:

result = prepare_psf_array(npupil,psfsize)

Variables:

- Inputs:

npupil : dimension of the pupil in pixels
psfsize : dimension of the array used for FFT generation of the PSF

- Outputs:

none

- Keywords:

none

8.14 Retail_nirvana_layer

Cut the unused portion of the layers to save memory.

Syntax:

result = retail_nirvana_layer(nsizex,nsizey,h,r,npupil,layer,channel)

Variables:

- Inputs:

nsizex : X-SIZE USED
nsizey : Y-SIZE USED
h : layers altitude
r : is the dimesion to be cut
npupil : pupil dimension
layer : the original phase screens
channel : the arms of LBT

- Outputs:

none

- Keywords:

none

8.15 Save_ascii_piston

It saves the piston series in a ascii output file.

Syntax:

save_ascii_piston,file,data,nloop,dummyX,dummyY,deltat

Variables:

- Inputs:

file :string, the name of the output file
data :array, the Piston data
nloop : overall number of iteration
DummyX : X coordinate of the Piston-stars in arcsec

DummyY : Y coordinate of the Piston-stars in arcsec
deltat : the iteration time used

- Outputs:

none

- Keywords:

none

9 Authors and developers

Carmelo Arcidiacono, carmelo@arcetri.astro.it. University of Florence
Emiliano Diolaiti , diolaiti@pd.astro.it. Astrophysical Observatory of Bologna
Roberto Ragazzoni, ragazzoni@arcetri.astro.it, INAF- Astrophysical Observatory of Arcetri and Max-Planck Institut fur Astronomie (MPIA)
Massimiliano Tordi, tordi@pd.astro.it, University of Padova
Elise Vernet, elise@arcetri.astro.it, INAF- Astrophysical Observatory of Arcetri
Jacopo Farinato, farinato@arcetri.astro.it, INAF- Astrophysical Observatory of Arcetri

10 Acknowledgement

Thanks are due to Thomas Bertram for help in implemenatation and check of the LINC NIRVANA procedures.

11 References

1. Beckers, "Adaptive Optics for Astronomy: Principles, Performance, and Applications", *Annual Review Astronomy and Astrophysics*, **31**, 13-62, 1993
2. Diolaiti E., Farinato J., Ragazzoni R., Viard E., Baruffolo A., Arcidiacono C., Carbillet M., 2002, OWL-TRE-INA-60000-0055, LO WFS Final Design
3. Esposito & Riccardi , "Pyramid Wavefront Sensor behavior in partial correction adaptive optics systems, *A&A*, **369**, L9-L12, 2001
4. Farinato, Fedrigo, Marchetti, Ragazzoni, "More deformable mirrors (and higher Strehl) in Layer-Oriented for free", *ESO Conference and Workshop Proceedings*, Vol. **58**, 91, 2002
5. Ghigo, Crimi, Pérennès, "Construction of a pyramidal wavefront sensor for adaptive optics compensation", *ESO Conference and Workshop Proceedings*, **58**, 465, 2002
6. Noll, "Zernike Polynomials and Atmospheric Turbulence", *Journal of Optical Society of America*, **66**, 207-211, 1976
7. Ragazzoni, "Pupil plane wavefront sensing with an oscillating prism", *J. Mod. Opt.*, **43**, 289-293, 1996
8. Ragazzoni & Farinato, "Sensitivity of a pyramidic Wave Front sensor in closed loop Adaptive Optics", *A&A*, **350**, L23-L26, 1999
9. Ragazzoni, Farinato & Marchetti, "Adaptive Optics for 100-m-class Telescopes: New Challenges Require New Solutions ",Adaptive Optical Systems Technology, Peter L. Wizinowich, Ed., *Proc of SPIE* **4007**, 1076-1087, 2000
10. Ragazzoni, Diolaiti, Farinato, Fedrigo, Marchetti, Tordi, Kirkmann, "Multiple Field of View Layer Oriented Adaptive Optics", *A&A*, **396**, 731-744, 2002
11. Ragazzoni, Diolaiti & Vernet, "A pyramid wavefront sensor with no dynamic modulation", *Optics Communications*, **208**, 51-60, 2002

12. Rousset "Wavefront Sensors", *Adaptive in Astronomy*, F. Roddier Ed., Cambridge University Press, 91, 1999.
13. Straubmaier et al., "Near infrared Fringe and Flexure Tracker system, in *LINC--NIRVANA Preliminary Design Review document*, (Max Planck Institute for Astronomy, Heidelberg Germany, 2003) pp. 123--137.
14. Tordi, Ragazzoni & Diolaiti, "Simulation of a Layer-Oriented MCAO System", *ESO Conference and Workshop Proceedings*, **58**, 223, 2002
15. Winker, "Effect of a finite outer scale on the Zernike decomposition of atmospheric optical turbulence", JOSA, **8**, 1568-1573, 1991.