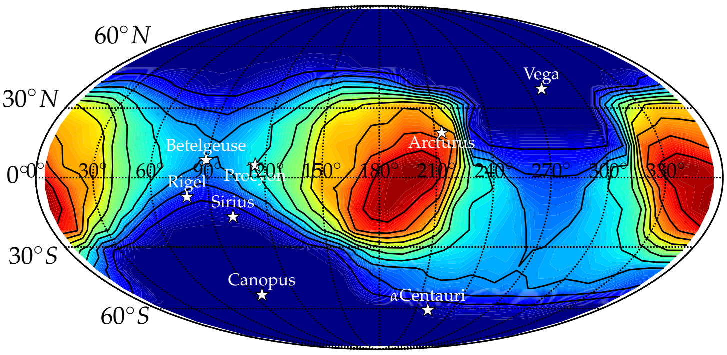
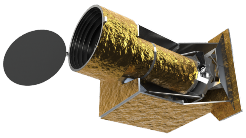
SALSA: StrAy Light SimulAtor v0.7

Documentation





Thibault Kuntzer[*] & Andrea Fortier[†]

Version 1 Revision 0 – June 20, 2014

[*] thibault.kuntzer@epfl.ch

[†] andrea.fortier@space.unibe.ch

Cover image: computer rendering of CHEOPS instrument pointing towards a cumulative observation map for faint stars.

Contents

1 Scope

Solar photons travel to Earth surface and may be reflected off the surface of the Earth back to space. Space-borne observatory that are bounded to an orbit around the Earth will be contaminated by this *Earth stray light* radiation. The amount of such radiation received by an optical detector onboard the observatory will vary with time according to several criteria that correlate notably with the chosen orbit, the period of the year, the design of the telescope itself and the pointing of the satellite.

The goal of this code is to get a first-order approximation of the Earth stray light contamination for space-borne instrument design in the first step of the mission. This tool was – and still is – developed to assess whether the design of an instrument could enable a photometric mission to observe faint stars ($M_V \sim 12$) for a long period of time and on a spatially large field. More information on this exoplanet CHEOPS mission is available in §**??**. This code and the analysis tools provided are therefore influenced by the task that are at hand. It is designed in a very modular way which enables the user to develop new modules that fit his needs. In its current version, the scope of the code extends from the simulation of the stray light contamination to its analysis in terms of "observable" sky. The definition of observable is given by criteria on the available time and the deepest measurable magnitude.

This software is not designed as stand-alone analysis tool of this particular aspect of the mission. It needs to be fed with the trajectory of the satellite throughout the epoch (hence enabling complex orbits) as well as the position of the Sun during this period. Those files can be obtained by dedicated software. The CHEOPS orbit was generated using the commercial `STK` software from company AGI. A satellite orbiting the Earth – especially Low-Earth Orbit (LEO) – can only observe some regions in the sky due to the presence of the Earth itself and is also limited by the presence of the Moon or other bright object in the field. The "allowed" region (or absolute or raw observation maps) were must also be computed beforehand. In the case of CHEOPS, this was done using an ad hoc `MATLAB` script. The transmission of a given photon entering the telescope off-axis is characterised by a axisymmetric function about the line of sight: the Point Source Transmittance (PST). This important function depends on the exact design of the telescope and requires particular attention. Several commercial software are available to compute the PST of a given design. A satellite may have period during which it is unable to observe – due for example of the presence of the South Atlantic Anomaly (SAA) – hence another table describing shutdown phases must be provided. The European Space Agency (ESA) built a tool – Spenvis – that can create such tables.

As it was already mentioned, this must be seen as a first approximation software for worst case scenario, because of a simplification: the albedo of the Earth. The exact amount of photon reflected off the surface of the Earth depend on the type of material making up – or covering – the surface as well as on the weather. Around the poles, the albedo is higher due to the ice and the snow (around 0.9) while oceans or continents have a much lower albedo (down to 0.1) (de Pater & Lissauer, 2001). In the code, the Earth is considered as a perfect sphere with an albedo of 1. This choice was made to (i) greatly simplify the code, (ii) the weather is impossible to include in this simulation and (3) the worst case estimate is wanted here. Other limitations are that only *Earth* stray light is included there is no consideration of the radiation due to the Moon or any other body. The cycle of the Sun is not included in the software, but this is far from being a first-order or even a second-order effect.

The output of this code are as follow:

- A file per minute per orbit with the table of observable region in the sky and the corresponding stray light value;
- Statistics files about the simulations;
- Error analysis of adaptive orbit step (see 3);
- Analysis of the mean and the maximum stray light flux;
- Movie of the evolution of stray light and corresponding deepest magnitude observable on short time scale;
- Movie of available zone in the sky versus a minimum stray light exclusion angle (see 3);
- Observation capabilities on the sky summed over the whole epoch according to observation rules;
- Prediction tables of observability of given target in the sky (beta version);
- Planet transit detection probabilities over the whole epoch.

The purpose of this document is to explain the usage of SALSA. §2 describes the installation process, §4 shows how to use the present code, which was used for a LEO exoplanet transit observation mission – CHEOPS. The results of this study are summarised in §??. §5 is an attempt at troubleshooting problems the user could encounter while §6 lists the known issues of the current version.

2 Installation

2.1 Download

SALSA can be downloaded from its GitHub repository: https://github.com/kuntzer/SALSA-public.

2.2 External Dependencies

The following software and packages are required for the code to run smoothly, so make sure they are installed before starting the simulations or the analysis. The executables for these external dependencies must all be located in your `PATH`.

- Python
  - Numpy (version ≥ 1.7) & SciPy (http://www.scipy.org/)
  - MatPlotLib (http://matplotlib.org/)
  - Basemap (http://matplotlib.org/basemap/)
- LaTeX (including `texlive-latex-base` and `texlive-fonts-recommended`)
- `gfortran`
- `epstopdf` for fancy plots.
- `pdfcrop` for fancy plots.

2.3 Required Data

As treated in §1, the code is not thought as stand-alone. It necessitate data on the satellite and on its orbit to work. The following lists the required files as well as their format and how they could be generated. In the following `<id>` is an integer indexing the trajectory of the satellite and `<k>` is a free integer variable describing an indexation.

PIPELINE

p1 - Trajectory of the satellite `orbit_<id>.dat`
Description: Table containing the position of the satellite with respect with the centre of the Earth.
Format: t x_{sat} y_{sat} z_{sat} (blank space separators)
Units: t [minutes] from the beginning of the epoch, x_{sat} y_{sat} z_{sat} [km].
Generation: `STK` (http://www.agi.com/)

p2 - Position of the Sun `sun_<id>.dat`
Description: Table containing the position of the Sun with respect with the centre of the satellite.
Format: t $x_{\odot}^{(\mathrm{sat})}$ $y_{\odot}^{(\mathrm{sat})}$ $z_{\odot}^{(\mathrm{sat})}$ (blank space separators)
Units: t [minutes] from the beginning of the epoch, $x_{\odot}^{(\mathrm{sat})}$ $y_{\odot}^{(\mathrm{sat})}$ $z_{\odot}^{(\mathrm{sat})}$ [km].
Generation: `STK` (http://www.agi.com/)

p3 - Raw Observable Maps `raw_maps_<id>/orbit_<k>.dat`
Description: Tables containing the allowed observation regions (cells of a grid) in the sky per one orbit and per minute. k runs on all orbits.
Format: t, α, δ
Units: t [minutes] from the beginning of the epoch, α (right ascension), δ (declination) [rad].
Generation: Ad hoc software that takes into account the constraints (SL exclusion angle, Moon exclusion angle, Solar system bodies, ...) on the observations.

p4 - PST `pst.dat`
Description: Table of the PST values.
Format: $\theta \quad PST$
Units: θ [degree], PST [-]
Generation: Ad hoc software such as `Zeemax` or `TracePro`.

ANALYSIS TOOLS

a1 - Index of the Orbits `minute_table_<id>.dat`

Description: Contains a conversion table between orbit k and start and stop time stamps. The keyword `rel` means time since the beginning of the current orbit while `abs` since the beginning of the epoch. Hence t_{rel} runs from 0 to one period.
Format: $k_{\text{orbit}}, t_{\text{start,rel}}, t_{\text{start,abs}}$ [new line] $k_{\text{orbit}}, t_{\text{stop,rel}}, t_{\text{stop,abs}}$
Units: k_{orbit} [-], t [min]
Generation: Ad hoc script that analyses `orbit_<id>.dat`.

a2 - Shutdown Events `SAA_table_<id>.dat`

Description: Describes when the observations are not possible (notably when over the South Atlantic Anomaly – SAA).
Format: $t_{\text{start}}, 1$[new line]$t_{\text{stop}}, 0$
Units: $t_{\text{start}}, t_{\text{stop}}$ [min]
Generation: ESA tool Spenvis http://www.spenvis.oma.be/

a3 - Position of the Moon `moon_<id>.dat`

Description: Table containing the position of the Moon with respect with the centre of the satellite.
Format: $t\ x_m^{(\text{sat})}\ y_m^{(\text{sat})}\ z_m^{(\text{sat})}$ (blank space separators)
Units: t [minutes] from the beginning of the epoch, $x_m^{(\text{sat})}\ y_m^{(\text{sat})}\ z_m^{(\text{sat})}$ [km].
Generation: `STK` (http://www.agi.com/)

2.4 Setup

The complete code can be seen as two separate codes. The first part is called *pipeline* in the following during which the stray light flux on the detector is computed. The second part is the analysis (*Analysis Tools*) of the simulated stray light flux. They can be run only sequentially which increase modularity at the cost of increasing the required time. Therefore, the pipeline can be set up separately from the analysis tools. They both require some common libraries as well as several strictly-organised software. The version downloaded should already contain a working folder tree.

2.4.1 Pipeline

Several instances of the pipeline can be run simultaneously. Therefore, for the pipeline only, a variable `part/p` is added. The situation can be summarised as follow (with `id` the orbit id, `SL` the SL exclusion angle, `p` the CPU number):

In the `resources/` folder, the following files are required:

- `minute_table_<id>.dat`
- `sun_<id>.dat`
- `moon_<id>.dat`
- `orbit_<id>.dat`
- `saa_table_<id>.dat`

To adjust the wavelengths of the instrument of the satellite, the Solar constant in this band must be modified in `constants` while the pixel size and the minimal SL exclusion angles must be given in `parameters`. Other characteristics of the detector must be entered in `parameters.py` in the root directory. Those include:

- `radius_psf` The size of the PSF in pixel
- `radius_tele` The radius of the telescope in metres
- `dlambda_over_lambda` the factor $\frac{\delta\lambda}{\lambda}$ (See §3.5 for more information).

One important parameter must be set up in the `straylight_<id>_<p>/CODE/parameter` file :

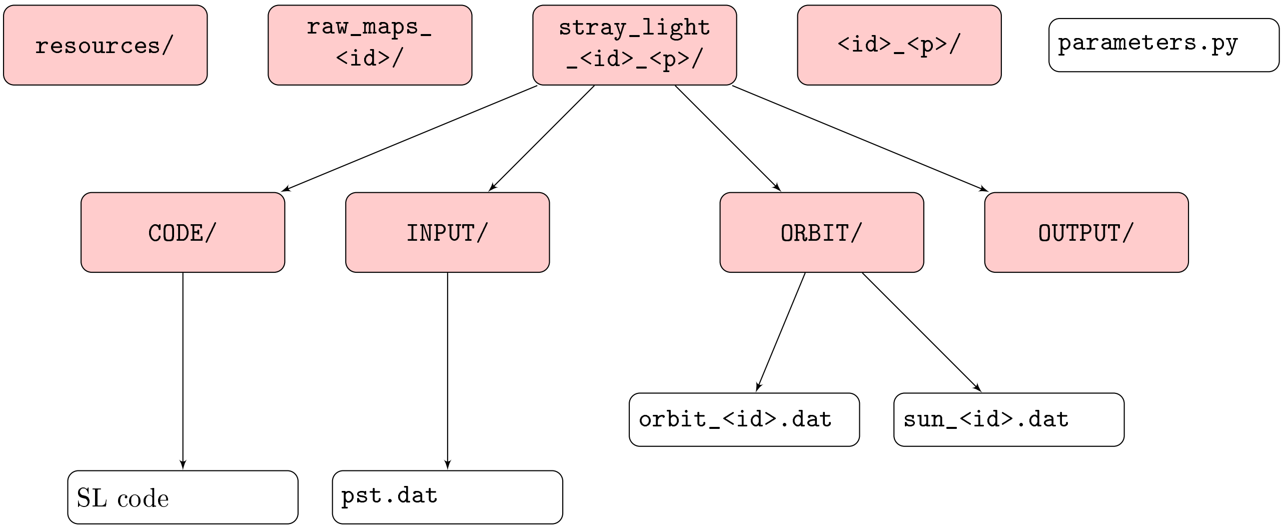


Figure 1: *Flow chart of the different folders in the pipeline folder. The folder raw_maps_<id>/ contains in addition the files p3 deg/orbit_<k>.dat. The list of for the folder resources is available in the following. The folders <id>_<p>/ will contain the results of the simulation.*

- `pixel` which is the pixel size in cm^2.

Further parameters control debugging functions and should be self-explanatory. A grid representing the Earth of at least 1001 × 501 is required. This is needed for the simulations to converge to a good estimation of the flux.

2.4.2 Analysis Tools

There, at least the following three folders must exists:

- `<id>_figures/` contains all the figures generated
- `<id>_flux/` contains all the flux data generated by the pipeline
- `<id>_misc/` contains all data generated by the analysis

In addition, the file `parameters.py` must contain the following information:

- `last_orbits` a Python dictionary `<id>:<last orbit>`
- `ppm_threshold` the minimum SNR acceptable in ppm.
- `resx,resy` the resolution of the rectangular sky grid.
- `mirror_efficiency` the efficiency of the optical system. I.e. the ratio of photons reaching the detector by the number of photons entering the optical system.
- `SL_post_treat_reduction` efficiency of the stray light removal algorithm in the data reduction pipeline.

The other options are optional and self-explanatory. Further setup of the analysis tools are described in each module. In `resources/constants.py`, there are definitions of the wavelengths used:

- `wavelenght_visual`
- `Fv`
- `Jy`

For more information about the variables, refer to 3.5.

3 Numerical Methods

In this section, the algorithms and the softwares used and prepared are presented. The presentation starts by discussing the calculation of the visible area that was coded by Luzius Kronig from EPFL/Swiss Space Center, continues by describing the stray light code of Andrea Fortier at Unibe and Thibault Kuntzer and ends by the description of the various algorithms and techniques developed to derive stray light maps and tables. A flow chart summarises the numerical codes used in Fig. 2.

3.1 Observable Area

An visible region in the sky is an area that the satellite can observe at a given time. The geometry of this zone is primarily defined by the position of the satellite relative to objects that could contaminate the image either by direct imaging or by diffuse light such stray or zodical lights. Most of the constraints are therefore derived from the pointing direction – the line of sight – of the telescope. This paragraph describes quickly the steps leading to the computation of observable sky.

Constraints The first constraints on this area are that it is forbidden to look directly at or close to bright objects namely the Sun, the Earth and the Moon. Moreover, to reduce the contamination by diffuse light of the Sun or Earth and Moon stray lights, there are conditions on the minimum angles between the line of sight (LOS) of the telescope and the limb of those objects.

Earth Occultation The telescope shall not point to a target, which projected altitude from the surface of Earth is less than 100 km (thickness of the Earth's atmosphere);

Earth stray light exclusion angle In order to limit stray light contamination, the minimum angle allowed between the line of sight and any (visible) illuminated part of the Earth limb, the so-called Earth stray light exclusion angle shall be $\alpha_\oplus$;

Sun exclusion half-angle The Sun must be outside the cone around the line of sight (LOS) of the telescope having a half-angle, the so-called sun exclusion angle, of $\alpha_\odot$;

Moon exclusion half-angle The bright Moon must not be inside a cone around the line-of-sight of the telescope having a half-angle, the so-called moon exclusion angle, of α_m.

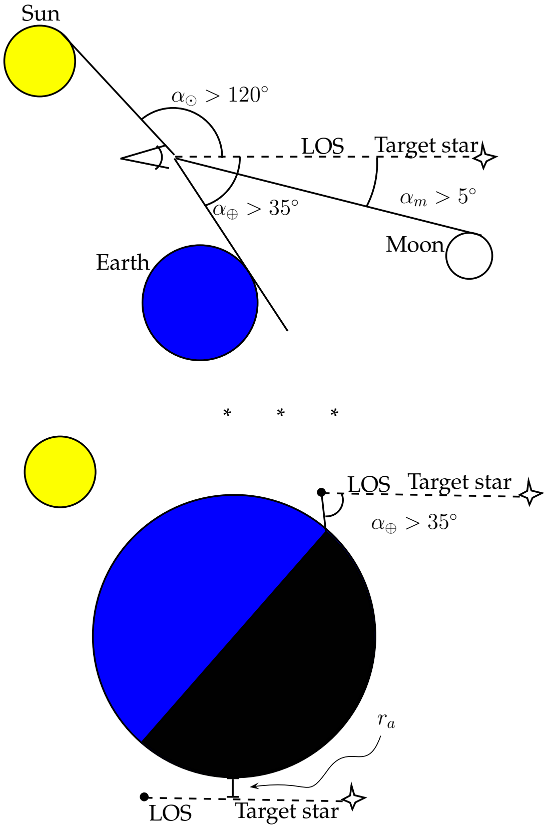


Figure 3: Top. *Exclusion angles from the line of sight (LOS) of the telescope.* Bottom. *Close-up for different cases for observations above the dark side of the Earth. All values refer to the CHEOPS mission.*

The constraints shape the observable regions; figure 3 depicts those exclusion angles for CHEOPS.

The South Atlantic Anomaly (SAA) plays also an important role as crossing it requires instrument shutdown.

The maps of the visible regions in the sky are generated without taking the SAA into account and the restriction is applied afterwards during the analysis. Otherwise, if the SAA would already be included in the observability maps, it would be impossible to perform an adaptive orbit step as described in the next section. Indeed, the crossing of the SAA does not always occur at the same time in the orbit. Moreover, the time spent over the SAA changes as well.

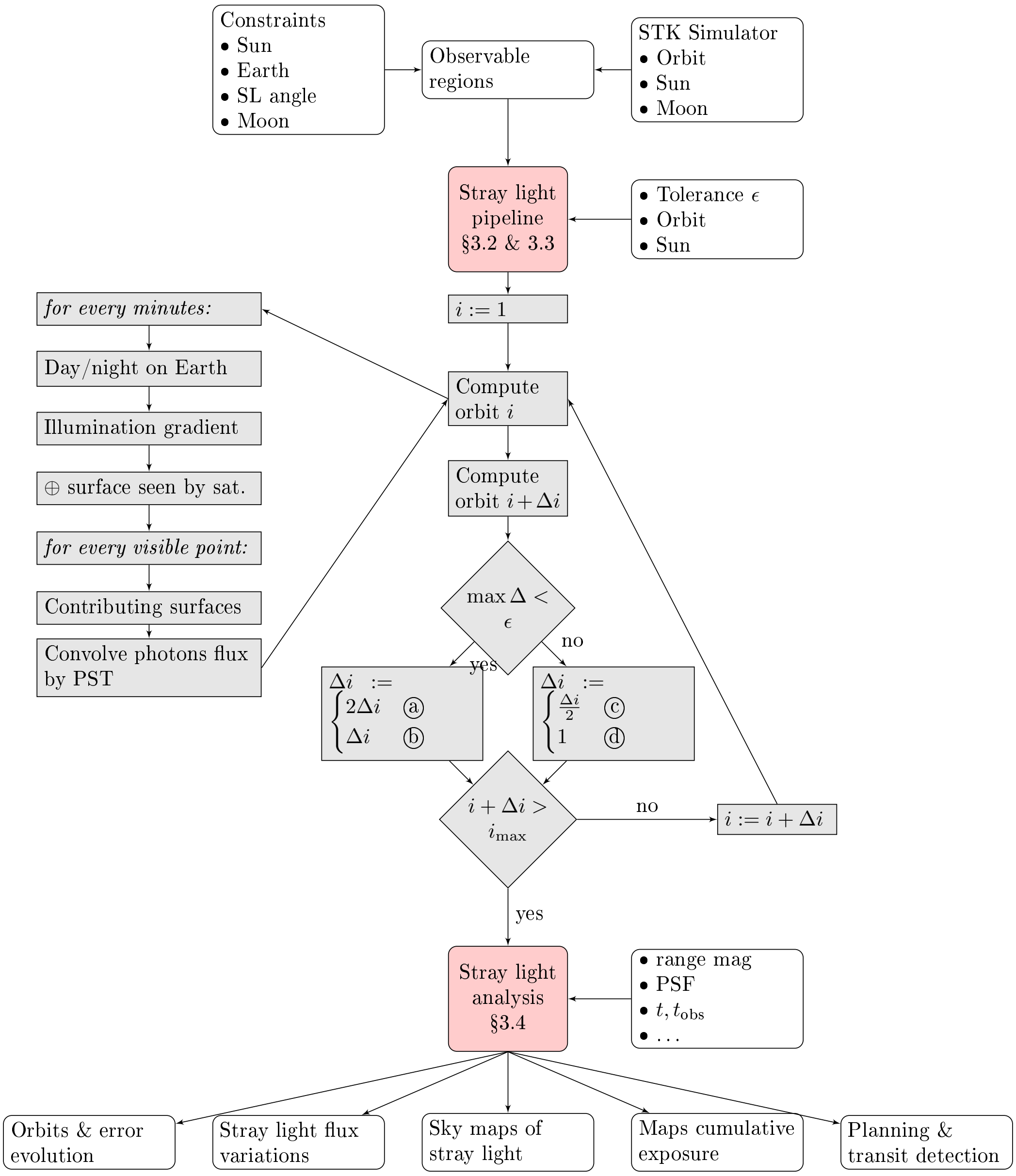


Figure 2: *Flow chart of the project. Red blocks represent the different code suites used in this project. Gray cells depict algorithmic instructions and white either inputs or outputs of the project. STK is the commercial software used to generate the orbit of the satellite. i is the orbit number, Δi represent the orbit step size and i_{max}. $\max \Delta$ is the maximal (relative) difference in the equivalent magnitude. The conditions on the orbit step size are ⓐ if Δi is smaller than a maximum orbit size; ⓑ else; ⓒ if $\frac{\Delta i}{2} \geq 1$; ⓓ else.*

Coordinate System. The observable region in the sky must be in reference to the satellite. To describe the system (Sun, Earth, Moon, satellite) a common coordinate system must be chosen. The chosen frame is the International Celestial Reference Frame (ICRF). In order to perform the computations, the centre of this coordinate system is translated into the satellite. Therefore, all positions are translated to the satellite and not the Earth.

Discretisation in Space. The sky is simulated by a grid of points described by the following relationships:

$$\begin{aligned} \alpha_g(i) &= \frac{\Delta\alpha}{2} + i\Delta\alpha,\ i = \{0, \dots, n-1\} \\ \delta_g(j) &= -\frac{\pi}{2} + \frac{\Delta\delta}{2} + j\Delta\delta,\ j = \{0, \dots, m-1\} \end{aligned} \tag{1}$$

where n, m describe the resolution of the gird. This grid yield cells on the sky of $\Delta\alpha \times \Delta\delta$ or $9^\circ \times 9^\circ$. Fairly large cell are to be used. It arises from the computation time of the different steps which scales at least as $\mathcal{O}(n \cdot m)$.

Discretisation in Time. The orbital period of the satellite depends of course on the altitude following the famous third Kepler law:

$$P = 2\pi\sqrt{\frac{a^3}{Gm_\oplus}}$$

The observability maps will be used to plan the observation and therefore, it is better to have a time in minute rather than the position in a given orbit. The time $t_0 = 0$ min is the beginning of the epoch.

Outputs The output files must be a list of the right ascension α and the declination δ of the grid points which are observable. In order to reduce the number of files generated, the visible points are all grouped in one single file for one orbit. Hence, the final output files are given in the following format : `t, ra, dec` in, respectively, minutes and radians.

3.2 Stray Light Code

To compute the Earth stray light that enters the telescope and hit the detector, a dedicated code in `Fortran` was designed by Andrea Fortier and Thibault Kuntzer. This program named `stray_light.f` can compute independently of the observability map code, the stray light contamination at any time and in any direction.

The Steps to Stray Light Evaluation. The stray light computation is divided in several steps:

1. Compute the illuminated part of the Earth;
2. Compute the gradient of the illumination;
3. Compute which parts of the illuminated Earth are visible by the telescope;
4. For every visible region or "targets", do the following:
   (a) Compute the flux of light reflecting off Earth's surface that reaches the telescope;
   (b) Convolve this signal by the PST to get the flux of Earth stray light at the detector.

Those steps are explained in further details below. It is necessary to caution the reader that the description of the process is very linear here and does not exactly mirror the implementation in which numerical optimisation was planned.

In the software, a few simplifications are made: Firstly, only the *Earth* stray light is computed. Contamination from the Moon is not considered as it is assumed that its exclusion angle is sufficient. Secondly, the Earth is considered to be a perfect sphere thus neglecting its bulge and hence slightly miscalculating the off-axis angle of an incoming photon. The surface of this sphere is supposed to be uniform: no distinction is made between oceans or land, the weather is not simulated (therefore thunderstorms and aurorae are not considered) and cities not modelled. To get a upper estimate of the stray light, the albedo of the Earth is set to 1. The albedo is a multiplicative factor in the calculation of the flux (See §3.5 for more details). The value of the averaged albedo ($\sim 0.3 - 0.5$) on the Earth could therefore be used. However, it was chosen to set the albedo to 1 to compute a worst case scenario, but with little difference to a finely tuned value because of the logarithmic behaviour of the final output of the project (as discussed in §6).

Illuminated Earth. The first step is to describe with enough precision the illuminated part of the Earth. The only interesting part of the Earth at this point is the one in daylight. Indeed, it is considered that half of the Earth which is the night does not contributed to stray light.There are thunderstorms, aura or even city lights that could contaminate the signal if the LOS were too close to the limb of the Earth. Those effects which are extremely complex to simulate are not integrated in the code, but the constraint to point above the atmosphere with a given minimal altitude used to ensure that there is no contamination.

The surface of the Earth is represented by a grid for which every cell is determined to be either in the day or in the night depending on the dot product of its normal and the direction of the Sun. The minimum grid that can achieve sufficient accuracy, *i.e.* that the flux of stray light has converged, is quite large with 1000 by 500 cells.

Illumination Gradient. For the cells that are in the sunlight, the intensity I of the light is computed by determining the cosine of the angle at which the light impacts the cell c. This depends upon the relative position of the cell and the Sun and can be computed using the following formula:

$$I_c \propto \cos\varphi \tag{2}$$

where φ is the angle between the direction of the Sun and the normal vector of the cell (See §3.5). The amount of radiation that arrives at a particular cell is also given by the energy received by the Sun at the distance of the orbit of the Earth. Here, the common value of about 1360 W/m^2 is not considered as it takes the whole spectrum into account. The solar constant must be reduced to the applicable band window. For example, CHEOPS instrument is sensitive to the range of wavelengths from 400 to 1100 nm and the solar "constant" for this interval can be computed to be 880 W/m^2.

Changing Viewpoints. The two previous parts can be carried out without the knowledge of the orbit at that particular time. Indeed, the satellite plays no role. This step determines which illuminated cells can be seen by the satellite at any given time. Assuming a position for the satellite, the code finds all cells within a cone defined by all visible tangent directions on the surface of the Earth and flags them as relevant to the stray light computation. See figure 4 for an example.

Flux of Photons at the Telescope. The flux of photons at the entrance of the telescope (not at the detector or somewhere else in the optical path) depends upon the direction of the line of sight. Let θ be the angle between the LOS and the photon path. The target star is denoted by $\boldsymbol{r}_\star$, the position of the satellite by $\boldsymbol{r}_s$ the cell position by $\boldsymbol{r}_c$ all with reference to the centre of the Earth. The direction to which the photon bounced off the cell to the satellite can be defined by the vector $\boldsymbol{v} = \boldsymbol{r}_s - \boldsymbol{r}_c$. If the angle θ is larger than $\pi/2$, then no photons will hit the telescope:

$$\boldsymbol{r}_\star \cdot \boldsymbol{r} = |\boldsymbol{r}_\star| \cdot |\boldsymbol{r}| \cos\theta \tag{3}$$

$$\cos\theta = \begin{cases} > 0 & \text{No hit} \\ \leq 0 & \text{photon hit} \end{cases} \tag{4}$$

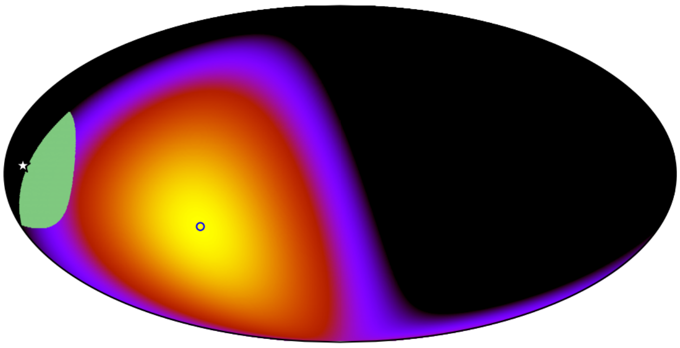


Figure 4: *Mollweide projection of the illuminated Earth. The circle on top of the maximal illuminated region is the projection of the position of the Sun, the white star-shaped marker symbolises the satellite and the green region is the illuminated Earth seen by the satellite.*

To obtain the value of the total flux of photons all that is left is to integrate on all possible directions. The surface of the Earth is assumed to be Lambertian[1]. This is justified as the albedo is assumed to be constant and because the radiant intensity (power per unit solid angle) depends upon the cosine of the angle described by the direction of the radiation leaving the point. The same cosine factor is applied to the surface that emits the radiation and therefore they cancel out.

Accounting for the Photon Rejection. Up until this point, the considerations were mostly geometrical. The Earth is assumed to be a perfect sphere which reflects light with the same radiance in any direction. The cells that can be seen by the satellite are defined by its orbit. Hence computing the angles from the LOS to them is easy. So, at this point in the code, the amount of photons reaching the pupil of the telescope is computed each with an off-axis angle θ. The transmission of the incoming photons through the telescope can be approximated by a axisymmetric function that depends upon the off-axis angle θ of the radiation : the point source transmission (PST) – See for example Breault (2010) or Pompea (1995). The energy that arrives at the detector is multiplied by the value of the PST as function of the off-axis angle of the photon. An interpolation of the PST is performed to fill the gaps in the knowledge of this function which is sampled using large steps. The flux is also converted to photons per second per pixel. As the stray light is scattered by different parts of the telescope and not through the optical path of the mirrors and focal planes, its signal is not convolved by the PSF and hits a single pixel.

Practicalities. The code was optimised to perform stray light computations for known observability maps and ephemerides for the Sun. It takes as input parameters a PST (`theta [deg],PST`), the position of the satellite (in the frame of reference of the Earth) and the Sun in separate file containing `x,y,z,ra,dec,r` with distances in km and angles in radians. Finally two files describe the target list: one containing the number of targets to optimise the `Fortran` memory management and file handling and a list of targets with two columns: the right ascension and the declination both in radians.

3.3 A Pipeline to Compute the Stray Light Flux

To generate a stray light map of the observable sky at a given time, a `Python` script looks for the corresponding observability map, creates the list of visible points which are then given as inputs to the stray light calculator along with a few other variables (position of the Sun and the Earth, . . .) as described in section §3.2. To speed up computation, an adaptive time step is implemented. An orbit is computed every minute, however as there are around 15 orbits per day in LEO, consecutive orbits are very similar in most of the cases. Hence the capability of skipping orbits that are too similar. This is

[1] A Lambertian surface is a surface whose radiance (power per unit solid angle per unit projected source area) is the same independently of the angle of view. Its apparent brightness is the same.

done by computing orbit i and then orbit $i + 10$. If the comparison of the two orbits $(i, i + \Delta i)$ reveals a difference of more than 5% in stray light flux, then the orbit $i + \Delta i/2$ is computed. Those parameters can be changed. The algorithm always tries to increase the number of step and doubles the step size if it is (1) smaller than the optimum and (2) the error to the previous orbit is less than the threshold. The maximum bearable error to the next step can be chosen such that it would ensure that the *maximal* error in the orbit in term of equivalent magnitude (see §3.5).

This step of generating stray light maps for the whole epoch is the most lengthy one. The time needed to compute a whole year with the adaptive orbit step described in the above depends on the altitude as the stray light is less prominent at high altitude. The outputs are one file per minute containing the following characteristics: time, grid point position in radians (right ascension, declination) and the flux of stray light in ph px^{-1} s^{-1}. The relative high number of output files ($\sim 5 - 10 \cdot 10^5$) weight around 1–2 GB and will be analysed in the next steps for a whole year. The temporal resolution is chosen to be the same as the observability maps.

3.4 Analysis of Stray Light Data

To interpret the data, several analysis tools were developed. The scripts range from the very simple counting of computed orbits to a much more elaborated probability of observing a transit estimation. A description of some of those tools is provided here.

Computed Orbits. This is the first logical step – compulsory in order to proceed to other functions –: check which orbits have been computed. This seems redundant as this list can easily be retrieved from logging at the previous step when computing the stray light maps. In order to speed up the computations, most of the stray light data can be generated with several CPUs: not in parallel, but working on different periods in the epoch. To avoid any mistake when merging back the results of the different CPUs, this scripts tests if minutes 0, 20 and 60 were computed. As the SAA (shutdown periods) is not considered, each of those minutes had to be computed. If any fails, then it is considered that this orbit was not computed.

Analysis of the Error Evolution. The second step is to check that the error from one orbit to the next is not larger than the limit or if it is, then that every orbit is computed. This operation is rather slow as the code must compare the minutes pairwise in each orbit and due to non-integer period, several combinations must be made to find the best fit.

Analysis of the Stray Light Flux. From this point forward, the data is considered as robust if the two previous steps did not pointed out to erroneous behaviours. The stray light flux depends upon the amount of Earth illuminated cells seen and their illumination. In order to characterise those changes the stray light flux is analysed by four different quantities:

1. The *maximal value*: the maximal value of the flux for a given orbit;
2. The *maximal mean*: taking the maximum value of the flux for one minute, repeating for every minute in the orbit and averaging;
3. The *twice-averaged mean*: taking the average of the flux at a particular moment in the orbit, repeating for every minute in the orbit and taking the average on those averages per minute;
4. The *evolution in one orbit* of the stray light in one direction.

Stray Light Flux Maps. The list of the visible targets is completed by the stray light flux. Therefore, a projection of the targets can be made on a map with a colour assigned to the stray light. A conversion to limiting magnitude can be made to represent how faint the observed stars can be given a certain SNR (see §3.5). Those maps have the advantage – beside being aesthetically pleasant – of showing the varying visible regions in the sky and the effect of the relative alignment of the Earth and the Sun.

Cumulative Observation Maps. The cumulative observation time is a value that is computed on a long period of time. There is a minimum of observation time to consider the observation worth it. On top of this minimal observation time, the telescope takes time to align its LOS to the star: the acquisition time. Those maps give informations about which regions of the sky are invisible which can be observed the longest and the effects of the constraints and the SAA. Those maps are relatively long to establish as the appearance and disappearance times – which occurs many times over the course of the year – of every grid point must be computed.

The time of observation can be understood in different ways. We provide proposed definitions:

Cumulative observation time describes how much time can the satellite observe in one orbit whose start and ending are arbitrary defined. Interruptions in the cumulative observation time are acceptable. In this document, it is often short-handed to "observation time".

Consecutive observation time describes how much time can the satellite observe without interruptions in the observations. If needed arbitrary start and ending time of the orbits can be defined.

Accumulated observation time is the total amount of either cumulative or consecutive time.

Period of observation represent the number of days a target is observable. It does not describe the observable time, rather how many days the target can be followed.

Observation maps are produced for the two definitions (cumulative and consecutive observation time). Subtraction of two maps to see the difference between two observational strategies are

Probability of transit detection SALSA was developed for a mission which goal is to characterise planets by observing their transits. This tool aims at estimating the probabilities of transit detection given the absolute observation regions and the stray light contamination. This tool is limited to looking at the transit duration versus the effective observation possibility. It does not evaluate the possibility of detecting the transit on the basis of its depth. SALSA looks for specific transit patterns. They represent certain families of objects transiting their star. Those families have typical period of rotation and transit time (The CHEOPS Study Team, 2013). Their ingress and exgress time is supposed to be unknown. SALSA assumes that the ingress time is described by an uniform probability distribution. The next step is to find how many of those transit times per rotation around the star are visible. Depending upon the observational conditions, certain objects can be observed for more than the time between two transits and thus will have probabilities of more than one. If an object can be observed non-consecutively and the total observation time is still larger than the time between two orbits, the probability is still considered more than one.

Almanac of catalogs Using a similar procedure and providing a catalog of objects, their rise and setting time in the effective observational region are computed. Such kind of data can be used for the planning of the observations. This module can then easily be turned into an observation scheduler.

3.5 Computational Details

From Stray Light Flux to Maximum Magnitude Visible. The question asked here is: given the stray light flux, what is the faintest target visible? The function `flux2mag` (and the reciprocal `mag2flux`) yields this limiting magnitude (or inversely the flux) of the target star. The noise represented by the stray light to the flux of the star must be at maximum T. The flux is related to the magnitude m by:

$$F = 10^{-2/5\cdot m} \tag{5}$$

which has units of ergs s^{-1} cm^{-2}. The flux of stray light is given in units of ph s^{-1} px^{-1}. Moreover, only photons with a certain energy matter and not all the spectrum of light. Therefore the flux is now given by:

$$F_{\text{sl}}(m_V) = \underbrace{F_V J_V \frac{\Delta\lambda}{\lambda}}_{\equiv A} \cdot \underbrace{\left(\frac{R_{\text{tel}}}{R_{\text{PSF}}}\right)^2}_{\equiv B} \cdot 10^{-2/5\cdot m_V} \tag{6}$$

where $F_V = 3640$ Jy cm^2/ergs converts from ergs to Jansky for the band V, J_V transforms from energy to the flux of photons for the band V: $J_V = 1.51 \cdot 10^7$ photons s^{-1} m^{-2} / Jy. $\frac{\Delta\lambda}{\lambda}$ is the bandwidth of the instrument. Those three parameters grouped in A depend upon the choice of the band. R_{tel} is the radius of the aperture in metres. R_{PSF} px is the radius of the PSF in pixels. The PSF is modelled here as an axisymmetric step function (top hat function) such as

$$\text{PSF}(r) = \begin{cases} 1 & r \leq R_{\text{PSF}} \\ 0 & r > R_{\text{PSF}} \end{cases} \tag{7}$$

Those two radius are geometrical parameters and their ratio squared transforms from m^2 to pixels and is noted for convenience B.

The SNR for the stray light is given by $SNR = 1/T$ and therefore, the maximum stray light flux tolerable for a given magnitude is:

$$F_{\text{sl}}(m_V) = T \cdot AB \cdot 10^{-2/5 \cdot m_V} = \frac{AB \cdot 10^{-2/5 \cdot m_V}}{SNR} \tag{8}$$

This computation is implemented in `mag2flux`. The reciprocal function uses equation 8 to express $m_V(F_{\text{sl}})$. This implies that there is a logarithmic of base 10 function. In order to avoid numerical errors while performing this computation, the stray light flux is clipped to a minimal value of 10^{-40} ph s^{-1} px^{-1}.

Comparing Two Orbits. This function is important as it decides whether the orbit step size in the calculations of the stray light flux must be reduced. It must, therefore, read the data from the two orbits and compare them. This function is quite conservative in the sense, that it returns the *maximal* difference in the orbit and not the *mean*. The two orbits (the `reference` and the `current` as defined in the code) are compared pairwise: minute by minute. As evoked before, this is a none trivial problem as the orbital period is not an integer of minutes. Therefore, three combinations are tested using a shift $s = \{0, 1, 2\}$ in time between the reference and the current orbits.

Great Circle Distance. The angular separation of two points is required several times in the different steps of this project. This separation must be computed reliably and robustly otherwise several numerical issues arise. Those numerical issues are due to two things: (1) singularities of the inverse trigonometric functions at multiples of $\pi/2$ and (2) the inverse cosine which handles badly two close-by points (Vincenty, 1975). Therefore, to compute the angular distance between two points 1 and 2, the following formula was used:

$$\begin{aligned} \varphi &= \arctan\left(\frac{\sqrt{a^2+b^2}}{\sin\delta_1 \sin\delta_2 + \cos\delta_1 \cos\delta_2 \cos\Delta\alpha}\right) \\ \text{where } a &= \cos\delta_2 \sin\Delta\alpha, \\ b &= \cos\delta_1 \sin\delta_2 - \sin\delta_1 \cos\delta_2 \cos\Delta\alpha \end{aligned} \tag{9}$$

Similarly, the angular separation cannot be simply computed by the scalar product, but by the equivalent, but always correct cross product:

$$\varphi = \arctan\left(\frac{|\boldsymbol{r}_1 \times \boldsymbol{r}_2|}{\boldsymbol{r}_1 \cdot \boldsymbol{r}_2}\right) \tag{10}$$

where $\boldsymbol{r}_i$ are vectors that point to the position of the two coordinates.

4 Usage

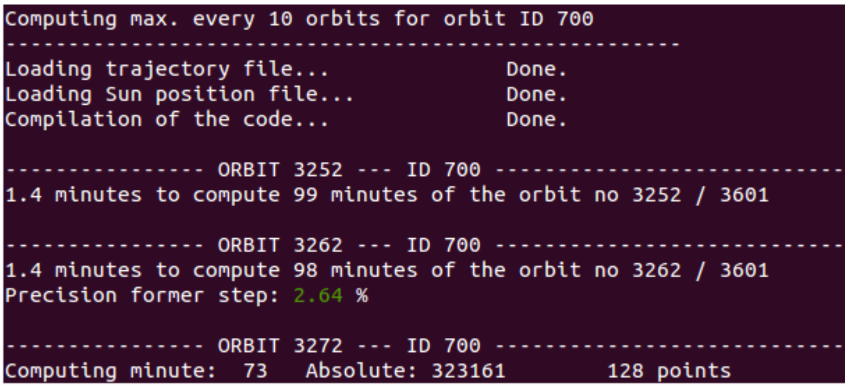
4.1 Running the Pipeline

If the setup was done correctly (§2.4), then running the pipeline should be relatively easy. During this part of the simulations, the code can be run in parallel. However, there is no built-in parallelism management system therefore, the user should duplicate all the files and folder that are indexed by `<p>` in Fig. 1.

The file `1-compute_MASTER.py` must be copied, renamed and parametrised to reflect the number of CPU in use and the number of orbit that the code should simulate. To run the code simply type:

```
python 1-compute_<id>.py
```

After a few moments, the screen should look like Fig. 5. The amount of time necessary to compute an orbit depends on the complexity of the calculation and to a lesser extent the number of cells that are observable. An orbit very close to the surface of the Earth will take a large amount of time to simulate. The likelihood of two orbits separated by a step size being similar to some degree is less than high orbits as the flux of stray light radiation is much larger. With the current version of the code, for a circular orbit at 700–800 km and an CHEOPS-like instrument, the code needs of the order of 20 hours to complete the simulation for one year. Thanks to the adaptive orbital step approach, the number of orbits computed is reduced. However, for the same example and using a maximum step size of 10, the number of resulting file is still of the order of $\sim 50 \cdot 10^3$. While this is still very manageable, lower orbits or worse PST may generate up to $\sim 500 \cdot 10^3$ files which does slow the analysis.



```
Computing max. every 10 orbits for orbit ID 700
---------------------------------------------------------
Loading trajectory file...              Done.
Loading Sun position file...            Done.
Compilation of the code...              Done.

---------------- ORBIT 3252 --- ID 700 ---------------------------
1.4 minutes to compute 99 minutes of the orbit no 3252 / 3601

---------------- ORBIT 3262 --- ID 700 ---------------------------
1.4 minutes to compute 98 minutes of the orbit no 3262 / 3601
Precision former step: 2.64 %

---------------- ORBIT 3272 --- ID 700 ---------------------------
Computing minute:  73   Absolute: 323161        128 points
```

Figure 5: *Typical screen output of the pipeline*

4.2 Running the Analysis Tools

Once the simulations of the SL contamination are finished, the data must be copied in the folder `<id>_flux/`. The only mandatory module to run is the first one (`2-orbits_computed.py`) as it builds the index of the *effectively* computed orbits. The order proposed in the original pack is increasing in the complexity of the analysis. It first focuses on the parameters of the simulation and testing the robustness of the data. The few last modules analyses the data from the point of view of observation capabilities and prepares the planning of the observations.

All the provided modules contain a documentation at the beginning of the file explaining the purpose of the file, the parameters, requirements, ... Some modules require additional folder to store the output figures, maps or data. The amount of time required for a given module can be quite long and depends on the length of the epoch and usually on the number of orbits computed during the pipeline phase.

5 Troubleshooting

5.1 Pipeline

The error is always greater than the threshold There is a parameter that could influence the error made from one orbit to the next: the parameter `magnitude_max` in `parameters.py` which is used to clip low values of the SL flux. If this number is set too high, then it becomes highly sensitive.

The pipeline refuses to compile the code There is a problem of permission for the file `stray_light_<id>/CODE/compile`. Make sure that the CHMOD is configured to 777 for this file.

The compilation of the Fortran code necessitate the `gfortran` compiler. The automatic compilation can be turned off in `1-compute-MASTER.py`.

5.2 Analysis Tools

Python cannot write data file This should start when you run this error analysis already. The distribution of Numpy may still be lower than 1.7.0. If this is the case, we recommend you install the latest version (downloaded from their website) manually. If this does not fix the issue, make sure that the output directory exists.

Replacing flux files For some reason, if some part of the simulations were not done correctly, it may be necessary to replace some of the data. Manipulating the flux files by hand can reveal very tedious. A script was designed to replace a portion of the data. Running

```
python b–replace_flux.py
```

will move the current data used in the analysis to a trash folder which must *exist* prior to running the code. The new data are copied from the the pipeline folder. See the file itself for the parameters.

6 Known Issues

In this section, the known weakness of the code are presented. User who notice other issues are welcomed to send their comments to the authors.

- A constant albedo is used not modelling correctly the surface of the Earth, the weather or the thunders. In the stray light code, the albedo A is taken to be one to yield a worst case scenario. This factor is a multiplicative factor in the computation of the stray light as:

$$F_{\mathrm{sl}} = S_V \int_{\Omega(\mathrm{i,s},\oplus)} A \cos\vartheta \cos\varphi \mathrm{d}\Omega \tag{11}$$

  where Ω(i,s,$\oplus$) is the illuminated Earth visible from the satellite, S_V the Solar flux in the relevant band, ϑ the angle LOS–illuminated surface element, φ angle direction of Sun–normal of surface element. It can be computed that changing the albedo to an averaged value of 0.35 (de Pater & Lissauer, 2001) would push the limiting magnitude by about 1.1 magnitude fainter. Around the poles however, the albedo is higher due to the ice and the snow (around 0.9) while oceans or continents have a much lower albedo (down to 0.1).

- PSF, Aperture and Uncertainty. Changing the aperture of the telescope by 1 cm with a constant PSF increases the flux by $\sim 13\%$ or a 0.13 mag brighter in the CHEOPS case. To get a more precise value therefore of the equivalent magnitude, a convolution of the signal is to be made. Given a change in what would be the equivalent PSF radius for a top-hat function of $\pm 20\%$ would lead to a change of ${}^{-0.20}_{+0.25}$ respectively in the magnitude.

- The function `compare_two_orbits()` may cause some problems. The cause is that the time 0 in an orbit may not always be located at the same point in the orbit as the period of the orbit is not an integer. The code will try to find the best possible fit by adjusting a shift between the two orbits. It is also sensitive to the parameter `ppm_threshold` set in `parameters.py`.

- When drawing images of the stray light on the sky (`9-plot_flux.py`), the function `draw_boundaries()` does not work properly if the observable zone is made up of more than one region.

- The script that calculate the almanac of a single target does not work properly. In general, ephemerides calculation from a target catalogue should be considered as beta implementation.

- There might be some unexpected differences (up to 1% in the total sky coverage) in observability maps as shutdowns of the satellite are not taken into account. This can be explained by the way the observable cells in the sky maps are computed.

Bibliography

Breault. 2010, Control of Stray Light *in* Handbook of Optics, Third Edition Volume II: Design, Fabrication and Testing, Sources and Detectors, Radiometry and Photometry, 3rd edn. (New York, NY, USA: McGraw-Hill, Inc.)

de Pater, I., & Lissauer, J. J. 2001, Planetary Sciences

Pompea, S. M. 1995, Space Science Reviews, 74, 181

The CHEOPS Study Team. 2013, CHEOPS Definition Study Report (Red Book) (European Space Agency), eSA/SRE(2013)7

Vincenty, T. 1975, Survey Review, XXIII (misprinted as XXII) (176), 88

Appendix A List of Abbreviations

CHEOPS	CHaracterizing Exoplanets Satellite
EPFL	Ecole Polytechnique Fédérale de Lausanne
ESA	European Space Agency
INAF	Istituto Nazionale di Astrofisica / Italian national institute for astrophysics
LEO	Low Earth Orbit
LOS	Line Of Sight
PST	Point Source Transmittance function
SAA	South Atlantic Anomaly
SL	Stray Light
SNR	Signal-to-Noise Ratio
Unibe	University of Bern