ESBO-ETC: The modular open-source Exposure Time Calculator

Lukas Klass[*,a], **Philipp Maier**[a], **Alfred Krabbe**[a]
[a]University of Stuttgart, Institute of Space Systems, Pfaffenwaldring 29, 70569 Stuttgart, Germany

Abstract. We present ESBO-ETC, a highly modular open-source exposure time calculator written in Python 3. Developed for the European Stratospheric Balloon Observatory (ESBO), the end-to-end simulation tool is now publicly available to the scientific community. ESBO-ETC features an extensive library of components like targets, optical elements and detectors that can be configured and combined via a configuration file to model the optical path. Thus, ESBO-ETC can be used to model and simulate observations with an arbitrary observatory. A modern object-oriented software architecture enables everyone to extend the software with custom components. ESBO-ETC has been verified and validated against other publicly available exposure time calculators yielding a deviation of less than 5% in all cases. To show the maturity of the software we present three simulated science cases with Herschel/HIFI, SOFIA/FORECAST and ESBO/GREAT.

Keywords: Exposure Time Calculator, Astronomy, Telescope, Simulation, Observatory, Astronomical Observations.

*Lukas Klass, lukas@lklass.de

1 Introduction

Different telescope designs, materials and observation strategies are under consideration during the development of the European Stratospheric Balloon Observatory (ESBO).[1] The decision for a specific component requires an estimate of the future observatory's performance given the component's properties. A modular end-to-end simulation tool of an observatory can standardize and significantly accelerate this decision process.

During the observatory's operational life, astronomers submit proposals for their observations. These proposals must include, inter alia, a detailed explanation of the scientific gain as well as an estimate for the necessary observation time. To assess the necessary observation time, a tool to simulate the observation of the specified astronomical target under given observation conditions using the ESBO platform is needed.

In a later stage of ESBO's operational lifetime, principal investigator (PI) instruments will be used besides the facility instruments. Again, a simulation tool is required to enable the developers of PI-instruments to investigate the performance of their detector concepts for a fixed telescope design.

For this reason, we developed the ESBO Exposure Time Calculator (ESBO-ETC), a highly modular exposure time calculator that fulfills all three of the aforementioned use cases. With other observatories in mind, ESBO-ETC has been developed as a versatile open-source software that can be used for the development and observation planning of any observatory.

This paper is structured into several sections. First of all, the features of ESBO-ETC and the usage of the software will be explained in Sec. 2. Section 3 and 4 give some information on the architecture and implementation of the software. This follows an explanation of the performed verification and validation steps in Sec. 5. Three simulated science cases for Herschel/HIFI, SOFIA/FORECAST and ESBO/GREAT are shown in Sec. 6. Section 7 closes with a brief outlook to future extensions of the software.

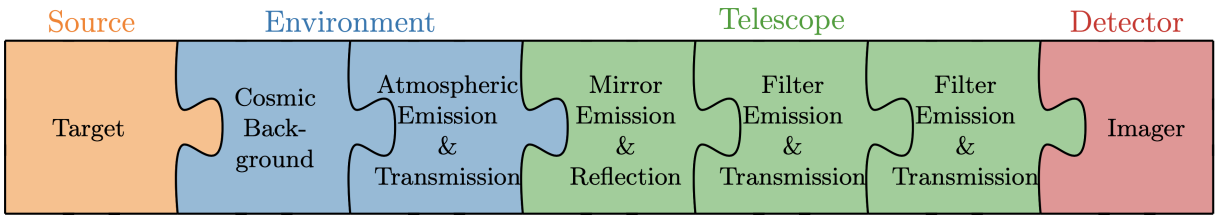

Fig 1 Illustration of ESBO-ETC's approach to model the optical path. Its library provides different implementations of sources, optical components and detectors that can be arranged and configured to fit optical path.

2 ESBO-ETC

ESBO-ETC is a highly modular exposure time calculator written purely in Python 3. The software is open-source and publicly available under the Apache License 2.0 that permits any modification and redistribution. The source code can be obtained from ESBO-ETC's repository given in section *Data, Materials and Code Availability*. An extensive documentation that covers the installation, usage and extension of ESBO-ETC is publicly available online.

The software is designed to model the optical path of the radiative transfer starting with the radiation source and ending with the measurement in the detector. ESBO-ETC offers therefore a library with a wide range of components that can be arranged to model the optical path. This circumstance is illustrated in Fig. 1 where each component is depicted as a jigsaw piece.

2.1 Components

The first release of ESBO-ETC provides 12 components that can be arranged as depicted in the example in Fig. 1. Most science cases and telescope configurations can be simulated using this collection. Nevertheless, more components can be easily added due to the chosen architecture described in Sec. 3. A guide on the development and integration of new components is provided in ESBO-ETC's documentation .

2.1.1 Targets

Targets are used to model sources of astronomical signal radiation in contrast to the background radiation. A simulation can only consider one target component which must be the first component in the optical path. ESBO-ETC supports both point sources and extended sources which differ by the measured quantity: For point sources, only the spectral flux density can be measured whereas the spectral radiance can be measured for extended sources.

- BlackBodyTarget: This component allows to model the thermal emission of a body with a given temperature and emissivity. The brightness of a point source can be defined by supplying its apparent magnitude; the brightness of an extended source can be defined by its surface brightness. The spectral band used to fit the thermal emission to tabulated values[2] for the given brightness has to be provided by the user.

- FileTarget: Arbitrary emissions of an astronomical source can be read by this component from a delimiter separated file. This file must contain two columns: The first column lists the wavelength resp. the frequency and the second column lists the corresponding spectral flux density or spectral radiance of the source. The units of both columns are read from the column headers.

2.1.2 *Optical Components*

The model of the optical path may include any number of optical components between the target and the detector. These optical components can not only alter the incoming radiation i.e. reduction by extinction but may also contribute to the background radiation by thermal emission for example.

- CosmicBackground: The CosmicBackground component can be used to model background radiation of thermal nature from a given temperature and emissivity. As the name suggests, this component can be used to include the cosmic background emission.
- StrayLight: Arbitrary background emissions can be modeled by this component. Analogous to the FileTarget, the values are read from a delimiter separated file containing two columns.
- Atmosphere: This component allows to consider an atmosphere-like element that reduces the incident radiation and emits additional thermal radiation. The emission can be either read from a file or modeled as a gray body radiator of a given temperature.
- ATRAN: The ATRAN component as a subclass of the Atmosphere component can be used to model the atmosphere. In contrast to the Atmosphere component, this component can read output files from the atmospheric transmission calculator ATRAN[3] or request a transmission profile from an online version of ATRAN.
- Mirror, Lens, BeamSplitter: The transmission, reflection and thermal emission of mirrors, lenses and beam splitters can be modeled using these three components. Each component can consider the beam obstruction by other components and their contribution to the background radiation by thermal emission.
- Filter: The influence of a filter in the optical path can be considered using either a file containing the spectral transmission coefficient, a defined spectral band or a given wavelength range. The component may additionally introduce thermal background radiation modeled as a gray body radiator. Again, this component can consider the beam obstruction by other components and their contribution to the background radiation by thermal emission.

2.1.3 *Detectors*

A detector component is used to compute the resulting signal-to-noise ratio (SNR) for a given exposure time or the required exposure time for a given SNR from the incident source and background radiation. As this component ends the optical path, only one can be used in a simulation.

- Imager: The imaging detector component can be used to model a generic imaging detector like a CCD sensor. The signal-to-noise ratio is calculated for a virtual photometric aperture that is computed for a given percentage of encircled energy. Therefore, the telescope's point spread function (PSF) and the pointing jitter are taken into account as explained in Sec. 2.2. The detector's influence on the measurement is characterized by the pixel's quantum efficiency, read noise and dark current.
- Heterodyne: High-resolution spectral measurements with a generic dual sideband (DSB) heterodyne instrument can be simulated using this component. The component requires the specification of the detector's aperture and main beam efficiency as well as the single sideband (SSB) receiver temperature.

2.2 Features

In contrast to many other popular exposure time calculators, ESBO-ETC provides some special features that not only increase the accuracy of the simulations but also improve its usability.

2.2.1 Point Spread Function

Even though most available ETCs take the telescope's PSF into account for the computation of the incident radiation per pixel, ESBO-ETC supports the calculation of the size of a virtual photometric aperture with a given percentage of encircled energy for the imaging detector. Three representations of the PSF can be used: an airy disk, a PSF calculated by Zemax OpticStudio and an arbitrary PSF stored as FITS image. The airy disk can adapt the PSF to the telescope's beam obstruction.

2.2.2 Pointing Jitter

ESBO-ETC models the telescope's pointing jitter as Gaussian distribution with a given standard deviation of the telescope's pointing error.[4] Mathematically, the spatial distribution of the incident radiation is convoluted with the gaussian bell curve to obtain the disturbed image.

2.2.3 Units

A special feature of ESBO-ETC is the consideration of units in all computations. This ensures not only the validity of all equations and avoids unit mismatches of input values but allows the user to use his favored units for all input values. The conversion capabilities of ESBO-ETC include the equivalency of wavelength and frequency as well as resolution, frequency and velocity.

2.2.4 Configuration Check

ESBO-ETC is controlled by a configuration file as explained in Sec. 2.3. Missing or unsuitable values in this configuration file may crash the software or lead to unrealistic results of the simulation. To ensure the validity of the used configuration file, ESBO-ETC performs an extended check of the file during the software's initialization phase. The software provides detailed information and recommendations to the user if any misconfiguration is found.

2.2.5 Modularity

The software architecture of ESBO-ETC was developed to be as modular as possible. New components can be added without having to modify existing files thus making future extensions of the software very simple. Furthermore, the core of ESBO-ETC is written as a self-contained Python module that can be imported and used by other Python programs.

2.3 Configuration File

The components listed in Sec. 2.1 can be arranged and configured to model the optical path as shown in Fig. 1 using a configuration file. This XML-file controls the entire simulation starting from the input values leading up to the output files. An exemplary configuration file is shown in Fig. 2 in two different representations: as raw file contents and as a tree.

A configuration file must contain the following four containers:

(a) (b)

Fig 2 Representation of the configuration file used for the simulation of a heterodyne instrument with ESBO as described in Sec. 6.3 as (a) raw file and (b) tree.

Table 1 Exemplary result output of the achieved SNR for a given exposure time of a simulation with ESBO-ETC.

#	Exposure Time	SNR
1	2.3000e+03 s	2.7273e+00

- common: General parameters and those common to all components are defined in this container. The parameters shown in Fig. 2 are the minimal set of parameters that are required in all configuration files.
- astroscene: This container is intended to accommodate all components required to model the optical path in front of the observatory. The definition of a target is required, all other optical components shown in Fig. 2 are optional.
- common_optics: The optical components of the telescope that are common to all instruments are contained in this container. If no optical elements are shared, this container may be left empty.
- instrument: The last required container might contain some more instrument specific optical components followed by a detector component. Depending on the used sensor, the detector component might contain some more parameters as it is the case in Fig. 2 for the heterodyne detector,

ESBO-ETC makes use of units as explained in Sec. 2.2.3 which have to be appended to the values in the configuration file as can be seen in Fig. 2.

2.4 Output

As command-line based program, ESBO-ETC prints the results of a computation in tabular form to the command-line's stdout. An exemplary output for the computation of the achieved signal-to-noise ratio for a given exposure time is shown in Table 1.

Besides the tabular output written to stdout, ESBO-ETC writes detailed computation results into a directory given in the configuration file. The available formats and the content of the files depend on the used detector type. The Imager detector writes the signal, background, read noise and dark current as total collected electrons as matrices either to CSV- or FITS-files. In the case of the heterodyne detector, the spectral signal temperature, background temperature, RMS noise temperature in Kelvins and the SNR are written to a table in a CSV-file in the output directory.

3 Architecture

The architecture of ESBO-ETC was chosen to make the development of future extensions as simple as possible. As python natively supports object-oriented programming (OOP), two popular design patterns for OOP could be used for the implementation.

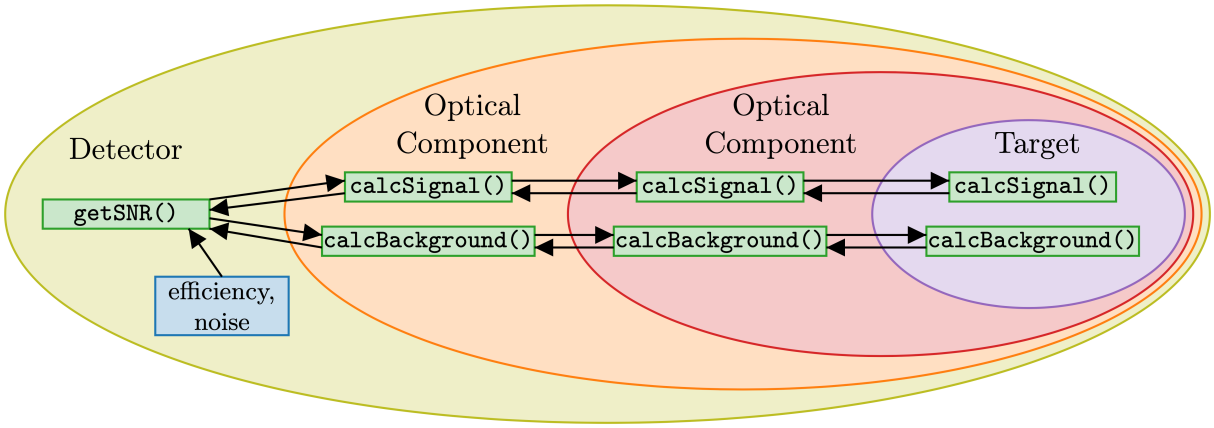

Fig 3 Decorator pattern used for ESBO-ETC. The methods calcSignal() and calcBackground() are called in a cascade from the outermost object to the innermost object.

3.1 Decorator Pattern

ESBO-ETC aims on simulating the emission, transfer and detection of electromagnetic radiation. All involved components in this unidirectional process are defined dynamically in the XML configuration file. This situation can be modeled perfectly using the decorator pattern. This pattern allows to dynamically attach additional responsibilities to an object by decorating the object with another object providing the additional features.[5] The decorating object, therefore, has to have the same methods as the decorated object and forwards any method call to the decorated object before returning the result. The use case of the decorator pattern for ESBO-ETC is shown in Fig. 3.

An astronomical target is always the start of the radiative transfer process and is therefore always the core object of the decorators. It might be decorated by any optical component like an atmospheric model or a mirror component that requests the signal and background radiation from the decorated object. Before returning the quantity, an optical component may increase or decrease the radiation by extinction or emission. Just like in reality a detector forms the end of the radiative transfer and can therefore not be decorated.

The selected architecture for the radiative transfer allows to dynamically add optical components of different kinds between the astronomical target and the detector. Each component may alter the incident radiation in its own way independent from the previous components. Furthermore, future targets, optical components and detectors can be added easily, as they are independent of the other classes and just have to follow the decorator pattern. Thereby, both the open-closed principle as well as cohesion and coupling of OOP are satisfied.

3.2 Factory Pattern

The factory pattern is used in OOP to define ”an interface for creating an object, but let subclasses decide which class to instantiate”.[5] This said, the factory pattern allows to add new components without having to modify the existing factories and fulfills thereby the open-closed principle of object-oriented programming. Besides that, the factory pattern increases to code’s coherence, as the instantiation and usage of objects are strictly separated.

Fig. 4 shows the factory pattern as it is used for ESBO-ETC . The abstract superclass AFactory defines the method create() which has to be implemented by all subclasses for the construction of the corresponding component. Each sort of component of the radiative transfer and detection

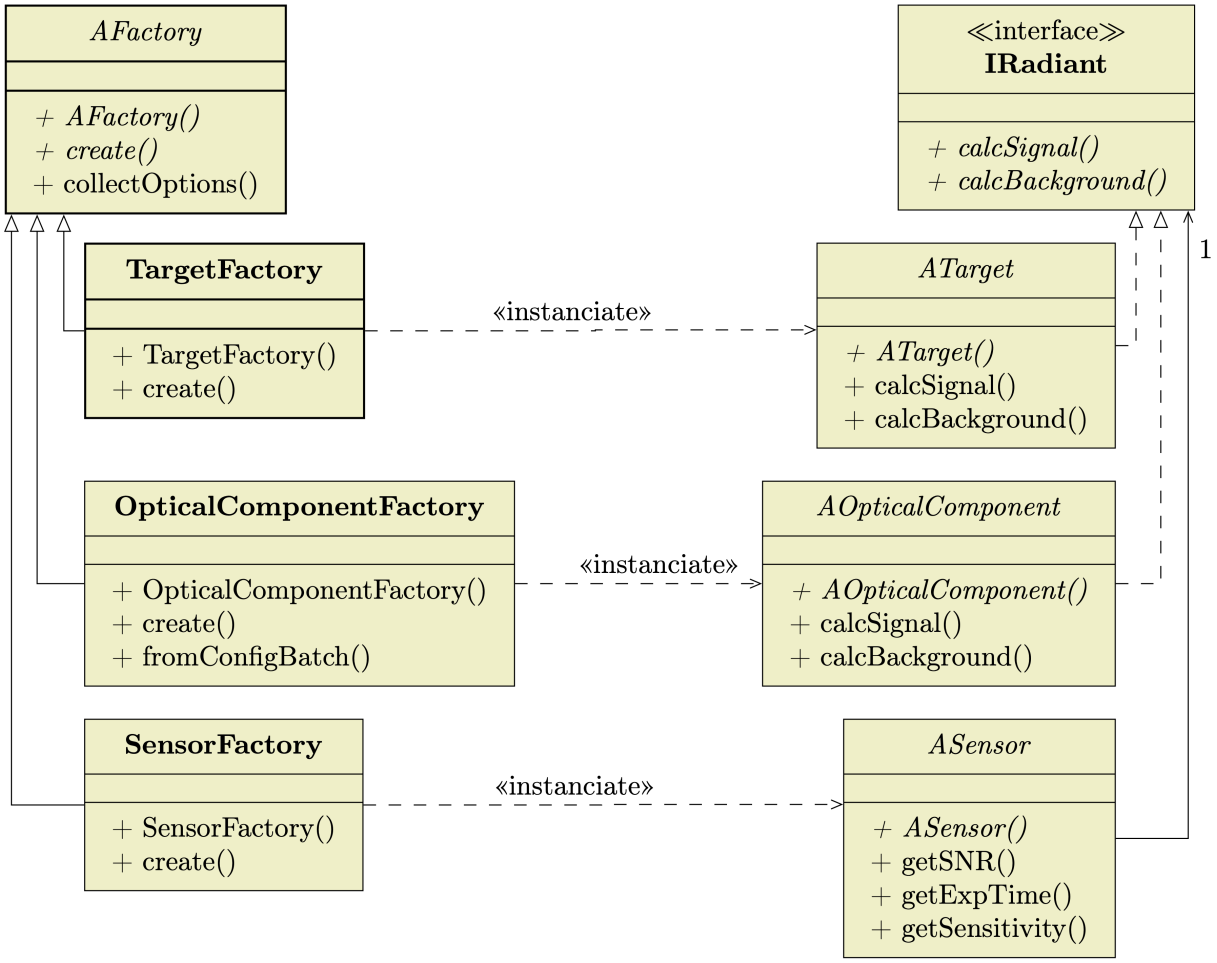

Fig 4 UML representation of the factory pattern used for ESBO-ETC.

process (target, optical component and detector), is created by a separate factory. As the class ASensor does not implement the interface IRadiant, it does not strictly follow the factory pattern. Yet it is shown in the diagram as its corresponding factory SensorFactory inherits from the superclass AFactory. The factory template AFactory additionally provides the method collectOptions() to all factories which allows to collect the required instantiation parameters of the components from the configuration file.

It has to be mentioned, that not the abstract classes ATarget, AOpticalComponent and ASensor are instantiated by the corresponding factories but rather their subclasses. These subclasses have been omitted in the class diagram for the sake of clarity.

To reduce the coupling of the source code, the three factories shown in Fig. 4 do not contain specific code for any class that can be instantiated. This proceeding enforces the open-closed principle as well because new subclasses of the abstract classes on the right-hand side of Fig. 4 can be added without having to modify the corresponding factory.

4 Numerical Approaches

Due to the finite precision of computers, numerical methods had to be used to some extent for the implementation of ESBO-ETC. The resolution of these numerical methods can be controlled by

the user of ESBO-ETC by the two configuration parameters res and osf. The parameter res defines the spectral resolution of spectral quantities as explained in Sec. 4.1 and thereby the precision of spectral integrations. The precision of computations with PSFs like the calculation of the virtual photometric aperture's size (see Sec. 2.2.1) is controlled by the PSF oversampling factor osf.

4.1 Spectral Quantities

The class SpectralQty was developed to handle spectral quantities like the spectral radiance of an astronomical target. The spectral quantity is stored discretized in two arrays: one holding the wavelength bins and one holding the corresponding values of the spectral quantity. To allow the use of these spectral quantities in mathematical equations, the most common mathematical operations are implemented using python's magic methods. Before a mathematical calculation with two spectral quantities can be carried out, their wavelength binning has to match. Therefore, the class' method rebin() allows changing the binning of a spectral quantity using linear interpolation.

5 Verification

The development of ESBO-ETC was driven by the fact, that no available exposure time calculator offers the desired functionalities. This said, the verification of all software requirements required the use of several existing and reviewed exposure time calculators. The Advanced Exposure Time Calculator (AETC)[6] was used for the verification of all target and optical components as well as for the imaging detector. The implementation of the heterodyne detector was verified using the exposure time calculator SITE of the Stratospheric Observatory for Infrared Astronomy (SOFIA).[7]

In some cases, the verification had to rely on manual calculations as no other exposure time calculator implements these functionalities. This was for example the case for the implementation of the telescope's pointing jitter.

A deviation of less than 5% (mostly less than 1%) of ESBO-ETC has been determined in 33 verification cases which is an excellent result. This said ESBO-ETC is a fully verified exposure time calculator ready for scientific application.

47 build tests have been defined throughout the development of ESBO-ETC to ensure that no future changes to the code violate the previous verifications. These build tests not only ensure the functioning of the existing code but also support the developer to locate any breaking changes.

6 Examples

To show the maturity of the software we present three simulations of different science cases with ESBO-ETC. These three applications were used for the validation of the developed software. The simulations including the code of ESBO-ETC and all input data can be freely accessed and executed through Code Ocean capsules. The references are given in section *Data, Materials and Code Availability*.

6.1 SOFIA/FORECAST

SOFIA is a flying observatory for the infrared regime aboard a Boeing 747-SP that provides a variety of scientific instruments. As an infrared observatory, its observations suffer from high thermal emissions of the telescopic components. By replacing the aluminum coating of the primary

Table 2 Simulation results of an observation of Pleione with SOFIA/FORECAST using SITE and ESBO-ETC.

Tool	SNR	Exposure Time
SITE	4.0	252 s
ESBO-ETC	4.0	268 s

and secondary mirrors with a better coating like gold, not only the thermal emission of the mirror can be reduced but also the loss along the optical path.

To assess the influence of the mirror coatings on the integration time, an observation of the star Pleione (BU Tau / 28 Tau) in the Pleiades (Messier 45) with the instrument FORECAST[8] has been simulated using ESBO-ETC. First of all, a simulation with the current telescope parameters has been carried out and compared to the SITE to gain an insight into ESBO-ETC's accuracy. In a second step, the simulation has been carried out for the improved coatings.

Pleione is modeled as a black body point source with a temperature $T_B = 12106$ K and an apparent magnitude $m = 5.19$ mag.[9] The results of the first simulation as listed in Table 2 show that ESBO-ETC's simulation is very close to SITE and verifies therefore once more the implementation of the software. The slight difference of 16 s can be attributed to a possible differing calculation of the number of pixels contained in the virtual photometric aperture.

A second simulation with gold-coated mirrors (reflectivity $\rho = 0.99$) using ESBO-ETC yields a reduction of the required exposure time by 10% per gold-coated mirror.

6.2 Herschel/HIFI

Hydrides are of great importance in the interstellar chemistry and their observation provides key insights into the physical and chemical conditions in their environment. Because of the molecule's small moments of inertia, their rotational transitions lie at high frequencies that are often inaccessible for ground-based telescopes. Due to Herschel's retirement in 2013, no space-based observatory is currently available for observations in this frequency range.

To overcome this issue, the observational capabilities of a heterodyne instrument like GREAT on ESBO has been assessed. For this reason, the observation of the HCl^+ absorption line (1444.2 GHz / 207.6 µm) in the interstellar medium towards W31C, as conducted in the PRISMAS program[10] using Herschel/HIFI,[11] has been modeled with ESBO-ETC. First of all, an ESBO-ETC simulation has been carried out for Herschel/HIFI to gain an insight into the accuracy of the chosen model as described in this section. The analysis of a heterodyne instrument on ESBO is explained in Sec. 6.3.

W31C is modeled as a black body extended source with a temperature $T_B = 31.6$ K and an HCl^+ absorption $\alpha = 0.19$ both derived from the results of the PRISMAS observation[10] (Observation ID 1342206601) in the Herschel Science Archive (HSA). The results of the simulation with ESBO-ETC are listed in table 3. The simulation results are very close to the real observation and differ only by 3.5%.

6.3 ESBO/GREAT

To assess the observational capabilities of the planned observatory ESBO, the simulation of a line observation in the far-infrared as explained in Sec. 6.2 has been conducted for a heterodyne

Table 3 Simulation results of an observation of the HCl^+ absorption towards W31C with Herschel/HIFI using ESBO-ETC as compared to the real observation in the Herschel Science Archive (HSA).

Tool	SNR	Exposure Time
HSA	42.7	405 s
ESBO-ETC	42.7	419 s

Table 4 Simulation results of an observation of the HCl^+ absorption towards W31C with ESBO/GREAT using ESBO-ETC.

Tool	SNR	Exposure Time
ESBO-ETC	42.7	533 s

instrument on ESBO. Therefore, the detector characteristics of GREAT[12] on SOFIA have been used. The results listed in Table 4 show that ESBO can reach almost the performance of Herschel and therefore promises a great scientific gain.

7 Conclusion

In this paper, we presented ESBO-ETC, a modular open-source exposure time calculator to the astronomic community. The software is fully verified and ready to use for scientific applications.

Due to the software's modular design and its open-source licensing, we expect a growth in the functionalities and number of components of ESBO-ETC in the future. Especially the implementation of interfaces towards other atmospheric transmission calculators besides ATRAN will be useful.

In the long run, a graphical user interface GUI has to be developed for ESBO-ETC to enhance the usability of the software. Thereby, all configuration options are immediately visible and no special knowledge in the syntax of XML-files would be required.

Disclosures

L. Klass has nothing to disclose.

Acknowledgments

ESBO DS has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 777516.

Data, Materials, and Code Availability

The source code of ESBO-ETC is available on the git server of the Institute for Space Systems at the University of Stuttgart https://egit.irs.uni-stuttgart.de/esbo_ds/ESBO-ETC. The latest version of the documentation for users and developers of ESBO-ETC is available at https://esbo-ds.irs.uni-stuttgart.de/esboetcdocs/. The three scientific applications explained in Sec. 6 are available on Code Ocean under

- SOFIA/FORECAST[13]
- ESBO/GREAT[14]
- Herschel/HIFI[15]

References

1 P. Maier, J. Wolf, T. Keilig, *et al.*, "Towards a European Stratospheric Balloon Observatory: the ESBO design study," in *Ground-based and Airborne Telescopes VII*, H. K. Marshall and J. Spyromilio, Eds., **10700**, 1470 – 1481, International Society for Optics and Photonics, SPIE (2018). [doi:10.1117/12.2319248].

2 M. V. Zombeck, *Handbook of Space Astronomy and Astrophysics*, ch. Standard photometric systems, 139. Cambridge University Press (2006). [doi:10.1017/CBO9780511536359].

3 S. D. Lord, "A new software tool for computing earth's atmospheric transmission of near- and far-infrared radiation," (1992).

4 H. Hasan and C. J. Burrows, "Telescope image modeling (TIM)," *Publications of the Astronomical Society of the Pacific* **107**, 289 (1995). [doi:10.1086/133552].

5 E. Gamma, R. Helm, R. Johnson, *et al.*, *Design Patterns*, MITP Verlags GmbH (2015).

6 M. Uslenghi, R. Falomo, and D. Fantinel, "AETC: a powerful web tool to simulate astronomical images," in *Modeling, Systems Engineering, and Project Management for Astronomy VII*, G. Z. Angeli and P. Dierickx, Eds., **9911**, 1007 – 1018, International Society for Optics and Photonics, SPIE (2016). [doi:10.1117/12.2233621].

7 P. Temi, D. Hoffman, K. Ennico, *et al.*, "Sofia at full operation capability: Technical performance," *Journal of Astronomical Instrumentation* **07**(04), 1840011 (2018). [doi:10.1142/S2251171718400111].

8 T. L. Herter, J. D. Adams, G. E. Gull, *et al.*, "Forcast: A mid-infrared camera for sofia," *Journal of Astronomical Instrumentation* **07**(04), 1840005 (2018). [doi:10.1142/S2251171718400056].

9 T. R. White, B. J. S. Pope, V. Antoci, *et al.*, "Beyond the Kepler/K2 bright limit: variability in the seven brightest members of the Pleiades," *Monthly Notices of the Royal Astronomical Society* **471**, 2882–2901 (2017). [doi:10.1093/mnras/stx1050].

10 M. D. Luca, H. Gupta, D. Neufeld, *et al.*, "HERSCHEL / HIFI DISCOVERY OF HCL^+ IN THE INTERSTELLAR MEDIUM," *The Astrophysical Journal* **751**, L37 (2012). [doi:10.1088/2041-8205/751/2/l37].

11 B. M. Swinyard, P. Ade, J.-P. Baluteau, *et al.*, "In-flight calibration of theHerschel-SPIRE instrument," *Astronomy and Astrophysics* **518**, L4 (2010). [doi:10.1051/0004-6361/201014605].

12 C. Risacher, R. Güsten, J. Stutzki, *et al.*, "The upGREAT dual frequency heterodyne arrays for SOFIA," *Journal of Astronomical Instrumentation* **07**, 1840014 (2018). [doi:10.1142/s2251171718400147].

13 L. Klass and P. Maier, "Code for the simulation of sofia/forecast in "esbo-etc: The modular open-source exposure time calculator"." https://www.codeocean.com/ (2020). [doi:10.24433/CO.9752252.v1].

14 L. Klass and P. Maier, “Code for the simulation of esbo/great in ”esbo-etc: The modular open-source exposure time calculator”.” https://www.codeocean.com/ (2020). [doi:10.24433/CO.9321102.v1].
15 L. Klass and P. Maier, “Code for the simulation of herschel/hifi in ”esbo-etc: The modular open-source exposure time calculator”.” https://www.codeocean.com/ (2020). [doi:10.24433/CO.3100674.v1].

First Author is an assistant professor at the University of Optical Engineering. He received his BS and MS degrees in physics from the University of Optics in 1985 and 1987, respectively, and his PhD degree in optics from the Institute of Technology in 1991. He is the author of more than 50 journal papers and has written three book chapters. His current research interests include optical interconnects, holography, and optoelectronic systems. He is a member of SPIE.

Biographies and photographs of the other authors are not available.

List of Figures

List of Tables