*The 2010 STScI Calibration Workshop*
*Space Telescope Science Institute, 2010*
*Susana Deustua and Cristina Oliveira, eds.*

# HST Cycle 19 Exposure Time Calculators

R. I. Diaz, I. Busko, P. Greenfield, T. Miller, M. Sienkiewicz, and M. Sosey

*Space Telescope Science Institute, 3700 San Martin Dr., Baltimore MD 21030*

V. G. Laidler

*Computer Sciences Corporation, Space Telescope Science Institute, 3700 San Martin Dr., Baltimore MD 21030*

**Abstract.** The Exposure Time Calculator (ETC) is a web-based application to assists users in calculating the exposure time needed for their HST observations, or the Signal-to-Noise Ratio (SNR) that can be achieved with a given HST observing time. These quantities are key for the preparation of proposals and observations during Phase I and Phase II of the proposing cycle and therefore must be sufficiently accurate for each of the supported observing modes of all the HST instruments. Developing a general tool that shares commonality among the different instruments is complicated, not only from the point of view of attaining accuracy of the calculations but also regarding reliability, portability, and maintainability. We are currently in the final stages of the development of the new ETC for Cycle 19 in Python. This new version improves these qualities and meets the needed level of reliability for the ETC. It also provides a basis for JWST ETCs which are in preliminary development, with a limited-functionality prototype planned to be available in Spring 2011. This papers shows the new tool, its improvements over the previous ETC and the current status of the new version.

## 1. Introduction

The ETC is a web-based application developed to assist HST observers in preparing their observations with any of the supported observing modes for HST. The main purpose of the ETC is to calculate the exposure time needed for observations requiring a particular SNR, and also to ascertain the SNR for simulated observations for a specified exposure time. Currently the ETC provides support for all active HST instruments: ACS, COS, NICMOS, STIS, and WFC3.

The ETC relies heavily on *pysynphot*. *Pysynphot* is a synthetic photometry package that performs photometry by computing detected photons as a function of wavelength (and, for spectrographs, detector bin) using a specified input spectrum, known throughput, and detector quantum efficiency curves (see Figure 1). It uses the throughput tables maintained as part of the Calibration Database System (CDBS) at STScI to compute instrument throughput and efficiency. This information is then used by the ETC to further evaluate the feasibility of the observations for the selected mode and the selected target. The results of the calculations are displayed to a web output form along with any relevant information or flags needed on the planing of the observations (see Figure 2).

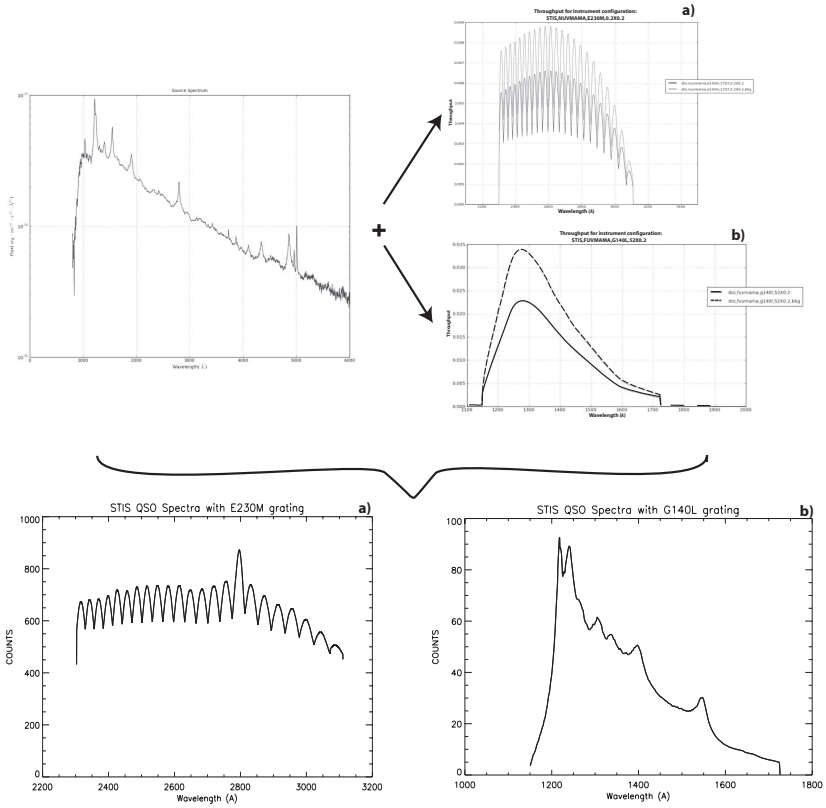


Figure 1: Diagram representing the elements used by *pysynphot* to simulate an HST observation. The top-left plot shows the source spectra. The top-right plots show the transmission curves for a) STIS E230M and b) STIS G140L modes. The bottom plots show the total counts as a function of wavelength for a QSO; as seen by the a) STIS E230M optical element at central wavelength 2707 Åand b) STIS G140L grating.

The ETC provides also the result of the calculations in the form of tables and plots that could be useful to the user. The results are saved in a database and the associated files are retained for future reference.This allows contact scientist to retrieve any observers calculation in case of later questions or problems. It also allows the STScI ETC helpdesk to determine what happened in cases where the observer encountered errors.

Other ETC capabilities include:

- The option to upload an user-input spectra for the ETC calculation. The input spectra can be provided via an ASCII or a FITS file
- Calculation of optimal SNR as would be obtained by PSF fitting
- Inclusion of instrument-specific effects like dark current and read noise
- Bright limit check for those modes where there are safety concerns.

## 2. ETC for Cycle 19

For Cycle 19, the ETC software is being re-written in *Python*. This new version looks to maintain the current functionality of the ETC while being flexible enough to accomodate

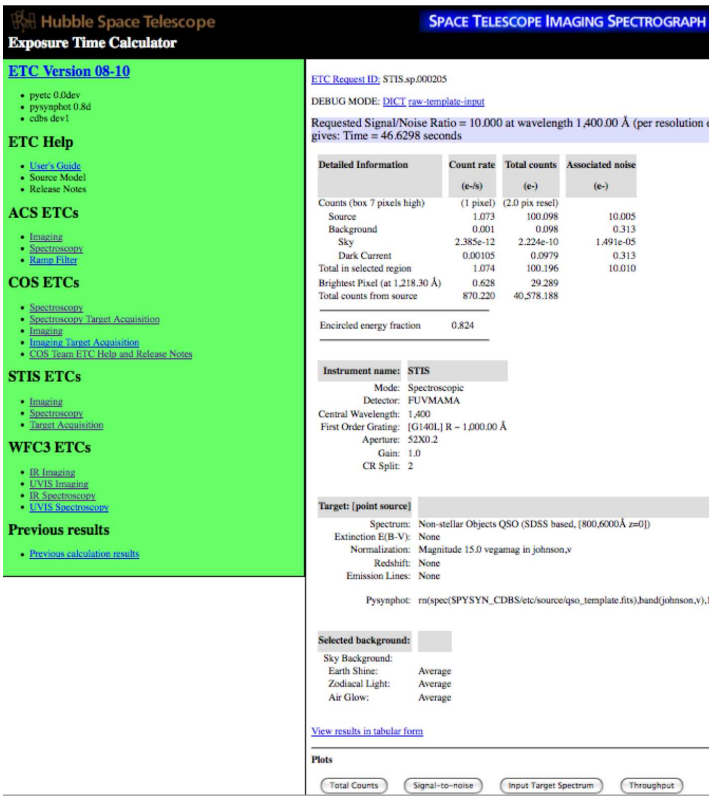


Figure 2: ETC output form.

quick updates to the software and the data; which sometimes is needed for the accurate planning of HST observations. The new ETC will also provide improved performance, reliability and testing. In addition, the computation engine will be readily adaptable to other telescopes, in particular JWST.

### 2.1. The Requirements

Previous experiences managing the Java version of the ETC as well as requests from the HST Instrument Teams and external ETC users were considered carefully when deciding on the configuration and design for the new system. The list of requirements that the new software were to satisfy are:

- *The new system can be installed in one step.* A build of a specific Compute Engine and Web Interface defined by a specific configuration file using a specific (and unique) set of uniform resource locators (URLs). The configuration of an installation includes version numbers of all subsystems, including CDBS.

- *Ability to support multiple installations on the same computer.* Different versions of the ETC can be run concurrently in the same computer.

- *Consistent test scheme and nightly regression testing.* All the instruments have a consistent set of test defined following the same input and reporting format. These test should be run every day in order to identify any problems with the system.

- *Separation of web and computational functionality.*

- *Ability to script ETC calculations from Python without a web server.* Astronomers with *Python* experience can use the calculation engine directly for batch processing or science exploration.
- *Handle failover and load balancing.* Using standard *Apache/Database* server schemes it should be possible to handle failover and load balancing.
- *Simple database structure.*
- *Concentrate instrument information in one place.* Instrument data should be stored and tested in accordance to the the CDBS quality and testing standards.
- *Same look and feel of previous ETCs*
- *Offer interactive form features that weren't working well in the old ETC.*

### 2.2. The Improvements

The functionality and look from the previous ETC is preserved in ETC 19.0. However, when necessary, changes have been made in order to optimize the tool and handling of the code as well to improve the rendering of information to the users.

What are the technological changes?

- The underlying technology (web infrastructure, database, plotting, and so on) has been implemented using several tools according to the part of the system they will manage. These include:
  - *Python*, the underlying script language used by pyetc
  - *Django*, the web framework used by pyetc
  - *MySQL*, the Database system that will store information on each of the ETC calls
  - *matplotlib*:, Used to provide plots of the throughput curves, source spectra, and calculated quantities.
  - *PyFITS*, used to read FITS files
  - *distutils*, provides support for building and installing additional modules into a Python installation
  - *nose*, provides an alternate test discovery and running process for unittest
  - *Pandokia*, a test management and reporting system.
- Version control for both software and data (instrument and model) has been improved
- Configuration and installation procedures have been simplified and standardized
- The underlying database schema has been simplified
- Testing is being significantly expanded so we find problems before users encounter them. New tests cover more of input parameter space as well as better end-to-end testing of the system.

What are the other improvements from previous ETC versions?

- Web pages (HTML) have been simplified & brought into closer compliance with W3C standards (see Figure 1 and 4)
- Internal handling of previous calculations has been simplified

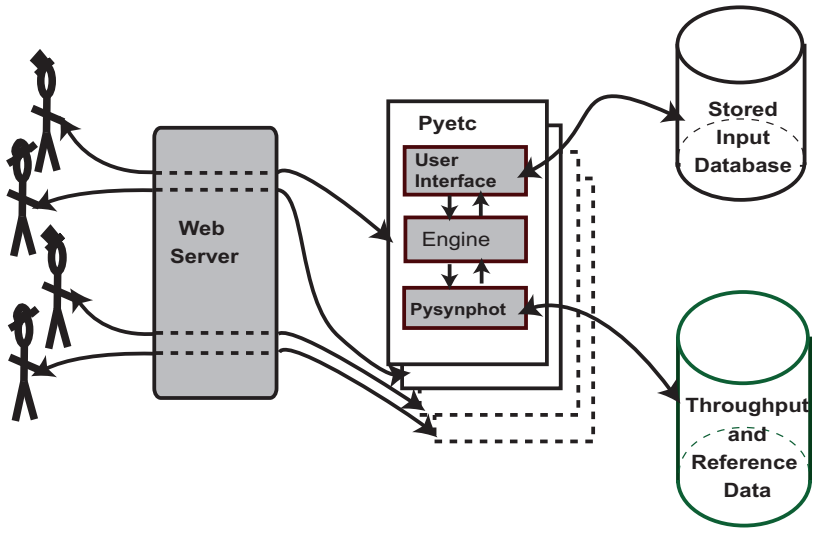


Figure 3: Diagram representing the configuration of the new pyetc

- PyETC is inherently parallel (see Figure 3). It can service multiple requests concurrently on a single computer, taking full advantage of multiple CPUs, if available. If necessary, standard web server load-balancing systems can distribute the workoad across multiple machines to service a greater workload
- This architecture offers improved reliability and performance
- The computation engine is readily adaptable to other telescopes
- The computation engine is being developed to support direct interactive or batch use as well as the web application, and will be publicly released and available to Python-savvy astronomers

## 3. Development Status

Nearly all the basic computational functionality has been implemented, as well as elements of the web interface including the plotting functionality, for all supported instruments and observing modes (ACS, STIS, WFC3, and COS). The new version is being commissioned by comparing its computational results to the previous version, with a maximum acceptable discrepancy of 5% (smaller for some values) except for known/expected differences between the two versions. More that 14000 tests, including old and new tests, are running nightly in an automatic testing and reporting system.

Currently we are completing unimplemented functionality (primarily in the web interface) and working on testing the user interface and load balancing capabilities. As development progresses and we look to resolve test differences, we keep modifying the code and data, as well as adding new test to improve the parameter space coverage. In order to remove redundancy, distributed logic, and dependencies, and make the code more readable and maintainable, we are also working on refactoring the code from the original set.

We are also implementing configuration control for ETC instrument, configuration, and model data via the CDBS system, and implementing the needed tools for the Bright Object Table generation. Before the start of the Cycle 19th call for proposals, and as time allows, we will continue working on other enhancements for the ETCs.

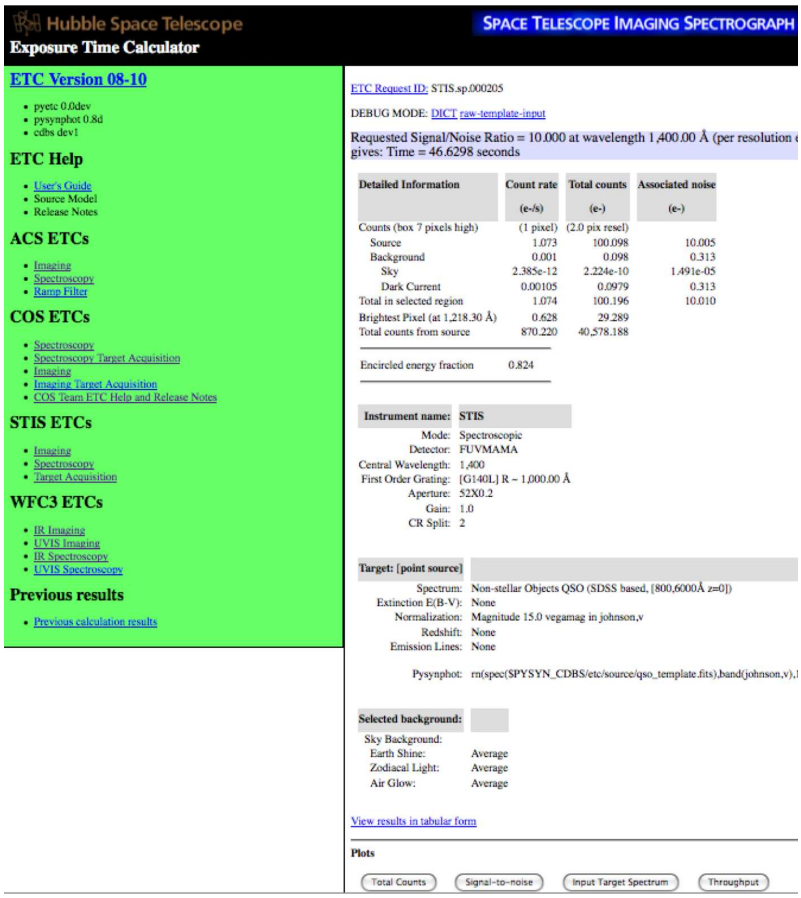


Figure 4: A section of the input ETC form for COS Spectroscopic modes.

We are also drafting the User and Developer documentation and starting to implement support for JWST instruments. The JWST ETCs are in preliminary development, with a limited-functionality prototype to be available in Spring 2011.

## References

Greenfield, P. et al.. 2010, Proceedings of the 9th Python in Science Conference SciPy 2010, to be published

Laidler et al. 2010, "Pysynphot Tutorial" (Baltimore, STScI; http://stsdas.stsci.edu/pysynphot/)