Overview

The class will deliver an exposure time calculator for the 3.5m telescope at Apache Point Observatory that will be tuned and validated using observations taken during our class trip.

Deliverables

The class will deliver either **one** or **two** software products that perform the required task. If two are delivered, they must be written in different languages. *However*, they must both be developed collaboratively using, e.g., the same software design, e.g., names and purposes of subroutines/functions; while they use the same design, however, they should be coded independently (not translated) so that they can serve to check one another. In other words, there should not be two separate groups working on this, just one single group with two implementations.

Each student will separately provide a brief description of their individual contribution to the project, and their overall assessment of the degree to which the collaborative work and the contributions of all group members was effective.

Performance Specifications

The code will be able to accept as target input any of the following:

- the magnitude of an object along with the specified bandpass and an effective temperature (to give the shape of the spectrum)
- the flux of an object at a specific wavelength, and an effective temperature.
- the flux spectrum of an object

The code will accept as instrument configuration input:

- Moon phase (note that the effect of moonlight can be assumed for this purpose as a function of moon phase only)
- seeing
- airmass of observation

The code will accept as choices for output:

- the required exposure time (accurate to 10%) to reach a specified S/N.
- the S/N achived at a specified exposure time.
- the calculated flux for a specified exposure time (will be needed to tune the calculator).

For all input, the code will provide reasonable default choices to the user.

It should be easy for the user to provide input. A graphical interface would probably be nice.

The code will produce the desired output for the given instrument configuration;

- a set of numbers for a set of filters for the imaging instruments, or numbers as a function of wavelength for the spectrographs. Graphical output is probably the best along with a small table, e.g., for the filters and for a few specified wavelengths for the spectrographs.
- information providing the relative noise contributions of the different noise sources: Poisson statistics from object, Poisson statistics from sky, and readout noise.

Code Specifications

The code should be designed to minimize the number of required lines. This will require careful advance thought and planning.

The code should be written so that is is flexible, e.g., so that it could easy be ported for use with another telescope/instrument suite.

The code should be maximally readable and modular. All procedures/subroutines/functions should have a brief description of their function, and a brief description of input and output quantitities.

It should be able to be printed with 80 characters per line without line wrap.

Python code should conform to PEP-8 specifications.

IDL code should use the DOC_LIBRARY standard format for documenting procedures/functions.

Comments should be complete but concise; over-commenting should be avoided. In-line comments should be used minimally, if at all. Documentation (see below) can be more extensive.

White space should be used appropriately, but judiciously; compact code is a plus.

Documentation specifications

The code should be delivered with documentation for how it should be used and how it works. It is strongly encouraged that such documentation be done using a code documentation system such as Sphinx or doxygen.

Personnel specifications

Workload should be divided among students, with some clear written identification of roles and responsibilities. Everyone should participate in code design and planning. Different people could take on different other roles, for example: information collector (details about instruments; mirror, instrument, filter, detector efficiencies as a function of wavelength), programmer, tester, documentation, etc.

Information collection

You will need to get estimates for telescope, instrument, and detector efficiencies. You may be able to get much of this from the web; beyond this, you should consult with Jon Holtzman as your primary source (he can point you to other people/resources if needed).

Note that there will probably be some empirical correction required to make your calculations sufficiently accurate. At some level, an exposure time calculator can be used as an instrument throughput indicator, and you should think about what observations you might require during our class trip to derive any needed empirical corrections. The code should be written anticipating the incorporation of such corrections.

Timeline

The tool should be complete by the last day of classes. The initial planning should be complete by spring break. Some initial code development probably needs to occur before the observing run 3/25-27, so that you can collect required information at the telescope to refine it (note that you have all written initial attempts at this, so much of what you need may already be done in some form).

The group will provide a weekly (every Friday) brief report on progress made during the preceeding week.