Introduction

The Chem**Cam** **P**assive **R**eflectance **O**bservations **Spe**ctral **C**alibration **T**ool (CCAM_PROSPECT) software was developed as part of the NASA PDART contract 80NSSC19K0415 "*An archive of Mars Science Laboratory ChemCam passive visible/near-infrared surface spectra*". It is written in Python and can be run on any platform that has an installation of Python 3. There is both a Graphical User Interface (GUI) option and a command line option for running the tool.

CCAM_PROSPECT calibrates raw Mars Science Laboratory (MSL) Chemistry and Camera (ChemCam) passive surface spectra (archived as "PSV" files in the Planetary Data System (PDS)) into radiance (240-905 nm) and relative reflectance (400-840 nm) spectra. Following the calibration algorithms described in the documentation accompanying the PDS-archived data set (*http address will be here*), standard calibration files are used to calculate relative reflectance. However, users may alternatively select their own calibration files. For each data file that is created, a PDS4 label is also created. Plotting options are provided to display calibrated relative reflectance spectra.

This document outlines how to install the tool, how to run it as a GUI application or as a command line application, and addresses some common issues and troubleshooting.

Installation

Prerequisites:

To install CCAM_PROSPECT, users must have an installation of Python 3.6 or higher, and the capability to install packages and set up a virtual environment. It is standard practice to create a virtual environment to install a new application. This allows users to set up a unique, isolated python environment and install the required packages, rather than installing everything globally. There are several ways to set up a virtual environment, many of which are outlined in the "Installing" section below.

Required:

a. Python 3.6 or higher
   This can be downloaded and installed from the python website here:
   https://www.python.org/downloads/
b. *pip*
   Pip should be installed by default, but directions for making sure it is installed and up to date can be found here:
   https://pip.pypa.io/en/latest/installing/
   When installing the application, pip will be used to install all required external libraries

Optional:

a. *virtualenv*
   Once pip is installed, it can be used to install virtualenv. Installing virtualenv is not necessary, because there is a simpler version distributed with Python 3 called venv. However, a user may choose to use virtualenv instead of venv. To install virtualenv,

use the following command:

MacOS/Linux: `$ pip install virtualenv`

Windows: `$ py -m pip install --user virtualenv`

For more information regarding virtualenv, visit https://virtualenv.pypa.io/en/latest/

b. Anaconda

Anaconda is also frequently used to setup virtual environments. Anaconda can be downloaded and installed here:
https://www.anaconda.com/distribution/
For more information on using virtual environments with Anaconda:
https://docs.conda.io/projects/conda/en/latest/user-guide/tasks/manage-environments.html

Installing:

The tool can be accessed and downloaded from the NASA GitHub website here:
https://github.com/NASA-Planetary-Science/CCAM_PROSPECT

If you are familiar with git and have a git installation, you can clone the repository. Otherwise you can download a zip archive of the source code. Both options can be found under the green "Code" dropdown on the right.

Once users have completed the Python 3.6 (or higher) installation and have downloaded the source code, follow the steps below to install the tool:

1. Install the source code
   * if cloned from git, this step is already done *
   Otherwise, move the archive ccam_prospect-x.x.x.zip to the desired directory and unzip it (where "x.x.x" corresponds to the version number of the software). This will create the following directories under the directory ccam_prospect-x.x.x:
   a. ccam_prospect
      i. constants
      ii. sol76
      iii. templates
      iv. utils
2. Create a new virtual environment
   Change directory into the root of the ccam_prospect project (where it was untarred in step 1) and then use one of the following methods to create the virtual environment named "*env*" here:
   a. Option 1 (Recommended): Use python built-in venv
      MacOS/Linux: `$ python3 -m venv env`
      Windows: `C:\> py -m venv env`
      Note – the python call highlighted in red may vary from system to system, depending on the python installation. To ensure the use of the correct Python installation (3.6 or higher), include the full path to the Python binary. For example, on a Mac, this may look like this:
      ```
      $ /Library/Frameworks/Python.framework/Versions/
      ```

```
3.8/bin/python3 -m venv env
```

or on Windows:

```
C:\> c:\Python35\python -m venv env
```

b. Option 2: Use virtualenv

```
$ virtualenv env
```

c. Option 3: Use anaconda – If using an anaconda distribution of python, conda can be used to create the virtual environment:

```
$ conda create -n env python=3.6 -y
```

Any of these commands will create a folder named "*env*", which holds a python distribution and required packages. This virtual environment will be referred to in the rest of the document as "*env*".

3. Activate the virtual environment

This adjusts some variables in the current shell so that when users type `python` and other python commands, it invokes the python binary inside the virtual environment, rather than the global python environment. Users will install packages here so as to not affect the global python environment.

The method with which to activate the environment depends on how the virtual environment was created. The following commands should either be run from within the *ccam_prospect-x.x.x* directory, or with the full path to the environment prepended. For example, use *full_path/ccam_prospect-x.x.x/env/bin/activate* instead of *env/bin/activate*.

a. If created with Options 1 or 2 (venv or virtualenv), use this command:
   i. bash - `$ source env/bin/activate`
   ii. tcsh - `$ source env/bin/activate.csh`
   iii. csh - `$ source env/bin/activate.csh`
   iv. fish - `$ source env/bin/activate.fish`
   v. Windows - `C:\>.\env\Scripts\activate`

b. If created with Option 3 (anaconda), use conda to activate the environment:

```
$ conda activate env
```

4. Install ccam_prospect in this environment

With a wheel file:

Place the ccam_prospect-x.x.x-py3-none-any.whl file into the directory ccam_prospect-x.x.x. From there run this command:

```
$ pip install ccam_prospect-x.x.x-py3-none-any.whl
```

this will install the ccam_prospect package and all required dependencies into the env "site packages", which will be located within the env folder: *env/lib/pythonx.x/site-packages/*

Without a wheel file:

```
$ python setup.py install
```

5. Deactivate virtual environment
   To deactivate the virtual environment at any point, type
   ```
   $ deactivate
   ```
   This is done to “reset” the environment back to the default. Any usage of python after deactivating the virtual environment will use the global python environment rather than the virtual environment. It can be reactivated at any time by the command in step 3.

Execution

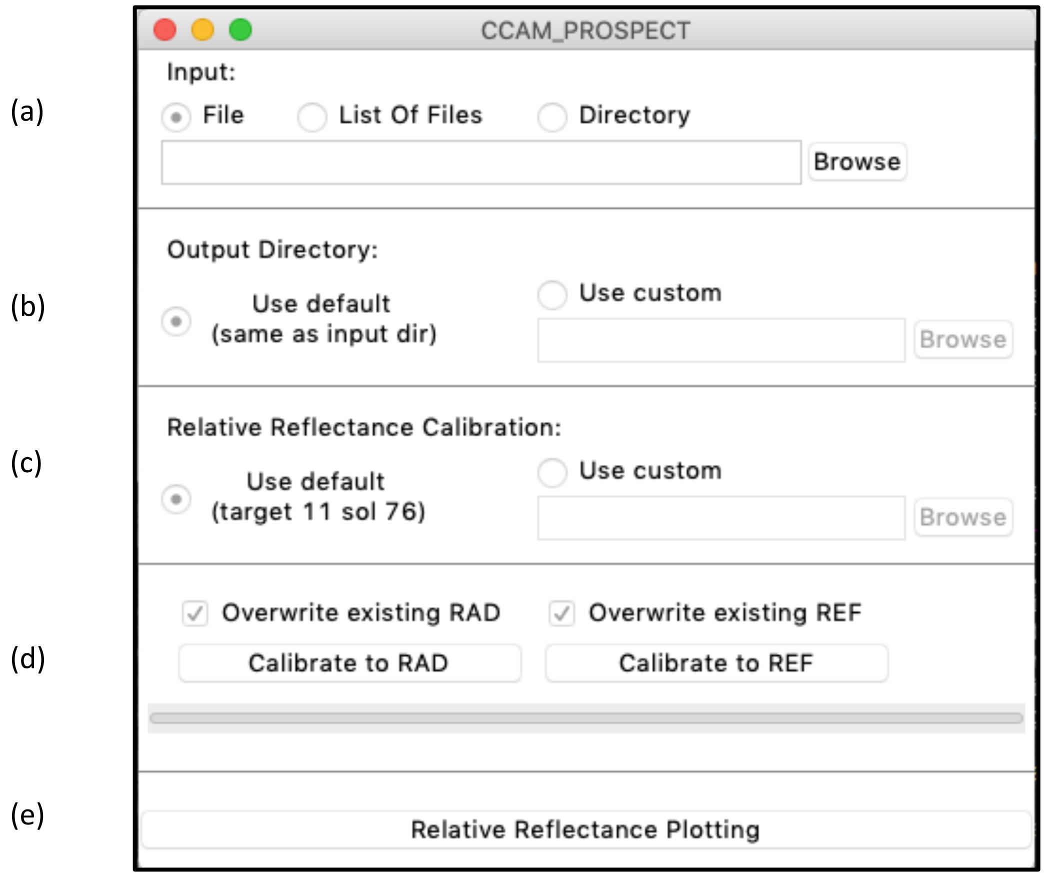
Once installed, the program can be run through a GUI or command line interface. Simply activate the virtual environment (step 3 above) and all required libraries and modules will be ready to use. This is the first step for either method of running the application.

```
$ source env/bin/activate
```
(or whichever shell command in step 3 above applies)

Graphical User Interface (GUI)

To run the program using the GUI, run the following command:

- if run from within the *ccam_prospect-x.x.x* directory:
```
$ python runApp.py
```
- if run from anywhere else, use the full path:
```
$ python full_path/ccam-prospect-x.x.x/runApp.py
```
this will open the GUI, as shown in figure 1.



Input: [✓ File] List Of Files Directory

Output Directory: [✓ Use default (same as input dir)] Use custom

Relative Reflectance Calibration: [✓ Use default (target 11 sol 76)] Use custom

[✓ Overwrite existing RAD] [✓ Overwrite existing REF]

Figure 1. The GUI, on load, with all defaults chosen. (a) the input details (b) the output directory (c) relative reflectance options (d) running options (e) plotting options

The GUI is divided into four sections that are explained below. The letter for each section corresponds to the letter in Figure 1 above. Walking through these four sections in order provides a logical flow for how to set up and run the calibration.

(a) Input:
There are 3 options for input type. Users may select one single file to calibrate, a text file containing a list of files to calibrate, or a directory of files to calibrate. Once the type of input is chosen using the radio buttons, users can choose the full path to the file or directory, as appropriate, by either entering it into the text box or selecting "Browse" to choose from a file browser.
Each input file must have "psv" in the name and end with ".tab" as is found in the PDS archives.
The directory option is recursive, so if users choose a directory as input, any subdirectories will also be searched for PSV files.
A list of files should be input as one single file, with each line of the file containing the full path to the file to be calibrated.

(b) Output Directory:
There are 2 options for output directory. The default option is to use the same directory as the input directory. This will place the calibrated files in the same directory as the raw files. Otherwise, users can select "Use custom" and enter or browse for a custom output directory.

(c) Relative Reflectance Calibration settings:
The default setting for the relative reflectance calibration algorithm is to use calibrated radiance data from the Sol 76 ChemCam calibration target #11 as the divisor for any input radiance spectrum (see PDS archive documentation for details). If the user chooses to use another radiance file for calibration to relative reflectance, users can select "Use custom" and enter a custom radiance file. Because the tool requires that the integration time of the input observation matches that of the reference standard, the custom calibration file should be a radiance file with the appropriate header values. The best way to ensure this is to use a file that was already calibrated to radiance using this tool, or the default Sol 76 target #11 radiance file that is already embedded in the code. If the integration time of a custom calibration file does not match that of the input file, an error dialog will be displayed and calibration of the current file will not continue.

(d) Running Options:
Users can choose to overwrite any existing radiance (RAD) files by selecting "Overwrite existing RAD" and any existing relative reflectance (REF) files by selecting "Overwrite existing REF".

The two buttons at the bottom are used to run the calibration. Selecting "Calibrate to RAD" will run the radiance calibration on input PSV file(s) and output to the appropriate directory, overwriting files as designated using the radio buttons described above.

Running "Calibrate to REF" will run both the radiance calibration and the relative reflectance calibration. Input for relative reflectance calibration can be either a raw PSV file or a RAD file created with CCAM_PROSPECT. If it is a PSV file, the tool will first create a RAD file and then create a relative reflectance file from that file.

Once all desired options and configuration are set, run the program by selecting the appropriate button in the Running Options section of the GUI. Progress will be shown in the progress bar as well as output on the terminal from which you started the GUI.

(e) Plotting Options
Clicking this button will open a separate window to plot relative reflectance spectra. Plotting is discussed in the Plotting Capabilities section on page 6.

Command Line
There is also an option to run the tool via command line. To run the radiance calibration from the command line, users will run the same initial setup step as above:

```
$ source env/bin/activate
```

(or whichever shell command in step 3 above applies)

Then run

```
python full_path/ccam-prospect-
x.x.x/ccam_prospect/radianceCalibration.py [-h] [-f CCAMFILE]
[-d DIRECTORY] [-l LIST] [-o OUT_DIR] [--no-overwrite-rad]
```

running with no arguments or with the -h flag will show a help menu that lists all argument options.
optional arguments:

```
-h, --help      show this help message and exit
-f CCAMFILE     CCAM psv *.tab file
-d DIRECTORY    Directory containing .tab files
-l LIST         File with a list of .tab files
-o OUT_DIR      directory to store the output files
--no-overwrite-rad  do not overwrite existing files
```

Just as in the GUI option, input can be a file, list of files, or directory. Users will select only one of the -f, -d, or -l flags to designate which type of input is provided, followed by that input. The -o flag is used for a custom output directory instead of the default option, which is to output to the same directory as input. All files will be overwritten by default, unless the –no-overwrite-rad argument is used. An example of running radiance calibration from the command line is:

```
$ python full_path/ccam-prospect-
x.x.x/ccam_prospect/radianceCalibration.py -f psvFile.tab
-o /Users/me/out/ --no-overwrite-rad
```

this will run the radiance calibration on psvFile.tab and save the output to the directory /Users/me/out/, but will not overwrite any existing files.

The relative reflectance command line arguments are very similar after following the same initial setup step as above:

```
$ source env/bin/activate
```

(or whichever shell command in step 3 above applies)

Then run

```
python full_path/ccam-prospect-
x.x.x/ccam_prospect/relativeReflectanceCalibration.py [-h]
[-f CCAMFILE]
[-d DIRECTORY] [-l LIST] [-o OUT_DIR] [--no-overwrite-
rad] [--no-overwrite-ref]
```

```
optional arguments:
  -h, --help      show this help message and exit
  -f CCAMFILE     CCAM psv or rad *.tab file
  -d DIRECTORY    Directory containing .tab files
  -l LIST         File with a list of .tab files
  -c CUSTOMFILE   custom calibration file
  -o OUT_DIR      directory to store the output files
  --no-overwrite-rad do not overwrite existing RAD files
  --no-overwrite-ref do not overwrite existing REF files
```

There are two additional optional arguments, *-c CUSTOMFILE*, and *–no-overwrite-ref*. The custom file option is an input file to use as the denominator in the relative reflectance calibration. An example of calibrating a whole directory to relative reflectance using a custom file is below:

```
$ python full_path/ccam-prospect-
x.x.x/ccam_prospect/relativeReflectanceCalibration.py -d
/Users/me/raw_files/ -c custom_rad.tab
```

Because an output directory was not provided in this example, the output will be saved to the same directory, /Users/me/raw_files/. By not providing a *–no-overwrite-rad* or *–no-overwrite-ref* option, any RAD or REF files will be overwritten if they already exist. This command will loop through all files in /Users/me/raw_files/ and its subdirectories, calibrating to both RAD and REF, and overwriting any existing files. Using just the *–no-overwrite-rad* option on its own would use the existing RAD files for the reflectance calibration and overwrite any existing REF files.

For either type of calibration, progress will be printed to the command line.

File formats and PDS Archive

The output files follow a specific naming convention for archive in the PDS, as shown in Table 1.

Input File (from PDS)	cl9_404236313psv_f0050104ccam01076p3.tab
Output Radiance File	cl9_404236313rad_f0050104ccam01076p3.tab
Output Relative Reflectance File	cl9_404236313ref_f0050104ccam01076p3.tab

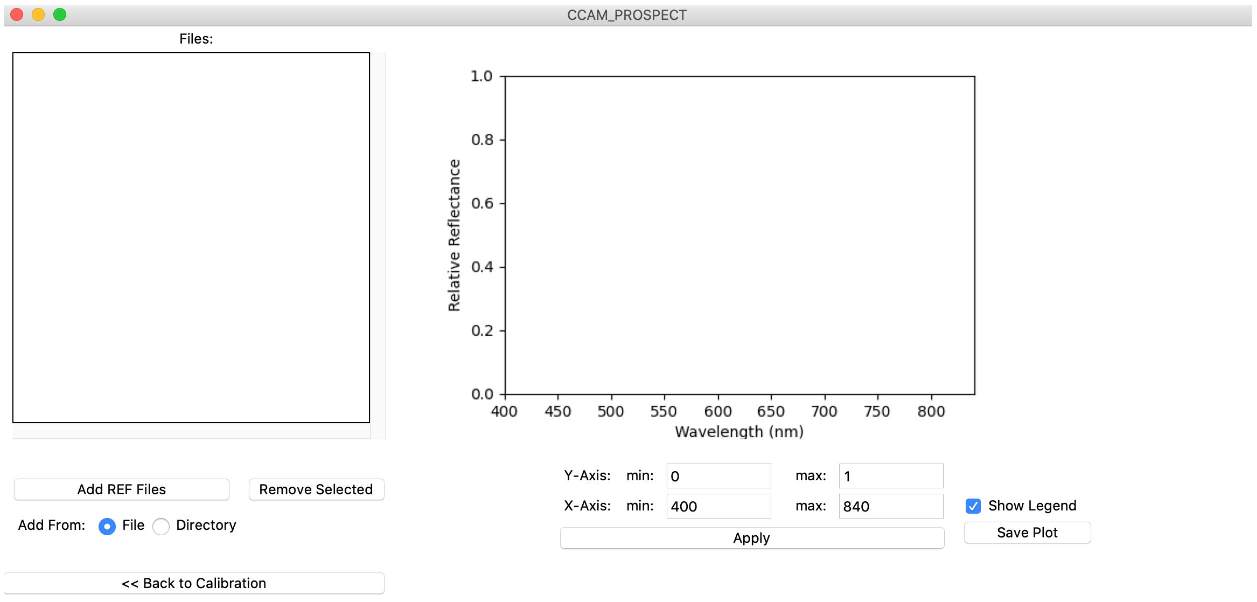
Table 1. File naming conventions for output files.

The format for each of the files will be two-column ASCII tables where the first column is the wavelength (in nanometers) and the second column is either the radiance value (in units of W/m2/sr/μm) or relative reflectance (0.0 to 1.0 for valid values). RAD files (output radiance files) will have the same 29-line header as the raw PSV files, with the table starting on line 30. The data table starts on line 1 of REF files.

A label that follows PDS4 standards will be created for each output file. This is an XML file with information about the RAD or REF file and the source PSV file that it was derived from.

Plotting Capabilities

CCAM_PROSPECT also has a plotting functionality, which can be used to plot relative reflectance spectra. This capability is accessed by clicking the "Relative Reflectance Plotting" button shown in Figure 1 (e). When selected, the GUI will switch to the plotting view, which is shown in Figure 2. On the left side, there is initially an empty list which will hold the REF files that are shown in the plot. The "Add" and "Remove" buttons can be used to populate and edit that list. Once files are added, they will be shown in the list on the left and plotted on the right. Files can be added individually or from a directory. Under the "Add REF Files" button there is a radio button option for adding from File or Directory. When "File" is selected, the file chooser will allow the user to add an individual REF file. When "Directory" is selected, the file chooser will allow the user to select a directory and will add each REF file from the chosen directory. The user can adjust the y- and x-axes with the controls under the plotting area. Lines can be removed from the plot by choosing the file in the list and selecting "Remove". The user can save the plot to a file by selecting "Save Plot" and choosing a location and file format. Once created (by adding lines to the plot), the legend can be moved around by clicking and dragging, and can be hidden by deselected "Show Legend".


CCAM_PROSPECT

Files:

Y-Axis: min: 0 max: 1

X-Axis: min: 400 max: 840

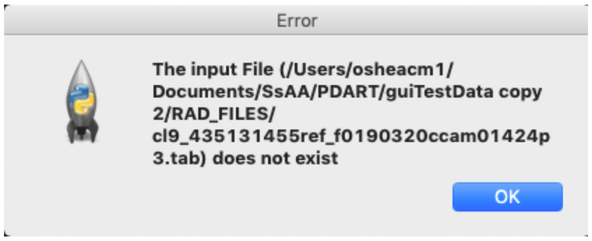
[✓ Show Legend]

Add From: [✓ File] Directory

Figure 2 Plotting view

Troubleshooting

There are several checks in place for input file existence and format. If an input file does not exist, you will see an error dialog such as the one below:



Error

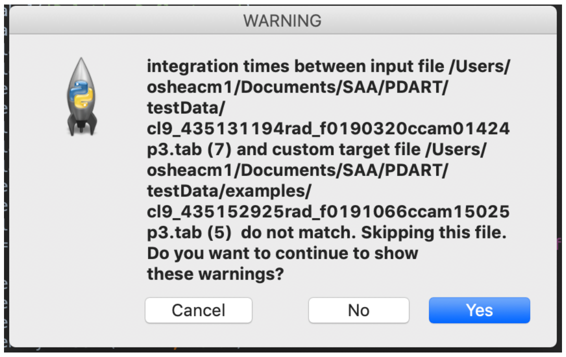
The input File (/Users/osheacm1/Documents/SsAA/PDART/guiTestData copy 2/RAD_FILES/cl9_435131455ref_f0190320ccam01424p3.tab) does not exist

Figure 3 An example Error dialog showing that an input file does not exist

If you see this error, check the spelling/path to the file. It is helpful to use the "Browse" option to ensure the path to the file and spelling are correct.

There is also a log created for each run called "badInput_yymmdd_HHMMSS.log". This log keeps track of every file in a directory or list of files that is skipped, and the reason for it. This most frequently occurs because the file is not a raw PSV or RAD file. The program expects a specific format for filenames. That is, for a raw file to be calibrated to radiance, the filename must include "psv" or "PSV" and end with ".txt". ".TXT", ".tab", or ".TAB". For a RAD file being calibrated to relative reflectance, it must contain "rad" or "RAD".

A file could also be skipped and logged in the bad input log if it is formatted incorrectly. Each raw PSV file and calibrated radiance (RAD) file is expected to contain 29 lines of header information. The PSV file should contain one column of data after the header, and any RAD or REF file should contain two columns of data.



WARNING

integration times between input file /Users/osheacm1/Documents/SAA/PDART/testData/cl9_435131194rad_f0190320ccam01424p3.tab (7) and custom target file /Users/osheacm1/Documents/SAA/PDART/testData/examples/cl9_435152925rad_f0191066ccam15025p3.tab (5) do not match. Skipping this file. Do you want to continue to show these warnings?

Figure 4 An example error dialog for mismatched integration times

For relative reflectance calibration, the two sets of spectral data that are being used in the ratio calculation are expected to have the same exposure times. The files should have one of four exposure times: 7 ms, 34 ms, 404 ms, or 5004 ms. Note that these values stored in the file headers correspond to the commanded exposure times (3, 30, 400, or 5000 ms) plus the 4 ms readout times. If the calculated exposure time is not one of these four times, and/or the exposure times of the two files do not match, the file will be skipped and logged in the bad input log. An error dialog such as the one shown in figure 3 will appear – this prompt will ask the user if they wish to continue seeing similar errors. If "Cancel" is chosen, the calibration will be stopped completely. If "No" is chosen, the calibration will continue and silently skip any file with the same error. If "Yes" is selected, the calibration will continue and will show this error dialog on any future errors of the same type.