CS486C – Senior Capstone Design in Computer Science
Project Description

Project Title:	**CraterStats Python GUI**
Sponsor Information:	Trent Hare, Cartographer USGS Astrogeology thare@usgs.gov

Project Overview:

Counting craters on planetary surfaces is the most common method to determine surface ages in the solar system (e.g., Mars, the Moon). Essentially, the more craters, the older the surface. On Earth, due to erosion, this is not possible and only a handful of craters are still visible on Earth (e.g., Meteor Crater near Flagstaff, AZ). To be able to determine the age of a surface, there are well known methods to model the craters using various plotting methods. These plots, called isochrons plots, can be displayed in cumulative, differential, R-plot and Hartmann presentations (Fig 1).

Two years ago, we (Astrogeology), supported a port of an existing tool, called CraterStats2, from the proprietary application called IDL, to a full open-source version written in Python (https://github.com/ggmichael/craterstats). During the conversion from IDL to Python, we made the decision to abandon the GUI interface, opting for a command-line only version. As Python GUI environments have become more stable, we are now interested in wrapping the command-line Python version with an interactive GUI.

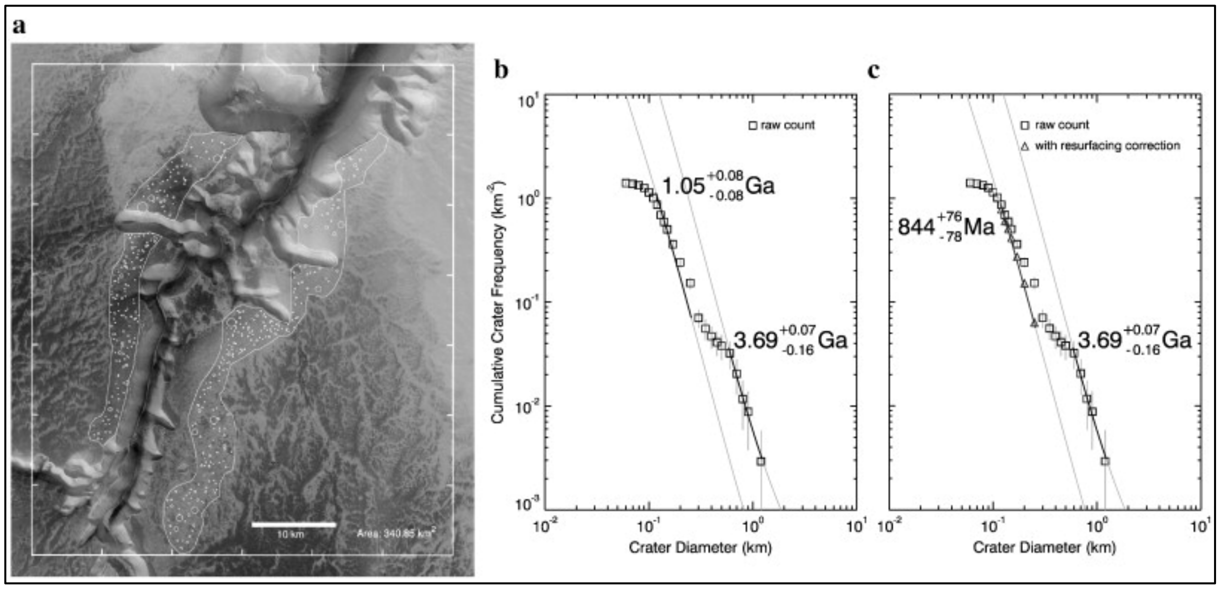
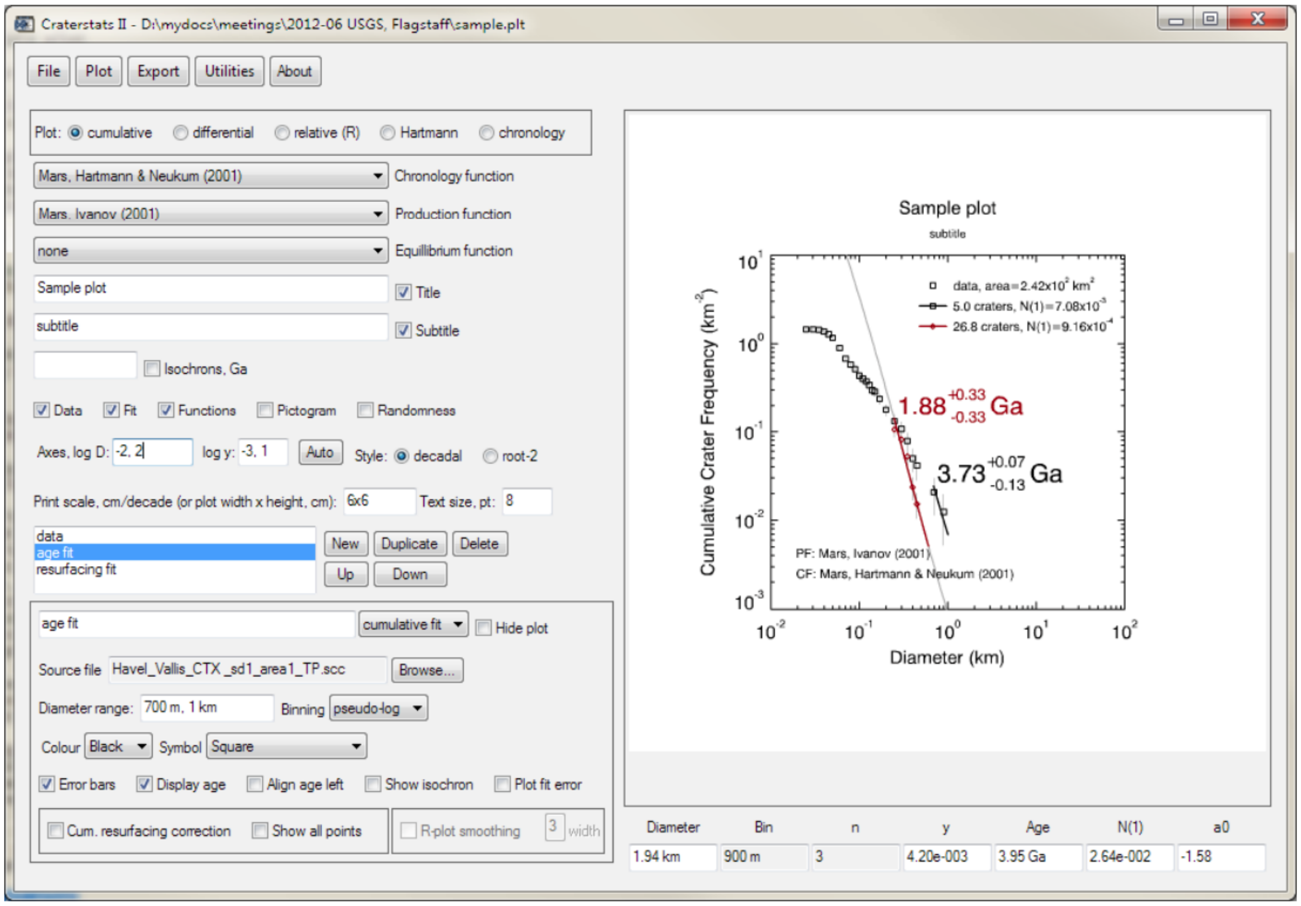


Fig 1. Cumulative crater frequency plots (b,c). Image: https://doi.org/10.1016/j.epsl.2009.12.041

Due the many combinations of options in cratstats, it is currently a burden to allow researchers to visualize the various settings using the command-line tool. Being able to quickly jump from the various isochron plots and their settings is best done interactively in a GUI. The goal is to not manipulate the plot (zoom-in, pan), but the settings which define the plot. The final plot is essential static based on the defined settings prior to selecting a new option. The GUI will also allow the user to tweak aspects of the plot, like the output size and title, which can be beneficial prior to exporting for publication in a presentation or scientific paper.

Fortunately, to complete this project, there is no need to actually learn how to map craters or even understand the plotting routines. The goal for the team to recreate a GUI interface to assist planetary researchers for running these plots. There are two main options to consider for the GUI. First is a stand-alone GUI using an existing Python environment. The second option we would like to see implemented would be web-based environment. This would require staging the interface with in a Python web-based interface like Flask (or as the team defines). Note that in the end, the GUI will be calling the Python-based craters stats to build the plot. As a first step, building the GUI can follow the original design using simple buttons, radio options, and pull-downs (Fig. 2). The current design is somewhat a clutter of options, thus a redesign would be welcome to streamline the interface. For example, adding tabs per section.



Craterstats II - D:\mydocs\meetings\2012-06 USGS, Flagstaff\sample.plt
File
Plot
Export
Utilities
About
New
Duplicate
Delete
Up
Down
Browse...
Auto
Sample plot
subtitle
Cumulative Crater Frequency (km^{-2})
Diameter (km)
10^{1}
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{2}
data, area=2.42x10^{3} km^{2}
5.0 craters, N(1)=7.08x10^{-3}
26.8 craters, N(1)=9.16x10^{-4}
1.88$^{+0.33}_{-0.33}$ Ga
3.73$^{+0.07}_{-0.13}$ Ga
PF: Mars, Ivanov (2001)
CF: Mars, Hartmann & Neukum (2001)

Plot: [✓ cumulative] differential relative (R) Hartmann chronology

Mars, Hartmann & Neukum (2001) Chronology function

Mars, Ivanov (2001) Production function

none Equilibrium function

Sample plot [✓] Title

subtitle [✓] Subtitle

[ ] Isochrons, Ga

[✓] Data [✓] Fit [✓] Functions [ ] Pictogram [ ] Randomness

Axes, log D: -2, 2 log y: -3, 1 Style: [✓ decadal] root-2

Print scale, cm/decade (or plot width x height, cm): 6x6 Text size, pt: 8

data
age fit
resurfacing fit

age fit cumulative fit [ ] Hide plot

Source file Havel_Vallis_CTX_sd1_area1_TP.scc

Diameter range: 700 m, 1 km Binning pseudo-log

Colour Black Symbol Square

[✓] Error bars [✓] Display age [ ] Align age left [ ] Show isochron [ ] Plot fit error

[ ] Cum. resurfacing correction [ ] Show all points [ ] R-plot smoothing 3 width

Diameter	Bin	n	y	Age	N(1)	a0
1.94 km	900 m	3	4.20e-003	3.95 Ga	2.64e-002	-1.58

Fig 2. The original GUI as written in IDL which can be emulated for the initial Python GUI prototype. https://pdsimage2.wr.usgs.gov/pub/pigpen/tutorials/FreieUni_Workshop2012/4_craterStats.pdf

One of the stretch goals I have defined is to also integrate the (same or similar) GUI within the open-source mapping application called QGIS (https://qgis.org/). The QGIS integration can be written in Python using QGIS's existing GUI environment called the Qt library or I imagine almost any other GUI environment the team selects. Really the only enhancement that the QGIS version would enable is the integration for the data directly used as already loaded in the mapping application, instead of supporting a "file open" and for the tool be released as a QGIS plug-in.

Project Requirements

Summary - the goal of this project is to take the existing open-source Python version of craterstats (https://github.com/ggmichael/craterstats) and wrap the command-line functionality within a Python-based GUI environment. The second goal would be to implement the same methods (uploading data and plotting) from a web-based GUI (ideally also written in Python or JavaScript/TypeScript). This web-based version will need to be easily installed via a docker or other method to stand-up a pre-built web infrastructure such that it can be hosted within a cloud-base hosting service (like Amazon).

The capabilities for the Python- and web-based interface need to support are:

- Opening (or upload) an existing crater counting file (examples will be provided)
- Allow the user to select the many options in the GUI as allowed by the current craterstats command-line version.
- Based on the selected settings, run craterstats to create the statically generated plot. This can be done by simply running the options via the command-line (from the GUI) or perhaps via the existing Python code-base.
- Once the user finalizes the plot, export to a supported file format as allowed by craterstats
- Stretch goal 1: Integrate the same or similar GUI within the QGIS mapping application. This would assume the data to be loaded into the mapping application, thus no data load would be needed. This also means the GUI and craterstats will need to be bundles as a QGIS plug-in.
- Stretch goal 2: Integrate wither Python or web-based interface for use within a Jupyter Notebook environment.

Learning craterstats, to define the available options for the GUI, can be gleamed from the robust test suite which comes with the craterstats installation. Also, the options can be itemized from the Python code (cli.py). Our team here at USGS can provide additional guidance for using craterstats and QGIS.

Knowledge, skills, and expertise required for this project

- Experience with Python will be essential as the original application, craterstats, is written in Python.
- Some familiarity building GUIs would be beneficial. The actual GUI environment can be selected by the team. The GUI should work across popular computing environments (Linux, Mac, and Windows).
- For the web-app version, then experience deploying a web environment like docker will be beneficial. But this can also be learned during the project.
- Test data and demo examples, come with craterstats. Supporting those tests is all that is expected.

Equipment Requirements:

- Python environment on any OS. For QGIS, Linux, Macintosh, and Windows builds are supported for free installation.

Software and other Deliverables:

Basic deliverables include (tagged as MVP, Minimal Viable Product, or as stretch goals):

- MVP: Standalone GUI environment for running craterstats. This also includes an interface redesign to streamline the interface. Design updates should be user-tested.
- MVP: Web-based version of the GUI written in Python, JavaScript, or TypeScript which is easily deployed (installed), for example as a docker environment.
- MVP: Design/Architecture documents demonstrating the system architecture.
- MVP: A well written report detailing the design and implementation of the product in a complete, clear and professional manner.
- MVP: Complete professionally-documented codebase delivered as a repository in GitHub, BitBucket,or some other version control repository. Licensed as BSD (same as craterstats) or CC-0.
- Stretch: QGIS version of the Python GUI, made available as a plug-in to QGIS.
- Stretch: Easily installed environment for use within a Jupyter notebook.

References:

- craterstats, https://github.com/ggmichael/craterstats
- Michael G.G., Neukum G., Planetary surface dating from crater size-frequency distribution measurements: Partial resurfacing events and statistical age uncertainty. Earth and Planetary Science Letters 294, 2010. URL: http://doi.org/10.1016/j.epsl.2009.12.041.
- Arvidson, R.E., Boyce, J., Chapman, C., Cintala, M., Fulchignoni, M., Moore, H., Neukum, G., Schultz, P., Soderblom, L., Strom, R., Woronow, A., Young, R. Standard techniques for presentation and analysis of crater size–frequency data. Icarus 37, 1979. URL: https://doi.org/10.1016/0019-1035%2879%2990009-5
- Introductory slide to craterstats: https://pdsimage2.wr.usgs.gov/pub/pigpen/tutorials/FreieUni_Workshop2012/
- QGIS: https://qgis.org/
- QGIS Developers Guide: https://docs.qgis.org/3.28/en/docs/developers_guide
- QGIS Plug-in:
  - https://docs.qgis.org/3.28/en/docs/pyqgis_developer_cookbook/plugins
  - https://www.qgistutorials.com/en/docs/3/building_a_python_plugin.html
- Python GUI, Qt Designer: https://realpython.com/qt-designer-python/
- Python GUI, PyQT6: https://www.pythonguis.com/tutorials/pyqt6-widgets/
- Python GUIs blog: https://www.activestate.com/blog/top-10-python-gui-frameworks-compared/