CRATERMAKER

A MODERN PYTHON-BASED CRATERED TERRAIN EVOLUTION MODEL

DAVID MINTON
DAMINTON@PURDUE.EDU

#2462
LPSC 2024

Earth Atmospheric Planetary Sciences
PURDUE UNIVERSITY

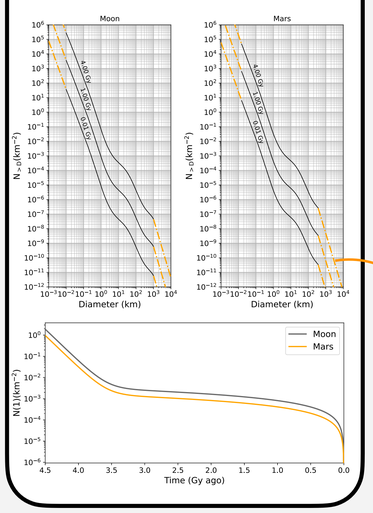
https://cratermaker.readthedocs.io/en/latest/

For more than a decade, our group has been actively developing and using the Fortran-based Cratered Terrain Evolution Model (CTEM). CTEM was originally written to investigate the problem of crater equilibrium [1], but has proven itself to be a useful tool for investigating a variety of problems in cratering science, including: investigating the influence of seismic shaking on asteroid craters [2], testing models for the size-frequency distribution of ancient lunar impactors [3], studying the transport of material across highland/mare boundaries [4], investigating regolith production and regolith evolution [4,5], investing the age distribution of impact melts [6] (see also Blevins et al. #2403), determining the processes that set equilibrium cratering for small simple craters [7] and large complex craters [8], and more. Currently our group is implementing sophisticated new morphology models for complex craters that are based in rigorous data analysis (see Du et al. #1382). Despite its many successful uses, it is a somewhat difficult piece of software to use and develop, and also contains some major structural limitations that make it difficult to apply to some problems.

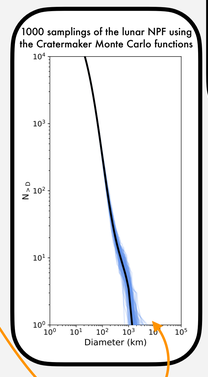
Cratermaker is CTEM on Easy Mode

CTEM is a difficult code to use. It is not well-documented, and requires a great deal of manual set up and configuration to even get working. One of our primary motivations in developing Cratermaker was to improve the ease-of-use. While CTEM has a Python front-end, it is primarily a Fortran code. Cratermaker is primarily written in Python, but with some compiled Fortran libraries for high-performance. Cratermaker can be run "out of the box" with very little configuration required.

When selecting the Moon or Mars as a target, Cratermaker will use a modified Neukum Production Function by default.

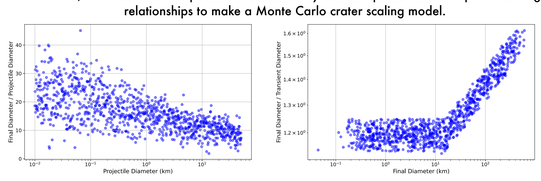


In addition to generating craters (or projectiles) by drawing them randomly from a production function, the user can input custom craters by specifying their size and location. The image here was the result of inputting a lunar crater database into Cratermaker [12]. Combining pre-defined craters and random craters was implemented recently in CTEM as "quasi-Monte Carlo mode," but it is far easier to do in Cratermaker.

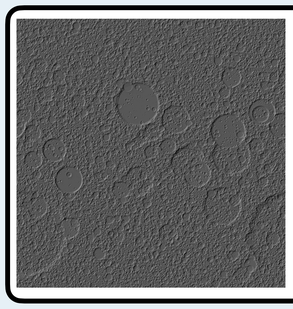


The large end is made steep to prevent overproduction of megabasins.

Similar to CTEM, the default crater size scaling model is based on that of Holsapple (1993) [1,11]. However, we have also incorporated the uncertainty values reported in the empirical scaling relationships to make a Monte Carlo crater scaling model.

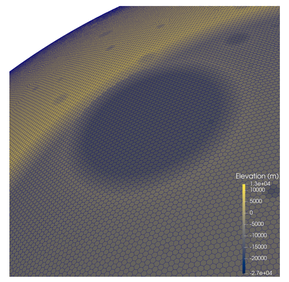


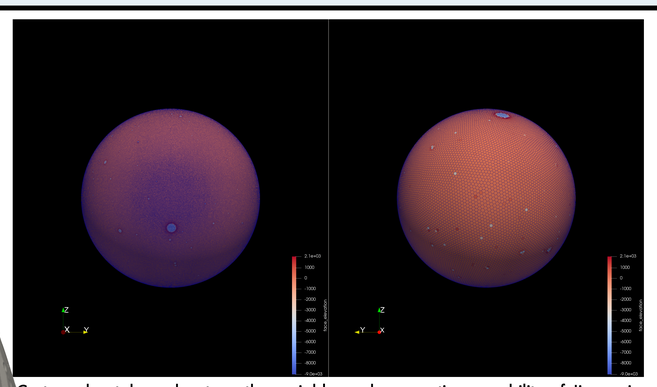
Escaping the Tyranny of the Flat Square Grid



CTEM represents a surface as a flat, square grid of pixels with a repeating boundary. For small regional "postage stamp" simulations, this approximation (mostly) works, but for global simulations, this geometry has proven to be very difficult to work with. When running in Quasi Monte Carlo mode, where "real" craters are emplaced along with randomly-generated ones, the relative distances between the real craters does not reflect their true distance on the spheroidal Moon. In addition, CTEM's geometry is not able to model antipodal deposits of ejecta.

Cratermaker represents the surface of a body using an unstructured mesh, which is generated using an open source tool called JIGSAW [11]. The JIGSAW mesh generator is used by Model for Prediction Across Scales (MPAS), which is an Earth-systems software package, and Cratermaker's data storage structure is modeled after that used by MPAS. This allows the Cratermaker user to take takes advantage of the extensive data analysis tools developed in the Earth sciences. For instance, the images shown here were generated using the ParaView visualization program, which is commonly used to visualize Earth-science data.

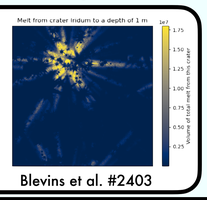




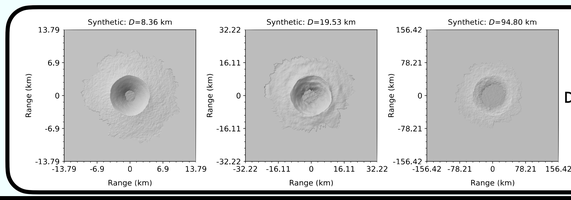
Cratermaker takes advantage the variable mesh generation capability of Jigsaw in order to model small patches at high resolution without the need of the "superdomain" that CTEM uses to model the effects of large, distant craters outside of the region of interest.

Development Roadmap

Cratermaker development only began in October 2023, so a number of important capabilities of CTEM have not yet been implemented in Cratermaker. Our goal in the coming months is to get the automated crater counting and topographic diffusion algorithms from CTEM implemented in Cratermaker. Purdue graduate student Austin Blevins plans to incorporate the impact melt and material tracking models, and Purdue postdoc Jun Du plans to incorporate our new realistic topography model as well.



Blevins et al. #2403



Du et al. #1382

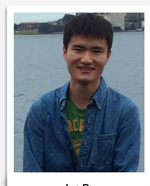
This work is made possible by:
NASA Lunar Data Analysis Grant #80NSSC21K1719

	Projectile->Crater	Crater->Projectile	NPF built in	Quasi-MC	Crater counting	Topographic diffusion	Material tracking	Melt generation	Seismic shaking	Realistic topography
CTEM	✅	❌	❌	✅	✅	✅	✅	✅	✅	❌
Cratermaker	✅	✅	✅	✅	❌	❌	❌	❌	❌	❌

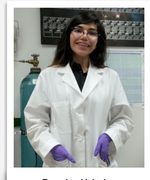
The Cratermaker Development Team

David Minton
Head of Development

Austin Blevins
Impact melt tracking
Quasi-Monte Carlo
Documentation



Jun Du
Morphology model



Dennise Valadez
Degradation model
Documentation

You
Your contribution

Cratermaker is still in the early stages of development, but we have already made the code publicly available and welcome contributions from interested members of the community.

Contact me if you would like to join our GitHub Project page and help contribute to the project.

References

[1] Richardson J. E. (2009) Icarus 204, 697–715.
[2] Richardson J. E. et al. (2020) Icarus 347, 113811.
[3] Minton D. A. et al. (2015) Icarus 247, 172–190.
[4] Huang Y.-H. et al. (2017) J. Geophys. Res. Planets 122, 1158–1180.
[5] Richardson J. E. and Abramov O. (2020) Planet. Sci. J. 1, 2.
[6] Huang Y.-H. et al. (2018) Geophys. Res. Lett. 45, 6805–6813.
[7] Minton D. A. et al. (2019) Icarus 326, 63–87.
[8] Riedel C. et al. (2020) J. Geophys. Res. Planets 125, e2019JE006273.
[9] Zhao C. et al. (2016) J. Adv. Model. Earth Syst. 8, 1751–1768.
[10] Engwirda D. (2014) Locally Optimal Delaunay-Refinement and Optimisation-Based Mesh Generation, The University of Sydney.
[11] Holsapple K. A. (1993) Annu. Rev. Earth Planet. Sci. 21, 333–373.
[12] Losiak A. et al. (2009) LPSC40 id.1532 1532. (2015 revision by Ohman)