GEOMORPHOMETRY 2021
PERUGIA, ITALY
SEPT 13 - 17 2021
PARTNERS SPONSORS

A New Analytic Framework and Notebook for Terrain Analysis

Charles Holderman, Steve Kopp[§], Nawajish Noman, Tania Lopez-Cantu
Esri
Redlands, California
[§] skopp@esri.com

Abstract— **Terrain analysis algorithms evolved along the same timeline as digital elevation model (DEM) spatial resolutions, with the typical implementation using only immediately adjacent cells (3x3 window) being appropriate for the landscape processes of interest. Finer resolution DEMs require new approaches to effectively model terrain forms and processes. Here, we present a new approach based on recent innovations and some longstanding ideas in terrain analysis, that allow users to control the scale of analysis through a window size, use improved polynomial surface fitting, and utilize geodesic calculations. This new framework has been implemented in ArcGIS to calculate aspect, slope, and multiple curvature types. To share our implementation methodology and enable others to build from it we released a Python implementation of the tool in a modular tutorial style as a Jupyter Notebook. Researchers and analysts can use this notebook to further experiment with the tool parameters and implement additional terrain metrics of interest.**

I. INTRODUCTION

The *Exploring Surface Parameters* project is an open source executable notebook implementation of the ArcGIS Surface Parameters tool [1]. This new tool introduces several recent community ideas on how to improve calculation of terrain descriptors. The goals of the project are threefold; to create more representative surface metric results from high resolution terrain data through the use of extended neighborhoods and quadratic surface fitting, to create a more accurate result anywhere in the world in any coordinate system by performing all computations in geodesic space, and to provide a better user experience by eliminating confusions over curvature types and any concern about map projections.

II. SOFTWARE FEATURES

The Exploring Surface Parameters software is written in Python and distributed as a BSD licensed Jupyter Notebook. Jupyter notebooks are a widely used and effective method for publishing open science algorithms and workflows [2]. The notebook is written in a modular tutorial style explaining key features and their options, then presenting the code, and result visualization. The modular approach isolates the calculation of surface metrics so adding or modifying a surface calculation is only a few lines of code.

A. Neighborhood distance

Issues related to scale in the Geomorphometry community have been well explored and described by [3, 4, 5] and more. Landscape features exist at a range of sizes and elevation data is rarely created specifically for computing landscape indices, so there is often a mismatch between the spatial resolution of available data and the landscape features of interest. A common approach is to resample the DEM, however resampling elevation affects the output surface parameters [6, 7]. DEM resampling tools already exist in ArcGIS, implementing a neighborhood distance option provides users with an alternative approach without degrading the original DEM values.

The scale of analysis, in this tool and notebook is defined by the size of the moving window, specified as the *neighborhood distance.* The neighborhood distance is the map distance from the center of the current processing cell to the center of an orthogonal neighbor.

The primary purpose of the neighborhood distance parameter is to choose an appropriate distance (scale) to describe the characteristics of the landform or feature of interest. A smaller neighborhood distance captures more local variability in the landscape, which yield characteristics of smaller landscape features. With higher resolution elevation data, a larger number of cells in the processing window can be more appropriate when

the landform of interest is more recognizable at larger distances. A secondary use of the neighborhood distance is for multi-scale analysis, evaluating terrain metrics at a range of neighborhood distances [4, 8].

Modern high-resolution elevation data in the range of 5 meter or less pixel size often contain fine scale noise that does not reflect terrain features of interest. Using a neighborhood distance that is more appropriate for the terrain features minimizes this problem, Fig 1.

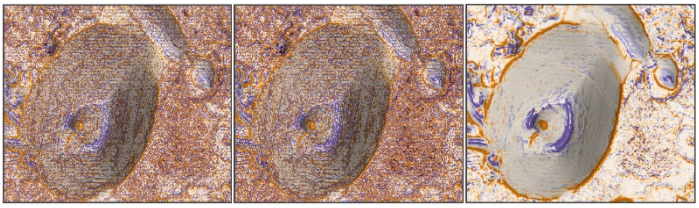

Figure 1. *Profile curvature computed on a 5 meter DSM with 3x3 (left), 5x5 (middle), and 15x15 (right)cell windows.*

In application, landscapes vary such that an appropriate neighborhood distance to describe a creek in one part of a study area may not be appropriate to describe a plateau in another part of the study area. Using too large of window size will hide characteristics of the creek, and using too small a window size will highlight insignificant features or noise on the plateau, Fig 2. To resolve this, we implemented an adaptive neighborhood distance following the approach described by James [9].

Figure 2. *Simulated adaptive window size over varying terrain.*

By evaluating the local variation in the neighborhood of each cell, the tool chooses an appropriate window size to capture the terrain features while minimizing the influence of noise. The implementation builds upon earlier work [9] using deviation from mean elevation *DEV* [10] as a proxy for surface complexity within the window. This metric is computed as the difference from the mean divided by the standard deviation.

$$\mathrm{DEV} = \frac{z_0 - \overline{z}}{\mathrm{SD}}$$

The DEV is computed on a 15x15 cell window, and the window progressively decreased in size until a threshold is met. In the ArcGIS Surface Parameters tool and the notebook, we use a threshold of 0.1. The use of DEV as the local measure of surface complexity and the threshold values used for choosing window size may be areas for further exploration by users of the notebook.

B. Geodesic calculations

The early decades of GIS and DEM-based terrain analysis were dominated by algorithms using planar geometry. During this time commonly used national and global extent elevation datasets such as the USGS National Elevation Dataset, NASA Shuttle Radar Topography Mission, and others were published with Earth-centered spherical coordinates (latitude, longitude), not in a planar cartesian coordinate system. This created confusion for software users processing spherical data with planar software, who were required to specify an appropriate scale conversion (Zfactor) for the latitude of their study area. This conversion factor provided an approximate conversion at a single latitude, but did not provide an accurate distance conversion across the study area extent. The size of study area was limited by the amount of distance distortion error the user was willing to accept. A common alternate approach was to reproject the DEM to a planar coordinate system, however reprojection and resampling affect the calculation of surface parameters [6, 7], and still incur distance distortion from the projection, limiting the spatial extent of accurate analysis. In recent years many geospatial software including ArcGIS, Google, PostGIS and others are writing software using geodesic algorithms instead of planar algorithms [11].

Computing distance is part of most geomorphometric calculations, and since no planar coordinate system can compute true distance in all direction from any location [12], we developed new DEM analysis algorithms using geodesic distances and angles instead of planar. Geodesic algorithms for slope and aspect were added to ArcGIS in 2017, and curvature in 2020, and are fully implemented in the Jupyter notebook. In addition to creating a more accurate result, using geodesic calculations also simplifies the user experience by reducing the need for a conversion factor. The conversion factor is now only used in situations where the DEM vertical units are not meters.

C. *Surface fitting*

Many terrain analysis algorithms operate by fitting a surface to a neighborhood of cells. For calculations such as slope and aspect, fitting a plane through the immediate 3x3 neighborhood was the traditional approach [13, 14]. However, a plane is seldom a good descriptor of the landscape and may mask or exaggerate natural variations of interest.

For curvature calculations a biquadratic surface [15] has been a common approach in ArcGIS, GRASS, and others. The biquadratic (also referred to as partial quartic, or fourth order polynomial) surface provides a tight fit to the data and is sensitive to noise and errors [16]. The impacts of noise and errors on terrain metrics are increasingly common with modern high resolution DEMs.

The quadratic (or second order polynomial) surface was found to provide a better surface representation for computing terrain characteristics from a DEM [3, 8, 16, 17]. The quadratic surface is a least squares fit of the points and does not pass exactly through all points. By not passing exactly through all points, the quadratic surface minimizes the impact of noisy surface data and creates a more representative surface metric output, which is especially important when computing curvature.

The new tool and notebook use a quadratic surface by default. The biquadratic approach is also provided in the notebook for comparison.

D. *Surface parameter types*

For the first release of the Surface Parameters tool and notebook we implemented slope, aspect, and several types of curvature, to improve upon their implementations in ArcGIS.

The tool has been developed with the intent to include additional terrain metrics (surface parameter types) in future releases while maintaining version compatibility. This is accomplished by including all surface parameter types in a keyword list. This pattern is also followed in the notebook.

The first release of the notebook is particularly focused on supporting multiple types of curvature. The history and confusion regarding names and algorithms for curvature types has been well documented [18, 19]. For maximum clarity the ArcGIS Surface Parameters tool and the notebook use the naming conventions and agree with the formulas in Minár [18]. In the notebook we have implemented the 3 geometric curvatures described as the "basic trio" [18] directly from the derivatives of the surface: Normal Slope Line Curvature, Normal Contour Curvature, and Contour Geodesic Torsion Curvature.

Normal Slope Line Curvature (Profile)

$$K_P = -\frac{f_{xx}f_x^2 + 2f_{xy}f_xf_y + f_{yy}f_y^2}{\left(f_x^2 + f_y^2\right)\left(f_x^2 + f_y^2 + 1\right)^{3/2}}$$

Normal Contour Curvature (Tangential)

$$K_T = -\frac{f_{xx}f_y^2 - 2f_{xy}f_xf_y + f_{yy}f_x^2}{\left(f_x^2 + f_y^2\right)\left(f_x^2 + f_y^2 + 1\right)^{1/2}}$$

Contour Geodesic Torsion Curvature

$$S_c = \frac{f_xf_y(f_{xx} - f_{yy}) - f_{xy}(f_x^2 - f_y^2)}{\left(f_x^2 + f_y^2\right)\left(f_x^2 + f_y^2 + 1\right)}$$

From these 3 curvatures the notebook shows, with simple arithmetic, how to derive additional geometric curvatures including Minimum, Maximum, Casorati, and Gaussian curvatures. Users of the notebook can add their own new types, while taking advantage of the geodesic math, window size, and other capabilities of the notebook mentioned above.

III. Results and Discussion

A Jupyter Notebook was created that allows users to explore and follow step-by-step the internal algorithms and logic in the ArcGIS Surface Parameters tool. Using this notebook, users can (1) understand the main steps carried out in the Surface Parameters tool, (2) review the code block of each step and understand at the algorithm level the different decisions and calculations carried out in each step, and (3) understand the impact of parameter choices (e.g. neighborhood distance, polynomial fit, etc) by testing with different parameter values.

Most of the notebook relies on standard Python libraries, however, some modules from the ArcGIS ArcPy Python package were used for Raster data I/O and coordinate transformation. The ArcPy package provides optimized modules and methods for these specific tasks and it was used in the notebook to take advantage of its capabilities. Therefore, the ArcPy package (part of the default Python distribution installed along with ArcGIS Pro and ArcGIS Server or through ArcGIS API installation) is required to run the notebook. The software is widely available internationally in Universities and government institutions, and also available for free trial usage.

To support researchers running without the ArcPy modules, we have broken down the code into multiple individual functions to facilitate the replacement of the data I/O and

coordinate transformation blocks. Alternatively, users can copy the functions of interest and use them within their own framework.

To run the notebook, users familiar with ArcGIS Pro can open the notebook within the application. Additionally, the notebook can also be opened through ArcGIS Notebook Server in an ArcGIS Enterprise portal, or through ArcGIS API for Python using the command *jupyter notebook* in a terminal session running with the appropriate conda environment where ArcGIS API has been installed. After the notebook has been opened using any of the methods described above, the user can run the analysis cell-by-cell in the same way as any other Jupyter Notebook.

The notebook is organized in a tutorial style, starting with a brief introduction and explanation of the high-level workflow and parameters that can be modified, as well as detailed explanations of each workflow step and comments throughout the code. The only requirement for users is to change the input and output folder destination and file name.

With respect to the ArcGIS Surface Parameters tool there are several expected differences between the calculations performed in the notebook that will result in slight differences in output.

First are slight differences on the order of 10^{-7} due to differences in the float precision and math libraries between Python and C++. Where these differences are seen, the ArcGIS Surface Parameters tool is considered more accurate than the notebook.

Second, within the notebook framework, we handle two scenarios regarding missing values. Missing values are referred to in ArcGIS and the sample notebook as NoData cells. First, an input NoData cell will remain NoData in the output raster. Second, an output cell will be NoData if there are fewer neighboring cells within the chosen window size than are required by the particular surface being fit: 6 for a quadratic surface and 9 for a biquadratic surface. In the ArcGIS Surface Parameters tool, these scenarios are handled differently at the edge of a raster surface given the type of window selected for analysis (fixed or adaptive), thus the notebook and tool results will be slightly different in the number of NoData cells at the outer edge of the output raster data.

In conclusion, we have presented a new approach for calculating surface characteristics that integrates scale control, polynomial surface fitting and geodesic calculations. This framework, already available in ArcGIS, was fully implemented in Python and Jupyter Notebook. This notebook serves as a learning and experimentation framework to demonstrate the algorithms and logic in the ArcGIS Surface Parameters tool. The tool and notebook introduce several recent or new ideas to the community, and we envision that this well-documented notebook will enable others to explore our techniques, critique them, and improve upon them.

To view the notebook in a web browser -
https://www.esriurl.com/SurfaceParametersPreview
To download the notebook -
https://www.esriurl.com/SurfaceParameters

References

[1] Esri, (2020). "How Surface Parameters Works" in ArcGIS Pro online documentation. Accessed 7/20/2021 at https://pro.arcgis.com/en/pro-app/latest/tool-reference/spatial-analyst/how-surface-parameters-works.htm
[2] Randles, Bernadette M., I. V. Pasquetto, M. S. Golshan, and C. L. Borgman, 2017. "Using the Jupyter Notebook as a Tool for Open Science: An Empirical Study." In 2017 ACM/IEEE Joint Conference on Digital Libraries (JCDL), 1–2. Toronto, ON, Canada: IEEE. https://doi.org/10.1109/JCDL.2017.7991618.
[3] Wood, J. D., 1996. "Scale-based characterization of digital elevation models", in Innovations in GIS 3, edited by D. Parker, chap. 14, Taylor and Francis, Philadelphia, PA.
[4] Li, Z., and Q. Zhou, B. Lees, and G. Tang, 2008. "Advances in Digital Terrain Analysis". Springer, 462 p.
[5] Drăguţ, L., and Clemens E., 2011. "Object Representations at Multiple Scales from Digital Elevation Models." Geomorphology. 129(3–4), 183–89. https://doi.org/10.1016/j.geomorph.2011.03.003.
[6] Grohmann, C. H., 2015. "Effects of Spatial Resolution on Slope and Aspect Derivation for Regional-Scale Analysis." Computers & Geosciences. 77, 111–17. https://doi.org/10.1016/j.cageo.2015.02.003
[7] Shih, P. T., 2014. "Evaluating Information Loss of SRTM DEM Data with Different Grid Sizes." Journal of Surveying Engineering. 140(4): 04014010. doi:10.1061/(ASCE)SU.1943-5428.0000132.
[8] Prasicek, G., J. Otto, D. R. Montgomery, and L. Schrott, 2014. "Multi-Scale Curvature for Automated Identification of Glaciated Mountain Landscapes." Geomorphology. 209, 53–65. https://doi.org/10.1016/j.geomorph.2013.11.026.
[9] James, D. E., M. D. Tomer, and S. A. Porter. 2014. "Trans-Scalar Landform Segmentation from High-Resolution Digital Elevation Models." Poster presented at the ESRI Annual Users Conference, San Diego, CA, July.
[10] Wilson, J. P., and J. C. Gallant, 2000. "Terrain Analysis: Principles and Applications". Wiley, 479 p.
[11] Martínez-Llario, José C., Sergio Baselga, and Eloína Coll. 2021. "Accurate Algorithms for Spatial Operations on the Spheroid in a Spatial Database Management System" Applied Sciences 11, no. 11: 5129. https://doi.org/10.3390/app11115129

[12] Kennedy, M., and S. Kopp, 2000. “Understanding Map Projections”. Redlands, CA: ESRI, Environmental Systems Research Institute.
[13] Burrough, P.A., and R. A. McDonell, 1998. “Principles of Geographical Information Systems”. Oxford University Press, 199 p.
[14] Esri, 1992. “Slope” in ARC/INFO GRID Command References. Environmental Systems Research Institute, Inc.
[15] Zevenbergen, L. W., and C. R. Thorne, 1987. “Quantitative Analysis of Land Surface Topography.” Earth Surface Processes and Landforms. 12(1), 47–56. doi:10.1002/esp.3290120107.
[16] Schmidt, J., I. S. Evans, and J. Brinkmann, 2003. “Comparison of Polynomial Models for Land Surface Curvature Calculation.” International Journal of Geographical Information Science. 17(8), 797–814. doi:10.1080/13658810310001596058.
[17] Evans, I.S., 1980. An integrated system of terrain analysis and slope mapping. Zeitschrift für Geomorphologie. N.F. Suppl.-Bd. 36: 274-295.
[18] Minár, J., I. S. Evans, and M. Jenčo, 2020. “A Comprehensive System of Definitions of Land Surface (Topographic) Curvatures, with Implications for Their Application in Geoscience Modelling and Prediction.” Earth-Science Reviews. 211, 103414. https://doi.org/10.1016/j.earscirev.2020.103414.
[19] Krebs, P., M. Stocker, G. B. Pezzatti, and M. Conedera, 2015. “An Alternative Approach to Transverse and Profile Terrain Curvature.” International Journal of Geographical Information Science. 29(4), 643–66. https://doi.org/10.1080/13658816.2014.995102.