Overview of the geo Framework

Seth Goodman
Ariel BenYishay
Dan Runfola

AidData

Institute for the Theory and Practice of International Relations

The College of William and Mary

sgoodman@aiddata.org
danr@wm.edu
abenyishay@aiddata.org

geo.aiddata.org
geoquery.org

Summary

The geo framework is an open source, spatial data management platform that aims to remove the barriers to incorporating spatial data into research in any discipline. Researchers have access to spatial data in many different formats and from many different sources. Access to spatial data has been enabled by fleets of satellites that image the entire Earth at high resolution multiple times per day, platforms which geocode news from around the world, geo-referenced census and survey instruments, and billions of GPS enabled consumer devices. Researchers looking to utilize these resources face challenges in acquiring or identifying the expertise to find, manage, and integrate large quantities of data from numerous sources.

geo framework

The geo framework underpins a number of modules which enable individuals with limited expertise in the use of spatial data to retrieve such data in easy-to-use formats. By handling the complexities of working with spatial data, it allows users to focus on using the data output, rather than producing it. It is designed to work in cluster computing environments and can handle requests of hundreds of thousands of units of observation - making data accessible to users in a way that it has never been before.

geo(query)

Custom data requests into the *geo* framework can be made via a simplified online interface, *geo(query)*. The open source *geo(query)* module provides access to a curated subset of spatial data. These datasets are provisioned by spatial data specialists, who are responsible for finding, downloading and preparing datasets so that they can be ingested into the framework - a process frequently done in consultation with the practitioners or researchers that produce the data. These specialists further record extensive metadata, as well as make all scripts used and steps taken during data processing publicly available. Once a request is made and processed, results and documentation are sent via email that include permanent download links so that users can always find the exact data used for a specific project - facilitating the replication of research findings.

Table of Contents

Acknowledgements

This work was made possible by the support of USAID, KFW, Humanity United, the World Bank, the Global Environmental Facility, the MacArthur Foundation, and the College of William and Mary. We would also like to thank the many team members who contributed to this project, including Ben Dykstra and the AidData Data Team (Zhonghui Lv, Lauren Harrison, Alex Kappel, Sid Ghose, Brooke Russell). This work was performed in part using computational facilities at the College of William and Mary which were provided with the assistance of the National Science Foundation, the Virginia Port Authority, Virginia's Commonwealth Technology Research Fund and the Office of Naval Research.

About AidData

AidData is a research and innovation lab located at the College of William & Mary that seeks to make development finance more transparent, accountable, and effective. Users can track over $40 trillion in funding for development including remittances, foreign direct investment, aid, and most recently US private foundation flows all on a publicly accessible data portal on AidData.org. AidDta's work is made possible through funding from and partnerships with USAID, the World Bank, the Asian Development Bank, the African Development Bank, the Islamic Development Bank, the Open Aid Partnership, DFATD, the Hewlett Foundation, the Gates Foundation, Humanity United, and 20+ finance and planning ministries in Asia, Africa, and Latin America.

Recommended Citation

Goodman, S., BenYishay, A., Runfola, D., 2017. *Overview of the geo Framework*. AidData. Available online at http://geo.aiddata.org/. DOI: 10.13140/RG.2.2.28363.59686

Introducing the *geo* Framework

Extracting value from existing spatial information can be extremely challenging for non-expert users. Programs such as ArcGIS and Q can take years to master, datasets can be large and unwieldy, and spatial weighting and corrections can be difficult to understand and implement. Historically, the human and computational costs associated with obtaining relevant spatial data - e.g. the intensity of nighttime lights, the amount of tree cover, or the distance from a major road - at meaningful units of analysis, such as census tracts or village administrative boundaries, has prevented spatial analyses in a wide range of fields.

Some recent initiatives have simplified access to the rapidly growing array of rich spatial data for non-experts. Of particular note are the efforts of PRIO grid, TerraPop, growUP, and a number of NASA products. These tools engage slightly different user groups, ranging from academic researchers to practitioners. The (geo) framework adds to this ecosystem by providing a scaleable, highly parallelized computational framework designed to enable non-experts to quickly extract massive amounts of spatial information at fully customizable geographic units. Where previous tools have relied heavily on preprocessing, revolved heavily around a single data source, or pre-determined the geographic boundaries a user can utilize, the geo framework allows all of these factors to be dynamic. Through (geo)query, an online interface enabling access into the geo framework's store of data, we provide a flexible, expert-curated solution which enables easier access to a wider set of spatial information than has been available to date.

This document provides users of the (geo) framework - or related tools such as (geo)query - with information on the technical steps taken to process and integrate data sources. It is aimed at users with little technical background in the use of spatial data, and seeks to introduce both the challenges and solutions (geo) provides. We further provide detailed information on the procedure used to integrate international aid information published by AidData, which represents a novel source of geographic data available within this tool.

Technical Overview: The *geo* Framework

A key component of making spatial data easily accessible to users is providing a simple frame of reference for all spatial data, regardless of format or origin, that can also be utilized in non spatial applications. However, different data types, projections, and many other issues can challenge even experts' ability to integrate across different spatial data types. The core of the *geo* framework is built on the concept of making this process easier by leveraging expert curation and large, scalable computation to integrate different types of spatial data, including the commonly used raster and vector based representations, into a single data framework.

Introduction to Spatial Data Formats & Data Extraction

Vector data formats, such as ESRI's shapefile[1] and the open GeoJSON[2] standard, are generally used to describe discrete spatial features including points, polygons and lines. Examples of spatial features defined by vector data are administrative boundaries, buildings and roads, and points where measurements were taken. Rasters are a way of representing a grid of spatial data by georeferencing a set of pixels. Each pixel in a raster represents a value at a specific location defined by the resolution of the raster and how it was georeferenced. The raster format is most commonly associated with satellite imagery but can also be used to display surfaces produced by models or sets of point data such as from ground sensors (e.g., weather stations). Similar to vector data, there are many different standards for raster data such as the commonly used GeoTiff[3] and or ESRI ASCII Grid[4]. Most vector and raster data utilize formats that are incorporated in the Geospatial Data Abstraction Library (GDAL)[5], which is an open source programming library used by many Geographic Information System [GIS] applications to read and write vector and raster data from different formats. Figure 1 contains an example of each type of data.

[1] https://www.esri.com/library/whitepapers/pdfs/shapefile.pdf
[2] RFC 7946 GeoJSON (https://tools.ietf.org/html/rfc7946)
[3] https://trac.osgeo.org/geotiff/
[4] http://resources.esri.com/help/9.3/arcgisdesktop/com/gp_toolref/spatial_analyst_tools/esri_ascii_raster_format.htm
[5] http://www.gdal.org/

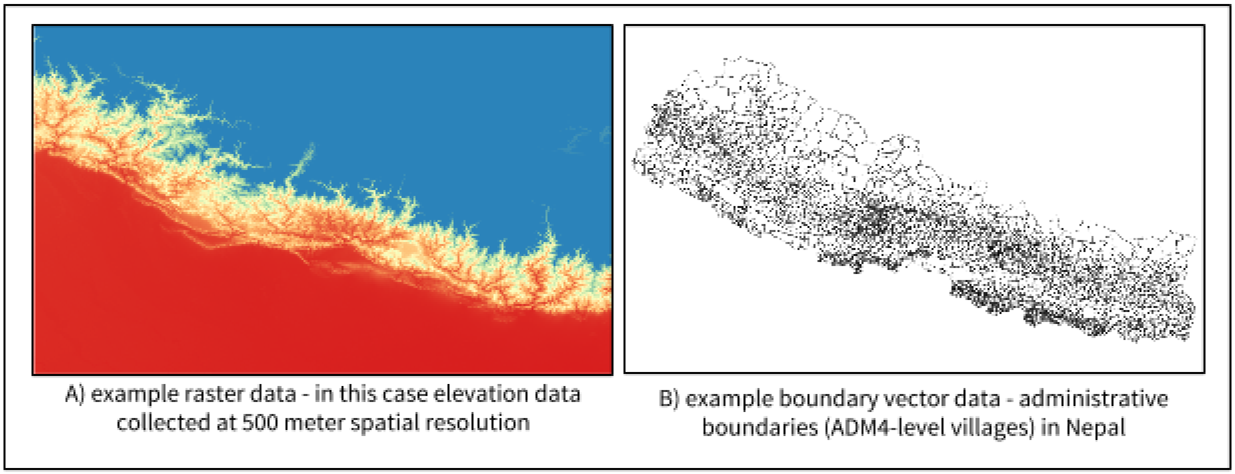

A) example raster data - in this case elevation data collected at 500 meter spatial resolution
B) example boundary vector data - administrative boundaries (ADM4-level villages) in Nepal

Figure 1

Research with spatial data is frequently conducted using units of analysis which are defined by boundaries (vector data features) such as administrative zones or grids. In software or tools designed to use spatial information (i.e., GIS platforms such as ESRI's ArcMAP and Quantum GIS), information about measurements taken within a set of boundaries are generally saved as an attribute, or property, of each unit (i.e., the total population that falls within a geographic boundary would be an attribute of each spatial unit of observation, such as a district). These properties can be exported in familiar nonspatial formats such as a comma separated value (CSV) file, usable with a wide variety of applications (e.g., Excel, R, Stata, Python).

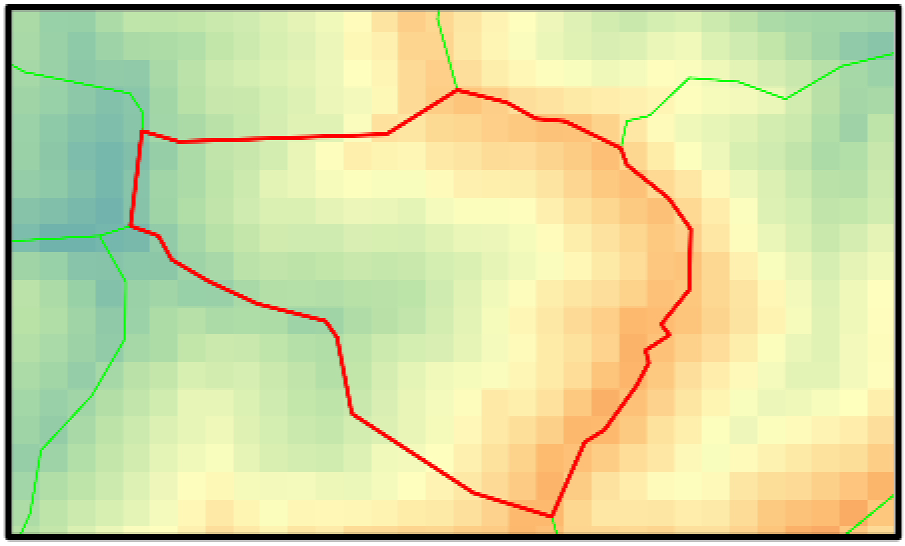


Figure 2. The village of Basa in Nepal covers approximately 185 cells of the elevation raster. Elevation values in Basa range from around 1200 meters above sea level, represented by the red-orange pixels on the right side of the village, to over 3000 meters on the left side in the blue-green cells.

Table 1. Extract Results	
Name	Basa
Count	185
Sum	393,202
Mean	2125.416

The most fundamental module in the *geo* framework is the data extraction module which is responsible for calculating the values of source data within arbitrary sets of boundaries. A key method used in this module involves extracting data from rasters to units of analysis (Figure 2; Table 1). This extract process for rasters is often referred to as "zonal statistics" in GIS tools and software[6], and is a common method of summarizing information from raster data sources within an area defined by vector features, such as administrative units.

[6] ArcGIS Zonal Statistics http://desktop.arcgis.com/en/arcmap/10.3/tools/spatial-analyst-toolbox/zonal-statistics.htm

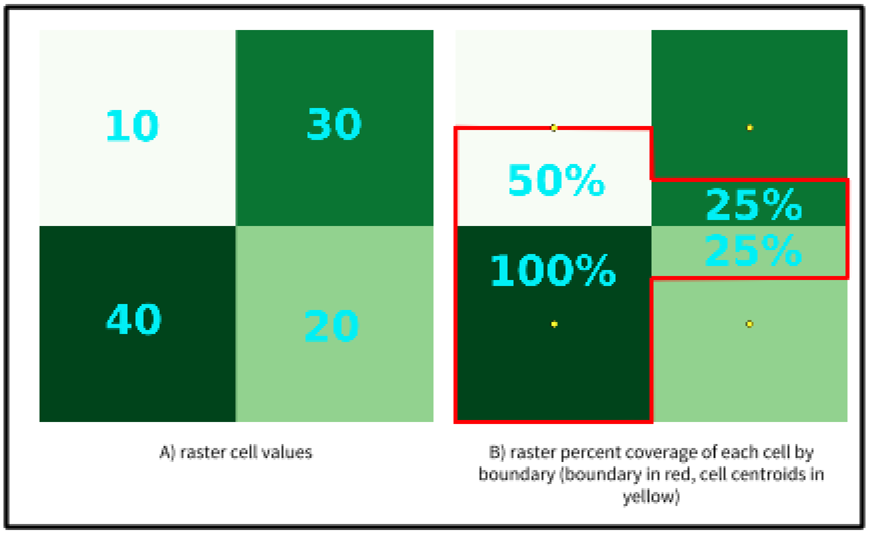
Raster Extract Methods

The *geo* framework performs raster extracts with methods similar to many other zonal statistics implementation, using a modified version of the Python package *rasterstats* (Perry 2016). Generating statistics for a boundary requires determining which cells in the raster are associated with the boundary, as well as the degree of overlap between each geographic boundary and raster cell. Selecting the relevant cells is accomplished by first rasterizing the boundary using the affine transformation (i.e., resolution and cell alignment) of the raster data. The rasterized boundary can then be applied as a mask to the raster data to select only the relevant cells whose values can then be passed along to a statistical function (mean, max, etc.)[7]. Although other implementations of zonal statistics use the same fundamental concept, improvements have been made to the methods used within the geo framework to increase the accuracy of the raster extract process.

Accounting for edge-case cells with partial coverage

During the raster extract process, cells along the exterior edge of a boundary are often only partially covered, such as in Figure 3b. When rasterizing a boundary that has partial cell coverage, it is common for zonal statistics methods[8] to either include or exclude an entire cell based on whether the centroid of the cell is covered by the boundary. Some tools provide alternatives to the centroid method which enable users to always include cells along the exterior of a boundary (Perry 2016) or to weight cells based on the actual coverage (Hijman 2015).

Figure 3 shows an example raster dataset, in this case representing estimates of the temperature within each cell[9]. Given this raster (Figure 3a) and a geographic boundary (red outline in Figure 3b), a common goal is to determine the mean value of raster cells associated with the boundary - i.e., our best estimate of the true temperature for the geographic area outlined in red.



A) raster cell values

B) raster percent coverage of each cell by boundary (boundary in red, cell centroids in yellow)

Figure 3

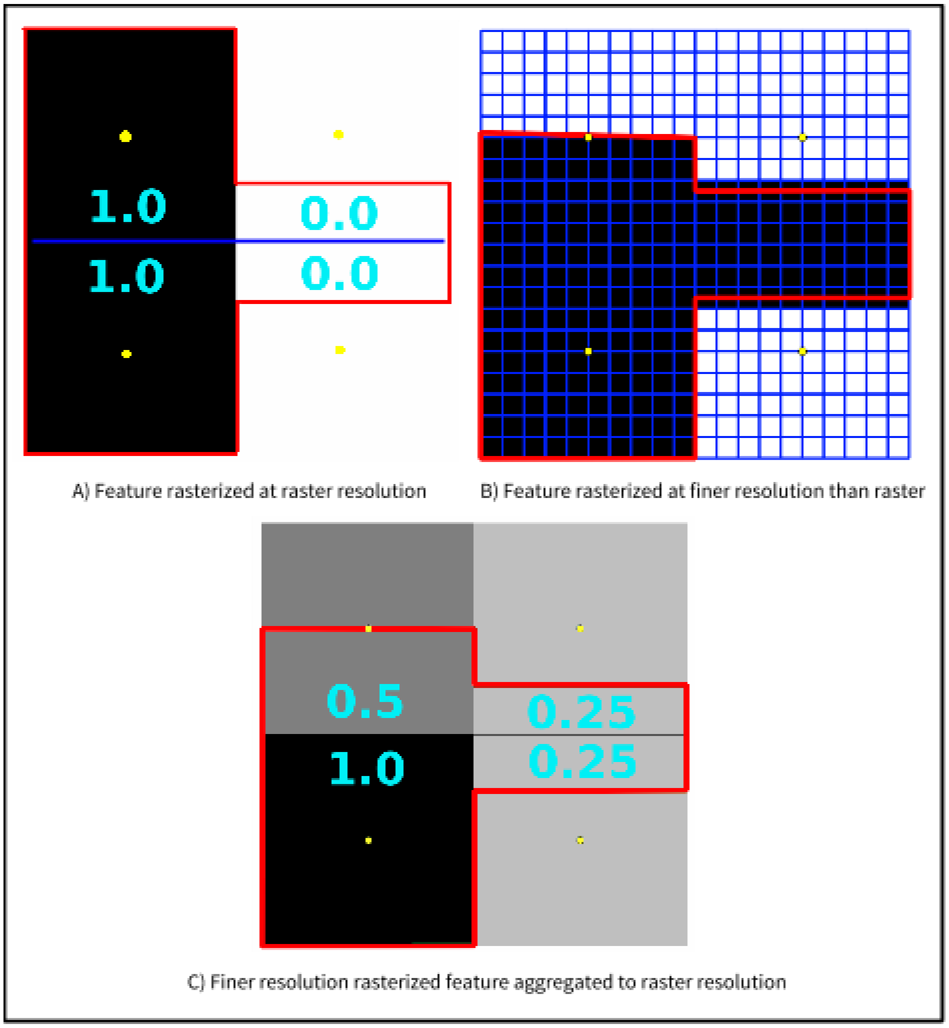
Using the centroid based approach to rasterize the boundary outlined in red, the two left cells of the raster would be averaged (in their entirety) and the two right cells would be ignored (Figure 4a). To produce a more accurate estimate of this area's mean temperature, the *geo* framework leverages an alternative approach that estimates the percentage of overlap between each raster cell and the geographic boundary. To avoid the increased computational complexity of determining the exact overlap (i.e., running an intersection of each raster cell's bounding box with the geographic boundary geometry), we instead estimate the overlap by exploiting the existing behavior of the boundary rasterization process.

[7] ArcGIS (http://desktop.arcgis.com/en/arcmap/10.3/tools/spatial-analyst-toolbox/h-how-zonal-statistics-works.htm)

[8] ArcGIS, Quantum GIS

[9] The data in this example is NOT real and has been created purely to illustrate the methods being discussed.

The existing centroid based rasterization can be utilized to produce coverage estimated by applying the boundary rasterization at a finer resolution than the actual resolution of the raster data being extracted[10]. The cells in this fine resolution rasterization can then be aggregated back to the original resolution to produce a coverage estimate. Figure 4 and table 2 provide an example which compares ignoring percent coverage and incorporating it; in this example, the centroid based zonal statistics method produces an estimated mean of 25, while the weighted method based on the percent coverage of cells estimates a mean of 28.75, or approximately a 13% difference.



A) Feature rasterized at raster resolution

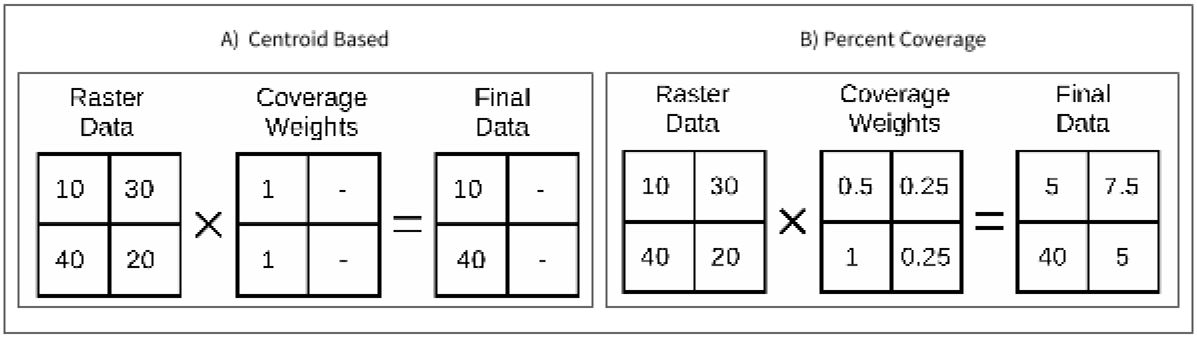
B) Feature rasterized at finer resolution than raster

C) Finer resolution rasterized feature aggregated to raster resolution

Figure 4

[10] The resolution of the boundary rasterization (i.e., the finer scale grid which is overlaid) is relative to the original raster resolution based on a scaling factor (e.g., a scale of 10 equates to the boundary being rasterized at 1 order of magnitude finer resolution - a single cell becomes a 10x10 grid of cells). Determining an optimal scaling factor involves balancing accuracy with computational processing time and memory usage. Simulations suggest a scaling factor of 10 will usually result in estimates of overlap within reasonably low error (<10%) compared to the actual overlap, with minimal computational/memory cost increases. Larger scaling factors, such as two orders of magnitude finer resolution or more, can result more accurate (<5% error) coverage estimates but are slower and creates a larger memory footprint (when working with large boundaries and extremely high resolution rasters, memory can be a significant bottleneck, even on systems with unusually large amounts of memory). As a tradeoff between computational efficiency and accuracy, the geo framework defaults to a resolution 1 order of magnitude finer than the source raster, but allows for the scaling parameter to be set according to the precisional needs of individual research projects.

Table 2. Percent coverage comparison calculations



A) Centroid Based

Raster Data × Coverage Weights = Final Data

$$\begin{bmatrix}10 & 30\\40 & 20\end{bmatrix}\times\begin{bmatrix}1 & -\\1 & -\end{bmatrix}=\begin{bmatrix}10 & -\\40 & -\end{bmatrix}$$

B) Percent Coverage

Raster Data × Coverage Weights = Final Data

$$\begin{bmatrix}10 & 30\\40 & 20\end{bmatrix}\times\begin{bmatrix}0.5 & 0.25\\1 & 0.25\end{bmatrix}=\begin{bmatrix}5 & 7.5\\40 & 5\end{bmatrix}$$

$$\text{Mean} = \frac{10+40}{1+1} = 25.00 \qquad \text{Mean} = \frac{5+7.5+40+5}{0.5+0.25+1+0.25} = 28.75$$

Accounting for actual area represented by cell

Because data collected over space is generally represented on a 2D plane (i.e., a satellite image), and the earth is a 3D sphere, there are issues that must be accounted for to ensure that measurements are correctly ascribed to areas of interest. In particular, when extracting data from an area represented by raster cells, it is important to consider the physical area represented by each cell (and, how these areas may vary across cells). To establish this, information on both the coordinate reference system (CRS) and projection used for the data must be known. Most widely used and publicly available global datasets utilize CRS EPSG:4326, commonly referred to as WGS84. Unlike equal area projections (such as Lambert Azimuthal Equal-Area projection with CRS EPSG:3408), which maintain a constant cell area across latitudes, the area represented by cells in a WGS84 dataset is dependent on latitude. In WGS84, as cell observations approach the poles, lines of longitude converge and cell area decreases (Figure5).

To accurately account for area when performing analyses using WGS84 datasets, there are two possible methods: reprojecting the datasets to an equal area projection, or weighting cells based on latitude during the extract process. The most commonly employed approach to this procedure is reprojection of both the source (raster) and boundary (vector) datasets to equal area projections. However, in order to reproject a raster dataset each cell must be treated as a point (or a finite set of points), generally based on the centroid of the cell in existing implementations. Because of this, during the process of raster reprojection it is necessary to resample the underlying data in order to rebuild the surface using the new projection. Resampling can introduce changes into the data depending on the data type, resampling method, and raster resolution.

Because of this limitation, we instead (a) reproject boundary information to the same projection as the source raster (a process that can be done with perfect accuracy (Kugler 2015)), and (b) weight cells based on latitude during the data extraction process. After the boundary is reprojected, weights for the underlying raster data are generated for each row of pixels, utilizing the Haversine distance formula to account for variable size of raster cells as distance from the equator increases (Figure 5). This approach incurs minimal computational costs during the extract process and requires no additional preprocessing or management of the data outside of the extract process.[11]

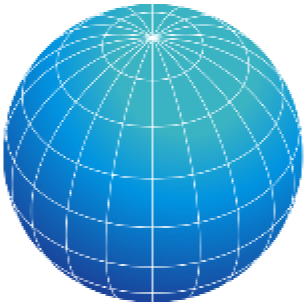


Figure 5. Lines of longitude converging at poles

Using the same example data from Figure 3, we assume that the underlying raster that is measuring temperature has a (extremely coarse) resolution of 40 decimal degrees, and the top left corner is located at a latitude of 80 degrees (projected in WGS84). Using the latitude at the center of each row of cells (60 degrees for the top row of raster, 20 degrees for the bottom row of this example raster) the ratio of spherical distances calculated using the haversine formula were 0.5 for the top row and 0.766 for the bottom row.

[11] Distance between lines of latitude remains constant, so a measurement of longitudinal distance can be used to weight each cell rather than the actual cell area (i.e., latitude distance x longitude distance)

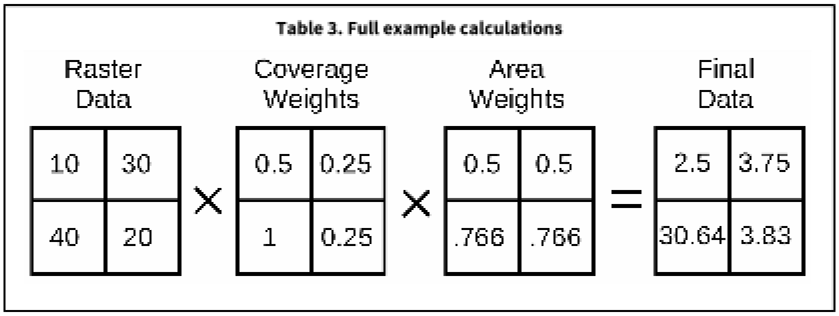


Table 3. Full example calculations

$$\begin{array}{c}\text{Raster Data}\\ \begin{array}{|c|c|}\hline 10 & 30\\ \hline 40 & 20\\ \hline\end{array}\end{array} \times \begin{array}{c}\text{Coverage Weights}\\ \begin{array}{|c|c|}\hline 0.5 & 0.25\\ \hline 1 & 0.25\\ \hline\end{array}\end{array} \times \begin{array}{c}\text{Area Weights}\\ \begin{array}{|c|c|}\hline 0.5 & 0.5\\ \hline .766 & .766\\ \hline\end{array}\end{array} = \begin{array}{c}\text{Final Data}\\ \begin{array}{|c|c|}\hline 2.5 & 3.75\\ \hline 30.64 & 3.83\\ \hline\end{array}\end{array}$$

$$\text{Mean} = \frac{2.5 + 3.75 + 30.64 + 3.83}{0.5\times0.5 + 0.25\times0.5 + 1\times0.766 + 0.25\times0.766} = 30.56$$

The final calculation takes into account the underlying values of the raster data, the overlap between the boundary and rasters on edge cases ("coverage weights"), and the weighting to account for the relative areas of each cell ("area weights"). This results in a final set of data (Table 3) which is used to identify the best approximated weighted mean for a given geometry - in this case, 30.56. In this example, the raster resolution was exaggerated to illustrate how the cell area weighting based on latitude functions with a simplified (four-cell) raster. However, when applied to actual extractions with finer resolution and less variation in latitude it is still an important consideration and can noticeably change the results if not accounted for.

Incorporating Geocoded Tabular Information (AidData)

Along with data from third party sources, the geo framework and geo(query) module also provide a new way of accessing AidData's geocoded research releases on development finance and international aid. By utilizing new methods of transforming tabular geocoded aid data into raster surfaces, the *geo framework* produces more accurate representations of information on where aid is allocated than existing best practices, as well as providing researchers with the ability to incorporate uncertainty metrics into analyses using aid data.

AidData's methodology[12] for georeferencing or geocoding development projects was originally developed in 2010 by AidData in partnership with the Uppsala Conflict Data Program (UCDP)[13]. The methodology was later revised by IATI[14] and eventually adopted as a global reporting standard. Following the geocoding methodology, teams of geocoders use project information to identify where projects took place and assign the most accurate location possible, based primarily on Geonames[15]. For each location, coordinates are assigned along with codes signifying how well the coder was able to match a Geonames location and the type of location matched (e.g., city, second order administrative zone, hospital, etc.).

Describing spatial features, which are often areas and not points, with a point based precision (or similar) coding system has inherent limitations that can make spatial analysis difficult and inaccurate. By defining rules which allow points to be converted to geometry based on the coding system used, it is possible to generate rasters which represent a distribution of aid across the geometry where it was allocated (as opposed to a point-based representation). These spatial representations (generalized using raster format data) make it possible to produce statistics for areas represented by arbitrary user defined boundaries, resulting in aid estimates reflective of aid in a region based on the geometries to which it is known aid was allocated. This represents an improvement on the currently, commonly-employed point-in-polygon aggregation (i.e., spatial join) using point-based representations of aid flows. In addition to traditional statistics (e.g., sum of aid in an area), the process of extracting data from aid rasters makes it possible to produce new measures of spatial uncertainty about aid estimates for use in models.

Procedures for Integrating Tabular Information (AidData)

The value of, and need for, aid rasters can be explained and visualized using a simple example of an individual location associated with a project in Nepal[16] which has a geocoded latitude and longitude provided. Aid information for this project

[12] AidData Geocoding Methodology (http://aiddata.org/sites/default/files/ucdp_aiddata_codebook_published.pdf/0

[13] UCDP (http://www.pcr.uu.se/research/UCDP/)

[14] IATI (http://www.aidtransparency.net/)

[15] Geonames(http://www.geonames.org/)

[16] AidData Nepal Geocoded Research Release (NepalAMP_GeocodedResearchRelease_Level1_v1.3

location will be aggregated to a sub-district unit of analysis using both a traditional point-in-polygon style spatial join[17] and the method employed by the *geo framework* in order to compare the two methods.

Relevant information about the hypothetical project location which is used in this example can be found in Table 4. Figure 6 shows the location of the project location, as well as the ADM4 units proximate to it (outlined in yellow). In this hypothetical case, a practitioner seeks to identify the total amount of aid being sent to each 4th order administrative boundary, and is provided with information from Table 4 for the example project location. This is a very common use-case for information which is geocoded from text-based source documents (i.e., news reports, or in this case project PDFs). Currently, the most common method used for aggregating data is to first drop out extremely coarse data (i.e., all locations that are not at least as precise as the unit of analysis in order to avoid concerns of spatial ambiguity), and then conduct a point-based spatial join (or other spatial-analytic technique) to the units of analysis a researcher is interested in. Recent examples of this include Buchanan 2016[18], Runfola 2016, Odokonyero 2015[19], Briggs 2015[20], Dreher 2015[21], Weezel 2015[22] and Masaki 2015[23]. Using this method introduces some significant issues. First and foremost, there is significant bias in using point-based spatial joins when points are truly representative of geographic boundaries, an example of which is provided in Figures 6-9. Additionally, in the case of international aid, by dropping out projects which do not have exact geographic information it is possible systematic biases will be introduced into analyses, i.e., projects with poor documentation could be systematically implemented in different ways than those with rich documentation along meaningful dimensions.

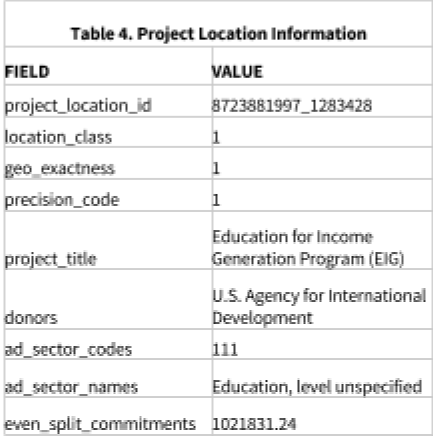


Table 4. Project Location Information

FIELD	VALUE
project_location_id	8723881997_1283428
location_class	1
geo_exactness	1
precision_code	1
project_title	Education for Income Generation Program (EIG)
donors	U.S. Agency for International Development
ad_sector_codes	111
ad_sector_names	Education, level unspecified
even_split_commitments	1021831.24

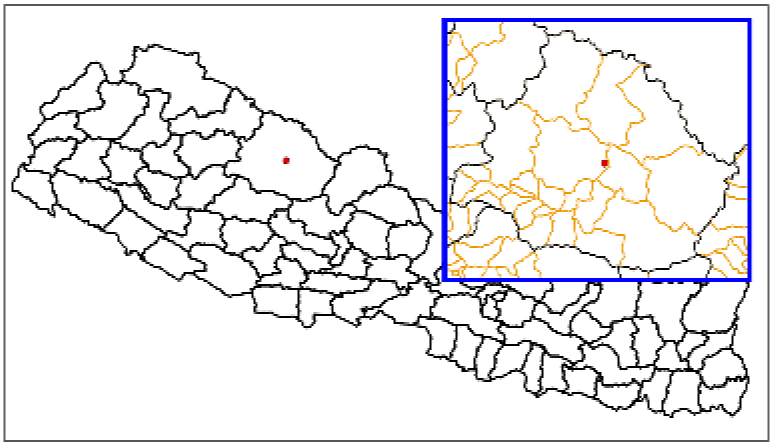


Figure 6. Project location in Nepal with ADM3 boundaries (ADM4 boundaries inset)

The technical process followed by a spatial join procedures is to iterate over all project locations and perform a spatial check to find the geographic boundary (i.e., unit of observation) they are located within. The aid associated with a project location is then added to the aid attribute for that unit. Figure 7 represents aid aggregated to the ADM4 level using a spatial join, following the example data in Table 4. All of the aid was assigned to the ADM4 unit the point is within, which is shown as dark green. These steps represent an example of the currently employed best practices.

Within the geo framework, we leverage additional information on the geographic boundaries associated with each point. Based on our project location information in Table 4, there is also additional information on the precision of the geographic boundary - i.e., this point was geocoded with a precision code of 1. In the case of AidData, this information allows us to ascribe a 25 kilometer buffer - i.e., we know the aid went somewhere within a 25km area around the point, but the exact latitude and longitude may not be meaningful. Following this, the case seen in Figure 7 which assigns all aid to a single ADM4 unit because geocoded coordinates lie within it, is erroneous, and can lead to misattribution and further modeling errors.

[17] http://www.qgistutorials.com/en/docs/performing_spatial_joins.html
[18] AidData Working Paper: wps20_world_bank_biodiversity
[19] AidData Working Paper: wps18_sub-national_perspectives_on_aid_effectiveness
[20] AidData Working Paper: wps13_does_aid_target_the_poorest
[21] AidData Working Paper: wps9_aid_and_growth_at_the_regional_level
[22] AidData Working Paper: wps8_a_spatial_analysis_of_the_effect_of_foreign_aid_in_conflict_areas
[23] AidData Working Paper: wps5_the_political_economy_of_aid_allocation_in_africa

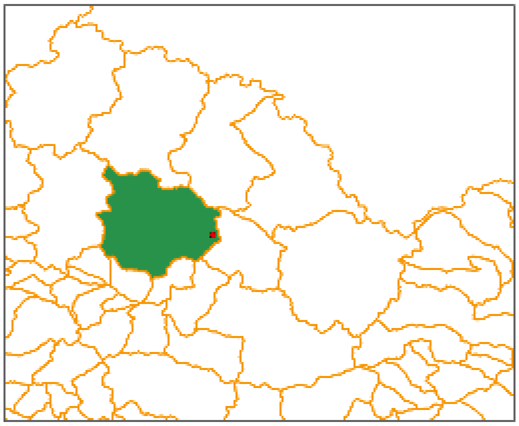
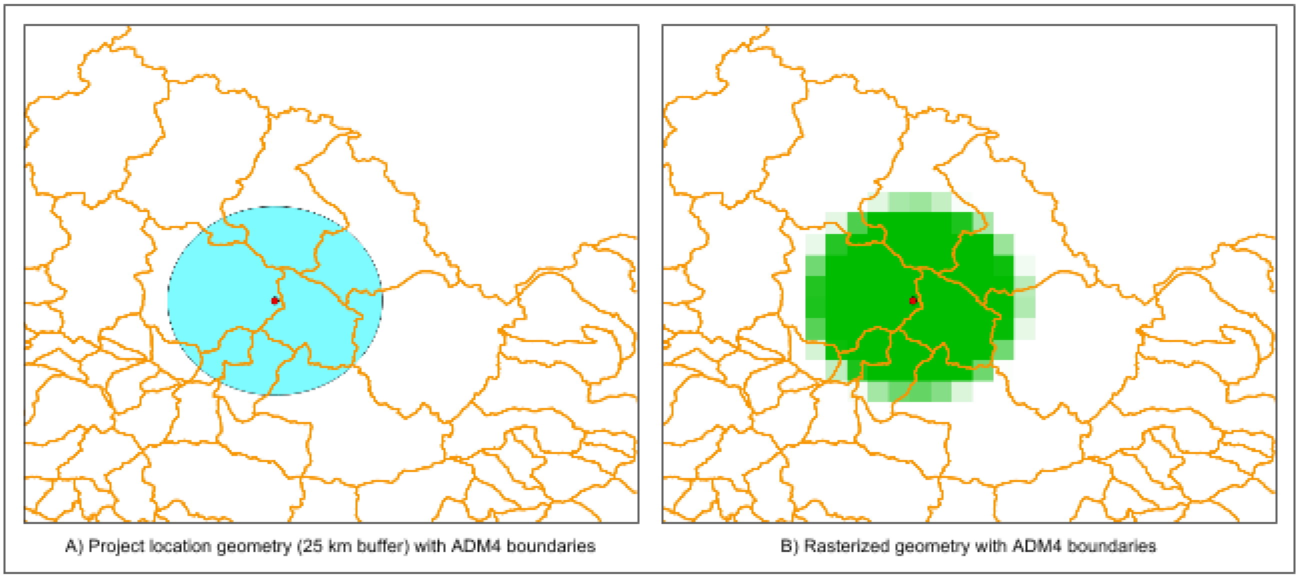


Figure 7. Point-in-polygon extract to ADM4

Instead, we represent aid project locations with their geometries - in this example, using a 25 km buffer around the coordinates - i.e., the area we know the aid went within, rather than the exact location. The buffer for our project location at least partially covers 10 ADM4 units. As a baseline "no information" case, *geo(query)* defaults to evenly split aid (a) equally across the buffered area, (b) across all locations associated with a project, and (c) across sectors (in the event that multiple sectors are associated with a project). This assumption - the "equal split" case - can be modified by researchers that are interested in exchanging additional assumptions for more precision (i.e., a practitioner may want to assume that aid is more likely to be allocated to areas with higher temperatures, but the *geo(query)* system defaults to the no assumption case). With this spatial surface, we can extract a summary of the raster cells within each boundary of interest. This process is summarized in Figures 8 and 9.



A) Project location geometry (25 km buffer) with ADM4 boundaries

B) Rasterized geometry with ADM4 boundaries

Figure 8

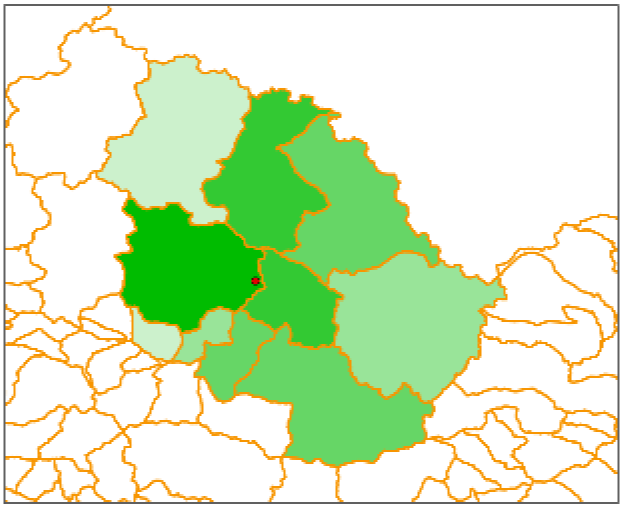


Figure 5. Raster extract to ADM4

As Figure 9 shows, by following this procedure the aid is now spread across the ADM4 units proximate to the geocoded location, based on their relative overlap with the buffered region. A comparison of the current best practice (spatial joins) and geo framework's boundary-based extraction results can be seen in Figure 10. The data that would result from these two approaches can be seen in Table 5. The results for the geo framework extract also provide a "reliability" column, which is a metric of spatial certainty derived based on the geographic boundaries aid was allocated to. More information on how to use this reliability information in modeling efforts can be found in a follow-up to this document, Modeling Spatial Imprecision with geoSIMEX, available online via geo.aiddata.org.

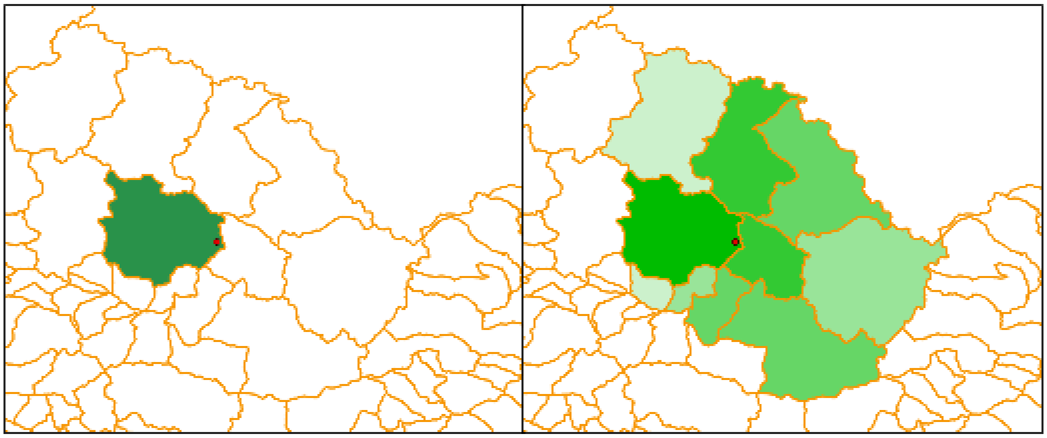


Figure 10. Comparison of spatial join (left) and raster extract (right)

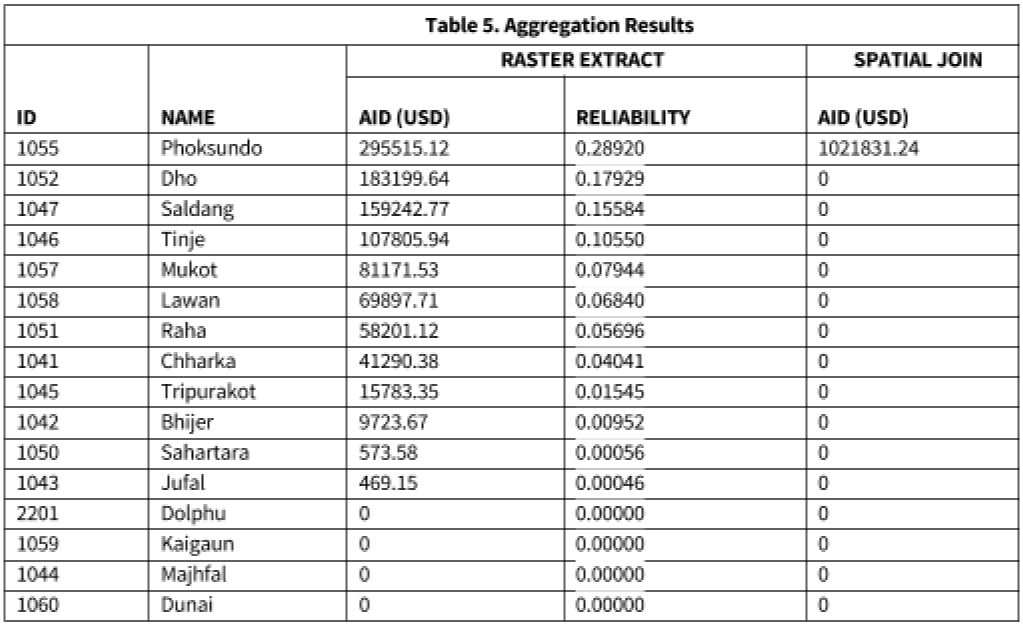


Table 5. Aggregation Results

		RASTER EXTRACT		SPATIAL JOIN
ID	NAME	AID (USD)	RELIABILITY	AID (USD)
1055	Phoksundo	295515.12	0.28920	1021831.24
1052	Dho	183199.64	0.17929	0
1047	Saldang	159242.77	0.15584	0
1046	Tinje	107805.94	0.10550	0
1057	Mukot	81171.53	0.07944	0
1058	Lawan	69897.71	0.06840	0
1051	Raha	58201.12	0.05696	0
1041	Chharka	41290.38	0.04041	0
1045	Tripurakot	15783.35	0.01545	0
1042	Bhijer	9723.67	0.00952	0
1050	Sahartara	573.58	0.00056	0
1043	Jufal	469.15	0.00046	0
2201	Dolphu	0	0.00000	0
1059	Kaigaun	0	0.00000	0
1044	Majhfal	0	0.00000	0
1060	Dunai	0	0.00000	0

Conclusion and Further Resources

In this introduction of the AidData geo framework, we outline the underlying theories we leverage to support large-scale spatial data extractions based on wide sets of geospatial data. Further, we provide insight into how we integrate tabular-based geocoded information from the AidData database with these sources. We argue that by coupling a framework which supports the processing of large sets of curated spatial information with an easy-to-use interface, called geo(query), users who are currently unable to use spatial data due to the significant associated costs will be enabled to do so. Further, by providing a framework which can provide more accurate, geometry-based representations of data which was geocoded from news articles or other textual sources, we seek to promote more accurate use of such information.

Further resources regarding the *geo framework* can be accessed via geo.aiddata.org.

References

AidData. 2016. NepalAIMS_GeocodedResearchRelease_Level1_v1.4.1 geocoded dataset. Williamsburg, VA and Washington, DC: AidData. http://aiddata.org/research-datasets.

Boshuizen, C., Mason, J., Klupar, P., & Spanhake, S. (2014). Results from the planet labs flock constellation. http://digitalcommons.usu.edu/cgi/viewcontent.cgi?article=3016&context=smallsat

GeoNames. GeoNames. http://geonames.org/. Retr. June 17, 2009.
QGIS Development Team, 2016. QGIS Geographic Information System. Open Source Geospatial Foundation Project. http://www.qgis.org/

Hijmans, R.J. (2015). raster: Geographic Data Analysis and Modeling. R package version 2.5-2. http://CRAN.R-project.org/package=raster

Kugler, T. A., Van Riper, D. C., Manson, S. M., Haynes, D. A., Donato, J., & Stinebaugh, K. (2015). Terra Populus: Workflows for Integrating and Harmonizing Geospatial Population and Environmental Data. Journal of Map and Geography Libraries, 11(2), 180-206. DOI: 10.1080/15420353.2015.1036484

Leetaru, K., & Schrodt, P. A. (2013, April). Gdelt: Global data on events, location, and tone, 1979–2012. In ISA Annual Convention (Vol. 2, No. 4).

Perry, M. (2016). rasterstats: Summarize geospatial raster datasets based on vector geometries. Python package version 0.10.3. https://pypi.python.org/pypi/rasterstats/0.10.3

Runfola and Napier. Migration, climate, and international aid: examining evidence of satellite, aid, and micro-census data. Migration and Development, 2016.

Terra Populus, Minnesota Population Center. http://www.terrapop.org. 2015.

Tollefsen, Andreas Forø, Karim Bahgat, Jonas Nordkvelle and Halvard Buhaug (2015). PRIO-GRID v.2.0 Codebook. Peace Research Institute Oslo.

Tollefsen, Andreas Forø; Håvard Strand & Halvard Buhaug (2012) PRIO-GRID: A unified spatial data structure. Journal of Peace Research, 49(2): 363-374. doi: 10.1177/0022343311431287

Zandbergen and S. J. Barbeau. Positional accuracy of assisted gps data from high-sensitivity gps-enabled mobile phones. The Journal of Navigation, 64:381–399, 2011.