**NASA DEVELOP National Program**



NASA Ames Research Center

*Fall 2015*

Lake Tahoe Water Resources

Creating a Global Continuous Detection Lake Level Monitoring Algorithm using Landsat Imagery

**Technical Report** 

Final Draft – November 19, 2015

Nolan Cate (Project Lead)

Anton Surunis

Chelsea Ackroyd

Dr. Brian Coltin, NASA Ames Research Center (Science Advisor)

Dr. Juan Torres-Pérez, Bay Area Environmental Research Institute (Science Advisor)

# I. Abstract

As global climate change continues to escalate and droughts become more frequent and severe, it becomes increasingly necessary to monitor and regulate available water resources. Lake Tahoe (CA/NV) is an important reservoir for tourism, local ecosystems, and drinking water. Its nearly 5 million annual visitors contribute at least $300 million to the local economy, making it one of California’s most popular attractions. Decreasing water levels are a concern for residents, the economy, and a number of endangered species that live on Lake Tahoe’s shores, such as the yellow cress. Current methods of monitoring lake levels, however, rely on depth gauges that require time-intensive fieldwork to retrieve data and are limited in their spatial coverage. Satellite imagery provides a far greater spatial extent while still providing regular measurements. Utilizing satellite imagery from the Landsat program, the Lake Level Automated Monitoring Algorithm (LLAMA) is a continuous detection lake level monitoring algorithm that uses a Modified Normalized Difference Water Index (MNDWI), thermal band analysis, and visible band reflectance values processed through Google’s cloud-based geospatial program, Earth Engine. In addition to a lake area measurement, LLAMA is able to show measurements of turbidity and algae levels over any given lake worldwide. Thus, LLAMA has the ability to provide water managers near real-time data regarding the turbidity, algae, and water levels for any lake or reservoir of sufficient size via Google Earth Engine.

**Keywords**

Remote Sensing, Landsat, Continuous Detection, Python, Lake Level, Google Earth Engine, Crisis Mapping Toolkit

# II. Introduction

**2.1 - Background**

Drought is one of the most common threats to the health and economy of populations worldwide as weather and climate patterns continually shift. In 1988, severe drought cost the United States economy nearly $40 billion (Kogan, 1997). According to the World Meteorological Organization of the United Nations, of the 2.8 billion people who encountered weather-related disasters from 1967 to 1991, approximately 50% of the individuals suffered due to drought. It is also known that drought can increase the risk of waterborne diseases and malnutrition, (Davies et al., 2014), and that aquifers that people rely on may lose the ability to recharge after a prolonged drought (Sampson et al., 2016). Thus, the consequences of drought are a concern for many as droughts worldwide have increased in their frequency, duration, and severity (Allen et al., 2009), particularly in the state of California, where water levels are exceptionally low at this time.

Lake Tahoe’s profitability, along with many other water bodies in California, is now in jeopardy as the state faces declining water levels. Lake Tahoe is an important water feature for it has an exceptional impact on local residents, economies, and ecological factors. Many residents rely on the lake for drinking water (North Tahoe Public Utility District, 2014), and it supports one of the state's largest tourism industries, drawing 23 million visitor-days per year (Kocher & Cobourn, 2007) and generating over $300 million in revenue annually. Furthermore, the lake also has a diverse ecosystem and is home to Tahoe Yellow Cress, an endemic species that has recently been considered endangered (Joey Keely, Personal Communication, 10/7/15). For these reasons, declining water levels are now a major concern for Tahoe’s local residents, economies, and endangered species; the need to consistently monitor lake levels is essential in preserving future water resources and protecting local counterparts.

**2.2 - Study Area**

Lake Tahoe (CA/NV) is the primary study area. This project focused on the ability to monitor the level of Lake Tahoe as well as other large lakes in the region, specifically Fallen Leaf Lake.

**2.3 - Project Partners**

The Lake Tahoe Basin Management Unit (LTBMU) of the USDA Forest Service was the primary end user of this project. As Lake Tahoe falls in their regional jurisdiction, the LTBMU is responsible for managing the water resources in the area, including relatively small water bodies such as Fallen Leaf Lake. The LTBMU does not currently have a consistent timeframe for monitoring lake level and, as a result, their measurements are sporadic, infrequent, and difficult to analyze (Joey Keely, Personal Communication). Both the USGS and the University of California Davis Tahoe Environmental Research Center (TERC) have *in situ* data for specific locations throughout Lake Tahoe, but the current methods have yet to assess the lake as a whole. As California’s water supplies are rapidly decreasing, the LTBMU has a vested interest in having access to near real-time monitoring of Tahoe Basin water levels in order to improve management of the remaining water.

**2.4 - Objectives**

This project developed the Lake Level Automated Monitoring Algorithm (LLAMA) tool for continually monitoring lake levels in near real-time using Landsat imagery. The Landsat program has provided consistent imagery acquisition at a 30-meter spatial resolution since 1984 and will continue to do so into the future. Each Landsat platform has a return interval of 16 days; this temporal resolution makes it a powerful tool for continuous change detection monitoring. With the use of Landsat imagery in conjunction with the cloud-based geospatial tool Google Earth Engine (See Section 2.5), LLAMA provides a means of lake level monitoring that automatically updates with the latest set of Landsat imagery. In addition to lake level monitoring, LLAMA will be able to provide simple estimates of water quality and algal blooms. The goal was to provide water managers with the ability to have accurate, continuous, and on-demand lake measurements for use in their decision-making processes.

The original lake measure algorithm, which LLAMA is based upon, was written by Dr. Brian Coltin of NASA Ames Research Center, and was calibrated to output only the lake level of Mono Lake via a series of Digital Number (DN) thresholds from a Landsat L1T product. The chief purpose of this project was to adjust the algorithm so that it can provide near real-time lake level monitoring for Lake Tahoe, CA, and be a useful tool for the LTBMU.

Given that the LTBMU is the end user of this project, the focus was to provide them with an effective tool that accurately monitors Lake Tahoe water levels. In addition to assisting the LTBMU, the ultimate goal was to create a tool that any water manager can use to monitor the level of a lake or other terrestrial water body of sufficient size anywhere on the globe.

**2.5 - Google Earth Engine**

Google Earth Engine (separate from Google Earth or Google Maps Engine) is a cloud-based geospatial software. Its purpose is to allow developers to use raster and vector data to conduct planetary scale geospatial analysis. Earth Engine also includes collections of satellite imagery. All Landsat imagery is included on Earth Engine and divided by sensor (i.e. TM, ETM+, and OLI are all different collections). As Landsats 7 and 8 continue to orbit and collect current data, new images are added to the collection that is stored on Google servers. After the lake area and water quality computations are completed, the user can then download the final output. Thus, Earth Engine offers various advantages for the user. For instance, Earth Engine does not require the user to download any satellite imagery during the computing process, which would otherwise occupy extensive storage space on the user’s computer. Instead, all of the land cover classification and geospatial analysis is completed in the cloud. Likewise, Earth Engine automatically updates the collections with new imagery, allowing the user to conduct continuous detection without needing to preprocess the imagery or use specialized imagery software.

**2.6 - Addressed NASA National Applications**

This work is categorized as NASA Applied Sciences Water Resources project, as it addresses water levels of Lake Tahoe and surrounding lakes. This will aid resource managers in the future, especially with the growing concern of managing water in drought conditions.

# III. Methodology

**3.1 Selecting a Water Body**

A LLAMA user must specify which water body it would like the algorithm to examine. LLAMA uses a polygon of the selected lake’s borders for the next step of analysis. The available polygons in LLAMA are stored in a Google Fusion Table, which is a table populated with attributes, including a water body’s name and location. Currently, LLAMA is capable of examining approximately 3,700 water bodies that are in the Fusion Table. More water bodies can be added to the Fusion Table as long as the user has a polygon of the desired study area. Instructions for adding to the Fusion Table can be found in the LLAMA tutorial (See Appendix 2). After loading the vector boundary, LLAMA applies a one-kilometer buffer to the polygon to ensure that a rising lake does not extend beyond the area of analysis. Each water detection step that is outlined in Section 3.2.2 is calculated within this polygon.

**3.2 Modifications to Existing Lake Level Measurement Algorithm**

**3.2.1 Google Earth Engine Landsat TOA Image Collections**

The original algorithm used Landsat L1T products, which come with pixel values in DN. Given the change in water body delineation method (Section 3.2.2), pixel values needed to be converted to reflectance rather than the scaled radiance that DN represents. Converting from the Landsat L1T DN values would add complexity and processing time to LLAMA. Rather than add a conversion step, LLAMA uses image collections from Google Earth Engine that come in values of Top of Atmosphere (TOA) reflectance. Google Earth Engine offers a completed collection of the Landsat Thematic Mapper (TM), as neither Landsat 4 nor 5 is currently in operation. However, the Landsat Operational Land Imager (OLI) collection is regularly updated as Landsat 8 acquires new imagery. Landsat 7’s Enhanced Thematic Mapper Plus (ETM+) was also considered; however, due to its scan line error starting in 2003, all data from ETM+ was disregarded for LLAMA’s purposes in terms of providing consistent measurements to the end user.

**3.2.2 Water Body Delineation**

LLAMA uses a four-step process to identify water. Any one of these steps may be able to independently classify a pixel as water, but each step would be more likely to encounter unique errors associated with that method of water detection. Using multiple steps, however, provides error checking for each stage of the process and decreases the likelihood that a pixel is incorrectly classified as water. The benefit of a four-step process is to cancel out potential misclassification errors by using a variety of water detection methods. Each pixel is assigned a score based on these four metrics, where water-covered pixels produce lower scores. Scores that fall below a given threshold are then classified as water (see Section 3.2.4). Each step in the scoring process contributes to a pixel’s water score. The four steps are as follows:

**Step 1 - MNDWI**

This step identifies water using the Modified Normalized Difference Water Index (MNDWI) outlined in Xu, 2006 (see Appendix 1). Contrary to the Normalized Difference Water Index (NDWI), which uses a green band and a near infrared band (NIR), the MNDWI uses a green band and the first shortwave infrared band (SWIR1) to assign a higher value to water and a lower value to vegetation than the NDWI (Xu, 2006). The resulting index image is rescaled using a linear rescale function in Earth Engine (see Appendix 1) so that the lowest value is 0.3 and the highest value is 0.8. A higher MNDWI value will have a greater impact on the water score, so a higher value means the pixel is more likely to be classified as water.

MNDWI can also be used to detect snow and is sometimes referred to as the Normalized Difference Snow Index (NDSI). This means that snow-covered pixels will be given a high value. This step alone would require a separate snow mask to ensure snow-covered pixels are not counted as water. Given that LLAMA employs a four-step method to detect water, this problem is resolved.

**Step 2 - Dark Objects**

Clear water is spectrally dark. To identify dark pixels, LLAMA takes the reflectance values of the NIR, SWIR1, and SWIR2 bands and sums them together into one single image. It then uses the same rescale function as found in step 1 to rescale the values and select the darkest pixels in the image. The values contribute to the water score, making darker pixels more likely to be classified as water.

**Step 3 - Blue Band Reflectance**

While water is spectrally dark, it tends to be brighter in the blue band than it is in the other spectral bands. LLAMA uses a Z-score to determine if the reflectance of a pixel in the blue band is significantly greater than the reflectance of a pixel in the green, red, NIR, SWIR1, and SWIR2 bands. If so, it will contribute to the water score and make the pixel more likely to be classified as water.

**Step 4 - Thermal Bands**

LLAMA examines the thermal bands of the Landsat instruments to determine if the temperature is above freezing. A pixel's brightness temperature contributes to its water score. If water is above freezing, the pixel is more likely to be classified as water.

**3.2.3 Cloud Masking**

While the water detection method in LLAMA is not likely to classify a cloud pixel as water, it does need to know how much of the water body is covered in clouds. Otherwise, cloudy images may give users a “false low” by reporting a low lake level simply because the water is not visible to the sensor. Using the original lake polygon, LLAMA counts all of the pixels inside that could potentially be used as water. It then uses a cloud cover mask that is a built-in function of Google Earth Engine to detect clouds. If the selected water body has <5% cloud cover, the image is discarded and not used as part of the lake level monitoring.

**3.2.4 Adjusted Water Detection Threshold**

LLAMA has a water threshold setting that can be adjusted to be more or less sensitive to water detection. After some preliminary testing, it was found that a higher threshold (less sensitive to water) was producing more accurate results in the winter months as it was not misclassifying snowy mixed pixels near the edge of the lake that would otherwise likely be counted as water due to their high MNDWI value. However, in the summer months, it was beneficial to have a lower threshold (more sensitive to water) because some images showed Lake Tahoe partially obscured (possibly by smoke, haze, or high altitude clouds). These images should still be useful for LLAMA to take a measurement. The solution is to allow LLAMA to adjust its threshold based on the sun angle in a Landsat image’s metadata. The threshold (which has potential values of 0-1) will oscillate between 0.05 and 0.5 based on a sun elevation threshold equation in LLAMA (see Appendix 1).

**3.3 Incorporating Water Quality Measurements**

**3.2.1 NDTI**

LLAMA is able to calculate a Normalized Difference Turbidity Index (NDTI, see Appendix 1) outlined by Güttler et al. (2013). The index uses the value in the red band along with an invariant bright value and an invariant dark value. These invariant values may be altered in the source code, but in LLAMA they are in a default setting used in an automated turbidity code provided by the Bay Area Environmental Research Institute. LLAMA provides the user with raster files showing turbid areas in the lake, indicated by a higher NDTI value.

**3.2.2 FAI**

LLAMA also calculates a Floating Algae Index (FAI, see Appendix 1) outlined by Hu, 2009. The index estimates algal blooms by identifying areas of greater chlorophyll-a concentration. Similar to the NDTI, FAI outputs raster images showing areas with higher levels of algae.

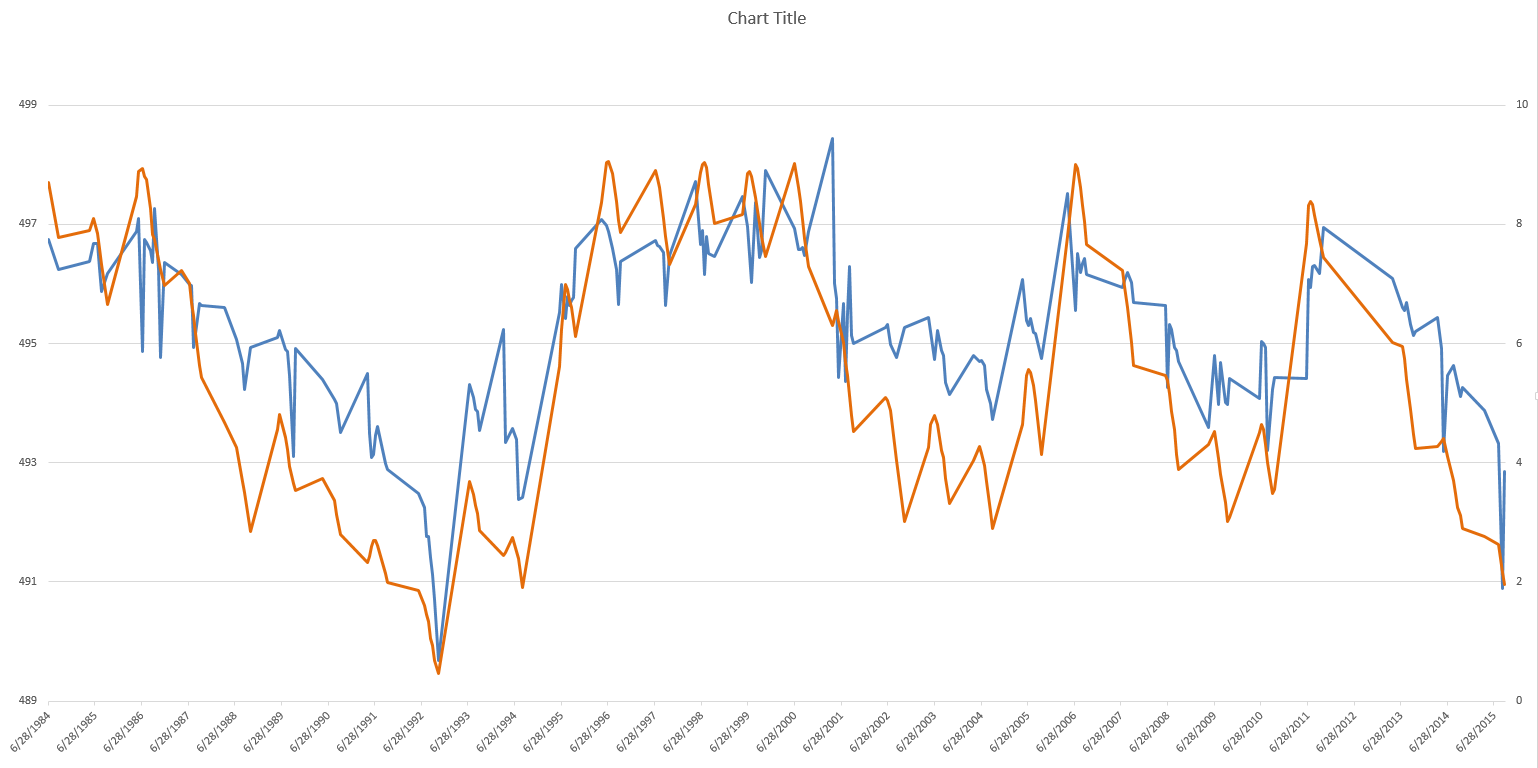
**3.4 *In situ* Validation**

For validation purposes, remotely-sensed lake level measurements from LLAMA were compared to *in situ* measurements taken by a USGS gauge station located near Tahoe City. LLAMA was run from April 25, 1984, to October 24, 2015. The final measurements were graphed next to the USGS lake level data, and a correlation value was computed to determine accuracy (see Section 4). To begin testing the usefulness of LLAMA as a global tool, the same methods were used to test the accuracy of LLAMA at the Salton Sea and Mono Lake, two prominent California water bodies south of Lake Tahoe.

# IV. Results & Discussion

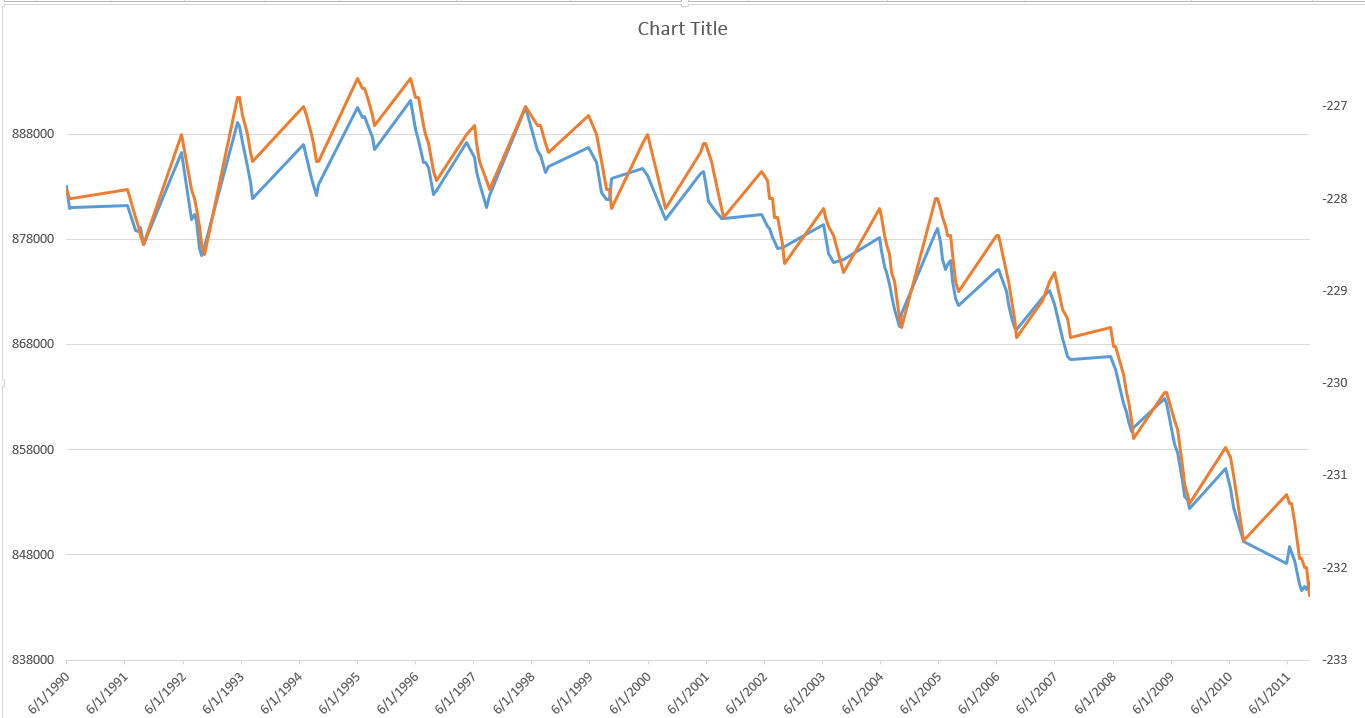
**4.1 Results and Analysis**

Computing a Correlation Coefficient (r) between the lake area that LLAMA calculates and the lake level measured by a gauge station gives a measurement of how close the two sets of data are and, therefore, a measurement of the accuracy of LLAMA.



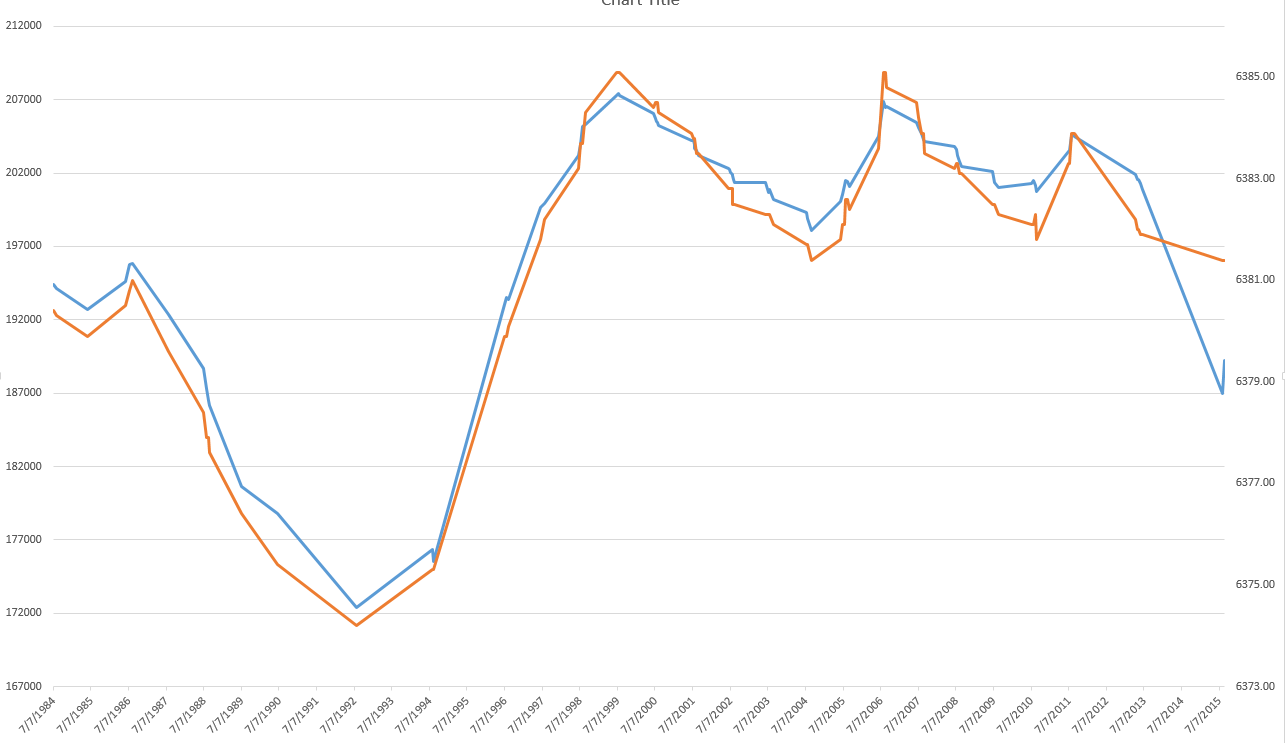
**Fig. 1:** This graph of Lake Tahoe shows the lake area in pixels that LLAMA measured along the y-axis (blue), and the lake level in feet measured by a USGS gauge station along the secondary y-axis (orange). The x-axis shows a date range from June 28, 1984 to June 28, 2015.

LLAMA demonstrated a high correlation (r=0.814) between the two datasets. While the measurements taken by LLAMA do not show a perfect match between the two datasets, it should be noted that both datasets show a distinct pattern of low levels towards the left of the graph in the early- to mid-1990’s, and two additional smaller drops in lake level towards the right of the graph. This, along with the strong correlation, suggests that LLAMA is an effective tool for monitoring the trends in lake level over time as well as monitoring drought conditions.



**Fig. 2:** This graph of the Salton Sea shows the lake area in pixels that LLAMA measured along the y-axis (blue), and the lake level in feet measured by a USGS gauge station along the secondary y-axis (orange). The x-axis shows a date range from June 1, 1990 to June 1, 2011.

The measured results for the Salton Sea show a very strong correlation (0.992) and show that for a lake with an ideal basin (see Section 4.2), LLAMA can be used not only as a tool to monitor lake trends and drought conditions, but also as a direct measurement of lake level. The strong correlation between lake area from LLAMA measurements and lake level measured by a USGS gauge station indicates that the Salton Sea has an ideal basin to be used with LLAMA (Fig. 2).



**Fig. 3:** This graph of Mono Lake shows the lake area in pixels that LLAMA measured along the y-axis (blue), and the lake level in feet measured by a gauge station along the secondary y-axis (orange). The x-axis shows a date range from July 7, 1984 to July 7, 2015.

Similar to the Salton Sea, Mono Lake shows a strong correlation between the lake area measured with LLAMA and lake level measured by a gauge station (r=0.970), indicating that for this lake, LLAMA can be used very effectively to monitor lake level.

**4.2 Discussion**

LLAMA is measuring change in water-covered pixels as a measurement of lake area. Lake Tahoe, however, is in such a steep basin that large changes in its lake level produce relatively small changes in its water pixel count. As a result, it is more difficult to measure changes in its area. This means its basin is not “ideal” to monitor using a 30-meter raster grid.

Mono Lake and the Salton Sea both see greater changes in area in terms of number of changed pixels. This means they have an “ideal basin” for remote sensing measurements, making both lakes easier for LLAMA to measure with greater correlation to their lake level.

**4.3 Errors and Uncertainty**

This project was specifically targeting Lake Tahoe. However, Lake Tahoe is in a very steep basin, which indicates that its shoreline changes relatively little even with large changes in lake level. As a result, a gauge station may register a relatively large change, but the moderate 30-meter resolution of the Landsat constellation only notices a modest number of pixels changing. The steeper the basin, the less the shoreline will change as a function of lake level, which decreases the accuracy of LLAMA. The steepness of the Lake Tahoe basin explains why LLAMA’s measurement showed a slightly lower correlation (r=0.814) between lake area LLAMA measured and the lake level measured by a USGS gauge station (Fig. 3).

The size of the water body must also be taken into account. Given that the lake’s area is calculated via a 30-meter raster grid, a single pixel on a smaller water body is responsible for a larger portion of the waterbody overall, making pixel errors increase total error significantly more as size of the water body decreases. This is not a problem for the large water bodies analyzed in Section 4.1.

**4.4 Limitations and Future Work**

In LLAMA’s present state, it is not able to take a measurement of a lake that is situated between two or more Landsat tiles. There are a number of ways to address this problem. LLAMA could take Landsat images with close proximity dates and add them together. It could calculate separate pixel counts of the portion of the water body that exists in a given tile and average their measurements over time. As a drastic alternative, LLAMA could utilize MODIS imagery when the lake is too large to fit on a single Landsat tile. MODIS imagery has a larger spatial extent (250m, 500m, 1km) and could measure larger water bodies. This, however, sacrifices spatial resolution.

In its present state, LLAMA relies on a Google Fusion Table (see Section 3.1) to find a lake. In the future, an “auto-detect” lake feature would be useful for LLAMA so that it may be used for any lake the user chooses. This would make the user input simpler and would make LLAMA more reliable for all water bodies, as it would not depend on pre-existing polygons.

A user of LLAMA may benefit in the future to have a more in-depth and interactive user interface (UI). The current UI of LLAMA effectively utilizes all the current features of LLAMA and is in its best possible condition given the very brief time allotted to its construction. With more time and personnel, the UI could be made more straightforward and interactive.

The oscillating water detection threshold was a solution put in place to solve the problem of snow misclassification on certain images during winter months and smoky or hazy images, mostly in the summer. Future work could be done to fine-tune and test the oscillating water detection threshold to ensure that it is adjusted to its most effective level.

# V. Conclusions

Lakes that fall have ideal basins were showing near-perfect correlations with their gauge station measurements. This means that LLAMA worked as expected and can be used by water managers worldwide to take lake level measurements, and monitor water resources and drought conditions consistently.

Lakes and water bodies that may not be contained in ideal basins, such as Lake Tahoe, are still able to be effectively measured by LLAMA. Given LLAMA’s ability to measure lake level trends over time means that it is well-placed in the Crisis Mapping Toolkit as a tool to monitor drought.

Additionally, this project was useful in examining the potential effectiveness of Google Earth Engine as a geospatial tool. Cloud-based computing offers a number of advantages including the fact that none of the large imagery files needs to be downloaded and held on a user's computer. It is also another advantage that a LLAMA user does not need any prior knowledge of remote sensing, imagery analysis, or Earth Engine to use LLAMA.

# VI. Acknowledgments

* Dr. Brian Coltin (NASA Ames Research Center)
* Dr. Scott McMichael (NASA Ames Research Center)
* Dr. Juan Torres-Pérez (Bay Area Environmental Research Institute)
* Dr. Joey Keely (Lake Tahoe Basin Management Unit, USDA Forest Service)
* Emily “Chippie” Kislik (NASA Ames Research Center)
* Victoria Ly (NASA Ames Research Center)

Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Aeronautics and Space Administration.

This material is based upon work supported by NASA through contract NNL11AA00B and cooperative agreement NNX14AB60A.

# 

# VII. References

Allen C., A. Macalady, H. Chenchouni, D. Bachelet, N. McDowell, M. Vennetier, T. Kitzberger, A. Rigling, D. Breshears, E. Hogg, P. Gonzalez, R. Fensham, Z. Zhang, J. Castro, N. Demidova, J. Lim, G. Allard, S. Running, A. Semerci, N. Cobb *A global overview of drought and heat-induced tree mortality reveals emerging climate change risks for forests* For. Ecol. Manag., 259 (2010), pp. 660–684 http://dx.doi.org/10.1016/j.foreco.2009.09.001

Davies, G., Mciver, L., Kim, Y., Hashizume, M., Iddings, S., & Chan, V. (2014). Water-Borne Diseases and Extreme Weather Events in Cambodia: Review of Impacts and Implications of Climate Change.*International Journal of Environmental Research and Public Health IJERPH,* 191-213.

Güttler, F., Niculescu, S., & Gohin, F. (2013). Turbidity retrieval and monitoring of Danube Delta waters using multi-sensor optical remote sensing data: An integrated view from the delta plain lakes to the western–northwestern Black Sea coastal zone. *Remote Sensing of Environment,* 86-101.

Hu, C. (2009). A novel ocean color index to detect floating algae in the global oceans. *Remote Sensing of Environment,* 2118-2129.

Kogan, F. (1997). Global Drought Watch from Space. *Bull. Amer. Meteor. Soc. Bulletin of the American Meteorological Society,* 621-636.

Sampson, D., Quay, R., & White, D. (2016). Anticipatory modeling for water supply sustainability in Phoenix, Arizona. *Environmental Science & Policy,* 36-46.

# VIII. Content Innovation

<https://drive.google.com/drive/folders/0B-CNovn6Jojjdm1XdXowSjFnMEU>

# IV. Appendices

**Appendix 1 - Equations**

**Appendix 2 - LLAMA tutorial**

1. **Accessing LLAMA**
   1. Google Earth Engine

LLAMA requires Google Earth Engine in order to operate. See appendix 3 to install Google Earth Engine to your computer:

* 1. Crisis Mapping Toolkit

<https://github.com/nasa/CrisisMappingToolkit>

The link above will allow you to download the Crisis Mapping Toolkit (CMT). LLAMA is contained in the CMT under the bin folder.

1. **Setting Parameters**
   1. Select Lake

Select the desired lake from the dropdown menu.

**Note:** If the lake of interest is not found in the database, please see the *Fusion Tables* section below for further instructions.

Fusion Tables

If the desired lake or water body is not presently in the LLAMA fusion table, you may add additional lakes upon request.

1. Create a shapefile of the lake or water body of interest using the WGS 1984 Web Mercator projection.
2. Upload the zipped shapefile to Shape Escape ([shpescape.com](http://shpescape.com/)).
3. Copy the link to the new fusion table provided by Shape Escape and send it to Dr. Brian Coltin ([brian.j.coltin@nasa.gov](mailto:brian.j.coltin@nasa.gov)) with a request to add it to the LLAMA fusion table.
   1. Select Date Range  
      Select the start and end dates for the study period (mm/dd/yyyy). The data selection begins April 25, 1984. It is important to recognize that LLAMA may or may not use the requested date; LLAMA’s image selection is dependent on Landsat’s temporal resolution and it may also discard an image if there is a high percentage of clouds.

**Note:** It is recommended to run LLAMA for the entire date range (1984-present) the first time you use the tool. It will run slowly the first time as it communicates with the Earth Engine Servers. Once it has been run once on all dates, it will run much more quickly when it is used in the future.

* 1. Select Rasters
     1. Heatmap:

The heatmap option will download a grayscale raster to your computer showing areas where the lake has had the most change.

* + 1. Turbidity:

Turbidity is measured using a turbidity index built into LLAMA. Selecting this option will instruct LLAMA to output a raster image showing areas of more turbid water. This image will be grayscale and can have a palette applied to it using your favorite geospatial software for more informative visualization.

* + 1. Algae:

This is measured using an algae index that is sensitive to chlorophyll-a concentrations. Selecting this option will instruct LLAMA to output a raster image showing areas of more algae. This image will be grayscale and can have a palette applied to it using your favorite geospatial software for more informative visualization.

* 1. Select Charts
     1. Graph

The graph option will provide a graph of the selected data with the date on the x-axis and lake area on the y-axis. This option is useful for seeing lake changes over time.

* + 1. Table

The table option will export LLAMA’s calculations to a .csv file. If you intend to use the data for your own research, choose this option.

1. **Interpreting Results**

File Structure: LLAMA will output several files, depending on which parameters you select before running the program.

**Text file**: Water counts, cloud counts, and various other data will be saved in a text file regardless of which parameters are selected. This file is important for improving the speed of subsequent runs of LLAMA, and should only be deleted for debugging purposes. This file will be saved in a folder named: ‘Results’ which, by default, will be in the folder where you saved the LLAMA script. Additionally, this file will automatically be named [LAKE\_NAME].txt. This file can be view manually, although this contains all the raw pixels counts at every date and has not had any statistical operations performed on it (such as removing outliers and cloudy images). If you wish to perform your own statistical operations on the data, the conversion factor of a Landsat pixel to square-kilometers is 1 square-kilometer per (1/.03/.03) pixels.

**Note:** Make sure not to edit the file in any manner, as this could cause errors in LLAMA's caching system.

**CSV File**: If the chart option is selected, a .csv file with the name [LAKE\_NAME].csv will be saved in the ‘results’ folder. This file will have outliers and cloudy data points removed using LLAMA's automated data clean-up functions.

**Note:** This is useful if you would like to display LLAMA’s results in Excel.

**Rasters**: If any or all of the raster check boxes are selected prior to running LLAMA, corresponding rasters of each type will be saved to the 'results' folder. The file structure will be as follows:

results/[LAKE\_NAME] Rasters.

Within this folder will be separate folders named according to each type of raster they contain (e.g. Turbidity, Algae, etc.). Within each of these folders are zipped folders containing the rasters. These can be unzipped in bulk using a program such as WinRAR or 7Zip. Each zipped folder is named with the convention [LAKE\_NAME]\_[IMAGE\_DATE]\_[RASTER\_TYPE].zip. The GeoTIF rasters within these zipped folders follow the same naming convention, as well. It is recommended that these be displayed with a color palate in a geospatial software for easier viewing.

**Graphs**: While LLAMA does not automatically save the graphs it outputs, they can be saved using the window containing the graph by clicking the Save button at the top of screen. It is important to note that this graph will be saved as an image, and cannot be re-formatted in any way. In order to create a custom-formatted graph, the user can use the CSV file outputted when the Table check box is checked. This graph contains the exact same data used in LLAMAs graph, and can be opened in any standard spreadsheet software, such as excel.

**Appendix 3 - Python installation tutorial of Google Earth Engine**

Google Earth Engine Installation Instructions Windows, Python 2.7 32-bit

**1. Sign up for Google Earth Engine access at: https://docs.google.com/forms/d/17-LSoJQcBUGIwfplrBFLv0ULYhOahHJs2MwRF2XkrcM/viewform**

**2. Setup Python 2.7 Paths for Windows**

(a) Go to: My Computer > Properties > Advanced System Settings >   
 Environment Variables

Then under system variables I create a new Variable called PythonPath. In this variable copy and paste:

C:\Python27\Lib;C:\Python27\DLLs;C:\Python27\Lib\lib-tk;C:\Python27\Lib\site-packages

(b) Also, append your installation path of python (ex. C:\Python27\) and

Variables C:\Python27\Scripts to the PATH variable in System variables. Do this by first locating PATH and then highlighting it and pressing Edit. Then add your installation and script paths to the end of the list, with each path separated by a semicolon.

**3. Install required python packages for Earth Engine**

(a) Go to: http://www.pythonware.com/products/pil/ and download the correct package or use this direct link: http://effbot.org/downloads/PIL-1.1.7.win32-py2.7.exe

(b) Download and Install Python Imaging Library for Python 2.7

(c) Run the Python Command Line Tool ('python')

(d) At the python prompt type "import Tkinter"

(e) At the python prompt type "from PIL import ImageTk"

(f) Both those commands should be error-free

**4. Install the Google Python API**

(a) At the cmd prompt run 'easy\_install google-api-python\_client' (or 'easy\_install --upgrade google-api-python\_client')

**5. Install the OAuth2 Client libraries**

(a) At the cmd prompt run 'easy\_install oauth2client' (or 'easy\_install --upgrade oauth2client')

**6. Install pyOpenSSL**

(a) At the cmd prompt run ‘easy\_install pyOpenSSL

(b) If it fails, download pyOpenSSL from here:

https://pypi.python.org/packages/py2.py3/p/pyOpenSSL/pyOpenSSL-0.15.1-py2.py3-none-any.whl#md5=8ca6d74b42df1e3178e59e9b08746d77

At the cmd prompt type ‘cd [insert path to file here]’

Followed by ‘pip install pyOpenSSL-0.15.1-py2.py3-none-any.whl’

**7. Install the Earth Engine API**

(a) At the cmd prompt run 'easy\_install earthengine-api' (or 'easy\_install --upgrade earthengine-api’)

(b) Or, navigate to the Earth Engine GitHub page at: https://github.com/google/earthengine-api, download the ZIP file and extract. Then run setup.py located in …\earthengine-api-master\earthengine-api-master\python\ using either IDLE or the command prompt.

**8. Setup OAuth2 account**

(a) Locate the ‘authenticate.py’ file located in C:/Python27/Lib/site-packages/ee, and run it using either the python command on the cmd line, or using IDLE.

(b) The program will ask you to enter an authentication code. A browser should open up and send you to a site with the authentication code on it. Copy and paste this into the cmd line or IDLE to continue.

Your set-up of Google Earth Engine should now be complete, and you should be able to use the Earth Engine module by first importing it into python using:

>>> import ee

>>> ee.Initialize()