Reservoir Assessment Tool (RAT): A Python Package for monitoring the dynamic state of reservoirs and analyzing dam operations

Sanchit Minocha[1], Pritam Das[1] and Faisal Hossain[1]*
[1]Department of Civil and Environmental Engineering, University of Washington, Seattle, WA, USA
* Corresponding author

Abstract: Advancements in satellite remote sensing has enabled the global tracking of reservoir operations, with the Reservoir Assessment Tool (RAT) emerging as a scalable modeling framework that captures the complete dynamic picture of a reservoir consisting of inflow, storage change, evaporation, surface area and outflow. To address the challenges regarding time-consuming installation and the lack of proper forum and documentation to update and test newer versions, we present RAT as a Python library, conveniently packaged with conda, and as a user-friendly command line utility. This simplifies installation, catering to both programming and non-programming users. Test cases and a Continuous Integration/Continuous Development (CI/CD) pipeline ensure robust code integration and community contributions. Thorough documentation is hosted at http://ratdocs.io. The transformation of RAT from a modeling framework into a self-reliant package enhances its usability, reliability, and collaborative nature, solidifying its position as a valuable tool at the nexus of remote sensing and hydrologic modeling.

Keywords: Dam Monitoring, Reservoir Operations, Hydrological Modeling, Satellite Remote Sensing, Python

Highlights:

- A globally scalable modeling framework, called RAT (Reservoir Assessment Tool), is presented as a python library and a command line utility.
- RAT can be easily installed using conda package manager along with its dependencies and the hydrological model.
- Testing functionality is available to verify if the installed RAT is functioning properly or if the changes in the source code are correct or not.
- Continuous integration and continuous development (CI/CD) pipeline has been established to automate and haste up the process of integrating additional code contributed by the community.

Software Availability: The conda python library is available on anaconda server at https://anaconda.org/conda-forge/rat. The source code of the python library is available on GitHub (https://github.com/UW-SASWE/RAT) under the GPL v3 license. Documentation on RAT is available at - https://rat-satellitedams.readthedocs.io/en/latest/.

1. Introduction

The data related to inland water quantities, regulated by dams with reservoirs, holds profound significance not only for water scientists but also for various fields such as food production, energy generation, and climate modeling (Jackson et al., 2001). The availability of surface water in a region is no longer solely determined by natural factors such as precipitation; human activities. Dam operations now exert significant influence over reservoir levels (Adrian et al., 2009). From here onwards, the terms 'dam' and 'reservoir' will be used interchangeably to refer to the artificial lakes created behind the dams. Water managers rely on this data to allocate water for domestic, industrial, and irrigation purposes, informing future plans and policies. Climatologists utilize this information to analyze the impact of dams on the environment and the dynamic climate. Additionally, ecologists find value in this data to comprehend the role of dam operations in relation to fish and wildlife ecosystems (Johnson et al., 2004). Therefore, this data plays a pivotal role in various fields, reflecting its broad-reaching implications beyond the realm of water science.

Traditional methods of monitoring and acquiring data, such as gauges and flow meters, are not only time-consuming but also economically burdensome. Moreover, the collected data is not consistently shared openly. Restrictions to data are often imposed by political hurdles that do not align with the basin-wide need for reservoir management. This disparity in data sharing can lead to downstream nations experiencing the adverse effects of poorly managed dam operations by their upstream counterparts, including droughts, floods, or compromised water quality (Hossain et al., 2023).

In addressing these challenges, satellite remote sensing can be a promising solution for monitoring dynamic reservoir conditions. Not only does it offer a cost-effective alternative, but it also provides real-time global data (Bonnema and Hossain, 2017; Bonnema et al., 2021). Despite these advantages, existing studies and remote-sensing tools often exhibit limitations, such as regional constraints or an inability to capture the complete dynamics of reservoir operations. Though, there are Python packages that can measure water body surface areas, they often overlook the potential of cloud computing for satellite image processing, do not integrate distributed physical modeling of the landscape, or are unsuitable for automating workflows (Owusu et al., 2022; Cordeiro et al., 2021).

Reservoir Assessment Tool (RAT) is one such modelling platform that can estimate various parameters like inflow, outflow, storage change, evaporation, and water level/surface area of a reservoir using satellite-based observations (Biswas et al., 2021; Das et al., 2022). To the best of our knowledge, there is no python package or any other remote-sensing based tool that offers such a comprehensive and complete view of dynamic state vector of reservoirs (comprising inflow, outflow, surface area, elevation, and evaporation).

RAT uses daily meteorological data that includes precipitation, minimum and maximum temperature, and wind speed from different webservers. This data is then fed to a hydrological model, which currently is Variable Infiltration Capacity (VIC) (Hamman et al., 2018), to derive inflow into a reservoir (figure 1 (e) & (f)). Satellite imagery is used to extract the surface area of a reservoir at different dates (figure 1 (b) & (c)). Penman equation (Penman, 1948) is used to estimate evaporation from a reservoir using surface area and the meteorological data while the difference in the surface area at two different dates is used along with Area Elevation Curve (AEC) to get elevation difference and thus, the storage change (figure 1 (a) & (d)). AEC can be provided by the user for each reservoir or will be generated automatically using digital elevation model. Once, inflow, evaporation, and storage change are estimated, outflow is calculated using the mass balance approach, ignoring any soil seepage as illustrated in figure 1 (Minocha et al., 2023).

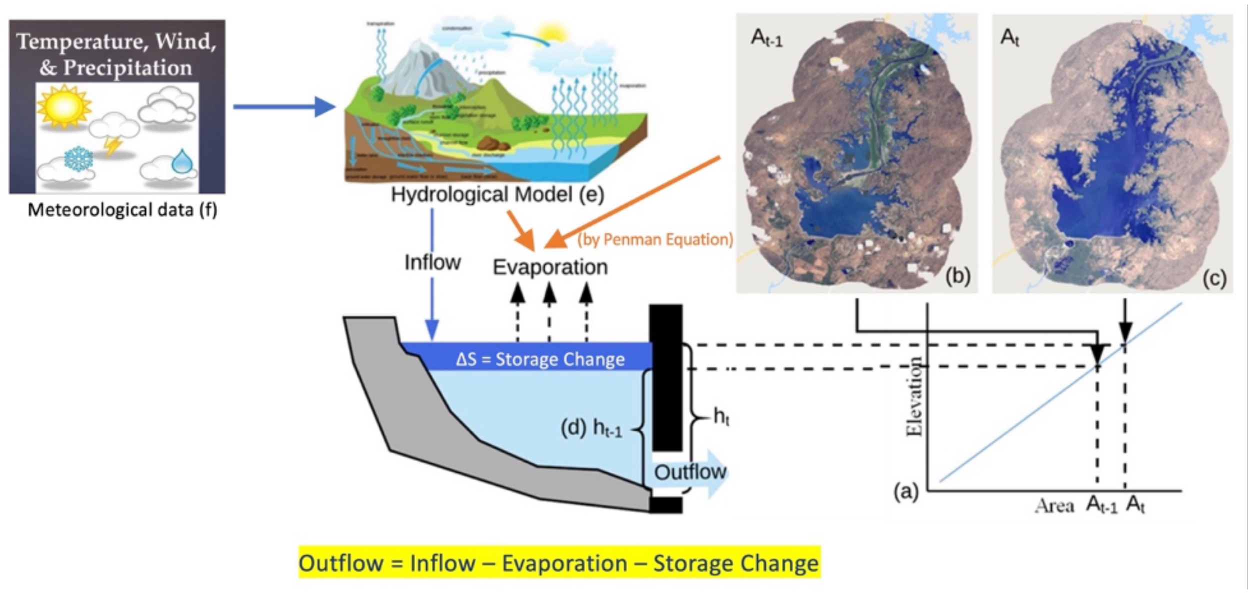

Figure 1: Different components of RAT Framework to estimate inflow, evaporation, storage change and outflow for a reservoir (After Minocha et al., 2023).

Recently, the third and latest version of RAT has acquired a completely overhauled architecture that makes it applicable worldwide. Noteworthy features of this architecture include scalability, error-handling capabilities, and robustness in dealing with missing meteorological data, a crucial component for running hydrological models. The revamped architecture also enhances computational efficiency by incorporating parallel processing for the downloading and processing of meteorological data and makes it user-friendly, requiring minimal inputs a user (Minocha et al., 2023). These attributes position RAT as a tool for enabling breakthroughs across hydrology-related fields.

While RAT holds the potential to drive advancements in research and enhance water management practices, its widespread adoption may be hindered by the absence of a streamlined channel for setting up the model and its dependencies. Despite the user-friendly architecture of RAT 3.0, the tool might struggle to reach a broader audience within the water community if the installation process proves challenging. Additionally, there is a critical need for a systematic method to test and verify the correct installation of the model. Given RAT's status as an open-source model that encourages contributions from the community, there is a further imperative for automated testing mechanisms to ensure that contributed code functions as intended. To facilitate seamless utilization, users should have straightforward ways to update and access the latest model versions. Therefore, addressing these installation and testing challenges is essential for maximizing the impact of RAT and promoting its effective integration into water-related research and management practices.

This paper introduces RAT as a Python library and presents it as a command line utility, with the goal of enhancing its accessibility to a broader community of tool developers for river studies. Because RAT harnesses the synergies of remote sensing and hydrologic modeling, we believe it should be presented as a cohesive package. This approach alleviates concerns about the installation of disparate components of the hydrologic model or other dependencies, offering users a seamless experience. To facilitate this, Conda serves as a channel, streamlining the installation process across diverse operating systems. Consequently, a self-installation feature has been integrated into the RAT package, ensuring the download of all necessary requirements.

The paper delves into the system requirements for the RAT library and explores how computational efficiency can be heightened by utilizing a greater number of CPU cores. The establishment of the library also facilitates the development of plugins by the growing community, catering to diverse functionalities. It can also benefit the research community by providing time-saving utility functions to analyze the resulting outputs from RAT. A case-study showcasing an application of the library is also included to demonstrate the ability of RAT in tracking reservoir operations.

2. Building a Python Library

2.1. Dependencies

While RAT's source code is entirely Python-based, its execution requires the installation of a hydrological model and essential Python dependencies. The current RAT architecture uses the VIC hydrological model to compute reservoir inflows. However, VIC relies on MetSim (Bennett et al., 2020) for input forcings to generate surface runoff across a river basin which then requires a routing model (Hamman et al., 2018) to estimate streamflow at the dam location. Though MetSim operates in Python, VIC, coded in C, requires compilation using a C compiler like gcc. The Fortran-based routing model used by RAT requires a compiler such as gfortran. Additionally, creating executables for each of these models involves the use of the make utility to read makefile. Consequently, a Unix-based Operating System is crucial for the entire hydrological model to operate seamlessly.

The multiple language dependencies, spanning Python, C, and Fortran, poses a challenge for users attempting to install them separately, particularly given their OS-dependent nature. To streamline the installation of these diverse language dependencies, many software applications make use of a package manager. Conda, a cross-platform, language-agnostic binary package manager, adeptly handles dependencies across multiple languages. Therefore, RAT has been packaged using conda and integrated into conda-forge, a community-led organization that hosts conda packages for a wide array of software applications (conda-forge community, 2015). Beyond addressing dependency issues, conda also simplifies the process for users to update the software to its latest version.

S. No.	**Dependency Name**	**Functionality**
1.	python >=3.6	To execute RAT's source code
2.	make	For reading makefile and creating executable of routing
3.	gfortran	For compiling routing source code
4.	wget	To download meteorological data from different servers

5.	rasterio	To read and write raster data like precipitation from/to raster files
6.	xarray	To manipulate raster data in the form of data structures
7.	rioxarray	To read and write raster data from/to the data structures
8.	earthengine-api	To access Google Earth Engine Python API
9.	pandas	To read, write and manipulate tabular data like inflow timeseries
10.	geopandas	To read and write georeferenced tabular data like shapefiles
11.	numpy	For mathematical operations in the form of arrays
12.	netcdf4	To read and write NetCDF files.
13.	dask	To do parallel computations for routing; download weather data
14.	scipy	Interpolation methods used by TMS-OS (Das et al., 2022)
15.	scikit-learn	Clustering methods used to extract height from altimetry data
16.	ruamel_yaml	To read and write configuration file with comments
17.	yaml	To read and write configuration file
18.	gdown	To download global database from google drive

Table 1: List of dependencies of RAT python library.

Table 1 lists the dependencies installed when RAT python library is installed using conda. While conda effectively handles the installation of the dependencies necessary for setting up the hydrological model (such as the make utility), it doesn't automatically install specific components of the hydrological model that includes VIC, MetSim, and the routing models. Both VIC and MetSim are available as conda packages, but challenges arise due to compatibility issues stemming from their need for different versions of common libraries. This incompatibility necessitates users to install them in separate environments. Additionally, the routing model, not offered as a conda package, mandates users to download the source code and manually compile it using the compiler installed by conda during the RAT installation. Collectively, these factors make it necessary to install all the three models individually and manually. This problem has been addressed in section 3.1.

2.2. System Requirements

The execution of RAT involves two computationally intensive processes. Firstly, the hydrological model, and secondly, the processing of satellite imagery to derive surface area estimates for a reservoir. For efficient satellite imagery processing, RAT leverages cloud computing through Google Earth Engine (GEE) (Gorelick et al., 2017). As a result, the primary requirements of RAT are driven by the hydrological model. Currently RAT uses VIC hydrological model for inflow estimation which requires a Unix-based operating system.

Like most other Python libraries, there is no specific minimum specifications for a machine for RAT. A standard laptop equipped with 4 GB RAM, a 512 GB hard disk, and 2 CPU cores is sufficient to run RAT. However, more powerful machines with increased RAM and CPU cores can significantly reduce computational time. Table 2 compares the computational time (measured in wall time) on two different machines—a standard laptop (MacBook) and a server machine. The execution of RAT was conducted for the Hidkal Dam in the Krishna River Basin (see figure 2) over a one-year period from January 1, 2019, to December 31, 2019. No spin-up for hydrological model was required as initial state files were used.

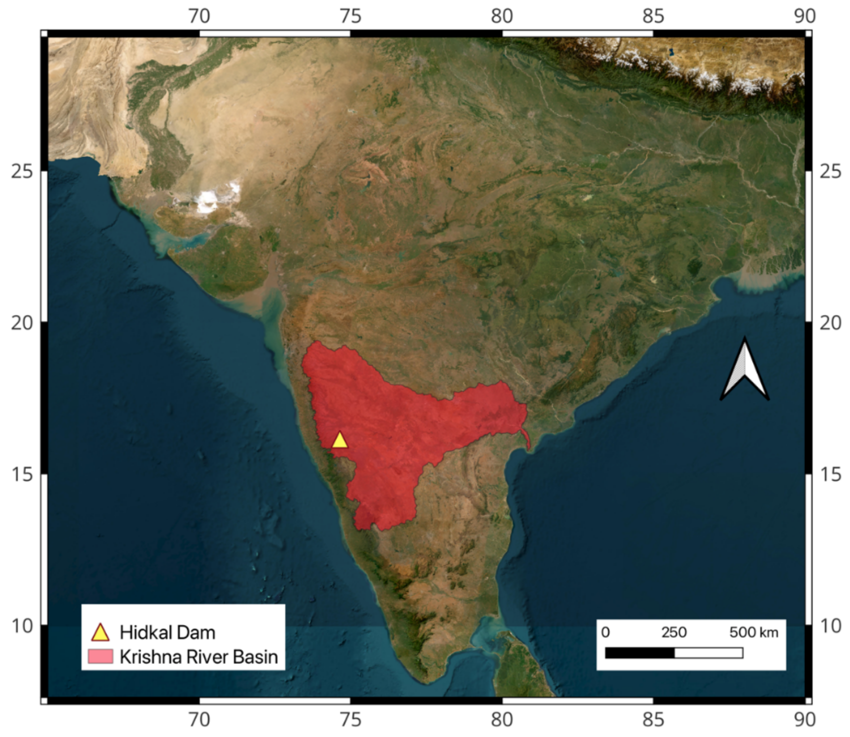

Figure 1: Map of the Krishna River basin, showing location of the Hidkal dam/reservoir.

It should be noted that wall time for surface area estimation is independent of the machine specifications as it is dependent on the cloud machine provided by GEE. Also, with greater number of cores the total run time reduces significantly as RAT makes use of parallel computations. Another noteworthy observation is the substantial increase in MetSim's wall-time on server machine when utilizing a single core compared to a single core on a less powerful machine. One of the potential reasons for it could be that the computational efficiency of MetSim may not be fully optimized for the enhanced processing capacity of the stronger machine. This behavior indicates the need for a deeper investigation into the MetSim's compatibility with varying hardware configurations, shedding light on potential optimizations and areas for improvement.

System Specifications	**Macbook Laptop**: 2.3 GHz Dual-Core Intel Core i5 CPU; 16 GB RAM; Mac OS		**Server Machine**: 3 GHz Intel(R) Xeon(R) Gold 5317 CPU with 48 cores; 256 GB RAM; Linux OS	
Number of cores used	1	4	1	48
Run Time to download meteorological data	240.17 minutes	170.95 minutes	22.02 minutes	3.2 minutes
Run Time for MetSim	17.95 minutes	15.08 minutes	68.32 minutes	8 minutes

Run Time for VIC	15.63 minutes	15.05 minutes	9.97 minutes	1.72 minutes
Run Time for Routing	3 seconds	3 seconds	2 seconds	1 second
Run Time for Surface Area Estimate	7.93 minutes	8.32 minutes	9.68 minutes	10.52 minutes
Total Run Time	280.48 minutes	211.82 minutes	110.8 minutes	24.4 minutes

Table 2: Computational wall time comparison of RAT for different machines using different number of CPU cores.

3. Enhancements Incorporated in the Python Library

To improve accessibility and expand the user base, a command-line interface (CLI) has been integrated into RAT while building the package using conda. This CLI not only introduces additional functionalities for enhanced user convenience but also promotes inclusivity by catering to individuals with diverse technical backgrounds. Besides CLI, several utility functions have been added as a part of the library that facilitate users and developers while using RAT. This section describes the various functionalities added to the RAT package.

3.1. Initialization

As discussed in section 2.1, RAT requires separate and independent downloading and installation/compiling of Metsim, VIC and routing model. To simplify this process, the CLI command 'rat init' streamlines RAT's initialization. It accomplishes this by automatically downloading and installing the required Python environments for MetSim and VIC, as well as fetching and compiling the routing scripts. Moreover, it acquires several default parameter files necessary for VIC, MetSim, Routing, or RAT with the option to modify these files as necessary during calibration and validation. This command also offers users the option to download a global database, encompassing multiple global datasets, which can serve as the default values for a newly setup RAT instance (Minocha et al., 2023). The command requires a path to the directory where to install all the models and download data, referred to as the project directory in the context of RAT. Notably, the initialization of RAT is a one-time requirement following the installation of the RAT conda package.

3.2. Testing

Once RAT has been installed and initialized, users can test the functioning of RAT by using ‘rat test’ command in CLI. It requires users to provide a secret file that contains credentials for using google earth engine and to download IMERG precipitation data (Huffman et al., 2014) along with Jason-3 altimetry data. Other than a secret file a user needs to tell the path of the project directory and the name of the test river basin. Currently, users can choose between two test basins Nueces in Texas (figure 3) or Gunnison in Colorado (figure 4) to test if RAT has been installed and initialized correctly. This command downloads a test dataset with parameter files for the chosen test river basin and expected results. It runs RAT for that test basin and compares the outputs to expected results across six test cases: inflow, outflow, surface area, storage change, evaporation, and area elevation curve. It also validates the user provided credentials in the secret file.

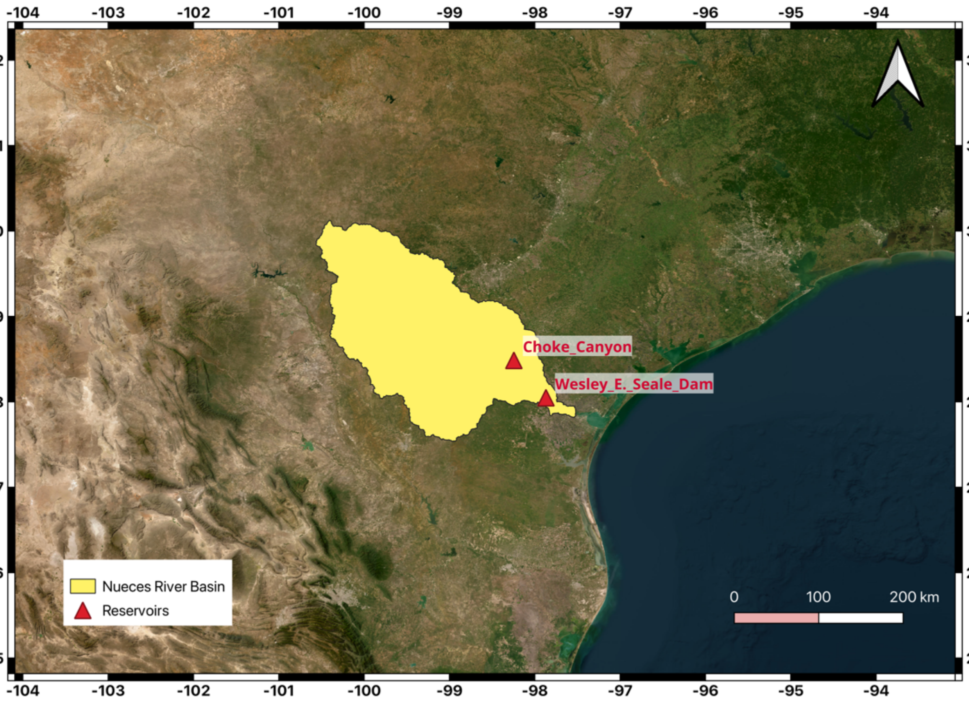

Figure 3: Map of the Nueces River basin, showing location of the reservoirs used for test case.

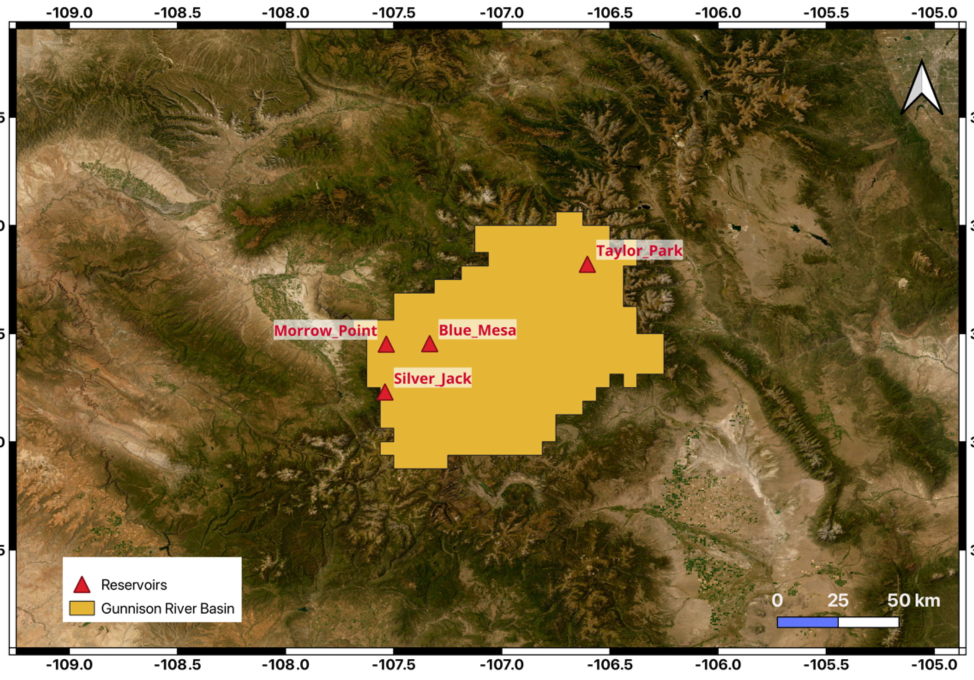

Figure 4: Map of the Gunnison River basin, showing location of the reservoirs used for test case.

3.3. Execution

To run RAT through the command-line interface, one can utilize the 'rat run' command by specifying its configuration file. It also offers users the option to run RAT in an operational mode, which means that RAT will resume from the last state of a previous run, commencing execution from the end date of that prior run. Moreover, executing and operationalizing RAT can also be accomplished through Python scripts for more flexibility and customization.

3.4. Configure

Given that the RAT configuration file contains numerous parameters, many of which may prove perplexing for beginners while being useful for advanced users, there exists a 'rat configure' command. This command automatically fills in specific parameter values in the configuration file using user-provided options like number of CPU cores to use and secret file path. If users have downloaded the global database, they can provide the path of this database to set default values to a lot of parameters in the configuration file, enhancing convenience and clarity.

3.5. Toolbox

A ‘toolbox’ named module has been developed and added to the library that can be accessed in python. The purpose of this toolbox is to provide utility functions that make it easy to access and use RAT. It is now possible to update the RAT’s configuration file from within python with just one line of code. It is especially useful when RAT needs to be executed in iteration for example while calibrating VIC model. It also provides functionality to compare RAT estimates with in-situ data or data from other sources with different metrics like Nash-Sutcliffe efficiency (nse) and Kling-Gupta efficiency (kge). With the toolbox, users can now also create interactive visualizations for the time-series of different variables like inflow, outflow, and storage-change.

3.6. Plugins

With the introduction of the RAT library, it is now convenient for the developing community of RAT to create and add plugins to the existing framework of RAT. These plugins can be built to cater to the different needs of the water community. One of the recent plugins that has been developed for RAT is Reservoir Operations driven River Regulation abbreviated as ResORR (Das et al., 2023). ResORR is a model that accounts for regulation of rivers due to upstream reservoir operations in the estimation of downstream reservoir inflow and outflow. It is scalable and can be applied globally as it requires minimal parameterizations and uses satellite data along with intermediary outputs of RAT.

Another plugin that is under development for RAT is to provide up to 15 days of forecasted inflow for any reservoir worldwide using the weather forecast. It will be helpful to track the extreme flood events and will provide water managers a chance to better handle the water level in the reservoirs. Even though RAT can account for the changing bathymetry of a reservoir due to sedimentation if survey-based in-situ area elevation curve (AEC) is provided, there is no inherent way to know how much storage has lost due to the trapping of sediments in the reservoir. Research is on-going to develop a plugin for RAT that can give some insight on the storage loss due to reservoir sedimentation using remote sensing data.

4. Maintenance of the Python Library

As a thorough and up-to-date documentation is a key component of any conda library, a comprehensive and meticulous documentation is readily available and continually maintained at two URLs: http://ratdocs.io and https://rat-satellitedams.readthedocs.io/. This documentation serves as a valuable resource for users of RAT, offering a user-friendly guide for swift and trouble-free installation. Within this documentation, users will find a wealth of information, including:

- **User Guide**: This guide simplifies the installation process for the RAT Python package, making it quick and accessible.
- **Command Line and Python Interface**: Instructions on how to use command line interface of RAT to initialize and verify the successful installation of the package are provided. Commands that can be used in python to execute RAT and analyze results are also listed.
- **Directory Structure**: Details about the directory structure created by RAT, along with guidance on locating data files generated by the RAT tool.
- **Global Datasets**: Descriptions of the global datasets included with RAT for user convenience.
- **Model Architecture**: A comprehensive exploration of the model's architecture is found in the 'Computational Model' section, while the 'Conceptual Model' section elucidates the theoretical model.
- **Configuration Parameters**: Each configuration file parameter is explained, including its necessity, description, and required syntax.
- **Developer Guide**: Easy guide summarizing how to use developer version for new developers who want to contribute towards the improvement and enhancement of the software.

Furthermore, the documentation not only covers the creation and acquisition of various credentials needed for successful RAT execution but also offers guidance on consolidating them into a single 'secrets' file.

This documentation plays a pivotal role in democratizing access to information regarding the dynamic state of reservoirs and acts as the main guide for developers to contribute to the source code of RAT. The authors encourage contributions from the community and report issues faced at https://github.com/UW-SASWE/RAT/issues. The discussion forum dedicated to support the user community of RAT can be accessed at https://github.com/UW-SASWE/RAT/discussions.

5. Application of the Python Library

This section demonstrates how the RAT 3.0 library can be used not only to accelerate and do research analysis but also has significant potential for water managers to understand surface water availability where ground data is not available. For this application, RAT 3.0 has been applied in the Krishna River Basin in India. To keep the scope of this application small, only single dam has been selected which is Hidkal dam, also known as, Raja Lakhamagouda dam (figure 2).

There is no doubt that heavy and erratic rainfall is one of the reasons for floods in several parts of the world. But that is often coinciding with dams being full due to poor management, resulting in dam-induced floods. In the first week of August 2019, due to heavy rainfall in the Monsoon season, severe flood affected the southern Indian State of Karnataka. Thousands of people were evacuated to safer places and relief camps. One of the districts that was drastically affected by these floods is Belagavi district where Hidkal dam is located. It is speculated that the dam worsened the impact of floods. So, this application of RAT attempts to verify the role of Hidkal dam in the Karnataka floods of 2019.

5.1. Setting Up

RAT 3.0 was installed and initialized using conda along with the global database (Minocha et al., 2023). The shapefile delineating the Krishna River basin was sourced from the Global Runoff Data Centre's (GRDC) river dataset (GRDC, 2020), while the shapefiles for the Hidkal dam and reservoir were obtained from the Global Reservoir and Dam Database (GRanD) v1.3 (Lehner et al., 2011). Both datasets are provided as a part of RAT's global database. The different service accounts required to execute RAT 3.0 were established and the credentials were consolidated into a single secret file. Essential parameters, including the number of cores for RAT's multiprocessing mode, execution steps, and the specified start and end dates, were defined in RAT's configuration file. Subsequently, RAT was executed based on the configured settings, covering a one-year duration from January 1, 2019.

5.2. Inflow

The inflow into the Hidkal reservoir was simulated for the entire year of 2019 at a daily timestep and subsequently compared against in-situ data obtained from the Karnataka Water Resources Department (figure 5). In particular, the model accurately identified the peak inflow during the flood on August 8 and 9. However, the model indicated a substantial offset in the magnitude of the peak, suggesting a potential underestimation in the rainfall dataset used (IMERG) as it is difficult to achieve such high inflow by the calibration of the VIC model. Despite the magnitude offset, the model's ability to precisely capture the onset and recession time of the flood event, as demonstrated by Suresh et al. (2023), underscores RAT's capability to effectively track such events using solely satellite data.

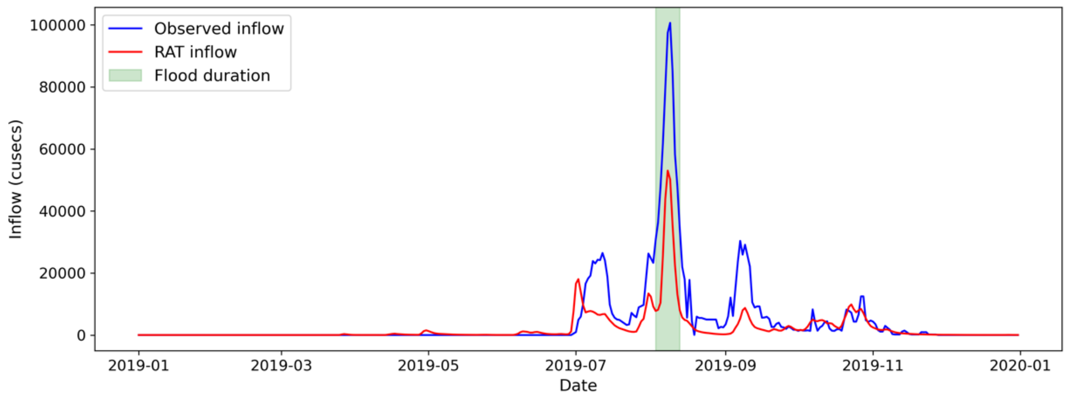

Figure 5: The timeseries of inflow (measured in cubic feet per second) into the Hidkal Reservoir throughout the year 2019. The peak flood duration is highlighted in greenish color.

5.3. Surface Area

RAT tracked the surface area of the reservoir using Google Earth Engine (GEE) Python API from the start of the year 2019 to the end of 2019 (figure 6). The surface area estimates were estimated using the Tiered Multi Sensor-Optical and SAR (TMS-OS) algorithm at a frequency of

2-5 days. As satellite remote sensing data is often recognized for qualitative trends due to potential bias issues, the trend of reservoir level has been accurately picked by RAT. The reservoir level was lowered from March to June which is the pre-monsoon period in South Asia. This standard practice is typically employed to accommodate the anticipated high inflows during the monsoon, allowing the reservoir to reach its full capacity by the end of the monsoon season. However, in this particular case, the reservoir was almost filled completely by early August, which is only half-way through the monsoon.

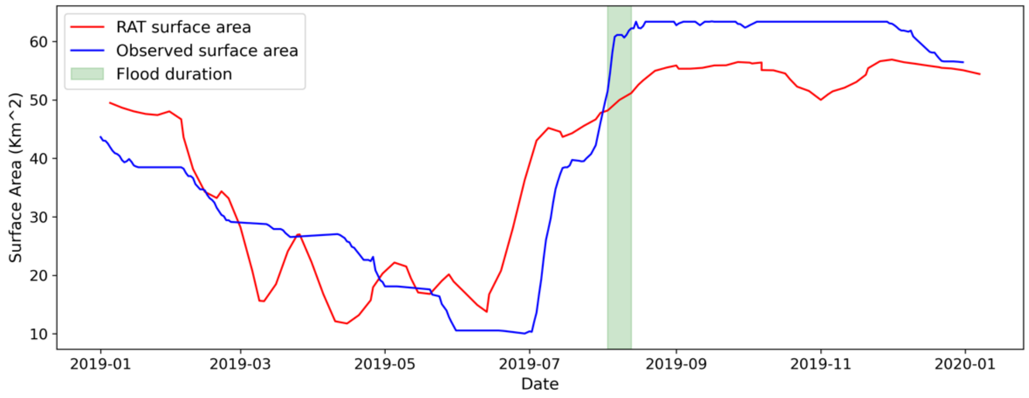

Figure 6: The timeseries of surface area (measured in Km^2) of the Hidkal Reservoir throughout the year 2019. The peak flood duration is highlighted in greenish color.

5.4. Evaporation

The amount of evaporation from the reservoir was estimated by RAT using the penman's equation at a daily frequency (figure 7). At the time of flood peak, the evaporation is very low which is probably due to high humidity in the air in that region.

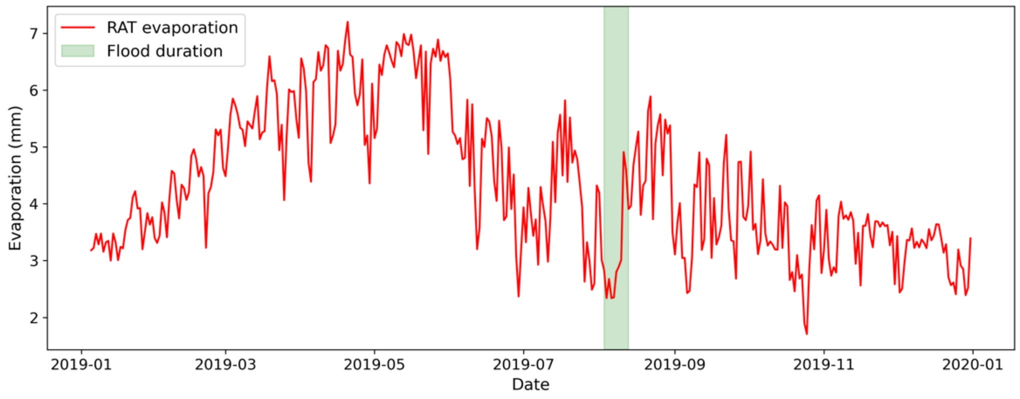

Figure 7: The timeseries of evaporation (in mm) from the Hidkal Reservoir throughout the year 2019. The peak flood duration is highlighted in greenish color.

5.5. Area-Elevation Curve

RAT constructed the area elevation curve (AEC) for the Hidkal reservoir (figure 8) by leveraging the SRTM digital elevation model available in Google Earth Engine (GEE), as no survey-observed AEC was provided as input (Farr et al., 2007). Since SRTM data was recorded in 2000 whereas the reservoir was created before 2000, it does not have elevation data below the reservoir level which was at the time SRTM was recorded. RAT 3.0 therefore extrapolates the AEC for lower reservoir levels. Users are advised to utilize in-situ AEC data to enhance the precision of storage change and outflow estimations derived from RAT.

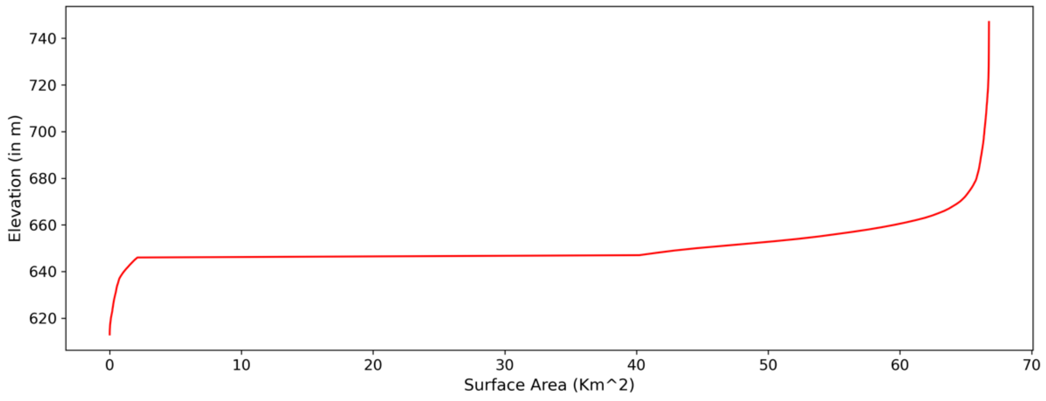

Figure 8: The Area Elevation Curve (AEC) for the Hidkal Reservoir generated by RAT 3.0 using SRTM digital elevation model.

5.6. Outflow

The estimation of outflow from the Hidkal dam involves utilizing the mass balance approach, which considers inflow, storage change, and evaporation. RAT calculates storage change using surface area estimates and the area elevation curve (AEC), resulting in an outflow temporal frequency matching that of surface area estimates. While the mass balance approach incorporates uncertainties from each component, the lower temporal frequency further amplifies these uncertainties. Despite these challenges, RAT successfully identifies the timing of the peak outflow during the flood event (figure 9). The considerable difference in the magnitude of the peak outflow, compared to in-situ measurements, is primarily attributed to the underestimation of inflow at that particular time. The figures 5 and 9 illustrate a noticeable offset in the magnitude of the estimated peak inflow and outflow when compared to in-situ data, indicating a consistent level of discrepancy.

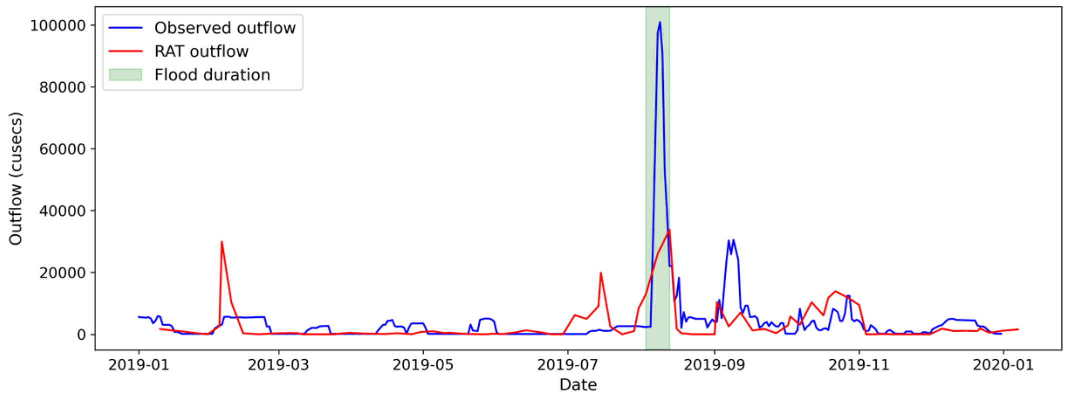

Figure 9: The timeseries of outflow (measured in cubic feet per second) from the Hidkal Reservoir for the year 2019. The peak flood duration is highlighted in greenish color.

The above application shows that RAT was able to capture the timing of the onset and recession of the flood. It was also able to track the reservoir level during the flood. With this analysis, it can be noticed that the reservoir was filled almost to its maximum capacity when there was still half of the monsoon period left. This forced the dam operators to release all the water that the reservoir was getting during the flood with no scope of managing the flood water. This can be verified by looking at the ratio of the RAT generated estimates of inflow and outflow.

6. Conclusions and Discussion

With the increasing number of dams and reservoirs all around the world, the availability of surface water is determined by a complex interaction of natural processes and human decisions. Human control in terms of reservoir operations largely influences the quantity of surface water available in a region. Therefore, it is important to monitor reservoir operations. However, the high cost of installation and maintenance of monitoring tools make it impossible to track every reservoir on the globe. Satellite remote sensing reduces this cost as it gives real-time data globally.

Reservoir Assessment Tool (RAT) is one such remote-sensing based reservoir monitoring tool that gives data related to reservoir dynamics consisting of inflow, outflow, storage change, evaporation, and surface area at a frequency of 2-5 days. The latest release of RAT 3.0 has introduced significant changes in the software architecture of RAT 3.0 that has improved the reservoir tracking tool in terms of computational and memory efficiency, easy-of-use, computational robustness, and global scalability. Despite all these enhancements, there is the challenge of time-consuming installation and setting up of the tool for-the-first user. Such users also need to verify the correct installation of the tool. A well-maintained, well-documented and continuously evolving software package requires a dedicated forum to archive the newer and latest versions for the users to upgrade. Similarly, the developer community requires proper testing functionality to auto-verify the correctness of their contributed code for improvement of the RAT 3.0 package.

In this paper RAT 3.0 has been presented as a python library that streamlines the installation process to maximize the chances of a seamless experience for the first-time user. The python library is created using the conda package manager that also acts as the channel for users to easily upgrade the library to the latest version available. The library has been hosted on the conda-forge repository where most of the open-source conda packages are hosted. At the time of writing this manuscript, the python library has already been downloaded more than 1350 times, out of which more than 60-70% of the downloads are from the community of first-time users (i.e., non-developers). Packaging the python library using conda also proved beneficial to distribute RAT 3.0 as a command line utility. Different command line functions have been added to increase the versatility of the tool. It is now possible to install the complete VIC hydrological model along with its components MetSim and Routing using a single command.

The installation of RAT 3.0 can now be verified by the user using any of the two test river basins. Similarly, the tests are also used to verify the code contributed by the developing community. Other user commands that can be used in command line are 'configure' that partially auto fills a configuration file and 'run' that executes RAT given a configuration file. With the release of this python library, it is now easy to develop and add plugins for RAT 3.0 to serve additional functionalities. Toolbox has been developed for the users to easily create and update configuration file, create interactive visualization plots of the RAT outputs and to compute the error between the estimated and the in-situ data using different error metrics.

Along with the python library, efforts have been made to maintain it and keep it updated. Continuous Integration / Continuous Development (CI/CD) pipeline has been established using GitHub actions that automatically verifies the working of the software after incorporating the contributed code by the developers and then creates a newer version, if passed. A detailed and thorough has also been prepared and hosted online. It will be useful not only for users but also for the developing community as they will be able to understand the code and improve it further by their contribution.

This paper has also showcased the use and ability of RAT to track the onset and offset of the flood using only remote-sensing data. It is able to track the reservoir level or its surface area before and during the flood bringing light to the role of dam operations in the damage caused by floods. The application of RAT is not limited to track extreme events like floods or droughts. It can be used as a framework to model the aspect of reservoirs in climate models without any assumptions related to functioning of dam. It can be used as a data driven way of understanding how the reservoirs are being operated at a global level. The availability of surface water and how much of that is being lost due to evaporation can be tracked using RAT.

Even though there are certain limitations in terms of the precision of quantitative analysis, the model can be calibrated to in-situ data to achieve more accurate results. The estimates of inflow or surface area/reservoir level can be swapped with the actual observed data to reduce the uncertainty of the outflow estimates. Using AECs representing the latest bathymetry of the reservoir, which considers the storage loss due to sediment trapping, can further yield more accurate estimations.

As the CI/CD pipeline has been set up, it is now easy for the community to tackle the limitations and improve the software. New and advanced satellite missions like Surface Water and Ocean Topography (SWOT) can be incorporated to improve the accuracy of the surface area estimates which can further improve the estimation of outflow from the reservoir. Plugins can be developed that make use of RAT outputs and further give insights on reservoir sediment trapping efficiency, rule-curves, or water quality. The important thing to note is that the infrastructure, the forum, and the pipeline for open-source satellite-based reservoir monitoring is now established so that the continuous improvement of the RAT software can be initiated by anyone from anywhere in the world.

Acknowledgements:

This work was supported by the NASA Applied Science Program for Water 80NSSC22K0918. Additional support to the first author from the NSF-supported Graduate Training program called Future Rivers and the Ivanhoe Foundation is gratefully acknowledged.

7. References

Adrian, R., O'Reilly, C. M., Zagarese, H., Baines, S. B., Hessen, D. O., Keller, W., ... & Winder, M. (2009). Lakes as sentinels of climate change. *Limnology and oceanography*, 54(6part2), 2283-2297.

Bennett, A., Hamman, J., and Nijssen, B. (2020). MetSim: A Python package for estimation and disaggregation of meteorological data. *Journal of Open Source Software*, *5*(47). https://doi.org/10.21105/joss.02042

Biswas, N. K., Hossain, F., Bonnema, M., Lee, H., and Chishtie, F. (2021). Towards a global Reservoir Assessment Tool for predicting hydrologic impacts and operating patterns of existing and planned reservoirs. *Environmental Modelling and Software*, *140*. https://doi.org/10.1016/j.envsoft.2021.105043

Bonnema, M. and Hossain, F. (2017). Inferring reservoir operating patterns across the Mekong Basin using only space observations, *Water Resources Research*, *53*, 3791–3810, (http://dx.doi.org/10.1002/2016WR019978).

conda-forge community. (2015). The conda-forge Project: Community-based Software Distribution Built on the conda Package Format and Ecosystem. Zenodo. http://doi.org/10.5281/zenodo.4774216

Cordeiro, M. C., Martinez, J. M., & Peña-Luque, S. (2021). Automatic water detection from multidimensional hierarchical clustering for Sentinel-2 images and a comparison with Level 2A processors. *Remote Sensing of Environment*, *253*, 112209.

Das, P., Hossain F., Minocha S., Suresh S., Darkwah G., Andreadis K., Lee H., Laverde M., Oddo P.(2023) ResORR: A Globally Scalable and Satellite Data-driven Algorithm for River Flow Regulation due to Reservoir Operations, *Environmental Modelling and Software* (In review)

Das, P., Hossain, F., Khan, S., Biswas, N. K., Lee, H., Piman, T., Meechaiya, C., Ghimire, U. and Hosen, K. (2022). Reservoir Assessment Tool 2.0: Stakeholder driven improvements to satellite remote sensing based reservoir monitoring. *Environmental Modelling & Software*, *157*, 105533. https://doi.org/10.1016/j.envsoft.2022.105533

Farr, T. G., Rosen, P. A., Caro, E., Crippen, R., Duren, R., Hensley, S., Kobrick, M., Paller, M., Rodriguez, E., and Roth, L. (2007). The shuttle radar topography mission. *Reviews of Geophysics*, *45*(2).

Gorelick, N., Hancher, M., Dixon, M., Ilyushchenko, S., Thau, D., & Moore, R. (2017). Google Earth Engine: Planetary-scale geospatial analysis for everyone. *Remote Sensing of Environment*, *202*. https://doi.org/10.1016/j.rse.2017.06.031

GRDC. (2020). *Major river basins of the World/Global Runoff Data Centre (GRDC)*. Federal Institute of Hydrology (BfG) Koblenz, Germany.

Hamman, J. J., Nijssen, B., Bohn, T. J., Gergel, D. R., and Mao, Y. (2018). The variable infiltration capacity model version 5 (VIC-5): Infrastructure improvements for new applications and reproducibility. *Geoscientific Model Development*, *11*(8). https://doi.org/10.5194/gmd-11-3481-2018

Hossain, F., Alwash, A., Minocha, S., and Eldardiry, H. (2023). Restoring the Mesopotamian Rivers for Future Generations: A Practical Approach. *Water Resources Research*, *59*(5). https://doi.org/10.1029/2023WR034514

Huffman G., Bolvin D., Braithwaite D., Hsu K., Joyce R., & Xie P. (2014). Integrated Multi-satellitE Retrievals for GPM (IMERG), version 4.4. In *NASA's Precipitation Processing Center*.

Jackson, R. B., Carpenter, S. R., Dahm, C. N., McKnight, D. M., Naiman, R. J., Postel, S. L., & Running, S. W. (2001). Water in a changing world. *Ecological applications, 11(4), 1027-1045.*

Johnson, B. M., Saito, L., Anderson, M. A., Weiss, P., Andre, M., & Fontane, D. G. (2004). Effects of climate and dam operations on reservoir thermal structure. *Journal of Water Resources Planning and Management, 130(2), 112-122.*

Lehner, B., Liermann, C. R., Revenga, C., Vörösmarty, C., Fekete, B., Crouzet, P., Döll, P., Endejan, M., Frenken, K., and Magome, J. (2011). High-resolution mapping of the world's reservoirs and dams for sustainable river-flow management. *Frontiers in Ecology and the Environment, 9(9), 494–502.*

Minocha, S., Hossain, F., Das, P., Suresh, S., Khan, S., Darkwah, G., Lee, H., Galelli, S., Andreadis, K., & Oddo, P. (2023). Reservoir Assessment Tool Version 3.0: A Scalable and User-Friendly Software Platform to Mobilize the Global Water Management Community [Preprint]. *Hydrology*. https://doi.org/10.5194/gmd-2023-130

Owusu, C., Snigdha, N. J., Martin, M. T., & Kalyanapu, A. J. (2022). PyGEE-SWToolbox: A Python Jupyter Notebook Toolbox for Interactive Surface Water Mapping and Analysis Using Google Earth Engine. *Sustainability, 14(5), 2557.*

Penman, H. L. (1948). Natural evaporation from open water, bare soil and grass. Proceedings of the Royal Society of London. Series A. *Mathematical and Physical Sciences, 193(1032), 120–145.* https://doi.org/10.1098/rspa.1948.0037

Suresh, S., Hossain F., Minocha S., Das P., Khan S., Lee H., Andreadis K. and Oddo P.(2023). Satellite-based Tracking of Reservoir Operations for Flood Management during the 2018 Extreme Weather Event in Kerala, India, *Remote Sensing of the Environment* (In revision).