SOFTWARE TOOL ARTICLE

lakemorpho: Calculating lake morphometry metrics in R [version 1; referees: awaiting peer review]

Jeffrey Hollister¹, Joseph Stachelek²

¹US Environmental Protection Agency, Office of Research and Development, National Health and Environmental Effects Research Laboratory, Atlantic Ecology Division, Narragansett, RI, USA
²Michigan State University, Department of Fisheries and Wildlife, Natural Resources Building, East Lansing, MI, USA

Abstract
Metrics describing the shape and size of lakes, known as lake morphometry metrics, are important for any limnological study. In cases where a lake has long been the subject of study these data are often already collected and are openly available. Many other lakes have these data collected, but access is challenging as it is often stored on individual computers (or worse, in filing cabinets) and is available only to the primary investigators. The vast majority of lakes fall into a third category in which the data are not available. This makes broad scale modelling of lake ecology a challenge as some of the key information about in-lake processes are unavailable. While this valuable in situ information may be difficult to obtain, several national datasets exist that may be used to model and estimate lake morphometry. In particular, digital elevation models and hydrography have been shown to be predictive of several lake morphometry metrics. The R package lakemorpho has been developed to utilize these data and estimate the following morphometry metrics: surface area, shoreline length, major axis length, minor axis length, major and minor axis length ratio, shoreline development, maximum depth, mean depth, volume, maximum lake length, mean lake width, maximum lake width, and fetch. In this software tool article we describe the motivation behind developing lakemorpho, discuss the implementation in R, and describe the use of lakemorpho with an example of a typical use case.

This article is included in the RPackage gateway.
Introduction

The study and quantification of lake shape (i.e. lake morphology and morphometry) is one of the foundations of limnology, and for students of limnology, some of the first lessons are centered around a typical suite of metrics and how to calculate them. It is also widely accepted that the morphometry of lakes and ponds can impact available nutrients and thus overall productivity. For instance, the widely used Vollenweider input-output models that are used to estimate nutrient concentrations rely on hydraulic residence time and sometimes mean depth, both of which are derived from total lake volume\(^4\). Also, clear water versus turbid water states in lakes have been linked in part to lake morphology, in particular mean depth\(^5\). In short, limnologists have long recognized the importance of lake morphology as one factor controlling a variety of ecological processes in lakes.

Traditional methods for calculating lake morphometry metrics have relied upon the use of paper bathymetry maps, planimeters, or simple heuristics\(^2\). In addition, detailed bathymetry is a requirement for the calculation of most lake morphometry metrics, but is generally only available for a relatively small number of lakes. Although this is not a problem when the focus of a study is a single lake, a small number of lakes, or a group of well-studied lakes, reliance on complete bathymetry becomes a limitation when attempting to conduct regional or national-scale lake studies. For instance, Soranno et al. found that for some water quality datasets lake depth, in spite of its importance, was not always available\(^6\). In cases such as these, alternative approaches for estimating lake morphometry are required.

Recent work has demonstrated the ability to estimate many of these metrics from ubiquitous spatial data\(^7\). For instance, maximum depth and lake volume may be predicted using the lake polygon and surrounding topography provided by the National Hydrography Dataset Plus and the National Elevation Dataset, respectively. These methods were initially developed with proprietary tools, thus limiting their accessibility. In an effort to reach a broader audience, the tools were converted to R, expanded to include a more complete suite of lake morphometry metrics and compiled into an R Package.

Methods

Implementation in R

Using R as a Geographic Information System is now possible, as several packages provide spatial data handling, geospatial analysis, and visualization. It is because of these packages that lakemorpho was implemented as an R package\(^6\). In particular, lakemorpho relies on the following packages: maptools, rgdal, raster, rgeos, sp, and geosphere\(^17\). In addition to these packages, two external libraries, the Geospatial Data Abstraction Library (GDAL) and Geometry Engine, Open Source (GEOS), are needed. Their availability to R and installation varies by operating system\(^18\).

Using lakemorpho

The lakemorpho package includes one function to create a lakeMorpho object, several functions to calculate morphometry metrics, a default plotting function, two example datasets, and an example lakeMorpho object.

A typical workflow for using lakemorpho to calculate lake metrics would include pulling spatial data into R (e.g. as shapefiles, tiff, etc.), creating a lakeMorpho object and calculating the desired lake morphometry metrics. The following sections provide details on the type of input data required and demonstrate use of the available functions with the provided example data.

The lakeMorpho Class and lakeSurroundTopo function. Many of the lake morphometry metrics rely on the same information about the lake. For instance, the functions to estimate maximum depth, mean depth, and volume rely on statistical summaries of the surrounding topography as well as the maximum in-lake distance to shoreline\(^19\). To avoid recalculating these values, a lakeMorpho object was created to link information on surrounding topography to the original datasets and facilitate default plotting of the outputs. All lake morphometry functions in the lakemorpho package require an object of class lakeMorpho as input. Some functions also return an updated lakeMorpho object that includes calculated spatial objects as output. At a minimum, a lakeMorpho object contains (see Figure 1):

- "lake" - A SpatialPolygons or SpatialPolygonsDataFrame object of the original input lake data.
- "elev" - A RasterLayer representing the elevation in a suitably large area around the lake.
- "surround" - A SpatialPolygons or SpatialPolygonsDataFrame object representing the land area defined as the surrounding topography.
- "lakeDistance" - A RasterLayer object of the euclidean distance from the shoreline to center of each pixel. Maximum value is equal to the maximum in-lake distance.
- "lakeOnEdge" - A logical value indicating if the "surround" polygon falls on the edge of the "elev" raster (i.e. would contain missing (i.e. NA) elevation data).
The `lakeSurroundTopo` function is the primary mechanism for creating a `lakeMorpho` object. There are two required inputs and one optional input for `lakeSurroundTopo`. The first required input is a `SpatialPolygons` or `SpatialPolygonsDataFrame` of the lake. Only a single lake is accepted as input, although this lake may be composed of multiple polygons (i.e. a lake with islands). If metrics for multiple lakes are required they will need to be passed to the suite of `lakemorpho` functions separately. The second required input is a `RasterLayer` of the elevation surrounding the lake. The default raster size is taken from the resolution of the input elevation data but may be specified separately. The third input specifies the area representing the surrounding topography. By default, this is a buffer of the lake shoreline, with the buffer width equal to the maximum in-lake distance. An optional `SpatialPolygons` object of any polygon intersecting the lake (e.g. catchments) can be used to define the surrounding topography instead of the default buffer. An object of class `lakeMorpho` is returned from `lakeSurroundTopo`.

In addition to providing the required inputs, users should pay attention to both the extent of the input elevation dataset as well as the coordinate reference systems being used. First, the elevation data must be of a large enough extent so that the surrounding topography does not include land area outside that extent (i.e. would return NA values). As noted above, the `lakeOnEdge` item indicates if the surrounding topography is on the edge of the user supplied elevation and thus would return some missing data. Second, all of the functions of `lakemorpho` assume that projections have been handled prior to creating the `lakemorpho` class or calculating the metrics. If the input data are not of the same projection, `lakeSurroundTopo` will return an error. The data must be re-projected into the same coordinate reference system (CRS). The units of all metrics are determined by the CRS and care must be taken to make sure that the vertical units of the elevation are the same as horizontal units of the projection. For instance, elevation data may be available in meters yet the CRS is specified in feet. In cases such as these, a conversion of the vertical data should be done. Lastly, care must be taken in choosing an appropriate CRS for the area under consideration. This is because all CRS will distort area, distance, shape, or direction. Thus a projection that minimizes distortions of distance and area are preferable. A useful reference for further exploring coordinate reference system is Iliffe and Lott’s 2008 book on the topic.

Usage of `lakeSurroundTopo` and generation of a `lakeMorpho` object from the example data included with `lakemorpho` is done as follows:

```r
# Load data
data(lakes)

# Create lakeMorpho object, example_lakeMorpho, with required inputs
eexample_lakeMorpho <- lakeSurroundTopo(exampleLake, exampleElev)
```

The resulting object contains the minimum set of components that make up a `lakeMorpho` object. We can verify that the components are of the expected class with the following command:

```r
lapply(example_lakeMorpho, class)
```

## $lake
## [1] "SpatialPolygonsDataFrame"
## attr(,"package")
Lake Morphometry Functions. Each of the remaining functions expects a `lakeMorpho` object as input and returns a numeric value. Some of the functions also have a side effect of adding a spatial object to the input `lakeMorpho` object.

`calcLakeMetrics`  
The `calcLakeMetrics` function is a convenience function that will calculate all of the `lakemorpho` metrics for a single `lakeMorpho` object. It requires an input `lakeMorpho` object, a `bearing` for calculating `lakeFetch`, and `pointDens` for maximum lake length and width (defined below).

```
calcLakeMetrics(example_lakeMorpho, 0, 250)
```

```
## $surfaceArea
## [1] 16453180
##
## $shorelineLength
## [1] 45991.38
##
## $shorelineDevelopment
## [1] 3.198502
##
## $maxDepth
## [1] 99.17621
##
## $volume
## [1] 4802535
##
## $meanDepth
## [1] 28.94864
##
## $maxLength
## [1] 9479.313
##
## $maxWidth
## [1] 3166.225
##
## $meanWidth
## [1] 1735.693
##
## $fetch
## [1] 6336.798
```
湖面取值
湖面取值是给定方向的最大开水平距离，并且可以作为一种混合指标来使用，因为较大的湖面取值意味着由于风影响而形成的更大的波浪可能性。`lakeFetch`函数计算输入方向的湖面取值。输入方向可以是0到360度的任何值，0和360度都代表北，而相反方向（例如东和西）的湖面取值是相同的。

要计算输入湖泊的湖面取值，使用以下代码：

```r
# Fetch for north
lakeFetch(example_lakeMorpho, 0)
## [1] 6336.798

lakeFetch(example_lakeMorpho, 360)
## [1] 6336.798

# Fetch for west
lakeFetch(example_lakeMorpho, 270)
## [1] 3129.997
```

湖面主轴长度
湖面主轴长度定义为穿过湖泊多边形的凸包的最长线，而其值代表湖泊的跨度，而与陆-水配置无关。相比之下，`lakeMaxLength`函数的值代表湖泊的跨度，而与陆-水配置无关。

要计算输入湖泊的湖面主轴长度，使用以下代码：

```r
lakeMajorAxisLength(example_lakeMorpho, addLine = TRUE)
## [1] 13159.64
```

湖面最大深度
湖面最大深度提供了可以用于估算湖泊停留时间的信息，连同流量率。当没有替代的现场测量时，湖泊的最大深度可能可以由湖泊周围的地形估算。`lakeMaxDepth`函数以Hollister等人提出的模型为基础，通过指定一个已知的深度校正因子，可以提供湖泊的深度估计。它只需要一个`lakeMorpho`对象作为输入。

要计算湖泊的最大深度，使用以下代码：

```r
# Maximum lake depth
lakeMaxDepth(example_lakeMorpho)
## [1] 99.17621
```

由于各种因素，这些最大深度预测的准确性在不同地区可能会有所不同，通常需要使用现场测量数据进行校正。例如，Hollister等人11证明，对于新英格兰和大西洋沿岸的美国东海岸地区，初步预测值大于真实值，并且需要降低。

湖面最大长度
湖面最大长度是湖泊内部的最大开水平距离，类似于湖面取值，是一个可以用于估算混合潜力的指标12。当前实现中，`lakemorpho`将等距离的点放置在湖泊的沿岸，然后找到不相交于陆地（例如半岛或岛屿）的点对点的最远距离。可选参数`addLine`具有默认值TRUE，并允许将`SpatialLines`对象存储在输入的`lakeMorpho`对象中（图2）。
To calculate maximum lake length use:

```r
#Max length with a point density of 250
dlakeMaxLength(example_lakeMorpho, 250, addLine = FALSE)
```

```r
## [1] 9025.769
```

The `pointDens` parameter can have an impact on both the processing time and the resulting value and both of these can vary as a function of the complexity of the shape of the lake with less complex lakes providing more consistent lake length across a number of points. Given this caveat, care must be taken in choosing an appropriate number of points (and thus lines) to use to calculate maximum lake length. Several densities should be tested and the smallest number of points that produce a stable estimate should be used.

`lakeMaxWidth`

Maximum lake width is the maximum shore to shore distance that is perpendicular to the line representing maximum lake length and is another metric related to mixing.\(^{1,28}\) The `lakeMaxWidth` function requires a `lakeMorpho` object and `pointDens` value which is used to determine the number of points along the maximum lake length line. The issue with `pointDens`, discussed above, also exists for the use of `pointDens` with `lakeMaxWidth` and care should be taken to determine an appropriate number of lines to test.

Usage of `lakeMaxWidth` is:

```r
#Max width with a point density of 250
dlakeMaxWidth(example_lakeMorpho, 250)
```

```r
## [1] 3177.625
```

`lakeMeanDepth`

Mean depth of a lake is calculated as the volume of the lake divided by the area.\(^{1,28}\) This function requires only a `lakeMorpho` object and returns a numeric value of the mean depth. Usage of the function is:

```r
dlakeMeanDepth(example_lakeMorpho)
```

```r
## [1] 28.94864
```

There is an optional `zmax` argument that allows a user to specify a maximum lake depth if one is available. If not supplied, the maximum depth will be estimated using `lakeMaxDepth`. For instance, in the above example, the maximum depth without using a correction factor is estimated at 99 meters which results in a mean depth estimate of 28.95. The
measured maximum depth, 32 meters, is much less than the estimate depth. To use this information you would simply add the measured valued in for the zmax argument.

```r
lakeMeanDepth(inputLM, zmax = 32)
```

## [1] 9.340511

lakeMeanWidth

The mean width of a lake is defined as lake area divided by maximum lake length\(^1\). Input for this function is a lakeMorpho object that has the maximum lake length line added via ‘lakeMaxLength’. This requirement is checked and returns an error if the maximum length line is missing.

```r
# Add the maximum lake length line
lakeMaxLength(example_lakeMorpho, pointDens = 100, addLine = TRUE)
```

## [1] 8194.247

```r
# Calculate mean width
lakeMeanWidth(example_lakeMorpho)
```


lakeMinorAxisLength

The minor axis of a lake is defined as the shortest line intersecting the convex hull formed around the lake polygon while passing through its center. In contrast to lakeMaxWidth, its value represents the distance across a lake with regard to the the convex hull and without consideration of the land-water configuration.

```r
lakeMinorAxisLength(example_lakeMorpho, addLine = TRUE)
```

## [1] 6926.263

lakeMinorMajorRatio

The ratio of the lake major axis length to the minor axis length is also known as the aspect ratio. Circular lakes have aspect ratios approaching 1 while thin-elongated lakes have aspect ratios approaching 0. If major and minor axis length have not already been added to the lakeMorpho object, these are calculated. The addLine argument adds the lines for the lake’s minor and major axes to the lakeMorpho object.

```r
lakeMinorMajorRatio(example_lakeMorpho, addLine = TRUE)
```

## [1] 0.5263261

lakeShorelineDevelopment

The shoreline development metric provides a measure of the complexity of the shoreline. It is a ratio of the perimeter of the lake to the perimeter of a circle of the same area. Values will be 1 or greater with value of 1 indicating a circular lake. This metric is used as an indicator of potential habitat\(^1\). It only requires a lakeMorpho object as input.

```r
lakeShorelineDevelopment(example_lakeMorpho)
```

## [1] 3.198502

lakeShorelineLength and lakeSurfaceArea

Shoreline length is simply the total perimeter of the lake polygon and, as with all other functions, requires a lakeMorpho object as input. To calculate the shoreline length:

```r
lakeShorelineLength(example_lakeMorpho)
```

## [1] 45991.38
Similarly, surface area for a lake is the total area of the lake polygon. It is calculated via:

```r
lakeSurfaceArea(example_lakeMorpho)
```

## [1] 16453180

**lakeVolume**

The `lakeVolume` function uses maximum lake depth (see `lakeMaxDepth`) and methods outlined by Hollister et al.\(^9\) to estimate lake volume. The method assumes that the maximum in-lake distance \((D_{\text{max}})\) from the shoreline is also the deepest part of the lake \((Z_{\text{max}})\). The `lakeVolume` function creates a raster of the in-lake distance to shoreline and converts those distances, using \(Z_{\text{max}}:D_{\text{max}}\), to depths and then summing the volume of each pixel to estimate total lake volume.

```r
lakeVolume(example_lakeMorpho)
```

## [1] 476297184

Similar to `lakeMeanDepth`, there is a `zmax` argument to be used for a known maximum lake depth.

**Use case**

A common application of `lakemorpho` is to calculate the full suite of lake metrics for multiple lakes. This use case demonstrates how to do that with a commonly encountered GIS data file, the shapefile. To do this we iterate through the lakes, calculate metrics for each lake and include the metrics on an output shapefile. The data for this use case is from Rhode Island (Figure 3). The data for the lakes were downloaded from the Rhode Island Geographic Information Systems (RIGIS)\(^29\) and the elevation data are from Amazon Web Services Terrain Tiles via the `elevatr` package\(^30\).

![Figure 3. Map of lakes used in the case study example of the `lakemorpho` package.](image-url)
Read in data
This use case relies on the sp and rgdal packages for the spatial data handling. These are dependencies for lakemorpho, thus no additional installs are required. To read in the data we utilized rgdal::readOGR and read in the ri_lakes.shp from the current directory. This file is available for download from https://github.com/USEPA/lakemorpho_manuscript/blob/master/ri_lakes.zip.

```r
# Load packages
library("sp")
library("rgdal")
library("lakemorpho")

# Read the shapefile and plot
ri_lakes <- readOGR(".", "ri_lakes")
```

## OGR data source with driver: ESRI Shapefile
## Source: ".", layer: "ri_lakes"
## with 5 features
## It has 2 fields

Iterate through lakes and calculate metrics
In R, there are many ways to iterate. For simplicity and clarity we use a for loop to iterate through all lakes and calculate the full suite of lake metrics with calcLakeMetrics. We will utilize the elevatr package which provides access to elevation data from various sources. In this example we will use the Amazon Web Services terrain tiles. The vertical elevation data are in meters and the Rhode Island lake data are projected in Rhode Island State Plane Feet, thus we will convert the elevation data into feet.

```r
library(elevatr)
output <- data.frame()
for (i in seq_along(ri_lakes)){
  dem <- get_elev_raster(ri_lakes[i,], z = 12, expand = 1000, src = "aws") * 3.281
  lmorph <- lakeSurroundTopo(inLake = ri_lakes[i,], inElev = dem)
  lmetric <- calcLakeMetrics(lmorph, bearing = 270, pointDens = 100)
  output <- rbind(output,
                   data.frame(NAME = ri_lakes[i]$NAME,
                               data.frame(lmetric)))
}
```

We can now merge the morphometry metrics back to the lake polygons.

```r
ri_lakes_m <- merge(ri_lakes, output, by="NAME")
dplyr::tbl_df(ri_lakes_m)
```

## A tibble: 5 x 12
## * NAME   Acres surfaceArea shorelineLength 
## <fctr> <dbl>       <dbl>           <dbl> 
## 1 Indian Lake 268.55 11698076        18857.94 
## 2 Yawgoo Pond 144.37 6288693        10562.24 
## 3 Worden Pond 1098.64 47856596        34494.32 
## 4 Larkin Pond 43.85 1910010        5989.30 
## 5 Hundred Acre Pond 87.64 3817524       15278.75 

## Conclusions
The lakemorpho package provides functions to calculate common lake morphometry metrics in R. For those conducting lake analyses in R this allows for streamlined analysis workflows. Also, lakemorpho provides a foundation for additional metrics. For instance, it might be possible to combine hydrological methods for estimating stream flow into and out of lakes with lake volume and add a function for calculating residence time.
Beyond adding additional metrics, more fundamental rewriting of the package may also be useful. For instance, lake-morpho currently is built on top of the current spatial data standard for R, sp. This allows a clean interface with many existing tools; however, it is likely that sp will be replaced in the next several years by the sf package\textsuperscript{21,31}. Future versions of lakemorpho might benefit from using the sf tool chain and the “tidy data” framework\textsuperscript{32}.

In summary, lakemorpho provides limnologists and aquatic ecologists with a consistent framework in R for calculating a suite of the most common lake morphometry metrics. This paper outlines the currently available functions and provides an example through a typical use case of calculating many metrics for several lakes.

**Software availability**

The lakemorpho version 1.1.0 package is currently available directly from the Comprehensive R Archive Network (CRAN) and may simply be installed and loaded in R via:

```r
install.packages('lakemorpho')
library('lakemorpho')
```

To access the help pages (including a version of this manuscript) use:

```r
help(package='lakemorpho')
```

There are tentative plans to continue developing new functions for lakemorpho and these new features will be available first through the development version on GitHub at http://github.com/usepa/lakemorpho.

To install and load the development version requires use of the devtools package. This may be done with:

```r
install.packages('devtools')
library('devtools')
install_github('USEPA/lakemorpho')
library(lakemorpho)
```

Archived source code as at the time of publication: http://doi.org/10.5281/zenodo.863051\textsuperscript{33}

---

**Competing interests**

No competing interests were disclosed.

**Grant information**

The author(s) declared that no grants were involved in supporting this work.

**Acknowledgements**

We would like to thank Stephen Shivers (ORISE), Betty Kreakie (US EPA, Atlantic Ecology Division), Bryan Misteck (US EPA, Atlantic Ecology Division), Joe LiVolsi (US EPA, Atlantic Ecology Division), Tim Gleason (US EPA, Atlantic Ecology Division), and Wayne Munns (US EPA, Atlantic Ecology Division) for constructive reviews of this paper. Special thanks to TBD for reviews of the submitted manuscript. The views expressed in this article are those of the authors and do not necessarily represent the views or policies of the U.S. Environmental Protection Agency. Any mention of trade names, products, or services does not imply an endorsement by the U.S. Government or the U.S. Environmental Protection Agency. The EPA does not endorse any commercial products, services, or enterprises. This contribution is identified by the tracking number ORD-022603 of the Atlantic Ecology Division, Office of Research and Development, National Health and Environmental Effects Research Laboratory, US Environmental Protection Agency.
References

28. LAKEWATCH F: Department of fisheries and aquatic sciences, a beginner’s guide to water management-lake morphometry. 2001. Reference Source
29. (RIGIS) RIGIS: Lakes and ponds (1: 5000); lakes5k10. 2010. Reference Source