## CONTENTS

1 About 3  
1.1 Technologies ................................................................. 3  

2 Command Line Interface 5  
2.1 extract ................................................................. 5  
2.2 shatter ............................................................... 5  
2.3 info ................................................................. 6  
2.4 initialize ............................................................. 6  

3 API 7  

4 QuickStart 9  
4.1 Installation ............................................................ 9  
4.2 Initialization ............................................................ 10  
4.3 Shatter ................................................................. 11  
4.4 Extract ................................................................. 11  

5 Development 13  

6 Indices and tables 15  

Index 17
SilviMetric is an open source library and set of utilities from Hobu, Inc. that are useful for summarizing point cloud data into raster and raster-like products.

Find out more about SilviMetric by visiting About. A slide deck about SilviMetric is also available on Google Slides.
Summarizing and filtering point cloud data into useful information for modeling is challenging. In forestry applications in particular, the FUSION software toolkit is often used to extract information in preparation for modeling. FUSION, however, has a few missing features that make it

Working with Bob McGaughey and the USFS GTAC team, Kyle Mann and Howard Butler from Hobu, Inc., developed the initial prototype of SilviMetric to implement an alternative approach to computing the “GridMetrics” component of typical FUSION processing pipelines.

SilviMetric does this by breaking apart the computation of metrics into three distinct steps – info, shatter, and extract. SilviMetric takes an infrastructure computing approach to the challenge by applying emerging open source technologies that speak cloud, are nimble with data formats, and compute in a more friendly language - Python.

1.1 Technologies

SilviMetric stands on the shoulders of giants to provide an integrated solution to computing rasterized point cloud metrics. These technologies include:

- **PDAL** reads point cloud content and allows users to filter or process data as it ingested.
- **Dask** processes tasks for ‘ref shatter’ and extract in a highly parallel, cloud-friendly distributed computing environment.
- **TileDB** stores metrics in cloud object stores such as S3 in addition to typical filesystems.
- **Python** computes metrics and provides a diverse and convenient computing capability for users to easily add and extract their own metrics to the database.
2.1 extract

2.1.1 Synopsis

Usage: silvimetric DATABASE extract [OPTIONS]

Extract silvimetric metrics from DATABASE

Options:
-a, --attributes ATTRS List of attributes to include in Database
-m, --metrics METRICS List of metrics to include in Database
--bounds BOUNDS
-o, --outdir TEXT [required]
--help Show this message and exit.

2.2 shatter

2.2.1 Synopsis

Usage: silvimetric DATABASE shatter [OPTIONS] POINTCLOUD

Insert data provided by POINTCLOUD into the silvimetric DATABASE

Options:
--workers INTEGER
--bounds BOUNDS
--tilesize INTEGER
--threads INTEGER
--watch
--dasktype [cluster|threads|processes|single-threaded]
--help Show this message and exit.
2.3 info

2.3.1 Synopsis

Usage: silvimetric DATABASE info [OPTIONS]

Print info about Silvimetric database

Options:
   --history
   --help    Show this message and exit.

2.4 initialize

The initialize subcommand constructs the basic TileDB instance to host the SilviMetric data. It can be either a local filesystem path or a S3 URI (e.g., `s3://silvimetric/mydata`).

2.4.1 Synopsis

Usage: silvimetric DATABASE initialize [OPTIONS] BOUNDS CRS

Initialize silvimetrics DATABASE

Options:
   -a, --attributes ATTRS  List of attributes to include in Database
   -m, --metrics METRICS   List of metrics to include in Database
   --resolution FLOAT      Summary pixel resolution
   --help                  Show this message and exit.
SilviMetric depends upon Conda for packaging support. You must first install all of SilviMetric’s dependencies using Conda:

This tutorial shows you how to initialize, shatter, and extract data in SilviMetric using the Command Line Interface. We are going to use the Autzen Stadium as our test example.

**Note:** The Autzen Stadium has units in feet, and this can sometimes be a source of confusion for tile settings and such.

### 4.1 Installation

Open a Conda terminal and install necessary dependencies

```
conda env create
  ...environment.yml
  -n silvimetric
```

**Note:** We are installing the list of dependencies as provided by the SilviMetric GitHub listing over the internet.

**Warning:** If you are using windows, line continuation characters are ^ instead of \

2. Activate the environment:

```
conda activate silvimetric
```

3. Install SilviMetric:

```
pip install silvimetric
```
4.2 Initialization

Initialize a SilviMetric database. To initialize a SilviMetric database, we need a bounds and a coordinate reference system.

1. We first need to determine a bounds for our database. In our case, we are going to use PDAL and jq to grab our bounds.

   ```
   pdal info https://s3.amazonaws.com/hobu-lidar/autzen-classified.copc.laz \ 
       --readers.copc.resolution=10 | jq -c '.stats.bbox.native.bbox'
   ```

   Our boundary is emitted in expanded form.

   ```
   {"maxx":639003.73,"maxy":853536.21,"maxz":615.26,"minx":635579.2,"miny":848884.83,"minz":406.46}
   ```

   **Note:** You can express bounds in two additional formats for SilviMetric:

   • `[635579.2, 848884.83, 639003.73, 853536.21]` – `[minx, miny, maxx, maxy]`
   • `([635579.2,848884.83],[639003.73,853536.2])` – `([minx, maxx], [miny, maxy])`

   **Note:** You can install jq by issuing `conda install jq -y` in your environment if you are on Linux or Mac. On Windows, you will need to download jq from the website and put it in your path. [https://jqlang.github.io/jq/download/](https://jqlang.github.io/jq/download/)

2. We need a coordinate reference system for the database. We will grab it from the PDAL metadata just like we did for the bounds.

   ```
   pdal info --metadata https://s3.amazonaws.com/hobu-lidar/autzen-classified.copc.laz \ 
       --readers.copc.resolution=10 | \ 
       jq -c '.metadata.srs.json.components[0].id.code'
   ```

   Our EPSG code is in the `pdal info --metadata` output, and after extracted by jq, we can use it.

   2992

   **Note:** Both a bounds and CRS must be set to initialize a database. We can set them to whatever we want, but any data we are inserting into the database must match the coordinate system of the SilviMetric database.

3. With bounds and CRS in hand, we can now initialize the database.

   ```
   silvimetric autzen-smdb.tdb \ 
       initialize \ 
       '{"maxx":639003.73,"maxy":853536.21,"maxz":615.26,"minx":635579.2,"miny":848884.83,"minz":406.46}" \ 
       EPSG:2992
   ```

   **Note:** Be careful with your shell’s quote escaping rules!
4.3 Shatter

We can now insert data into the SMDB

```
silvimetric autzen-smdb.tdb \
  shatter \ 
  https://s3.amazonaws.com/hobu-lidar/autzen-classified.copc.laz \ 
  --threads 10 \ 
  --watch
```

4.4 Extract

After data is inserted, we can extract

```
silvimetric autzen-smdb.tdb extract -o output-directory
```
CHAPTER FIVE

DEVELOPMENT

SilviMetric is released under the Apache 2.0 License.
SilviMetric is managed and developed on GitHub at https://github.com/hobuinc/silvimetric
CHAPTER
SIX

INDICES AND TABLES

• genindex
• modindex
• search
INDEX

E
    extract, 5

I
    info, 6
    initialize, 6

S
    shatter, 5