Parrish Lab
Oregon State University
Civil and Construction Engineering

Authors: Forrest Corcoran
and Keana Kief

The comprehensive Bathymetric Lidar Uncertainty Estimator (cBLUE)

Revision History

Revision	Date	Author(s)	Description
1.0	June 13, 2022	F.C.	First Draft.
1.1	October 31, 2023	K.K.	Second Draft.
1.2	June 4, 2024	K.K.	Third Draft.

Contents

1 cBLUE Overview

1.1 Statement of Purpose

The comprehensive Bathymetric Lidar Uncertainty Estimator (cBLUE) is a software tool produced and maintained by the Parrish Lab at Oregon State University, College of Engineering, School of Civil and Construction Engineering, Geomatics Group in collaboration with NOAA's National Geodetic Survey and the University of New Hampshire Center for Coastal and Ocean Mapping. cBLUE is designed to provide Total Vertical Uncertainty (TVU) and Total Horizontal Uncertainty (THU) for bathymetric lidar surveys.

1.2 License

cBLUE V3.2

2 Installation

2.1 Installating cBLUE

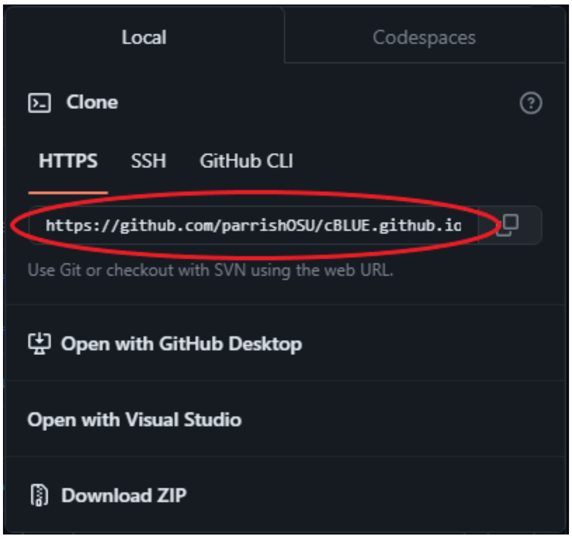
This section provides step-by-step instructions to download cBLUE and install the various `Python` dependencies need to calculate TPU from `.las` files. While `git` is the suggested method for downloading cBLUE, users unfamiliar with version control software are also free to download cBLUE as a `.zip` file. Instructions for both are provided below:

2.1.1 Download via `git`

2.1.1.1 Steps:

1. In a web browser, navigate to `https://github.com/parrishOSU/cBLUE.github.io`

2. In the top right of the repository, locate the green `code` menu and copy the url under `HTTPS`



3. Clone the repository using either your computer's terminal/command prompt or the Github Desktop Application:

2.1.1.2 Terminal

(a) Navigate to the folder where you'd like to install cBLUE

```
$ cd location/of/cBLUE
```

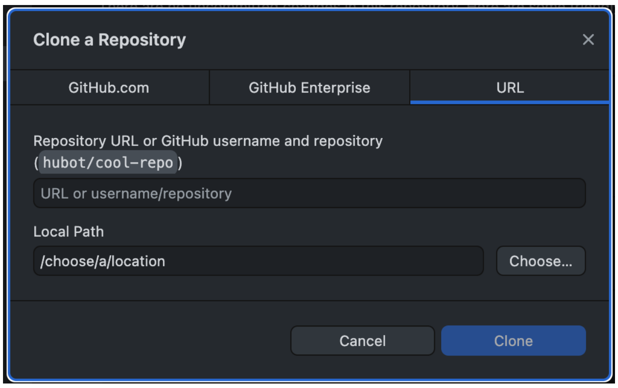
(b) Clone the repository using the url from github

```
$ git clone https://github.com/parrishOSU/cBLUE.github.io.git
```

2.1.1.3 GitHub Desktop

(a) Open GitHub Desktop and locate the "Add" button

(b) Under "Add" choose "Clone Repository" and select "URL"



(c) Enter the path to the location you'd like to install cBLUE and paste the url found in Step 2 in the appropriate boxes

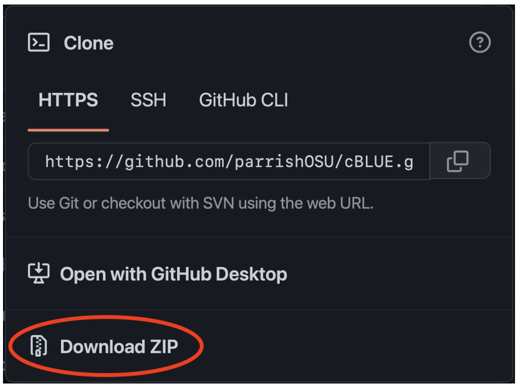
(d) Press the "Clone" button to begin download

4. Once the download is complete, check that the folder `cBLUE.github.io` is in the desired location.

2.1.2 Download as `.zip`

1. In a web browser, navigate to https://github.com/parrishOSU/cBLUE.github.io

2. In the top right of the repository, locate the green `code` menu and click "Download Zip"



3. Once the download is complete, move the `.zip` to your desired location and extract to `cBLUE.github.io`.

2.2 Install cBLUE Dependencies

cBLUE is designed to be a cross-platform software, therefore this installation guide should be valid for all Windows, MacOS, and Linux users. However, cBLUE does require Python 3 to be installed on the user's machine, as well as all the Python library dependencies needed to run cBLUE. Additionally, while it is possible to install cBLUE using any Python package manager, we strongly suggest using `conda`, which can be downloaded for free here. In this installation guide we provide instructions to install and run cBLUE using both the Anaconda Navigator GUI as well as the `conda` CLI.

2.2.1 conda

1. In your computer's terminal/command prompt, navigate to `cBLUE.github.io`
```
$ cd location/of/cBLUE/cBLUE.github.io
```

2. Locate the file `cblue.yml`

3. Create a new virtual environment using the command
```
$ conda env create -f cblue.yml
```

4. cBLUE dependencies should begin downloading shortly.

5. Once all dependencies
   have successfully downloaded, activate the new environment using the command

```
$ conda activate cblue
```

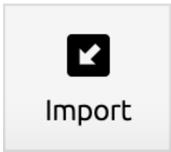
6. To check that the dependencies are installed properly, use the command

```
$ python CBlueApp.py
```

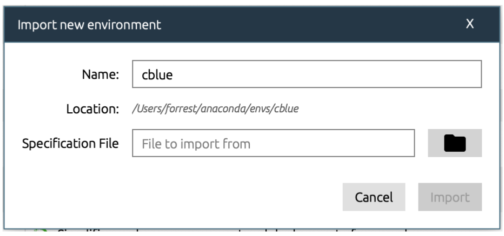
2.2.2 Anaconda Navigator

1. Open "Anaconda Navigator" and navigate to the "Environments" menu on the left hand side.

2. Locate the "Import" button on the bottom left



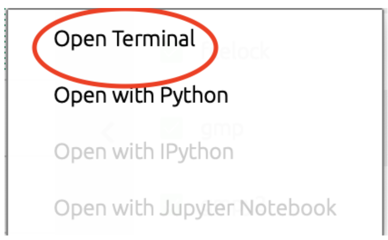
3. In the "Import New Environment" dialog box, enter "cblue" for "Name" and navigate to the file `cblue.yml` for the "Specification File"



4. Click "Import"

5. Once the downloads are complete, locate "cblue" in the environments list and click on it to activate. A green triangle should appear next to the environment name once it is activated.

6. To check that the dependencies are installed correctly, click the green triangle next "cblue" and select "Open Terminal"



7. Navigate to the location of `cBLUE.github.io`

```
$ cd location/of/cBLUE/cBLUE.github.io
```

8. Open cBLUE using the command
```
$ python CBlueApp.py
```

3 User Guide

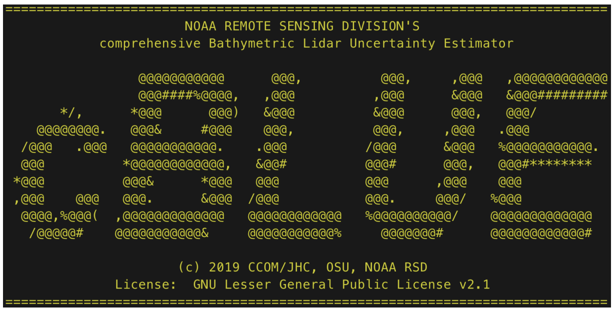
This section provides step-by-step instructions to perform a Total Propagated Uncertainty (TPU) estimation from bathymetric lidar data. In order to perform this estimation, it is assumed that the user has properly install cBLUE and its dependencies. If cBLUE or its dependencies are not installed, please refer to Section 2 of this manual.

In order to perform a TPU estimation using cBLUE, it is necessary to have two sets of files: at least one `.las` file containing information on the lidar return signal and at least one corresponding SBET (`.txt`) file containing information on the sensor trajectory at the time of each lidar pulse. These files are (usually) obtained directly from the sensor platform. Further information on these file types is outside the scope of this manual.

3.1 Graphical User Interface (GUI)

To open cBLUE, in the terminal of your choosing, execute one of the following commands from inside the `cBLUE.github.io` folder.
```
$ python CBlueApp.py
```
or
```
$ python CBlueAppGui.py
```

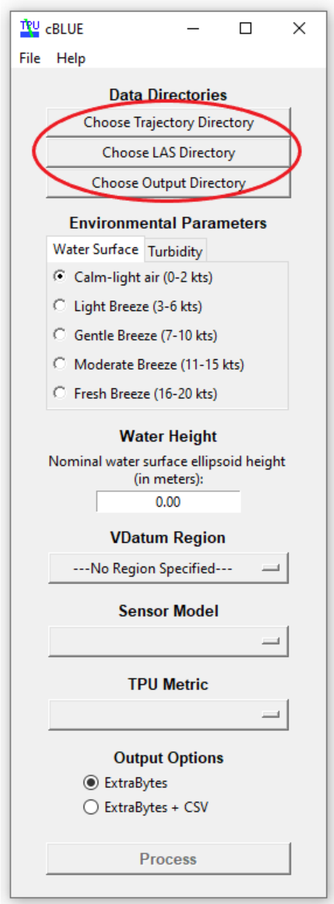
The following message should appear in your terminal:



(Note: the `cblue` conda environment must be installed and activated in order to run cBLUE. If cBLUE fails to open, it is likely that your dependencies are not installed or the environment is deactivated. Refer to Section 2 of this manual for information on resolving these issues.)

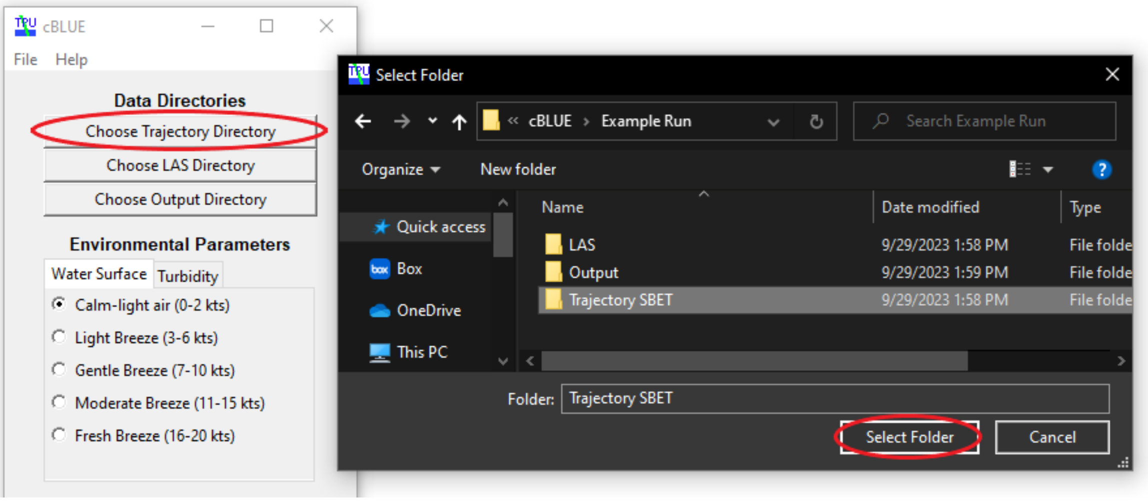
3.1.1 Data Directories

The first step in performing a TPU estimation using cBLUE is to point the software to the LAS (`.las`) and SBET (`.txt`) files on your system. This is done using the Data Directories selection menus.

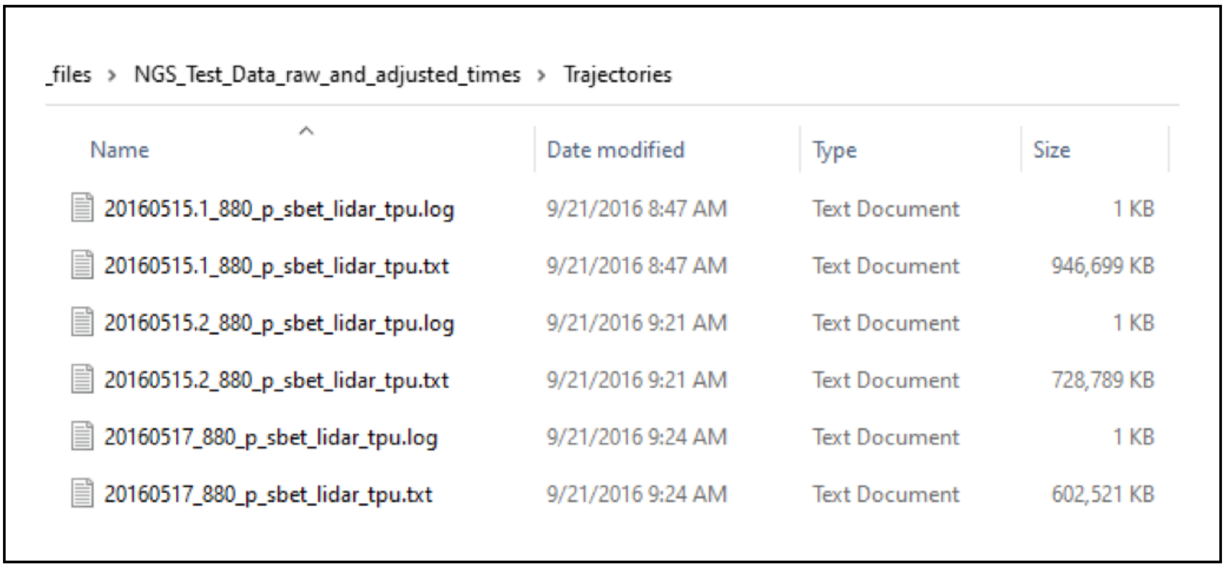


3.1.1.1 Trajectory

The "Trajectory Directory Set" button allows the user to select the location of the SBET files containing the sensor trajectory parameters at the time of each laser pulse.

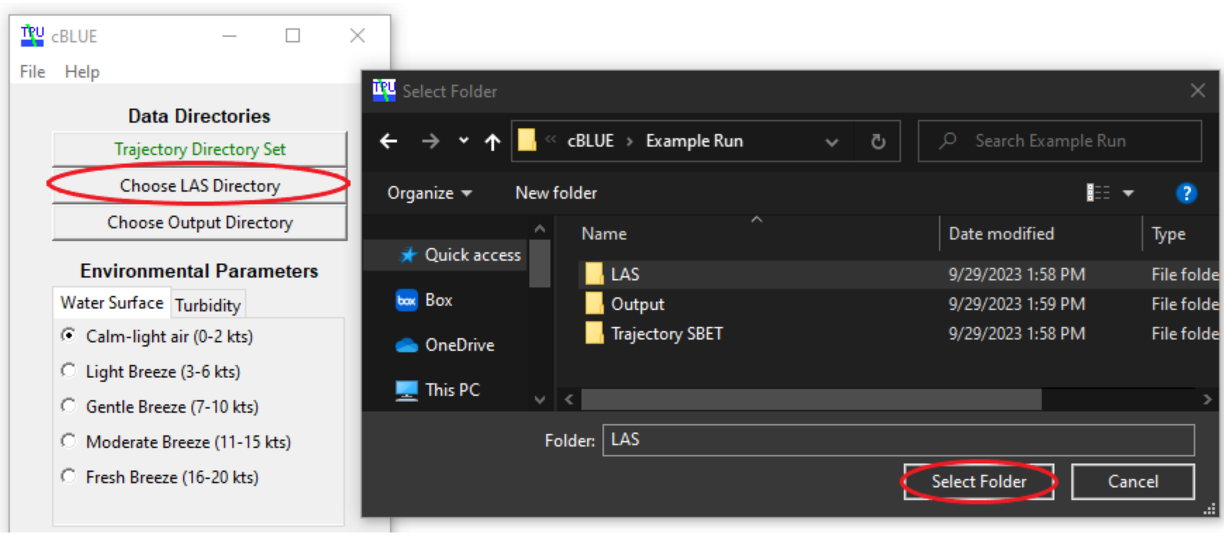


These files are usually generated by the lidar system and therefore their generation and file standards are outside the scope of this manual, however a brief overview of ASCII SBET formatting is availble in Appendix A. If you run into issue regarding the SBET files, please consult the lidar system's user manual or contact the manufacturer before raising an issue with the cBLUE maintenance team, as these issues are often the result of erroneous data and are therefore difficult to reproduce/troubleshoot. An example of SBET files is shown in the figure below.



3.1.1.2 LAS Files

The "LAS Directory Set" button allows the user to select the location of the `.las` files containing the point cloud coordinates from each laser pulse.



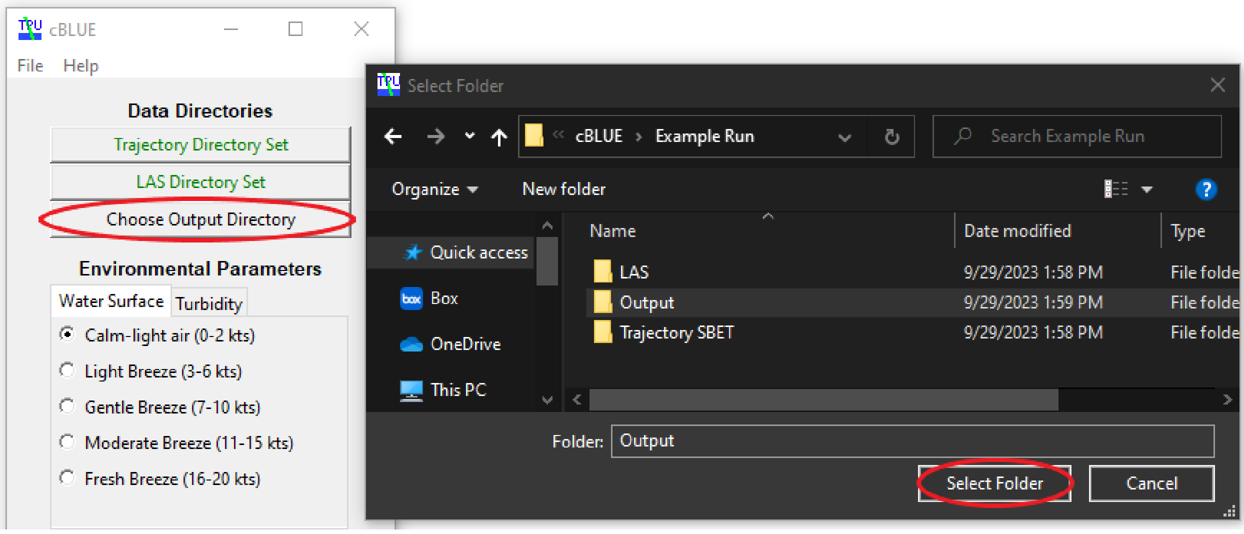
Currently, given the popularity of `.las`, especially in the bathymetric lidar community, these are the only files types supported. There are no plans to add further file types*. Additionally, note that the GPS times in the LAS file should be Adjusted Standard GPS Time. This is important because the GPS times are how cBLUE matches up points in the LAS file with corresponding records from the trajectory. For more information on LAS file specifications, please refer to the ASPRS LASER (LAS) FILE FORMAT EXCHANGE ACTIVITIES, found here.

Note: cBLUE expects the coordinate system of the x and y values from the input LAS file to be in UTM projection.

*As of cBLUE V3.0, compressed LAS files (`.laz`) are also accepted by cBLUE.

3.1.1.3 TPU Directory

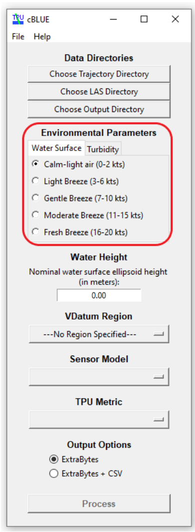
The "TPU Directory Set" button allows the user to select the location where the output files will be generated.



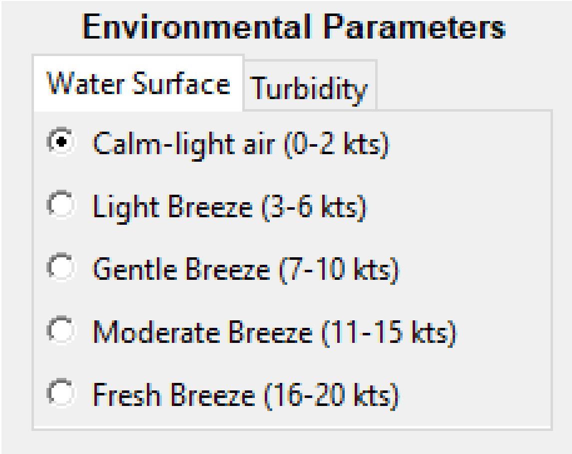
The files output by cBLUE will match the `.las` files in the "LAS Directory Set" location, with `_TPU` appended to the file name. These output files will contain the TPU as Extra Bytes fields. More information on LAS Extra Bytes can be found here.

3.1.2 Environmental Parameters

cBLUE operates by using a combined subaerial and subaqeuous model. The subaqeuous model is designed to simulate the range of possible values for each laser shot, given a set of environmental parameters. Therefore, it is important that the user tune these parameters to reflect the best approximation of the water conditions at the time the bathymetric survey was conducted.

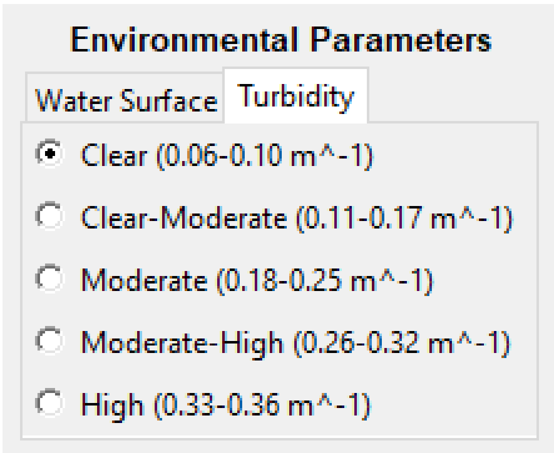


3.1.2.1 Water Surface The "Water Surface" menu allows the user to select the range of water surface parameters that best approximate the water surface conditions at the time of the bathymetric survey.



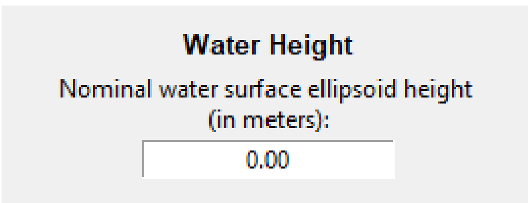
(Note: cBLUE V2.X included an option to model the water surface from using the pointcloud data. This option has been removed as of cBLUE V3.0. For more information on why this option was removed, please refer to Section 4 of this manual).

3.1.2.2 Turbidity The "Turbidity" menu allows the user to select the range of turbidity parameters that best approximate the water clarity conditions at the time of the bathymetric survey.



3.1.3 Water Surface Ellipsoid Height

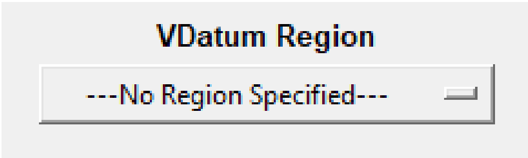
The next field asks the user to enter the ellipsoid height of the water surface as a float value in meters. The purpose of this parameter is to enable a slant range correction. This value should be close, but it does not need to be perfect, as changes in this parameter generally only impact the computed TVU on the order of millimeters.



The preferred method of obtaining this value is to use an average ellipsoid height of water surface (Class 41) points in the LAS file. An alternative method of obtaining an approximate value for this parameter is to use VDatum online to compute the separation between the NAD83 ellipsoid and one of the following tidal datums, depending on which was closest to the water level at the time the bathymetric lidar dataset was acquired: mean lower low water (MLLW), mean low water (MLW), mean tide level (MTL), mean high water (MHW), or mean higher high water (MHHW). If the stage of tide at the time the bathymetric lidar was acquired is unknown, it is usually reasonably safe to select local mean sea level (LMSL) as the “to” datum; typically, this will only impact the computed TPU values on the order of millimeters.

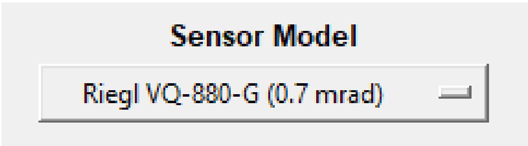
3.1.4 VDatum Region

Vertical datum uncertainty is a component uncertainty often included in bathymetric TPU modeling. The "VDatum Region" menu allows the user to select a VDatum region to use in the modeling process. The purpose of this setting is to enable users to include VDatum datum transformation uncertainty (`https://vdatum.noaa.gov/docs/est_uncertainties.html`) as a component uncertainty in the TPU computation. However, this parameter is not required to perform the uncertainty estimation and may be left as "–No Region Specified–" by the user.



3.1.5 Sensor Model

As of cBLUE V3.2, there are 8 lidar sensor platforms available in cBLUE. These include the Riegl VQ-880-G, Leica Chiroptera 4X, Leica Hawkeye 4X, Areté PILLS, Fugro RAMMS, Teledyne CZMIL, Teledyne CZMIL Nova, and Teledyne CZMIL Supernova. The Parrish Group at OSU is also actively working to develop subaqeuous models for more sensors to be included in future versions of cBLUE. To request a sensor model, please post an issue on the cBLUE GitHub page here.



3.1.5.1 Riegl VQ-880-G The Riegl VQ-880-G sensor includes a programmable beam with a selectable beam divergence. Subaqueous models are available for each beam divergence in cBLUE. The following values are the beam divergences modeled by cBLUE in milliradians.

- 0.7 mrad
- 1.0 mrad
- 1.5 mrad
- 2.0 mrad

3.1.5.2 Leica Chiroptera 4X As of V3.0, a subaqeuous model is available to estimate TPU from LAS data collected by Leica Chiroptera 4x sensors.

3.1.5.3 Leica Hawkeye 4X As of V3.0, a subaqeuous model is available to estimate TPU from LAS data collected by Leica Hawkeye 4x sensors. (Note: the "shallow" channel of the Hawkeye 4x sensor is identical to the Chiroptera 4x.

Therefore, similarity between estimates calculated using either of these two models on the same dataset may be nearly identical in shallow waters.)

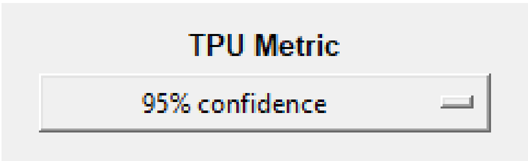
3.1.5.4 Areté PILLS and Fugro RAMMS As of V3.1, a subaqeuous model is available to estimate TPU from LAS data collected by Areté PILLS or Fugro RAMMS sensors.

3.1.5.5 Teledyne CZMIL or CZMIL Nova As of V3.2, a subaqeuous model is available to estimate TPU from LAS data collected by the Teledyne CZMIL or CZMIL Nova sensors.

3.1.5.6 Teledyne CZMIL Supernova As of V3.2, a subaqeuous model is available to estimate TPU from LAS data collected by the Teledyne CZMIL Supernova sensor.

3.1.6 TPU Metric

As of cBLUE V3.0, users may select to output TPU values at either 95% Confidence or 1-σ.



Previous versions of cBLUE defaulted to 1-σ. Conversion between 1-σ and 95% Confidence for Total Vertical Uncertainty (TVU) and Total Horizontal Uncertainty (THU) can also be performed after estimation using the relationships below:

3.1.6.1 TVU:

$$\text{95\% Confidence} = 1.96 \times (1\text{-}\sigma)$$

3.1.6.2 THU:

$$\text{95\% Confidence} = 1.7308 \times (1\text{-}\sigma)$$

3.1.7 Output Options

cBLUE V3.X allows the user to choose between exporting the estimated TVU/THU solely as Extra Bytes in the new LAS file, or to also include a comma separated values (CSV) table with the TVU/THU. This option may be suitable for users with small datasets who wish to work with the resulting uncertainties in software that does not handle LAS files, or those who are unfamiliar with the LAS format. However, the process of building and saving a CSV file with information for each point can be slow and is therefore not recommended for users with large files, or a large number of files.

The information included in the optional output CSV is detailed in Table 2.

Field	Description
GPS Time	The GPS Time of each point in the format of the corresponding input LAS/LAZ file
X	The X coordinate of each point in the format of the corresponding input LAS/LAZ file
Y	The Y coordinate of each point in the format of the corresponding input LAS/LAZ file
Z	The Z coordinate of each point in the format of the corresponding input LAS/LAZ file
THU	The Total Horizontal Uncertainty of each point as estimated by cBLUE
TVU	The Total Vertical Uncertainty of each point as estimated by cBLUE
Classification	The classification of each point as specified in the input LAS/LAZ file

Table 2: Output CSV fields

While other fields can be added to the CSV by modifying cBLUE's code, it is strongly recommended that users wishing to access additional fields simply export as Extra Bytes and use the `las2txt` tool from LASTools, which can be downloaded here.

3.1.7.1 Recommended Software for reading Extra Bytes

Not all commercial software that can read LAS files are capable of reading Extra Bytes. Below is a list of programs that are capable of reading Extra Bytes.

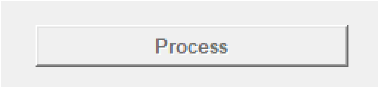
- las2txt tool from LASTools.

  Note: cBLUE uses `laspy` to add Extra Bytes to the LAS file, which stores the THU in the 2nd Extra Byte (Index 1) and TVU in the 3rd Extra Byte (Index 2).

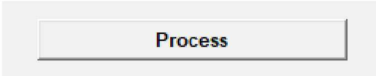
  Example code to read Extra Bytes with las2txt:
  ```
  $ las2txt -i "path/to/LAS_file" -otxt -parse xyz12
  ```
- TerraScan
  - Making Use of Extra Bytes
- LP360
  - Making Use of Extra Bytes
- laspy

3.1.8 Process TPU

After indicating the appropriate data directories and selecting the appropriate parameters for the lidar survey, the "Process" button will become clickable. Clicking this button will initiate the TPU estimation model.



(a) Unclickable

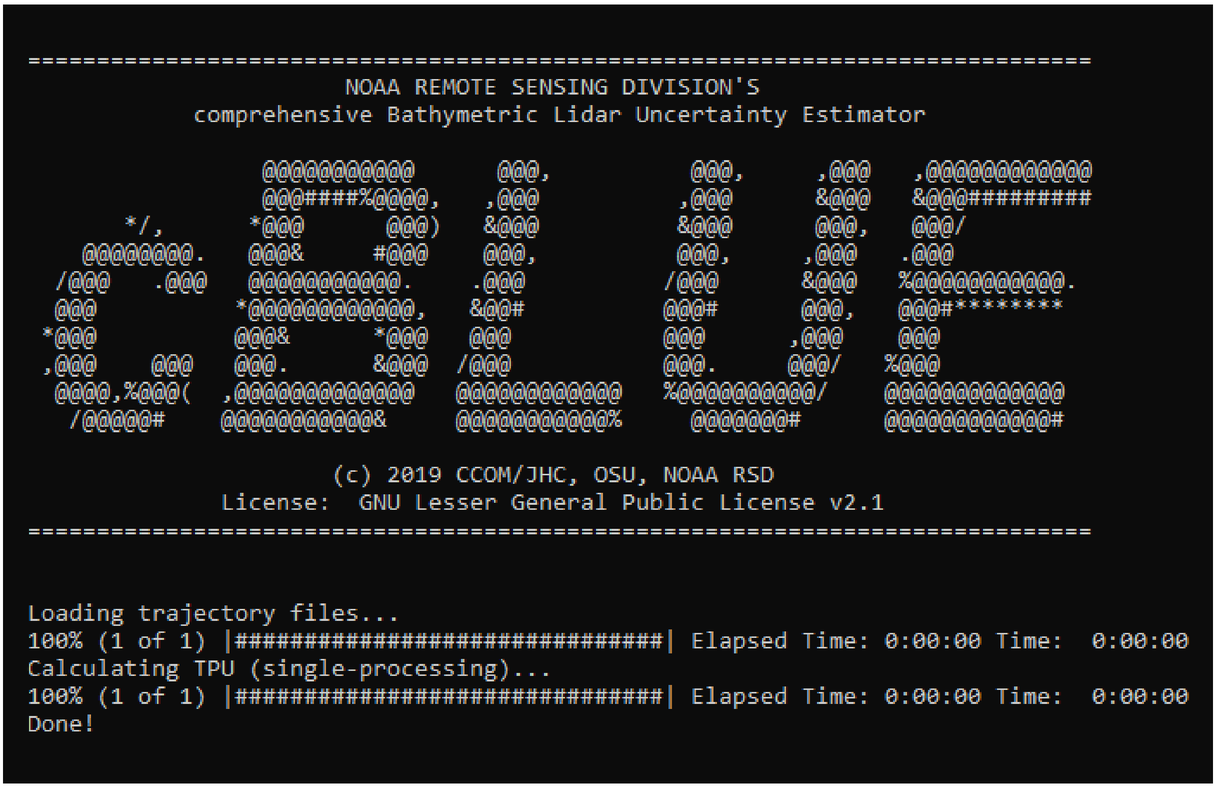


(b) Clickable

Processing TPU may take several minutes, during which time you can track the progress in your terminal/command window.

3.1.9 Monitoring Progress

To monitor the progress of cBLUE, please refer to your terminal/command window. When the model has completed successfully, you will see the following:



3.2 Command Line Interface (CLI)

As of v3.1, cBLUE can be run through the command line interface. This can be useful if you plan to integrate cBLUE into larger programs or run batches of data.

Command line usage looks like:

```
$ python CBlueApp.py in_sbet_dir in_las_dir output_dir wind_speed turbidity
mcu sensor tpu_metric water_height [-vdatum_region VDATUM_REGION] [-csv]
[-just_save_config] [-h]
```

An example command:

```
$ python CBlueApp.py "\SBET\Dir\Path" "\LAS\Dir\Path" "\Output\Dir\Path" 0 2
22.6 7 1 -38.00 -vdatum_region ‘‘Washington/Oregon - Columbia River and Southern
Washington’’ --csv
```

This command would run cBLUE with the selections for wind speed = Calm-light air (0-2 kts), turbidity = Moderate (0.18-0.25 m^{-1}), MCU = 22.6, sensor = HawkEye 4X 600m AGL, tpu metric = 95% Confidence, water height = -38.00 m, the VDatum region name provided will be logged in the metadata file, and the csv output option is turned on.

The help documentation goes into detail on each of the command line arguments.

3.2.1 Help Documentation

Help documentation can be viewed by running the command:

```
$ python CBlueApp.py -h
```

or

```
$ python CBlueApp.py --help
```

```
usage: cblueapp.py [-h] [-vdatum_region VDATUM_REGION] [--csv]
                   [--just_save_config]
                   in_sbet_dir in_las_dir output_dir wind_speed turbidity mcu
                   sensor tpu_metric water_height

Run CBlueApp through the command line interface.

positional arguments:
  in_sbet_dir           Trajectory directory file path.

  in_las_dir            LAS directory file path.

  output_dir            Output directory file path.

  wind_speed            Choose an integer:
                        0 = Calm-light air (0-2 kts),
                        1 = Light Breeze (3-6 kts),
                        2 = Gentle Breeze (7-10 kts),
                        3 = Moderate Breeze (11-15 kts),
                        4 = Fresh Breeze (16-20 kts)

  turbidity             Choose an integer:
                        0 = Clear (0.06-0.10 m^-1),
                        1 = Clear-Moderate (0.11-0.17 m^-1),
                        2 = Moderate (0.18-0.25 m^-1),
                        3 = Moderate-High (0.26-0.32 m^-1),
                        4 = High (0.33-0.36 m^-1)

  mcu                   Input MCU value for the VDatum region. Enter a float value.
                        See .\lookup_tables\V_Datum_MCU Values.txt for MCU values for different VDatum regions.

  sensor                Choose an integer:
                        0 = Riegl VQ-880-G (0.7 mrad),
                        1 = Riegl VQ-880-G (1.0 mrad),
                        2 = Riegl VQ-880-G (1.5 mrad),
                        3 = Riegl VQ-880-G (2.0 mrad),
                        4 = Leica Chiroptera 4X (HawkEye 4X Shallow),
                        5 = HawkEye 4X 400m AGL,
                        6 = HawkEye 4X 500m AGL,
                        7 = HawkEye 4X 600m AGL,
                        8 = PILLS or RAMMS

  tpu_metric            Choose an integer:
                        0 = 1-σ,
                        1 = 95% confidence

  water_height          Nominal water surface ellipsoid height in meters. Enter a float value.

optional arguments:
  -h, --help            show this help message and exit
  -vdatum_region VDATUM_REGION
                        Adds the name of the VDatum region to the metadata log.
                        User must provide the region name after -vdatum_region flag.

  --csv                 Add the --csv flag to generate CSV output files.
  --just_save_config    Do not run process. Save config file only.
```

4 Updates

4.1 V3.0

The following list details the major updates associated with cBLUE V3.0. This is not a comprehensive list of updates to the code, which can be found under the github commit history associated with cBLUE.

- Added Sensor Models:
  - Riegl VQ 880-G :
    - 0.7 mrad
    - 1.0 mrad
    - 1.5 mrad
    - 2.0 mrad
  - Leica Chiroptera 4X
  - Hawkeye 4X
- Removed "Direct from Point Cloud" Option
- Added TPU Metric User Selection

- Updated Dependency (laspy 2.0 → laspy 3.0)
- Updated File Types (.laz files accepted)
- Added a "minimum" TPU value of 3.0 cm. Values below this threshold were determined to be erroneously small.
- Optional CSV output now available.

4.2 V3.1

- Added Sensor Models:
  - Areté PILLS
  - Fugro RAMMS
- TPU processing now done in one click in GUI.
- cBLUE can now be run through command line interfacing.
- Sensor information is stored in the lidar_sensors.json.
- Scan angle and range uncertainties now depend on sensor selection.
- Added dependency (openpyxl).
- Error type is now logged in the metadata file produced for each output LAS file.
- Optional output CSV file records coordinates for X, Y, and Z instead of the LAS point record value of X, Y, and Z.

4.3 V3.2

- Added Sensor Models:
  - Teledyne CZMIL
  - Teledyne CZMIL Nova
  - Teledyne CZMIL Supernova
- cBLUE will no longer add subaquous uncertainty to non-subaqueous points' TPU. Subaqueous processing only generates subaqueous uncertainty for a point if that point has a subaqueous class of 40, 43, 46, or 64.
- Subaqueous processing now handles linear or quadratic subaqueous lookup tables.
- Subaqueous processing is now versioned at V2.2. The current version of subaqueous processing is stored in the cblue_configuration.json and is included in the output of the metadata JSON file.

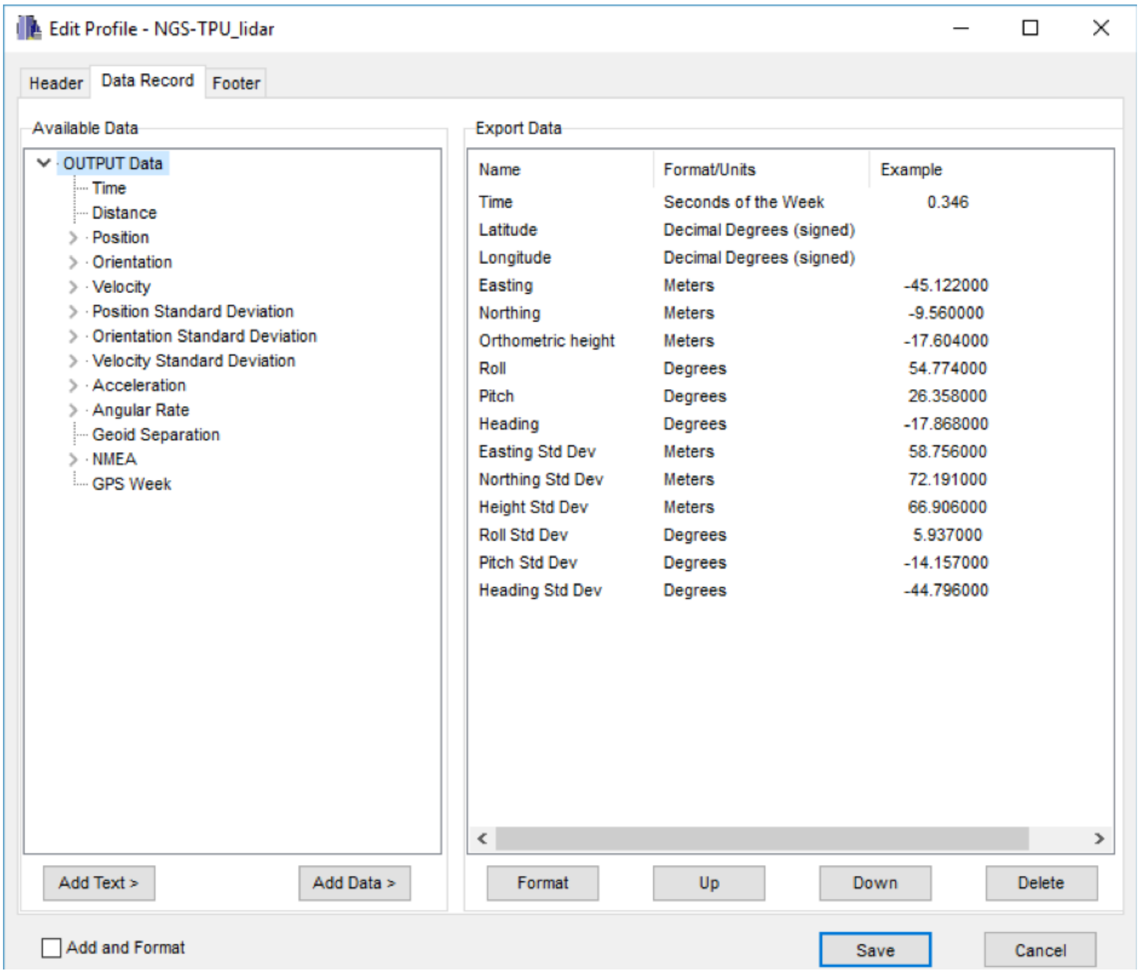
5 Open Issues

This section details the open "issues" for cBLUE. This is not a list of known bugs, but a list of possible additions and modifications that might be made to cBLUE. These additions may be tackled by members of the Parrish Group Cblue maintenance team at Oregon State University, or may be contributed by members of the cBLUE community pending review from the Parrish Group.

- Additional Sensors
  - In the future we hope to provide MATLAB code that will help generate a subaqueous lookup table for new sensors using the Monte Carlo process. Any additional sensor models generated by the cBLUE community are welcome and encouraged, and we would be happy to help you implement it in cBLUE.

A ASCII SBET Format

The format of the trajectory files needed for cBLUE is a custom ASCII SBET with the following format:



References

[1] Firat Eren, Jaehoon Jung, Christopher E. Parrish, Nicholas Sarkozi-Forfinski, and Brian R. Calder. Total Vertical Uncertainty (TVU) Modeling for Topo-Bathymetric LIDAR Systems. *Photogrammetric Engineering & Remote Sensing*, 85(8):585–596, August 2019.