

PARRISH LAB
OREGON STATE UNIVERSITY
CIVIL AND CONSTRUCTION ENGINEERING

AUTHORS: Chris Parrish,
Selina Lambert,
Keana Kief,
Matthew Holwill,
and Forrest Corcoran

The comprehensive Bathymetric Lidar Uncertainty Estimator (cBLUE)



Revision History

Revision	Date	Author(s)	Description
1.0	June 13, 2022	FC	First Draft.
1.1	October 31, 2023	KK	Second Draft.
1.2	June 4, 2024	KK	Third Draft.
1.3	April 16, 2025	KK	Added Features.
1.4	August 7, 2025	CP, SL, KK, MH	New major cBLUE version release (v4.0), including enhancement to the ocean optics models and Monte Carlo ray tracing, as well as adding new functionality.

Contents

1	cBLUE Overview	3
1.1	Statement of Purpose	3
1.2	License	3
1.3	Contact Information	3
2	Installation	4
2.1	Installing cBLUE	4
2.1.1	Download via git	4
2.1.2	Download as .zip	5
2.2	Install cBLUE Dependencies	6
2.2.1	conda	6
2.2.2	Anaconda Navigator	6
3	User Guide	7
3.1	Graphical User Interface (GUI)	7
3.1.1	Data Directories	8
3.1.2	Environmental Parameters	11
3.1.3	Water Surface Ellipsoid Height	12
3.1.4	VDatum Region	13
3.1.5	Optional User Input Uncertainty Component	13
3.1.6	Sensor Models	14
3.1.7	TPU Metric	15
3.1.8	Output Options	15
3.1.9	Process TPU	17
3.1.10	Monitoring Progress	17
3.2	Command Line Interface (CLI)	18
3.3	Configuration File (Config)	18
3.3.1	Directories	19
3.3.2	Multiprocessing	19
3.3.3	Versions	19
3.3.4	Subaqueous Classes	20
3.3.5	Other Config Variables	20
3.3.6	Help Documentation	20
4	Updates	22
4.1	V3.0	22
4.2	V3.1	22
4.3	V3.2	22
4.4	V3.3	23
4.5	V3.4	23
4.6	V4.0	23
5	Open Issues	24
A	ASCII SBET Format	25

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/Joint Hydrographic Center (CCOM/JHC). cBLUE is designed to provide Total Vertical Uncertainty (TVU) and Total Horizontal Uncertainty (THU) for bathymetric lidar surveys.

1.2 License

cBLUE V4.0

Copyright (C) 2019

Oregon State University (OSU)

Center for Coastal and Ocean Mapping/Joint Hydrographic Center,

University of New Hampshire (CCOM/JHC, UNH)

NOAA Remote Sensing Division (NOAA RSD)

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

1.3 Contact Information

Christopher Parrish, PhD

School of Construction and Civil Engineering

204 Owen Hall

Oregon State University

Corvallis, OR 97331

(541) 737-5688

Christopher.Parrish@oregonstate.edu

2 Installation

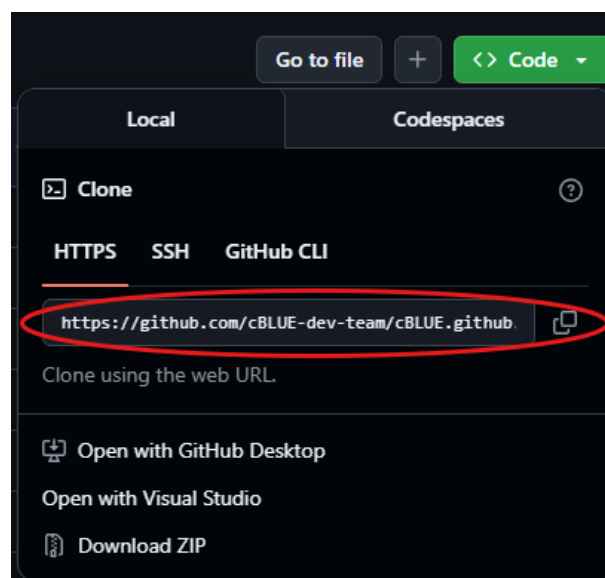
2.1 Installing cBLUE

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/cBLUE-dev-team/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

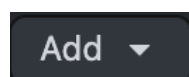
```
$ cd location/of/cBLUE
```

- (b) Clone the repository using the url from github

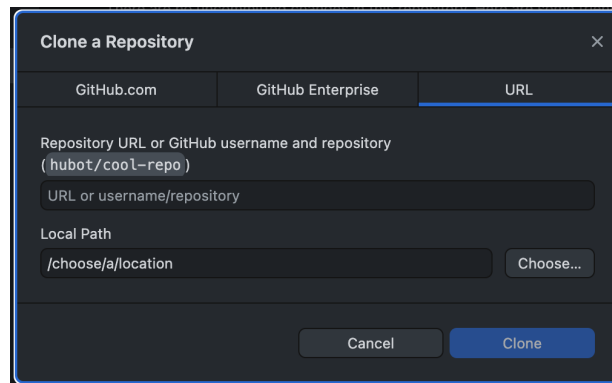
```
$ git clone https://github.com/cBLUE-dev-team/cBLUE.github.io
```

2.1.1.3 GitHub Desktop

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

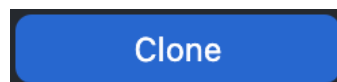


(b) Under "Add" choose "Clone Repository" and select "URL"



(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

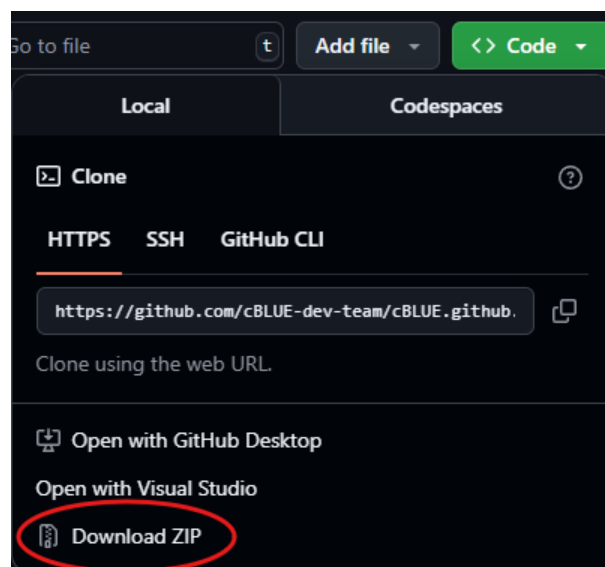
(d) Press the "Clone" button to begin download



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/cBLUE-dev-team/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 Miniconda, 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

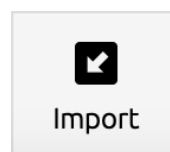
```
$ conda activate cblue
```

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

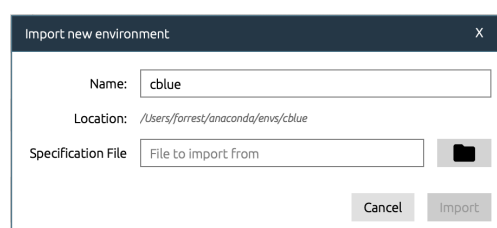
```
$ python CBlueApp.py
```

2.2.2 Anaconda Navigator

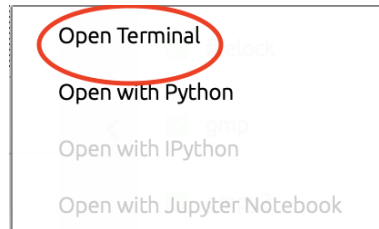
1. Open "Anaconda Navigator" and navigate to the "Environments" menu on the left hand side.
2. Locate the "Import" button on the bottom left



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 installed 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 the corresponding trajectory.

3.1 Graphical User Interface (GUI)

To open cBLUE, in the terminal of your choice, 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:

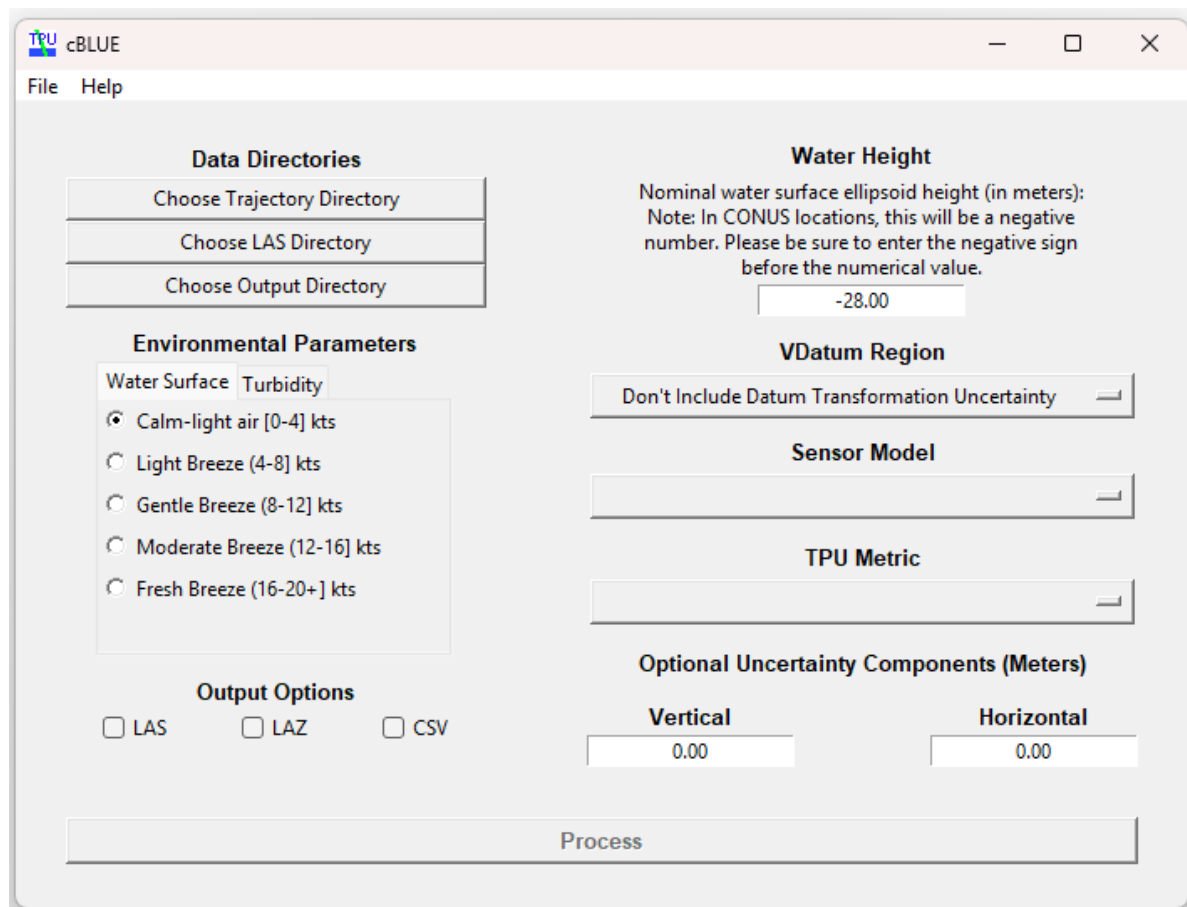
```
=====
NOAA REMOTE SENSING DIVISION'S
comprehensive Bathymetric Lidar Uncertainty Estimator

          @@@@@@@@@@   @@@,   @@@,   ,@@   ,@@@@@@@@@@@
          @@@###%@@@@,   @@@   ,@@   ,@@   %@@@@#####
          *,   *@@@   @@@   %@@@   @@@   @@@   @@@/
          @@@@@@@. @@@%   #@@@   @@@,   @@@,   ,@@   ,@@@
          /@@@   .@@@ @@@@@@@@@@@. .@@@   /@@@   %@@@@@@@@@@.
          @@@   *@@@@@@@@@@@@. %@@#   @@@#   @@@,   @@@#*****
          *@@@   @@@%   *@@@   @@@   @@@   ,@@@   @@@
          ,@@@   @@@   @@@.   %@@@   /@@@   @@@.   @@@/   %@@@
          @@@,%@@@   ,@@@@@@@@@@@@ @@@@@@@@@@@@ %@@@@@@@@@@/ @@@@@@@@@@@@
          /@@@@#   @@@@@@@@@@@% @@@@@@@@@@@% @@@@@@@# @@@@@@@@@@@#

          (c) 2019 CCOM/JHC, OSU, NOAA RSD
          License: GNU Lesser General Public License v2.1
=====
```

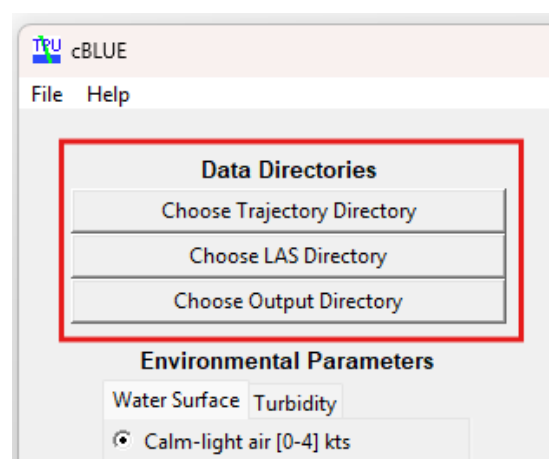

(Note: the cblue conda environment must be installed and activated in order to run cBLUE. If cBLUE fails to open, it is likely that your dependencies are not installed or the environment is deactivated. Refer to Section 2 of this manual for information on resolving these issues.)

The GUI should launch:



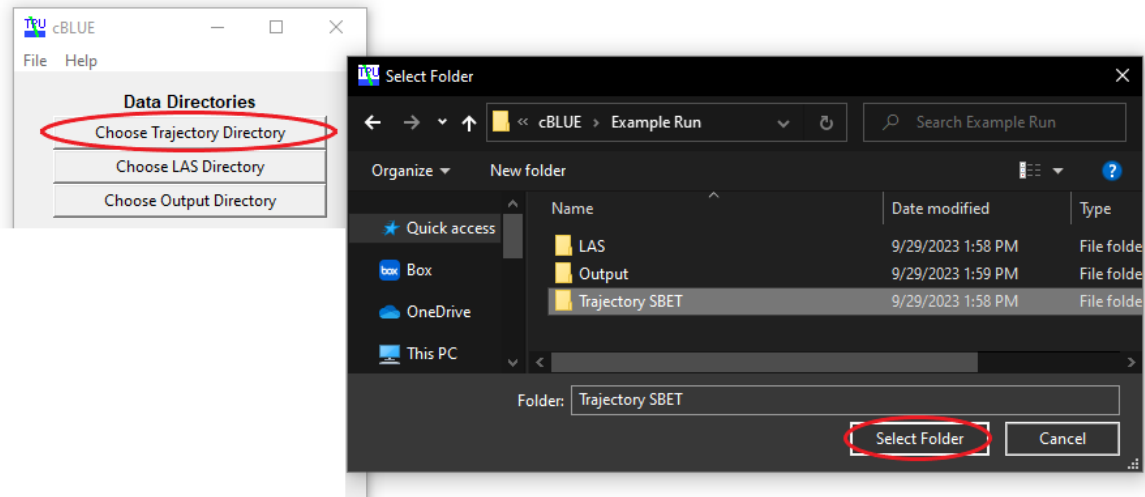
3.1.1 Data Directories

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.

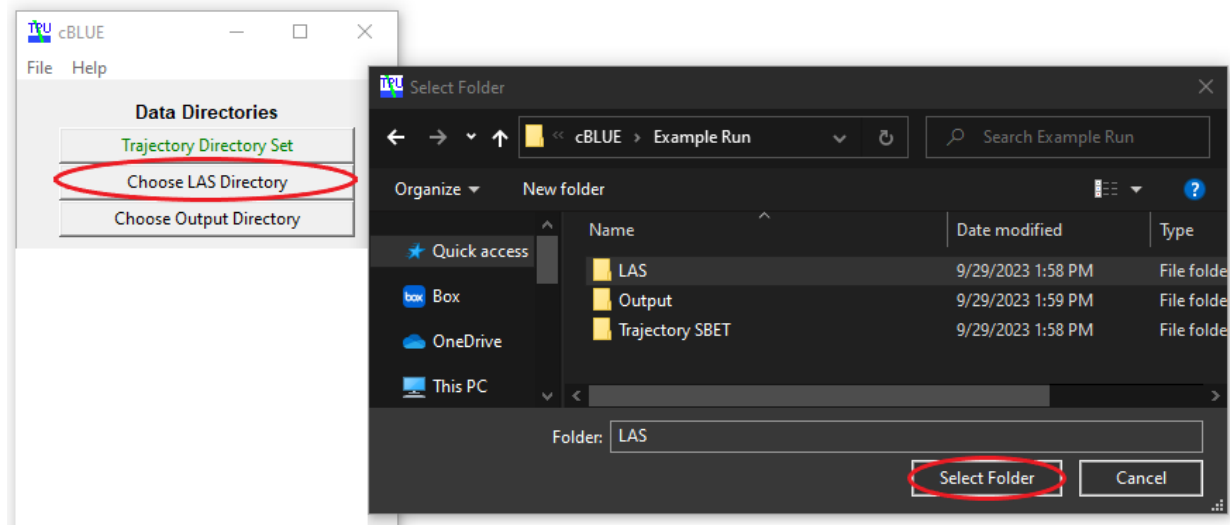


A brief overview of ASCII SBET formatting is available in [Appendix A](#). If you run into issues regarding the SBET files, please consult the lidar system's user manual or contact the manufacturer before raising an issue with the cBLUE DevTeam, as these issues are often the result of erroneous data and are therefore difficult to reproduce/troubleshoot. An example of SBET files is shown in the figure below.

_files > NGS_Test_Data_raw_and_adjusted_times > Trajectories				
Name	Date modified	Type	Size	
20160515.1_880_p_sbet_lidar_tpu.log	9/21/2016 8:47 AM	Text Document	1 KB	
20160515.1_880_p_sbet_lidar_tpu.txt	9/21/2016 8:47 AM	Text Document	946,699 KB	
20160515.2_880_p_sbet_lidar_tpu.log	9/21/2016 9:21 AM	Text Document	1 KB	
20160515.2_880_p_sbet_lidar_tpu.txt	9/21/2016 9:21 AM	Text Document	728,789 KB	
20160517_880_p_sbet_lidar_tpu.log	9/21/2016 9:24 AM	Text Document	1 KB	
20160517_880_p_sbet_lidar_tpu.txt	9/21/2016 9:24 AM	Text Document	602,521 KB	

3.1.1.2 LAS Files

The "LAS Directory Set" button allows the user to select the location of the .las or .laz files containing the processed point clouds.



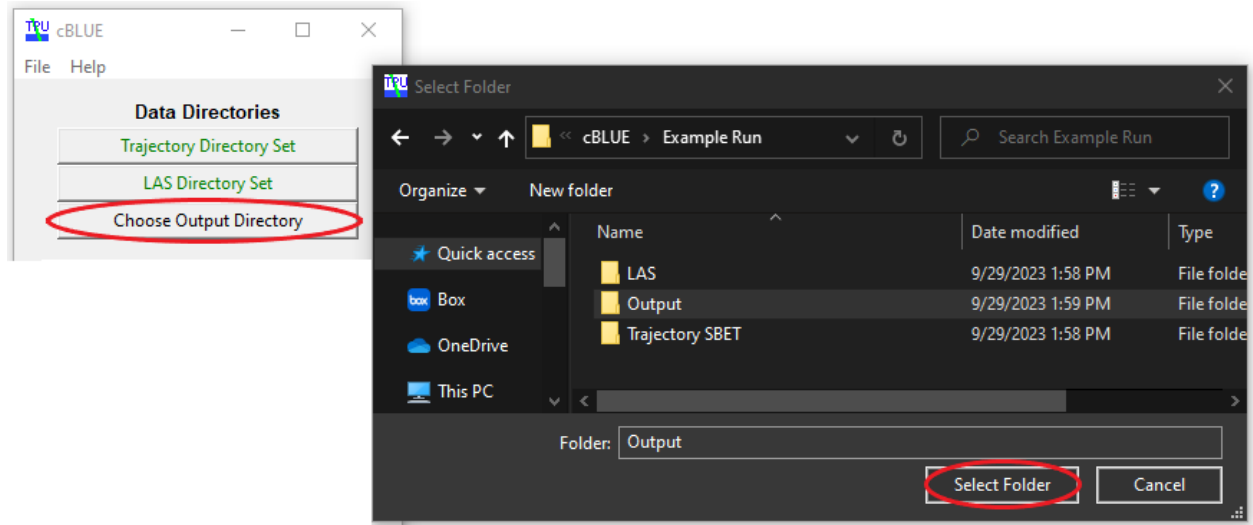
Given the widespread use of .las and .laz, especially in the bathymetric lidar community, these are the only file types supported. For more information on LAS file specifications, please refer to the ASPRS LAS Working Group and the ASPRS Bathymetric Lidar Working Group.

It is strongly recommended that the GPS times in the LAS (or LAZ) files be **Adjusted Standard GPS Time**. Time formatting is important, because GPS times are how cBLUE matches up points in the LAS file with corresponding records from the trajectory. As of v4.0 of cBLUE, the software will attempt to automatically detect time format inconsistencies and apply a conversion, if needed. However, if the conversion is unsuccessful, the LAS file and trajectory will not be merged, and the cblue.log file should display the warning: "trajectory and LAS data NOT MERGED."

Additionally, the LAS or LAZ files should contain **UTM coordinates**, with units of meters, and the Z values should be **ellipsoid heights**, also with units of meters. (Note: it may be possible to input LAS files with orthometric heights and to enter a water surface orthometric height in the "Water Height" field in the GUI. However, this is not something that the cBLUE DevTeam has tested, and we advise against it, pending further testing.)

3.1.1.3 TPU Directory

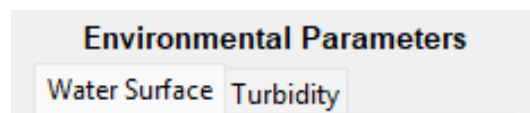
The "TPU Directory Set" button allows the user to select the location where the output files will be generated.



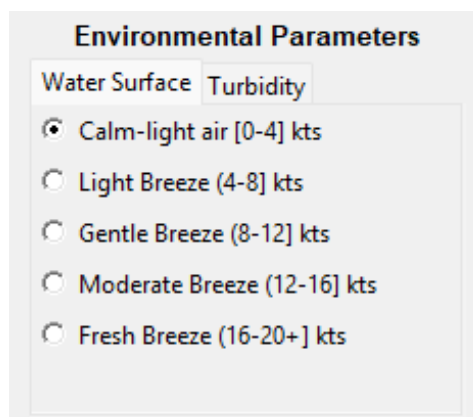
The files output by cBLUE will match the .las files in the "LAS Directory Set" location, with _TPU appended to the file name. These output files will contain the TPU as Extra Bytes fields. More information on LAS Extra Bytes can be found [here](#).

3.1.2 Environmental Parameters

cBLUE operates by using a combined subaerial and subaqueous model. The subaqueous 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.

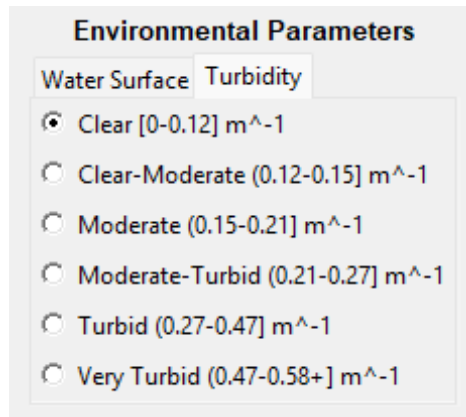


3.1.2.1 Water Surface The "Water Surface" menu allows the user to select the range of wind speed values that best approximate the water surface conditions at the time of the bathymetric survey. Note: the wind speed categories were updated in cBLUE v4.0.



(Note: cBLUE V2.X included an option to model the water surface 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. Note: the turbidity categories were updated between cBLUE v3.4 and v4.0.



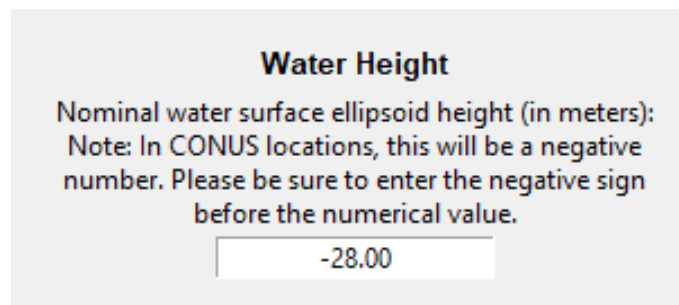
Environmental Parameters

Water Surface **Turbidity**

- ☒ Clear [0-0.12] m⁻¹
- ☐ Clear-Moderate (0.12-0.15] m⁻¹
- ☐ Moderate (0.15-0.21] m⁻¹
- ☐ Moderate-Turbid (0.21-0.27] m⁻¹
- ☐ Turbid (0.27-0.47] m⁻¹
- ☐ Very Turbid (0.47-0.58+] m⁻¹

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. Note: In CONUS locations, this will be a negative number. Please be sure to enter the negative sign before the numerical value.



Water Height

Nominal water surface ellipsoid height (in meters):
Note: In CONUS locations, this will be a negative number. Please be sure to enter the negative sign before the numerical value.

-28.00

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.

The screenshot displays the NOAA Online Vertical Datum Transformation tool and the cBLUE interface. In the NOAA tool, the 'Vertical Information' section shows a transformation from NAD83(2011) to LMSL (Mean Lower Low Water) with a height of 0.00. The cBLUE interface shows the output of this transformation as -23.343, which is rounded to -23.34 in the 'Water Height' field. A red arrow points from the output value in the NOAA tool to the 'Water Height' field in the cBLUE interface.

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. If this parameter is left at the default setting of "Don't Include Datum Transformation Uncertainty," then uncertainty associated with the vertical datum transformation will be ignored (set to zero) in the TPU computation.

The screenshot shows the 'VDatum Region' menu in the cBLUE interface. The menu is open, displaying the following options: 'Don't Include Datum Transformation Uncertainty', 'New Jersey - Coastal embayments - South', 'New Jersey/New York/Connecticut - Northern NJ, NY Harbor, western Long Island Sound', and 'New York - The Great South Bay'.

3.1.5 Optional User Input Uncertainty Component

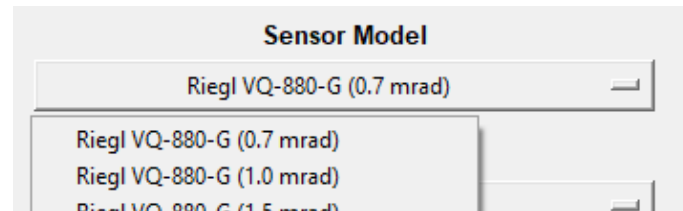
Introduced in V4.0, users may enter their own vertical and/or horizontal uncertainty components (VUC or HUC). The VUC or HUC entered should be a float value in meters. This optional uncertainty component will be added in the sum of squares calculation for TVU and/or THU.

One example of how this might be used is if a custom datum transformation is performed (outside of VDatum) that introduces an additional component uncertainty.

The screenshot shows the 'Optional Uncertainty Components (Meters)' dialog box. It contains two input fields: 'Vertical' and 'Horizontal', both of which are set to 0.00.

3.1.6 Sensor Models

As of cBLUE V4.0, there are 10 lidar sensor platforms available in cBLUE. These include the Riegl VQ-880-G, Leica Chiroptera 4X, Leica Chiroptera-5, Leica HawkEye 4X, Leica HawkEye-5, Areté PILLS, Fugro RAMMS, Teledyne CZMIL, Teledyne CZMIL Nova, and Teledyne CZMIL SuperNova. The cBLUE DevTeam is also actively working to develop subaqueous 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](#).



3.1.6.1 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

The subaqueous model for the Riegl VQ-880-G was updated in cBLUE V4.0.

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

The subaqueous model for the Chiroptera 4X was updated in cBLUE V4.0.

3.1.6.3 Leica Chiroptera-5 As of V4.0, a subaqueous model is available to estimate TPU from LAS data collected by Leica Chiroptera-5 sensors. Models are available for flying heights of 400, 500, and 600m.

3.1.6.4 Leica HawkEye 4X and 5 As of V3.0, a subaqueous model is available to estimate TPU from LAS data collected by Leica HawkEye 4x sensors. Models are available for flying heights of 400, 500, and 600m. (Note: the "shallow" channel of the HawkEye 4x and 5 sensors are identical to the Chiroptera 5. Therefore, similarity between estimates calculated using either of these two models on the same dataset may be nearly identical in shallow waters.)

The subaqueous model for the HawkEye 4X was updated in cBLUE V4.0 and now includes the HawkEye-5 system. cBLUE now stores LUTs generated for each subaqueous scanner and channel combination of the HawkEye System. During subaqueous processing for HawkEye systems, cBLUE uses the scanner channel and user data LAS variables to determine the appropriate LUT to call for each data point.

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

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

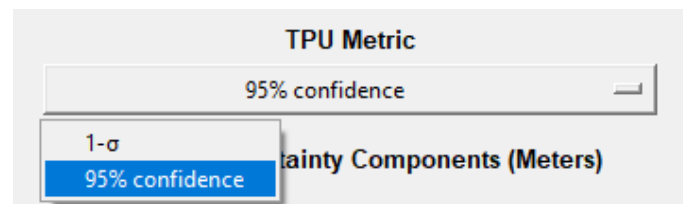
The subaqueous model for the CZMIL and CZMIL Nova was updated in cBLUE V4.0.

3.1.6.7 Teledyne CZMIL SuperNova As of V3.2, a subaqueous model is available to estimate TPU from LAS data collected by the Teledyne CZMIL SuperNova sensor.

The subaqueous model for the SuperNova was updated in cBLUE V4.0.

3.1.7 TPU Metric

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



Conversion between $1-\sigma$ and 95% Confidence for Total Vertical Uncertainty (TVU) and Total Horizontal Uncertainty (THU) can also be performed after estimation using the relationships below:

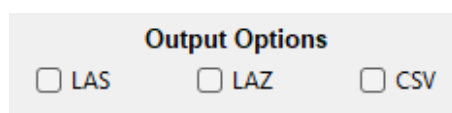
3.1.7.1 TVU:

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

3.1.7.2 THU:

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

3.1.8 Output Options



cBLUE allows the user to export the estimated TVU/THU as Extra Bytes in a new LAS file, Extra Bytes in a LAZ file, and/or as a comma separated values (CSV) table with TVU/THU fields. If no output option is chosen, a LAS file will be produced as the output file by default.

The CSV 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](#).

3.1.8.1 Metadata:

After processing is complete, cBLUE will always produce a metadata json file for each LAS file in the LAS directory selected by the user. The metadata file(s) includes information on the user input selected for the cBLUE run and flight line statistics relating to the trajectory files provided.

A metadata file will look approximately like this:

```

1  {
2    "Wind speed": "Light Breeze (4-8] kts",
3    "Turbidity": "Clear-Moderate (0.12-0.15] m^-1",
4    "VDatum region": "Don't Include Datum Transformation Uncertainty",
5    "VDatum region MCU": "0.0",
6    "Optional VUC": 0.0,
7    "Optional HUC": 0.0,
8    "Flight line stats (min max mean stddev)": {
9      "1001 (2283085/2283085 points with TPU)": [
10       "total_thu: 0.080 58.284 5.331 7.415",
11       "total_tvu: 0.089 1.189 0.129 0.082"
12     ]
13   },
14   "Sensor model": "Sensor Name",
15   "cBLUE version": "v4.0",
16   "Subaqueous processing version": "v3.0",
17   "CPU processing": [
18     "multiprocess",
19     4
20   ],
21   "Water surface ellipsoid height": -28.0,
22   "Error type": "95% confidence"
23 }

```

3.1.8.2 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](#)

3.1.9 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.



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

3.1.10 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:

```

=====
NOAA REMOTE SENSING DIVISION'S
comprehensive Bathymetric Lidar Uncertainty Estimator

      @@@@@@@@@@@@@ @@@,      @@@,      ,@@@      ,@@@@@@@@@@@@@
      @@@#####%@@@@, @@@      ,@@@      &@@@      &@@@#####
      */,      *@@@      @@@)      &@@@      &@@@      @@@/
      @@@@@@@@@. @@@&      #@@@      @@@,      @@@,      .@@@
      /@@@      .@@@ @@@@@@@@@@@@@. @@@      /@@@      &@@@      %@@@@@@@@@@@@.
      @@@      *@@@@@@@@@@@@@@@@, &@@#      @@@#      @@@,      @@@#*****
      *@@@      @@@&      *@@@      @@@      @@@      ,@@@      @@@
      ,@@@      @@@      @@@.      &@@@      /@@@      @@@.      @@@/      %@@@
      @@@@,%@@@( ,@@@@@@@@@@@@@@@@ @@@@@@@@@@@@@@ %@@@@@@@@@@@@/ @@@@@@@@@@@@@@
      /@@@@@# @@@@@@@@@@@@@& @@@@@@@@@@@@@@% @@@@@@# @@@@@@@@@@@@@@#

      (c) 2019 CCOM/JHC, OSU, NOAA RSD
      License: GNU Lesser General Public License v2.1
=====

Loading trajectory files...
100% (1 of 1) |#####| Elapsed Time: 0:00:00 Time: 0:00:00
Calculating TPU (single-processing)...
100% (1 of 1) |#####| Elapsed Time: 0:00:00 Time: 0:00:00
Done!

```

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] [-opt_vuc OPT_VUC]
[-opt_huc OPT_HUC] [--csv] [--las] [--laz] [--save_config] [--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" -opt_vuc 0.099 -opt_huc 0.123 --las --csv
```

This command would run cBLUE with the selections for wind speed = Calm-light air (0-4 kts), turbidity = Moderate (0.15-0.21 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, a VUC of 0.099m will be added in quadrature to TVU, a HUC of 0.123m will be added in quadrature to THU, and the output files will include LAS and CSV.

Note: As of v3.3 including the --csv flag will only produce a CSV file as the output. Previously including the --csv flag would generate LAS and CSV output. Add the --las or --laz flags in addition to the --csv flag to have multiple output file types.

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

3.3 Configuration File (Config)

The configuration file, **cblue_configuration.json**, stores information required for a run of cBLUE. It can be found in the main cBLUE.github.io folder.

```

{} cblue_configuration.json > ...
1  {
2      "directories": {
3          "sbet": "path/to/sbet/",
4          "las": "path/to/las/",
5          "tpu": "path/to/output/"
6      },
7      "multiprocess": "False",
8      "number_cores": 4,
9      "cBLUE_version": "v3.3",
10     "subaqueous_version": "v2.2",
11     "subaqueous_classes": [
12         "40",
13         "43"
14     ],
15     "sensor_model": "PILLS or RAMMS",
16     "water_surface_ellipsoid_height": -28.0,
17     "error_type": "95% confidence"
18 }

```

3.3.1 Directories

Path to directories used by cBLUE. sbet is the path to the directory containing SBET file(s) provided by the user. las is the path to the directory containing LAS or LAZ file(s) provided by the user. tpu is the path to the directory where output files will be written.

3.3.2 Multiprocessing

By default multiprocessing is turned off. To use multiprocessing change "multiprocess": "False" to "multiprocess": "True".

Adjust the number of CPU cores to be used in multiprocessing by adjusting the value after "number_cores". This value will be specific to the machine you are running cBLUE on.

On a Windows machine, the number of cores can be found by opening the Task Manager (ctl + Alt + Delete) and then clicking on the Performance tab in the sidebar.

On Mac, choose Apple menu > System Settings, then click General in the sidebar. Click About and then click System Report.

Known issue: Currently the process of loading the trajectory files is not included in the multiprocessing.

3.3.3 Versions

cBLUE_version is the version of cBLUE you are using.

subaqueous_version is the version of the cBLUE subaqueous processing used to produce the sensor LUTs.

3.3.4 Subaqueous Classes

cBLUE will calculate a combined aerial and subaqueous uncertainty for each point that has a classification values in the subaqueous_classes list. Points with classification values not in this list will have aerial uncertainty calculated only.

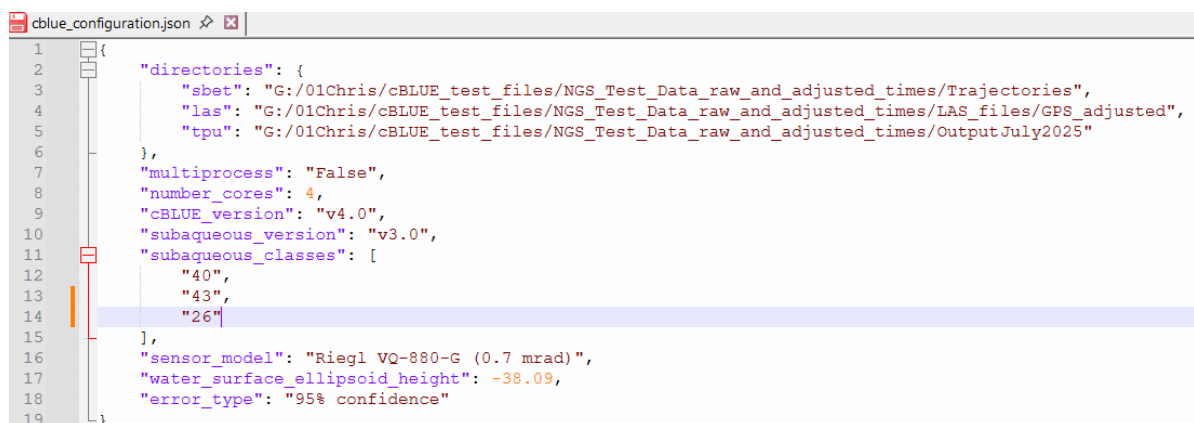
Class values in the subaqueous_classes list come from the ASPRS Proposed LAS Enhancements to Support Topo-Bathy Lidar [2].

40 - Bathymetric point (e.g., seafloor or riverbed; also known as submerged topography)

43 - Submerged object, not otherwise specified (e.g., wreck, rock, submerged piling)

If you want to add classification values for cBLUE to consider as subaqueous points, add the class values desired to the list. For example the current subaqueous_classes list is ["40", "43"]. If you wanted to add classes 46 and 64 you should edit your config file so that the subaqueous_classes list looks like ["40", "43", "46", "64"].

If Class 26 was used for bathymetry points in your LAS file, the figure below shows an example of how to make this edit to the cblue_configuration.json file using Notepad++.



3.3.5 Other Config Variables

sensor_model is the sensor used for data collection.

water_surface_ellipsoid_height is the ellipsoid height of the water surface as a float value in meters.

error_type shows if the output TPU values are 95% Confidence or 1- σ .

3.3.6 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] [-opt_vuc OPT_VUC]
                  [-opt_huc OPT_HUC] [--csv] [--las] [--laz] [--save_config]
                  [--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 [0-4] kts, 1 = Light Breeze (4-8] kts, 2 = Gentle Breeze (8-12] kts, 3 = Moderate Breeze (12-16] kts, 4 = Fresh Breeze (16-20+] kts
turbidity	Choose an integer: 0 = Clear [0-0.12] m ⁻¹ , 1 = Clear-Moderate (0.12-0.15] m ⁻¹ , 2 = Moderate (0.15-0.21] m ⁻¹ , 3 = Moderate-Turbid (0.21-0.27] m ⁻¹ , 4 = Turbid (0.27-0.47] m ⁻¹ , 5 = Very Turbid (0.47-0.58+] m ⁻¹
mcu	Input maximum cumulative uncertainty (MCU) value in cm 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 = Chiroptera 4X (HawkEye 4X Shallow), 5 = Chiroptera-5 400m, 6 = Chiroptera-5 500m, 7 = Chiroptera-5 600m, 8 = HawkEye 4X or 5 400m AGL, 9 = HawkEye 4X or 5 500m AGL, 10 = HawkEye 4X or 5 600m AGL, 11 = PILLS or RAMMS, 12 = CZMIL or CZMIL Nova (Shallow), 13 = CZMIL or CZMIL Nova (Deep), 14 = CZMIL SuperNova (Shallow), 15 = CZMIL SuperNova (Deep)
tpu_metric	Choose an integer: 0 = 1-σ, 1 = 95% confidence
water_height	Nominal water surface ellipsoid height in meters. Enter a float value. Note: In CONUS locations, this will be a negative number. Please be sure to enter the negative sign before the numerical 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.
-opt_vuc OPT_VUC	Optional user generated vertical uncertainty component (VUC) value in meters. Enter a float value.
-opt_huc OPT_HUC	Optional user generated horizontal uncertainty component (HUC) value in meters. Enter a float value.
--csv	Add the --csv flag to generate CSV output files.
--las	Add the --las flag to generate LAS output files.
--laz	Add the --laz flag to generate LAZ output files. Note: cBLUE will default to LAS output if no output flags (--csv, --las, or --laz) are provided.
--save_config	Updates the cblue_configuration.json in the main cBlue app folder with the settings for the current run. *WARNING* --save_config is not recommended when running multiple cBlue CLI processes concurrently because of potential multi-write conflicts.
--just_save_config	Do not run cBLUE process and update the cblue_configuration 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 subaqueous 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.

4.4 V3.3

- When cBLUE is run through the CLI the `cblue_configuration.json` file will no longer update by default. The config file will still update by default when run through the GUI. To update the config file when run through the CLI, add the `--save_config` flag to the command.
- Moved subaqueous class list to the `cblue_configuration.json` file. Allows the user to select the LAS point class(es) for which TPU is computed. Users can add or remove subaqueous class values by editing the `cblue_configuration.json` file.
- Added ability to handle LAS files that have ExtraBytes already added.
- Add the option for the user to select LAZ as the output file type. User can now use flags `--las`, `--laz`, and `--csv` to get output files of the respective type. No flags chosen will default to LAS file output. User can supply multiple output flags, for example: using `--las` and `--csv` will produce output files of both types. The GUI has also been modified to provide check boxes for LAS, LAZ, or CSV to match.
- Added additional help text to the water height field.

4.5 V3.4

- Fixed error when copying LAS header for output files. In Version 3.3 not all header values were being written to the output LAS files.

4.6 V4.0

- This version incorporates major enhancements to the ocean optics models and Monte Carlo ray tracing underlying the subaqueous TPU modeling. In the updated model, the physics of the ray tracing algorithm relies on the beam attenuation coefficient as a function of water turbidity to determine the probability of scattering versus absorption, and a scattering phase function to determine the probability of different scattering angles. The forward scattering phase function was updated from Henyey-Greenstein to Fournier-Forand, which better approximates the Peltzold scattering phase function and performs more realistically for very small and very large angle probabilities. The beam attenuation is a combination of empirically determined absorption and scattering coefficients and are linked to Jerlov water types. The water types III, 1C, 3C, 5C, 7C, and 9C describe near-coast clear ocean water (III) and five coastal types of increasing turbidity. These water types are mapped to their estimated K_{d490} in the cBLUE GUI. An important component change to this version is the incorporation of range bias uncertainty estimation, which is a function of the sensor transmit pulse width, the turbidity-dependent signal-to-noise ratios (SNR) and the difference between target depth and modeled mean path length. Ranging uncertainty is a function of sensor-specific peak finding applied to the return signal (typically via real-time or post-processing of waveforms), as well as sensor calibration, and a sensor-specific parameter adjusts this component uncertainty to the expected range. Additionally, the Monte Carlo ray tracing samples facets for each starting photon from an interpolated water surface within the on-water footprint in each repeated simulation rather than a randomly selected water facet for all photons. The coefficients used to fit the Monte Carlo ray tracing output have been updated to follow IHO S-44 type curves. Previous versions of cBLUE output a horizontal uncertainty that described the standard deviation of the center of the beam at depth, resulting in THU values on the scale of centimeters. In this update, THU values represent the 2-D distribution of all

points on the seafloor for each simulation, which results in THU values of 2+ meters. In the `cblue_configuration.json`, the subaqueous versioning is updated from v2.2 to v3.0. (Note that as of v4.0, the PILLS/RAMMS system still uses the old subaqueous v2.2 models; this will be updated in v4.1.)

- Added Sensor Models:
 - Leica Chiroptera-5
 - Leica HawkEye-5
- The user now has the ability to input additional (optional) vertical and/or horizontal uncertainty components (VUC or HUC). If included, these optional uncertainty components are added in quadrature with the other component uncertainties in the final TVU and/or THU computations.
- Updated HawkEye subaqueous processing to account for data collection from deep-narrow, deep-wide, and shallow (Chiroptera-5) scanners.
- cBLUE now attempts to automatically detect time format inconsistencies between the LAS file(s) and trajectory file(s) and to apply a conversion, if needed. If merging the LAS and SBET fails, cBLUE attempts to convert the LAS data from GPS standard time or UTC to adjusted standard GPS time and then retries merging the data.
- Fixed issue #72 with json file formatting. THU and TVU values > 9.999 now no longer run together.
- Updated VDatum values available for selection.
- Added VUC and HUC values to metadata json file. Adjusted key values names in the metadata json to be more consistent.
- GUI has been updated to have a more landscape appearance.
- Fixed an error introduced in version 3.3 when creating output files. Sorting of THU and TVU values was different than other field values. Fixed LAS field value sorting to match.

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 cBLUE DevTeam, 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:

The screenshot shows a software window titled "Edit Profile - NGS-TPU_lidar". It has three tabs: "Header", "Data Record", and "Footer", with "Data Record" currently selected. The window is divided into two main panels. The left panel, "Available Data", shows a tree structure under "OUTPUT Data" with items like Time, Distance, Position, Orientation, Velocity, Position Standard Deviation, Orientation Standard Deviation, Velocity Standard Deviation, Acceleration, Angular Rate, Geoid Separation, NMEA, and GPS Week. The right panel, "Export Data", contains a table with three columns: Name, Format/Units, and Example. Below the table are buttons for "Format", "Up", "Down", and "Delete". At the bottom of the window are checkboxes for "Add Text >", "Add Data >", and "Add and Format", along with "Save" and "Cancel" buttons.

Name	Format/Units	Example
Time	Seconds of the Week	0.346
Latitude	Decimal Degrees (signed)	
Longitude	Decimal Degrees (signed)	
Easting	Meters	-45.122000
Northing	Meters	-9.560000
Orthometric height	Meters	-17.604000
Roll	Degrees	54.774000
Pitch	Degrees	26.358000
Heading	Degrees	-17.868000
Easting Std Dev	Meters	58.756000
Northing Std Dev	Meters	72.191000
Height Std Dev	Meters	66.906000
Roll Std Dev	Degrees	5.937000
Pitch Std Dev	Degrees	-14.157000
Heading Std Dev	Degrees	-44.796000

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.
- [2] American Society for Photogrammetry and Remote Sensing (ASPRS). Las domain profile description: Topo-bathy lidar. url: https://www.asprs.org/wp-content/uploads/2010/12/LAS_Domain_Profile_Description_Topo-Bathy_Lidar.pdf, July 2017. 20