



The PINGMapper Handbook v1.0

Teachings from the delirious open-source developer

Cameron S. Bodine, PhD

[CSHEL - Coastal Sediments Hydrodynamics and Engineering Laboratory](#)

School of Marine Science and Policy

College of Earth, Ocean, and Environment

University of Delaware

[Orcid](#) | [GitHub](#) | [LinkedIn](#) | [Twitter](#) | [Website](#)

Prepared for “**Side-Scan Sonar: Cutting-Edge Tools and Techniques for Managing Habitat and Fisheries**” Continuing Education Course at the American Fisheries Society 155th Annual Meeting in San Antonio, Texas, USA

August 10, 2025, 8:00am – 5:00pm

Workshop Instructors:

Cameron Bodine, University of Delaware

Jesse Fischer, US Geological Survey

Adam Kaeser, US Fish & Wildlife Service

Josey Ridgway, US Geological Survey

Table of Contents

Introduction..... 3

1) Sonogram Tile Exports (non-georeferenced) 5

Sonar Georectified Exports (georeferenced) 13

Thread Count & Chunk Size: Effects on RAM usage and processing speed 14

Filtering: Heading Deviation 15

Filtering: AOI..... 16

Filtering: Time 17

EGN Corrections to Intensity..... 18

Substrate Mapping 19

Appendix A – Installation 20

Appendix B – Launch PINGWizard 23

Appendix C – Colormap Selection 25

References..... 27

DRAFT

Introduction

PINGMapper (PM) is an open-source Python interface for reading and processing side scan sonar datasets and reproducibly mapping benthic habitat features. PM transforms recreation-grade sonar systems (i.e. fishfinders) into scientific instruments, allowing researchers and citizens alike to reproducibly map their aquatic system with minimal expertise in data processing.

While many sonar processing software feature rich interfaces to see and interact with sonar data, PM is specifically designed to quickly generate datasets with minimal interaction with the software. The idea is that you go out for a day of sonar surveys, bring the data back to your computer, select the desired outputs, click run, and you will have data to look at in a matter of minutes. The data exports can be viewed in a GIS (ArcGIS, QGIS, etc.) or any utility that supports viewing image files (png, jpg, gtif). Metadata CSVs containing information related to vessel position, speed, depth, etc. can be analyzed in Excel, R, and Python.

This handbook is designed to give you a guided introduction to PINGMapper. *I guarantee that it does not cover every capability of the software.* This is because new features and updated workflows are being added regularly. The handbook will give you a minimum competency that will enable you to move forward and experiment with the software on your own datasets. It is designed to be run through sequentially, but you can jump around as necessary.

Compatibility

PM will process sonar recording from the following devices and file types:

- **Humminbird** (SON, DAT, IDX; Xplore, Apex, Solix, Helix, Onyx, 11XX, 9XX)
- **Lowrance** (sl2, sl3)
- **Garmin** (rsd)
- **Cerulean** (svlog; Omniscan 450SS)

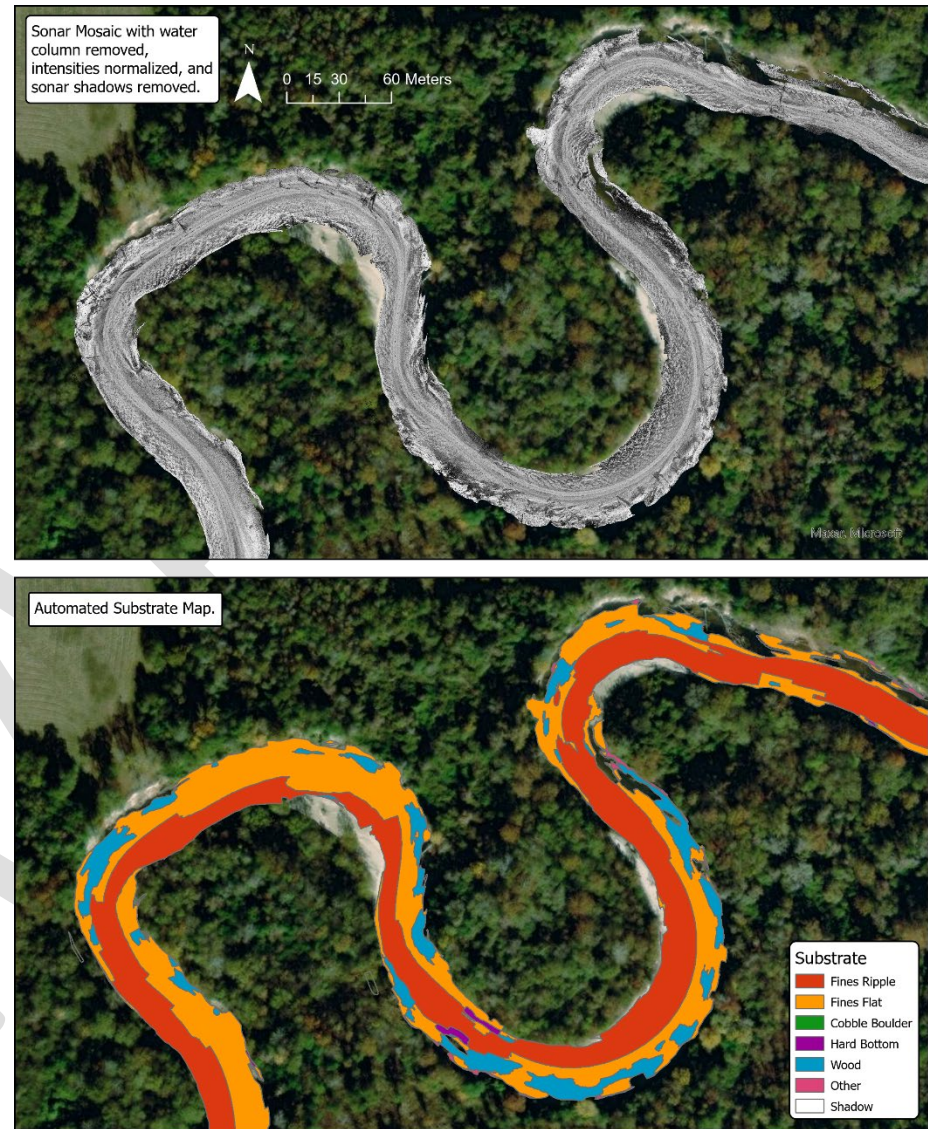


Figure 1 – Sonar mosaic and corresponding substrate maps exported from PINGMapper.

Handbook Requirements

You will need the following to get the most out of this handbook:

- **Computer** running Windows 11/10, MacOS, or Linux (Ubuntu).
- **PINGMapper** (see Appendix A – Installation).
- Ability to **launch PINGWizard** following installation (see Appendix B – Launch PINGWizard).
- **Sonar records/logs** – I provide some example datasets to start and test features, but you will get the most from doing these exercise on your own datasets.
- **GIS Software** – ArcGIS (Pro or Map) or QGIS.
- **A good attitude, willingness to experiment, and plenty of patience** – PINGMapper is open-source and has one primary developer. Bugs are expected; the software will crash. As I like to say, with open-source software, you get what you pay for! Additionally, sonar logs collected with fishfinders have their own quirks that can lead to unexpected results. We are in the wild west here folks – on the precipice of new frontiers! We are using 1) a sensor that wasn't designed for science *to do science*; and 2) a free software for post-processing sonar recordings developed by some dude from Utah who used to be afraid to swim in the local swimming pool. All that to say, things are going to go wrong. Let's give it a try (or multiple tries) anyway!

With all this in mind, let's jump on in!

1) Sonogram Tile Exports (non-georeferenced)

<https://cameronbodine.github.io/PINGMapper/docs/tutorials/SonogramTiles.html>

Overview

Sonograms are 2-dimensional images similar to what you see while collecting data with your fishfinder. While collecting data, side-scan sonar imagery will begin streaming on screen from top to bottom (Figure 2 right panel). Single beam echosounder data (Figure 2 top left) and downscan imagery (Figure 2 bottom left panel) is also streamed, in this case, from right to left. This tutorial specifically covers side-scan sonar.

Side-scan Sonar

What isn't immediately intuitive when examining raw 2-dimensional side-scan images is that there is actually 3 dimensions represented at different locations of the image. This includes (a) the portion of the water column at nadir directly beneath the boat (depth), (b) the portion of the seabed sonified by a side-scan sonar ping across the vessels track (range), and (c) the along-track distance covered by the moving vessel. Successive pings from the side-scan sonar while the vessel is in motion allows the seabed to be imaged.

This “waterfall” of sonar data can be saved to a log by recording the data. PINGMapper can be used to export sonar data from these logs to a variety of formats, including sonograms.

PM can export four different types of side-scan sonograms:

- WCP** : Sonograms with the water column present.
- WCM** : Sonograms with the water column masked (removed).
- SRC** : Sonograms with the water column removed and slant range corrections applied.
- WCO** : Sonograms with the water column only.

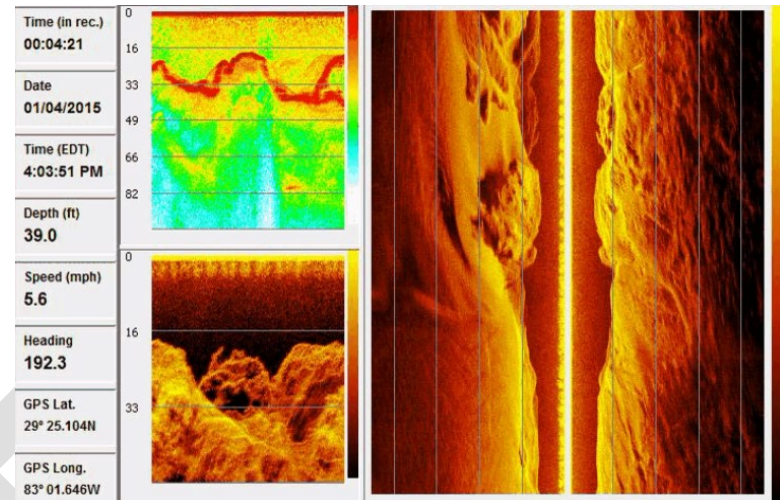


Figure 3 – Simulation of data collection. Animated video available [here](#).

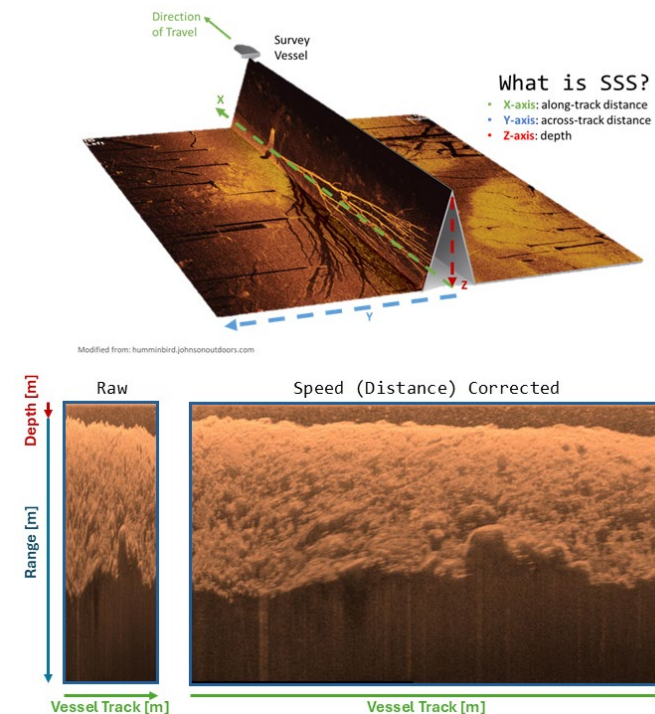


Figure 3 - Interpreting side-scan sonar water column (top); raw vs. speed corrected sonograms.

These four sonograms can be exported as a **raw** (compressed) image or as a **speed corrected** image where the distance of the vessel is used to stretch the image horizontally so that feature dimensions can be accurately measured, ensuring across and along-track pixel dimensions are approximately the same. The sonograms can be further customized by using three other settings in different combinations:

Shadow Removal : Use AI-models to automatically predict shadow regions;

Mask Shadows : Use the shadow predictions to mask out shadows;

Max Crop : Crop out the masked water column and shadow regions.

Let's try exporting all the possible sonograms in their raw form.

Why Sonograms?

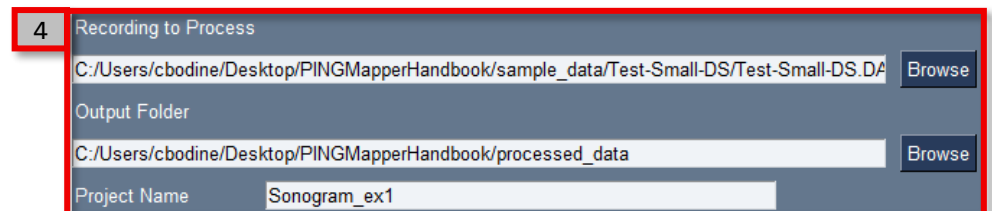
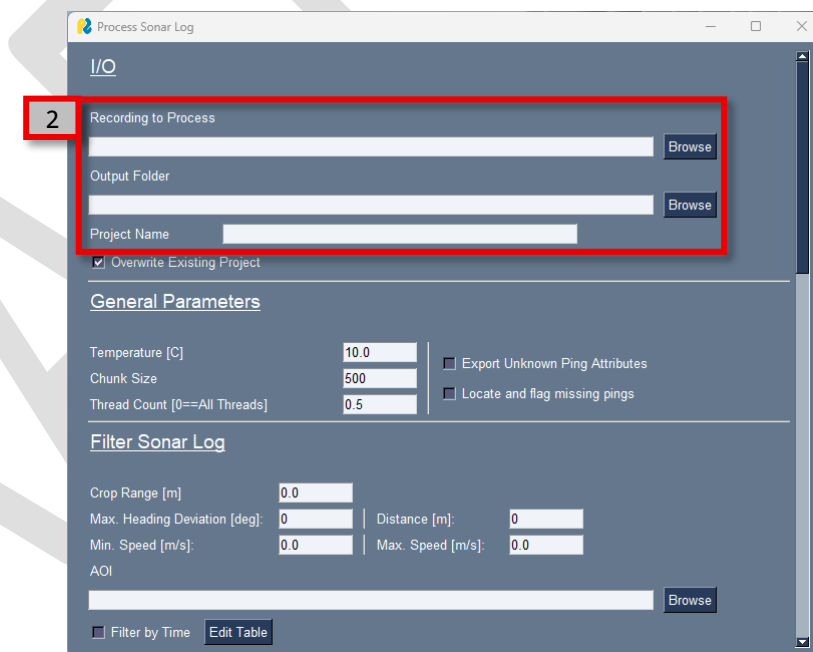
Exporting sonograms provides the quickest way to export sonar imagery. They are non-distorted representations of the surveyed area. They have also been useful for generating model training datasets (Bodine, Buscombe, & Hocking, 2024).

Dataset

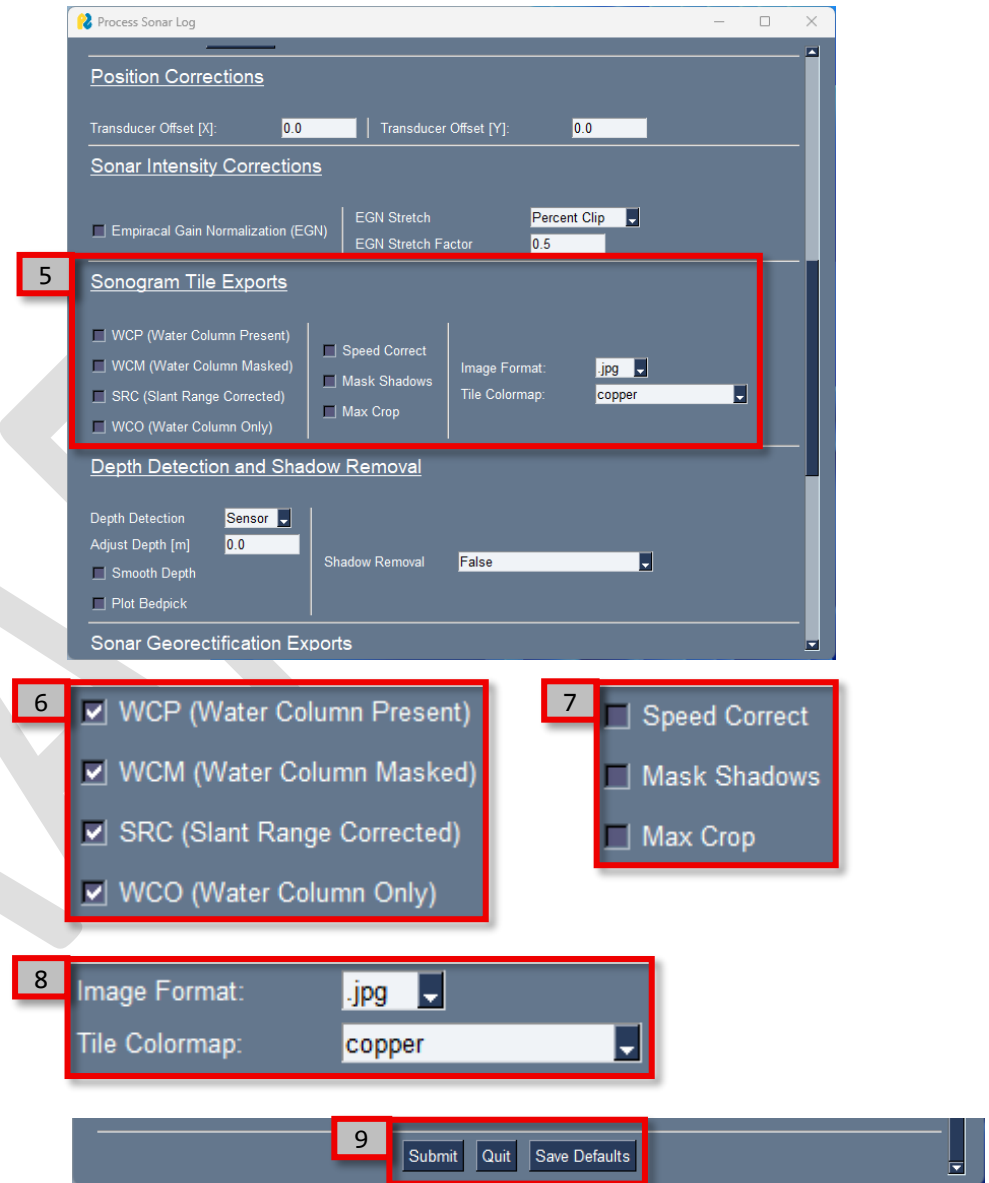
Test-Small-DS [\[Download\]](#) or use your own.

Part 1 – Raw Sonogram Processing Steps

1. Launch the PINGWizard (Appendix A) and select **Single Log**.
2. Use the **Browse** buttons to select the **Recording to Process** and select an **Output Folder**.
 - a. **Recording to Process** – Navigate to the dataset and select the file. In this case, we are selecting the Humminbird DAT file.
 - b. **Output Folder** – Choose the output parent directory.
3. Provide a **Project Name**. This will create a new folder in the **Output Folder** and also prepend all output files with the **Project Name**. Lets call it **Sonogram_ex1**.
4. All selections will be populated to the GUI.



5. Scroll down to the **Sonogram Tile Exports** section.
6. Check the box next to each of the following to export: **WCP**, **WCM**, **SRC**, and **WCO** imagery. We will examine each of these in turn.
7. Leave **Speed Correct**, **Mask Shadows**, and **Max Crop** unchecked for now.
8. Select an **Image Format** and **Tile Colormap**. Check out Appendix C – Colormap Selection for more information on colormap selection.
9. Scroll down to the bottom of the window and select **Save Defaults** (updates defaults based on currently selected parameters). Then click **Submit** to start processing. This will close the GUI. Processing progress can be monitored in the Command Prompt.
10. Processing should be completed in less than 1 minute.



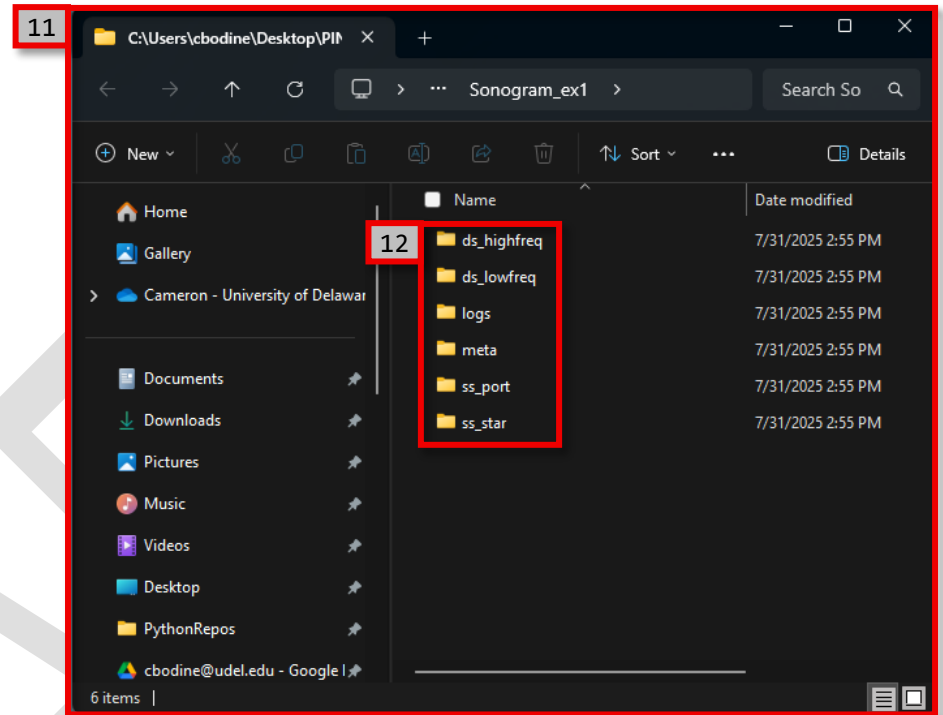
11. Open the File Explorer (Windows) or file browser and navigate to the output folder. This tutorial saved outputs to:

- a. `C:\Users\cbodine\Desktop\PINGMapperHandbook\processed_data\Sonogram_ex1`

12. Several folders have been created. Depending on the parameters set and beams available in the sonar recordings (remember these data are from a Humminbird), you will see varying number of folders. For this particular recording, we see:

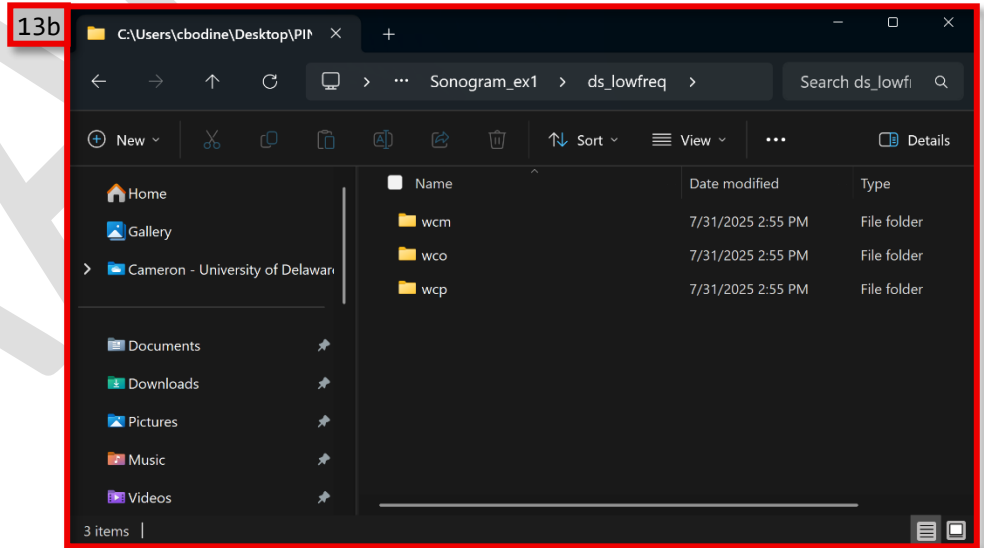
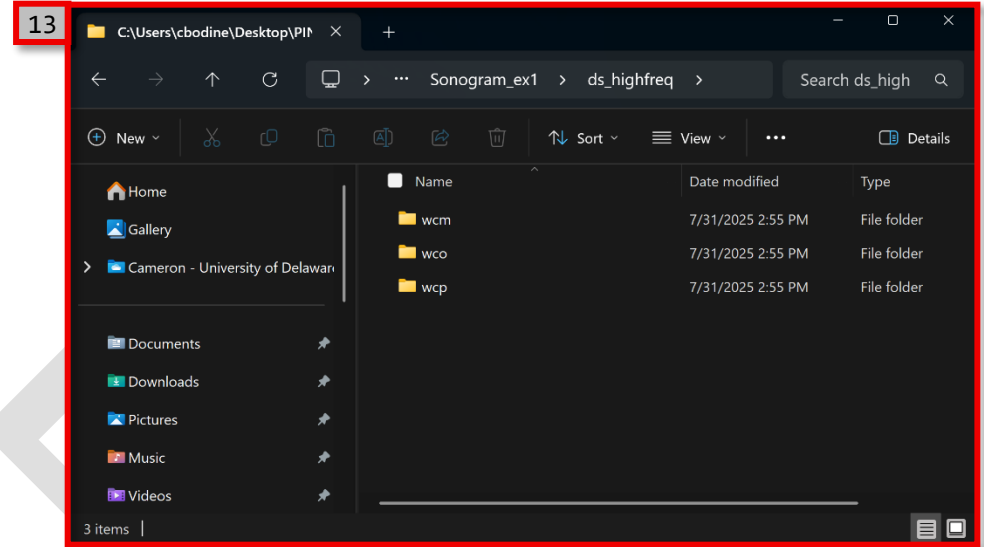
- a. `ds_highfreq` : images from down-scan 2D sonar at 200 kHz.
- b. `ds_lowfreq` : images from down-scan 2D sonar at 83 kHz.
- c. `logs` : PINGMapper process logs.
- d. `meta` : CSVs & PINGMapper meta files for each sonar beam's ping attributes (see [Bodine et al. 2022](#)).
- e. `ss_port` : Portside Side-scan sonar (455/800/1200 kHz depending on survey setting).
- f. `ss_star` : Starboard Side-scan sonar (455/800/1200 kHz depending on survey setting).

Let's examine the contents of the sonar beam folders first.



13. ds_highfreq

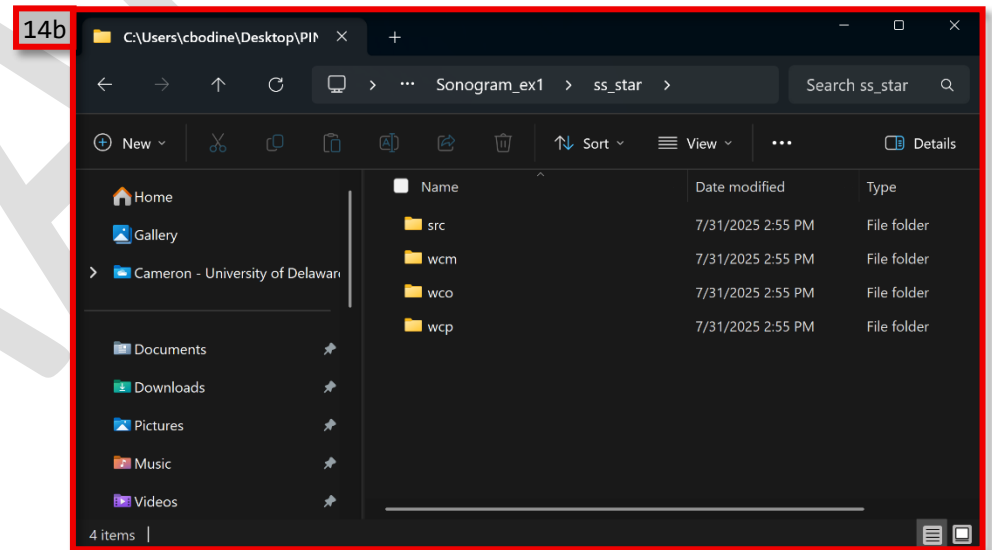
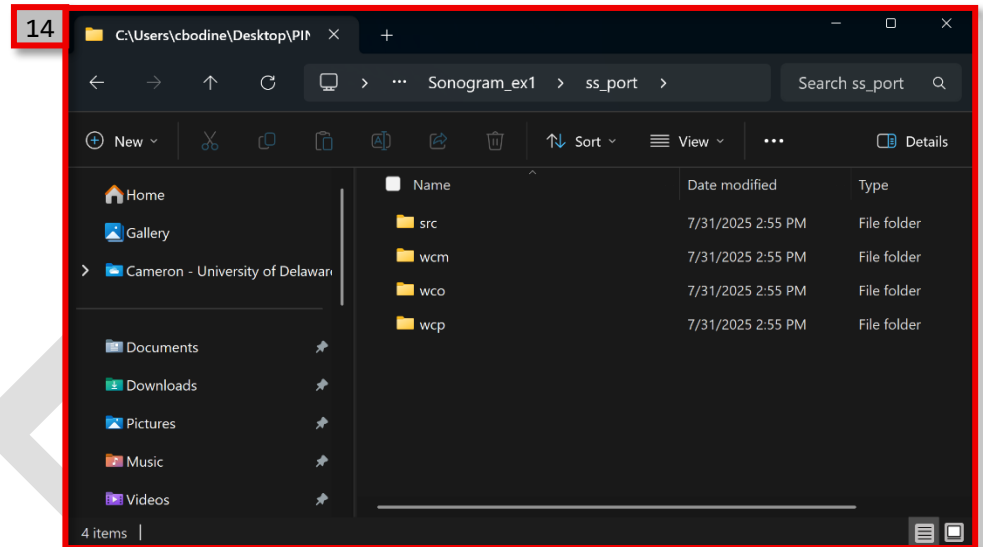
- a. Three folders are visible, based on the selected outputs from step 6: wcm, wco, and wcp. But Cam, we selected 4 outputs, including src. That's right. However, both the ds beams are oriented at nadir and it doesn't make sense to slant-range correct the imagery because the range for these datasets is the depth. Slant-range correction only applies to ss (side-scan) beams (see Bodine et al. 2022 and associated references for more information).
- b. The same folders are also present in the ds_lowfreq folder.



14. `ss_port`

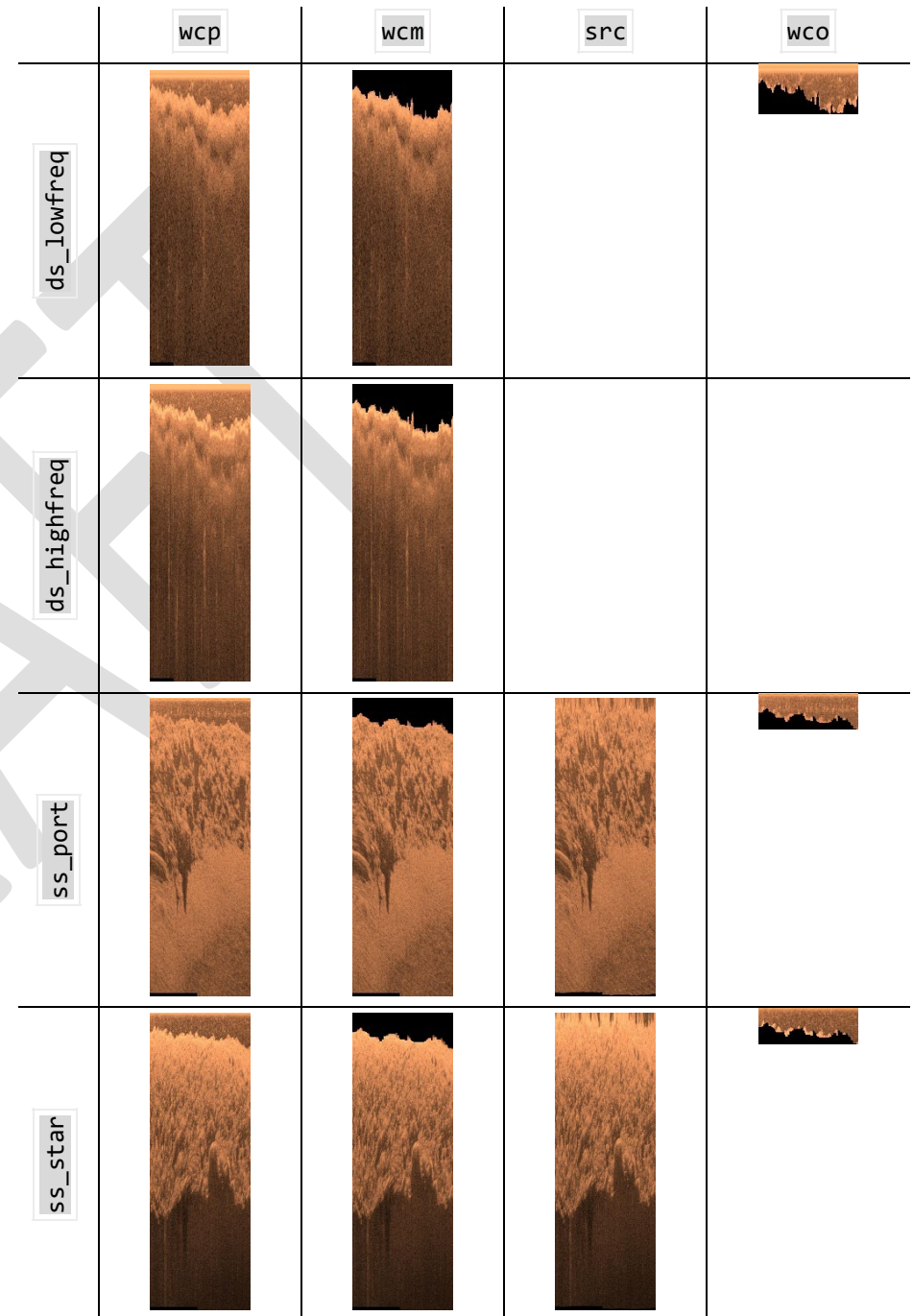
- a. Three folders are visible, based on the selected outputs from step 6: `src`, `wcm`, `wco`, and `wcp`.
- b. The same folders are also present in the `ss_star` folder.

Now let's explore the outputs in each of these folders.



15. Shown at the right are examples from each of the sonar beams (rows) and the same chunk for each of the sonogram types (columns):

- a. `ds_lowfreq`
- b. `ds_highfreq`
- c. `ss_port`
- d. `ss_star`



DRAFT

Sonar Georectified Exports (georeferenced)

DRAFT

Thread Count & Chunk Size: Effects on RAM usage and processing speed

Explanation on chunk size

When to export sonograms vs gtiffs

DRAFT

Filtering: Heading Deviation

DRAFT

Filtering: AOI

DRAFT

Filtering: Time

DRAFT

EGN Corrections to Intensity

DRAFT

Substrate Mapping

DRAFT

Appendix A – Installation

PINGMapper is a software (i.e. package) written in Python. PINGMapper uses a variety of Python packages (NumPy, Pandas, Tensorflow, etc.), or dependencies, that allow you to process Humminbird® sonar recordings and generate a variety of GIS datasets.

PINGMapper uses conda to ensure the installation is configured correctly. Specifically, conda is used to create a virtual environments called ping, a container storing all the correct versions of of the required dependencies, that ensures PINGMapper runs as expected.

Conda comes in several flavors, however, we will use Miniforge as it is free for anyone to use.

[Miniforge](#): Free for all;

This tutorial will demonstrate how to install and configure PINGMapper. After installing Miniforge, we will install and run PINGInstaller. PINGInstaller automatically creates the ping environment, installs the appropriate packages from the PING Ecosystem (PINGMapper, PINGWizard, PINGVerter, etc.), and other necessary dependencies.

Let's get started!

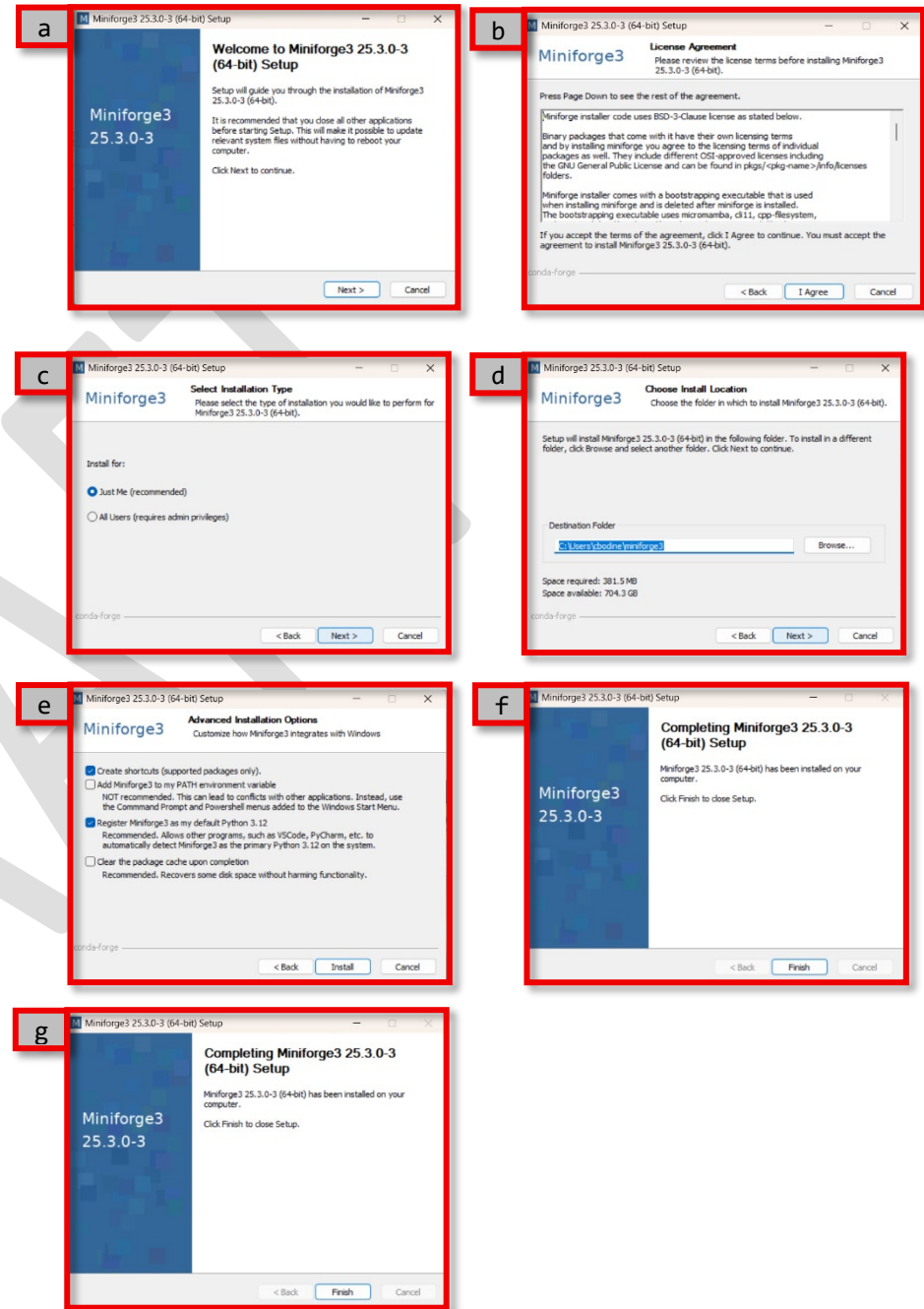
Procedure

1. Install Miniforge

Go to the [Miniforge Website](#) and download the software. Choose the appropriate installer for your computer's operation system. This tutorial was made on a Windows machine but the process should be similar on other operation systems. Click the file and it will download to your Downloads folder, or you can right-click and select "Save Link As..." and choose an alternative location to save the install file.

Double click the file to begin the installation file. This will open an installation window.

- Click Next and you will see the license agreement (2). After reviewing the license agreement, you must select **I Agree** to continue with the installation.
- After you agree, you will have an option to install Miniconda for **Just Me** or **All Users**. You want to install Miniforge in your user folder so that you have the necessary permissions to install the Python dependencies, so select **Just Me** and click **Next**.
- Accept the default installation location and click **Next**.
- This will open the Advanced Installation Options window. We will accept the default options and click **Next**.
- Once installation is complete, you will see the window indicating Miniconda was successfully installed. Click **Finish** to close the window.



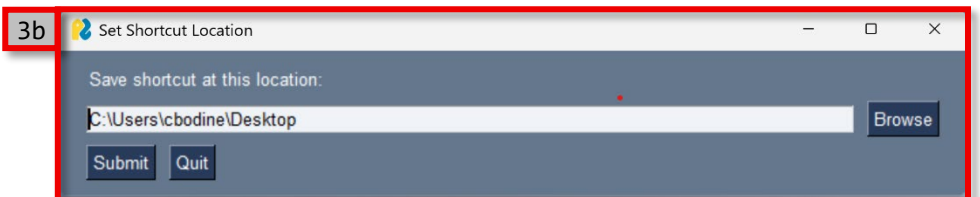
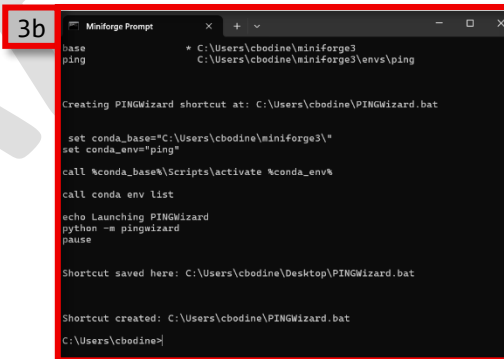
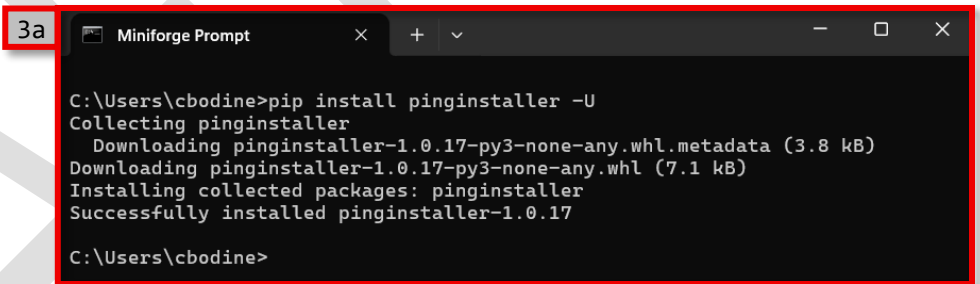
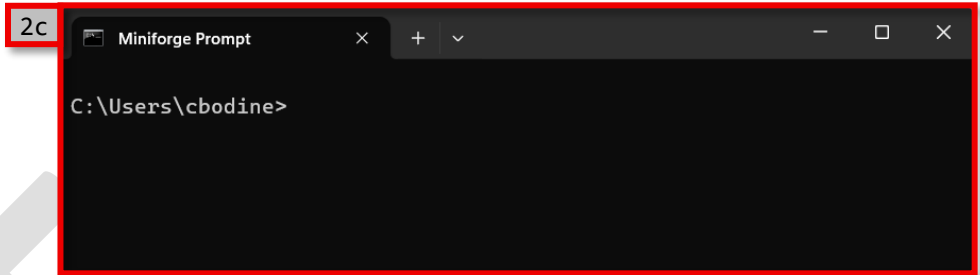
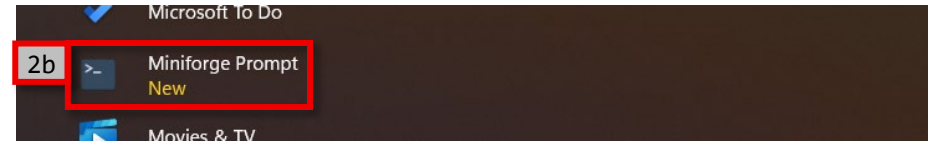
2. Open Miniforge Prompt

- a. Now for the scary part! We are going to open a command prompt so that we can submit a series of commands to Miniforge. If you want to gain some familiarity with navigating with the prompt, you can watch [this video](#).
- b. Miniforge is a command prompt that we will use to install and run PINGMapper. On Windows, click the start button and scroll through your installed applications until you find Miniforge Prompt.
- c. Click the icon to open the prompt.

3. A package called PINGInstaller is used to install and setup PINGMapper. We will install PINGInstaller with the following command and pressing **Enter** :

- a. `pip install pinginstaller -U`
- b. Installation will take approximately **5-10 minutes**. You can monitor the progress in the prompt.
- c. At the end of the install process, a window will prompt where to save the `bat` or `sh` shortcut file. Browse to the desired location and click Submit.

4. PINGMapper is now ready to go. B) Launch PINGWizard a new interface for PINGMapper, to start processing data.



Appendix B – Launch PINGWizard

<https://cameronbodine.github.io/PINGMapper/docs/gettingstarted/PINGWizard.html>

Overview

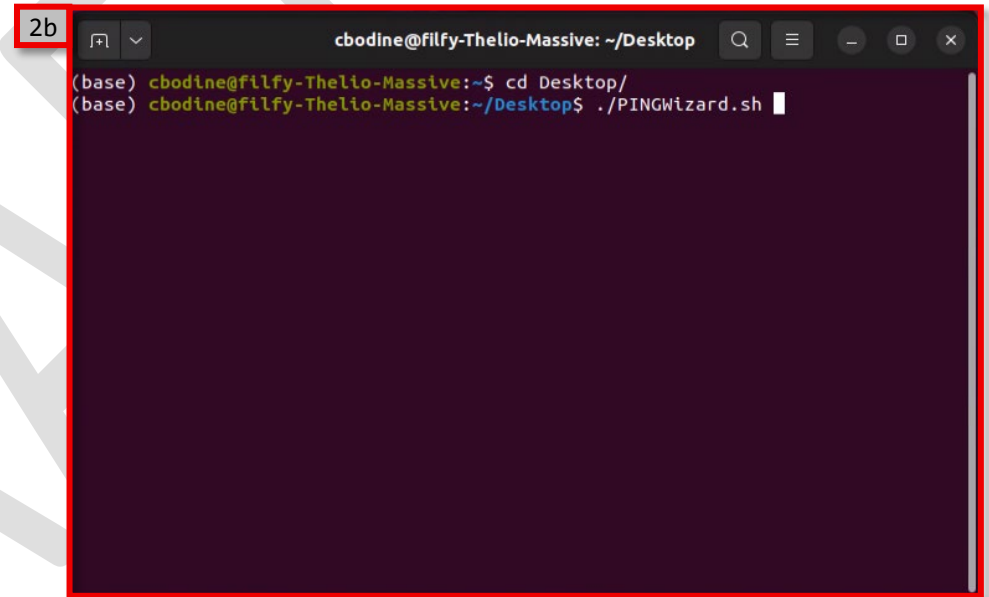
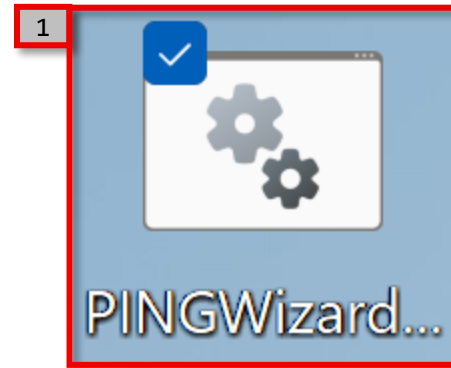
PINGWizard is a light-weight interface for launching PINGMapper utilities, including running the tests, processing a sonar log, batch process multiple sonar logs, and updating the installation.

There are two options for launching PINGWizard: a) with a shortcut or b) command prompt.

Option A – Shortcut

During installation, you were prompted to select a location to save a batch (Windows) or bash (Linux/Mac OS) shortcut file. This file contains the commands to activate the `ping` conda environment and run PINGWizard.

1. On Windows, simply double click the PINGWizard.bat file.
2. On Linux/Mac OS, open a command prompt, change directory to where you saved the shortcut, and launch the bash script by entering the following and press `Enter`
 - a. `./PINGWizard.sh`



Option B – Conda Command Prompt

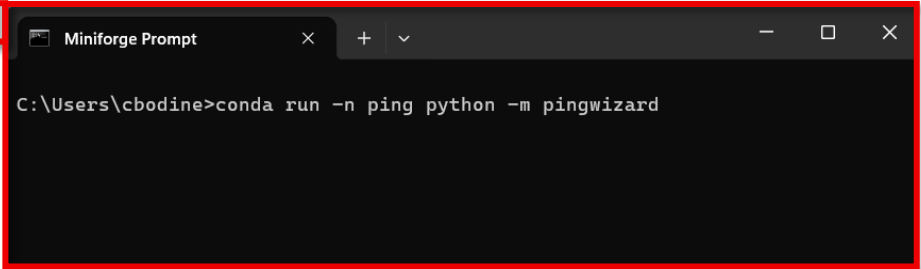
1. Open the Conda Command Prompt used during installation.
Activate the ping environment and launching PINGWizard by entering the following and pressing **Enter**:

- a. `conda run -n ping python -m pingwizard`

PINGWizard

PINGWizard will launch and present a menu of buttons to run various PINGMapper utilities. As new utilities are developed, they will be accessible from the wizard, ensuring easy access to the latest utilities.

1a



```
Miniforge Prompt
C:\Users\cbodine>conda run -n ping python -m pingwizard
```



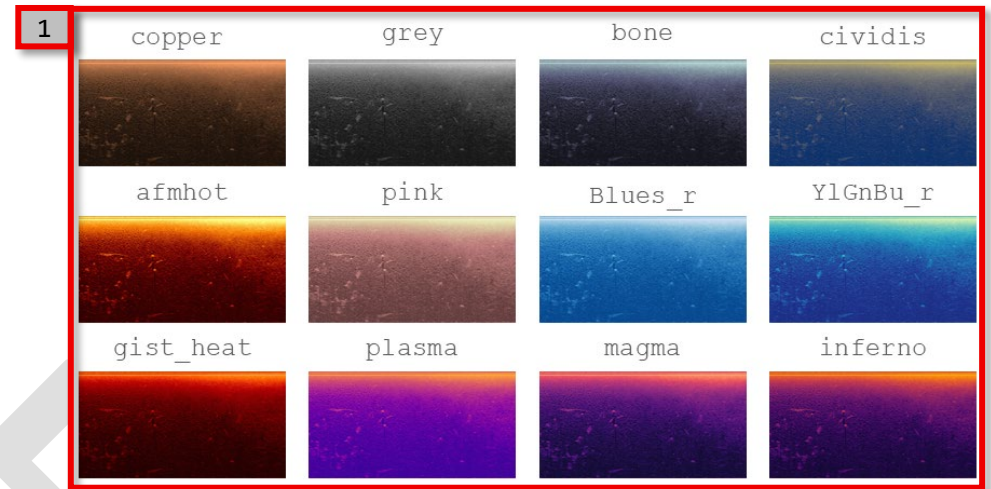
Appendix C – Colormap Selection

Overview

PINGMapper uses [Matplotlib colormaps](#) to color sonar image exports. If the colormap needs to be reversed, append `_r` to the colormap name.

Recommended Colormaps

1. Example of recommended colormaps. The first row are highly recommended.



Appendix D – Errors & Logs

DRAFT

References

- Bodine, C. S., Buscombe, D., & Hocking, T. D. (2024). Automated River Substrate Mapping From Sonar Imagery With Machine Learning. *Journal of Geophysical Research: Machine Learning and Computation*. doi:10.1029/2024JH000135
- Bodine, C. S., Buscombe, D., Best, R. J., Redner, J. A., & Kaeser, A. J. (2022). PING-Mapper: Open-Source Software for Automated Benthic Imaging and Mapping Using Recreation-Grade Sonar. *Earth and Space Science*. doi:10.1029/2022EA002469

DRAFT