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

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)

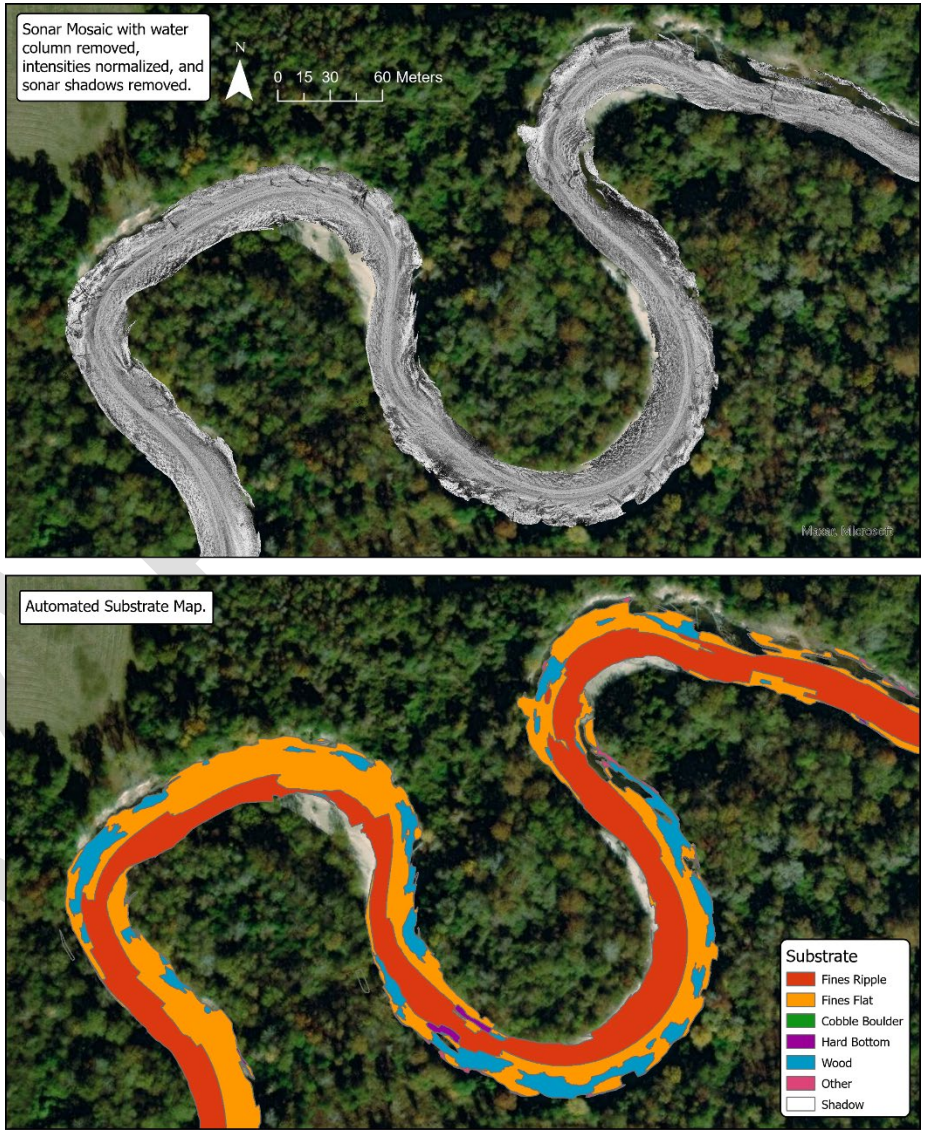


Figure 1 – Sonar mosaic and corresponding substrate maps exported from PINGMapper.

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).
- Abilit 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 3 right panel). Single beam echosounder data (Figure 3 top left) and down-scan imagery (Figure 3 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 esonified 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.

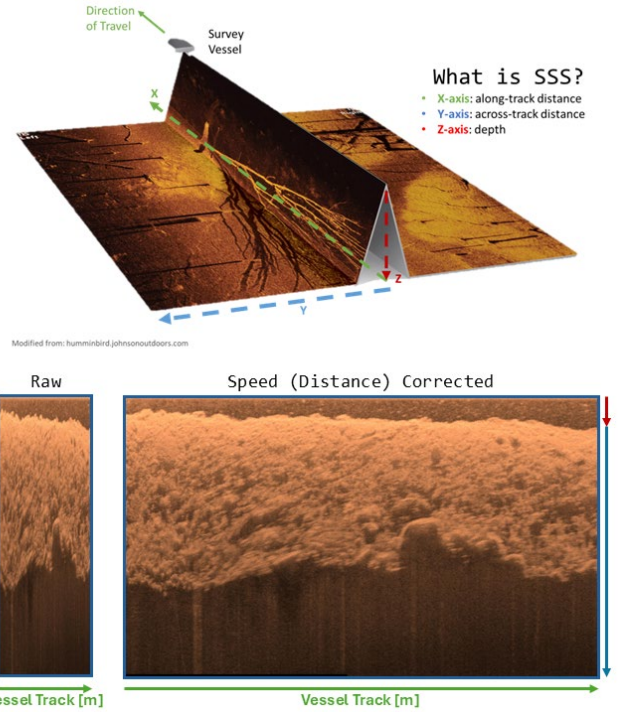


Figure 3 - Interpreting side-scan sonar water column (top); raw vs. speed corrected sonograms.

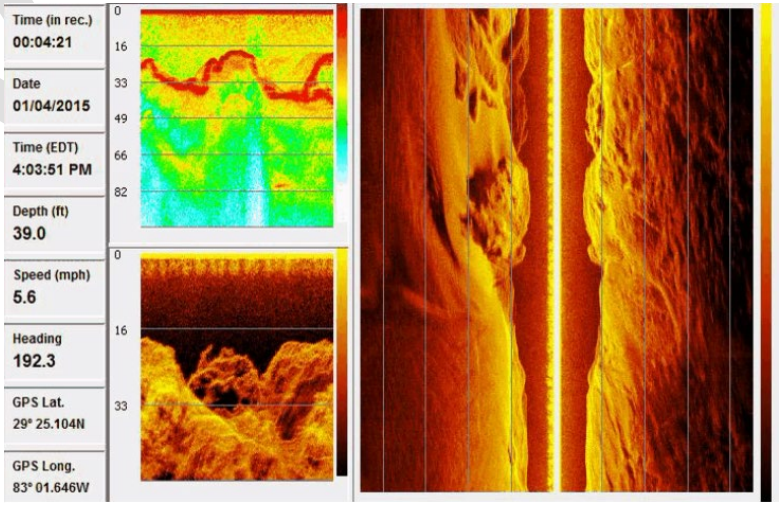


Figure 2 – Simulation of data collection. Animated video available here.

These four sonograms can be exported either as `raw` (compressed) or as `speed-corrected` images, in which the distance the vessel travels is used to horizontally stretch the imagery. This correction ensures that feature dimensions are accurately represented, with approximately equal across- and along-track pixel resolutions. 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.

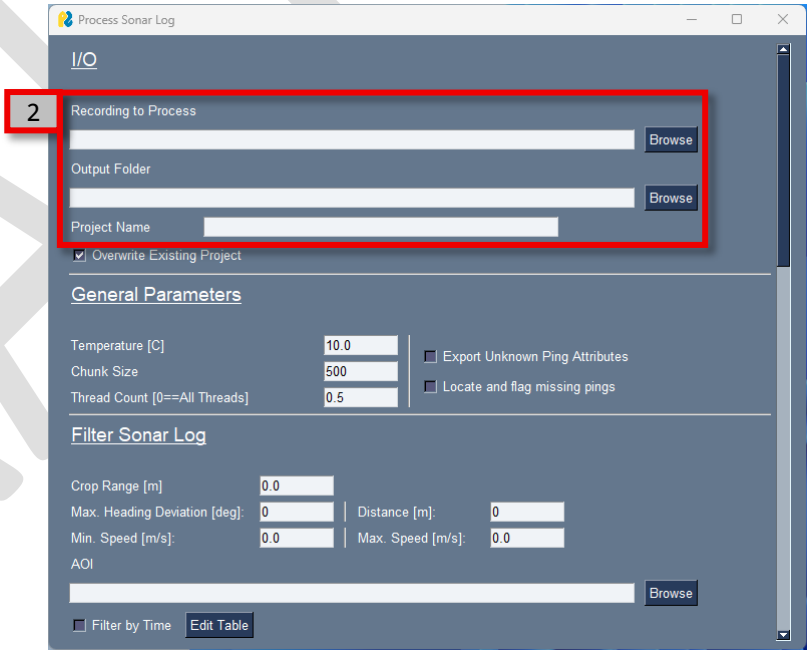
Why Sonograms?

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 [1].

Dataset

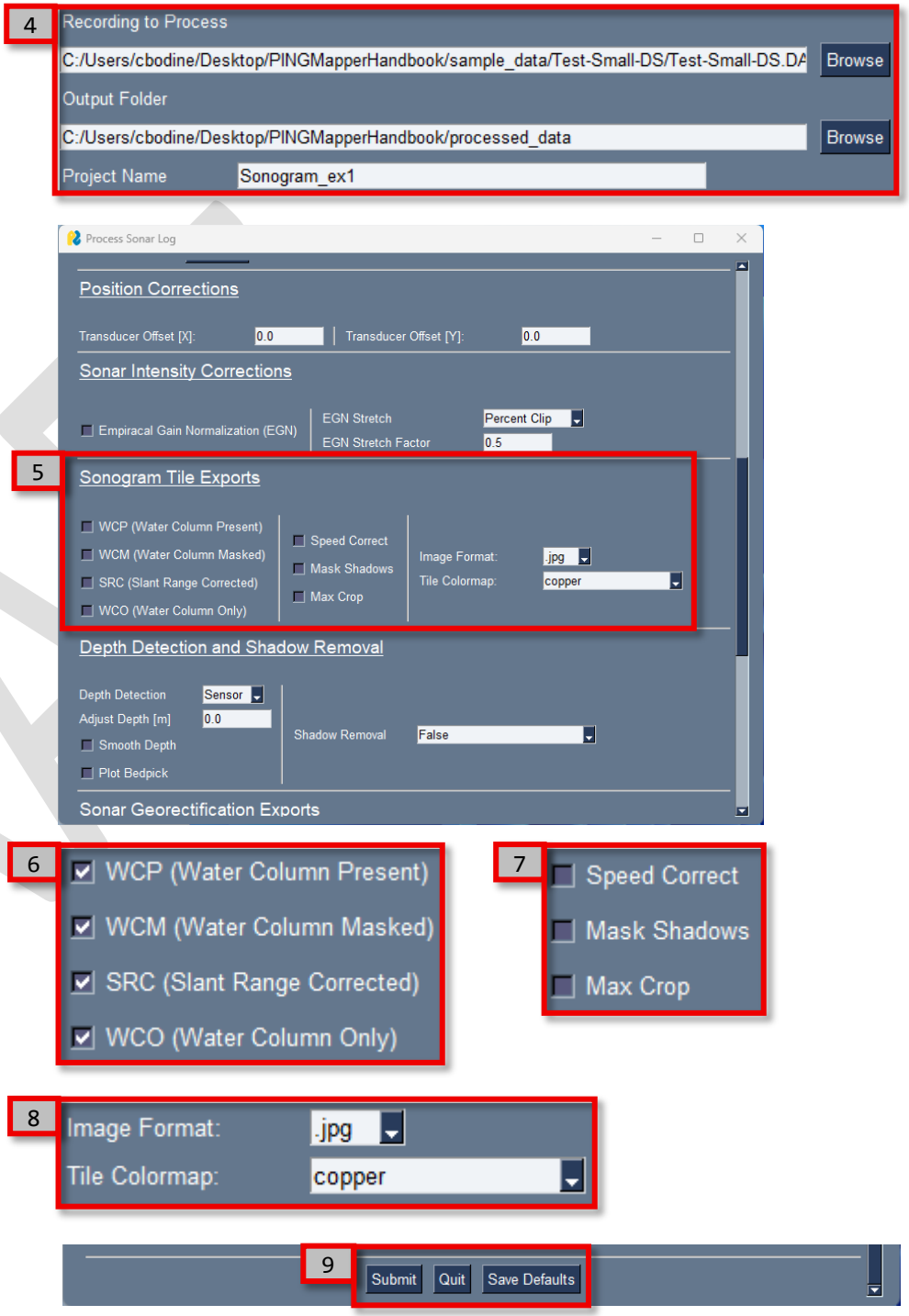
Test-Small-DS [Download] or use your own.

A) – Export Raw Sonogram Processing

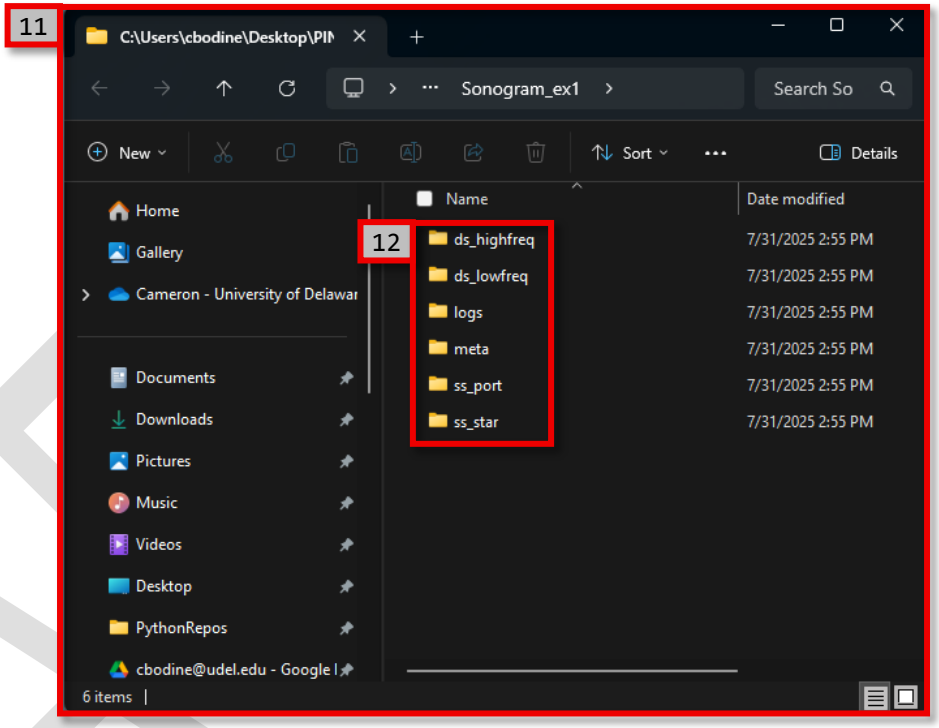


1. Launch the PINGWizard (Appendix A) and select `Single Log`.
2. Use the `Browse` buttons to select the `Recording to Process` and select an `Output Folder`.
   a. `Recording to Process` – Navigate to the dataset and select the file. In this case, we are selecting the Humminbird DAT file.
   b. `Output 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`. Let's call it `Sonogram_ex1`.

4. All selections will be populated in 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.

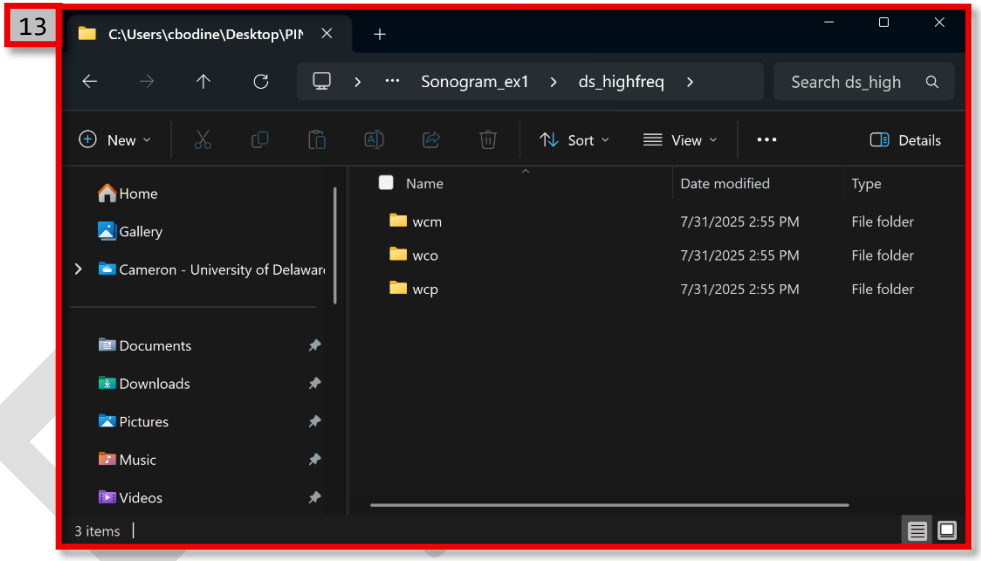


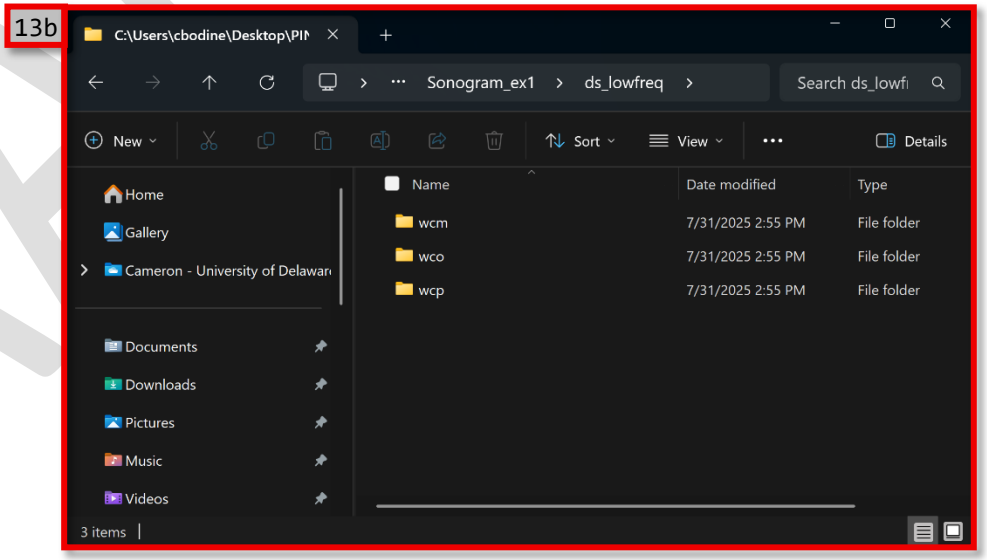
11. Open the File Explorer (Windows) or file browser and navigate to the output folder. This tutorial saved outputs to:
    a. `C:\Users\cbodine\Desktop\PINGMapperHandbook\processed_data\Sonogram_ex1`
12. Several folders have been created. Depending on the parameters set and beams available in the sonar 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 [2].
    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 Dr. 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 [2] and associated references for more information).
    b. The same folders are also present in the `ds_lowfreq` folder.



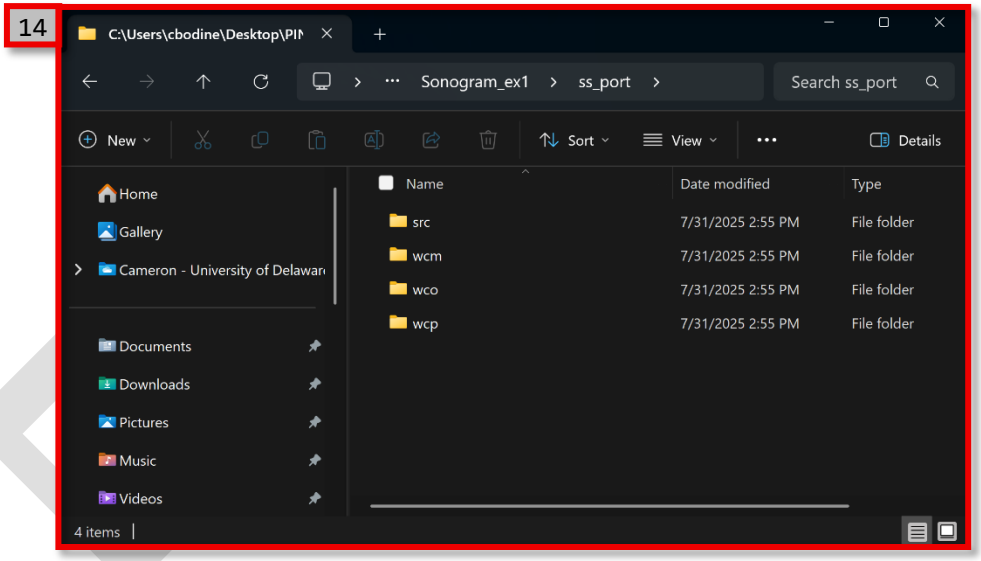


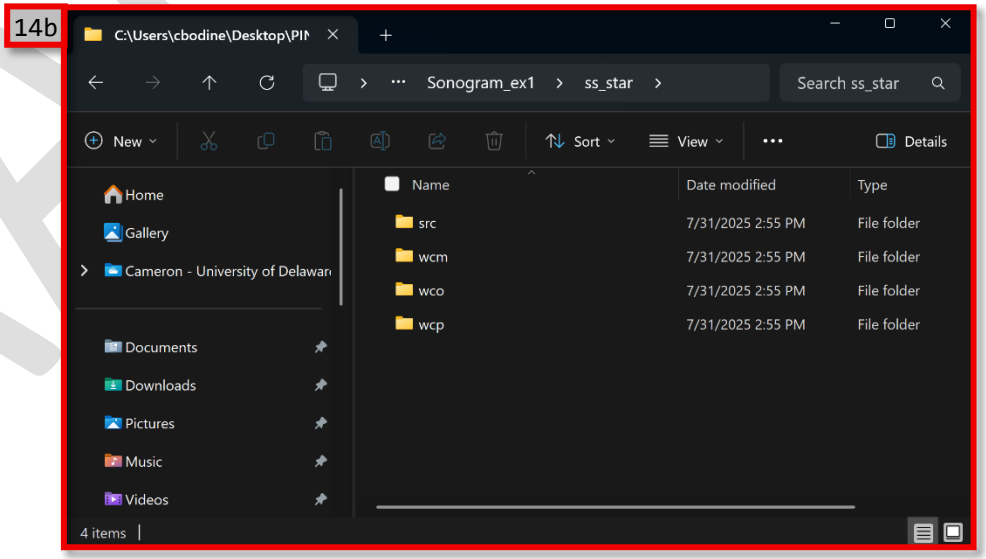
14. `ss_port`

    a. Four folders are visible, based on the selected outputs from step 6: `src`, `wcm`, `wco`, and `wcp`.

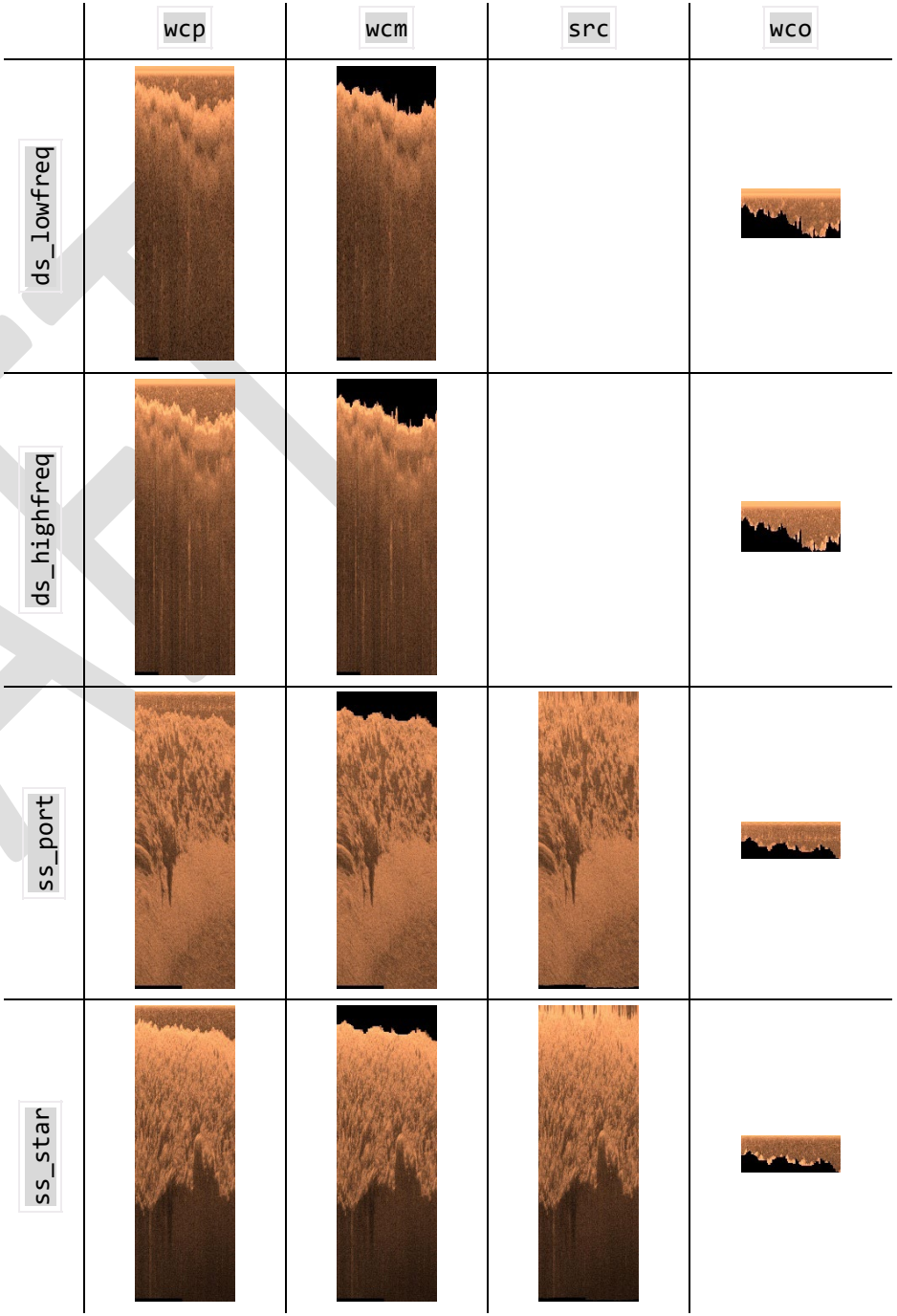
    b. The same folders are also present in the `ss_star` folder.

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



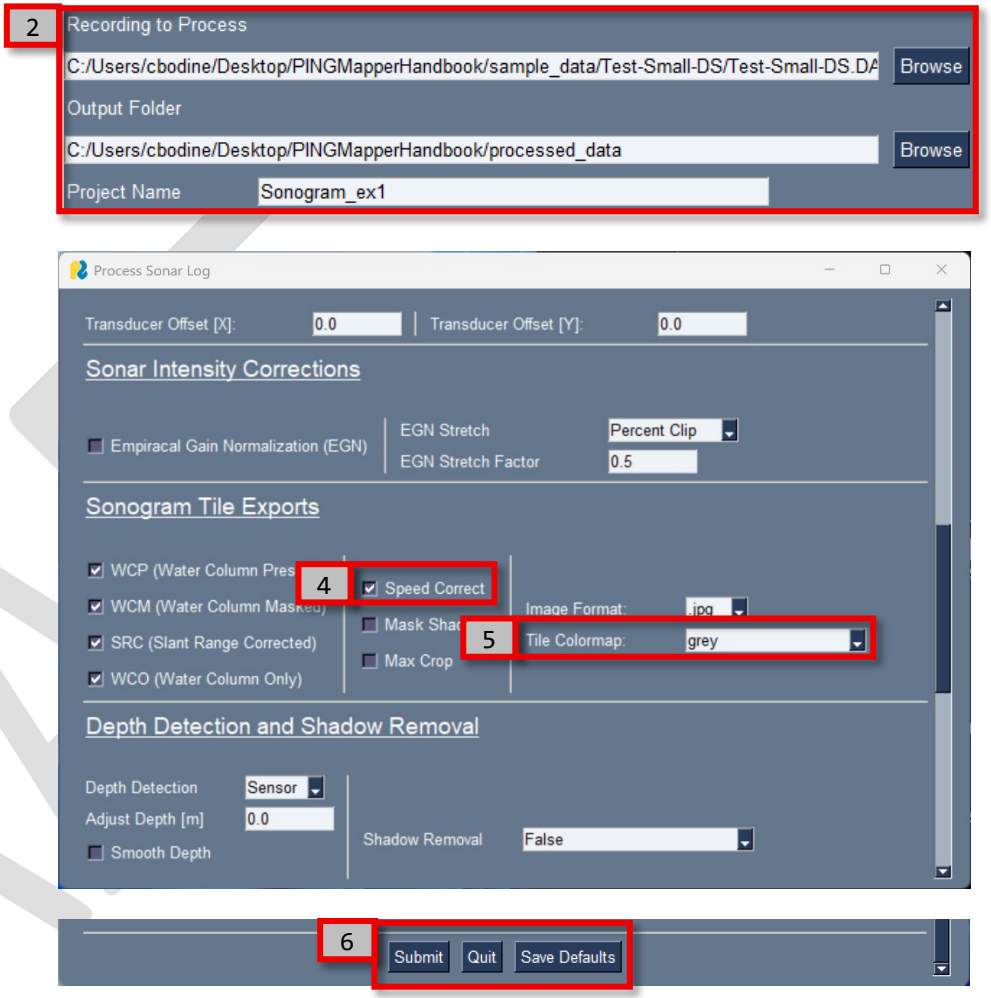


15. Shown at the right are examples from each of the sonar beams (rows) and the same chunk for each of the sonogram types (columns):
    a. `ds_lowfreq`
    b. `ds_highfreq`
    c. `ss_port`
    d. `ss_star`

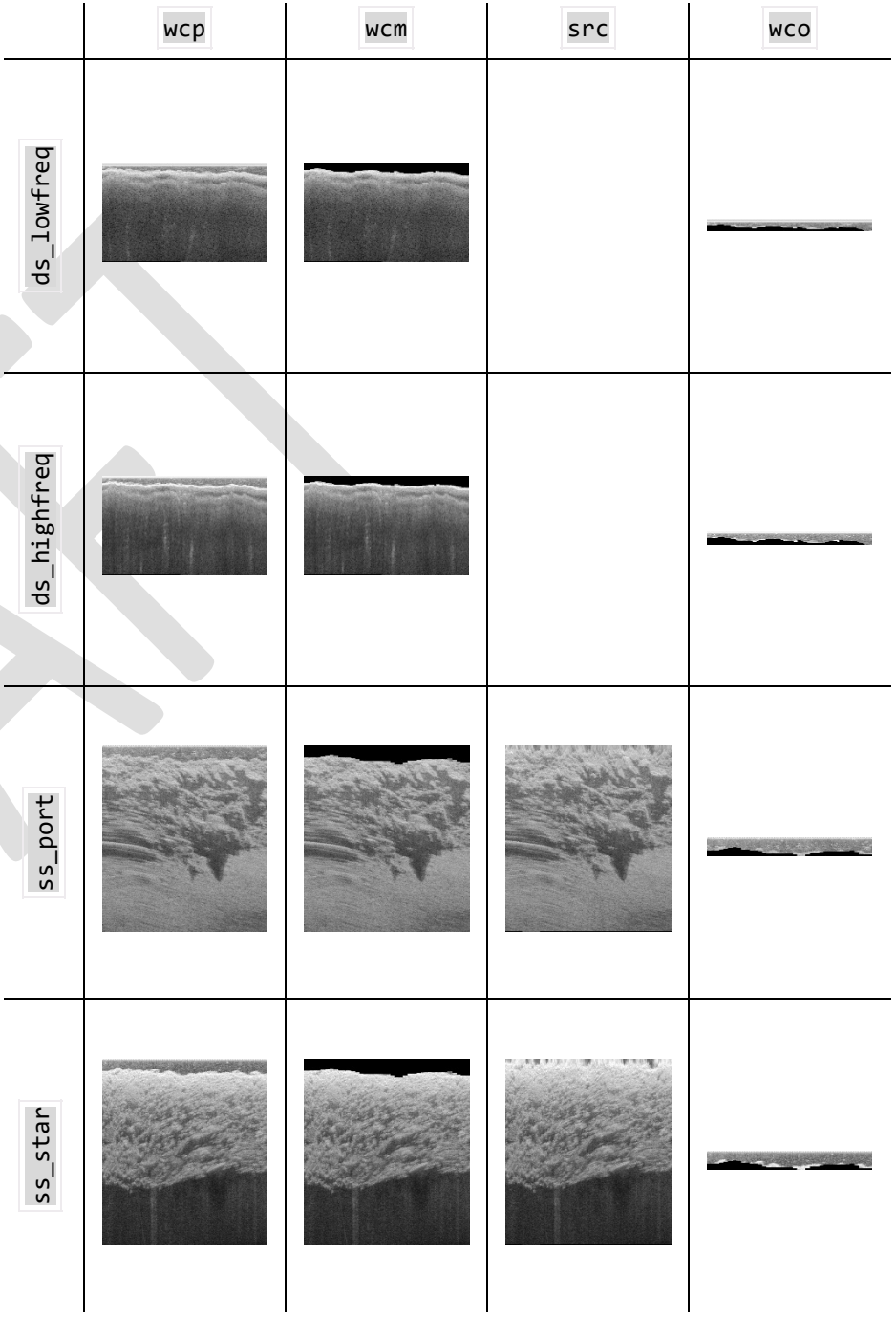


B) – Export Speed Corrected Sonograms

1. Launch the PINGWizard (if not already open; Appendix A) and select `Single Log`.
2. Since we saved out processing defaults previously, we should already see the small dataset selected in `Recording to Process` and the `Output Folder` will remain the same.
3. Update the `Project Name` to `Sonogram_ex1b`.
4. Scroll down to `Sonogram Tile Exports` and check `Speed Correct`. This will use the vessel speed to resize the output sonograms such that the along- and across-track pixel dimensions are approximately equal.
5. Let's be adventurous and choose a different color. Besides copper, grey is another all-time favorite.
6. Scroll to the bottom of the window and select `Submit` to begin processing. Again, processing should take less than 1 minute.



7. 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`



Lesson Complete!

Congratulations! You just took the first steps towards becoming a PINGMapper Jedi Master. Gold stars all around!

But wait!, you say. *There were other options, more boxes to check. What do they do?*

This is where you explore on your own! Or you can head to the Appendices and see some examples:

- Appendix E – Sonogram tile export examples

Why are there so many options? Doesn't it get confusing?

Don't ask why, ask why not! And yes, very confusing. But that's how it goes.

2) Sonar Georectified Exports (georeferenced)

Overview

Sonograms can be georectified and mosaicked and brought into a GIS for additional analysis. Either the sonar sensor depth estimates, or the automated depth estimates can be used for water column removal and slant range correction.

`WCP` : Water Column Present

Mosaics with the water column present can also be exported, however, these mosaics are not geographically correct due to the presence of the water column. Remember that weird accordion folded sonogram we talked about earlier as shown in Figure 2? The water column really is a Z-dimension (vertical) in comparison to the X- and Y-dimensions that correspond to geographic coordinates. By keeping the water column in the 2D sonogram, this forces the bed pixels away from the trackline. Nevertheless, these types of imagery are very beneficial for fish enumeration studies. They are also helpful in substrate analysis as bed structure beneath the boat can be seen. However, it is not recommended to delineate substrates on these images as resulting maps will misrepresent substrate coverage and area.

Figure 4 - Sonar mosaic with the water column present (i.e. preserved).

`WCR` : Water Column Removed

Mosaics with the water column removed (e.g. slant-range corrected; Figure 5) are spatially accurate representations of the imaged area. Well, almost. Really, they are an approximation of feature locations in a scanned area that must be geometrically corrected based on a big assumption: a flat bottom assumption! Nobody wants a flat bottom (or do they?) but that's what we have to work from.

Figure 5 - Sonar mosaic with the water column removed (i.e. slant-range corrected)

What are we even talking about here…? Let's explore the concept of slant-range. When you go into the field and collect side-scan imagery, you will set a range. This range is actually a time that the sonar will listen for returns after emitting a ping. The distance a ping travels

through water varies by the density (i.e. salinity) of the water. That's why you select the type of water body you are surveying: salt, brackish, and fresh. Each configuration is based on an assumed average speed of sound in water. This value informs the duration the receiver actively listens for return signals, determined by the range specified for the survey.

That's great, but what about slant-range? Right! When the ping is emitted, it is an envelope of sound constrained by a vertical angle (water surface to nadir) and a horizontal angle (along-track). Side-scan has a very wide vertical angle (nearly 90°) and a very narrow horizontal angle (Figure 6). Within this envelope, sound is moving in all directions at once. The distance the sound travels is the shortest path between the transducer and the feature being imaged, making it a slant-range.

This is where geometric corrections and a flat bottom assumption come into play. Slant-range correction is the process of calculating the true horizontal range to the features surveyed. It is a simple trigonometry exercise. We can create a right triangle where the hypotenuse (long edge opposite the right angle) is the slant-range, the depth to the surveyed feature is the adjacent side, leaving one side unknown – this is the range. Since we don't know the depth to the feature, we assume that it is the same as the depth beneath the boat (flat bottom assumption) and use that value instead. Plug it through the Pythagorean theorem, and voilà – we have the range! Of course, survey areas will not have the same depth throughout, so we use the depth beneath the boat as a replacement, generating an imperfect geometric correction known as slant-range correction!

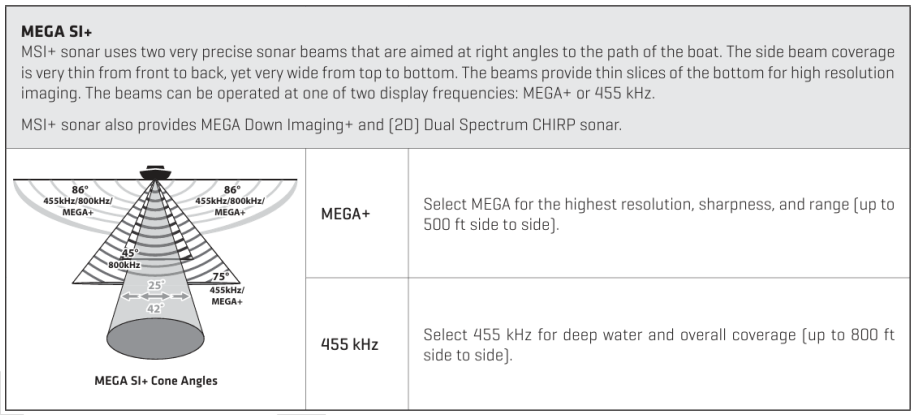


Figure 6 - Beam coverage for Humminbird transducers(From Humminbird Operations Manual).

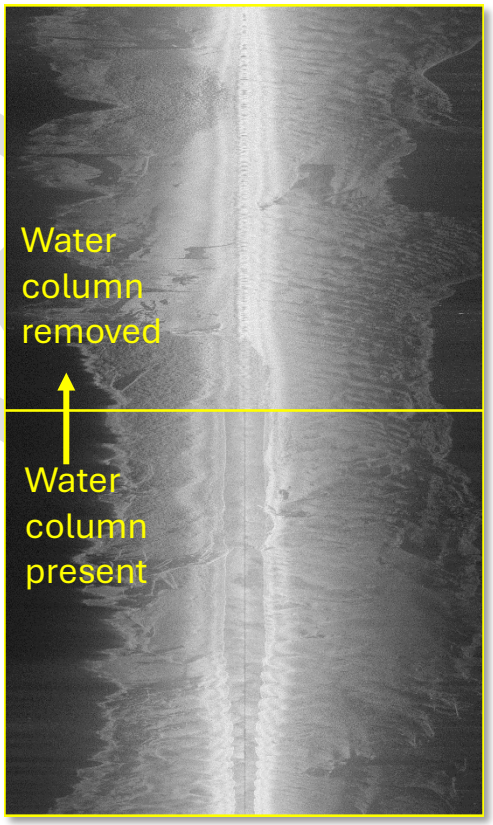


Figure 7 - WCP vs WCR; Adapted from Bodine Dissertation Defense.

Rubber Sheeting vs. Heading/COG Rectification

Rectification is the process of tacking an image that looks like a sonagram, apply some type of transformation to the pixels of the imagery, that results in each of those pixels aligning with other georeferenced datasets. This process requires the GPS coordinates of the transducer, heading or course-over-ground (COG) of the vessel, and estimates of range. This information and simple trigonometry enable calculation of coordinates for the sonar returns.

When PINGMapper was first introduced, it used an approach called `rubber sheeting` to warp a sonogram into a georeferenced image using a Piecewise Affine Transform (Figure 8 and Figure 9 left panel). If you want to learn more about this process, see [2]. While rubber sheeting is fast to process, it can have undesired effects, like compressing the features, making two targets (Figure 9 1b) appear to be one (Figure 9 1b), or spreading features, causing a target to appear larger (Figure 9 2a) then it actually is (Figure 9 2b).

Since the release of PINGMapper v4.2, a second rectification option was added. Using either the vessel `Heading` or `COG`, the coordinates of every sonar return is calculated, resulting in improved spatial accuracy (Figure 9 left panel). This comes at a computational cost due to interpolation, resulting slower processing speed.

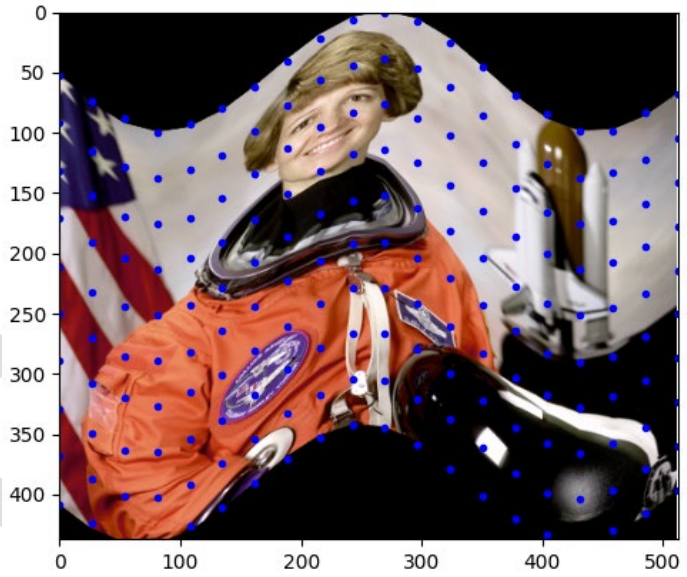


Figure 8 - Example of rubber sheeting using a Piecewise Affine Transform. Retrieved from Sci-Kit Image.

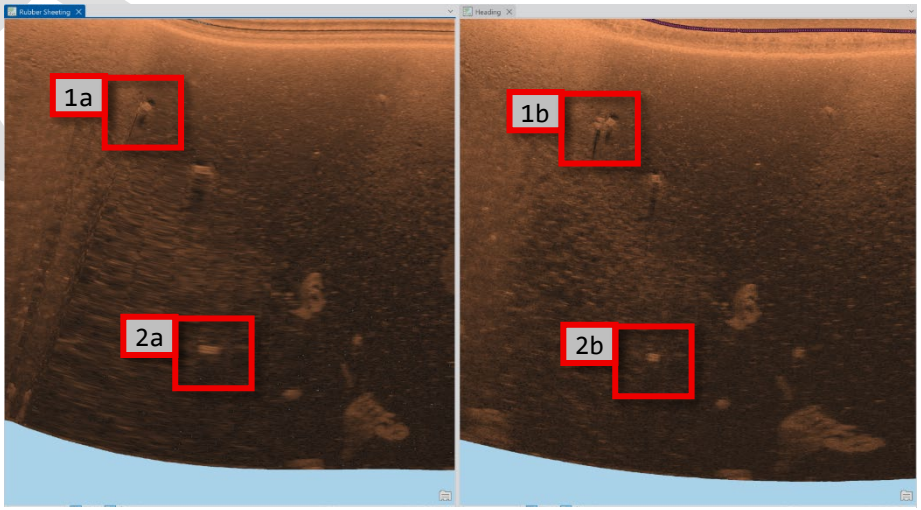


Figure 9 - Rectification with rubber sheeting (left) compared to heading (right).

TL;DR

- Export `WCR` mosaics if identification of features or structure in water column is critical.
- Export `WCP` mosaics for spatial accuracy and mapping.
- Use `Rubber Sheeting` when you need to quickly view georeferenced data and/or spatial accuracy is not critical.
- Use `Heading` or `COG` when correct positioning, target enumeration, or area calculations are critical.

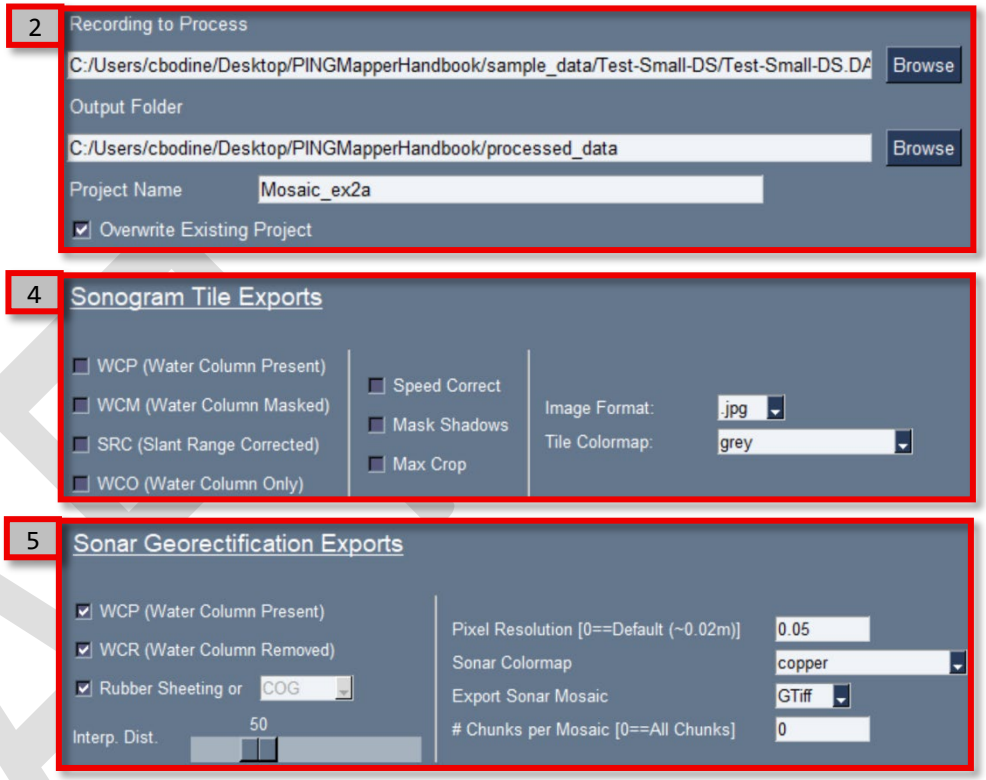
Wow that seemed like an unnecessary amount of information for a data processing guide… You are right on that one! Let's get to the data processing exercise!

Dataset

Test-Small-DS [Download] or use your own.

A) Export Georeferenced Mosaics

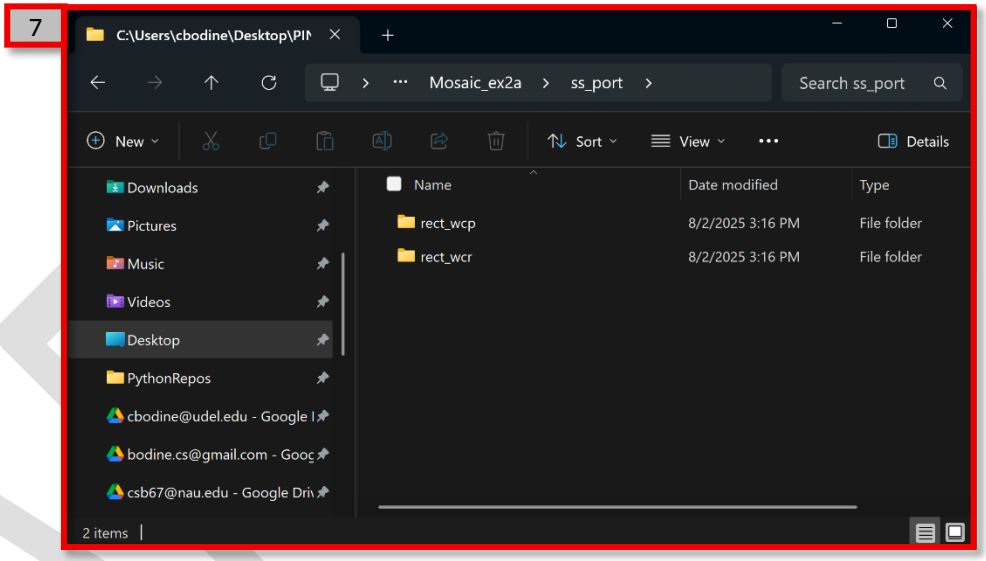
1. Launch the PINGWizard (if not already open; Appendix A) and select `Single Log`.
2. Since we saved out processing defaults previously, we should already see the small dataset selected in `Recording to Process` and the `Output Folder` will remain the same. You can always select new input and output options with `Browse`.
3. Update the `Project Name` to `Mosaic_ex2a`.
4. We saved our default selections during the previous exercise. As we scroll down to `Sonogram Tile Exports`, there are checked parameters. To avoid exporting the sonograms again, uncheck all the boxes.
5. Scroll down to Sonar Georectification Exports and check or select the following:
   a. `WCP`: This will export `water column present` georeferenced gtiffs for each chunk.
   b. `WCR`: This will export `water column present` georeferenced gtiffs for each chunk.
   c. `Rubber Sheeting`: Rectify sonar chunks with `rubber sheeting` workflow.
   d. `Pixel Resolution`: Set to `0.05`. Larger values generate low-resolution images with small file sizes. Small values generate high-resolution images with large file sizes.
   e. `Sonar Colormap`: Choose what you like (Appendix C – Colormap Selection).

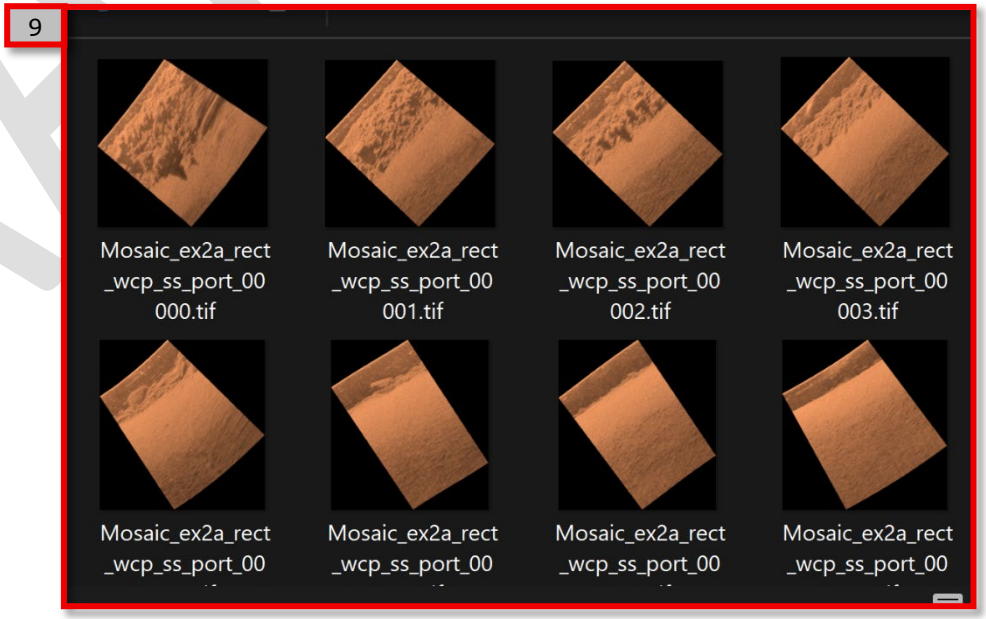


f. `Export Sonar Mosaic`: Select `GTiff`. This will mosaic each of the georeferenced chunks into a `WCP` and `WCR` mosaic. Use `VRT` to generate a small file which references the location of each of the chunks which will mosaic all the images on-the-fly. `VRT` is recommended if saving hard drive space is important. However, the absolute paths to each chunk is stored, meaning the paths will be broken if the files are moved to a new directory.

g. `# Chunks per Mosaic`: Set to `0`. Specify some positive value to mosaic the specified number of chunks into their own mosaic. For example if a recording generates 45 500-ping chunks and a value of `10` is specified, 5 mosaics will be generated. The first 4 will each with 10 chunks, and the last will have 5.

6. 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.
7. Processing will take longer than before (1) Sonogram Tile Exports (non-georeferenced)). Since the sonar recording is small it should process in ~1 minute.
8. Open the output folder (e.g. `C:\Users\cbodine\Desktop\PINGMapperHandbook\processed_data\Mosaic_ex2a`) and open `ss_port`. There is a `rect_wcp` and `rect_wcr` folder where rectified (`rect`) imagery with water column present (`rect_wcp`) and water column removed (`rect_wcr`) are saved. The `ss_star` folder has the same folders and corresponding imagery from the starboard beam.
9. Open the `rect_wcp` folder. These are the same data we exported in the Sonogram Tile exercises.
   a. We used a default `chunk size` of `500`. Each image except the last, has 500 pings. Whoops! I actually did 250, not 500!
   b. The raw sonogram tile was warped using `rubber sheeting` and projected to make the georeferenced GTiff.

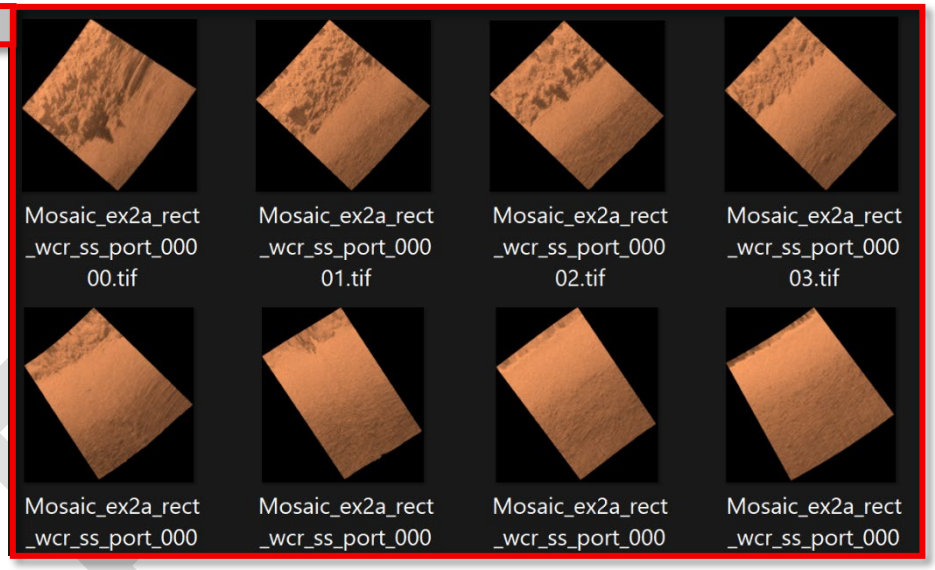




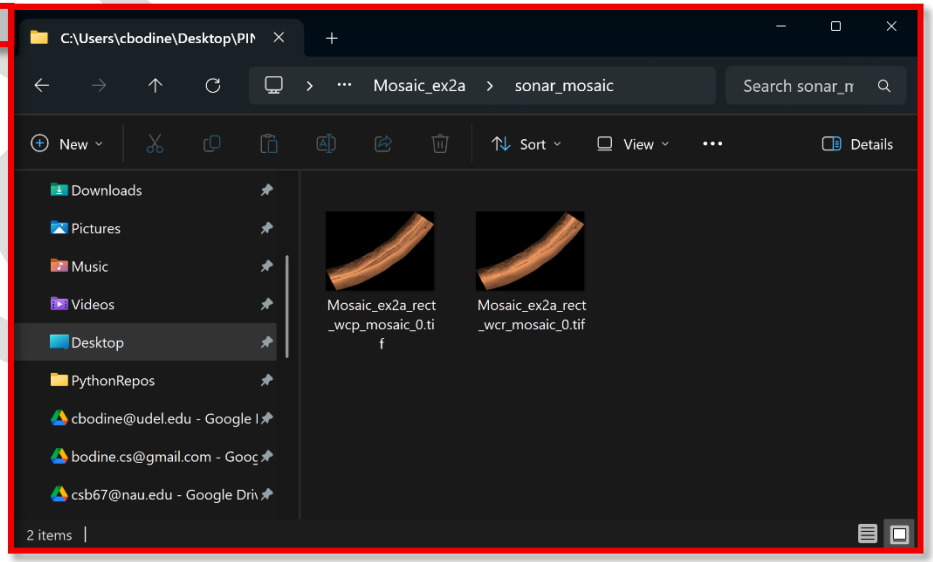
c. Each file name starts with the `project name` followed by a 5 digit integer ID (e.g. chunk ID), ordered by time of data collection. 00000 is the beginning, followed by 00001, and so on along the survey transect.

10. Open the `rect_wcr` folder. We see the same georeferenced chunks with the water column removed and slant-range corrected.
11. Go back to the top project (`C:\Users\cbodine\Desktop\PINGMapperHandbook\processed_data\Mosaic_ex2a`) directory and open the sonar_mosaic folder. There are two output mosaics: water column present and water column removed) comprised of data from both the portside and starboard beams.

10

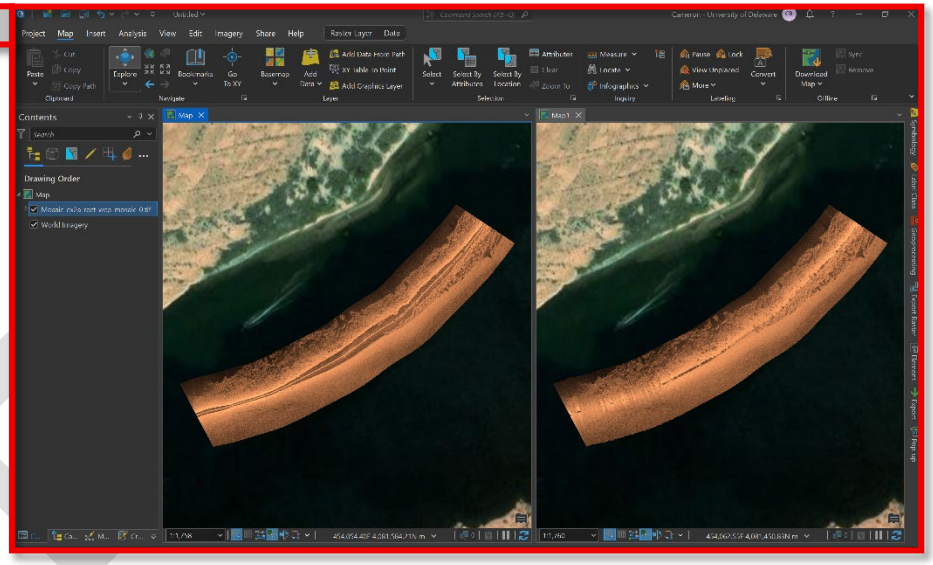


11



12. Open your GIS software. This tutorial will use ArcGIS Pro. *I thought you were all about open-source software! Why not use QGIS?* I know, I should. But a) I know Arc better; b) I'm lazy; and c) this handbook is already taking too damn long to write! *Ok, ok, ArcGIS is…* Open a new project and add the Gtiffs. You can add them to This is as easy as clicking and dragging the image from the file explorer into the map.
13. If we zoom out, we can see that the sonar data are in an iconic location: Horseshoe Bend in Glen Canyon, Arizona USA, downstream of the Glen Canyon Dam and Lake Powell. This recording was provided by Dr. Dan Buscombe during his time working with US Geological Survey in the Grand Canyon. Thanks for the data, Dr. Dan!

12



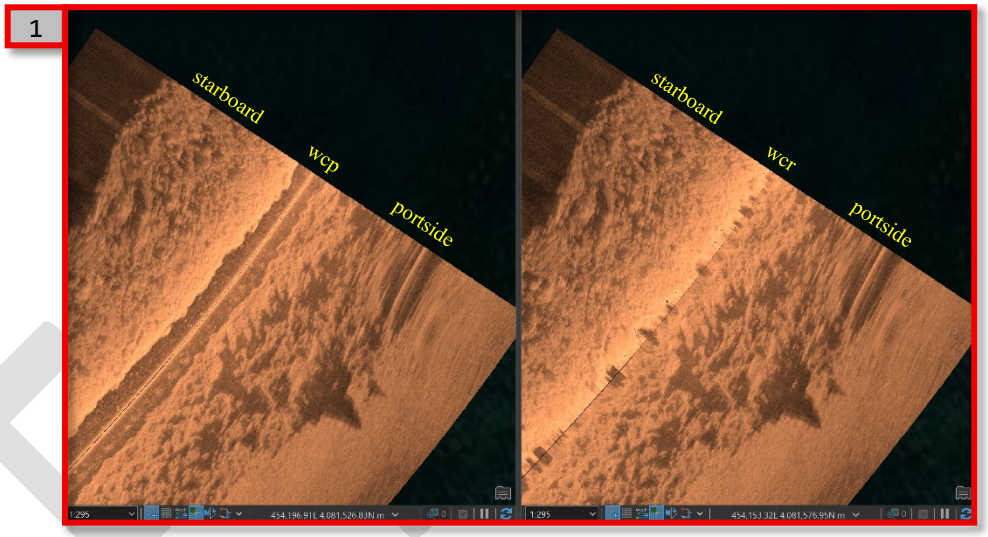
13

B) Analyzing Features and Assessing Quality of Sonar Mosaics

This is a very brief introduction to analyzing sonar mosaics. Developing expertise in sonar analysis takes time and practice. Importantly, it requires comparing features in sonar imagery to in-situ observations. If you are lucky enough to work in clear water environments, ground-truthing is relatively straightforward. In turbid, tannic, or deep aquatic systems, ground-truthing becomes much more difficult. If tasked with analyzing sonar collected by another team, I highly recommend going out with that team at least a few times so that you can compare the sonar data with the environment in real time. Look for clues along the bank that might indicate what you see on the sonar screen. Bring poles, ponar grabs, drop-cameras, on any other tool you might imagine to ground-truth different features seen in the imagery. Add waypoints on the sonar so that you can visit at a later time. Additionally, I recommend a literature review, consulting training resources (e.g. [3]), and reaching out to colleagues with experience (like me!). You can also find resources at PINGHub, a new one-stop-shop for everything fishfinder- and ping-related.

Let's examine the sonar mosaics we previously exported.

1. Add the data to a GIS map(s) and zoom in on the north-eastern portion of the scan. The figure shows `WCP` in the left map and `WCR` in the right.
    a. There are two basic bed types in the image. The starboard and half of the portside near the vessel track are composed of boulders and cobbles. You could use the ruler tool to get a sense of the size of some of these features.
    b. The portside furthest from the vessel track are composed of grain sizes smaller than the imaging resolution of the sonar. My best guess, without directly ground-truthing, is that they are gravel, small cobble, and potentially sand. The smooth, homogeneous texture indicates no bedforms (i.e. sand dunes or ripples).

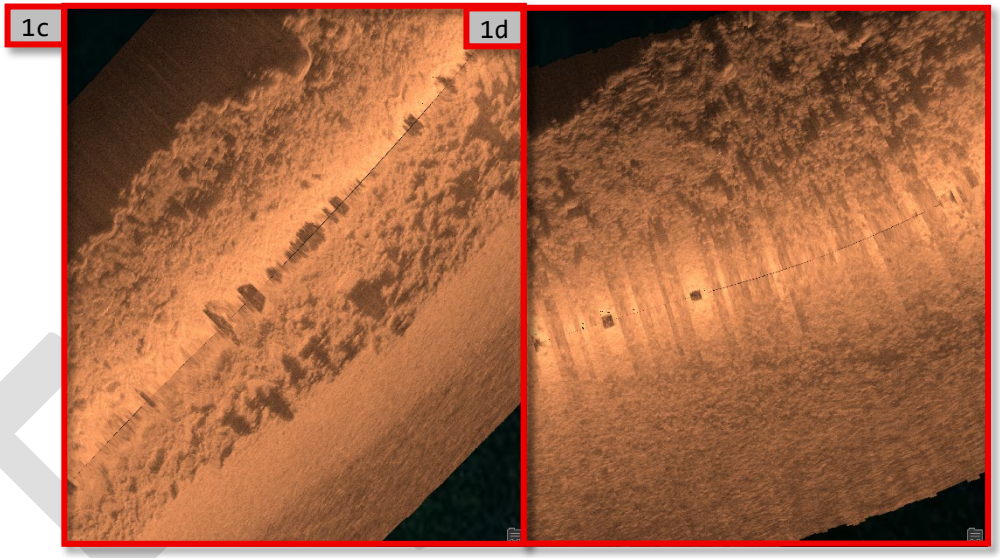


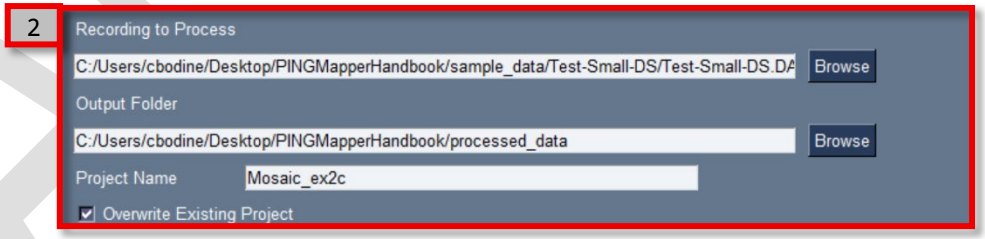
c. Examining the `WCP` mosaic, strange dark regions are visible along the trackline where portside and starboard scans meet.
d. Further downstream, strange striping effects are visible perpendicular to the vessel track.

What is causing these artifacts in the `WCP` mosaics? Since slant-range correction depends on an accurate depth estimate at nadir, an inaccurate estimate can lead to the presence of these artifacts. In the next part of the exercise, we will export bedpicks to see if bad depth estimates are the cause, and introduce a processing feature in PINGMapper that, in some cases, can help remedy this problem.

C) Automated depth estimation & exporting bedpick plots

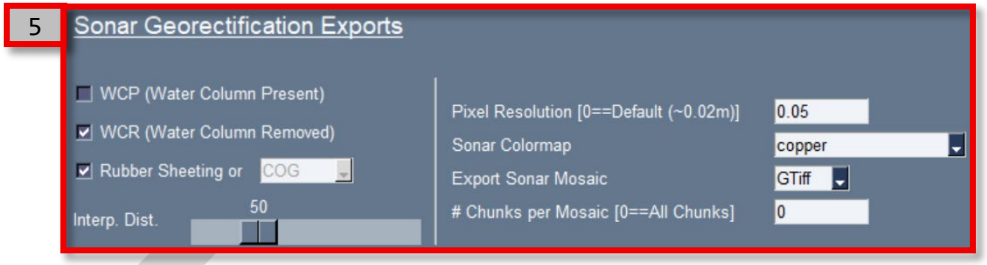
1. Launch the PINGWizard (if not already open; Appendix A) and select `Single Log`.
2. Since we saved out processing defaults previously, we should already see the small dataset selected in `Recording to Process` and the `Output Folder` will remain the same. You can always select new input and output options with `Browse`.
3. Update the `Project Name` to `Mosaic_ex2c`.
4. Scroll down to Depth Detection and Shadow Removal. Make selections based on those shown to the right.
   a. `Depth Detection`: select `Auto` as described in [1]. Basically, a segmentation model will be used to determine the depth from side-scan imagery. `Sensor` will use the sonar's depth estimate.
   b. `Adjust Depth`: Leave this at `0`. This can be used with either `Sensor` or `Auto` to push the depth deeper (positive) or shallower (negative).

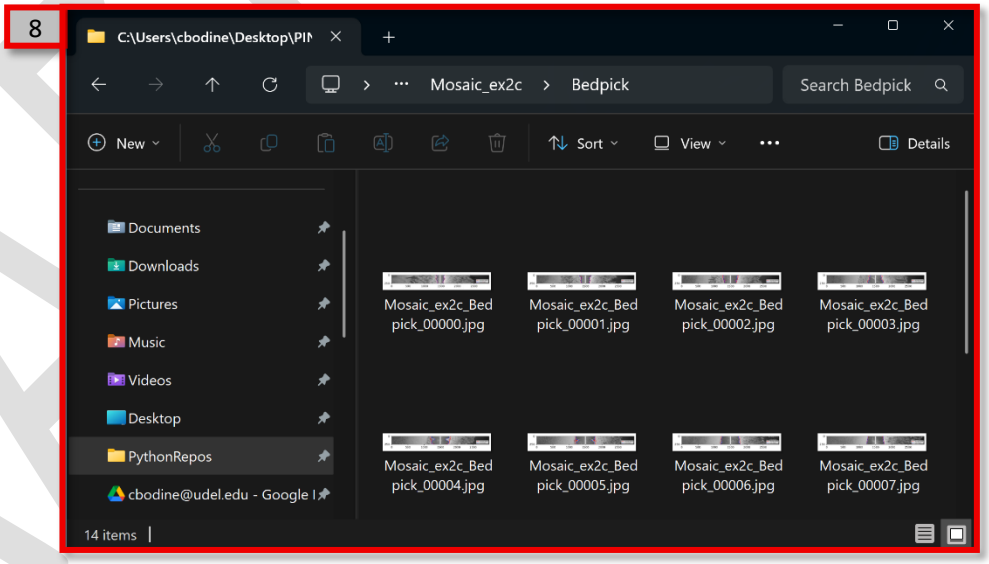


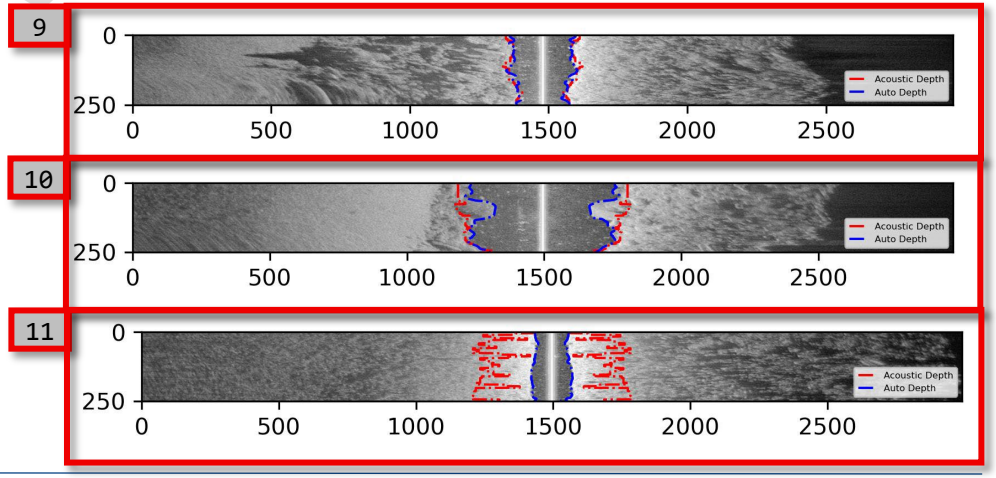


c. `Smooth Depth`: Applies a smoothing algorithm to `Sensor` or `Auto` depth. This is useful to smooth inaccurate depth estimates over short segments.
d. `Plot Bedpicks`: Export bedpick plots showing `Sensor` and `Auto` (if selected) depth picks.

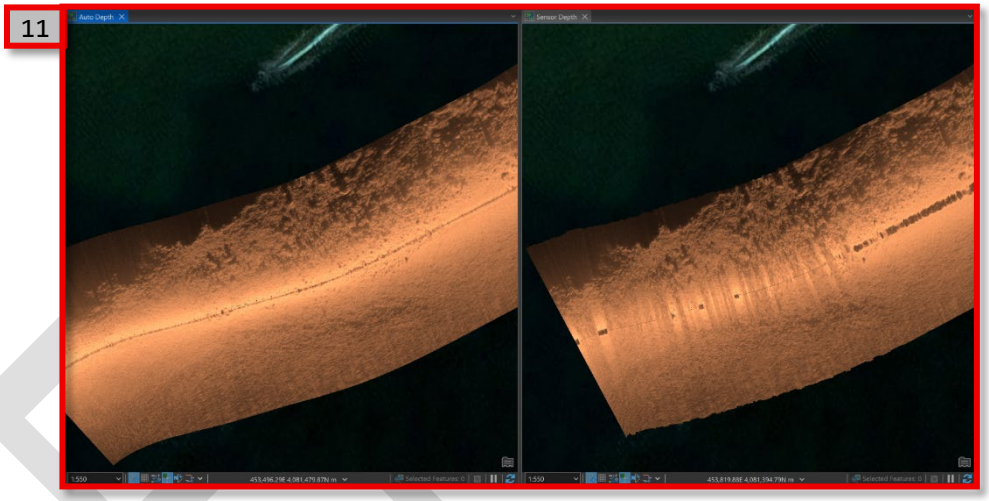
5. Scroll down to Sonar Georectification Exports. We will keep the same settings as before, except uncheck `WCP` as we already exported this data.
6. Scroll down to the bottom of the window and select `Save Defaults` (updates defaults based on currently selected parameters). Then click `Submit` to start processing.
7. Processing will take longer than before (1) Sonogram Tile Exports (non-georeferenced)). Processing time will be slightly longer due to running the depth model.
8. Open the output folder (e.g. `C:\Users\cbodine\Desktop\PINGMapperHandbook\processed_data\Mosaic_ex2c`). There is a new folder called `Bedpick`. Open that folder.
9. Open a plot in an image viewer. The `sensor` (acoustic) depth is shown in blue and `auto` depth in red. At the beginning of the recording, there is good agreement between the two.
10. The `auto` depth shows improved tracking over structure such as boulders.
11. Towards the end of the file, the `sensor` is clearly having issues detecting bottom. This is the same area shown in step B1c. The `auto` depth clearly outperforms the sensor depth in this case.







12. Let's add the sonar mosaic to a map to see if better bottom tracking (i.e. depth detection) improved the sonar mosaic. The left map shows the mosaic derived from `auto` depth and the right shows the mosaic previously created with `sensor` depth. The improved depth estimates eliminated the anomalies seen previously. This results in a much cleaner and representative output. It is important to note that there are times when the depth model will fail miserably. By exporting `bedpick plots`, you can determine which approach will work for a particular recording.



Lesson Complete!

Congratulations! Your path to PINGMapper Jedi Master continues! In this lesson, we learned how to export georeferenced sonar mosaics. We explored mosaics with the water column present and removed. We had a brief primer on analyzing imagery and identifying data quality issues. Finally, we learned how to use the automated depth detection abilities available in PINGMapper to improve image quality and remove artifacts.

3) Automated Substrate Mapping

Overview

Side-scan sonar generates imagery that is rich in detail. The wide variety of textures and variations in intensity combined with contextual clues can aid the interpretation of habitat features, including substrate. The typical approach to substrate mapping is to a) perform side-scan sonar surveys; b) post-process into mosaics; c) visually analyze the imagery to identify homogeneous patches of substrate; and d) manually delineate non-overlapping polygons around the patches to generate a seamless map of substrate.

This process of manual substrate mapping is terribly tedious! I did it for five years while working for the Florida Fish and Wildlife Research Institute. From my time there, I figured a single day in the field would result in more than a month of office work to generate the sonar mosaics and map substrates. This pain and suffering lead me to do a PhD with Dr. Dan Buscombe, creator of PyHum [4], on a Gulf Sturgeon spawning habitat project in collaboration with Dr. Adam Kaeser who pioneered habitat mapping with fishfinders for fisheries science applications [5]. The goal of the project was to develop a reproducible way of processing sonar recordings and automatically habitat. This process resulted in PINGMapper.

The deep learning model underlying the automated substrate mapping workflow in PINGMapper was trained on sonar datasets I visually analyzed and manually mapped. The sonar data were collected from rivers in Mississippi (Figure 10) and mapped according to substrate types I could visually identify (Table 1). The complete process is detailed in [1]. Therefore, it can only be expected that the model maps substrate as well as I can from sonar imagery that share characteristics with Gulf coastal plain river systems. That said, they might prove useful in initial characterization of substrates and habitats in your aquatic systems.

In this lesson, we will learn how to generate and evaluate automated substrate maps.

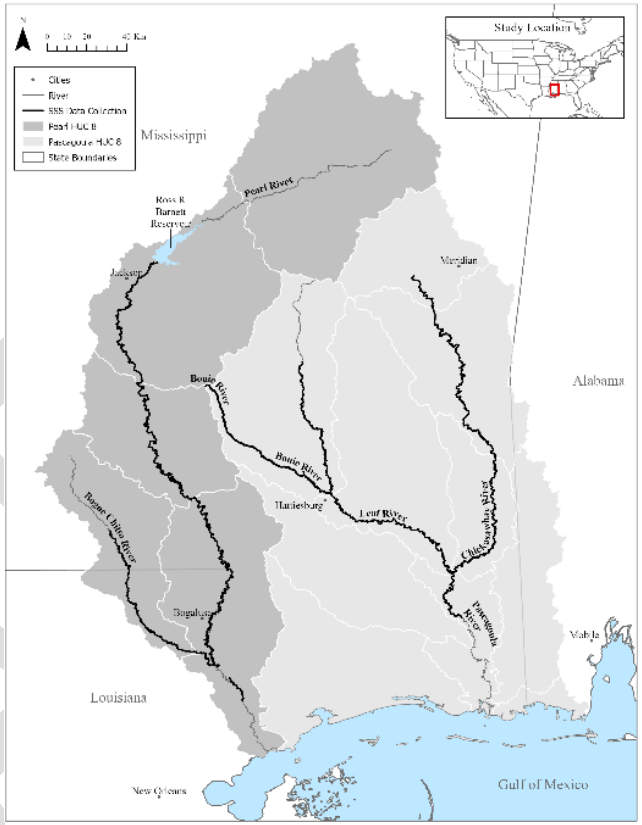


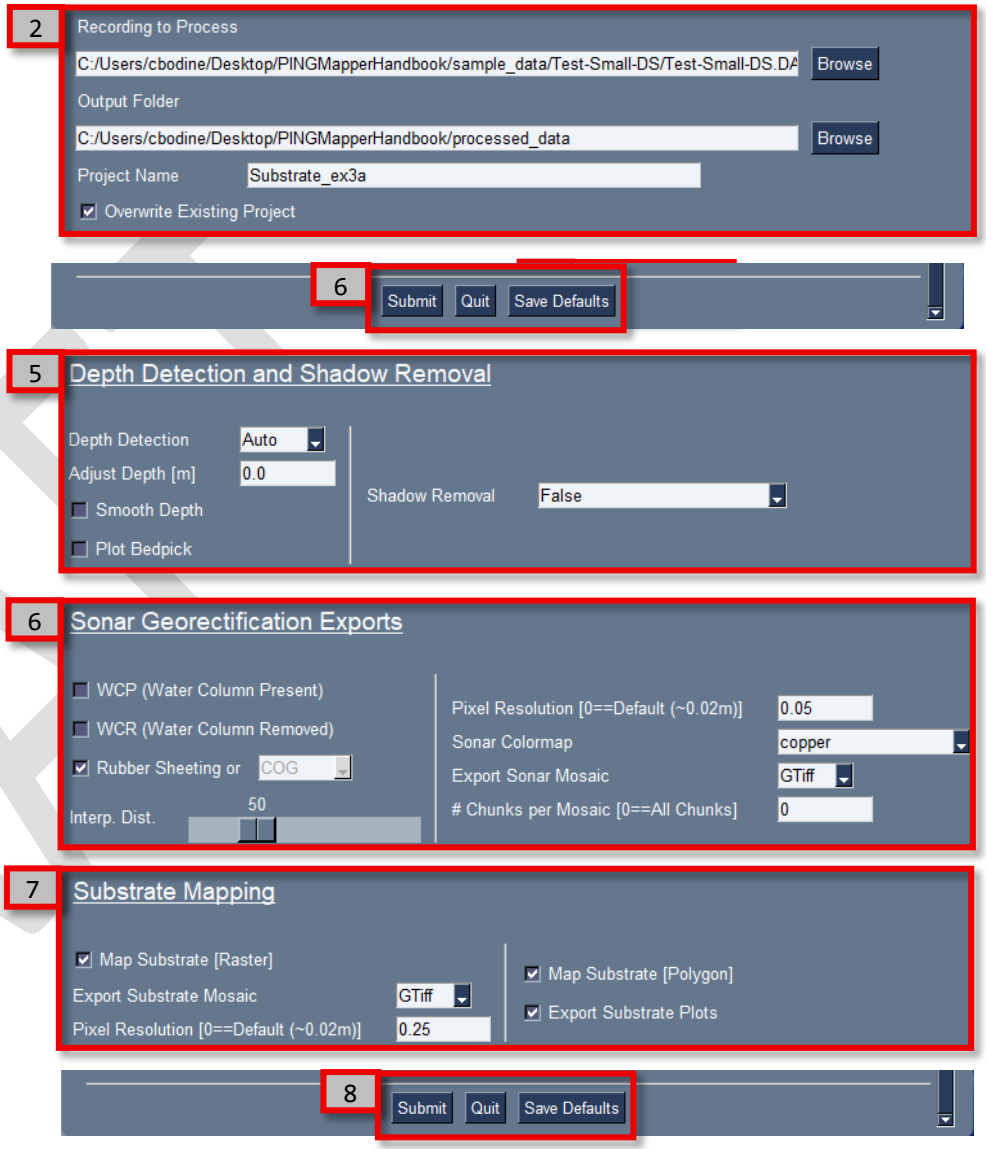
Figure 10 Coverage of side-scan sonar data collected on the Pearl and Pascagoula watersheds in Mississippi, USA. Retrieved from [1].

Table 1 - Substrate Classification Scheme for Sonogram Label Generation and Modeling. Retrieved from [1].

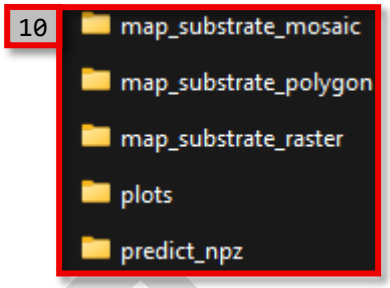
Fines—Rippled	Sand	Sonogram patches with clearly visible ripples.
Fines—Flat	Gravel, potentially sand, silt or mud	Sonogram patches with smooth bedforms and homogeneous textures.
Cobble/Boulder	Cobble and boulder chunks composed of rock, bank material, or hard clay	Sonogram patches with discernible objects, which cast shadows.
Hard bottom	Hard clay and bedrock	Sonogram patches with variable textures, edges, and sonar intensities.
Wood	Large tree trunks and branches	Sonogram patches with linear branching features with shadows.
Other	Unsure, cultural, etc.	Sonogram patches where substrate cannot be inferred or are constructed such as bridge pilings.

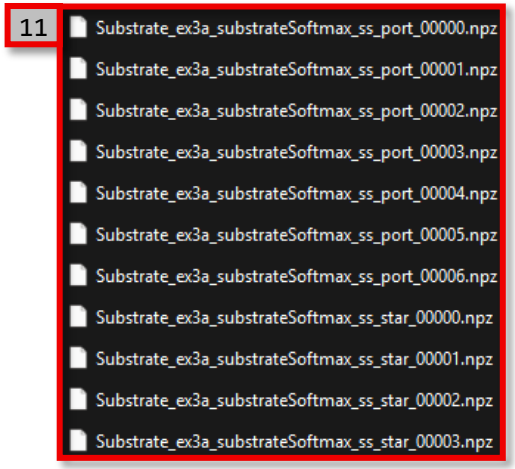
A) Exporting Substrate Maps Automatically

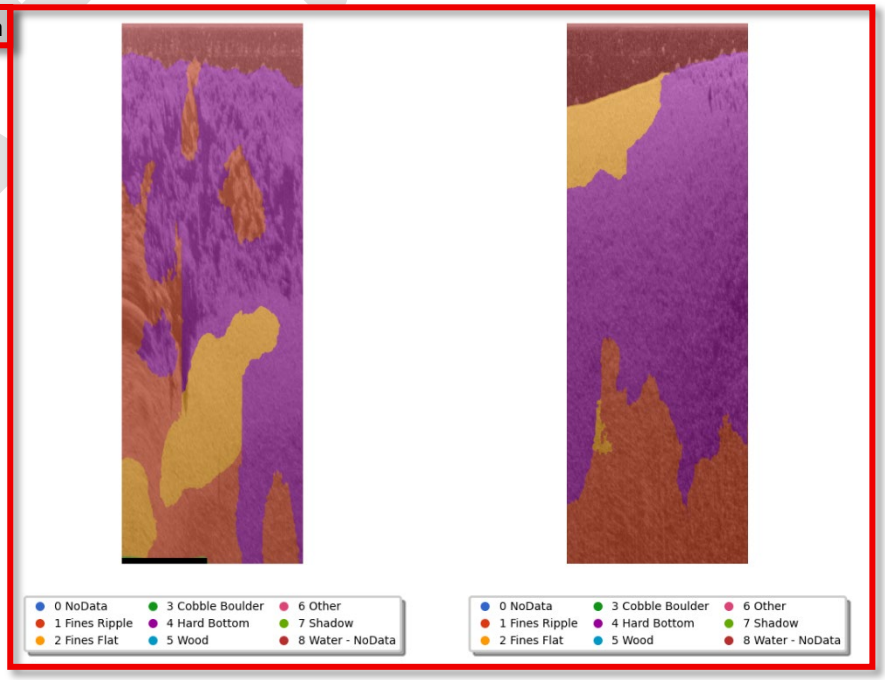
1. Launch the PINGWizard (if not already open; Appendix A) and select `Single Log`.
2. Since we saved out processing defaults previously, we should already see the small dataset selected in `Recording to Process` and the `Output Folder` will remain the same. You can always select new input and output options with `Browse`.
3. Update the `Project Name` to `Substrate_ex3a`.
4. Scroll down to General Parameters and set the `chunk size` to `500`. The substrate model was trained with 500 ping-long images. If you vary the chunk size, the resulting map will be different.
5. Scroll down to Depth Detection and Shadow Removal. Make sure `Depth Detection` is set to `Auto`. Uncheck `Plot Bedpicks` as they were previously exported.
6. Scroll down to Sonar Georectification Exports. Uncheck `WCP` and `WCR` as they were previously exported.
7. Scroll down to Substrate Mapping.
   a. `Map Substrate [Raster]`: Export a GTiff raster of the map.
   b. `Map Substrate [Polygon]`: Export a shapefile of the map.
   c. `Export Substrate Mosaic`: Similar to Sonar Georectification Imports, clicking `Raster` will result in a file for each chunk of data. These can be mosaiced into a single raster by selecting `GTiff` (or `VRT`).
8. 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.



9. Processing will take a few minutes. Three models will be run in succession. First is automated depth detection. Second is shadow removal, a pre-processing step for substrate mapping. Third is the substrate model.
10. Open the output folder (e.g. `C:\Users\cbodine\Desktop\PINGMapperHandbook\processed_data\Substrate_ex3a`). Open the `substrate` folder. We see five total folders:
    a. `map_substrate_mosaic`: A raster GTiff comprising a mosaic of each chunk's substrate prediction (`map_substrate_raster`).
    b. `map_substrate_polygon`: A shapefile (shp) of the substrate map in a vector format.
    c. `map_substrate_raster`: The substrate prediction for each 500-ping chunk.
    d. `plots`: Plots to help visualize the substrate predictions and model outputs.
    e. `predict_npz`: The model outputs stored in a zipped numpy file (`npz`).
11. Open the `predict_npz` folder. This contains the model outputs (logits, or scores) from the substrate model. During inference, the model makes a prediction on a sonogram tile. For each substrate class (Table 1), a 2D array with the same dimensions of the sonogram tile stores negative to positive values. These are stacked into a 3D array and zipped into a `npz` file. We will visualize the contents of the files with the `plots` we exported.
12. Open the `plots` folder. There are two types of plots generated for each chunk.
    a. The `*_classified_max_ss_*.png` plots show the final substrate classification from the model logits. This classification is created by applying a softmax activation function to a likelihood (i.e. probability) that a pixel belongs to a class. The class with the highest likelihood is chosen for the final classification. The substrate classification is colored and overlain on the sonar image in the plot.



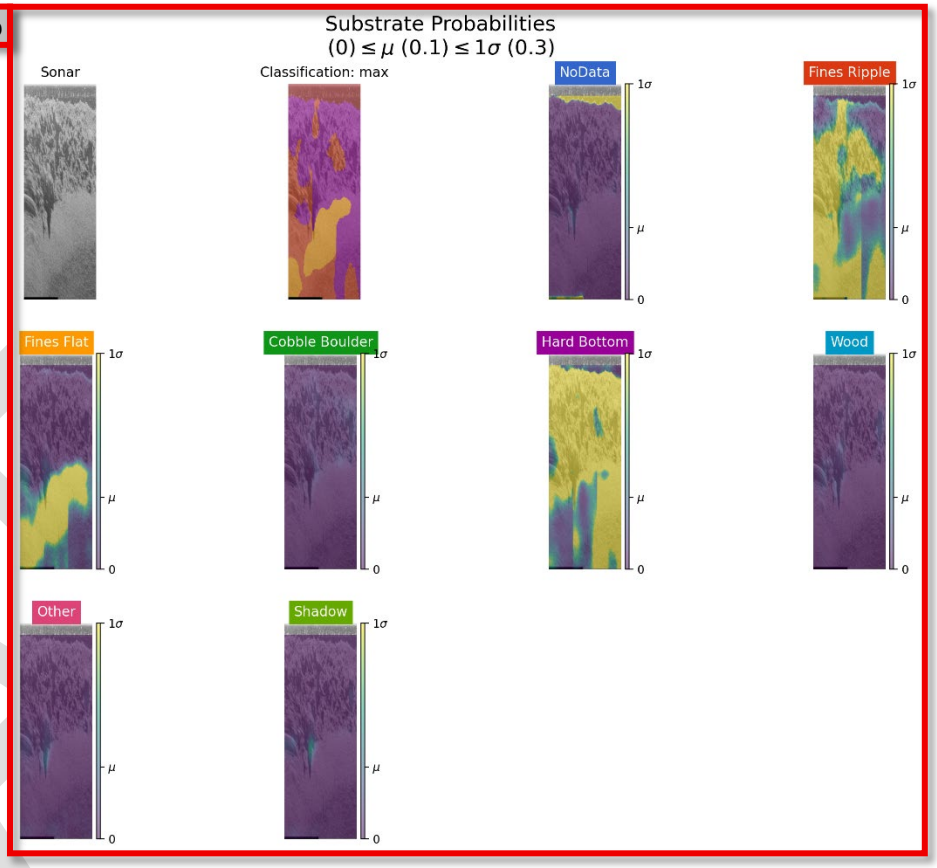




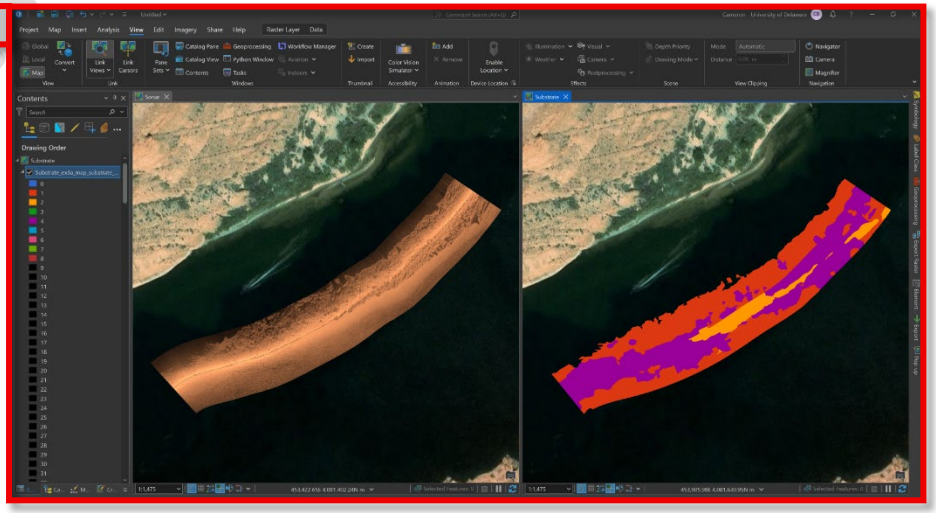
b. The `*_probability_*.png` plots show the probability (i.e. likelihood) of each pixel belonging to each class. Purple (dark) colors indicate low likelihood and Yellow (bright) indicate high likelihood of a pixel belonging to a class. These plots help uncover locations where the model is confident (a pixel belongs to only one class) or less confident (a pixel may belong to class a or d).

13. Open the `map_substrate_raster` folder. You will see GTiffs substrate maps for each of the chunks we saw in the plots. Corresponding `ss_port` and `ss_star` chunks have been merged into a single file.
14. Open the `map_substrate_mosaic` folder. Each chunk's substrate map has been merged into a single GTiff. We can examine this in GIS and make comparisons with the sonar mosaic exported in a previous exercise. Each class is coded as an integer. That integer corresponds to the class shown in the plots seen in step 12a.

12b



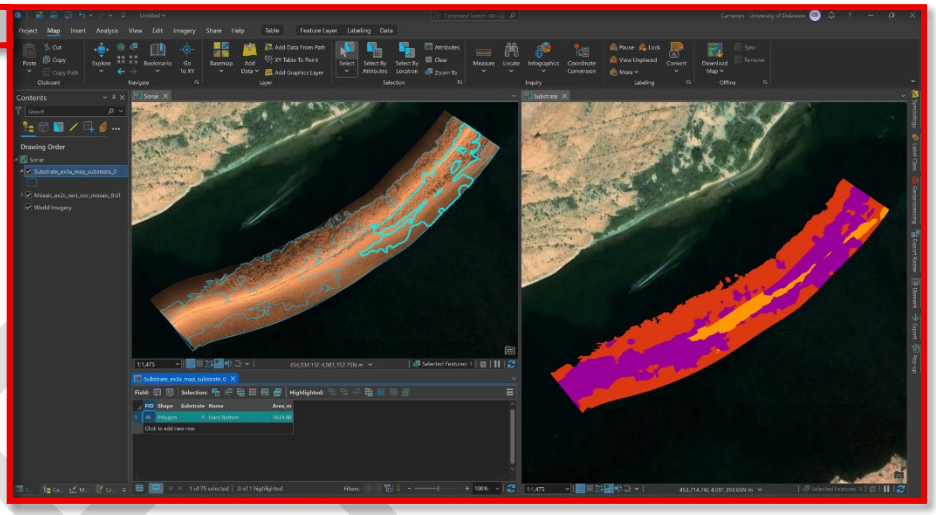
14



15. Open the `map_substrate_polygon` folder. A shapefile (*.shp) and corresponding *.dbf, *.prj, and *.shx files are present. The shapefile contains a polygon for each contiguous substrate patch. The raster code is stored in the Substrate field. The substrate name is shown in Name field. And the area in m^2 is calculated.

In the next exercise, we will take a closer look at the substrate map to determine how well the map reflects class distribution and area.

13



B) Evaluating quality of automated substrate map

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

Explanation on chunk size

When to export sonograms vs gtiffs

Table 2 – Computational performance of multithreaded processing. From [2].

Table S2. Computation time for sequential and multi-threaded algorithms on a test dataset[a]

Threads (t)	Sequential Algorithms[b]	Decode SON[c]	Bedpick Plots[c]	Sonogram Tiles[c]	Rectify Sonograms[c]	Total (s)	Total (hh:mm:ss)
1	367.3	52.8	423.3	1158.6	4349.0	6351.0	01:45:51
2	363.4	35.3	261.0	723.2	3197.1	4580.0	01:16:20
4	361.5	25.7	186.7	531.2	1960.9	3066.0	00:51:06
6	359.0	23.9	179.6	533.7	1451.8	2548.0	00:42:28
8	363.2	24.0	175.8	522.1	1388.9	2474.0	00:41:14

[a] Test dataset from a Humminbird Helix with Mega imaging. Sonar recording includes high-frequency down image (200 kHz), very high-frequency down image (1.2 MHz), and two very high-frequency side scan images (1.2 MHz). Total duration of the recording is 01:00:06 (hh:mm:ss). Total ping count is 279,915. Range setting is 1,398 returns per ping, or 26.6m.

[b] Sequential (e.g., non-parallel) algorithms include decoding DAT file, determining SON file structure, depth processing, trackline smoothing, and generating rectified mosaics.

[c] Multi-threaded processing algorithm.

Filtering Sonar Logs

content to be added…

A) Heading Deviation

content to be added…

B) AOI

content to be added…

C) Time

content to be added…

EGN Corrections to Intensity

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 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!

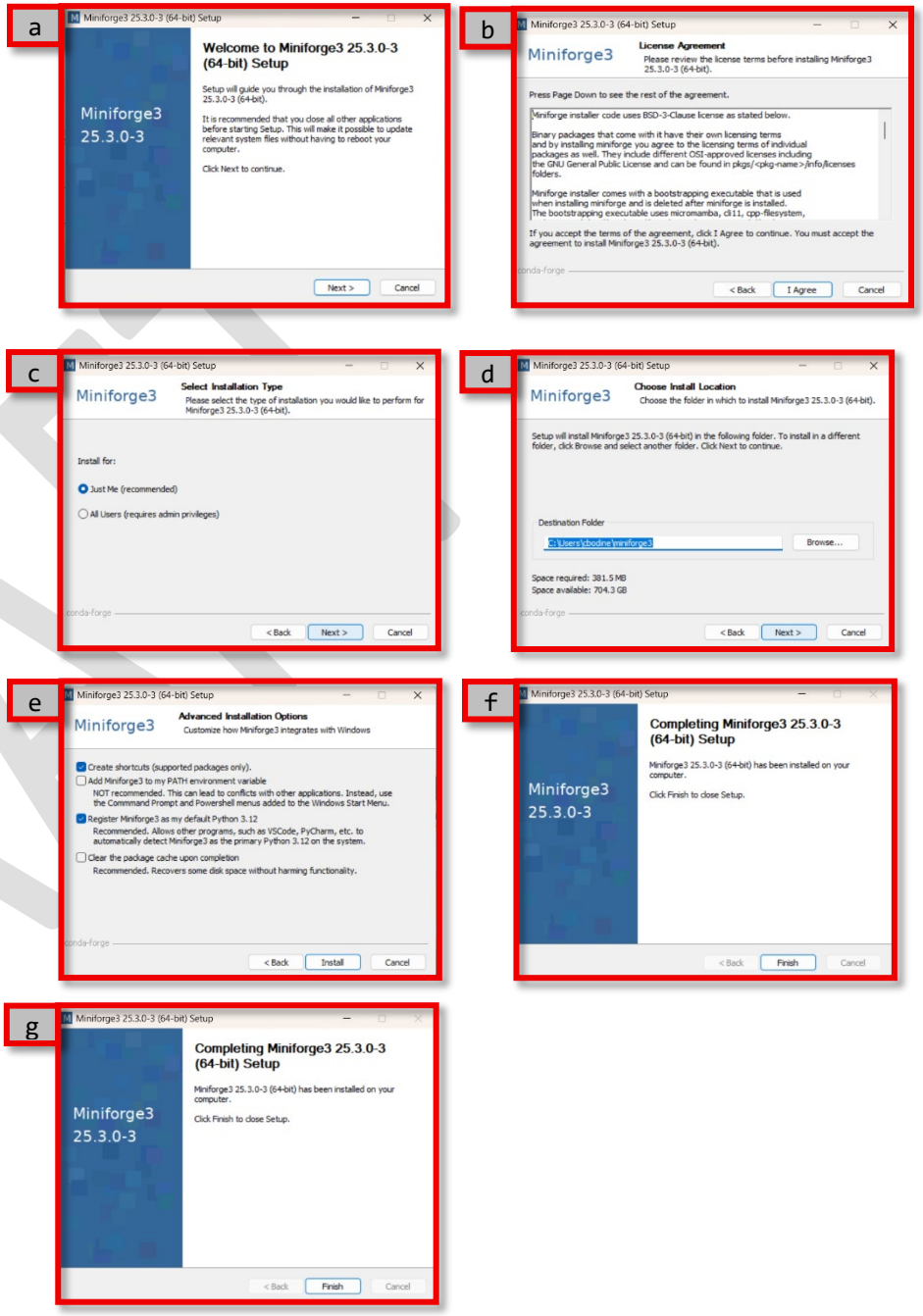
Procedure

1. Install Miniforge

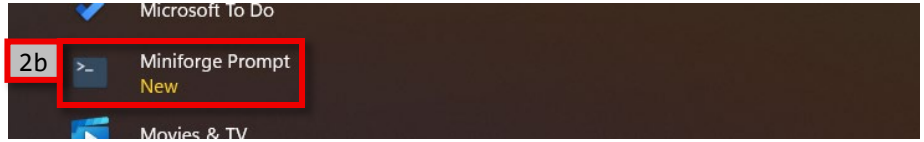
Go to the Miniforge Website and download the software. Choose the appropriate installer for your computer's operation system. This tutorial was made on a Windows machine but the process should be similar on other operation systems. Click the file and it will download to your Downloads folder, or you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

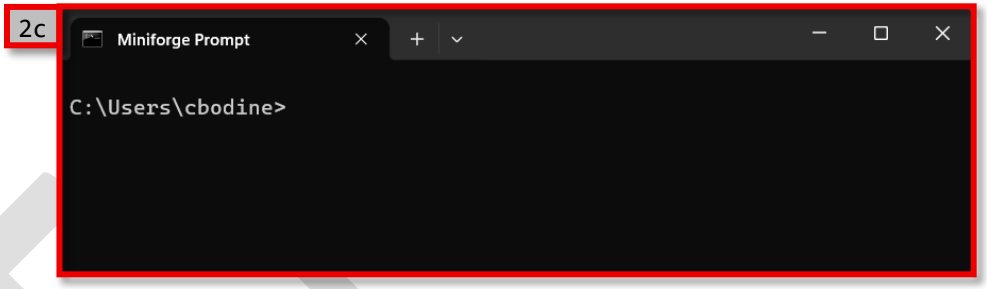
Double click the file to begin the installation file. This will open an installation window.

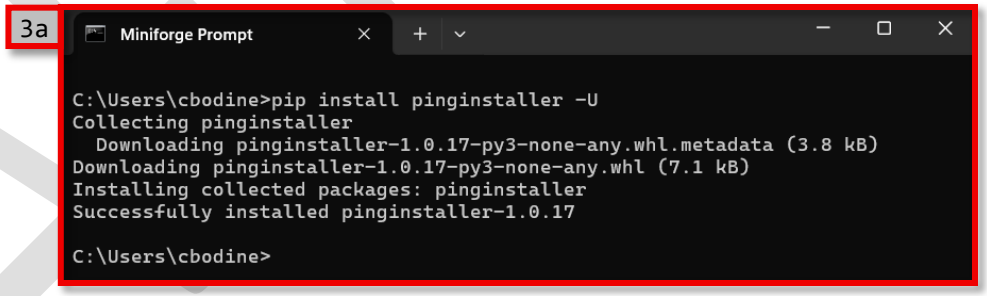
a. Click Next and you will see the license agreement (2). After reviewing the license agreement, you must select `I Agree` to continue with the installation.
b. After you agree, you will have an option to install Miniconda for `Just Me` or `All Users`. You want to install Miniforge in your user folder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

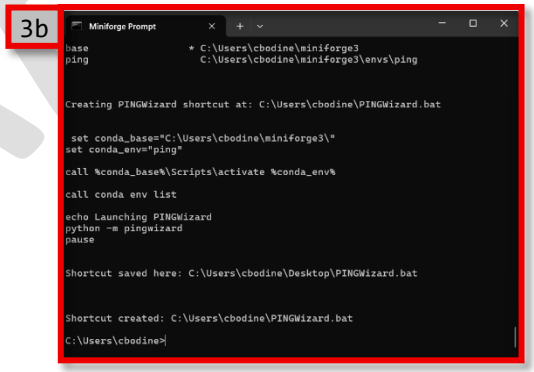


2. Open Miniforge Prompt
    a. Now for the scary part! We are going to open a command prompt so that we can submit a series of commands to Miniforge. If you want to gain some familiarity with navigating with the prompt, you can watch this video.
    b. Miniforge is a command prompt that we will use 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

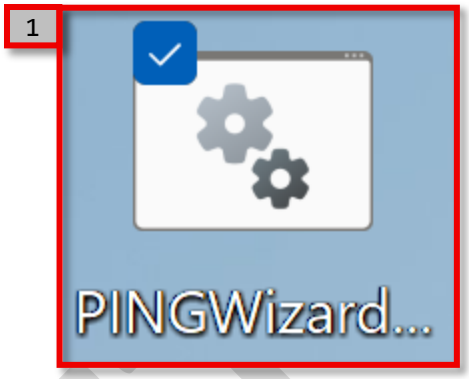
PINGWizard is a light-weight interface for launching PINGMapper utilities, including running the tests, processing a sonar log, batch process multiple sonar logs, and updating the installation.

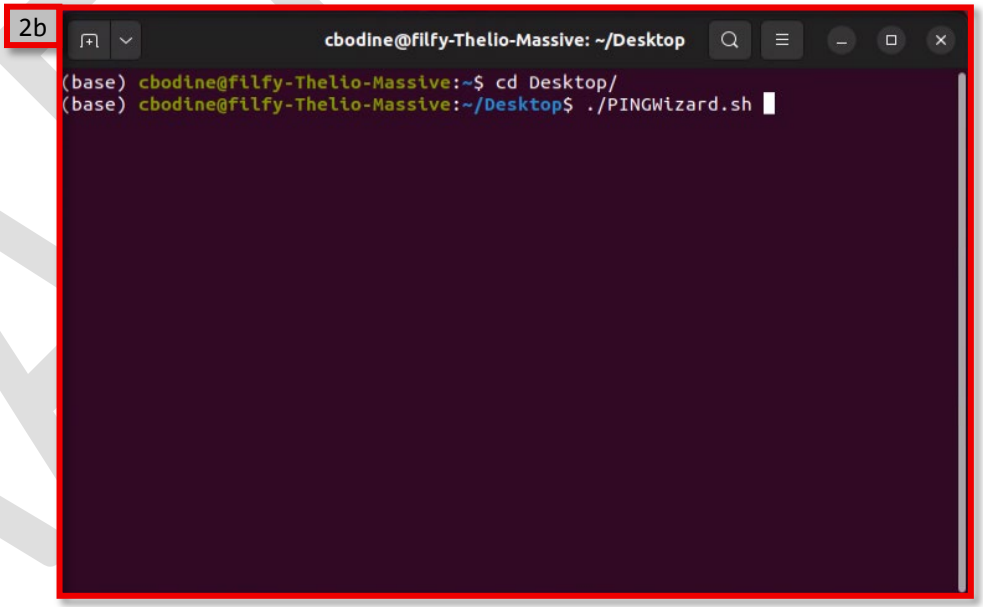
There are two options for launching PINGWizard: a) with a shortcut or b) command prompt.

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`

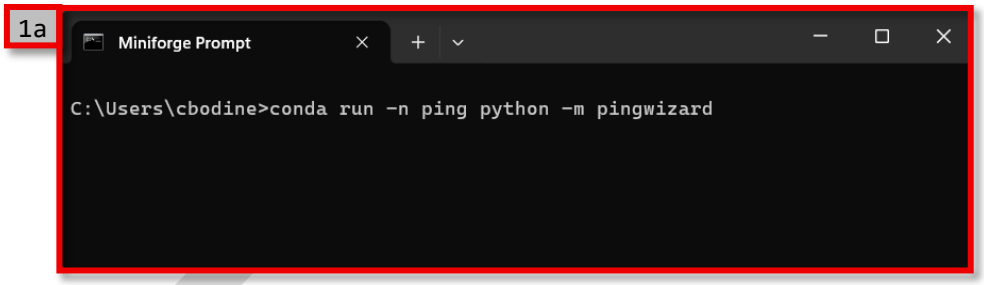




Option B – Conda Command Prompt

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.

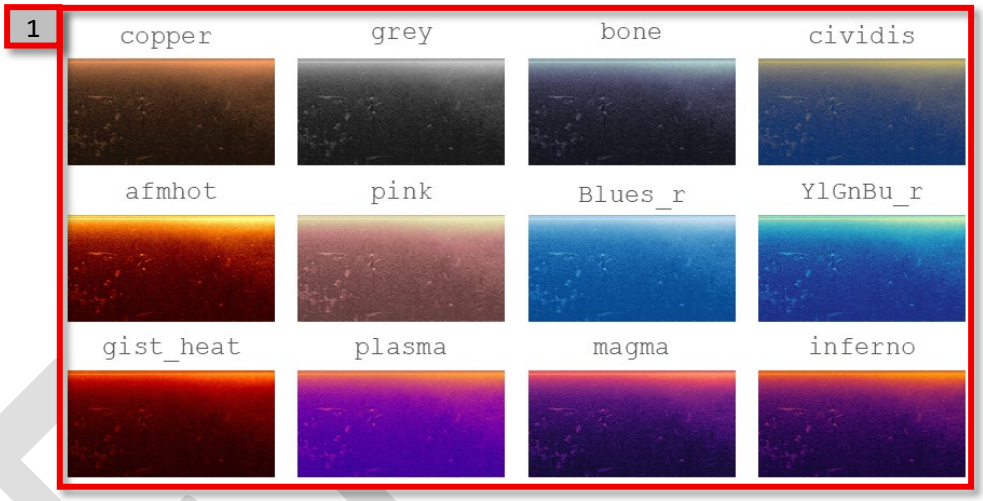
Appendix C – Colormap Selection

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

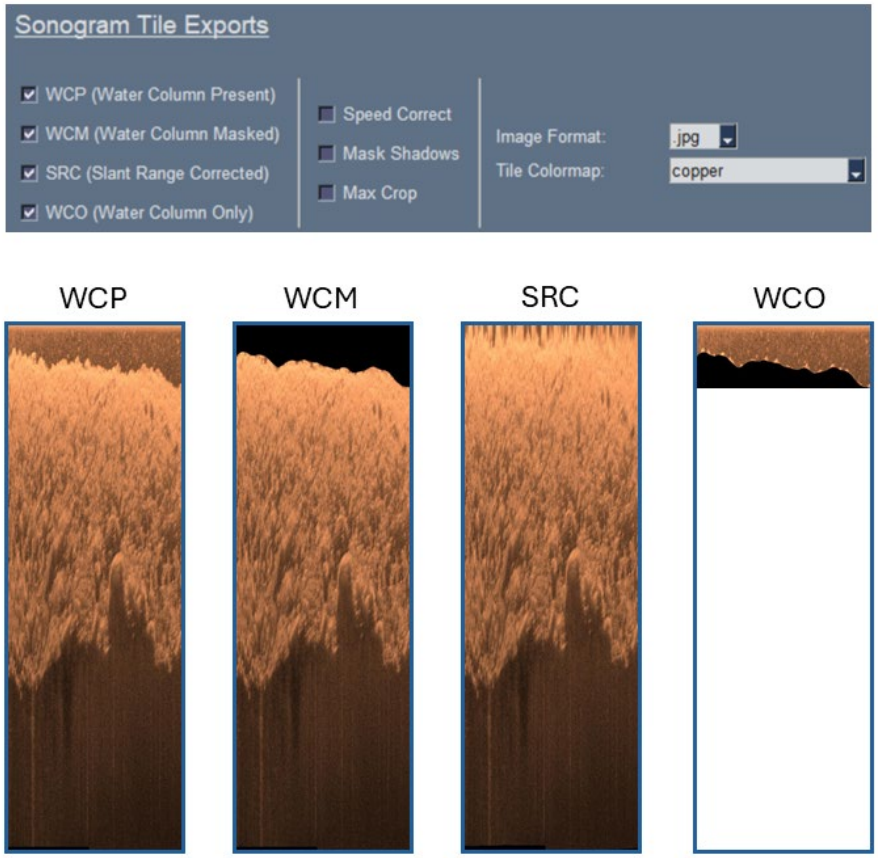
1. Example of recommended colormaps. The first row are highly recommended.



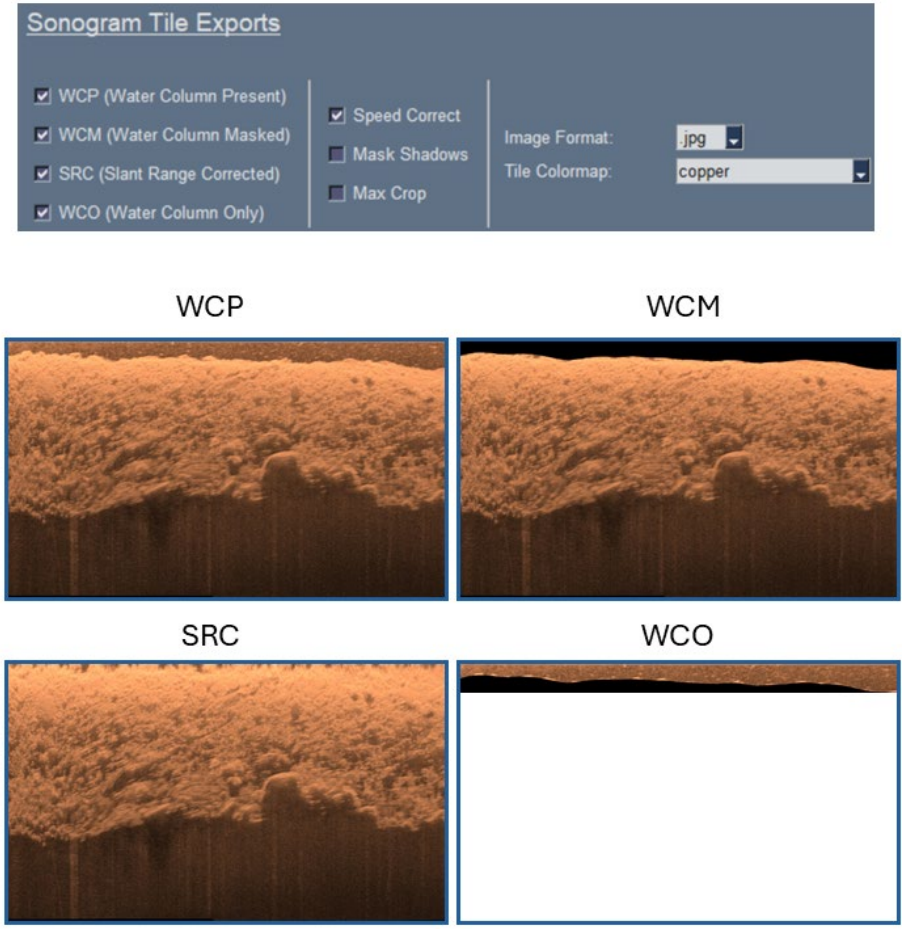
Appendix D – Errors & Logs

Appendix E – Sonogram tile export examples

Raw

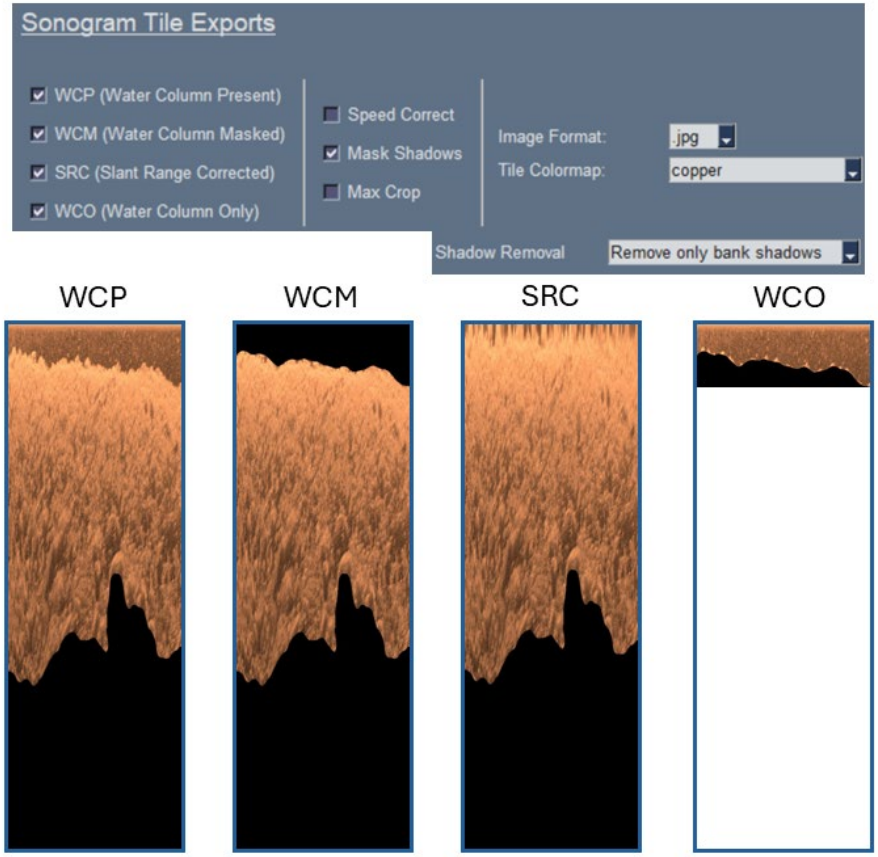


Speed Corrected

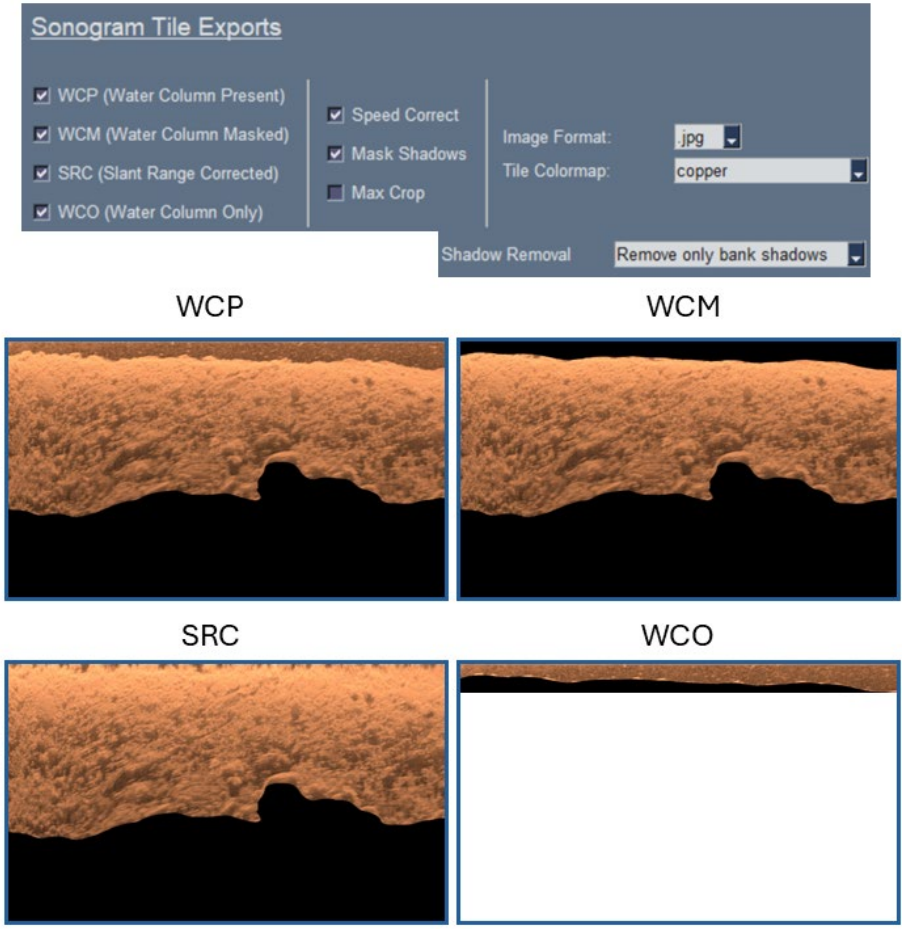


Mask Shadows

Raw

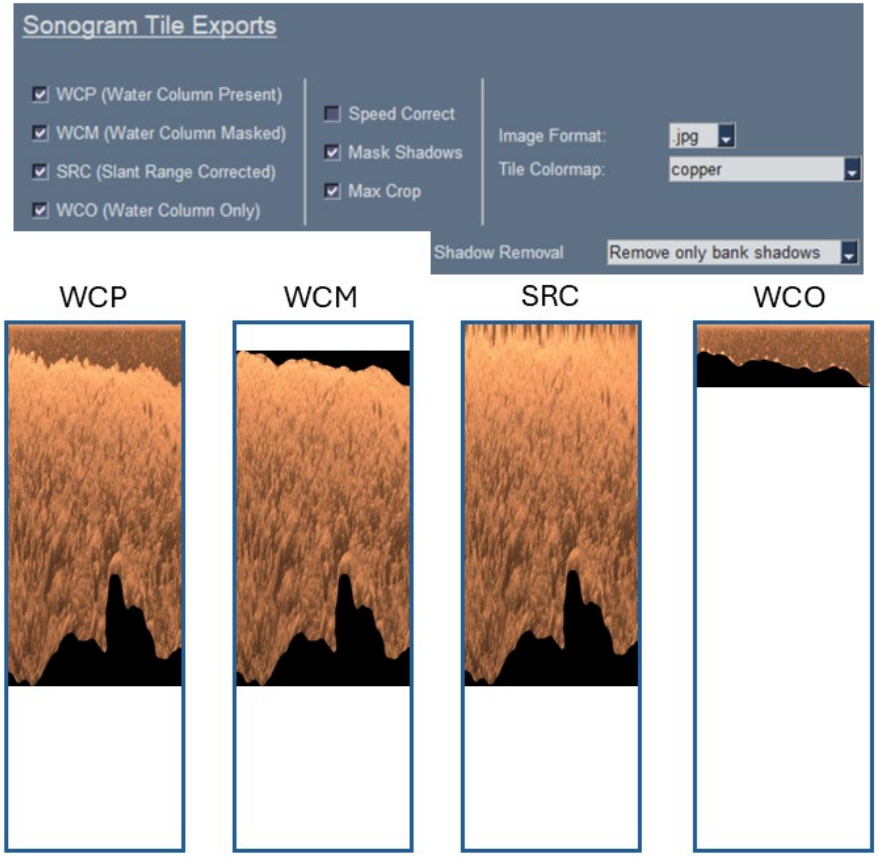


Speed Corrected

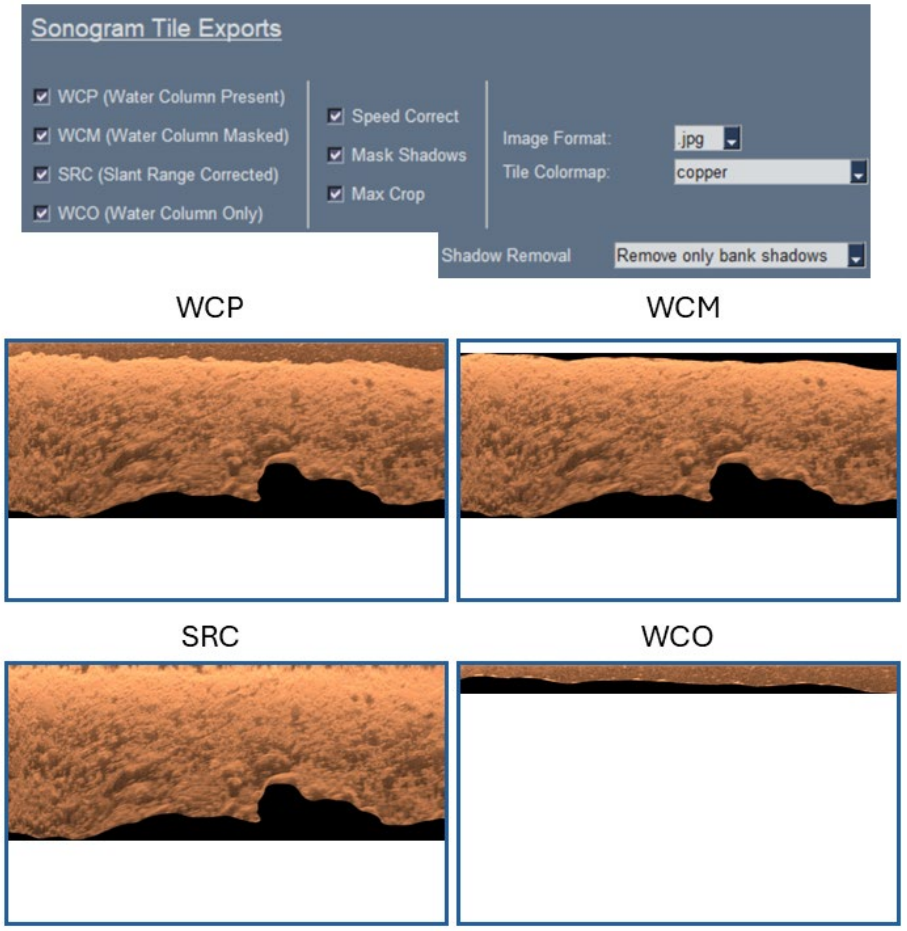


Mask Shadows – Crop

Raw

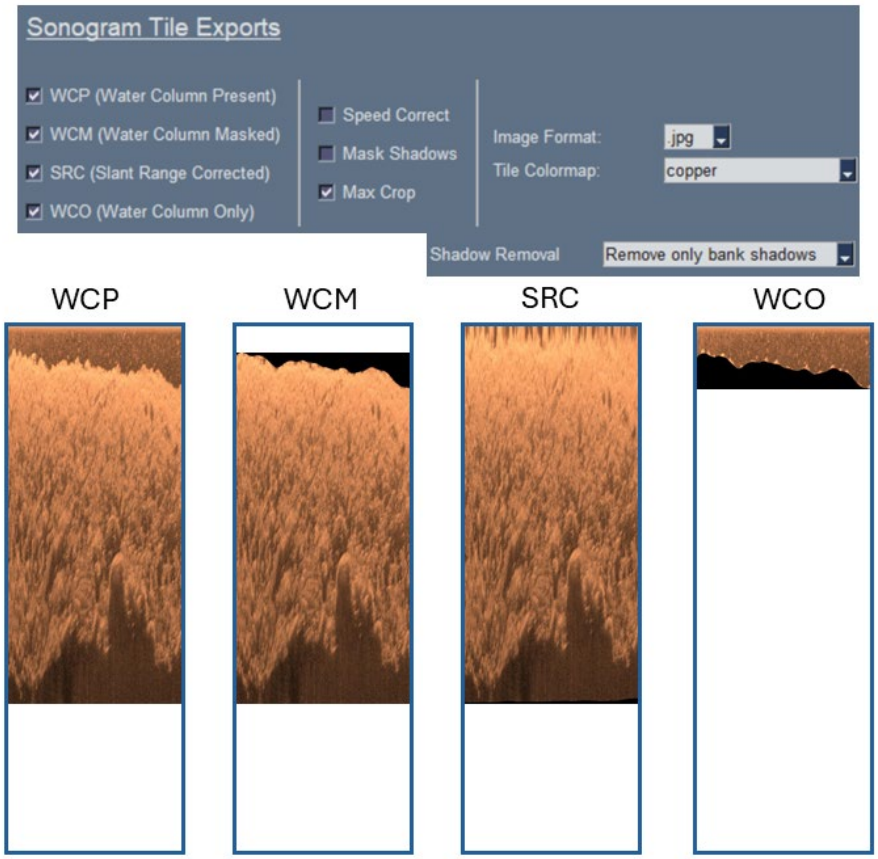


Speed Corrected

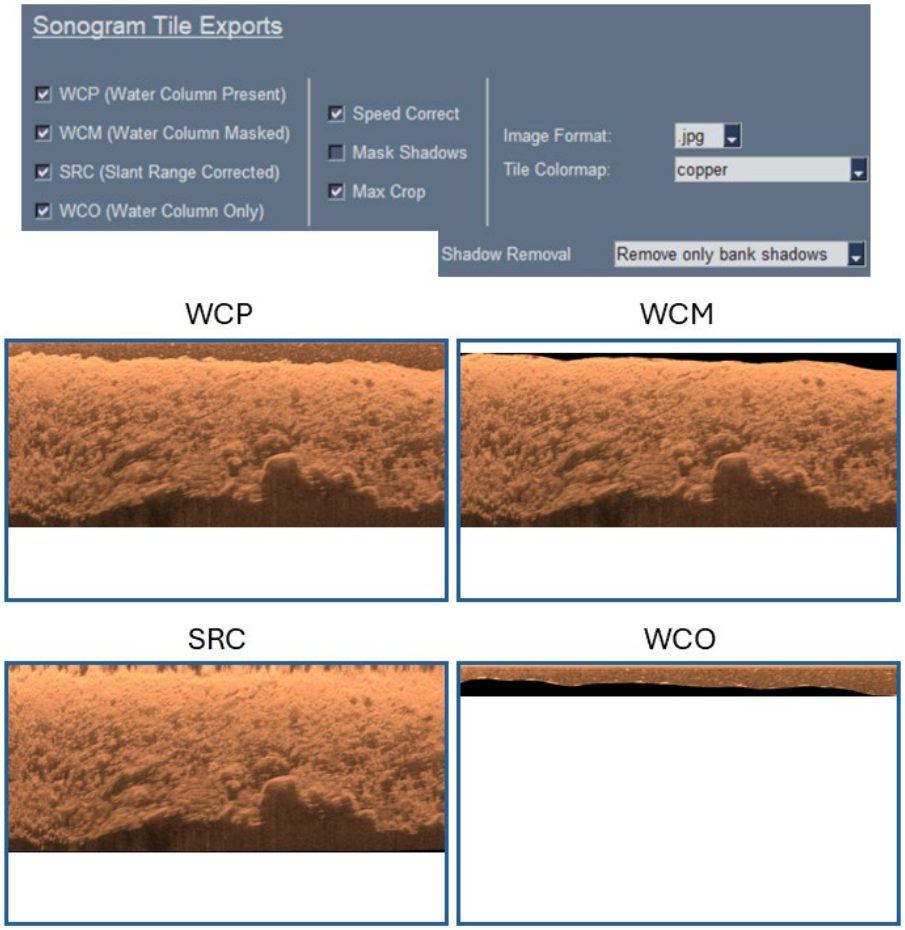


Crop – With Shadow Model

Raw



Speed Corrected



References

[1] C. S. Bodine, D. Buscombe and T. D. Hocking, "Automated River Substrate Mapping From Sonar Imagery With Machine Learning," *Journal of Geophysical Research: Machine Learning and Computation,* 2024.

[2] C. S. Bodine, D. Buscombe, R. J. Best, J. A. Redner and A. J. Kaeser, "PING-Mapper: Open-Source Software for Automated Benthic Imaging and Mapping Using Recreation-Grade Sonar," *Earth and Space Science,* 2022.

[3] V. Capone, "Training Videos," Black Laser Learning, [Online]. Available: https://blacklaserlearning.com/product-category/training-video/. [Accessed 2025].