OBSrange v1.0 README

Stephen Mosher, Josh Russell and Zachary Eilon

August, 2018

Contents

1   Scope

The primary goal of **OBSrange** is to provide a robust, efficient, open-source OBS location code to the marine geophysical community.

    OBSrange is a set of scripts written in both MATLAB and PYTHON for precisely locating ocean bottom seismometers (OBSs). The starting point for the code are sets of acoustic ranging survey files obtained during deployment. Using these survey files, the code inverts for instrument locations, turn-around times, and depth averaged sound speeds in water. Additionally, **OBSrange** generates several figures visualizing these results as well as estimates of parameter uncertainties. For a more detailed description of the algorithms we use for our inversion, synthetic tests, and our results, please refer to our paper (put reference here).

2   Getting Started

2.1   Preliminaries

Note that the MATLAB implementation of **OBSrange** is completely self-contained, meaning that the MATLAB scripts dont require any toolboxes beyond those available with the standard installation. In the case of the PYTHON implementation, the scripts have been written for PYTHON 3 and require the following open-source libraries (its recommended to have versions at least as great as the versions listed):

- *numpy* v1.13.1

- *scipy* v0.19.1

- *matplotlib* v2.2.2

- *pymap3d* v1.7.4

2.2   Survey File Format

Since acoustic ranging survey files are the input for **OBSrange**, in its current incarnation, these files **must** follow the format shown in Figure 1. Note that the survey file in this example is a *.txt* file that has been generated by a Scripps Institute of Oceanography (SIO) PYTHON script (Ernest Aaron, pers. comm.). In practice the acoustic ranging survey is conducted using an EdgeTech 8011M acoustic command and ranging deck box or a similar instrument. The header information is contained in exactly 9 lines followed by a blank line at the 10th line. From the 11th line onward the results of individual pings (sonar sends and receives) are logged. The necessary data obtained from the header of these *.txt* files by **OBSrange** are the site name, drop coordinates, and estimated drop depth. Below the header, **OBSrange** reads in the two-way travel time of each ping, the ship coordinates when each ping was received, as well as the UTC time at which each ping was received. Finally, in the *.txt* file shown, we see that bad results either begin as "Event skipped ..." or are

flagged by an asterisk. ***OBSrange*** can handle both of these cases, in that it does not read in such results, but note that bad results **must** be designated in this exact manner for the time being.


```
  EE01.txt          x
1   Ranging data taken on:   2018-04-16 03:42:00.588000
2   Cruise:                  obs-cruise
3   Site:                    EE01
4   Instrument:
5   Drop Point (Latitude):   -4.94605
6   Drop Point (Longitude):  -130.38178
7   Depth (meters):          4670
8   Comment:                 confusing depth
9   ================================================
10
11    6206 msec. Lat: 4 56.7506 S  Lon: 130 22.9231 W  Alt: 22.20 Time(UTC): 2018:106:10:42:35
12    6206 msec. Lat: 4 56.7506 S  Lon: 130 22.9231 W  Alt: 22.20 Time(UTC): 2018:106:10:42:35
13    6205 msec. Lat: 4 56.7497 S  Lon: 130 22.9234 W  Alt: 23.49 Time(UTC): 2018:106:10:42:43
14    6205 msec. Lat: 4 56.7491 S  Lon: 130 22.9234 W  Alt: 24.17 Time(UTC): 2018:106:10:43:43
15    6206 msec. Lat: 4 56.7477 S  Lon: 130 22.9239 W  Alt: 25.43 Time(UTC): 2018:106:10:44:43
16    6205 msec. Lat: 4 56.7474 S  Lon: 130 22.9223 W  Alt: 25.25 Time(UTC): 2018:106:10:45:44
17    6204 msec. Lat: 4 56.7552 S  Lon: 130 22.8851 W  Alt: 24.39 Time(UTC): 2018:106:10:46:44
18    6206 msec. Lat: 4 56.8168 S  Lon: 130 22.8162 W  Alt: 25.79 Time(UTC): 2018:106:10:47:44
19    6218 msec. Lat: 4 56.8816 S  Lon: 130 22.7224 W  Alt: 24.21 Time(UTC): 2018:106:10:48:44
20    6246 msec. Lat: 4 56.9525 S  Lon: 130 22.6149 W  Alt: 23.71 Time(UTC): 2018:106:10:49:45
21    6295 msec. Lat: 4 57.0161 S  Lon: 130 22.4979 W  Alt: 22.30 Time(UTC): 2018:106:10:50:45
22    6359 msec. Lat: 4 57.0784 S  Lon: 130 22.3800 W  Alt: 20.71 Time(UTC): 2018:106:10:51:45
23    6438 msec. Lat: 4 57.1397 S  Lon: 130 22.2617 W  Alt: 18.13 Time(UTC): 2018:106:10:52:46
24   Event skipped - Timeout or Badly formatted data was received
25   Event skipped - Timeout or Badly formatted data was received
26    6584 msec. Lat: 4 57.2314 S  Lon: 130 22.0897 W  Alt: 16.39 Time(UTC): 2018:106:10:54:13
27    6703 msec. Lat: 4 57.2942 S  Lon: 130 21.9712 W  Alt: 16.22 Time(UTC): 2018:106:10:55:14
28    6831 msec. Lat: 4 57.3579 S  Lon: 130 21.8519 W  Alt: 16.05 Time(UTC): 2018:106:10:56:14
29    6976 msec. Lat: 4 57.4197 S  Lon: 130 21.7348 W  Alt: 15.86 Time(UTC): 2018:106:10:57:14
30   *7132 msec. Lat: 4 57.4839 S  Lon: 130 21.6154 W  Alt: 19.01 Time(UTC): 2018:106:10:58:15
31   Event skipped - Timeout or Badly formatted data was received
32   Event skipped - Timeout or Badly formatted data was received
33   Event skipped - Timeout or Badly formatted data was received
```

Figure 1: Example survey *.txt* file.

2.3   Coordinate System

2.4   Setting Parameters

Slight design differences exist between the MATLAB and PYTHON versions of the code, but the main usage remains the same between the two. In both cases parameters are set in a single main script which will run and execute every other aspect of the code. Ideally, the only edits users ever need make are in editing the parameters of these main scripts. In the case of the MATLAB version, this script is called *OBSrange.m* and these parameters are set in the top lines of that script. In the case of the PYTHON version, the main script is similarly called *OBSrange.py*, and parameters are set in that script. The parameters to be set by users are described and compared in Table 1 (sorted alphabetically by MATLAB parameter names). Again, in both the MATLAB and PYTHON versions, once these parameters have been set, these scripts may be run and will produce results.

3

Table 1: **OBSrange** Parameter Descriptions.

MATLAB Parameter	PYTHON Equivalent	Description
datapath	survey_fles	Path to the directory containing the survey files.
ifplot	-	Option to plot results. In PYTHON plots are created and saved by default but not displayed when **OSBrange.py** is run. In MATLAB plots are created, saved, and displayed while running **OBSrange.m** if this parameter is set to 1.
ifQC_ping	QC	Option to perform quality control on ping results obtained from the survey files. Pings with two-way travel times beyond a certain threshold are filtered out of any analysis (see res_thresh below).
ifsave	-	By default the MATLAB version will write single station results to .txt files. If this parameter is set to 1, then it will additionally write single station results to .mat files. PYTHON writes single station results to both .pkl and .txt files by default.
onesta	-	Option to process a single station. PYTHON will process whatever survey files (.txt files) are located in the directory represented by survey_fles.
outdir	output_dir	Path to output directory.
par.dampdvp	dampdvp	Normal damping for water sound speed.
par.dampTAT	damptat	Normal damping for turn-around time.
par.dampx	dampx	Normal damping for station x-coordinate.
par.dampy	dampy	Normal damping for station y-coordinate.
par.dampz	dampz	Normal damping for station z-coordinate.
par.epsilon	eps	Global norm damping for stabilization.
par.if_twtcorr	twtcorr	Option to apply a correction to two-way travel times due to the ship's radial velocity.
par.N_bs	N_bs	Number of bootstrap iterations.
par.E_thresh	E_thresh	RMS reduction threshold for the inversion.
par.npts_movingav	npts	Number of points in an N-point moving average smoothing filter applied to the ship's velocity. Note that if this parameter is set to 1 that no smoothing is applied.
par.TAT_bounds	bounds	Bounds for turn-around times ($msec$).
par.TAT_start	tat	Turn-around time ($msec$).
par.vp_w	vpw	Water velocity (m/s).
projpath	-	Directory for both input and output (MATLAB only).
res_thresh	res_thresh	Residual threshold for pings if applying quality control ($msec$, see ifQC_ping/QC above).

2.5   Structure

In this section we briefly describe the general structure of **OBSrange**, illustrated in Figure 2. In both the MATLAB and PYTHON versions of the code, the top-level directory of **OBSrange** includes the main script (*OBSrange.m* or *OBSrange.py*, respectively), in which the parameters for running the code are set by the user (see Section 2.4). Once parameters have been set, the main script can be run; it will loop through files in the survey files directory and call functions contained within the functions directory. All results will be
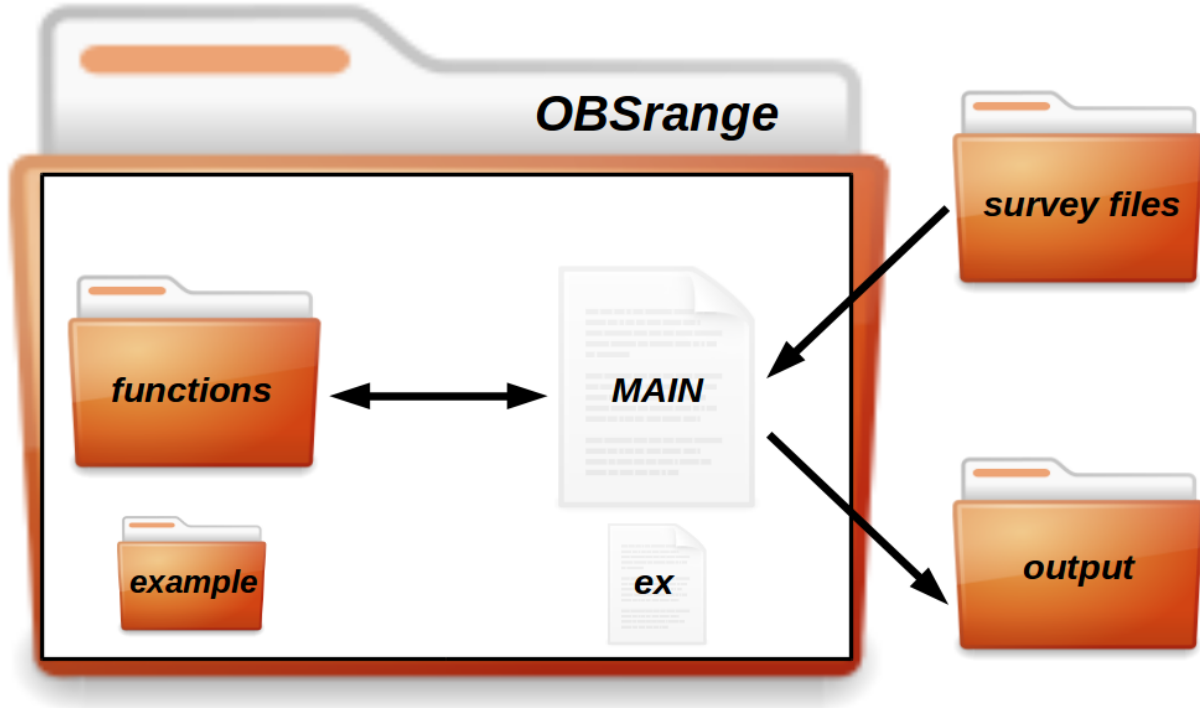
Figure 2: **OBSrange** structure.

written into an output directory. Note that paths to the directories for the survey files and output are specified by the user in the main script. In the case of the MATLAB version, these directories are themselves contained within a single folder, set via the *projpath* variable. Finally, **OBSrange** includes a single station example which will be discussed in the Section 3.
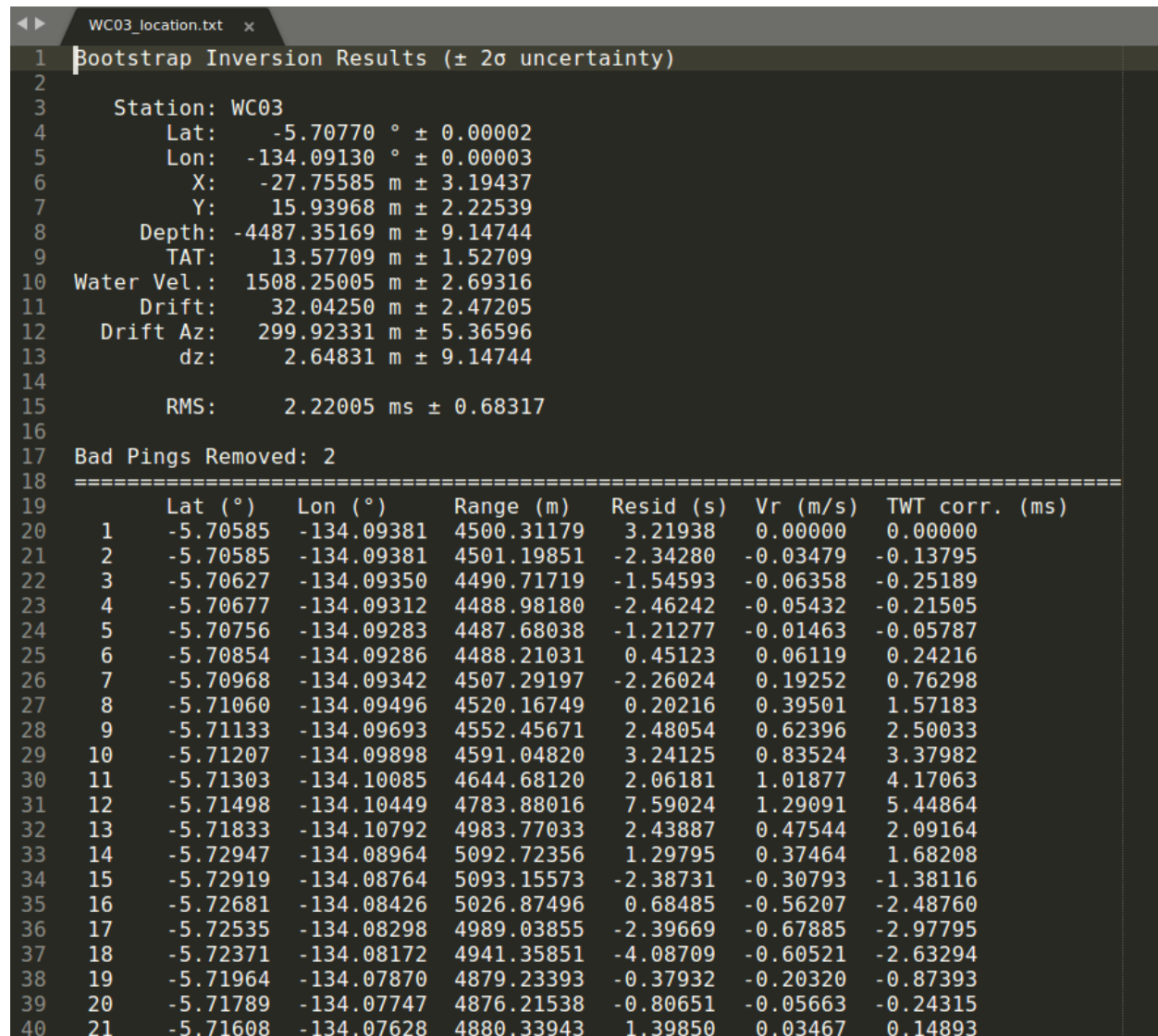
3 Example

3.1 General Output

The example provided with **OBSrange** locates the OBS package deployed at site WC03 of the PacificORCA array (reference). Simply run the example script for your corresponding version of the code, either *OBSrange_example.py* or *OBSrange_example.m*. These scripts will read the example survey *.txt* file contained within **OBSrange**'s example directory (see Figure 2) and will also write the output into that directory. In the case of the PYTHON version, the output will consist of a *.pkl* file named *WC03_location.pkl*, a *.txt* file similarly named *WC03_location.txt*, and 6 figures. In the case of the MATLAB version the output will consist of a *.txt* file also named *WC03_location.txt*, a *.mat* file called *WC03_data.mat*, and the same figures.

3.2 The *.txt* Files

The *.txt* files created by the MATLAB and PYTHON versions are formatted slightly differently, but the results are the same. In Figure 3 we show the first 40 lines of *WC03_location.txt* created by the PYTHON example and describe the general features of this file.

```
  WC03_location.txt  ×
 1  Bootstrap Inversion Results (± 2σ uncertainty)
 2
 3     Station: WC03
 4         Lat:     -5.70770 ° ± 0.00002
 5         Lon:   -134.09130 ° ± 0.00003
 6           X:    -27.75585 m ± 3.19437
 7           Y:     15.93968 m ± 2.22539
 8       Depth: -4487.35169 m ± 9.14744
 9         TAT:     13.57709 m ± 1.52709
10  Water Vel.:  1508.25005 m ± 2.69316
11       Drift:     32.04250 m ± 2.47205
12    Drift Az:    299.92331 m ± 5.36596
13          dz:      2.64831 m ± 9.14744
14
15         RMS:      2.22005 ms ± 0.68317
16
17  Bad Pings Removed: 2
18  ===============================================================================
19          Lat (°)    Lon (°)    Range (m)   Resid (s)   Vr (m/s)   TWT corr. (ms)
20      1   -5.70585  -134.09381  4500.31179   3.21938    0.00000     0.00000
21      2   -5.70585  -134.09381  4501.19851  -2.34280   -0.03479    -0.13795
22      3   -5.70627  -134.09350  4490.71719  -1.54593   -0.06358    -0.25189
23      4   -5.70677  -134.09312  4488.98180  -2.46242   -0.05432    -0.21505
24      5   -5.70756  -134.09283  4487.68038  -1.21277   -0.01463    -0.05787
25      6   -5.70854  -134.09286  4488.21031   0.45123    0.06119     0.24216
26      7   -5.70968  -134.09342  4507.29197  -2.26024    0.19252     0.76298
27      8   -5.71060  -134.09496  4520.16749   0.20216    0.39501     1.57183
28      9   -5.71133  -134.09693  4552.45671   2.48054    0.62396     2.50033
29     10   -5.71207  -134.09898  4591.04820   3.24125    0.83524     3.37982
30     11   -5.71303  -134.10085  4644.68120   2.06181    1.01877     4.17063
31     12   -5.71498  -134.10449  4783.88016   7.59024    1.29091     5.44864
32     13   -5.71833  -134.10792  4983.77033   2.43887    0.47544     2.09164
33     14   -5.72947  -134.08964  5092.72356   1.29795    0.37464     1.68208
34     15   -5.72919  -134.08764  5093.15573  -2.38731   -0.30793    -1.38116
35     16   -5.72681  -134.08426  5026.87496   0.68485   -0.56207    -2.48760
36     17   -5.72535  -134.08298  4989.03855  -2.39669   -0.67885    -2.97795
37     18   -5.72371  -134.08172  4941.35851  -4.08709   -0.60521    -2.63294
38     19   -5.71964  -134.07870  4879.23393  -0.37932   -0.20320    -0.87393
39     20   -5.71789  -134.07747  4876.21538  -0.80651   -0.05663    -0.24315
40     21   -5.71608  -134.07628  4880.33943   1.39850    0.03467     0.14893
```

Figure 3: *.txt* file output of *OBSrange_example.py.*

In both versions, the header of *WC03_location.txt* summarizes the main results of the inversion performed by **OBSrange** on this station. Shown in the header are the final estimates for the X and Y coordinates of the package relative to the drop point, as well as their converted latitude and longitude, the sensor turn-around time (*TAT*), and the final depth estimate (*Depth*). Additionally, the header contains the total package drift distance (*Drift*) and azimuth (*Drift Az.*), the difference between the initial estimated depth and final depth estimate (*dz*), and the depth averaged velocity of sound in water (*Water Vel.*).

6

Finally, the header contains the overall RMS misfit for this site and also displays the number of pings that were removed via the ping quality control. Below line 18 of *WC03_location.txt* the details of each ping are logged, namely, the ship latitude, longitude, estimated distance to the sensor, two-way travel-time residual, the ship's radial velocity and corresponding travel-time correction (whether it was applied or not).

3.3   The *.pkl* and *.mat* Files

In this example PYTHON will also create a *.pkl* file called *WC03_out.pkl* and MATLAB will create a *.mat* file called *WC03_out.mat*. In essence, both of these files are simply containers which hold various results of the bootstrap inversion. The *.pkl* file contains a PYTHON *dictionary* object of various results and values and the *.mat* file contains a 1x1 *struct* object called *datamat*. Both data structures contain many of the same fields, all of which are listed in Table 2 (sorted alphabetically by MATLAB parameter names):

Table 2: Data fields contained in the *.mat* and *.pkl* files

MATLAB	PYTHON	Description of Field
-	vrs	Ship's radial velocity at each survey point.
-	Nbad	The number of pings removed via quality control.
-	dzs	The depth difference after each bootstrap iteration.
azi_bs	azs	Sensor drift azimuth after each bootstrap iteration.
Cm_mat	cov	Model covariance matrices after each bootstrap iteration.
drift_bs	drifts	Sensor drift distance after each bootstrap iteration.
drop_lonslatz	drop_geo	Geographic drop coordinates.
dtwt_bs	dtwts	Final twtt residuals at each survey point.
dtwtcorr_bs	corrs	Final twtt corrections at each survey point (whether applied or not).
E_rms	E_rms	RMS after each bootstrap iteration.
Ftest_res	Ftest_res	F-test grid search results.
lat_sta_bs	lat_sta	Sensor latitude after each bootstrap iteration.
lats_ship	svy_lats	Latitudes of survey points.
loc_lolaz	loc_geo	Final sensor location (geographic coordinates).
loc_xyz	loc_xyz	Final sensor location (Cartesian reference frame).
lon_sta_bs	lon_sta	Sensor longitude after each bootstrap iteration.
lons_ship	svy_lons	Longitudes of survey points.
mean_drift_az	drift_az	Final sensor drift distance and azimuth.
R_mat	resol	Model resolution matrices after each bootstrap iteration.
sta	sta	Station name.
TAT_bs	tats	Sensor turn-around time after each bootstrap iteration.
twtcorr_bs	twts	Final two-way travel-times (twtts) at each survey point.

3.4 Figures

In conclusion we show the figures produced after running **OBSrange.py**.

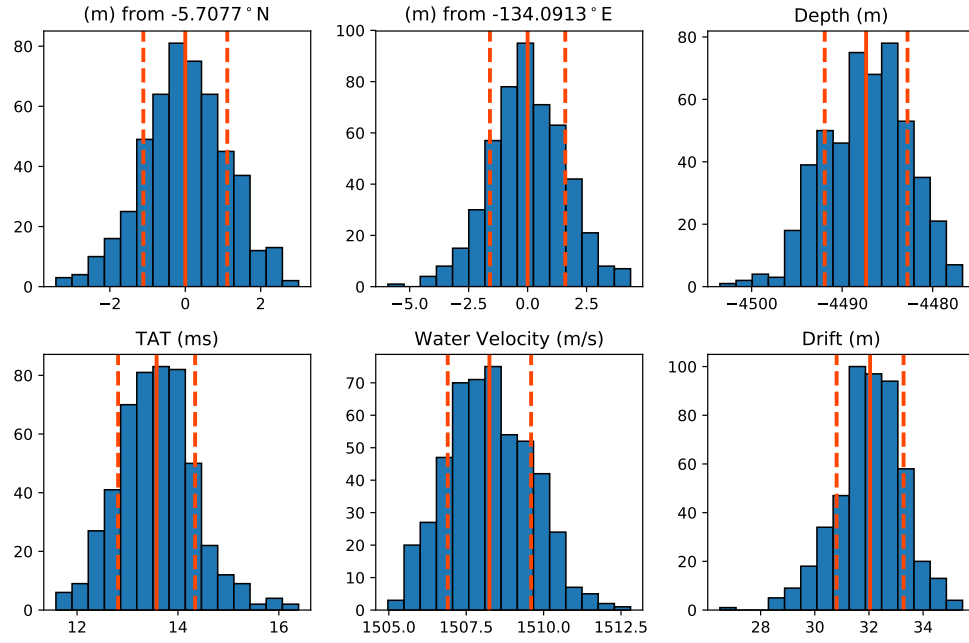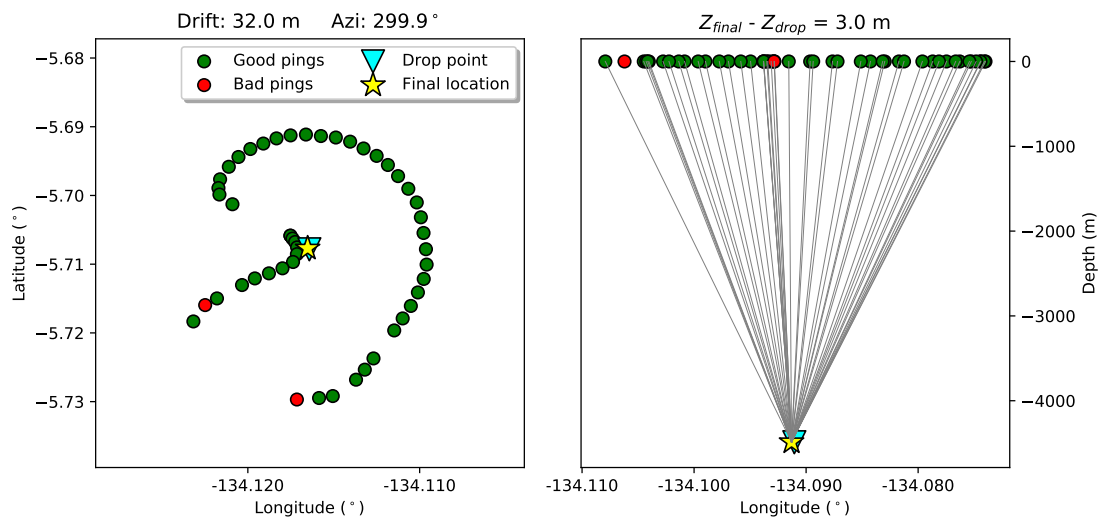Figure 4: Model parameter histograms.
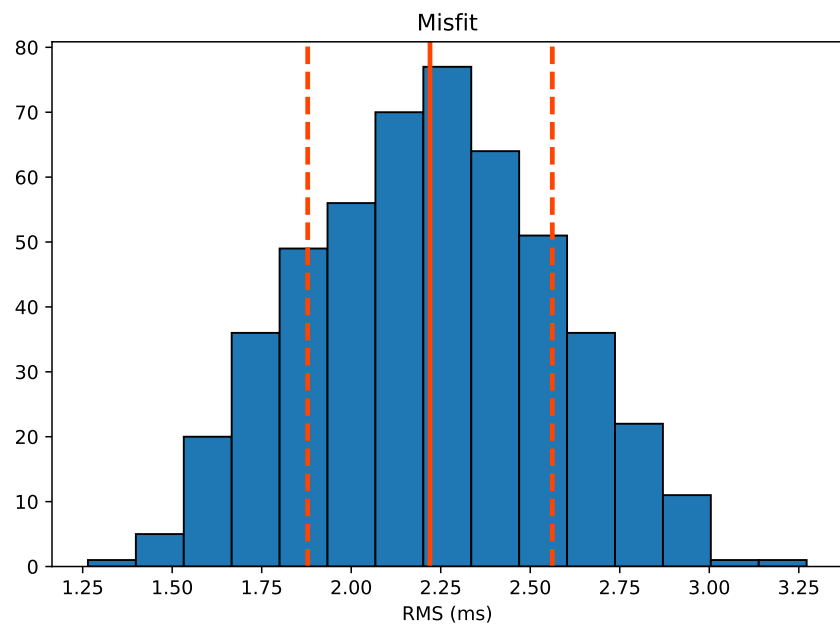
Figure 5: Survey maps.
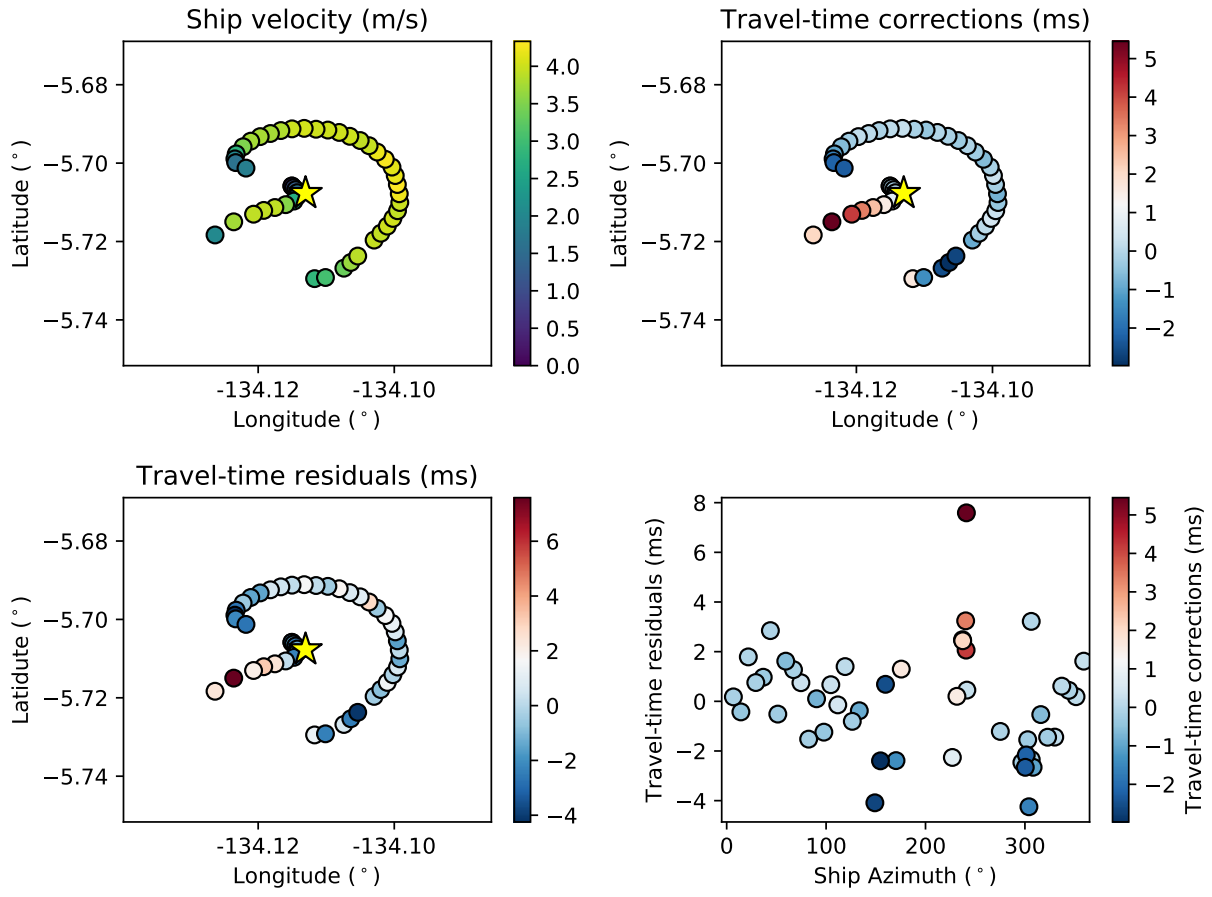

Figure 6: Final model misfit.

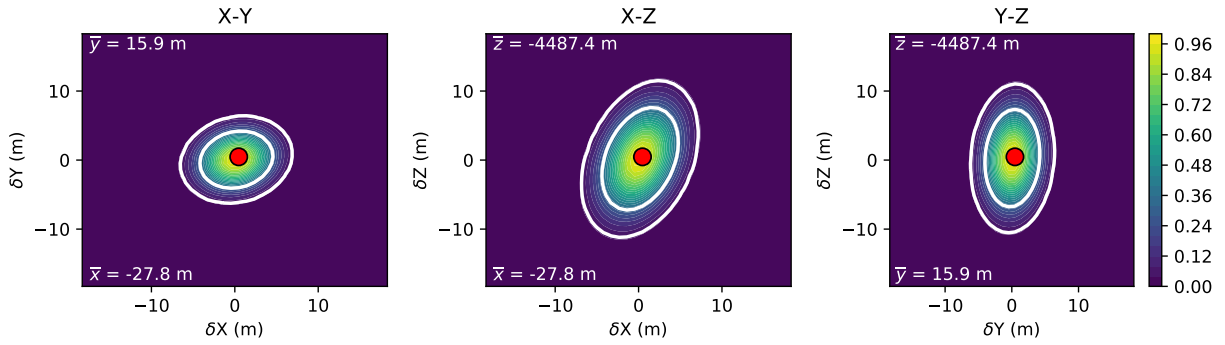Figure 7: Two-way travel-time residuals by suvery point
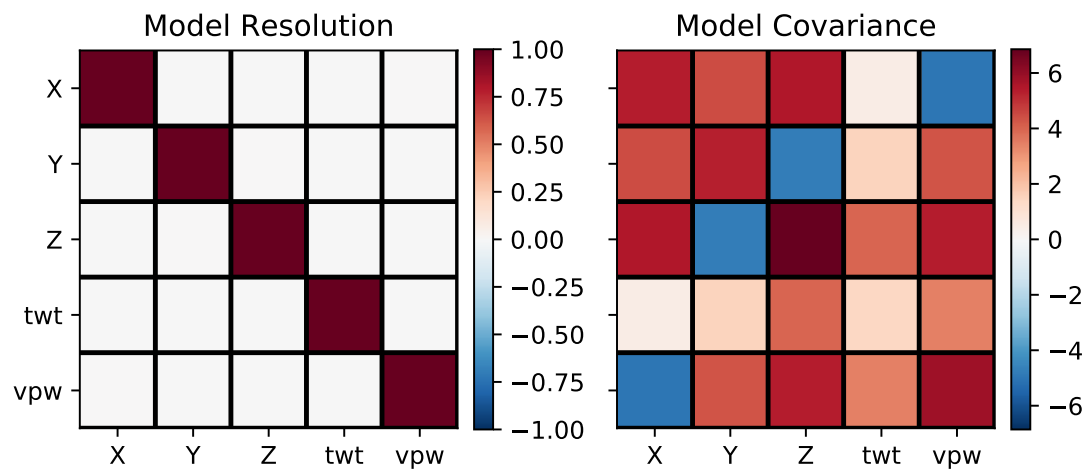

Figure 8: F-test derived location uncertainty

Figure 9: Final model desolution and covariance

References

Our paper.