OBSrange v1.0 README

Stephen Mosher, Josh Russell and Zachary Eilon

February, 2019

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, 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 don't 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 (it's 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
- *pickle*

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

```
Ranging data taken on:  2018-04-16 03:42:00.588000
Cruise:                 obs-cruise
Site:                   EE01
Instrument:
Drop Point (Latitude):  -4.94605
Drop Point (Longitude): -130.38178
Depth (meters):         4670
Comment:                confusing depth
==================================================

 6206 msec. Lat: 4 56.7506 S  Lon: 130 22.9231 W  Alt: 22.20 Time(UTC): 2018:106:10:42:35
 6206 msec. Lat: 4 56.7506 S  Lon: 130 22.9231 W  Alt: 22.20 Time(UTC): 2018:106:10:42:35
 6205 msec. Lat: 4 56.7497 S  Lon: 130 22.9234 W  Alt: 23.49 Time(UTC): 2018:106:10:42:43
 6205 msec. Lat: 4 56.7491 S  Lon: 130 22.9234 W  Alt: 24.17 Time(UTC): 2018:106:10:43:43
 6206 msec. Lat: 4 56.7477 S  Lon: 130 22.9239 W  Alt: 25.43 Time(UTC): 2018:106:10:44:43
 6205 msec. Lat: 4 56.7474 S  Lon: 130 22.9223 W  Alt: 25.25 Time(UTC): 2018:106:10:45:44
 6204 msec. Lat: 4 56.7552 S  Lon: 130 22.8851 W  Alt: 24.39 Time(UTC): 2018:106:10:46:44
 6206 msec. Lat: 4 56.8168 S  Lon: 130 22.8162 W  Alt: 25.79 Time(UTC): 2018:106:10:47:44
 6218 msec. Lat: 4 56.8816 S  Lon: 130 22.7224 W  Alt: 24.21 Time(UTC): 2018:106:10:48:44
 6246 msec. Lat: 4 56.9525 S  Lon: 130 22.6149 W  Alt: 23.71 Time(UTC): 2018:106:10:49:45
 6295 msec. Lat: 4 57.0161 S  Lon: 130 22.4979 W  Alt: 22.30 Time(UTC): 2018:106:10:50:45
 6359 msec. Lat: 4 57.0784 S  Lon: 130 22.3800 W  Alt: 20.71 Time(UTC): 2018:106:10:51:45
 6438 msec. Lat: 4 57.1397 S  Lon: 130 22.2617 W  Alt: 18.13 Time(UTC): 2018:106:10:52:46
Event skipped - Timeout or Badly formatted data was received
Event skipped - Timeout or Badly formatted data was received
 6584 msec. Lat: 4 57.2314 S  Lon: 130 22.0897 W  Alt: 16.39 Time(UTC): 2018:106:10:54:13
 6703 msec. Lat: 4 57.2942 S  Lon: 130 21.9712 W  Alt: 16.22 Time(UTC): 2018:106:10:55:14
 6831 msec. Lat: 4 57.3579 S  Lon: 130 21.8519 W  Alt: 16.05 Time(UTC): 2018:106:10:56:14
 6976 msec. Lat: 4 57.4197 S  Lon: 130 21.7348 W  Alt: 15.86 Time(UTC): 2018:106:10:57:14
  *7132 msec. Lat: 4 57.4839 S  Lon: 130 21.6154 W  Alt: 19.01 Time(UTC): 2018:106:10:58:15
Event skipped - Timeout or Badly formatted data was received
Event skipped - Timeout or Badly formatted data was received
Event skipped - Timeout or Badly formatted data was received
```

Figure 1: Example survey *.txt* file.

2.3 Coordinate System

During the sensor survey, whenever a ping is successfully received, the ship coordinates are logged as geodetic coordinates (latitude and longitude) using the ship’s GPS. Such coordinates can be seen in Figure 1. However ***OBSrange*** (notably the inversion within) works with locally approximated Cartesian coordinates. The geodetic coordinates are converted in ***OBSrange*** to a local Cartesian system (X, Y) using the World Geodetic System 1984 (WGS84) reference ellipsoid. After the transformation is applied, the Cartesian origin $(X = 0, Y = 0)$ of each individual survey pattern refers the corresponding instrument’s drop location.

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.

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.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.dforward	*dforward*	GPS-transponder offset (meters). If unknown set to 0. Positive means the transponder is further forward than the GPS.
par.dstarboard	*dstarboard*	GPS-transponder offset (meters). If unknown set to 0. Positive means the transponder is further starboard than the GPS.
par.E_thresh	*E_thresh*	RMS reduction threshold for the inversion
par.epsilon	*eps*	Global norm damping for stabilization.
par.if_raycorrect	*raycorr*	Option to apply a travel time correction for ray-bending. If you choose to do this you can either provide your own sound speed profile for each station or our code will calculate one for you. See *sspfile_dir/ssp_dir* below.
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.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. stuff
par.sspfiledir	*ssp_dir*	Path to directory of station sound speed profiles. If providing your own profiles, they **must** be named according to *SSP_stationname.txt*.
par.TAT	*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

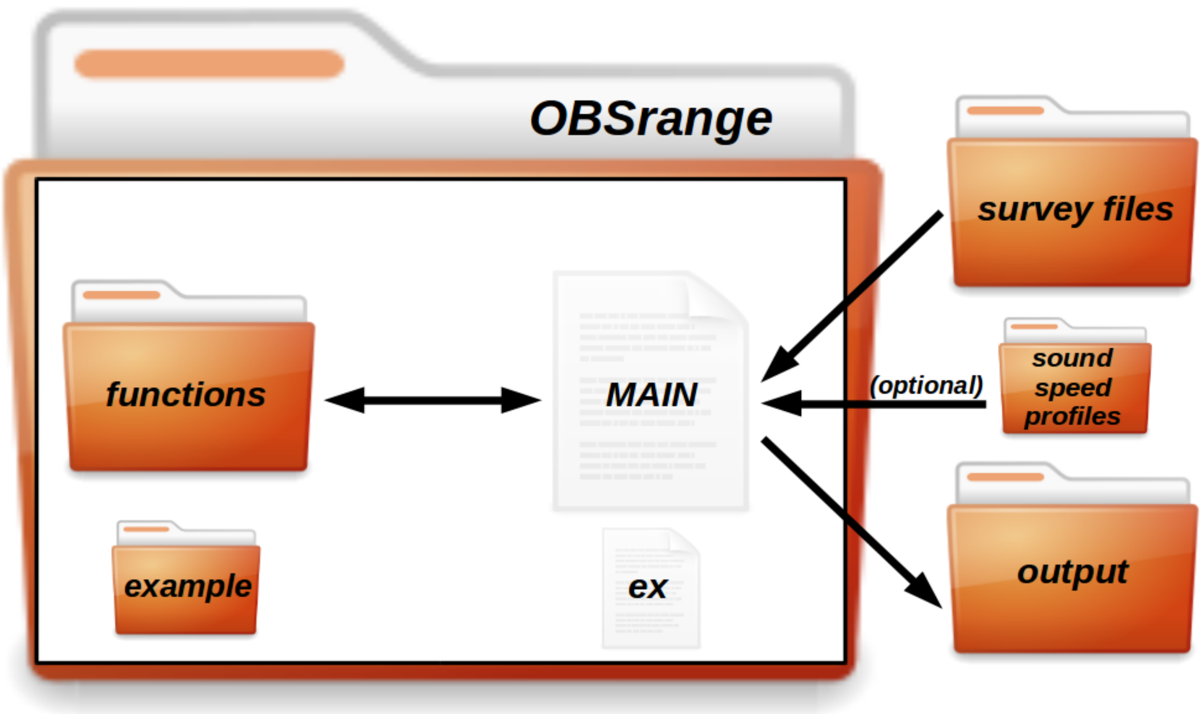


Figure 2: ***OBSrange*** structure.

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 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.

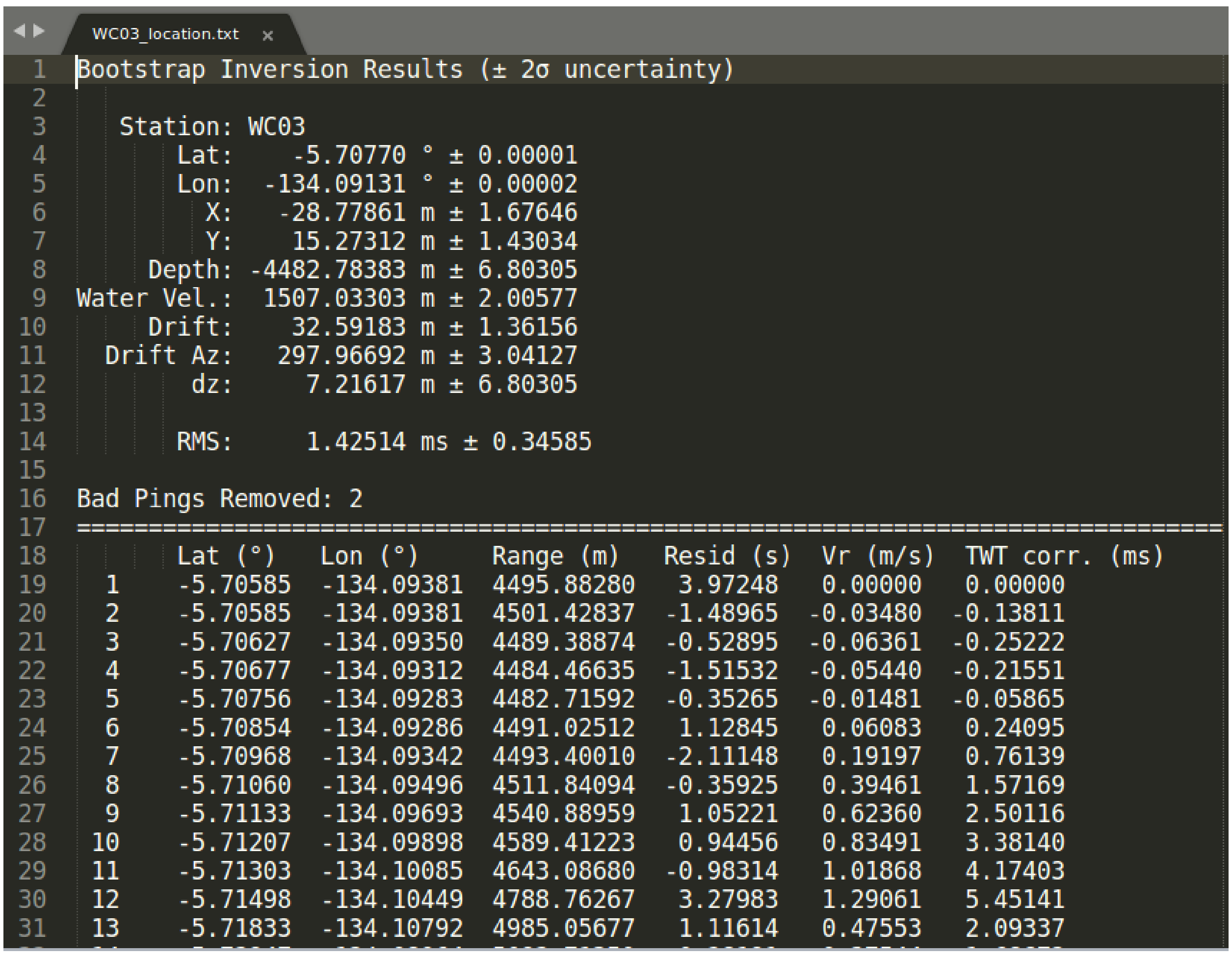
3 Example

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
Bootstrap Inversion Results (± 2σ uncertainty)

   Station: WC03
       Lat:    -5.70770 ° ± 0.00001
       Lon:  -134.09131 ° ± 0.00002
         X:    -28.77861 m ± 1.67646
         Y:     15.27312 m ± 1.43034
     Depth: -4482.78383 m ± 6.80305
Water Vel.:  1507.03303 m ± 2.00577
     Drift:    32.59183 m ± 1.36156
  Drift Az:   297.96692 m ± 3.04127
        dz:     7.21617 m ± 6.80305

       RMS:     1.42514 ms ± 0.34585

Bad Pings Removed: 2
==========================================================================
       Lat (°)    Lon (°)      Range (m)   Resid (s)  Vr (m/s)  TWT corr. (ms)
  1    -5.70585  -134.09381  4495.88280   3.97248   0.00000   0.00000
  2    -5.70585  -134.09381  4501.42837  -1.48965  -0.03480  -0.13811
  3    -5.70627  -134.09350  4489.38874  -0.52895  -0.06361  -0.25222
  4    -5.70677  -134.09312  4484.46635  -1.51532  -0.05440  -0.21551
  5    -5.70756  -134.09283  4482.71592  -0.35265  -0.01481  -0.05865
  6    -5.70854  -134.09286  4491.02512   1.12845   0.06083   0.24095
  7    -5.70968  -134.09342  4493.40010  -2.11148   0.19197   0.76139
  8    -5.71060  -134.09496  4511.84094  -0.35925   0.39461   1.57169
  9    -5.71133  -134.09693  4540.88959   1.05221   0.62360   2.50116
 10    -5.71207  -134.09898  4589.41223   0.94456   0.83491   3.38140
 11    -5.71303  -134.10085  4643.08680  -0.98314   1.01868   4.17403
 12    -5.71498  -134.10449  4788.76267   3.27983   1.29061   5.45141
 13    -5.71833  -134.10792  4985.05677   1.11614   0.47553   2.09337
```

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, 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.*). 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
-	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.
databad	Nbad	The number of pings removed via quality control.
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.
x_ship	svy_xs	x-coordinates of ship at each survey point.
x_sta_bs	x_sta	x-coordinates of sensor after each bootstrap iteration.
y_ship	svy_ys	y-coordinates of ship at each survey point.
y_sta_bs	y_sta	x-coordinates of sensor after each bootstrap iteration.
z_ship	svy_zs	z-coordinates of ship at each survey point.
z_sta_bs	z_sta	x-coordinates of sensor after each bootstrap iteration.
v_ship	svy_vs	Ship velocity at each survey point.

3.4 Figures

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

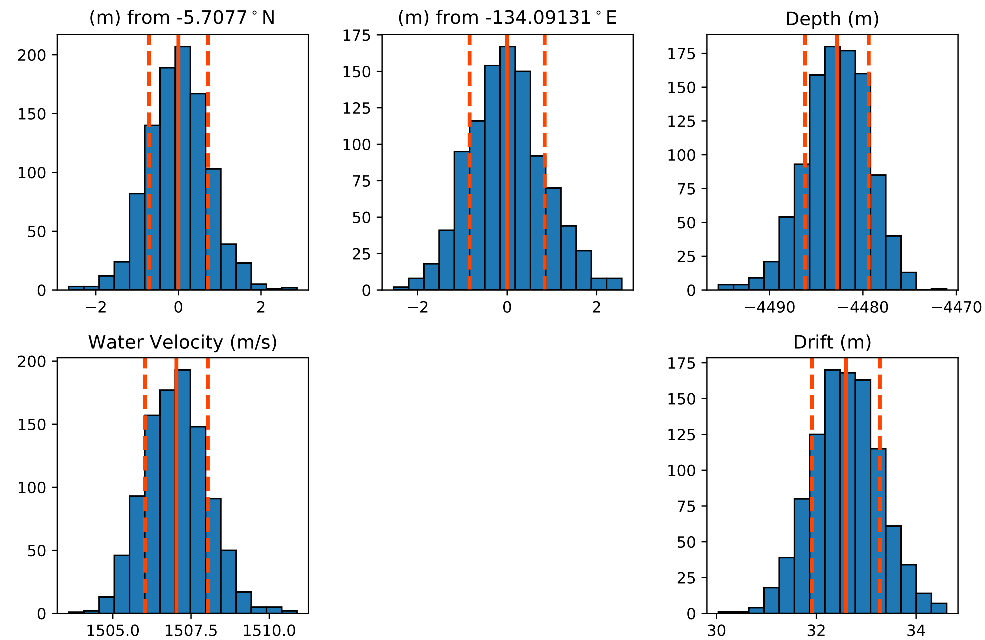


Figure 4: Model parameter histograms.

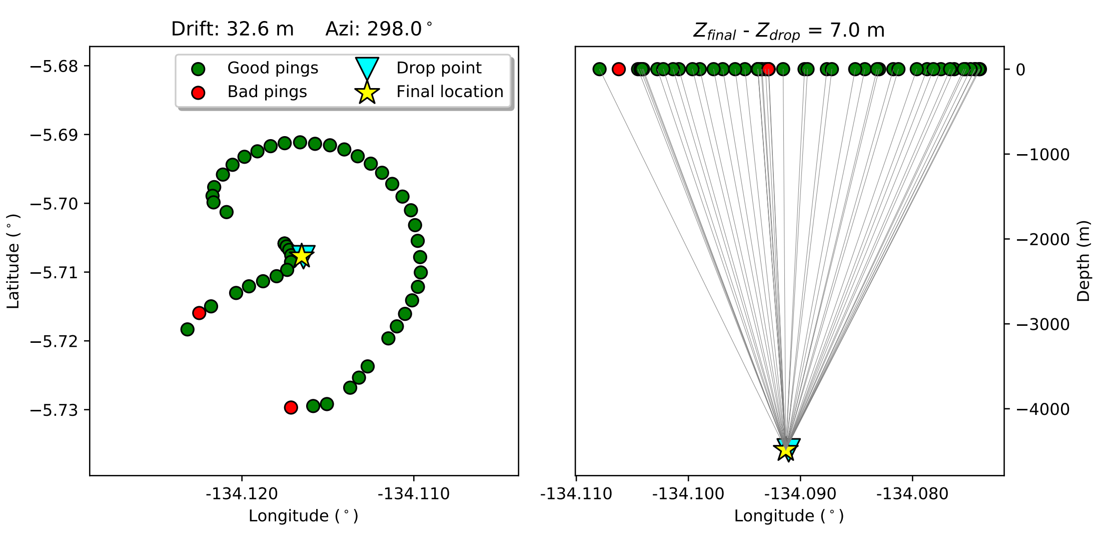


Figure 5: Survey maps.

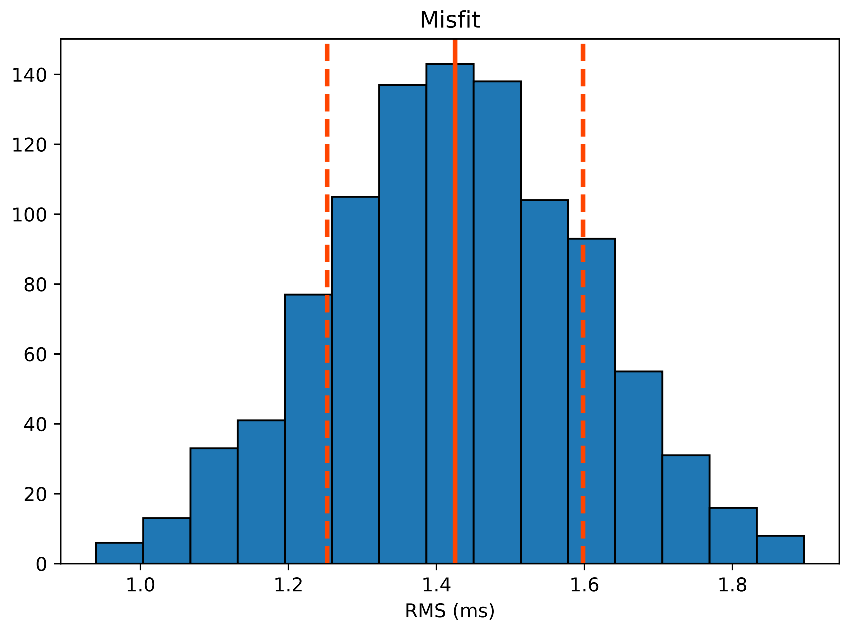


Figure 6: Final model misfit.

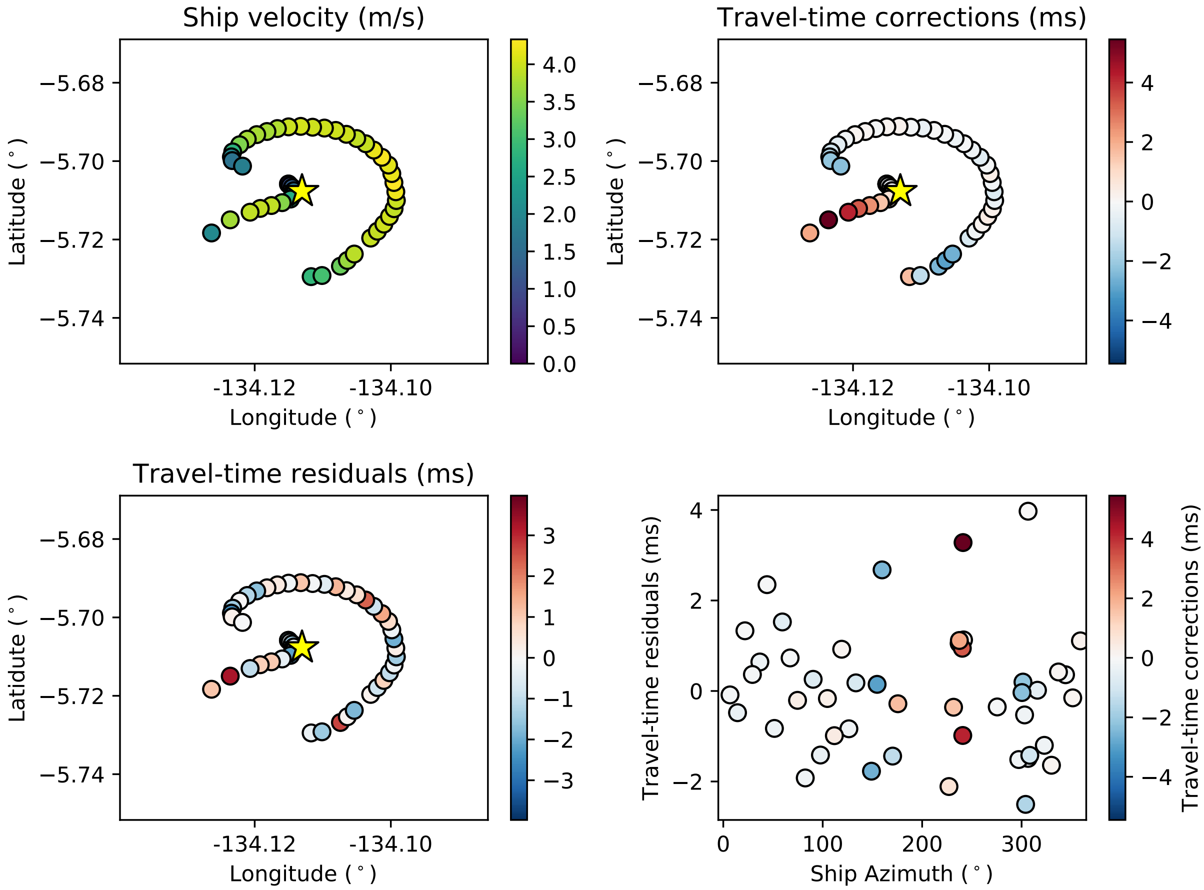


Figure 7: Two-way travel-time residuals by suvery point

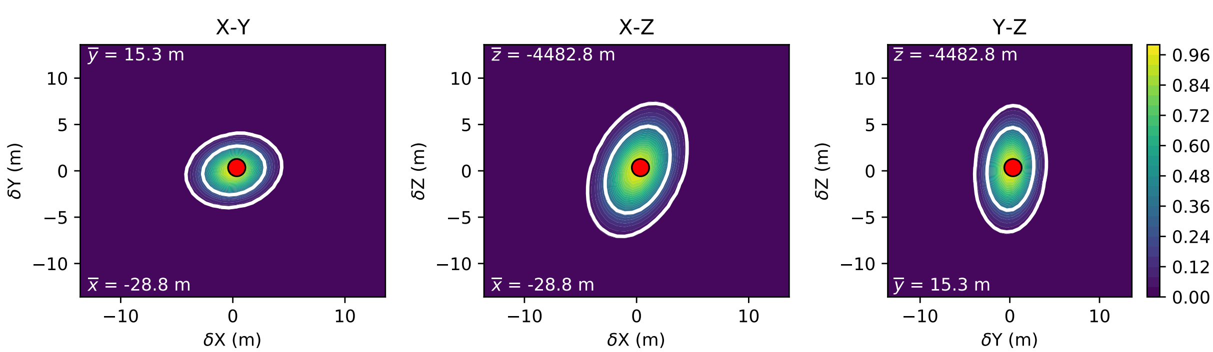


Figure 8: F-test derived location uncertainty

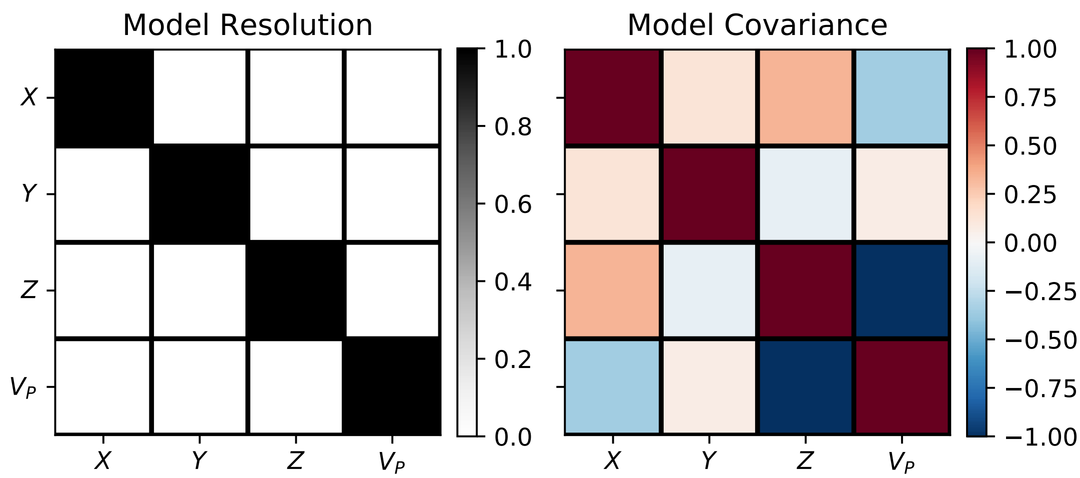


Figure 9: Final model desolution and covariance

References

Our paper.