Variable Latency Due To Extreme Depth Changes

By David P. Baalman, PLS, CFedS

My company was recently contracted by the US Army Corps of Engineers, Walla Walla District to conduct a sedimentation survey of the Dworshak Reservior in North Idaho. Dworshak Reservior is a large lake created by Dworshak Dam, the tallest straight axis dam in the United States at 717 feet tall. The scope of work for the project included 33 cross sections of the reservoir, from above the high water line on one side all the way across to above the high water line on the other side. This means that each line will include depths from as shallow as we can measure to nearly 700 feet in some cases. The side slopes of the reservoir are extremely steep, in many cases steeper than ½:1, with many nearly vertical cliffs both above and below water.

The hydrographic equipment we utilized consisted of an Odom CV100 single beam fathometer, and a Trimble R8 GPS receiver, using RTK mode (including RTK tides for water surface elevation monitoring). While this job presented many challenges, one that was extremely difficult was dealing with the combined latency offsets of the RTK and the fathometer. At first I attempted my normal approach, which is to run several test lines on the first day, after checking all other calibrations, and apply that average throughout the project. The results I got didn't sit right in my mind, the norm for this equipment is around 0.5 seconds, so when I got a big number, something around 2.5 seconds, it bothered me. I decided that since we would be collecting reciprocal runs on at least some of each line (to insure we collected data as close to each water's edge as possible), I decided to just set the latency offset at 0 and come up with an average latency for all lines after the fact, and apply that number in SBMax (SINGLE BEAM EDITOR) when processing.

Back at the office with all field work complete, I computed the latency for each line. The overall average came out to be 1.7 seconds, but my first clue that something was amiss was that the result for each line wasn't consistent. The lines at the top end of the reservoir (with the shallower water) showed a much smaller latency than those near the dam. I processed all the lines using the average latency of 1.7 anyway, and exported my final data using the Sort program, and brought it all into AutoCAD to further manipulate it per the client's requirements. That's where the magnitude of the issue really showed up. I had reciprocal runs on every line, for at least several hundred feet, usually on the steep side slopes of the reservoir. In the flat areas near the middle, I saw excellent repeatability, but on the steep side slopes saw as much as 30' of elevation difference between points in nearly the same location on runs going opposite directions.

After spending some time confirming every step of both the field and office procedures, I came down to the latency issue. As I said above, the fact that they latency was both different from what I normally see and variable depending on the water depth was a big clue. I selected a deep line near the dam and brought it into the Latency Test program, and began to play with different numbers for latency values. The 1.7 number I was using made the two runs line up near the middle of the slope, at around 250' depth, but near the water surface and down at the deepest areas the lines differed by quite a bit. By selecting a large latency number I could make the lower toes line up, and by selecting a smaller number the areas near the water surface would line up.

Past experience with similar surveys in much shallower water has shown me that latency can be consistent in depths from very shallow down to well below 100' depth, so I decided to re-

compute the latency in smaller chunks of 100’ each. I accomplished this by using the maximum and minimum depth limits in the Latency program and computing each line in 100’ depth bands. The overall average throughout the project for zero depth down to 100’ came out to be 0.6 seconds, right in line with what I am used to seeing with this equipment; however, the average for depths between 600’ & 700’ (an admittedly small sample size of just 2 lines) came out to be 3.7 seconds. That’s a huge difference! No wonder I couldn’t make an average across the entire job work.

FIGURE 1. *0.6 seconds latency applied. Note that the shallow depths line up nicely, but the toes of the slope do not.*

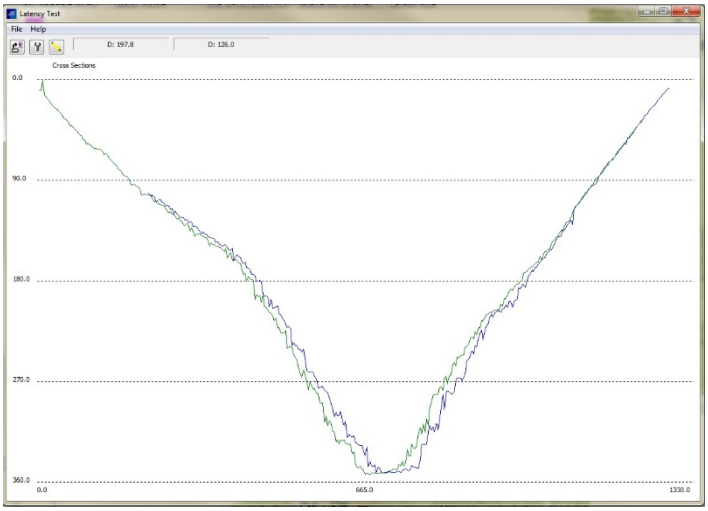
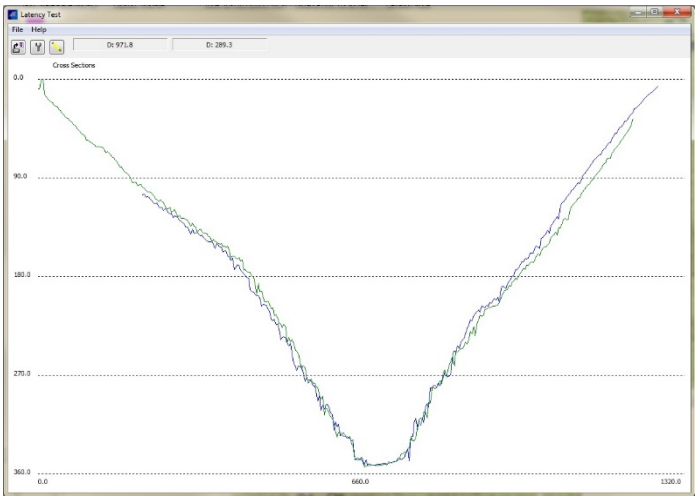


FIGURE 2. *2.6 seconds latency applied. Note that now the toes line up, but the shallower areas are not close.*



So now that I had a handle on the problem, the fun really began, while I tried to come up with an easy solution to deal with it. I spent some time trading phone calls & emails with HYPACK Tech Support, hoping for an easy answer as to how to apply a varying latency correction dependent on depth. Unfortunately, they didn’t have an easy answer, so I came up with a less than easy but definitely workable one. I decided I’d just use each average latency I came up with in the above test, and limit its application to depths in that specific band by using the

filters built into SBMax.

I opened up each log file, selected all files for that day, and set the latency for the depth band at hand, then set the filters to exclude every depth outside that depth band, and bingo, I'm down to just the depths valid for that latency. I then completed my single beam processing as normal, and ran the Sort program to thin the data per the client's requirements. The final step was to take the edited files and LOG file, and move them to a separate folder so that the next depth band wouldn't overwrite the edits for the previous depth band. Unfortunately, this method required each line to be run through SBMax up to 7 times to get each 100' band.

Results were excellent, reciprocal runs now agreed within normal tolerance, and runs near the shore agreed well with conventional shots taken in the water by the land crew. Some care did need to be taken at each transition point, but by bringing the final data into AutoCAD, I was able to select data that was smooth, and throw out the questionable data & large spikes.