Cycling around Galilee

Stewart A. Levin

ABSTRACT

Inspired by loop tie interpretation of intersecting 2D seismic lines, I adjust the depths of individual tracks from the venerable Sea of Galilee depth soundings in order to make their depth readings (more) consistent at crossing points. I also explore the notion that moveout along tracks is more reliable than absolute depths.

INTRODUCTION

For the last 15 years, SEP has struggled with the challenge of interpolating 1990 vintage depth sounding data from the Sea of Galilee. (Fomel and Claerbout, 1995; Karpushin and Brown, 2001; Guitton and Claerbout, 2003) In addition to noise spikes, the data were acquired at different times of day and times of year, with the result that depths recorded on intersecting tracks did not generally agree. The best results, while not totally artifact free, were obtained by Guitton and Claerbout (Guitton and Claerbout, 2004) and further refined by Claerbout with Fomel (Claerbout and Fomel, 2013).

While technically concerned with the Sea of Galilee dataset, the real reason for this exercise was to explore the option of using a modern sparse matrix direct solver instead of an iterative approach. With a direct solver, the matrix can be factored once and reapplied very quickly to many right hand sides. In this way, methodology developed for missing scalar data on a 2D plane may be applied to vector data, i.e. seismograms, without massively increasing computational cost.

THEORY

Ideally, depth soundings should satisfy Kirchhoff's Second Law; namely that following any closed loop would return to the same value. For the Sea of Galilee, one first interpolates new soundings where tracks cross and then should be able to follow any closed loop starting at any of those interpolated locations and come back to the same point with the same depth. Formally, by treating the sequence of soundings as a directed planar graph, any cycle should satisfy Kirchhoff's Second Law.

The Galilee depth soundings do not satisfy Kirchhoff's Second Law, though the reasons are not known and may be due to changes in the recording ship's weight, tides, or seasonal changes in fill and drainage of the Sea of Galilee. Let's assume for now that the depths from each track are internally consistent, i.e. not affected by tides or other intra-track sea surface variations. In this event, by appropriately shifting each track up or down, the depth soundings can be made consistent with Kirchhoff's Second Law. There is, of course, an inherent null space as all tracks can then be shifted upwards or downwards by a global constant and still satisfy Kirchhoff's Second Law. I constrain this in a simple way shown below. Any subsequent global shift may be used to honor alternative constraints such as the depth being zero at the shore line.

First some terminology. Let z_{ix} be the depth measured or interpolated at any location $\vec{x}$ along track i. Let Δ_{ijx} be the difference $z_{ix} - z_{jx}$ at the intersection of track i and track j at location $\vec{x}$. Finally, let σ_i be the to-be-determined constant adjustment to all depths on track i that will reduce or eliminate deviations from Kirchhoff's Second Law.

Setting up a least-squares formulation, let

$$F = \frac{1}{2} \sum_{ijx} (\Delta_{ijx} + \sigma_i - \sigma_j)^2 \tag{1}$$

be the function we wish to minimize. Note that $\Delta_{jix} = -\Delta_{ijx}$ and so the terms are symmetric when interchanging i and j, i.e. double counting each intersection.

Taking derivatives,

$$0 = \frac{\partial F}{\partial \sigma_k} = \sum_{jkx} (\Delta_{jkx} + \sigma_j - \sigma_k) \quad ; \quad j \neq k \tag{2}$$

defining a symmetric set of equations. Rearranging, produces

$$\sum_{jkx} (\sigma_k - \sigma_j) = \sum_{jkx} \Delta_{jkx} \quad ; \quad j \neq k \tag{3}$$

for $k = 0, \ldots, N-1$, where N is the number of tracks. As noted earlier, any global constant could be added to all the σ's without changing the solution. To avoid indeterminacy, I ask further that the answer be minimum L_2 norm, which is equivalent to adding an ϵ^2 to the matrix diagonal.

Points of intersection

Given adjacent pairs of points p_1, p_2 and q_1, q_2, when do the vectors from p_1 to p_2 and q_1 to q_2 intersect? Let the proposed point of intersection be $\alpha p_1 + (1-\alpha)p_2 = \beta q_1 + (1-\beta)q_2$ for some scalars α and β. If there is a solution with α and β between 0 and 1, the vectors intersect. Denoting $\Delta p = p_2 - p_1$ and $\Delta q = q_2 - q_1$ and rearranging yields

$$(p_2 - q_2) = \begin{pmatrix} \Delta p_x & -\Delta q_x \\ \Delta p_y & -\Delta q_y \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} . \tag{4}$$

In the special case that the determinant $\Delta p_y \Delta q_x - \Delta p_x \Delta q_y$ is zero, q_1 must lie on the line between p_1 and p_2 for there to be a solution. In particular, $|(p_2 - p_1) \cdot (q_1 - p_1)|^2$ must equal $|(p_2 - p_1) \cdot (p_2 - p_1)|\,|(q_1 - p_1) \cdot (q_1 - p_1)|$. Should this hold then setting β to 0 and 1 will determine if either q_1 or q_2 lies on the vector between p_1 and p_2.

When the determinant is not zero, we can use the explicit inverse formula

$$\begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \frac{1}{\Delta p_y \Delta q_x - \Delta p_x \Delta q_y} \begin{pmatrix} -\Delta q_y & \Delta q_x \\ -\Delta p_y & \Delta p_x \end{pmatrix} (p_2 - q_2) \tag{5}$$

to calculate α and β to see if both are in the interval [0,1].

FIRST EXPERIMENT

In order to apply the method to the Sea of Galilee soundings, we need to determine track boundaries. While these data are organized in sequential temporal order, there are no time stamps in the dataset and hence no direct way to determine when recording started and stopped. Hence the geometry of the soundings has to be used to subdivide the data. For this we can assume that large gaps between adjacent soundings are appropriate locations for track division. The question, of course, is what constitutes a "large" gap? The median of consecutive point-to-point distances is 26.6 meters. (Presumably the acquisition target was a nominal 25 meter spacing.) By taking "large" to be 10 times that median distance results in the 450 individual tracks. This is simply too many. Increasing to a factor of 40 reduced the number of tracks to 59, closer to the right ballpark. We do have one other guideline for track separation: boat speed. Using the median point spacing of 26.6 meters, then it takes about 10 seconds to cover that distance at 5 miles per hour. At that rate, there are about 360 soundings per hour and so around 4000 soundings per 12 hour day.

However, even with the initial overabundance of 450 available track shift parameters, the least-squares optimization, implemented using the Intel MKL sparse direct solver PARDISO rather than traditional SEP iterative conjugate gradients, improved the sum-of-squares misfit by less than 1.25%! The reduction to 59 tracks further reduced the misfit improvement to an even more paltry 0.13%. Clearly, I needed to manually examine the data to see whether there were data problems I had overlooked.

To address this, I modified my code to generate X-Y plots of selected subsets of the full dataset. As the order of X-Y-Z triplets in the dataset matched the sequence of acquisition, albeit without time stamps, this was a sensible way to inspect and manually subdivide the data into tracks. This led to the discovery of a few dozen that had coordinates either outside the lake or significantly displaced relative to the track to which they would have been expected to belong. Discarding these and using my judgment about where to separate the remaining points into tracks resulted in the 44 tracks shown in Figure 1 and a misfit improvement of 1.54%, still paltry.

Having cleared up X-Y locations and track separation, I then tackled issues with the depth soundings themselves. As previously reported, there were indeed isolated

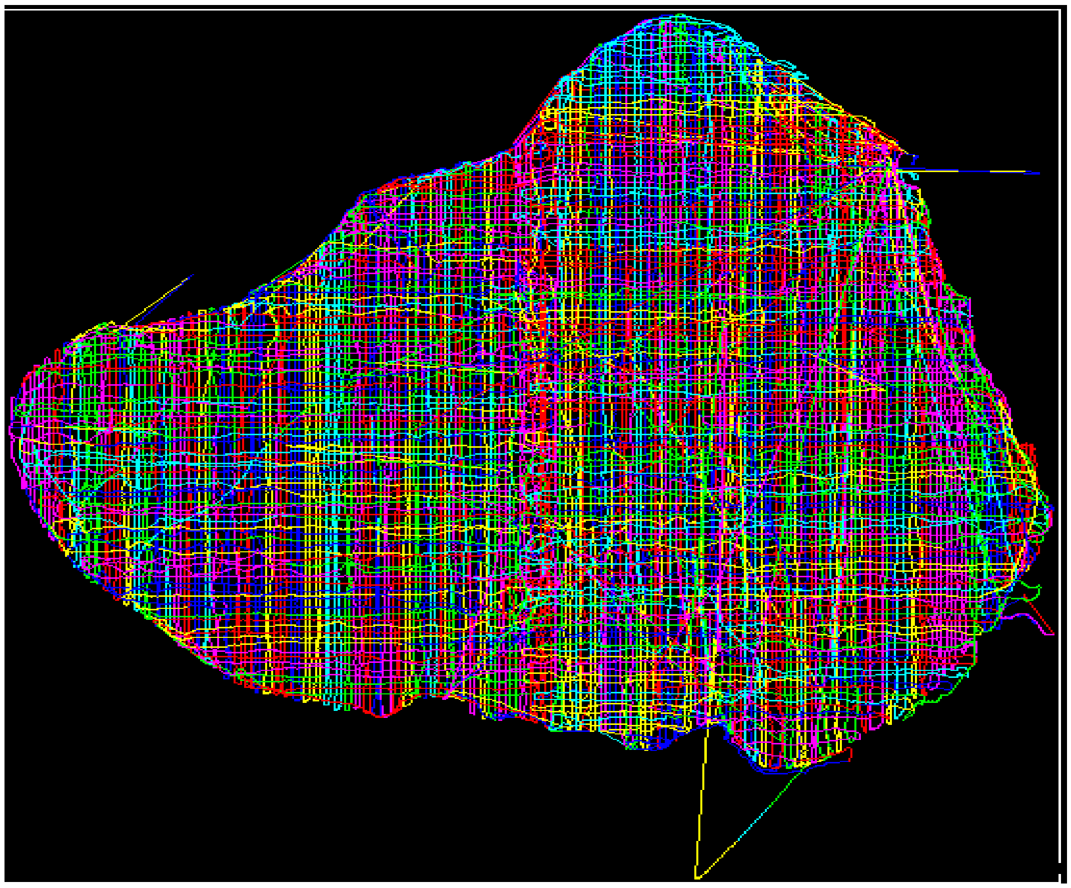

Figure 1: Plot of 44 individual tracks. [**CR**]

depth spikes all over the lake. Diagnosis: all the zero depth measurements were bogus because depths were all relative to sea level and the elevation of the Sea of Galilee is, according to Wikipedia, 212.07 meters below sea level. Cleaning out the zero depth points and rerunning the calculations improved the misfit to 4.83%, still a paltry improvement.

Conclusion: Track depth mismatches aren't caused by simple constant equipment elevation differences from day to day.

Where from here?

One straightforward step is to examine track self-intersections. While this, of course, depends upon where I placed track boundaries, any constant track shift should lead to equivalent depths where a track crosses itself. On the other hand, any clear trend in mismatches at these locations as we traverse a given track will supply a clue as to what model might be appropriate to fit track misfits.

SECOND EXPERIMENT

Fundamentally, though, the dataset was acquired for the purpose of detecting and outlining submerged ancient man-made structures. Finding the depth to them was secondary—their outlines were primary. For this reason it makes sense to look at the normals to the bathymetry; where they change rapidly and consistently is at edges. There is another very good reason to use the normals as our unknowns: while the absolute depths are unreliable, the differential changes in depth should be very consistent irrespective of when the boat passed over an area.

Estimating a vector field normal to the bathymetry

Having previously located all the track intersections, each pair of intersecting tracks provides corresponding track segments and thereby a local slope vector along of the depth topography in two different directions. Taking the cross product of the two slope vectors calculates a normal vector at that crossing point. As we expect normal vectors to point more or less upwards, I chose to rescale these vectors so that the z component was equal to 1. In this way, interpolating these normals to a regular grid did not involve quadratics, radicals or trigonometric functions.

I formulate the inversion problem by setting up a regular grid of unknown normal vectors with distance-weighted linear interpolation from the appropriate cell corners. The remaining track components are not ignored as they should, in a perfect world, be orthogonal to the to-be-determined normal vector field. To acknowledge this I impose the additional constraints that the normal vector field interpolated to a track segment midpoint should be approximately zero. This is done whether or not that segment also contains a track intersection.

This does not fully specify an inversion problem. To supplement the known data, I posit a regularization term.

Setting up the linear algebra

I formulate this as a least-square problem

$$\min \frac{1}{2} \left\{ \sum (\vec{v} \cdot \vec{\alpha})^2 + \lambda \sum |\vec{v}|^2 \right\} \ , \tag{6}$$

where the $\vec{v}$ represent the $(x, y, z = 1)$ unknown normals, α are interpolation coefficients summing to 1 for distance weighted averaging of the nearest $\vec{v}$'s to each track crossing and track midpoint, and λ is a regularization weight for the term favoring more vertical orientations. (λ could be spatially variable to account for places on the grid where we have additional knowledge.)

Setting this up as an explicit sparse matrix with a cell size of 50 m results in 237,720 unknowns and, using linear interpolation to estimate normals within cells, 18 times that number of nonzero elements in the sparse matrix, which is less than one hundredth of a percent of the size of the matrix. Running this with the Intel MKL PARDISO sparse solver took less than 2 seconds to complete. The output is shown in Figure 2 for $\lambda = 100$.

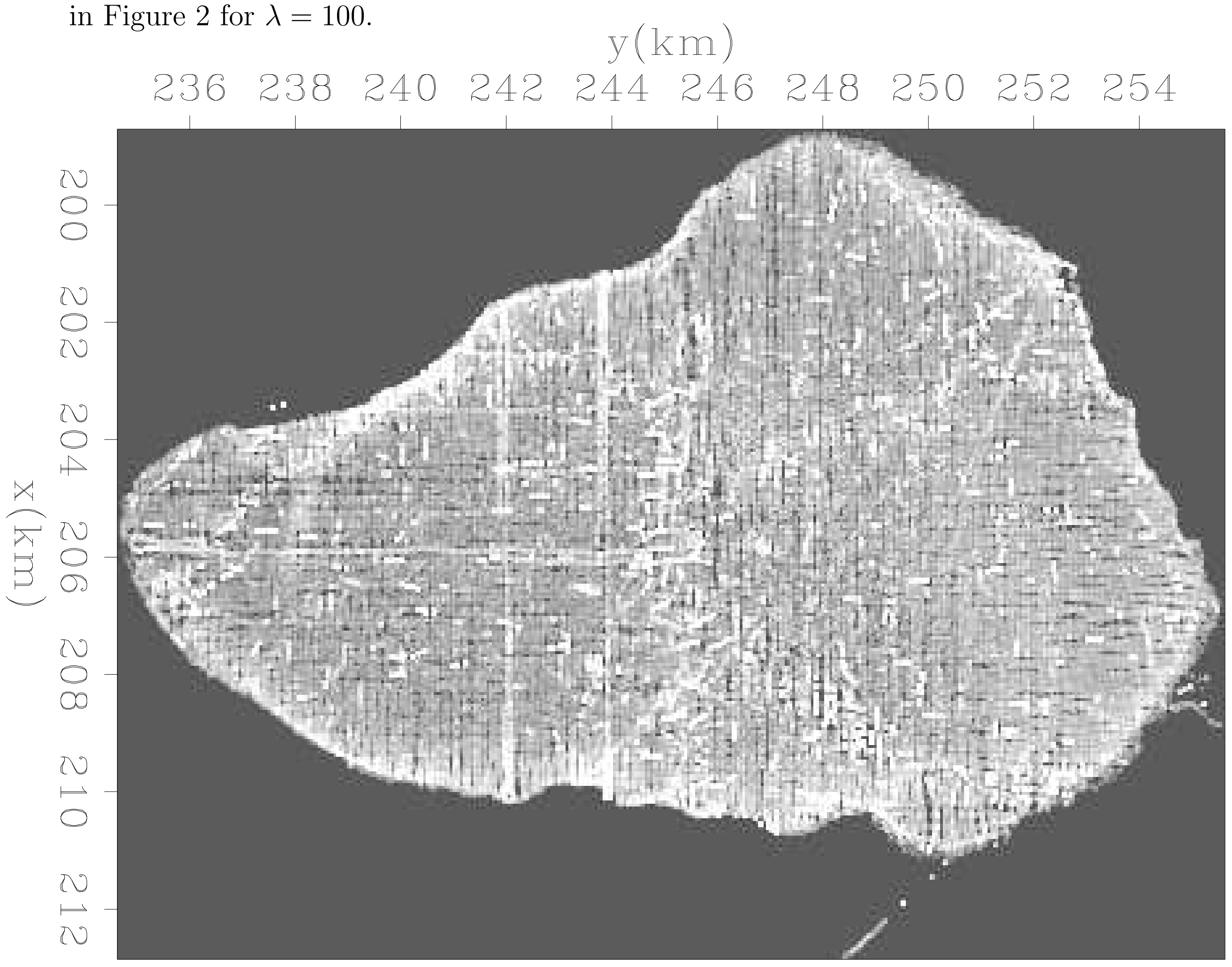

Figure 2: Magnitude of interpolated normals without smoothing regularization. [**CR**]

This result was disappointing but did show that estimated normals did generally tilt away from the shoreline and that my matrix setup did not have any outright errors. One immediate issue was that the track crossings only accounted for about 10% of the terms in equation 6 whereas they are the most important data for determining the normals. Computing using those terms weighted ten fold produced Figure 3.

As one can easily see, in addition to the clear track imprint, both of the two runs left holes where the solution was automatically vertical. While a larger bin

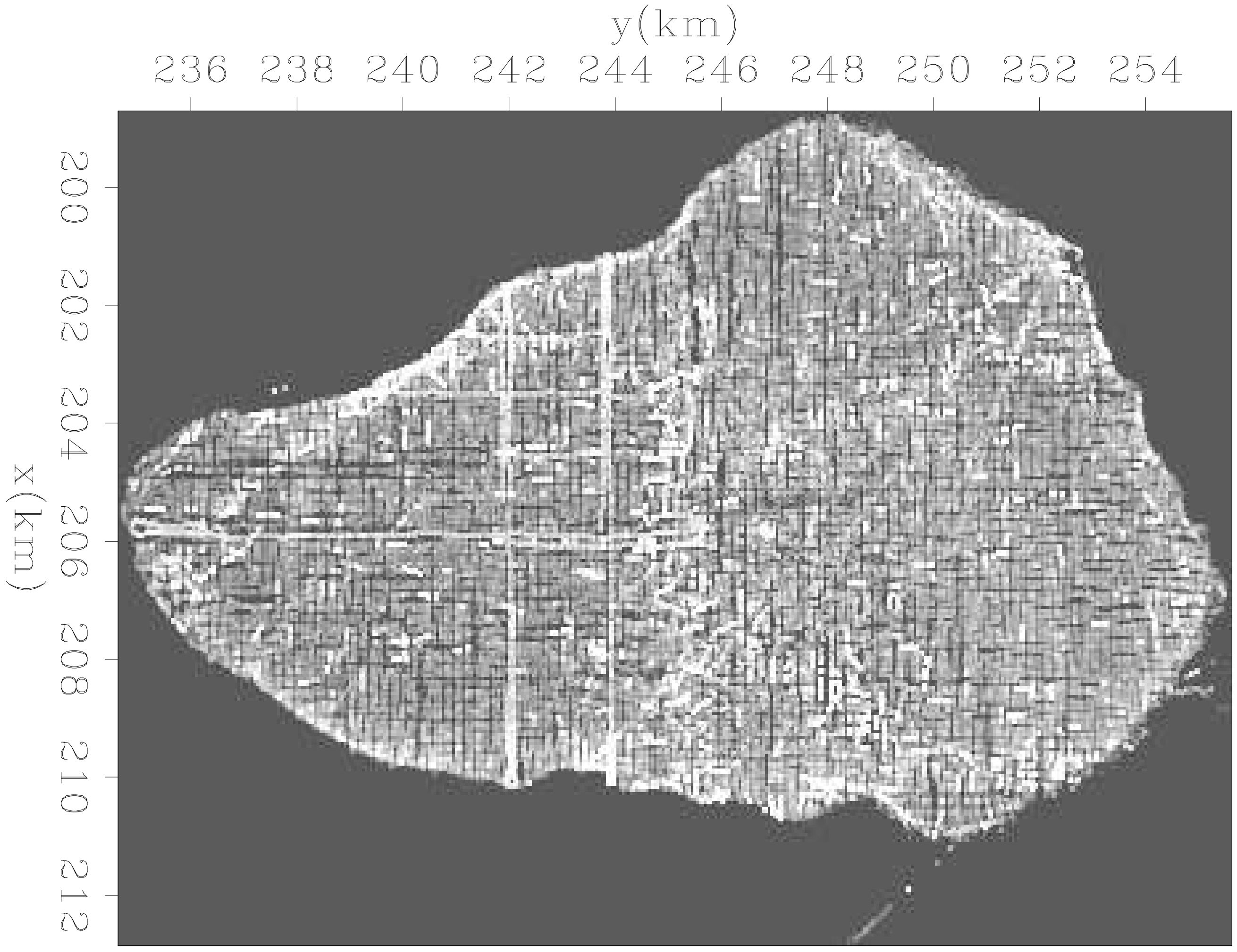

Figure 3: Magnitude of interpolated normals with track intersection contributions weighted by a factor of 10 and without smoothing regularization. [**CR**]

size would cover the holes, resolution would suffer and the track imprint would still appear. The holes problem was that the regularization term did not include an explicit smoothness term, only an indirect slanting of normals to the vertical lacking any other information. Adding Laplacian regularization produced Figure 4.

PRELIMINARY CONCLUSIONS

While there is always the possibility of programming bugs when setting up the internal sparse matrices and right hand sides[1], the results so far are at least reasonable if disappointing.

Possible avenues to pursue for the first experiment include adding horizontal dis-

[1]Indeed there are setup issues at the very edges of the grid which I masked by padding dummy cells around the border.

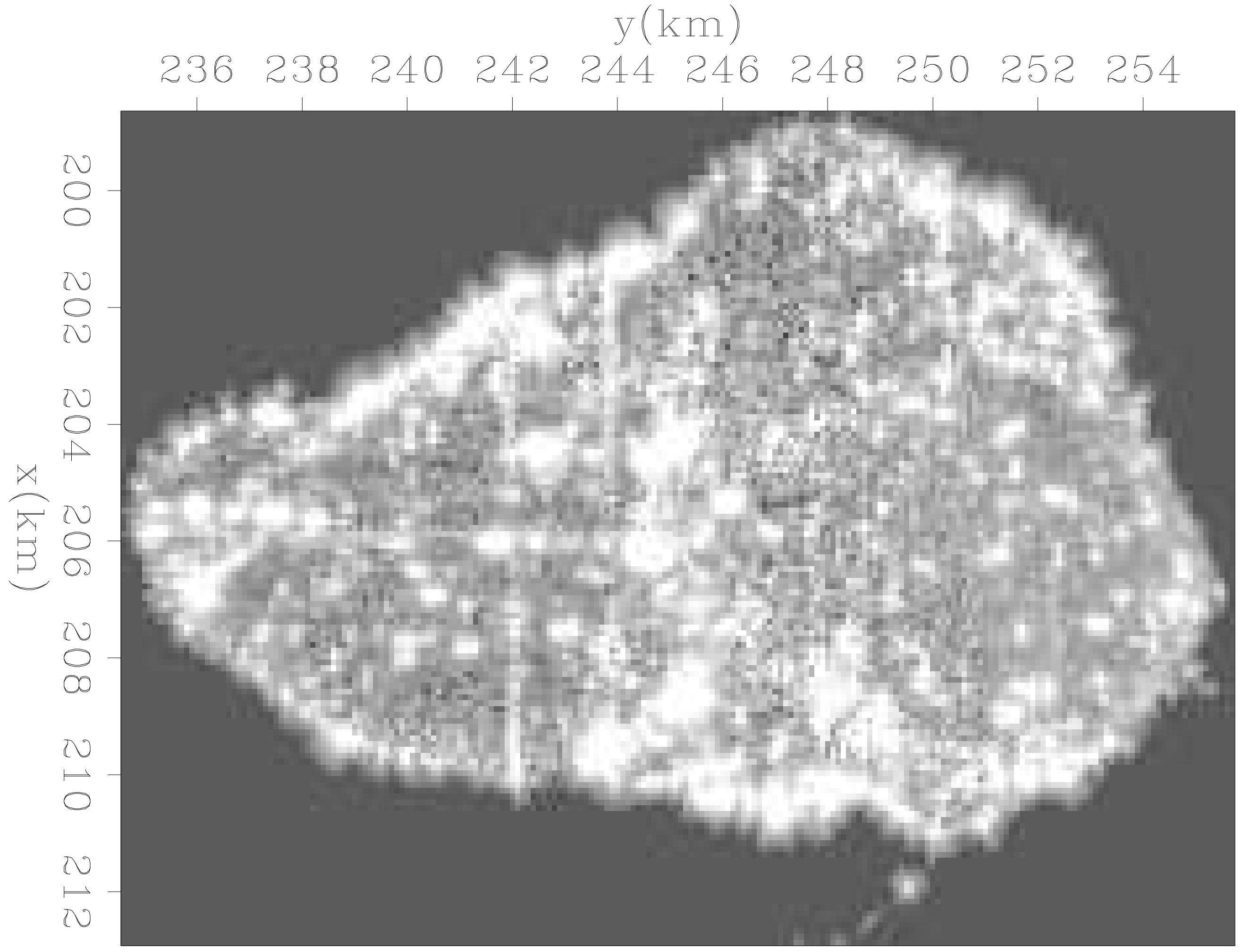

Figure 4: Magnitude of interpolated normals with both track segment midpoint contributions downweighted by a factor of 10 and with Laplacian smoothing regularization. [**CR**]

placements as unknowns to estimate and digging into trends in self-intersections. It may also be of value to tease out artifacts due to the boat rocking from side to side during recording. For the second experiment, the local slope estimates, currently based on connecting each consecutive sounding with a linear segment, could be extended by fitting a higher order interpolant to refine the tangent slopes that enter the calculations.

REFERENCES

Claerbout, J. F. and S. Fomel, 2013, Image Estimation by Example: Geophysical Soundings Image Construction: Stanford Exploration Project.

Fomel, S. and J. Claerbout, 1995, Searching the Sea of Galilee: The splendors and miseries of iteratively reweighted least squares: SEP-Report, **84**, 259–270.

Guitton, A. and J. Claerbout, 2003, Interpolation of bathymetry data from the Sea of Galilee: A noise attenuation problem: SEP-Report, **113**, 399–416.

———, 2004, Interpolation of bathymetry data from the Sea of Galilee: A noise attenuation problem: Geophysics, **69**, 608–616.

Karpushin, A. and M. Brown, 2001, Whitening track residuals with PEFs in IRLS approach to the Sea of Galilee: SEP-Report, **110**, 133–140.