REPRESENTING KRIGED TRAVEL TIME CORRECTION SURFACES WITH OPTIMAL TESSELLATIONS

James R. Hipp, Sanford Ballard, Marcus C. Chang, and Lee A. Jensen

Sandia National Laboratories

Sponsored by National Nuclear Security Administration
Office of Nonproliferation Research and Engineering
Office of Defense Nuclear Nonproliferation

Contract No. DE-AC04-94AL85000

ABSTRACT

Enhancing the accuracy and performance characteristics of event location is critical to seismic monitoring, during both the event association processing and event characterization stages. Recently, methods to compute the base model path-correction for travel time, azimuth, and slowness have been improved using the Bayesian Kriging approach.

The current Bayesian Kriging approach has proven to be a near ideal technique for providing robust estimates for path corrections, including both value (travel-time, azimuth, slowness, magnitude, etc.) and associated error, at specific interpolation locations on the Earth. However, due to the nature of the matrix inversion process, it has proven to exhibit less than desirable performance qualities when even moderate numbers of ground-truth events are involved in the calculation. For this reason, methods that yield the desired result with nearly equivalent accuracy, but with much higher performance, have been sought.

We have implemented one such method where an optimal tessellation representation of the Bayesian Kriging value and error surface is generated in-lieu of using the Bayesian Kriging approach directly. The new approach interpolates locally on the tessellation using natural-neighbor interpolation. The tessellation is optimized in a pre-processing step to minimize the difference between the true Kriged path-correction surface and the approximate tessellated surface. The technique uses successive triangle sub-division steps to sub-divide existing triangles into four new, smaller, triangles thereby reducing the difference between the true and approximate surface with each successive sub-division. The sub-division terminates when a preset absolute error tolerance is achieved as prescribed by the executing client. Mesh smoothing techniques are employed to ensure high-quality triangle formation following each sub-division step.

One draw-back to this approach is that data load times can increase significantly over that required by the Bayesian Kriging model. This limits the practicality of this approach to a server based application supplying requesting clients with interpolation results using a load-once run-many calculational approach. On the other hand, observed calculation times have improved by significant factors (and in some instances several orders of magnitude) over the Bayesian matrix inversion technique. In general, the amount of improvement depends strongly on the number of ground-truth events used in the Kriging approach, but is affected only slightly by the number of node points that comprise the tessellation.

This report presents performance comparisons between tessellated and Kriged representations including accuracy of representation and data load times required to initiate interpolation. The comparisons are calculated using the LocOO event location software to perform a series of benchmark location calculations.

OBJECTIVES

Obtaining an accurate location is one of the most important tasks in assessing any geo-seismic event. For large, well-recorded events, high-quality seismic locations can be obtained using global one-dimensional travel time models such as IASPEI91 (Kennett and Engdahl, 1991) or ak135 (Kennett et al., 1995). This is both because teleseismic distance paths have most of their lengths in the deeper, laterally homogenous parts of the Earth, and because whatever deviations do occur tend to be averaged out if the azimuthal coverage is good.

However, as the size of the event decreases and fewer stations detect the event, global one-dimensional models do not work as well and resulting mislocations can be large. Further, if the location error associated with the model is not properly accounted for in the location algorithm, then the associated confidence ellipses calculated for many smaller events do not actually contain the true locations as often as the calculated confidence would indicate.

Recently, Kriging methods have been employed to both enhance location estimates while simultaneously improving the associated error estimate (Shultz et al., 1998). These Kriged travel-time path corrections have significantly improved location estimates while simultaneously enhancing the associated accuracy of the estimate.

Unfortunately they come with a rather significant computational cost. The Bayesian Kriging method requires the inversion of a square matrix whose size is equivalent to the square of the number of ground-truth observations used in the kriging calculation. The inversion solution time is proportional to the cube of the number of ground-truths. The elements of the matrix are position dependent and force a new inversion calculation for each new interpolation location. With observation counts routinely composed of several hundred to a few thousand observations, Kriging calculation solution times are easily one or two orders of magnitude more costly than standard base model evaluations.

In order to alleviate this computational bottleneck, we propose an alternative data representation which can maintain both the path correction estimate and associated error to within a user specified tolerance. This new tessellated representation trades the computational inefficiency of Kriging with memory resource availability to achieve performance that is more consistent with the traditional one-dimensional velocity models mentioned above.

RESEARCH ACCOMPLISHED

Optimal Tessellation

We propose an approach where an optimal spherical geometry tessellation of well-shaped triangular elements is developed as an alternative representation for one or more Kriged surfaces. By optimal we mean a triangular tessellation whose nodal density and distribution characteristics are designed to meet a user specified absolute error tolerance with the original Kriged surface(s). The tolerance is specified such that the absolute value of the difference between the Kriged representation and the tessellated representation is less than the absolute error tolerance everywhere within the Kriged representation domain. With this definition we can produce an alternative data representation that matches the original Kriged, or true, surface, while ensuring that the tessellation is not over-refined with unnecessary nodes. This approach will minimize the amount of storage required to store the alternative data representation while guaranteeing that the error condition is met.

Obviously, in order to determine the difference between a true (Kriged) surface and a tessellated surface we must have a means of interpolating the tessellated surface. Many methods exist to interpolate a spherical geometry tessellation. Representatives include linear, natural-neighbor (Watson, 1992; Sambridge et al., 1995), or moving least-squares (Levin, 1998), as examples of increasing interpolation complexity, respectively.

Generally all of these methods are local interpolation techniques and only require a small subset of triangles that immediately surround the interpolation location to perform the calculation. The linear approach only requires data associated with the three nodes of the triangle that actually contains the interpolation location. The natural-neighbor interpolant requires data from the nodes of natural-neighbor triangle simplex, which are the triangles whose circum-circles (circles that pass through all three nodes of a triangle) contain the interpolation point. The natural-neighbor interpolation technique requires that the tessellation be a Delaunay representation (Delaunay, 1934), which is a set of triangles whose maximum angles have been minimized. The moving least-squares technique can use any local set of data, but generally the natural-neighbor simplex is a good choice. The moving least-squares technique finds a set of coefficients that minimize a standard least-squares functional (usually a polynomial) where the functional is weighted with a set of position dependent basis functions whose value is zero near the boundary of the local simplex but maximally large near the interpolation location.

The linear approach is the fastest, of course, but lacks smoothness and 2nd order differentiability conditions. The natural-neighbor approach has conditions similar to bi-linear or quasi-quadratic smoothness but is not differentiable at the tessellation nodes. The moving least-squares technique is continuous and infinitely differentiable everywhere, but takes the longest to evaluate.

For purposes of this paper we have selected the natural-neighbor technique to illustrate the optimal tessellation and to compare its performance to the Kriged surface approach. Regardless of the approach each method requires that a search of the containing triangle be accomplished before the interpolation calculation. This is the same for either interpolation method and tends to take on the order of $O(\sqrt{N})$ per search (Lawson, 1977), where N is the number of nodes (or triangles, which is generally twice the number of nodes in a spherical geometry tessellation). This value is precise for an equally spaced grid but is only an average taken over a set of random search locations for a tessellation possessing a variable node density whose nodal distribution is completely unstructured.

Tessellation Construction

Before discussing performance issues associated with optimal tessellations we need to briefly describe how they are assembled. Optimal tessellations are constructed in a set of three staged processes which include: 1) initial mesh construction and densification; 2) mesh accuracy refinement; 3) and mesh decimation. The first step is used to arrive at an initial construction that is reasonably dense where excessive surface curvature is expected, but not overly refined where there is no curvature. For Kriged path correction surfaces the primary curvature occurs around the observation points while very little curvature is found at distances that are 10 or 15% greater than the Kriged surface observation correlation range. This means that a technique that builds well shaped triangles that are coarse at distances far from the Kriged ground-truth observations, but increases the nodal density to a prescribed maximum around the ground-truths can be used to construct the initial mesh. Figure 1 illustrates such a configuration for an arbitrary set of ground-truth observations. Notice the high-density node distribution located around the ground-truths (blue points) which is gradually reduced, beginning at a preset distance from the ground-truth, to a coarser density at distances further away. This method has proved very effective for defining an initial guess for a specific optimal tessellation.

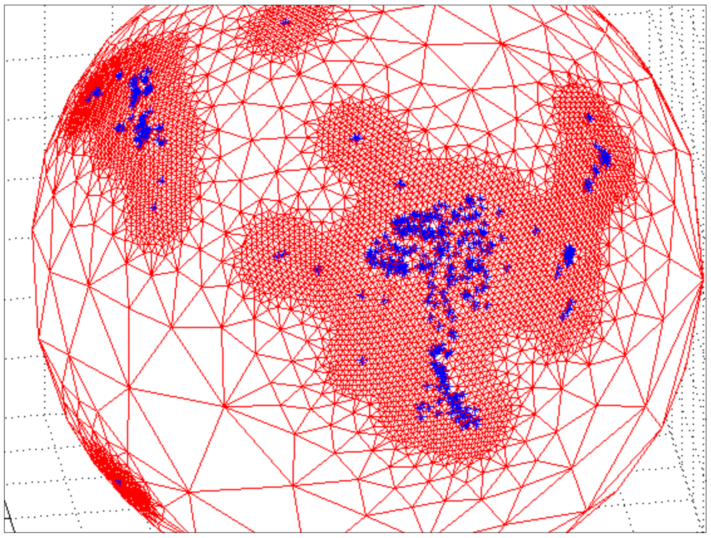

Figure 1. Initial Tessellation Construction

After the initial mesh has been constructed we can begin the accuracy refinement stage. In the accuracy refinement stage each triangle is interpolated in a sub-divided fashion at several internal points and compared with the true surface which is interpolated at the same locations. If the change in value of a surface attribute that is undergoing refinement exceeds the user defined relative sub-division error criteria across a sub-divided triangle, then the sub-triangle is sub-divided further and the algorithm is recursively called to continue the comparison on all of its' subdivided components. This process continues until the relative change in the value across the surface for a sub-triangle is less than a user prescribed relative error. If the relative error criteria is met then the algorithm returns (recursively) with the maximum discovered error for the triangle undergoing refinement.

Figure 2 illustrates the typical error distribution across a triangle during the accuracy refinement stage. Figure 2a depicts the true surface while Figure 2b shows the tessellated approximation. The absolute value of the difference between the two surfaces is shown in Figure 2c. These figures depict a typical triangle undergoing its first refinement. Since the natural-neighbor interpolator is exact at the nodes the surface difference always goes to zero at those locations and maximizes somewhere near the edge midpoints or in the interior of the triangle.

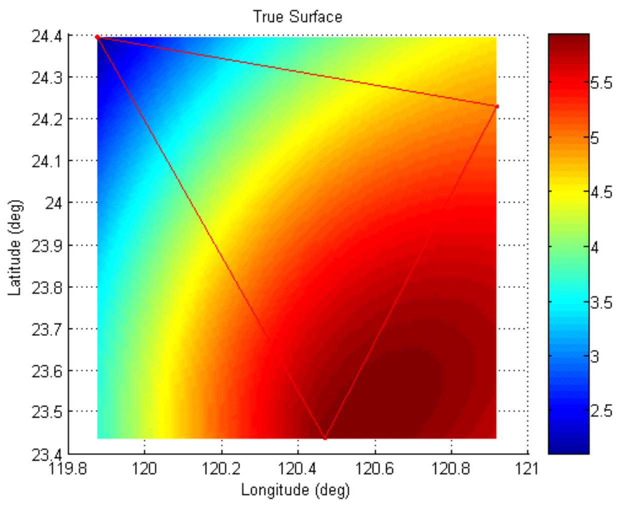

Figure 2a

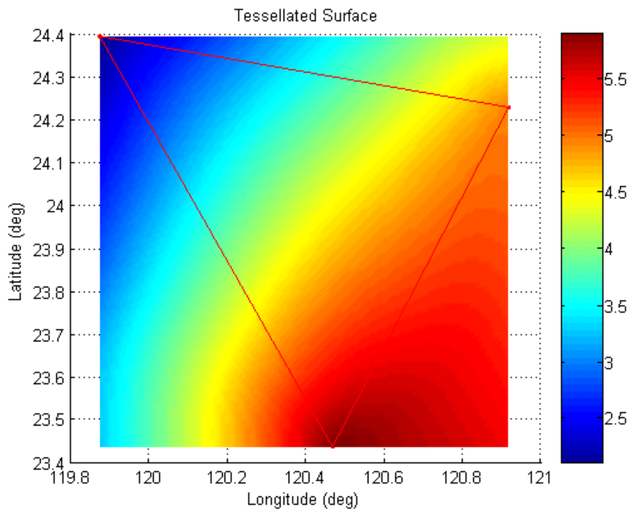

Figure 2b

If at any point during the refinement the error exceeds the user defined absolute error for a specific Kriged surface within the triangle undergoing refinement then the algorithm exits and the triangle is marked for refinement. When all triangles have been checked for refinement those that have been marked are sub-divided and the resulting new nodes are added to the tessellation and smoothed. Then the refinement process is repeated for all newly formed triangles and their immediate adjacent neighbors. The refinement process continues until no more triangles are in need of refinement. Figure 3 shows the previous dense mesh after having had 42 separate surfaces refined onto the initial grid. The resulting tessellation has approximately 144,000 nodes and was refined to an absolute error specification of .01 seconds between the true surface value and its tessellated equivalent.

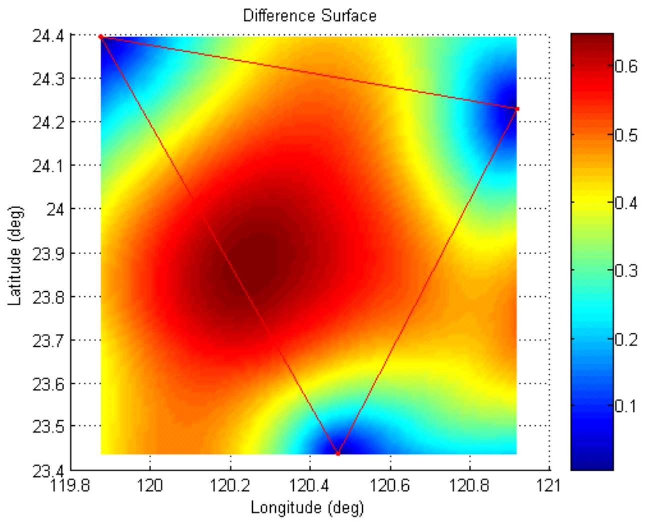

Figure 2c

The final construction stage is the decimation stage. Here any triangles that were never refined during the previous stage are checked to see if they are overly accurate. This is accomplished by removing the triangle (and generally, any associated neighbors) and testing to see if the new coarser set of triangles meets the absolute error criteria. If they do then the process is repeated. If they do not then the previous assembly of triangles is restored and marked as refined. This continues for all triangles that are undergoing decimation inspection until none remain.

The final resulting tessellation guarantees that all attributes undergoing refinement onto the tessellation can be interpolated to within the user defined absolute error criteria.

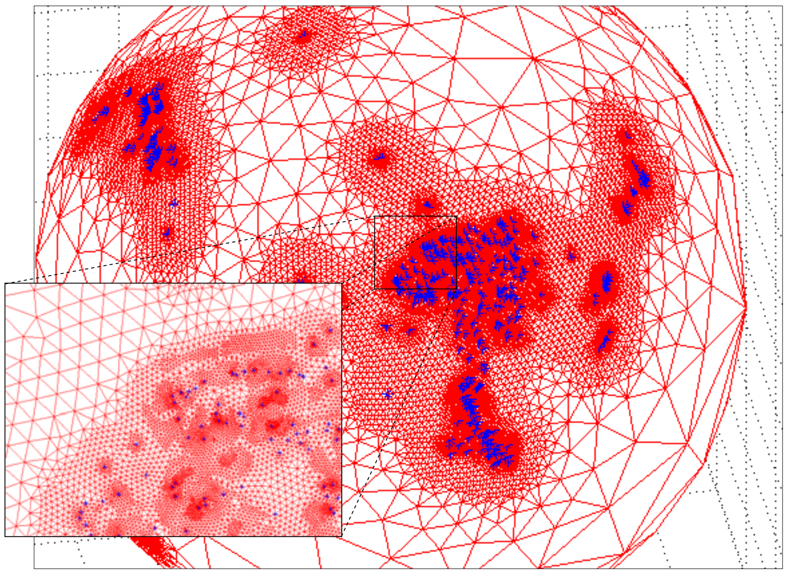
Figure 3. Final Refined Tessellation

Performance Considerations

Now that we have defined an optimal tessellation and briefly described how to construct them we can begin to examine some of their performance issues. We shall begin by defining the run-time behavior of the natural-neighbor tessellation interpolator using a triangle-walk searcher. This behavior can be characterized by the expression

$$t_I = c_{tw}\sqrt{N_N} + c_{iw} + c_{iws} \tag{1}$$

where t_I is the average time to interpolate a single location in any arbitrary tessellation, c_{tw} is the triangle-walk proportionality constant that gives the walk time as a function of the square-root of the local node density (or the total node count, N_N, if equally spaced on average), c_{iw} is the interpolation weight calculation time, and c_{iws} is the interpolation weight sum. Note that the walk time is an average assessment over many interpolations. Any specific walk may encompass a wide range of triangle traversals, including zero traversals if the beginning triangle contained the interpolation point, or very many traversals if the interpolation point lies far away from the initial triangle. The specific triangle-walk time is directly proportional to the number of triangles traversed. Since this varies each time we interpolate we use an average assessment to determine the interpolation performance. Understanding that this is not the specific time per interpolation is crucial later when we describe the benefit derived from the operational nature of the locator relative to our interpolation problem.

The interpolation weight calculation is a locally constant time process. On average approximately 4 to 7 nodes will participate in any arbitrary interpolation. The interpolation weight sum is simply the sum of the surface values assigned to the participating nodes times the weight determined for each node. Since the participating node count is effectively constant the weight sum is also.

To demonstrate the rough performance of the natural-neighbor tessellation interpolator we have constructed a synthetic data set of three separate tessellations with 1000, 3100, and 10000 nodes, respectively. We have defined 31 separate surfaces (station phase pairs of travel time corrections) on to each tessellation. With these tessellations we shall interpolate a random set of interpolation locations to assess the performance of the natural-neighbor interpolator.

The interpolations were performed for over 22,000 randomly generated locations in the region of ground-truth influence within the synthetic data set. The same set of 22,000 locations was used in all interpolations. Each surface was interpolated at the 22,000 locations over 6 separate independent executions of the test software. The six sets of results were then averaged to produce a single average value for the time spent in the triangle-walk, in the triangle-walk and node weight calculation, and to perform the entire end-to-end interpolation yielding the final interpolation result. From these pieces of data we can find the time to perform the weight calculation by subtracting the walk time from the walk + weight calculation. Similarly, the weight-sum calculation can be found by subtracting the walk + weight time from the total interpolation time. The total interpolation time, the weight-sum calculation time, the weight calculation time, and the walk time were then plotted as a function of the three evaluated node densities. Also, the proportionality constant for the walk time was found by dividing the walk time by the square root of the total node count. Table 1 and Figure 4 below show the results.

Table 1. Interpolation Results

Avg. Node Count	Load & Initialization	Triangle Walk	Walk + Weight	Total Interpolation Time	Weight Calculation	Weight Sum Calculation	c_{tw}
977.71	0.014317	6.96E-06	2.83E-05	2.97E-05	2.14E-05	1.36E-06	2.22E-07
3134.45	0.043051	1.31E-05	3.51E-05	3.64E-05	2.2E-05	1.25E-06	2.35E-07
10114.5	0.149752	2.42E-05	4.68E-05	4.81E-05	2.26E-05	1.35E-06	2.41E-07

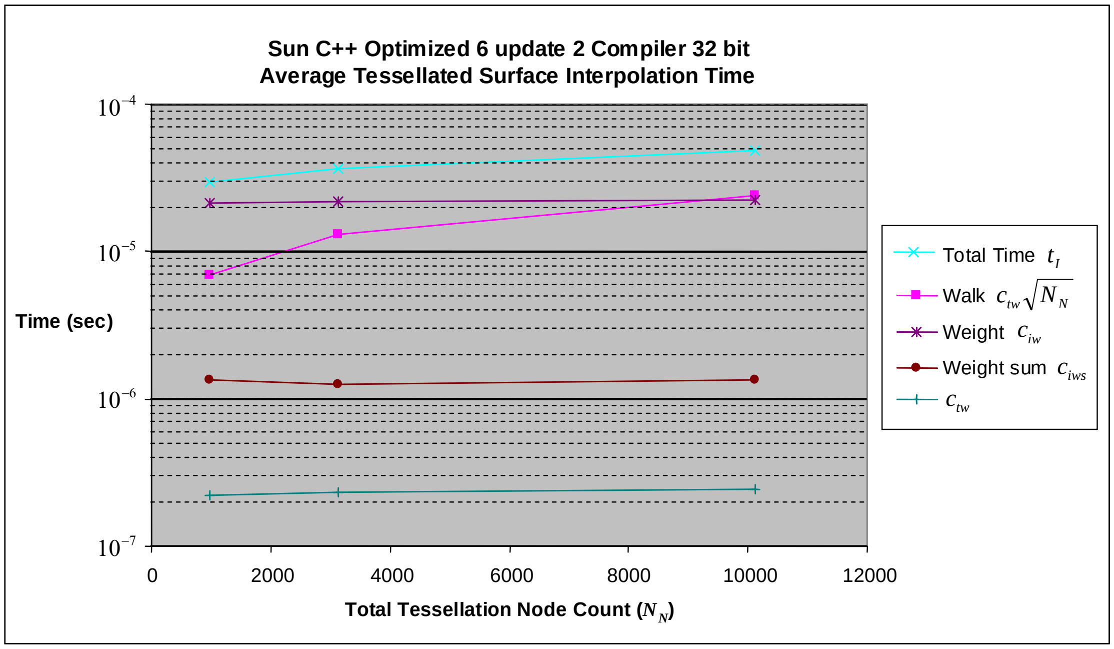

Figure 4. Natural-neighbor interpolation component breakdown illustrating the walk, weight, and weight-sum calculation times.

The load times were not displayed in the figure but average about 1.5 seconds per 10000 nodes loaded.

We can use the data developed from Figure 4 above to show the interpolation time over a broader range of nodes. This can be compared to the times determined for both base model evaluation and for the Kriging interpolation. First we shall describe how the base model interpolation times were defined and then we shall examine the Kriging times.

The base model interpolations were performed for over 16,000 randomly generated locations in two regions for the distance and depth dependent base models. The first region's set of data ranges from 0° to 180° distance and 0 to 400 km depth, covering teleseismic ranges, while the second region ranged from 0° to 20° distance and 0 to 5 km depth, to represent regional evaluation. The same two sets of 16,000 locations, one for both regions, were used in all interpolations. Each base model was interpolated at the 16,000 locations, in both regions, over 6 separate independent executions of the test software. The six sets of results were then averaged to produce a single average set of assessments. The minimum, maximum, and average for all 237 base models (IASPEI91 and ak135 evaluated for all available phases) were determined. The IASPEI91 P phase was also reported considering its general importance in overall location calculations. These were then consolidated into a single minimum, maximum, average, and IASPEI91 P phase evaluation time. The results included an average base model evaluation time of 6.81 $\times 10^{-5}$ sec within a range of 1.18×10^{-5} to 2.03×10^{-4} sec. The average IASPEI91 P phase evaluation was 2.01×10^{-5} sec. The wide range is caused by the different interpolation scenarios within the base model algorithm. Some base models have coarser tables yielding slightly faster results. Some have large regions that are undefined requiring extrapolation using the rational spline approximation, which can be very slow to evaluate. The end result is a factor of approximately 20 from the minimum to maximum base model evaluation times.

The Kriging interpolations were performed for the same set of over 22,000 randomly generated locations used by the tessellated surface tests in Figure 4. The Kriged surfaces were interpolated for all 31 stations (P phase only) and averaged together. The interpolation test was performed in 6 separate independent executions and the subsequent results were then averaged to give a single value of 9.09×10^{-4}sec. We know that this time is dominated by the cube of the number of ground-truths used to define the region, which for these Kriged surfaces, was approximately 100. However, the effective number of ground-truths used per interpolation was recorded for purposes of delineating these performance tests and was found to be on average 61 ground-truths per evaluation of the 22,000 interpolation locations. Using the cubic performance requirement for the Kriging solution we can determine that an effective ground-truth count of less than 17 would be required to enable Kriging with a performance equivalent to the IASPEI91 P phase base model evaluation.

Figure 5 illustrates the time to interpolate a single tessellated surface as a function of the tessellation node count. The base model and Kriging results are also shown for comparison.

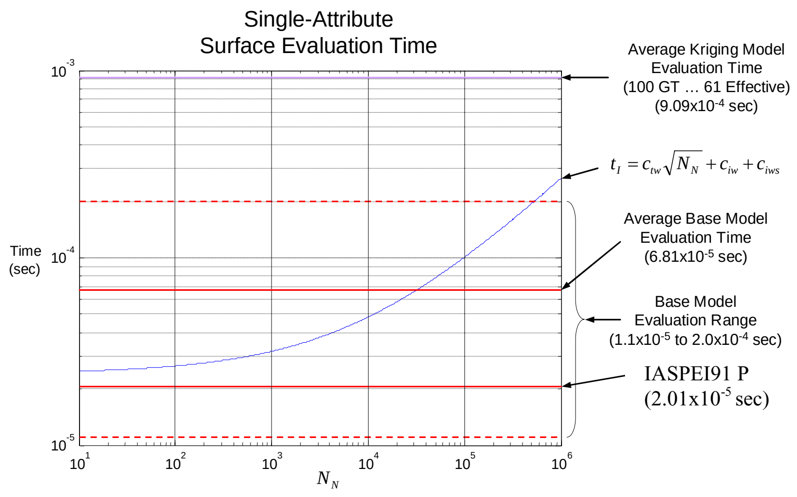

Figure 5. Single-attribute surface evaluation time given as a function of average node count. Base model and Kriged evaluations are also shown for comparison.

The fact that a tessellated surface can map more than a single attribute allows us to ask the question whether having one tessellation with many or all surface attributes assigned might not be a more efficient solution than many tessellations with only one attribute assigned per tessellation. Is there sufficient operational separation between steps that may allow us to take advantage of the fact that the first two costly steps are purely spatial (triangle search and weight calculation) and only the highly efficient last step (weight-attribute sum) requires knowledge of the attribute information?

We can answer this question by examining the previous expression for the run-time behavior of the natural-neighbor interpolation method. Let the number of interpolation attributes be denoted by N_{AI}. Since the containing triangle search and subsequent weight calculation only needs to be performed once in order to evaluate the resulting interpolation for all attributes the previous expression for the run-time behavior of the natural-neighbor interpolation can be rewritten as

$$t_I = c_{tw}\sqrt{N_N} + c_{iw} + N_{AI}c_{iws} \,. \tag{2}$$

Notice that only the last constant is affected by the extra calculation for the additional surface attributes. The value for that constant is slightly more than a micro-second ($1.3e^{-6}$). If we have, for example, a tessellation with 100 surface attributes mapped to its nodes, then from Figure 5 we see that the total solution time for surface with 100,000 nodes will climb from $1.0e^{-4}$ seconds to approximately $2.0e^{-4}$ seconds. That's only a factor of two for an additional 99 surface attribute evaluations. An equivalent number of IASPEI91 P phase base model evaluations will jump from 2.0×10^{-5} seconds to approximately 2.0×10^{-3} seconds. This result suggests that huge time savings can be recovered by saving multiple surface attributes onto the nodes of individual tessellations.

The previous discussion neglected the significant impact that many tessellations with a few attributes, or a few tessellations each with many attributes, can have on storage requirements and subsequent data load times. In the next section we shall examine the impact of both load time and run time for a more realistic set of tessellations that have large numbers of nodes with multiple surface attributes assigned to each. These comparisons will be accomplished using the Sandia National Laboratories' event location tool LocOO (Ballard, 2003), which will incorporate the effects of realistic software overhead into the results shown above.

Locator (LocOO) Specific Performance Comparison

To demonstrate the effect of the timing difference between using tessellations versus Kriging in an actual event location application, we set up a test using a synthetic data set developed for general purpose testing of path correction surfaces. The data set consists of 126 synthetic seismic events with 31 P wave travel time observations associated with each event. There are path corrections available for each of the 31 observations.

For this test, we compared the time it took to 1) load the model information from the file system into memory, 2) locate a set of events and 3) unload the model information from memory, using 5 different run conditions: 1) base model information only, 2) path corrections calculated using Kriging, 3) path corrections calculated based on a 24,000 node tessellation, 4) a 68,000 node tessellation and 5) a 226,000 node tessellation. Note that the time required to perform all data input/output functions is excluded from this analysis.

In the first run of the test, we located only one of the events in the data set. The results are illustrated in Figure 6.

In this scenario, the amount of time spent actually computing the event location was small compared to the time required to load and unload the model. A comparison of the base model and Kriged path corrections with the tessellated interpolation for the 226K tessellation shows that 90%, 50% and 99.8% of the time was spent loading and unloading the model information, respectively. The total time required by the 226K tessellation was 25 times greater than the total time required by the Kriging-only solution.

In the second run of the test, 126 events were located (see Figure 7). In this case, the models were only loaded/unloaded once but 126 events were located using the model while it was in memory. In this scenario, tessellation performed much better than Kriging only case because it took approximately 10 times longer to compute each location using Kriging as compared to any of the other models.

The bottom line is that if the location software is operated in a mode where it loads a model, computes a location and then unloads the model, Kriging may be a better option unless only small tessellations (a few thousand nodes) are involved. On the other hand, if the location software can be operated in a mode where the model is loaded once and many locations are computed, then a tessellation interpolation can significantly outperform Kriging.

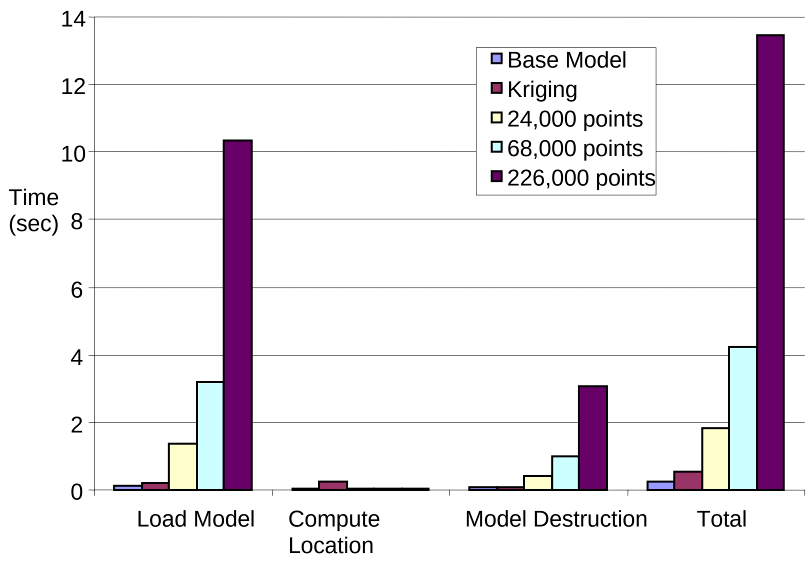

Figure 6. Time required performing various subtasks of a seismic event location calculation when locating single seismic events.

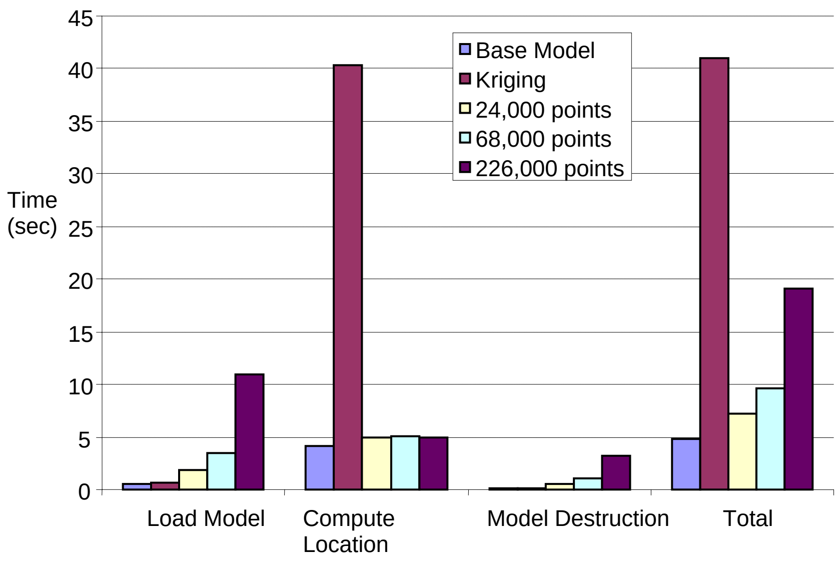

Figure 7. Time required performing various subtasks of a seismic event location calculation when locating 126 seismic events.

CONCLUSIONS AND RECOMMENDATIONS

In summary, the use of an optimal spherical geometry tessellation for representing Kriged path corrections surfaces can far surpass Kriging interpolation performance, and for multiple surface attributes assigned on the same tessellation, the method can be as fast as or faster than standard base-model evaluations. Further, this is true for tessellations with as many as 250,000 nodes and more than 60 surface attributes mapped to the tessellation. The primary problem with the tessellation approach concerns data storage and data load times which can be excessive. As long as sufficient storage is available and the operational use of the software is of the form of load-once and interpolate-many, then this method can be substantially faster than typical Kriging approaches.

ACKNOWLEDGEMENTS

We would like to thank Mark Harris and David Gallegos for thoughtful reviews. Their comments have improved the manuscript.

REFERENCES

Ballard, S. (2003). Seismic event location: Dealing with multi-dimensional uncertainty, model non-linearity and local minima, in *Proceedings from the 25th Seismic Research Review—Nuclear Explosion Monitoring: Building the Knowledge Base,* LA-UR-03-6029, Vol. 1, pp. 193-202.

Delaunay, B.N., (1934), Sur la Sphere Vide, Bull. Acad. Science USSR VII: Class. Sci. Math., 793-800.

Kennett, B.L.N., and E.R. Engdahl (1991). Traveltimes for global earthquake location and phase identification. *Geophys. J. Int*. 105: (2), 429-465.

Kennett, B.L.N., E.R. Engdahl, and R. Buland (1995). Constraints on seismic velocities in the earth from travel time. *Geophys. J. Int*. 122, 108-124.

Lawson, C.L. (1977). Software for C1 surface interpolation, in Mathematical Software III, J. Rice (ed). Academic Press, New York.

Levin, D. (1998), The Approximation Power Of Moving Least-Squares, *Mathematics Of Computation* 67: 224, 1517-1531, October 1998.

Sambridge, M., J. Braun, and H. McQueen (1995). Geophysical parameterization and interpolation of irregular data using natural neighbors, *Geophys. J. Int*., 122: (3), 837-857.

Schultz, C., S. Myers, J. Hipp, and C. Young (1998), Nonstationary Bayesian Kriging: Application of Spatial Corrections to Improve Seismic Detection, Location and Identification, *Bull. Seism. Soc. Am*., 1275-1288.

Watson, D. F., (1992). *Contouring: A Guide to the Analysis and Display of Spatial Data*, Pergamon, ISBN 0-08-040286-0, 321 pp.